**Amritsar College of Engineering & Technology,**

**Amritsar**

**(NBA Accredited)**

Six Months Industrial Training Final Report

On

**University Management System**

Submitted in the Partial fulfillment of the requirement for the Award of Degree

of

**Bachelors of Technology**

In

**COMPUTER SCIENCE & ENGINEERING**

Batch

(2010-2014)



**SubmittedTo*:* SubmittedBy*:***

Col.(Retd.) Gurmukh Singh Sagar Dua

HOD (CSE) U.Roll N:100030303531

Rana Jaskaran Singh

U.Roll N:100030303520

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

**PROJECT REPORT**

ON

**University Management System**



**ACKNOWLEDGEMENT**

At the outset, our primary thanks goes to **“Ms. Sampada Jain”**, who has guided and helped us at every stage of our project “**University Management System”.** We want to express our sincere gratitude towards those who have guided and helped us to complete this project, which is a culmination of the **Degree of B.Tech** (Bachelors of Technology) to be awarded by **Punjab Technical University, Jalandhar.**

We have been encouraged and given tips whenever we approached them with our problems. We were provided with latest software to make the presentation sleeker and more appealing. We owe them a great debt for their support and willingness to take up the tasks that we would not have been able to do without their help.

We also express our profound sense of gratitude to **Col.(Retd.) Gurmukh Singh**, HOD CSE Deptt. for providing various facilities needed for successful completion of our project & time to time enunciating concept related to project.

We are very thankful to our parents for their encouragement and moral support.

**Sagar Dua**

**Rana Jaskaran Singh**

**TABLE OF CONTENTS**

**S.No Contents Page No.**

1. ORGANIZATION PROFILE 2

2. INTRODUCTION 6

3. PROJECT DETAILS 9

4. PROJECT WORK 17

5. FEASIBILITY STUDY 18

6. FRONT END & BACK END 21

* Introduction to ASP.NET 28
* Introduction to C# 32
* Introduction to SQL SERVER 2008 34

7. SYSTEM ANALYSIS 38

8. SYSTEM DESIGN 42

* Data Flow Diagrams 43
* Use-Case Diagrams 55
* Table Description 62

9. SCREENSHOTS 63

10. SYSTEM TESTING 93

11. FUTURE ENHANCEMENT 111

12. BIBLIOGRAPHY 114

**ORGANIZATION PROFILE**



**BEBO TECHNICAL EDUCATION SERVICES**

BTES

SCO 58-59, Sector 34-A

Chandigarh, India 160034

Ph. 91-172-5086508-

Website: www.btes.co.in

**About Bebo Technologies Pvt. Ltd.**

Bebo Technologies Pvt. Ltd. launched their first ever finishing school in Chandigarh, the tricity region.

Bebo Technologies Pvt. Ltd., an ISO 9001:2008 compliant company, has now emerged into providing practical training on various software technologies under the banner of bebo Technical Education Services (BTES).

BTES focuses on strategic and practical approach in providing training to students and working professionals. Under the close guidance and supervision of experienced professionals, students are trained on various technical aspects to tackle industry challenges faced regularly in their professional life.

BTES cover everything necessary to build a strong professional foundation. Through courses, which are developed by industry experts, btes bring their learning and experience to students so that they excel in professional life. This brings them at par with working professionals and thus, meets the industry expectations.

BTES goal remains to provide practical knowledge with strong fundamentals so that student’s career is secured.

**Quality Policy**

BTES, being the gateway to professional education ensure industry ready and day one productive resources with comprehensive, strategic and practical hands on training.

Ensure that training is of the highest quality, aligned with the career objectives and expectations of our students, and best value for their time and money.

Serve the community by providing access to campus wide activities & career resources.

BTES also comply with the requirements and continually improve the effectiveness of **ISO 9001:2008** standards.

**Quality Objectives**

The management and employees of BTES Pvt. Ltd. Chandigarh are fully committed to adopt and continuous compliance of international quality management system.

To provide students with latest skill sets and courseware designed by industry experts commensurate with the expectations of the Industry.

To provide satisfactory customer service through continuous improvement of the existing product quality and adding new products.

Optimize training as per student's technical and soft skills need while meeting their career objectives.

Ensure that the training is a perfect blend of industry examples and real time case studies.

bebo Technical Education Services Pvt. Ltd. is backed by industry experts to solve real-time technical challenges.

**Why @BTES**

* Courses developed by industry experts.
* Batch size kept between 10 to 15 students to devote individual attention.
* Trainers are the “Specialists” in their respective areas to impart competent and quality training.
* After class easy availability of trainer to solve queries.
* State of the art labs with library facility.
* Real time project based training.
* Class room sessions with blend of industry examples and real time case studies.
* Seminars on various topics like soft skills, interview preparation and more.
* Interaction with HR and industry experts.
* Placement assistance to eligible students on successfully completing the training.

**Domain in Which We Work**

* Microsoft Visual Studio 2010
* SQL SERVER 2008
* AJAX
* HTML/DHTML
* CSS

Our core strengths are in organized code (so that it can be modified by anyone at any given time with least amount of efforts). All the above mentioned technologies are used for developing various systems in our company. Our programmers are some of the best ones in their fields. All the employees of the company are well educated and hold degrees in B-Tech and MCA. The strength of the company is the team work of the members of our company.

**INTRODUCTION**

# 2.1 PROJECT DESCRIPTION

## Project Definition

University Management System

## What is University Management System?

UNIVERSITY MANAGEMENT SYSTEM [UMS] deals with the maintenance of university, colleges , faculties and students information within the university. This project of UMS involved the automation of student information 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.

**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.

* University Administrator
* College Administrator
* User (Students / Faculties)

**University Administrator** has the functionality of registering new colleges and courses.

**College Administrator** has the rights of creating department, allocating courses to departments, creating faculties, students and allocating subjects to faculties, and modifications in the data entered by the user can also be done by the college administrator.

**User** of this may be faculty or students. Faculty has the facility of entering the marks and attendance of the students. Students can check their marks and attendance but there is no chance of modifications.

**PROJECT DETAILS**

**3.1 ABOUT UNIVERSITY MANAGEMENT SYSTEM**

**3.1.1** **Objectives of Project**

The main objectives of the project are to make a web application that can make functioning and maintenance of a university easy :-

* Providing an interface for the administrator which enable them to have access to colleges , facultities and students.
* Making a web application which provides multiple facilities on one interface.
* Maintaining data maintenance list in cccount.
* Providing better performance and access to details of each student and faculty member.
* Making the data accessible at any part of the country.
* Easy to search.
* Easy to retrieve records.
* Easy to access data regarding project.

**3.1.2** **Theme**

UMS is a web application that is designed to give the online facility to maintain the records of a university. This project is designed using ASP.NET and HTML Programming languages, Photoshop for the front end and the back end has been developed using SQL SERVER 2008 R2.

The operating system used is Windows7. The software tools required by this project are VISUAL STUDIO 2010, SQL SERVER 2008 R2, HTML.

The project requires the knowledge of the following subjects:

Software Language: ASP.NET

Database: SQL SERVER 2008 R2.

**For This Application, we will provide following users:**

1. **Admin Module**

The main functions of admin are as mentioned below.

1. **Login:** To Login the Admin Module
2. **Add College:** To add the college under the university.
3. **Add Student:** To add the student studying in the university.
4. **Add Faculty:** To add the faculty working in the university.
5. **Edit Profile:** To edit his/her own settings.
6. **Upload Result:** To upload result of the students.

In this module the after logging in, the admin is redirected to the admin homepage where the admin can have access to all the above web forms.

The admin has full powers to access anything in the application.

1. **College Admin Module**

The main functions of employers are as mentioned below:

1. **Login:** To Login into the College Admin Module.
2. **Add Student:** To add student under particular college.
3. **Add Faculty:** To add faculty under particular college.
4. **Add Batch:** To add batch in the college.
5. **Add Subject:** To add subjects taught in the college.
6. **Add Department:** To add new departments in the college.
7. **Update Information:** To update information regarding the college.
8. **Add User:** To give rights as admin.
9. **Faculty Module**

The main functions of the freelancers are as mentioned below:

1. **Login:** To Login the Faculty Module.
2. **Upload Marks:** To upload marks of the respective student.
3. **Upload Attendance:** To upload attendance of the respective students.
4. **Upload Assignment:** To upload assignments related to their respective subjects.
5. **Edit Profile:** To edit his/her own profile.
6. **Change Password:** To change password for login.
7. **Student Module**

The main functions of this type of users are as mentioned below:

1. **Login:** To Login the Student Module**.**
2. **Download Notes:** To download the notes uploaded by the concerned faculty.
3. **Download Assignment:** To download assignments posted by the respective faculty members.
4. **Upload Assignments:** To upload the completed assignments.
5. **Edit Profile:** To edit his/her own profile.
6. **Change Password:** To change password for login.
7. **View Attendance:** To view attendance according to lectures and praticals.
8. **View Result:** To view result subject-wise.

**2.1.4 Necessity:**

As the application developed in ASP.NET has good interface and design, so we decided to use ASP.NET TECHNOLOGIES and SQL SERVER in the implementation of our project. So the main necessity of the project was to make a application which is secure and useful. Hence, I decided to make the project and this is the part of the necessity of website development.

**3.2 HARDWARE & SOFTWARE REQUIREMENTS**

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
| **Operating Systems** | : | Microsoft Windows XP Professional With SP3 |
|  |  |  |
| **Hardware Requirement** | : | Pentium 90 MHZ or Faster and 1GB Ram (Client)  Pentium 133 MHZ or Faster and 1GB Ram (Server) |
|  |  |  |
| **Software Requirement** | : | Microsoft Visual Studio 2010 |
| **Front End** | : | Microsoft Visual Studio Asp.Net 4.0 |
| **Back End** | : | Microsoft SQL Server 2008 R2 |
| **Others** | : | * Microsoft Asp.Net * .Net Framework |
|  |  |  |
|  |  |  |

# 3.3 PURPOSE

**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.

* University Administrator
* College Administrator
* User (Students / Faculties)

**University Administrator** has the functionality of registering new colleges and courses.

**College Administrator** has the rights of creating department, allocating courses to departments, creating faculties, students and allocating subjects to faculties, and modifications in the data entered by the user can also be done by the college administrator.

**User** of this may be faculty or students. Faculty has the facility of entering the marks and attendance of the students. Students can check their marks and attendance but there is no chance of modifications.

# 3.4 TECHNOLOGY AND LITERATURE REVIEW

|  |  |
| --- | --- |
| **Operating System** | Window-XP sp3 |
| **Technology** | .Net 2010 with 4.0 architecture |
| **Language** | Asp.net 4.0 |
| **Database** | MS-SQL 2008 server r2 |

**PROJECT WORK**

**FEASIBILITY STUDY**

# 5.1 FEASIBILITY STUDY

Feasibility study is a process to check possibilities of system development. It is a method to check various different requirements and availability of financial & technical resources.

Before starting the process various parameters must be checked like:

* Estimated finance is there or not?
* The man power to operate the system is there or not?
* The man power is trained or not?

All the above conditions must be satisfied to start the project. This is why in depth analysis of feasibility is carried out.

There are four different ways feasibility can be tested:

1) Economical Feasibility

2) Technical Feasibility

3) Operational Feasibility

4) Behavioral Feasibility.

**Economical Feasibility:**

In economical feasibility, analysis of the cost of the system is carried out. The system should be only developed if it is going to give returned the current manual system user can get the price only by purchasing the newspapers. In addition if he/she wants to see archives of particular equity then he has to refer to all the old newspapers. For research reports he has to buy another magazine. So Instead of buying no of magazines user has to just go online and with a single click he can get whatever information he wants. So our project of online share news passes the test of economical feasibility.

**Technical Feasibility:**

It is basically used to see existing computer, hardware and software etc, weather it is sufficient or additional equipments are required? Minimum System Requirement is such that it can be affordable by of the user who is having computer. All the user requires is compatible browser and .net framework installed so our system is fully technical feasible.

**Operational Feasibility:**

Once the system is designed there must be trained and expert operator. If there are not trained they should given training according to the needs of the system.

From the user’s perspective our system fully operational feasible as it just requires some knowledge of computer. Operators only need add daily prices of various equities and there are enough validations available so operator does not require any special technical knowledge. So our system also passes the test of operational feasibility.

**Behavioral feasibility:**

Computers are known to facilitate change and people are usually resistant to change. In determining the behavioural feasibility, we make an estimate of how strong a reaction will be the user staff make towards the development of the computerized system and try to keep the user response positive.

In case of railway reservation there is a strong social aspect. In the organization there is always a situation where there a need of various type of information regarding the trains, so the staff is openly ready to accept the new system without any hesitation.

**FRONT END &**

**BACK END**

**6. FRONT END & BACK END OF THE SYSTEM**

**FRONT END: ASP.NET4.0**

**BACK END: SQL SERVER 2008 R2**

**6.1 The .Net Framework :**

A frame work is commonly though of as a set of class libraries that aid in the development of applications. The .net framework is more than just a set of classes. The .net framework is targeted by compliers using a wide variety of applications. Including everything from small components that run on handheld devices to large Microsoft ASP.NET application that span web farms, where multiple web serves act together to improve the performance fault tolerance of a web site. The .NET framework is responsible for providing a basic platform that these applications can share. This basic platform includes a runtimes set of services that oversee the execution of applications. A key responsibility of the runtime is to manage execution so that software written by different programming languages uses classes and other types safely.

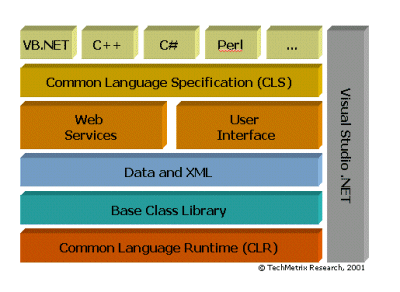
**6.1.1 Microsoft .net framework architecture**

Microsoft's .NET Framework is comprised of two main components - the Common Language Runtime (CLR) and the .NET Framework class libraries. The CLR is the real foundation of the .NET Framework. It is the execution engine for all .NET applications. Every target computer requires the CLR to successfully run a .NET application that uses the .NET Framework. The main features of CLR include:

* Automatic Memory Management
* Thread Management
* Code Compilation & Execution
* Code Verification
* High level of security
* Remoting
* Structured Exception Handling
* Interoperability between Managed and Unmanaged code.
* Integration with Microsoft Office System

All .NET applications are compiled into Intermediate Language code (MSIL). When executed on the CLR, MSIL is converted into native machine code specific to the operating platform. This process is done by a Just in Time (JIT) compiler. The code executed by the CLR is called as Managed Code. This code is type safe and thoroughly checked by the CLR before being deployed. The .NET runtime also provides a facility to incorporate existing COM components and DLL's into a .NET application. Code that is not controlled by the CLR is called Unmanaged Code.

The .NET Framework is further comprised of Common Type System (CTS) and Common Language Specification (CLS). The CTS defines the common data types used by .NET programming languages. The CTS tells you how to represent characters and numbers in a program. The CLS represents the guidelines defined by for the .NET Framework. These specifications are normally used by the compiler developers and are available for all languages, which target the .NET Framework.



**.Net architecture**

**6.1.2 Common Language Specification**

To fully interact with other objects regardless of the language they were implemented in, objects must expose to callers only those features that are common to all the languages they must interoperate with. For this reason, the Common Language Specification (CLS), which is a set of basic language features needed by many applications, has been defined. The CLS rules define a subset of the Common Type System; that is, all the rules that apply to the common type system apply to the CLS, except where stricter rules are defined in the CLS. The CLS helps enhance and ensure language interoperability by defining a set of features that developer can rely on to be available in a wide variety of languages. The CLS also establishes requirements for CLS compliance; these help you determine whether your managed code conforms to the CLS and to what extent a given tool supports the development of managed code that uses CLS features.

If your component uses only CLS features in the API that it exposes to other code (including derived classes), the component is guaranteed to be accessible from any programming language that supports the CLS. Components that adhere to the CLS rules and use only the features included in the CLS are said to be CLS-compliant components.

The CLS was designed to be large enough to include the language constructs that are commonly needed by developers, yet small enough that most languages are able to support it. In addition, any language constructs that makes it impossible to rapidly verify the type safety of code was excluded from the CLS so that all CLS-compliant languages can produce verifiable code if they choose to do so.

**6.1.3 Common Language Runtime**

The Common Language Runtime (CLR) is the virtual machine component of Microsoft's .NET initiative. It is Microsoft's implementation of the Common Language Infrastructure (CLI) standard, which defines an execution environment for program code. The CLR runs a form of byte code called the Microsoft Intermediate Language (MSIL), Microsoft's implementation of the Common Intermediate Language.

Developers using the CLR write code in a high level language such as C# or VB.Net. At compile-time, a .NET compiler converts such code into MSIL (Microsoft Intermediate Language) code. At runtime, the CLR's just-in-time compiler (JIT compiler) converts the MSIL code into code native to the operating system. Alternatively, the MSIL code can be compiled to native code in a separate step prior to runtime. This speeds up all later runs of the software as the MSIL-to-native compilation is no longer necessary.

Although some other implementations of the Common Language Infrastructure run on non-Windows operating systems, the CLR runs on Microsoft Windows operating systems.

The virtual machine aspect of the CLR allows programmers to ignore many details of the specific CPU that will execute the program. The CLR also provides other important services, including the following:

• Memory management

• Thread management

• Exception handling

• Garbage collection

• Security

**6.2 Introduction to ASP.NET**

Although so Microsoft Visual Basic.NET is a powerful but simple language aimed primarily at developers creating web applications for the Microsoft .NET platform. It inherits many of the best features of C++ and Microsoft Visual Basic, but with some of the inconsistencies and anachronisms removed, resulting in cleaner and logical language. VB also contains a variety of useful new innovations that accelerate application development, especially when used in conjunction with Microsoft Visual Studio .NET.

The Common Language Runtime provides the services that are needed for executing any application that’s developed with one of the .NET languages. This is possible because all of the .NET languages compile to a common Intermediate Language. The CLR also provides the common type system that defines those data types that are used by all the .Net languages. That way, you can use same data types regardless of what .NET language you’re using to develop your application implementations.

**ASP.NET:**  Microsoft, realizing that ASP does posses some significant shortcomings, developed ASP.net. ASP.net is a set of components that provide developers with a framework with which to implement complex functionality. Two of the major improvements of ASP.net over traditional ASP are scalability and availability. ASP.net is scalable in that it provides state services that can be utilized to manage session variables across multiple web services in a server farm. Additionally, ASP.net possesses a high performance process model that can detect application failures and recover from them. We use the fundamentals of programming with VB using Visual Studio .NET and .NET framework.

The project is the starting point for authoring applications, components & services in Visual Studio.NET 2010. It eats as a container that manages your source code, data connections & references. A project is organized as part of a solution, which can contain multiple projects that are independent of each other. C# project file has .asproj extension where as solution file has .sln extension.

In order to write code against an external component, your project must first contain a reference to it. A reference can be made to the following types of component.

(1) .NET class libraries or assemblies

(2) COM components

(3) Other class libraries of projects in the same solution

(4) XML web services

**Features of ASP.NET:**

(1) Component Infrastructure.

(2) Language Integration.

(3) Internet Interoperation.

(4) Simple Development.

(5) Simple Deployment.

(6) Reliability.

(7) Security

**CHOICE OF LANGUAGE:**

The reason for choosing above languages and database system to develop this project is:

1. **ASP.NET is integrated with the .NET Framework**

The .NET Framework is divided into an almost painstaking collection of functional parts, with a staggering total of more than 10,000 *types* (the .NET term for classes, structures, interfaces, and other core programming ingredients). The massive collection of functionality that the .NET Framework provides is organized in a way that traditional Windows programmers will see as a happy improvement. Each one of the thousands of classes in the .NET Framework is grouped into a logical, hierarchical container called a *namespace*.

Different namespaces provide different features. Taken together, the .NET namespaces offer functionality for nearly every aspect of distributed development from message queuing to security. This massive toolkit is called the *class library*.

2. **ASP.NET Is Compiled, Not Interpreted**

One of the major reasons for performance degradation in classic ASP pages is its use of interpreted script code. Every time an ASP page is executed, a scripting host on the web server needs to interpret the script code and translate it to lower-level machine code, line by line. ASP.NET applications are always compiled—in fact, it’s impossible to execute C# or Visual Basic code without it being compiled first.

3**. ASP.NET Is Multilanguage**

IL is a stepping stone for every managed application. (A *managed application* is any application that’s written for .NET and executes inside the managed environment of the CLR.) In a sense, IL is *the* language of .NET, and it’s the only language that the CLR recognizes.

**ASP.NET Is Hosted by the Common Language Runtime**

The most important aspect of the ASP.NET engine is that it runs inside the runtime environment of the CLR.

**6.3 Introduction to C#:**

C#, pronounced c sharp, is a programming language that can be used to give instructions to a computer. The instructions can be written from a text editor such as Notepad. Another way is to use a programming environment that is equipped with many tools that make it easy to work on projects, to create the necessary files, and to distribute a completed application.

**C#** (pronounced "see sharp") is a [multi-paradigm programming language](http://en.wikipedia.org/wiki/Multi-paradigm_programming_language) encompassing [imperative](http://en.wikipedia.org/wiki/Imperative_programming), [declarative](http://en.wikipedia.org/wiki/Declarative_programming), [functional](http://en.wikipedia.org/wiki/Functional_programming), [generic](http://en.wikipedia.org/wiki/Generic_programming), [object-oriented](http://en.wikipedia.org/wiki/Object-oriented_programming) ([class-based](http://en.wikipedia.org/wiki/Class_(computer_science))), and [component-oriented](http://en.wikipedia.org/wiki/Component-based_software_engineering) programming disciplines. C# is intended to be a simple, modern, general-purpose, object-oriented programming language.

**6.3.1 Features of C#:**

* There are no global variables or functions. All methods and members must be declared within classes. Static members of public classes can substitute for global variables and functions.
* Local variables cannot shadow variables of the enclosing block, unlike C and C++. [Variable shadowing](http://en.wikipedia.org/wiki/Variable_shadowing) is often considered confusing by C++ texts.
* [Multiple inheritance](http://en.wikipedia.org/wiki/Multiple_inheritance) is not supported, although a class can implement any number of interfaces. This was a design decision by the language's lead architect to avoid complication and simplify architectural requirements throughout CLI.
* [Enumeration](http://en.wikipedia.org/wiki/Enumerated_type) members are placed in their own [scope](http://en.wikipedia.org/wiki/Scope_(programming)).
* C# currently (as of version 4.0) has 77 [reserved words](http://en.wikipedia.org/wiki/Reserved_word).
* [Checked exceptions](http://en.wikipedia.org/wiki/Checked_exceptions) are not present in C# (in contrast to Java). This has been a conscious decision based on the issues of scalability and version ability.

**6.3.2 Advantages of C#:**

1. While no programming language lets programmers write entirely bug-free programs, C# goes a long way in comparison with C and C++.
2. With Windows Form and later on WPF (especially with the aid of Expression Interactive Designer), C# is great for Rapid Application Development (RAD).
3. C# is safer to run.  Since C# program is compiled into an intermediate language, the OS can always check it to see that no malicious code is about.
4. C# combines the old and the new in an almost perfect balance.  C# duplicates much of the concise syntax of C and also adds modern, object-oriented features while retaining very little unnecessary baggage from C.
5. Cost of maintenance for C# is definitely much lower than that of C++.  This is a positive side effect of C# helping programmers to write program that is as bug free as possible.
6. C# can make use of every feature available in WPF, making it one of the languages that can work perfectly with WPF.

**6.3.3 Disadvantages of C#:**

1. C# is slower to run.  This is somewhat taken care of when using WPF, although currently the launching of WPF application is still a bit slow.  However, after the program is launched, the animation effects are all very smooth.
2. C# is less flexible than C++.  C# depends greatly on .NET framework, anything that is not found in the .NET framework will be difficult to implement.

**6.4 Introduction to Microsoft SQL Server**

Microsoft SQL Server enhances the performance, reliability, and scalability provided by earlier releases of SQL Server by making the processes of developing applications, managing systems, and replicating data easier than ever.

All of data processing is involved with the operations of storing and retrieving data. A database, such as Microsoft SQL Server, is designed as the central repository for all the data of an organization. The crucial nature of data to any organization underlines the importance of the method used to store it and enable its later retrieval.

Microsoft SQL Server uses features similar to those found in other databases and some features that are unique. Most of these additional features are made possible by SQL Server’s tight integration with the Windows XP operating system. SQL Server contains the data storage options and the capability to store and process the same volume of data as a mainframe or minicomputer.

Like most mainframe or minicomputer databases, SQL Server is a Database that has seen an evolution from its introduction in the mid-1960s until today. Microsoft’s SQL Server is founded in the mature and powerful relational model, currently the preferred model for data storage and retrieval.

Unlike mainframe and minicomputer databases, a server database is accessed by users-- called clients--from other computer systems rather than from input/output devices, such as terminals. Mechanisms must be in place for SQL Server to solve problems that arise from the access of data from perhaps Hundreds of computer systems, each of which can process portions of the database independently from the data on the server. Within the framework of a client/server database, a server database also requires integration with communication components of the server in order to enable connections with client systems.

**Microsoft SQL Server 2008 R2:**

SQL server also contains many of the front-end tools of PC databases that traditionally haven’t been available as part of either mainframe or minicomputer databases. In addition to using a dialect of Structured Query Language (SQL), GUI applications can be used fro the storage, retrieval, and administration of the database.

SQL Server 2008 R2, like its predecessors, is more than a database engine. It is a   
collection of components that you can implement either separately or as a group   
to form a scalable data platform. In broad terms, this data platform consists of   
two types of components—those that help you manage data and those that help   
you deliver business intelligence (BI).

“Database Administration,” is written with the DBA in mind and introduces   
readers to the numerous innovations in SQL Server 2008 R2. “SQL   
Server 2008 R2 Editions and Enhancements,” discusses the key enhancements,   
what’s new in the different editions of SQL Server 2008 R2, and the benefits of   
running SQL Server 2008 R2 on Windows Server 2008 R2. “Multi-   
Server Administration,” readers learn how centralized management capabilities   
are improved with the introduction of the SQL Server Utility Control Point. Step-by-   
step instructions show DBAs how to quickly designate a SQL Server instance as   
a Utility Control Point and enroll instances for centralized multi-server management.   
 “Data-Tier Applications,” focuses on how to streamline deployment   
and manage and upgrade database applications with the new data-tier application   
feature.“High Availability and Virtualization Enhancements,”   
covers high availability enhancements and includes step-by-step implementations   
for ensuring business continuity with SQL Server 2008 R2, Windows Server 2008   
R2, and Hyper-V Live Migration. “Consolidation and Monitoring,”   
a discussion on consolidation strategies teaches readers how to improve   
resource optimization. This chapter also explains how to use the new dashboard   
and viewpoints to gain insight into application and database utilization, and it also   
covers how to use capacity policy violations to help identify consolidation opportunities,   
maximize investments, and ultimately maintain healthier systems.

“Business Intelligence Development,” readers discover components   
new to the SQL Server data platform, as well as significant enhancements to the   
reporting component. “Scalable Data Warehousing,” introduces the   
data warehouse appliance known as SQL Server 2008 R2 Parallel Data Warehouse   
by explaining its architecture, reviewing data layout strategies for optimal query   
performance, and describing the integration points with SQL Server BI components.   
 “Master Data Services,” readers learn about master data   
management concepts and the new Master Data Services component.

**SYSTEM ANALYSIS**

**SYSTEM ANALYSIS**

A study of system concept has three basic implications. A system must be designed to achieve a predetermined objective. Interrelation and interdependence must exit among the components. The objective of the organization as a whole has higher priority that the objective of its sub system. In the present study a system is an integrated collection of data files. The combination of all these programs and database made this system. The process of designing a system is not done in one phase. It consists of many phases that can be seen comprising of four phases:

1. SYSTEM STUDY AND PROBLEM DEFINATION
2. SYSTEM ANALYSIS
3. SYSTEM DECISION AND PROGRAMMING
4. IMPLEMENTATION AND DOCUMENTATION

It consists of the following steps: -

* + Feasibility study.
  + Information gathering.
  + Making algorithms and flowcharts.

**System study** involves the investigation of overall problems to be solved as in our system study, which is relating to database handling. It is the main problem to be investigated during system study. It also involves the documents to the handled and the overall flow and manipulation of data. The system analysis involves the fact finding of problem, information gathering, and identification of constrains, specific objectives, descriptions of outputs required for the concerned problems.

**INFORMATION GATHERING**

After defining the problem, the next step in the system analysis is the information gathering is an art as well as a science. Information gathering is neither easy nor routine. Much preparation, experience and training is required before one determines where to go for the information or what tool to user. The first requirement is to figure out what information to gather, the approach, the manner in which the information is gathered require a person with sensitivity, skill and commonsense of the following:

* 1. Purpose of gathering information
  2. What and when to gather
  3. Source of information
  4. What Channel to use in securing the information.

**PURPOSE OF GATHERING INFORMATION**

There are thre the main purpose of gathering information; we get familiar with the organization and present the system through available procedure manuals, official documents, and by knowing the flow of the information.

* Knowing how information is to be accessed from the system and subsequently making the access system simple and easy to use.
* Getting to know how and when old data is to be disposed.

**KIND OF INFORMATION NEEDED:** Before one determines where go and what tool to user, first requirement is to figure out what information to gather. The basic information required is how organization of personal or official contact as well as appointment takes place.

**SOURCES OF INFORMATION:** Information is gathered from two main sources namely personal and written document within the organization environment.

**INFORMATION GATHERING TOOLS:** There are no general rules for using the tool for the information gathering an important rule is that information is acquired accurately, methodically, under right conditions and with minimum interruption to the user and the staff. Following are the common information gathering tools :-

* Review procedure manuals.
* Onsite observation.
* Interviews.

**STRUCTURED ANALYSIS**

**Structured analysis** is the set of techniques and graphical tools that allow the analyst to develop a new kind of system specification that are easily understandable to the user. It is a way to focus on function rather than physical implementation. The traditional approach focuses on the cost/benefit analysis, feasibility, project management, hardware and software section and personal considerations in contrast. Structured analysis considers new goals and structured tools for analysis. There are different tools for structured system analysis some of them are as follows:

1. Dataflow diagrams

2. Data dictionary

3. Structured analysis

4. Decision tree

**SYSTEM DESIGN**

**8.1 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.

**8.1.1 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

**8.1.2 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.

**LEVEL 0 DFD**

**COLLEGE**

**UNIVERSITY**

**STUDENT**

**FACULTY**

DATA BASE

**University functionalities:**

**1st Level**

University

**College functionalities:**

**Level 1st**



College

**Faculty functionalities:**

**Level 1st**

**Faculty**

**Student functionalities:**

**Level 1st**

Student

**ER DIAGRAM**

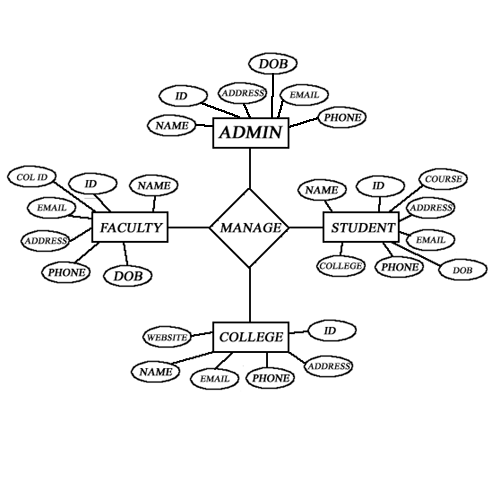
The relation upon the system is structure through a conceptual ER-Diagram, which not only specifics the existential entities but also the standard relations through which the system exists and the cardinalities that are necessary for the system state to continue.

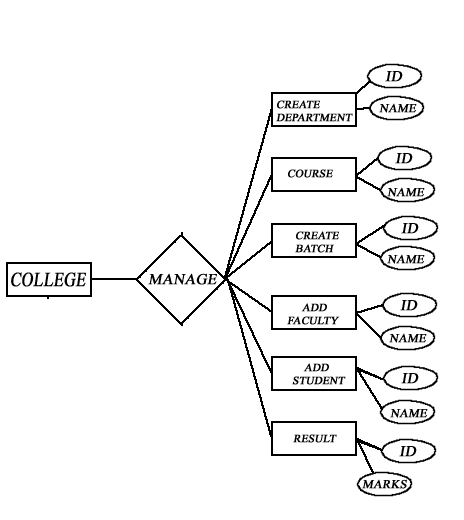
The entity Relationship Diagram (ERD) depicts the relationship between the data objects. The ERD is the notation that is used to conduct the date modeling activity the attributes of each data object noted is the ERD can be described resign a data object descriptions.

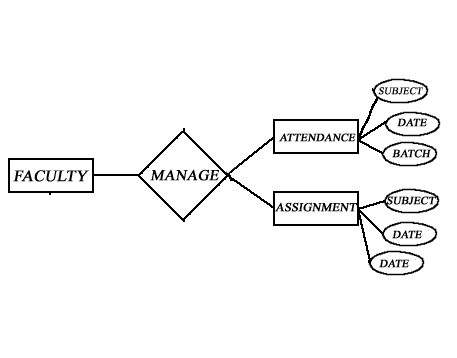
The set of primary components that are identified by the ERD are

* + Data object
  + Relationships
  + Attributes
  + Various types of indicators.

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







**8.2 USECASE DIAGRAMS**

**Usecase diagram for University Admin:**



|  |  |
| --- | --- |
| **Use case name** | Create courses |
| **Participating actors** | University 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** | University 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** | Reports |
| **Participating actors** | University administration |
| **Flow of Events** | Administrator creates reports of colleges and courses |
| **Entry condition** | Admin enter into the system with his own id and pass words |
| **Exit condition** | Successview reports |
| **Quality Requirements** | Generation of Reports |

**Usecase diagram for College Admin:**



|  |  |
| --- | --- |
| **Use case name** | Create batches |
| **Participating actors** | College 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 |

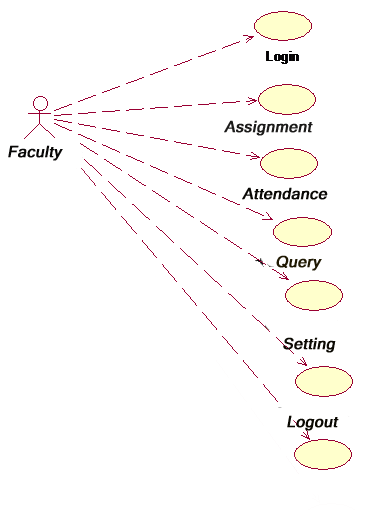
|  |  |
| --- | --- |
| **Use case name** | Create departments |
| **Participating actors** | College administrator |
| **Flow of Events** | College admin creates departments |
| **Entry condition** | College admin enter into with college id and password |
| **Exit condition** | Successful department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

|  |  |
| --- | --- |
| **Use case name** | Select university courses |
| **Participating actors** | College administrator |
| **Flow of Events** | College admin should select whatever courses to be added |
| **Entry condition** | College admin enter into the system with his own id and pass words and course id |
| **Exit condition** | Successful course addition |
| **Quality Requirements** | Duplication addition should not occur |

|  |  |
| --- | --- |
| **Use case name** | Create faculties |
| **Participating actors** | College 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** | College 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 |

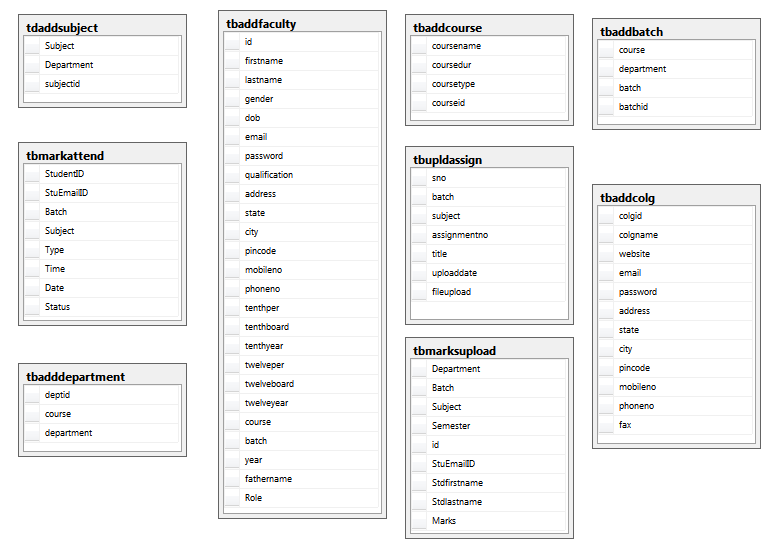
**Usecase diagram for Faculty:**



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

|  |  |
| --- | --- |
| **Use case name** | Assignment |
| **Participating actors** | College faculty |
| **Flow of Events** | Faculty upload Assignment to students and modify them to particular allocated subject |
| **Entry condition** | Faculty enters with his college name and his username and password |
| **Exit condition** | After successful upload assignment |
|  |  |

**8.3 DATABASE TABLES**



**SCREENSHOTS**

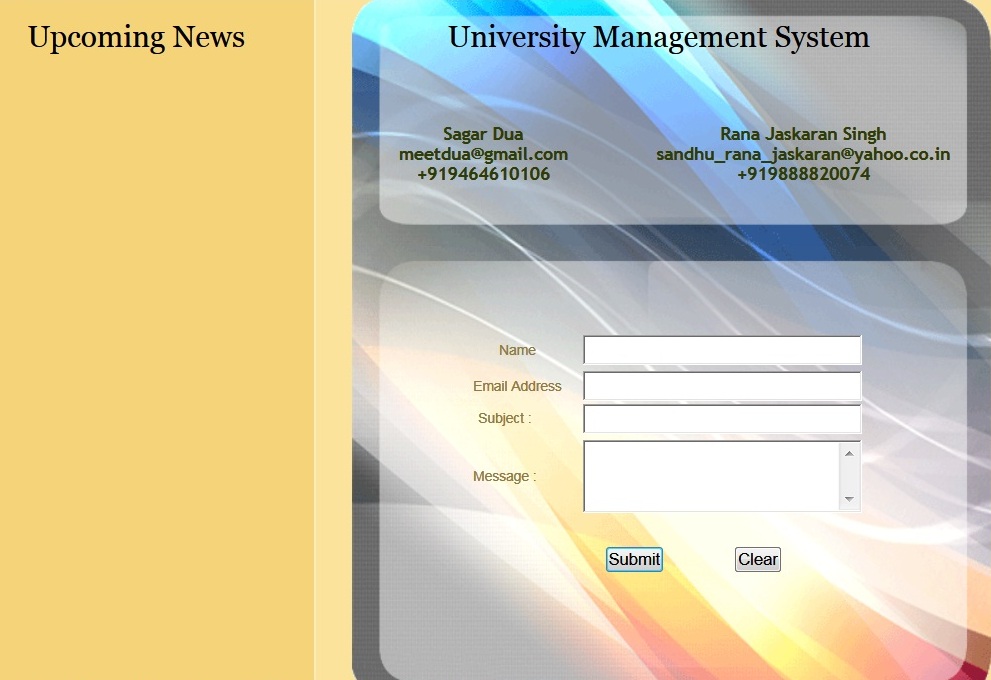
9.1 Homepage:



9.2 About UMS:



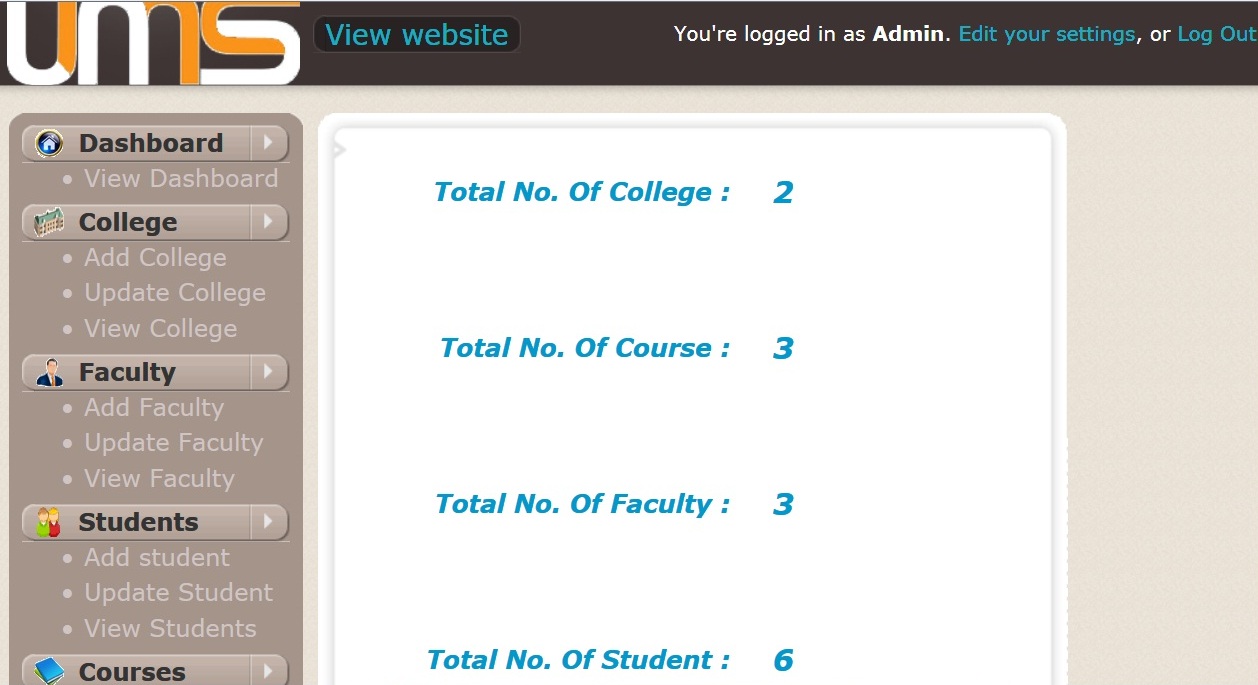
9.3 Contact Us:



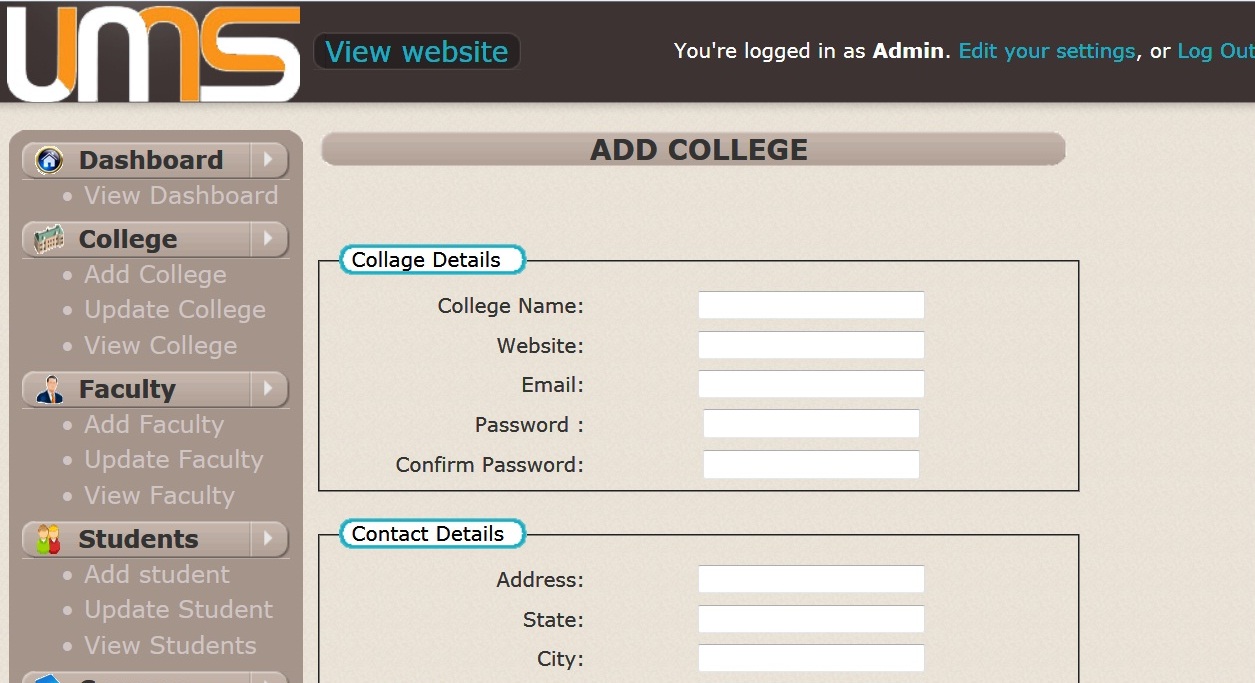
9.4 Login:



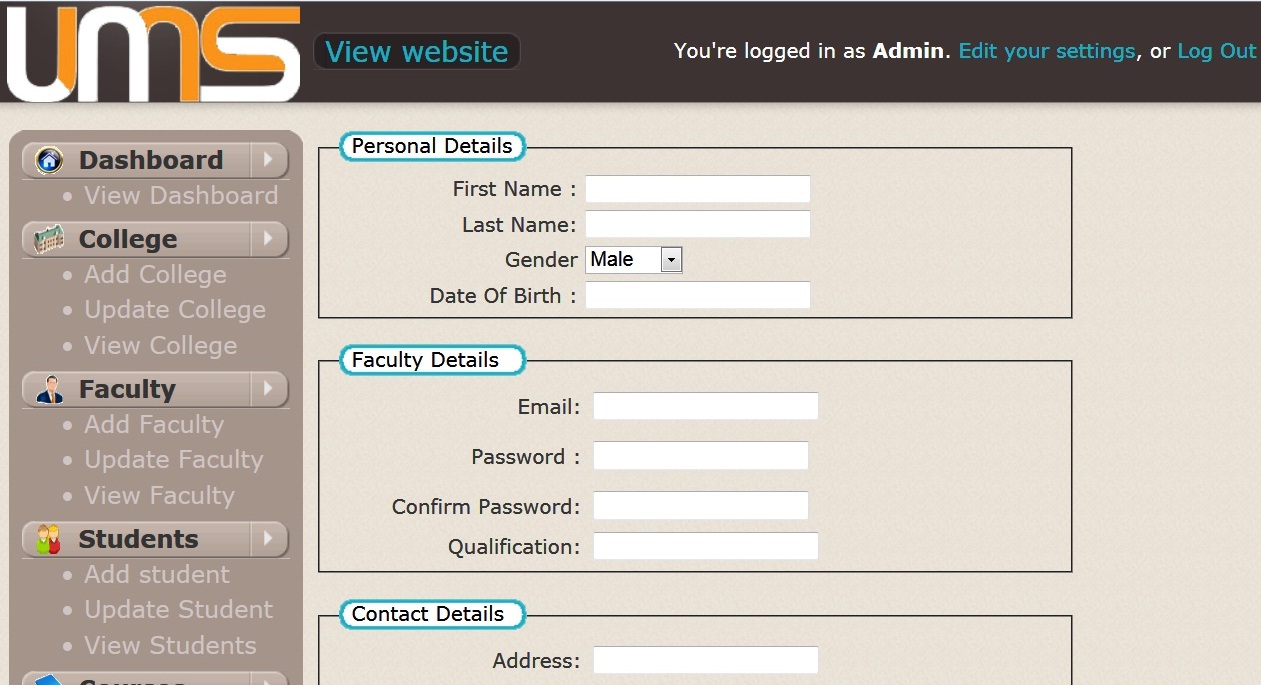
9.5 Admin Homepage:



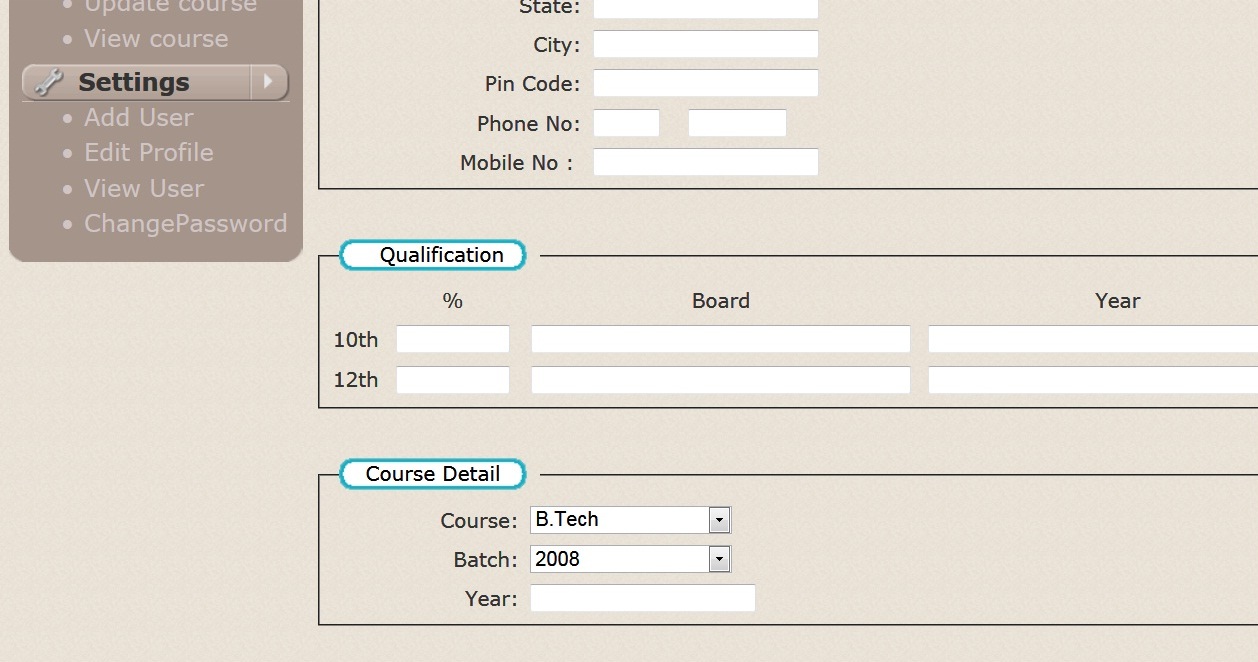
9.6 Add College:



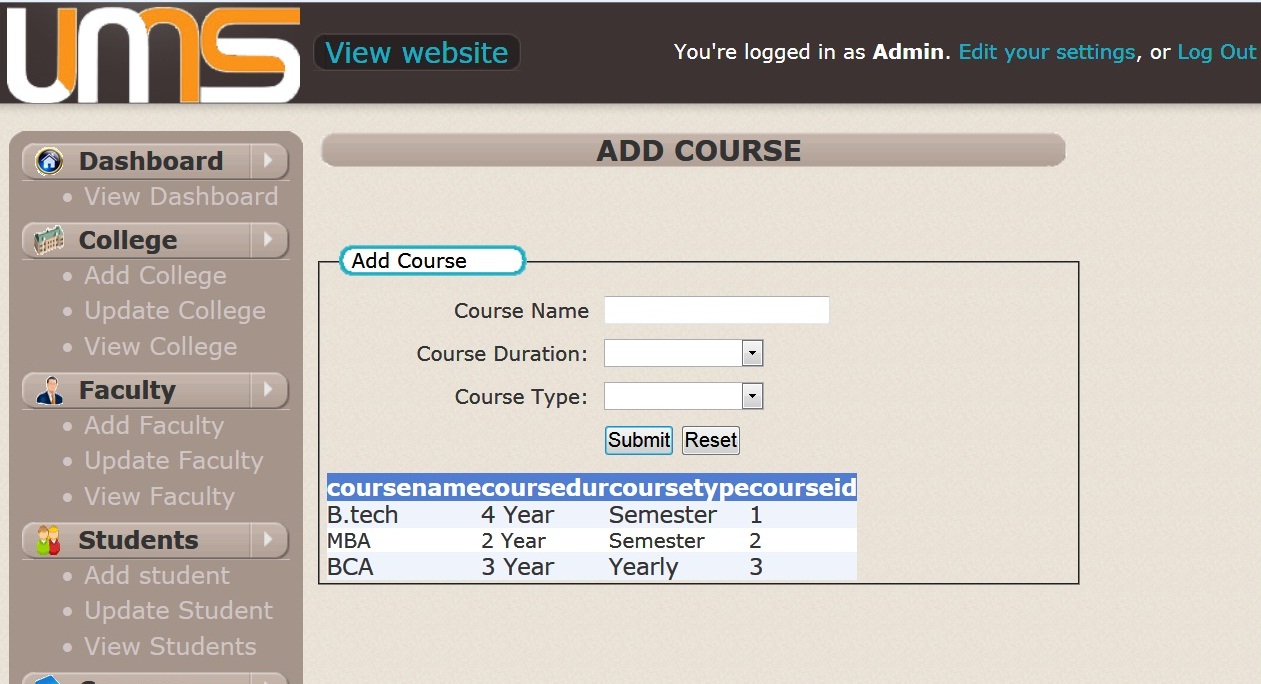
9.7 Add Faculty:



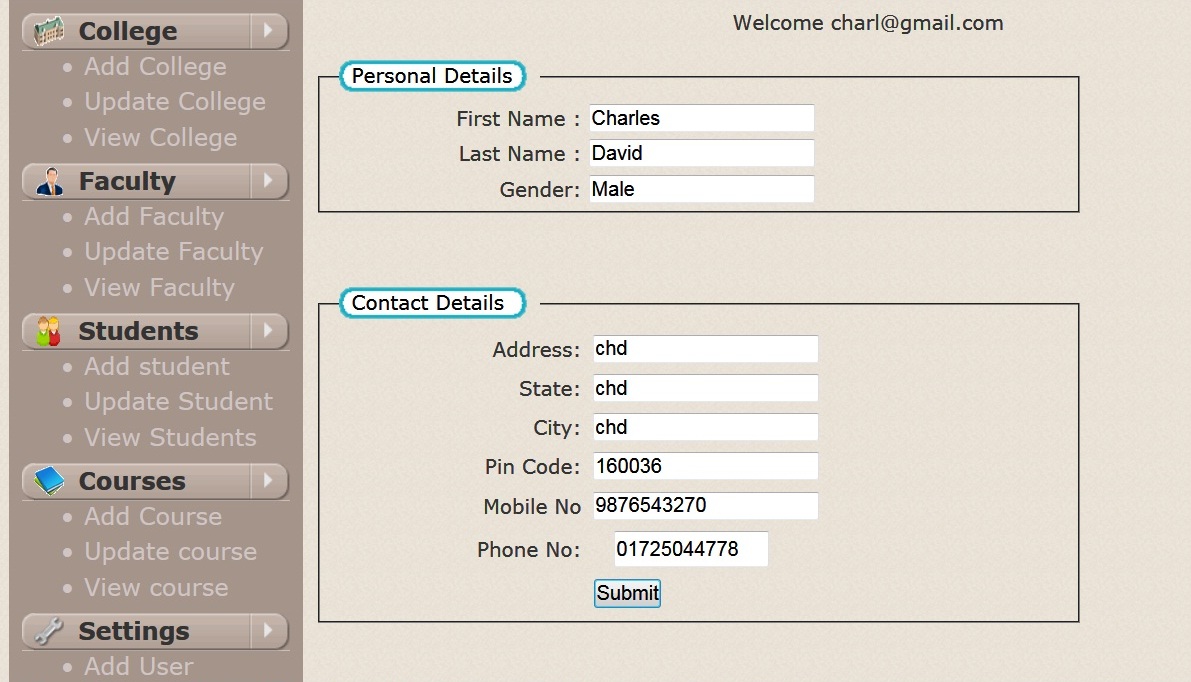
9.8 Add Student:



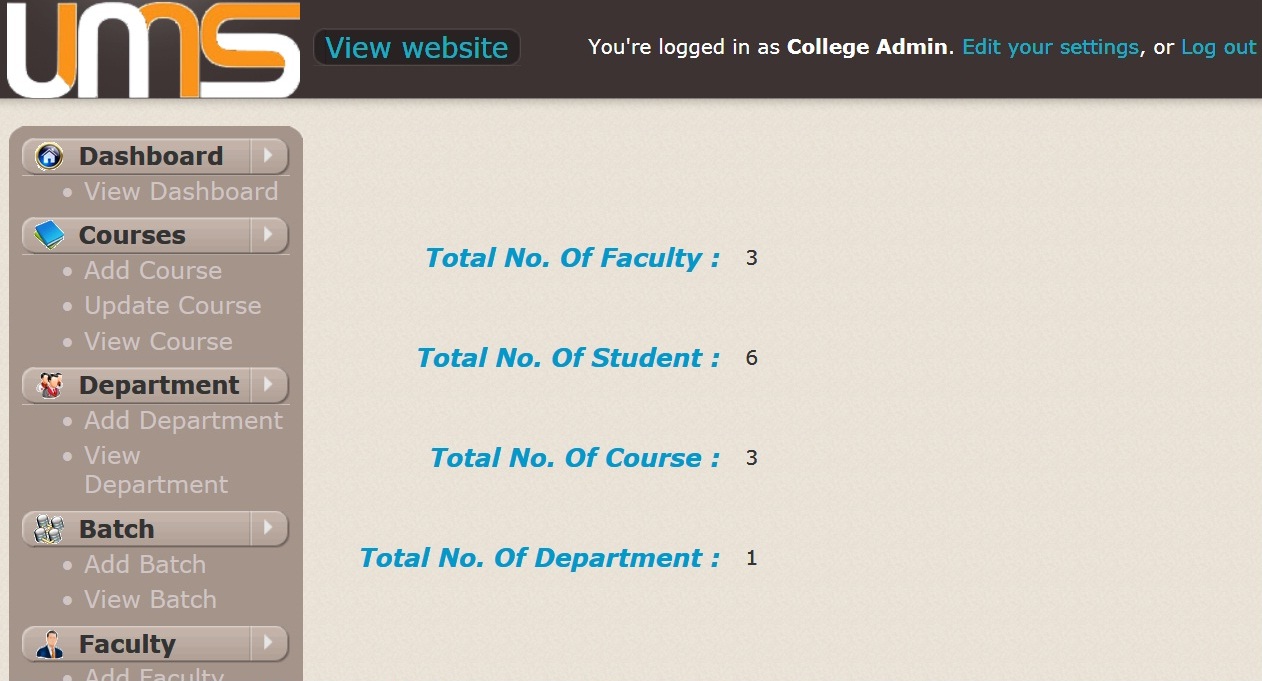
9.9 Add Course:



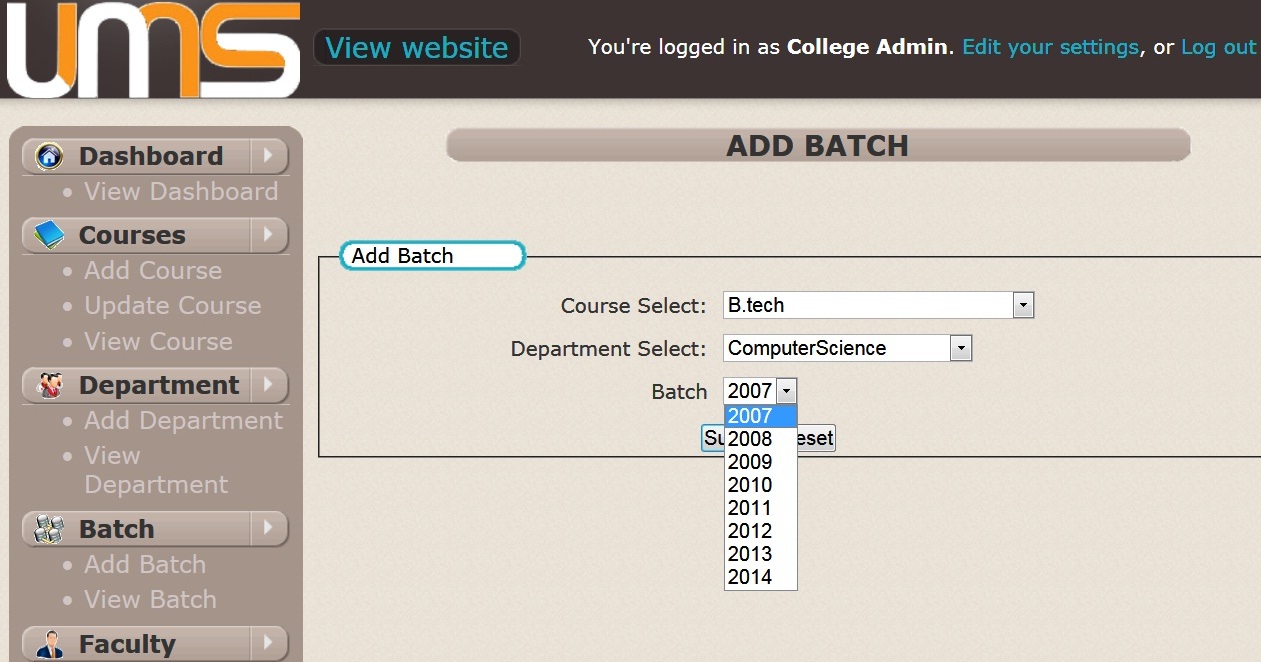
9.10 Edit Profile:



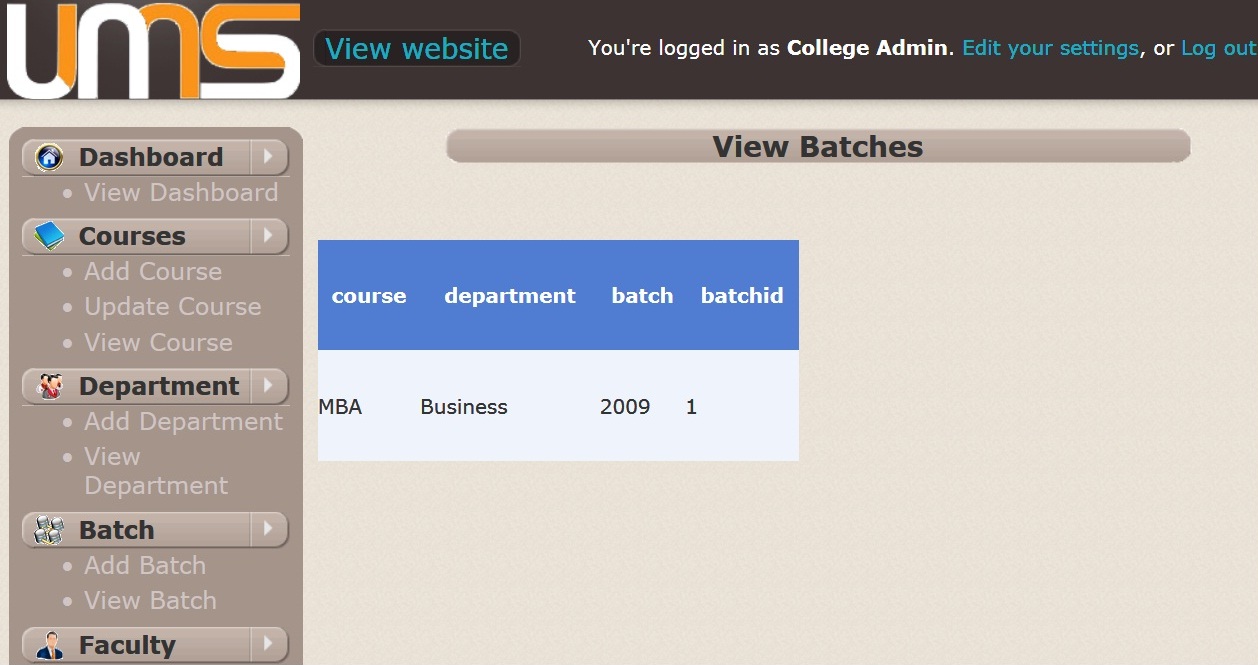
9.11 College Admin Homepage:



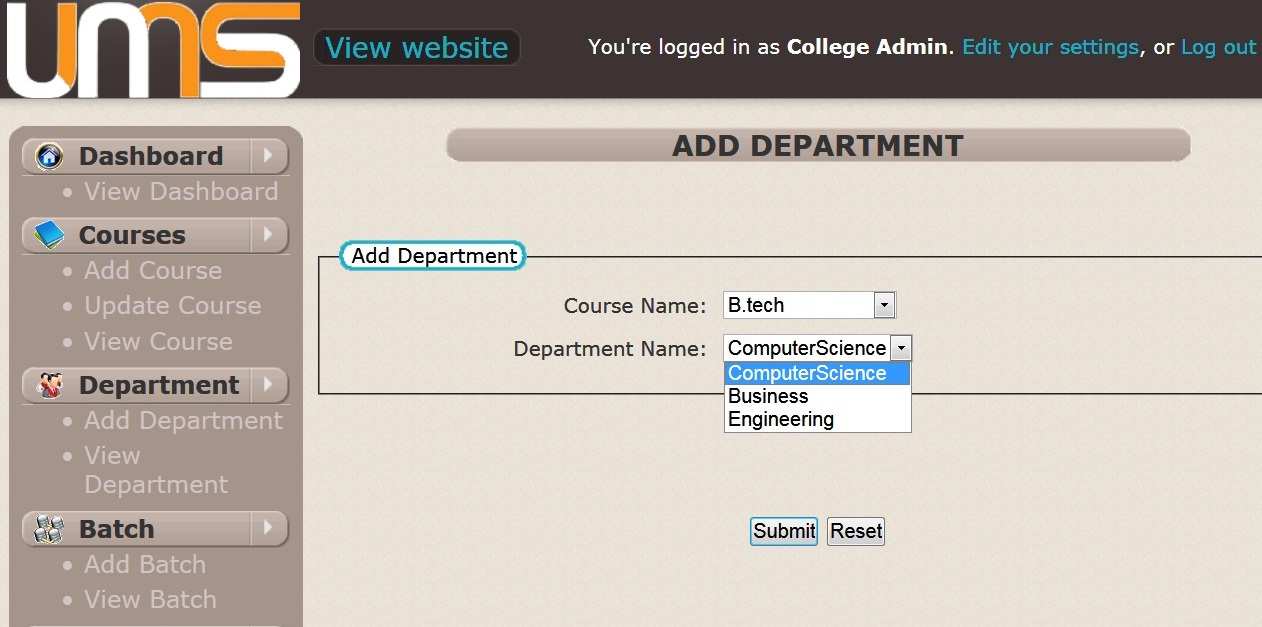
9.12 Add Batch:



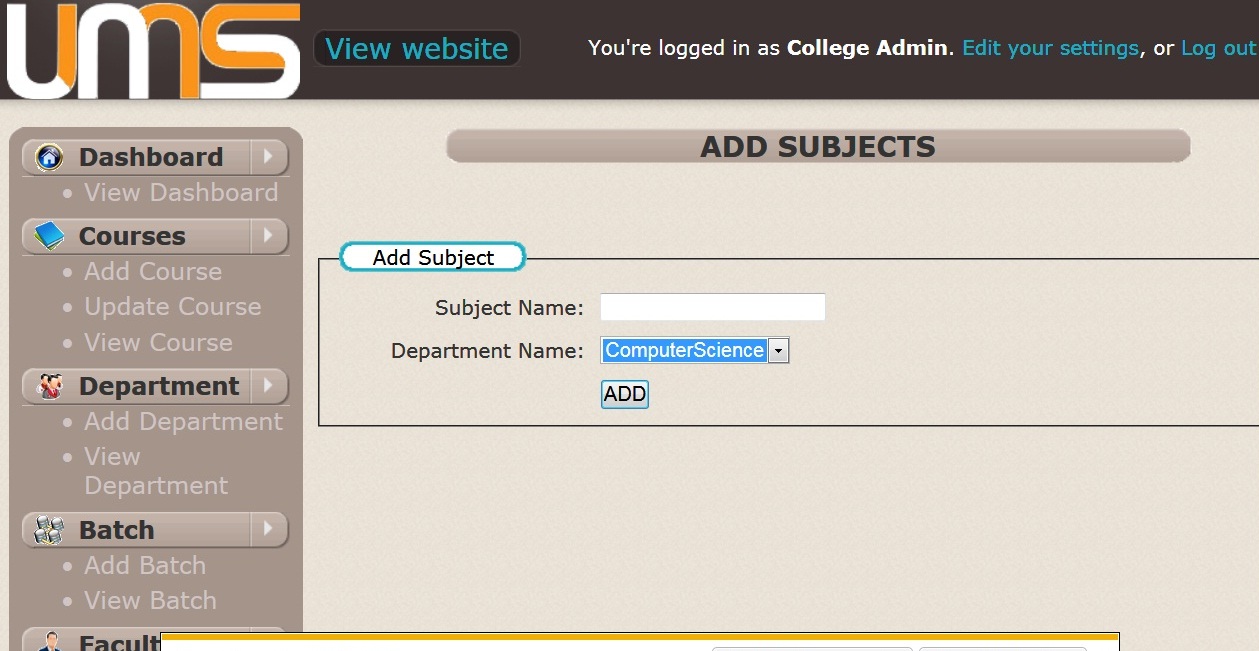
9.13View Batch:



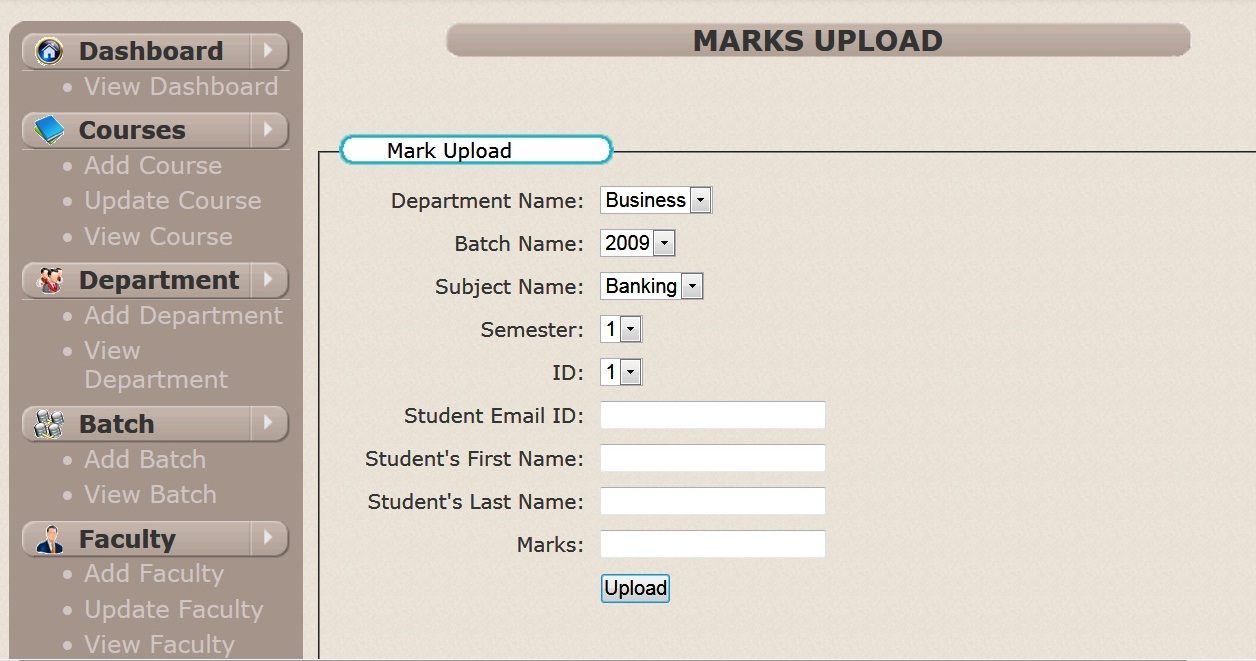
9.14 Add Department:



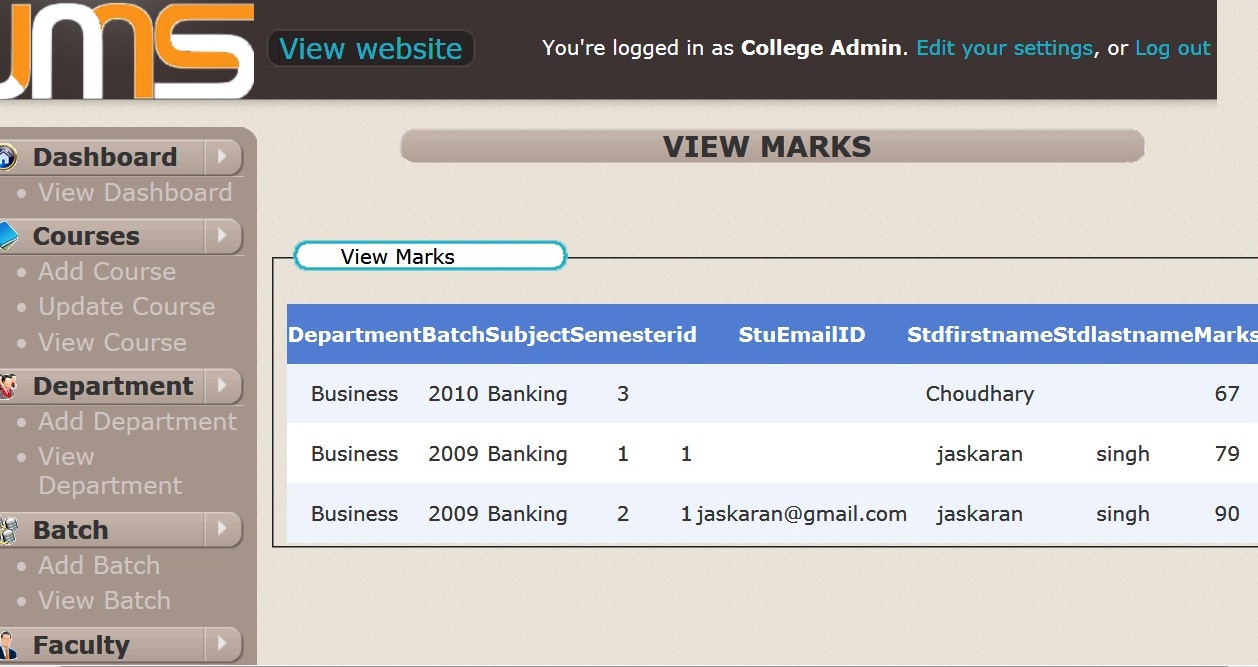
9.15 Add Subject:



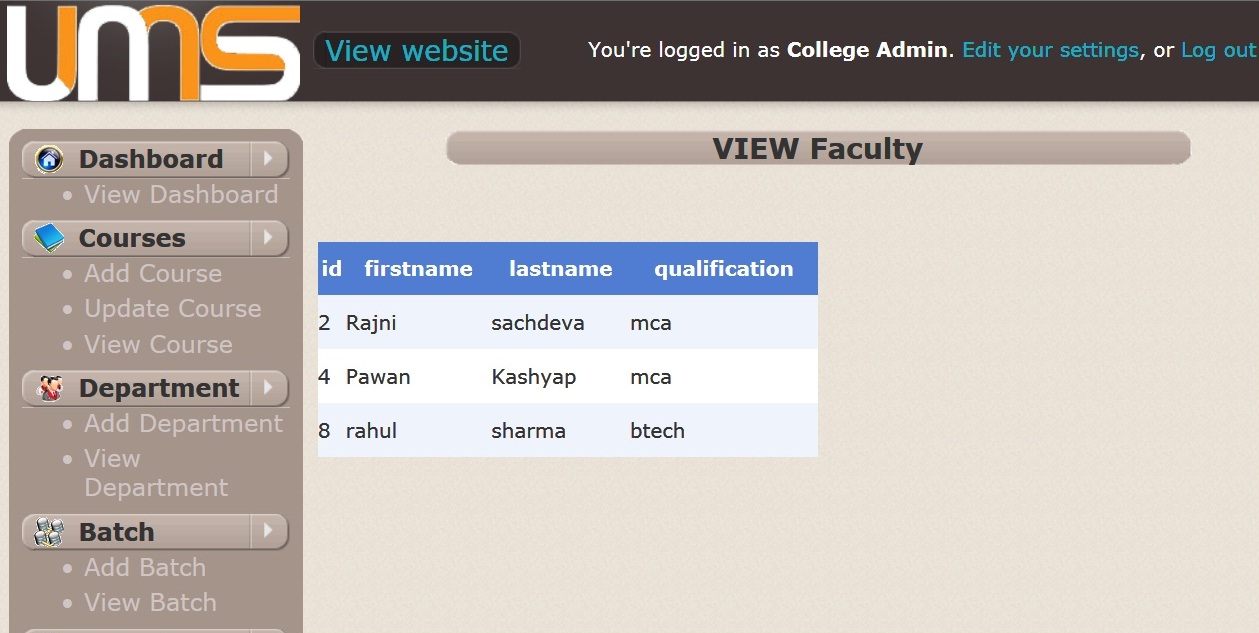
9.16 Upload Marks:



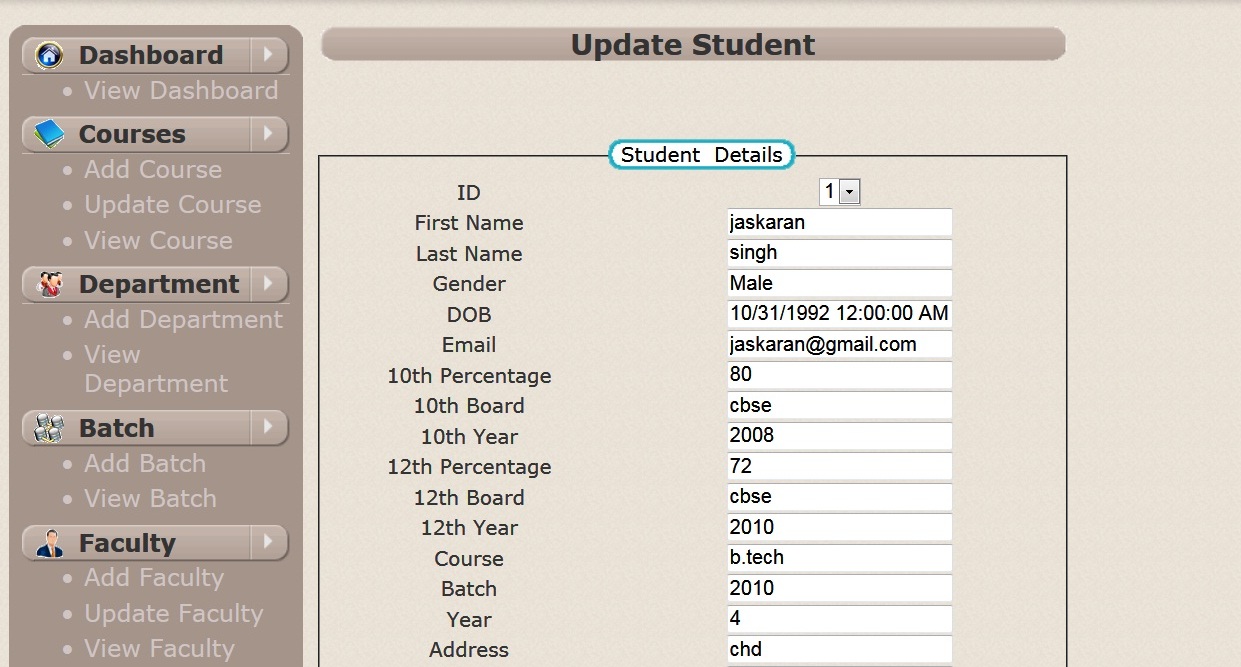
9.17 View Marks:



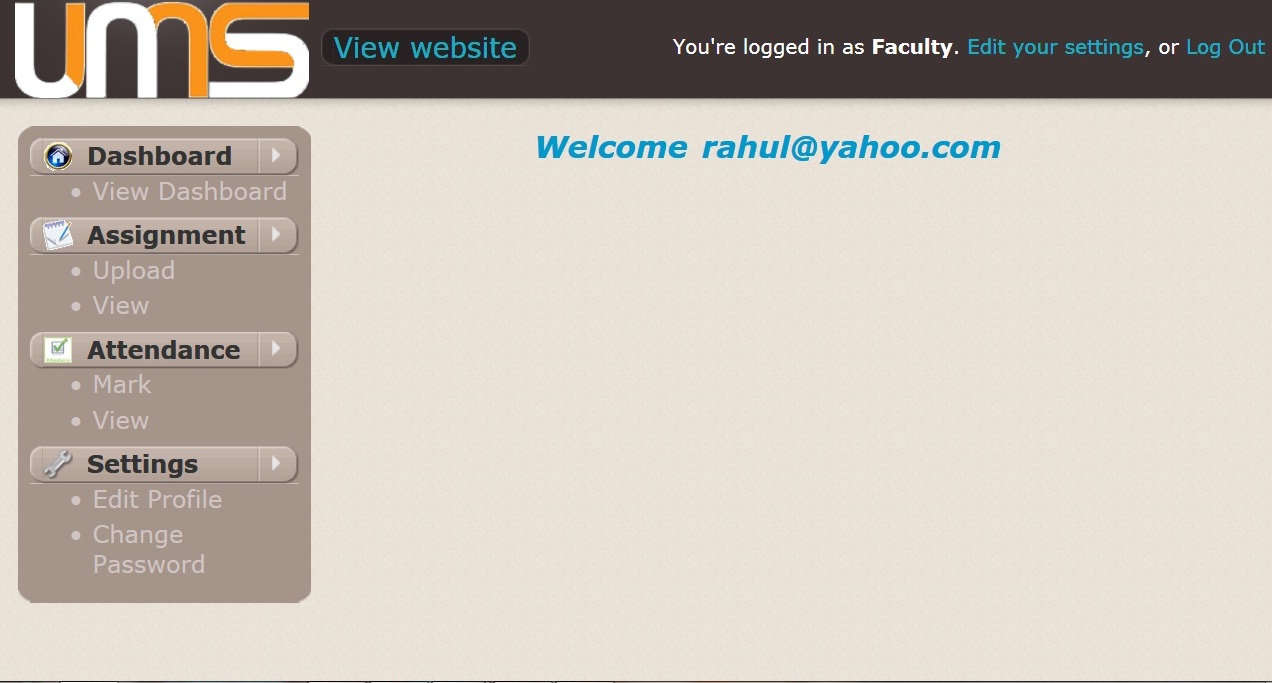
9.18 View Faculty:



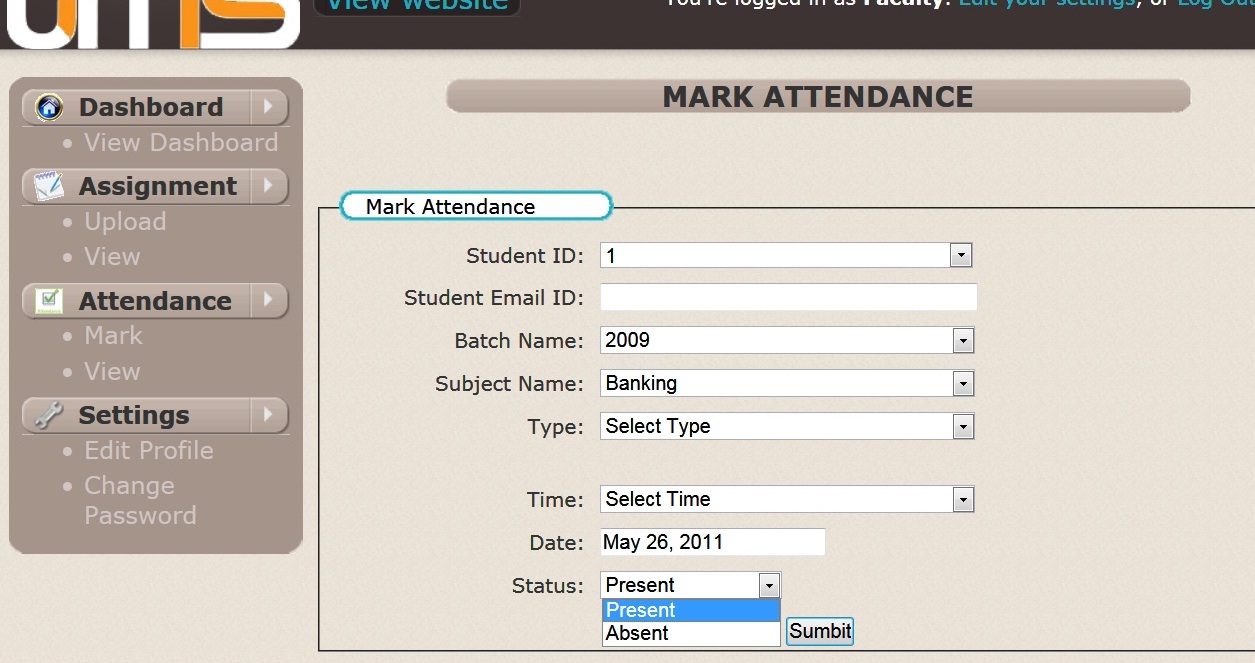
9.19 Update Student:



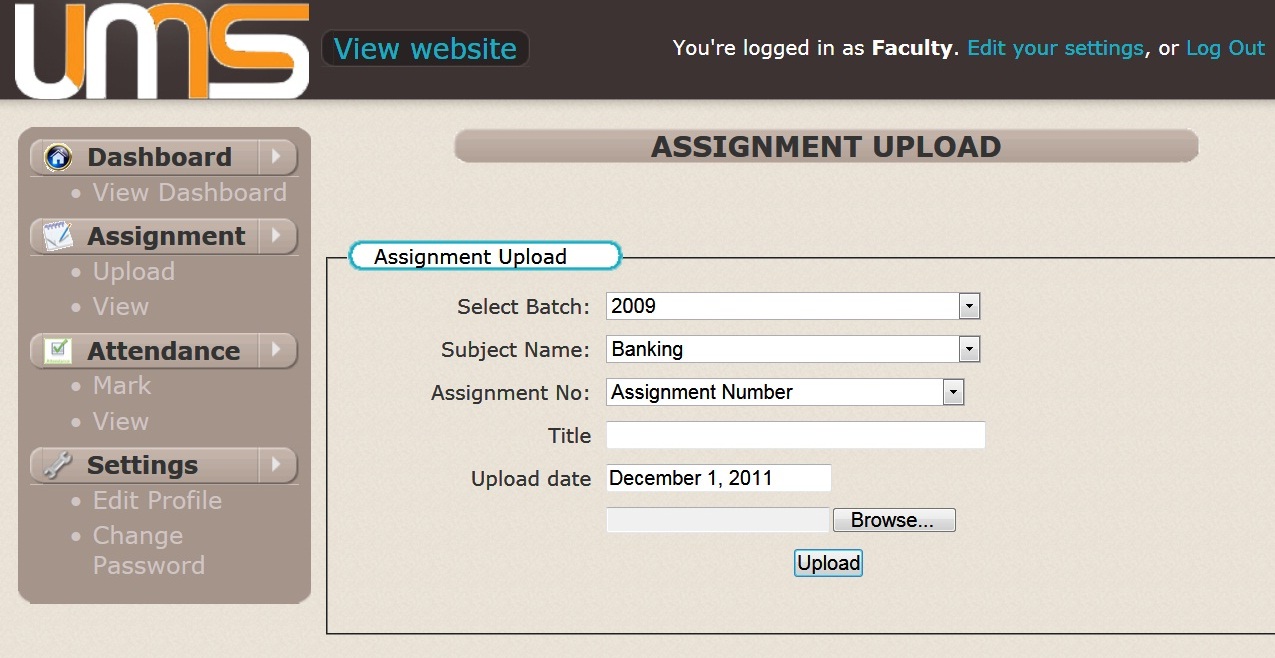
9.20 Faculty Homepage:



9.21 Mark Attendance:



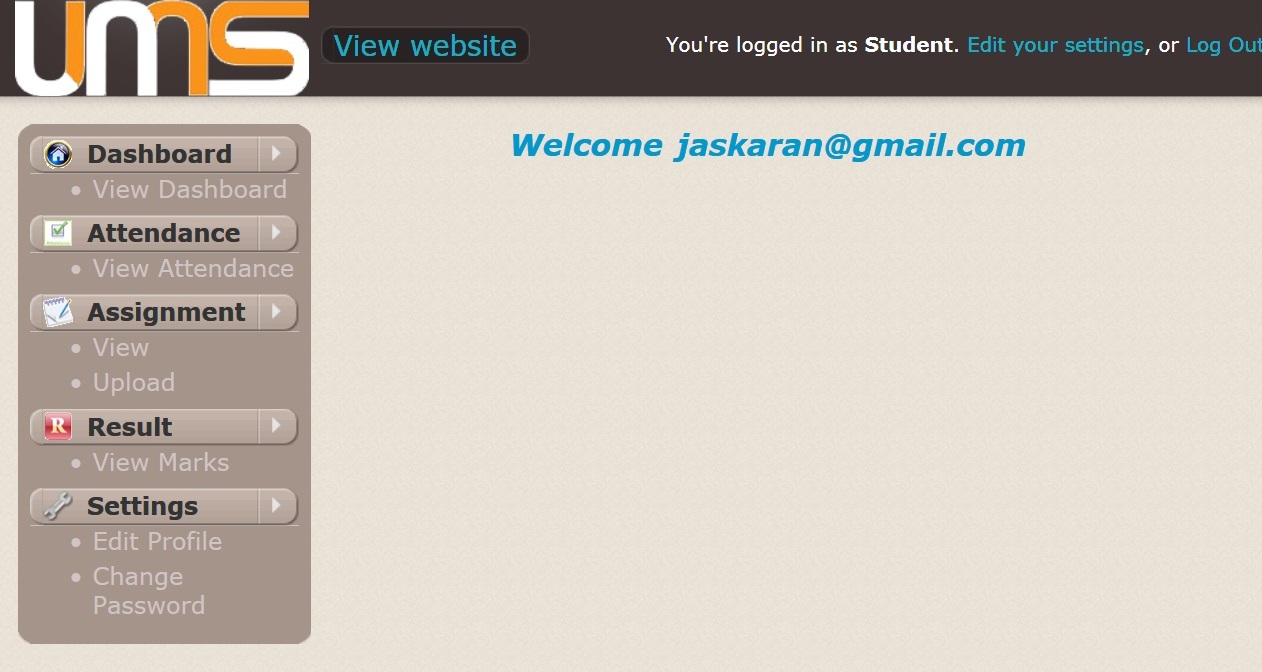
9.22 Upload Assignment:



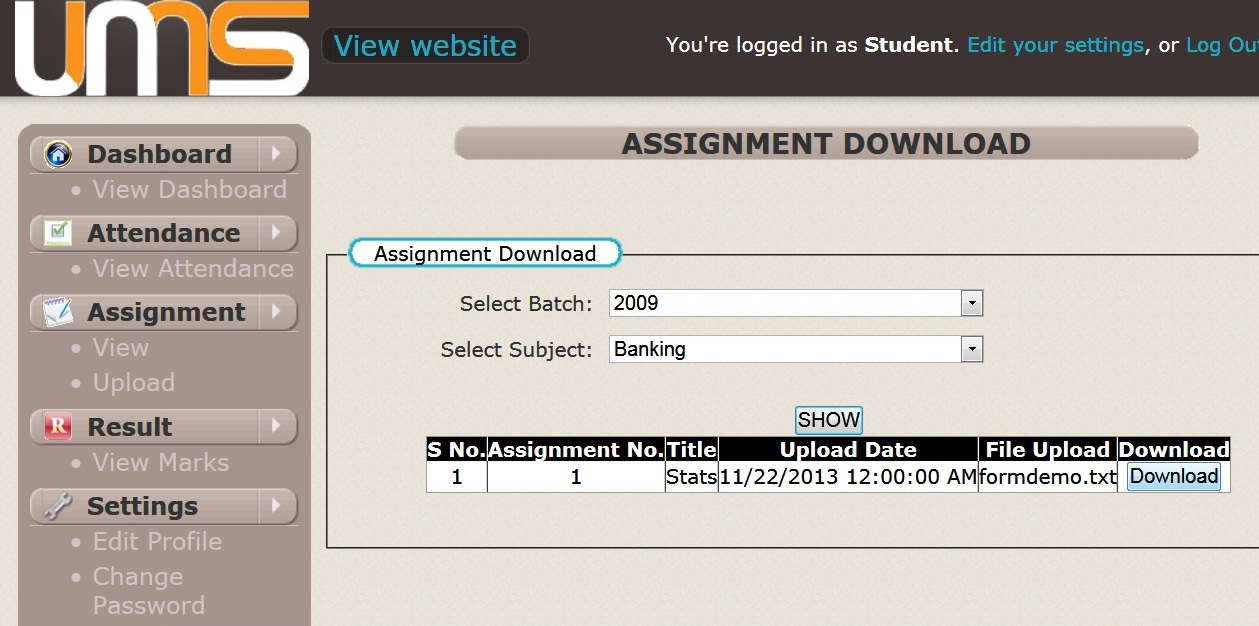
9.23 Change Password:



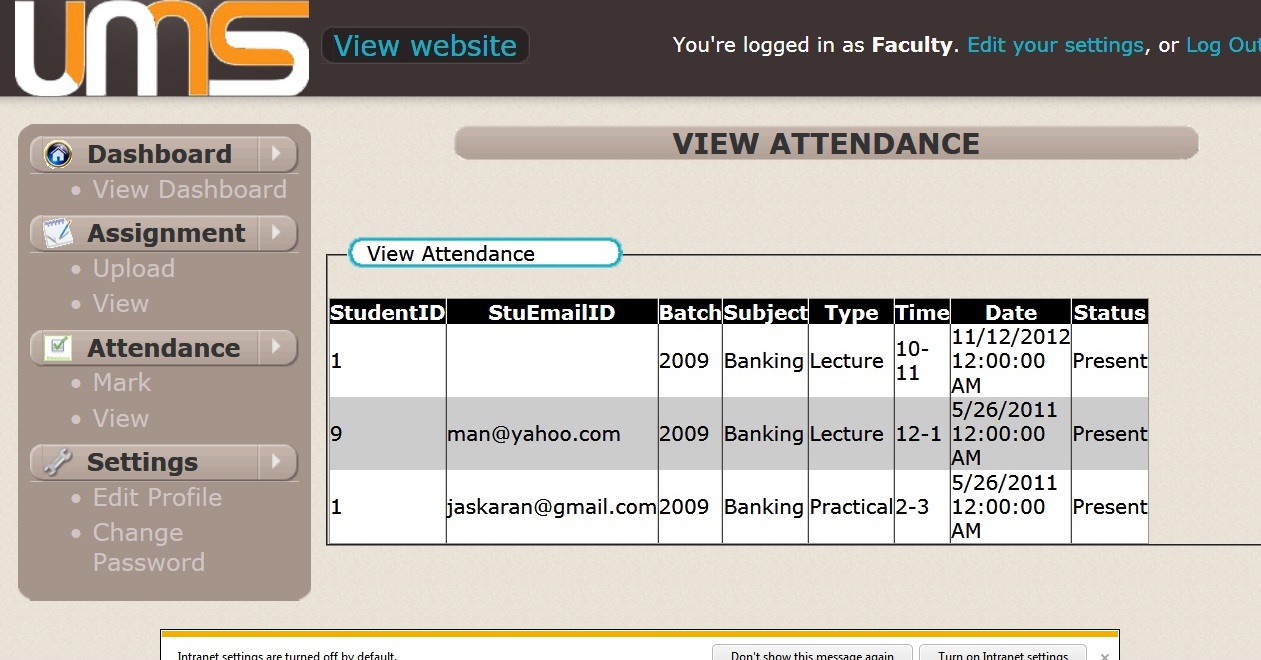
9.24 Student Homepage:



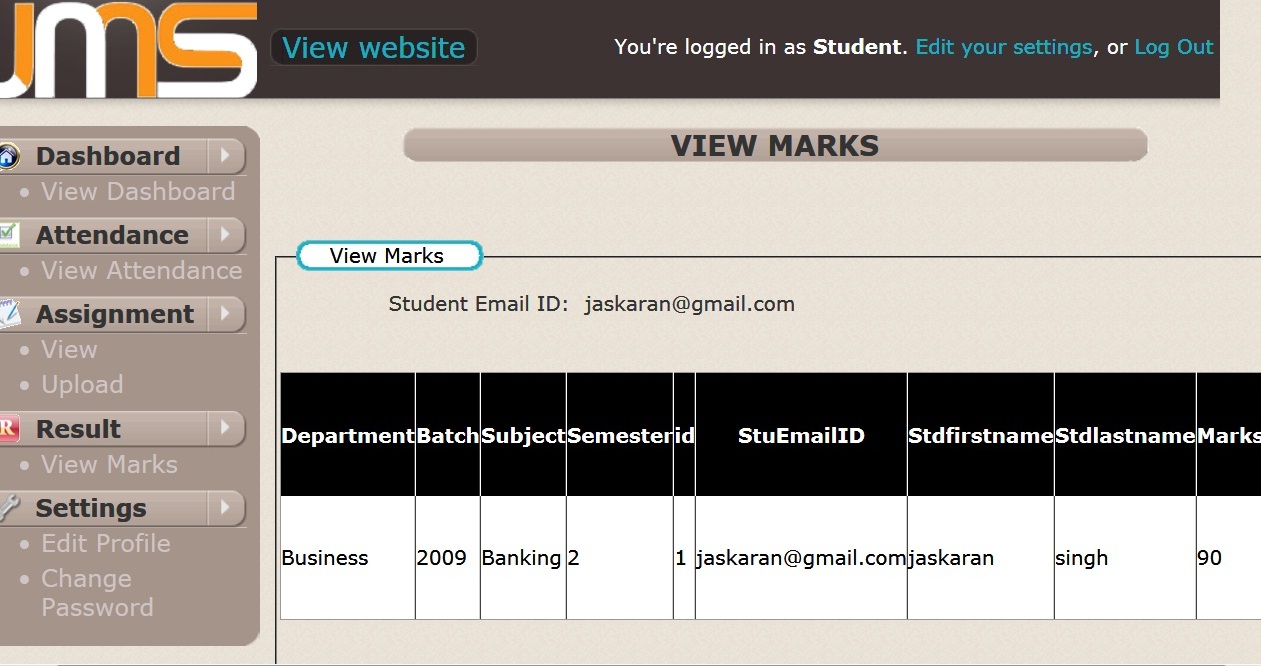
9.25 Download Assignment:



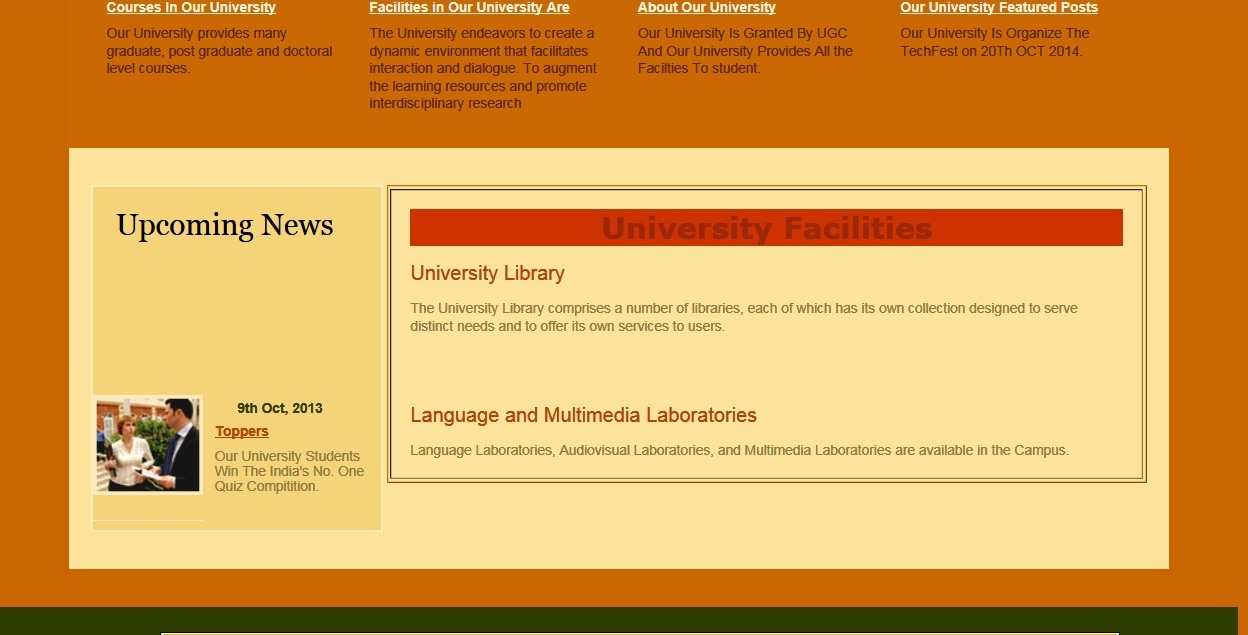
9.26 View Attendance:



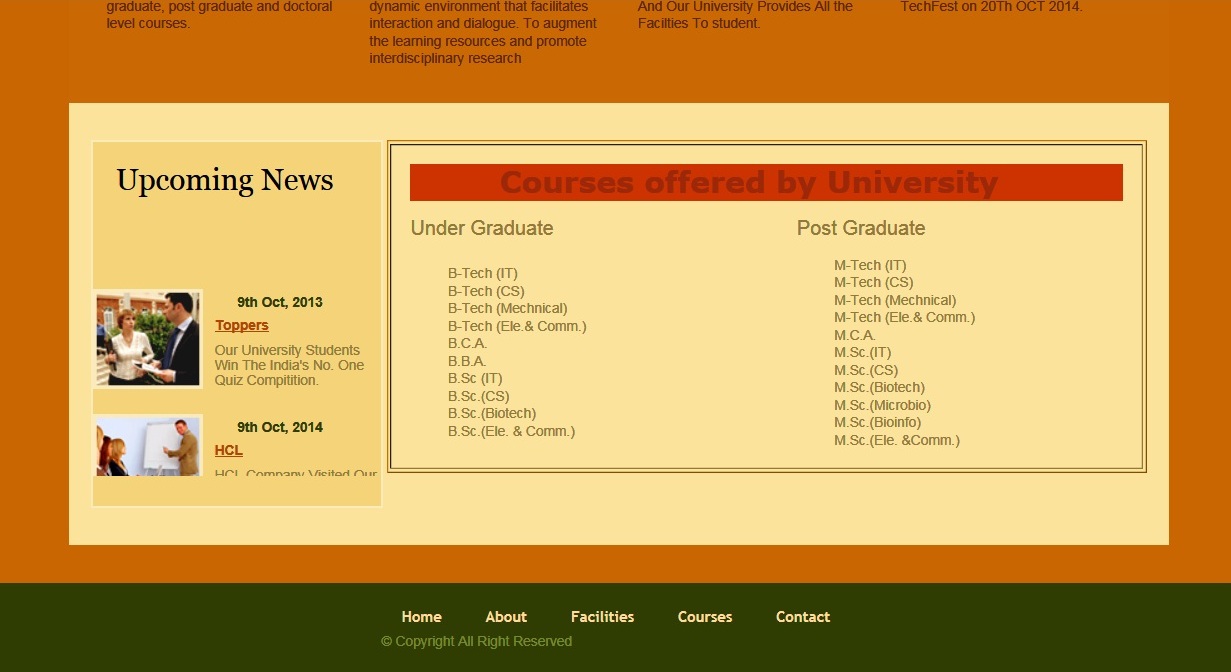
9.27 View Marks:



9.28 Facilities in the University:



9.29 Courses in the University:



**SYSTEM TESTING**

**10.1 TESTING PLAN**

Similar to the project plan, due to confidentiality issues, we cannot provide details test plan to the development team. We will still add the core components that make up our test plan.

1.1.1 Test plan identifier

1.1.2 References

1.1.3 Introduction

1.1.4 Test items (functions)

1.1.5 Application risk issues.

1.1.6 Features to be tested

1.1.7 Features not to be tested

1.1.8 Approach (strategy)

1.1.9 Item pass/fail criteria

1.1.10 Entry & exit criteria

1.1.11 Suspension criteria & resumption requirements

1.1.12 Test deliverables

1.1.13 Remaining test tasks

1.1.14 Environmental needs

1.1.15 Staffing and training needs

1.1.16 Responsibilities

1.1.17 Planning risks and contingencies

1.1.18 Approvals

1.1.19 Glossary

**10.2 TESTING STRATEGY**

Test More and Test Frequent is organization‘s tagline for testing. A typical screen in asp.net is tested at four levels before it goes for production.

Level 1 is generally the work to be tested by other developers or other interns (this is typical first level of testing where focus is not on requirement but end user testing) Ratio: 0% end user: 100% Technical

Level 2 is level where a senior programmer comes into the testing cycle of the screen that was unit tested by the developer in this phase the onus is to test software for technical requirements specified.

Ratio: 80% Technical: 20% end user

Level 3 is where a tester will come into picture. The tester will test the software for both end user as well as technical point of view.

The ratio here is: 50% Technical: 50% end user

Level 4 is where we make the code at Release-Ready. Here screen is tested to the core and each and every standard must be followed and verified.

Ratio here is: 80% User Testing – 20% Technical

This allows us to text a screen at four levels and at the end of four weeks when the screen goes to production, it is generally bug free because more people have looked at this screen from different viewpoints.

References available while testing:

* Project Plan.
* System Requirements specifications.
* High Level design document.
* Detail design document.
* Development and Test process standards.
* Methodology.
* Low level design.

Also organization has Separate module to store all bugs.

So each screen is released for testing as a build and all information for that screen (till release) is maintained using this particular build.

**CONTENT TESTING:**

Errors in Web Application content can be as trivial as minor typographical error as incorrect information, improper organization or validation of intellectual property laws. Content Testing attempt to uncover this and many other problems before the user encounter them.

Content Testing Objectives:

There are three types of objectives.

* To uncover syntactic errors in text-based documents, graphical representation and other media.
* To uncover semantic errors in any content object represented as navigation occurs, and
* To find errors in organization or structure of content that is presented to the end-user

**DATABASE TESTING:**

Modern Web Application does much more than present static content objects. In many application domains, Web Application interface with sophisticated database management system and build dynamic content object that are created in real time using the data acquired from a database.

Database Testing for Web Application is complicated by a variety of factor.

1) The original client side request for information is rarely presented in the form that can be input to a database management system.

2) The database may be remote to the server that houses the Web application.

3) RAW data acquired from the database must be transmitted to the Web application Server and properly formatted for subsequent transmittal to the client.

4) The dynamic content objects must be transmitted to the client in a form that can be displayed to the end user.

**Server layer-**

**Data transformation**

**Server layer- WebApp**

**Client layer-user interface**

Database layer – data access

**Server layer - data**

**Management**

sd

Database

[Layers of interaction]

In figure testing should be ensure that :

1. Valid information is passed between the client and server from the interface layer

2. The Web application process script correctly and properly extracts or formats user data.

3. Queries are passed to a data management layer that communicates with database access routines.

1. User data are passed correctly to a server side data transformation function that format appropriate queries.

**INTERFACE TESTING**

Interface design model is reviewed to ensure that generic quality criteria established for all user interfaces have been achieved and that application specific interface design issue has been properly addressed.

**Interface testing strategy:**

The overall strategy for interface testing is to (1) Uncover error related to specific Interface mechanisms (2) uncover errors in the way the interface implements the semantics of navigation, Web Application functionality, or content display. to accomplish this strategy, a number of objectives must be achieved:

Interface futures are tested to ensure that design rules, aesthetics, and related visual content are available for the user without error. Individual interface mechanisms are tested in a manner that is a logous to unit testing For example; tests are designed to exercise all forms, client-side scripting, dynamic HTML. Each interface mechanism is tested within the context of a use-case or NSU for a specific user category. The interface is tested within a variety of environments to ensure that it will be compatible.

**Testing Interface Mechanisms**

When a user interacts with a Web Application, the interaction occurs through one or more interface mechanisms.

Links: -

Each link is tested to ensure that the proper content object or

function is reached. The web engineer builds a list of all links

associated with interface layout. And then executes each

individually.

Forms: -

At a microscopic level, tests are performed to ensure that Labels correctly identified fields within the form and that mandatory fields are identified visually for the user. The server receives all information content within the form and their no data are lost in the transmission between client and server appropriate defaults are used when the user does not select from a pull down menu or set of buttons. Browser function don’t corrupt data enter in a form and Scripts that perform error checking on data entered work Properly and provide meaningful error message.

Client side scripting:-

Black box tests are conducted to uncover any error in processing As the script is executed. These tests are coupled with forms testing because script input is often derived from data provided as part of forms processing

Dynamic HTML:-

Each Web page that contains dynamic HTML is executed to ensure that the dynamic display is correct. In addition a compatibility test should be conducted to ensure that the dynamic HTML is work properly in the environmental configuration that support the Web application.

Application specific interface mechanisms:-

Test confirms to a checklist of functionality and features that are defined by the interface mechanism. Boundary test minimum and maximum number of item that can be placed in to shopping chart. Test to determine persistence of shopping chart contents. Test to determine whether the Web Application can be record shopping chart content at some future date.

**USABLITY TESTING**

Usability test may be designed by Web engineering team. Define a set of usability testing categories and identify goal for each. Design test that will enable each goal to be evaluated. Select participants who will conduct test. Instrument participant’s interaction with Web Application while testing is conducted. Develop a mechanism for assessing the usability of the Web Application.

The following test categories and objective illustrate establish testing

Interactivity –

Are interaction mechanisms easy to understand and use?

Layout-

Are navigation mechanisms, content and function place in a manner that allows the user to find them quickly?

Readability-

Is text well written and under stable?

Aesthetics-

Do layout color, typeface, and related characteristics lead to ease of use?

Display Characteristics-

Does the Web Application make optimal use of screen size and resolution?

Time Sensitivity-

Can important features, functions and content be used in a timely manner?

Accessibility-

Is the Web application accessible to people who have Disabilities?

**COMPATIBILITY TESTING**

Web application must operate within environment that differs from one another. Different computer, display device, OS, browser and network connection speed can have significant on Web application operation. Different browser some time produced slightly different results, regardless of the degree of HTML standardization within the Web application.

The Web Engineering team derives a series of compatibility, validation tests, derived from existing interface tests, navigation tests, performance tests and security tests.

**10.3 TESTING METHODS**

Testing presents an interesting anomaly for the software engineering activities, the engineer attempts to build software from an abstract concept to a tangible product. Now comes testing. The engineer creates a series of test case that are initiated to "demolish" the software that has been build. Infect, testing is the one step in the software process that could be viewed (psychologically, at least) as destructive rather than constructive.

Models of Testing:-

There are different Models of testing. On the basis of testing methods there are two types of testing:

1. Black-box testing.

2. White-box testing

Black-box tests are used to demonstrate that software functions are operational, that input is properly accepted and output is correctly produced, and that integrity of external information is maintained.

White-box tests are used to examine the procedural details. It checks the logical paths by test case. It can also checks the conditions, loops used in the software coding. It checks that loops are working correctly on defined boundary value.

**WHITE-BOX TESTING:**

White-box testing some times called glass-box testing, is a test case design method that users the control structure of the procedural design to drive the test case. Always we are thinking that there is no necessary to execute or checks the loops and conditions. And so large number of errors is uncovered. With using white-box testing methods, we have checked that; all independent paths within a function have been executed at least once.

All logical decisions on their true and false side. All loops working correctly at their boundary values and within their specified conditions.

In our coding we test that all the loops works truly in each module. The one technique of white-box testing is basis path testing. It contains two parts, one is flow graph notation and the second is cyclometer complexity. In flow graph notation we are checking logical control of flow. By using cyclometer complexity we find complexity of our project structure.

**BLACK-BOX TESTING:**

Black-box testing focuses on the functional requirements of the software. That is black-box testing enables the software engineer to drive sets of input conditions that will fully exercise all functional Requirements for the program. Black-box testing is not an alternative to white-box testing techniques. Rather, it is a complementary approach that is likely to uncover a different class of errors than white-box methods.

We use in our coding to find errors in the following categories:

* Incorrect or missing functions
* Interface errors
* Errors in database
* Performance errors
* Initialization and termination errors.

Unlike white-box testing, which is performed earlier in the testing process, black-box testing tends to be applied during later stages of testing. Because black-box testing purposely disregards control structure, attention is focused on the information domain.

By applying black-box techniques, we derive a set of test cases that satisfy following criteria

Test cases that reduce, by a count that is greater then one, the number of additional test cases must be designed to achieve reasonable testing.

Level 1 - Build Acceptance Tests

Other related test cases ensure that adopters received the proper Development Release Document plus other build related information (drop point, etc.). The objective is to determine if further testing is possible. If any Level 1 test case fails, the build is returned to developers un-tested.

Level 2 - Smoke Tests

The objective is to determine if further testing is possible. These test cases should emphasize breadth more than depth. All components should be touched, and every major feature should be tested briefly by the Smoke Test. If any Level 2 test case fails, the build is returned to developers un-tested.

Level 2a - Bug Regression Testing

Every bug that was “Open” during the previous build, but marked as “Fixed, Needs Re-Testing” for the current build under test, will need to be regressed, or re-tested. Once the smoke test is completed, all resolved bugs need to be regressed. It should take between 5 minutes to 1 hour to regress most bugs.

Level 3 - Critical Path Tests

Critical Path test cases must pass by the end of every 2-3 Build Test Cycles. They do not need to be tested every drop, but must be tested at least once per milestone. Thus, the Critical Path test cases must all be executed at least once during the Iteration cycle, and once during the Final Release cycle.

Level 4 - Standard Tests

Test Cases that need to be run at least once during the entire test cycle for this release. These cases are run once, not repeated as are the test cases in previous levels. Functional Testing and Detailed Design Testing (Functional Spec and Design Spec Test Cases, respectively). These can be tested multiple times for each Milestone Test Cycle (Iteration, Final Release, etc.).

Standard test cases usually include Installation, Data, GUI, and other test areas.

Level 5 - Suggested Test

These are Test Cases that would be nice to execute, but may be omitted due to time constraints

**Bug Regression**

Bug Regression will be a central tenant throughout all testing phases. When a Severity 1 bug fails regression, adopters testing team should also put out an immediate email to development. The Test Lead will be responsible for tracking and reporting to development and product management the status of regression testing.

**CONCLUSION AND FUTURE EXTENSION**

**11.1 CONCLUSION**

The system has been developed for the given condition and is found working effectively. The developed system is flexible and changes whenever can be made easy. Using the facilities and functionalities of .Net, the software has been developed in a neat and simple manner, thereby reducing the operators work.

The speed and accuracy are maintained in a proper way. The user friendly nature of this software developed in .net framework is very easy to work with both for the higher management as well as other employees with little knowledge of computer. The results obtained were fully satisfactory from the user point of view.

The system was verified with valid and invalid data in each manner. The system is run with an insight into the necessary modifications that may require in the future. Hence the system can be maintained successfully without much network.

**11.2 FUTURE EXTENSIONS**

* Standardization of application so that it may be replicate in other Institutes also.

**BIBLIOGRAPHY**

**Books**

* ASP.NET(Black Book).
* Professional ASP.NET(Wrox Publication).
* C# Vijaymukhi.
* ASP.NET Complete Reference.
* Software engineering Concepts By Roger S.Presman
* UML IN A NUTSHELL By Alhir
* Fundamentals of Software Engineering By Rajib Mall
* SQL Server 2008 (Wrox Publication).

**Web Sites**

* [www.w3schools.com](http://www.w3schools.com)
* www.developer.paypal.com