**SCHOOL MANAGEMENT SYSTEM**



**Submitted by**

Hafiz Tahir Mehmood 473

Touseef Ahmed 480

**Supervisors:**

Dr.Saleem-u-Allah

Mr. Musa

**IN THE FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF BACHELORS IN INFORMATION TECHNOLOGY**

**Session: 2011-2015**

**DEPARTMENT OF CS&IT**

**The Islamia University of Bahawalpur**



**AUTHOR’S DECLARATION**

Hafiz Tahir Mehmood, Bachelor scholar in the Department of CS&IT at The Islamia University of Bahawalpur do solemnly declare that the documentation entitled, “**School Management System**” submitted by me in fulfillment of the requirement of degree of BSIT. I solemnly declare that this is my original work and has not been submitted or published earlier and also shall not be submitted in future. It shall also not be submitted to obtain any degree to any other university or institution.

Hafiz Tahir Mehmood & Toseef Ahmed

**FORWARDING CERTIFICATE**

The documentation entitled “**School Management System**” is conducted under my supervision and the documentation is submitted to The Islamia University of Bahawalpur in the fulfillment of the requirement of the degree of Bachelor in Information Technology in DCS&IT with my permission.

Dr.Saleem-ullah

Mr. Musa

.

.

**APPROVAL CERTIFICATE**

The documentation entitled “**School Management System**” is accepted and approved in fulfillment of the requirement for the degree of Bachelor in Information Technology.

**Supervisor:**

**External Examiner:**

**Chairman:**

**Date:**

**DEDICATION**

**DEDICATED TO**

**My Loving Parents & Family**

**ACKNOWLEDGEMENT**

All praises for almighty Allah, who enables us to know about certain unknown things in the universe and helps us to overcome a lot of difficulties. All respect for Holy prophet Muhammad (PBUH) who clearly mentioned the difference of right and wrong path, to ensure the success in our lives.

I am deeply indebted and wish my utmost appreciation and gratitude to our Project supervisors **Dr.Saleem-u-Allah & Mr. Musa** for her encouragement, technical discussion, inspiring guidance, remarkable suggestions, keen interest and constructive criticism which enabled me to complete this project.

I express my thanks to honorable **Dr.Saleem-u-Allah & Mr. Musa** Ali for providing me possible facilities for the completion of this project. I have no describable emotional feelings for my Father and valuable supervision and nice cooperation, his moral training gave me the self -confidence and made me the optimistic. I cannot find proper words to express my heartiest gratitude to my loving mother, and family whose chain of prayers, cooperation have created incredible impression in my life.

Words are lacking to express my special gratitude and sincere thanks to my kind seniors, All staff of CS&IT Department and all others ,whose company will ever boost me in life and their way of encouragement ,support, constructive criticism and their loving behavior will be ever source of guidance, happiness and success in my life.

In the end, once again I am humbly thankful to Allah granted to me to complete this project.

Hafiz Tahir Mehmood

Toseef Ahmed

**INDEX**

**S. N CONTENTS**

1. **INTRODUCTION**
2. **ANALYSIS**

2.1. SYSTEM ANALYSIS

2.2. SYSTEM SPECIFICATIONS

1. **DESIGN APPROACH**

3.1. INTRODUCTION TO DESIGN

3.2. UML DIAGRAMS

3.3. DATA FLOW DIAGRAMS

3.4. E-R DIAGRAMS

1. **PROJECT MODULES**

4.1. CONCEPTS AND TECHNIQUES

1. **IMPLEMENTATION**
2. **TESTING**
3. **OUTPUT SCREENS**
4. **CONCLUSION**
5. **FUTURE ENHANCEMENTS**
6. **BIBILIOGRAPHY**

### 

### Introduction

**INTRODUCTION:**

As the modern organizations are automated and computers are working as per the instructions, it becomes essential for the coordination of human beings, commodity and computers in a modern organization.

At present the school management and it’s all procedures are totally manual based. It creates a lot of problems due to wrong entries or mistakes in totaling etc. This system avoided such mistakes through proper checks and validation control methods in checking of student record, fee deposit particulars, teachers schedule, examination report, issue of transfer certificates etc. I met personally to the principal and manager and discuss about the computerization of manual school management system. This system registers a student and confirms its admission in school. When a student registers in school a S.R. No (unique ID) is allotted to student. Student record is based on his/ her S.R. No.

It has facilities to generate various types of reports, which are required by the management during normal business operations to operate the business effectively.

### 

### Analysis

**SYSTEM ANALYSIS**:

1. **Existing System**

Existing system is a manual one in which users are maintaining records to store the information like Student Details, Instructor Details, Schedule Details and feedbacks about students who get admission in the institute.. It is very difficult to maintain historical data.

**DISADVANTAGES:**

The following drawbacks of existing system emphasize the need for computerization:

* 1. A lot of handwork have to be made
  2. A lot of correction work hence delay in giving the results
  3. A lot of tabulation work for each subject results

1. **Proposed System**

This application is used to keep computerized records of students. This application will perform correction, display the result immediately and also store it in database. This application provides the administrator with a facility to add new records. This application takes care of authentication of the administrator as well as the student.

**3. Objective of the System**

# The main objective of this project is to computerize the manual system & reduce the time consumption. In other words we can say that our project has the following objectives:-

# • Make all the system computerize

# • Reduce time consumption

# • Reduce error scope

# • All system managements are automated

# • Centralized database management

# • Easy operations for operator of the system

# • No paper work requirement

**System Specifications**

**Hardware Requirements:-**

* Pentium-IV (Processor).
* 256 MB Ram
* 512 KB Cache Memory
* Hard disk 10 GB
* Microsoft Compatible 101 or more Key Board

**Software Requirements: -**

* **Operating System :** Windows
* **Web-Technology:** PHP
* **Front-End:** HTML, CSS, JAVASCRIPT, JQUERY
* **Back-End:** MySQL
* **Web Server:** Apache SERVER.

### 

### Design

**INTRODUCTION:**

Design is the first step in the development phase for any techniques and principles for the purpose of defining a device, a process or system is sufficient detail to permit its physical realization.

Once the software requirements have been analyzed and specified the software design involves three technical activities - design, coding, implementation and testing that are required to build and verify the software.

The design activities are of main importance in this phase, because in this activity, decisions ultimately affecting the success of the software implementation and its ease of maintenance are made. These decisions have the final bearing upon reliability and maintainability of the system. Design is the only way to accurately translate the customer’s requirements into finished software or a system.

Design is the place where quality is nurtured in development. Software design is a process through which requirements are translated into a representation of software. Software design is conducted in two steps. Preliminary design is concerned with the transformation of requirements into data.

**UML Diagrams:**

**Actor:**

A coherent set of roles that users of use cases play when interacting with the use cases.

**Use case:**

A description of sequence of actions, including variants, that a system performs that yields an observable result of value of an actor.

UML stands for Unified Modeling Language. UML is a language for specifying, visualizing and documenting the system. This is the step while developing any product after analysis. The goal from this is to produce a model of the entities involved in the project which later need to be built. The representation of the entities that are to be used in the product being developed need to be designed.

There are various kinds of methods in software design:

They are as follows:

* Use case Diagram
* Sequence Diagram
* Collaboration Diagram
* Activity Diagram
* State chat Diagram
* **USECASE DIAGRAMS:**

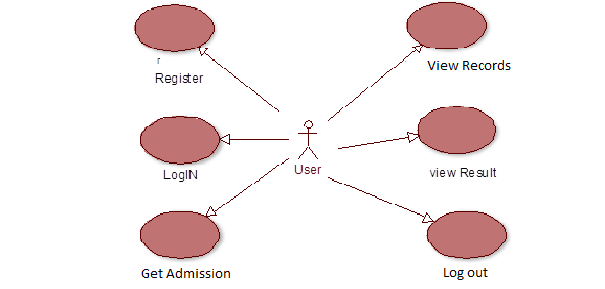
Use case diagrams model behavior within a system and helps the developers understand of what the user require. The stick man represents what’s called an actor.

Use case diagram can be useful for getting an overall view of the system and clarifying who can do and more importantly what they can’t do.

Use case diagram consists of use cases and actors and shows the interaction between the use case and actors.

* The purpose is to show the interactions between the use case and actor.
* To represent the system requirements from user’s perspective.
* An actor could be the end-user of the system or an external system.

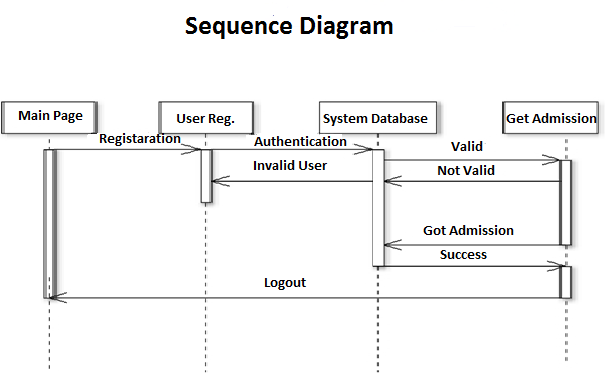
A Use case is a description of set of sequence of actions. Graphically it is solidified as an ellipse with solid line including only its name. Use case diagram is a behavioral diagram that shows a set of use cases and actors and their relationship. It is an association between the use cases and actors. An actor represents a real-world object. Primary Actor – Sender, Secondary Actor Receiver.



* **SEQUENCE DIAGRAM:**

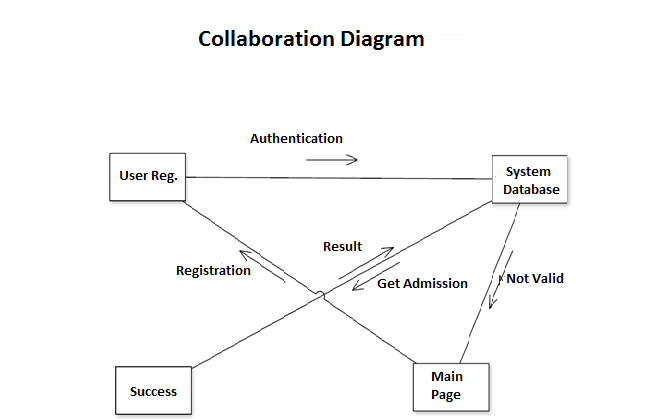
Sequence diagram and collaboration diagram are called INTERACTION DIAGRAMS. An interaction diagram shows an interaction, consisting of set of objects and their relationship including the messages that may be dispatched among them.

A sequence diagram is an introduction that identify with the time ordering of messages. Graphically a sequence diagram is a table that shows objects arranged along the X-axis and messages ordered in increasing time along the Y-axis



* **COLLABORATION DIAGRAM**

A collaboration diagram is an introduction diagram that emphasizes the structural organization of the objects that send and receive messages. Graphically a collaboration diagram is a collection of vertices and arcs.



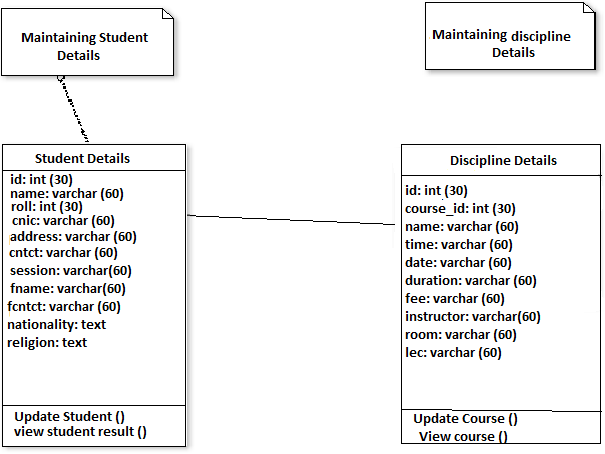
* **CLASS DIAGRAM:**

Class is nothing but a structure that contains both variables and methods. The Class Diagram shows a set of classes, interfaces, and collaborations and their relating ships. There is most common diagram in modeling the object oriented systems and are used to give the static view of a system. It shows the dependency between the classes that can be used in our system.

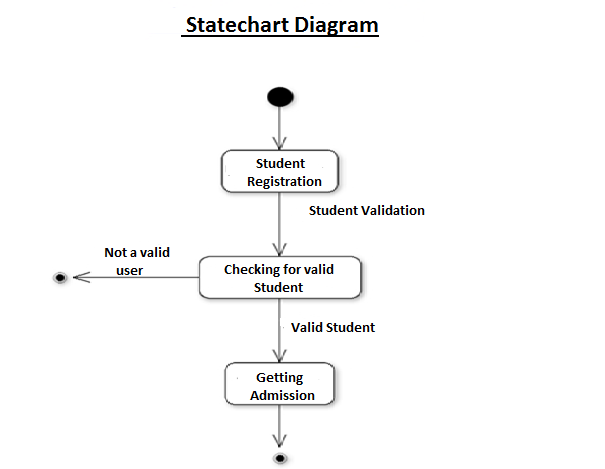
The interactions between the modules or classes of our projects are shown below. Each block contains Class Name, Variables and Methods.

**CLASS:**

A description of set of objects that share the same attributes, operations, relationships, and semantics



* **State Chart Diagram**



* **DATA FLOW DIAGRAMS**:

The DFD takes an input-process-output view of a system i.e. data objects flow into the software, are transformed by processing elements, and resultant data objects flow out of the software.

Data objects represented by labeled arrows and transformation are represented by circles also called as bubbles. DFD is presented in a hierarchical fashion i.e. the first data flow model represents the system as a whole. Subsequent DFD refine the context diagram (level 0 DFD), providing increasing details with each subsequent level.

The DFD enables the software engineer to develop models of the information domain & functional domain at the same time. As the DFD is refined into greater levels of details, the analyst perform an implicit functional decomposition of the system. At the same time, the DFD refinement results in a corresponding refinement of the data as it moves through the process that embody the applications.

A context-level DFD for the system produce information for use by the system and consume information generated by the system. The labeled arrow represents data objects or object hierarchy.

**RULES FOR DFD:**

* Fix the scope of the system by means of context diagrams.
* Organize the DFD so that the main sequence of the actions
* Reads left to right and top to bottom.
* Identify all inputs and outputs.
* Identify and label each process internal to the system with Rounded circles.
* A process is required for all the data transformation and Transfers. Therefore, never connect a data store to a data Source or the destinations or another data store with just a Data flow arrow.
* Do not indicate hardware and ignore control information.
* Make sure the names of the processes accurately convey everything the process is done.
* There must not be unnamed process.
* Indicate external sources and destinations of the data, with Squares.
* Number each occurrence of repeated external entities.
* Identify all data flows for each process step, except simple Record retrievals.
* Label data flow on each arrow.
* Use details flow on each arrow.
* Use the details flow arrow to indicate data movements.

**DATAFLOW DIAGRAMS:**

**Database:**

**Student**

**Registration**

**Take**

**Admission**

**Database**

**Student registration**

**Student registration**

**View Student**

**Details**

**Search for Student Details**

**Update Student Details**

**Register**

**Student**

**Taking Admission**

**Taking Admission**

**Take Admission**

**Sign in**

**Sign up**

**E-R Diagrams:**

The Entity-Relationship (ER) model was originally proposed by Peter in 1976 [Chen76] as a way to unify the network and relational database views. Simply stated the ER model is a conceptual data model that views the real world as entities and relationships. A basic component of the model is the Entity-Relationship diagram which is used to visually represent data objects. Since Chen wrote his paper the model has been extended and today it is commonly used for database design, the utility of the ER model is:

* It maps well to the relational model. The constructs used in the ER model can easily be transformed into relational tables.
* It is simple and easy to understand with a minimum of training. Therefore, the model can be used by the database designer to communicate the design to the end user.
* In addition, the model can be used as a design plan by the database developer to implement a data model in a specific database management software.

## Connectivity and Cardinality

The basic types of connectivity for relations are: one-to-one, one-to-many, and many-to-many. A *one-to-one* (1:1) relationship is when at most one instance of a entity A is associated with one instance of entity B. For example, "employees in the company are each assigned their own office. For each employee there exists a unique office and for each office there exists a unique employee.

A one-to-many(1:N) relationships is when for one instance of entity A, there are zero, one, or many instances of entity B, but for one instance of entity B, there is only one instance of entity A. An example of a 1: N relationships is

A department has many employees

Each employee is assigned to one department

A many-to-many(M:N) relationship, sometimes called non-specific, is when for one instance of entity A, there are zero, one, or many instances of entity B and for one instance of entity B there are zero, one, or many instances of entity A. The connectivity of a relationship describes the mapping of associated

## ER Notation

There is no standard for representing data objects in ER diagrams. Each modeling methodology uses its own notation. The original notation used by Chen is widely used in academics texts and journals but rarely seen in either CASE tools or publications by non-academics. Today, there are a number of notations used, among the more common are Bachman, crow's foot, and IDEFIX.

All notational styles represent entities as rectangular boxes and relationships as lines connecting boxes. Each style uses a special set of symbols to represent the cardinality of a connection. The notation used in this document is from Martin. The symbols used for the basic ER constructs are:

* **Entities** are represented by labeled rectangles. The label is the name of the entity. Entity names should be singular nouns.
* **Relationships** are represented by a solid line connecting two entities. The name of the relationship is written above the line. Relationship names should be verbs
* **Attributes**, when included, are listed inside the entity rectangle. Attributes which are identifiers are underlined. Attribute names should be singular nouns.
* **Cardinality** of many is represented by a line ending in a crow's foot. If the crow's foot is omitted, the cardinality is one.
* **Existence** is represented by placing a circle or a perpendicular bar on the line. Mandatory existence is shown by the bar (looks like a 1) next to the entity for an instance is required. Optional existence is shown by placing a circle next to the entity that is optional

### 

### Modules

### SCHOOL MANAGEMENT SYSTEM

### MODULES:

### ADMIN MODULE

### STUDENT MODULE

### ADMIN MODULE:

### REGISTER

### LOGIN

### CHANGE PASSWORD&FORGOTPASSWORD

### STUDENT -MODIFING DETAILS

### COURSE-ENTERING/MODIFYING DETAILS

### INSTRUCTOR DETAILS-MODIFYING DETAILS

* + 1. EMPLOYEE DETAILS-MODIFYING DETAILS

1. **REGISTER**:

### To be authenticated, first have to be registered.

1. **LOGIN:**

### The Registered user can be allowed to view inner details for which he permitted

### CHANGE PASSWORD&FORGOTPASSWORD:

### User has rights to modify his login details & also be informed through mails if he is unable to login.

1. **STUDENT -MODIFING DETAILS:**

Details can be modified to change status of each User.

1. COURSE-ENTERING/MODIFYING DETAILS:

### New courses adding and old courses deletions are done by the admin.

1. INSTRUCTOR DETAILS-MODIFYING DETAILS:

Admin can add or delete Instructors for specific platforms.

1. **EMPLOYEE DETAILS-MODIFYING DETAILS:**

Admin can add or delete employee for specific platforms.

## STUDENT MODULE:

### REGISTER

### LOGIN

### TAKE ADMISSION

### SEE ADMISSION DETAILS

### LOGOUT

### REGISTER:

### To be authenticated first have to be registered

1. **LOGIN:**

### The Registered User Can be allowed to view inner details for which he permitted

1. TAKE ADMISSION-

### The registered student allowed to get the admission

1. SEE ADMISSION DETAILS:

### Student can view admission details.

### V. LOGOUT:

### After the process of admission he turned to Logout page.

# OVERVIEW OF TECHNOLOGIES USED

## PHP

## PHP: Hypertext Preprocessor, is a widely used, general-purpose [scripting language](http://en.wikipedia.org/wiki/Scripting_language) that was originally designed for [web development](http://en.wikipedia.org/wiki/Web_development), to produce [dynamic web pages](http://en.wikipedia.org/wiki/Dynamic_web_page). It can be embedded into [HTML](http://en.wikipedia.org/wiki/HTML) and generally runs on a [web server](http://en.wikipedia.org/wiki/Web_server), which needs to be configured to process PHP code and create [web page](http://en.wikipedia.org/wiki/Web_page) content from it. It can be deployed on most web servers and on almost every [operating system](http://en.wikipedia.org/wiki/Operating_system) and [platform](http://en.wikipedia.org/wiki/Platform_(computing)) free of charge.

## PHP was originally created by [Rasmus Lerdorf](http://en.wikipedia.org/wiki/Rasmus_Lerdorf) in [1995](http://en.wikipedia.org/wiki/1995) and has been in continuous development ever since. The main implementation of PHP is now produced by The PHP Group and serves as the [de facto standard](http://en.wikipedia.org/wiki/De_facto_standard) for PHP as there is no [formal specification](http://en.wikipedia.org/wiki/Formal_specification). PHP is [free software](http://en.wikipedia.org/wiki/Free_software) released underthe [PHP License](http://en.wikipedia.org/wiki/PHP_License), which is incompatible with the [GNU General Public License](http://en.wikipedia.org/wiki/GNU_General_Public_License) (GPL) because of restrictions on the use of the term PHP

## PHP has evolved to include a [command line interface](http://en.wikipedia.org/wiki/Command_line_interface) capability and can also be used in [standalone](http://en.wikipedia.org/wiki/Standalone_software) [graphical applications](http://en.wikipedia.org/wiki/Graphical_user_interface).

**USAGE**

PHP is a general-purpose scripting language that is especially suited for [web development](http://en.wikipedia.org/wiki/Web_development). PHP generally runs on a [web server](http://en.wikipedia.org/wiki/Web_server). Any PHP code in a requested file is [executed](http://en.wikipedia.org/wiki/Execution_(computing)) by the PHP runtime, usually to create [dynamic web page](http://en.wikipedia.org/wiki/Dynamic_web_page) content. It can also be used for [command-line](http://en.wikipedia.org/wiki/Command-line) scripting and [client-side](http://en.wikipedia.org/wiki/Client-side) [GUI](http://en.wikipedia.org/wiki/Graphical_user_interface) applications. PHP can be deployed on most [web servers](http://en.wikipedia.org/wiki/Web_server), many [operating systems](http://en.wikipedia.org/wiki/Operating_system) and [platforms](http://en.wikipedia.org/wiki/Platform_(computing)), and can be used with many [relational database management systems](http://en.wikipedia.org/wiki/Relational_database_management_system). It is available free of charge, and the PHP Group provides the complete source code for users to build, customize and extend for their own use.

PHP primarily acts as a [filter](http://en.wikipedia.org/wiki/Filter_(software)), taking input from a file or stream containing text and/or PHP instructions and outputs another stream of data; most commonly the output will be HTML. Since PHP 4, the PHP [parser](http://en.wikipedia.org/wiki/Parser) [compiles](http://en.wikipedia.org/wiki/Compiler) input to produce [byte code](http://en.wikipedia.org/wiki/Bytecode) for processing by the [Zend Engine](http://en.wikipedia.org/wiki/Zend_Engine), giving improved performance over its [interpreter](http://en.wikipedia.org/wiki/Interpreter_(computing)) predecessor

Originally designed to create dynamic web pages, PHP now focuses mainly on [server-side scripting](http://en.wikipedia.org/wiki/Server-side_scripting), and it is similar to other server-side scripting languages that provide dynamic content from a web server to a [client](http://en.wikipedia.org/wiki/Client_(computing)), such as [Microsoft](http://en.wikipedia.org/wiki/Microsoft)'s [Active Server Pages](http://en.wikipedia.org/wiki/Active_Server_Pages), [Sun Microsystems](http://en.wikipedia.org/wiki/Sun_Microsystems)' [Java Server Pages](http://en.wikipedia.org/wiki/JavaServer_Pages) and [mod\_perl](http://en.wikipedia.org/wiki/Mod_perl). PHP has also attracted the development of many [frameworks](http://en.wikipedia.org/wiki/Software_framework) that provide building blocks and a design structure to promote [rapid application development](http://en.wikipedia.org/wiki/Rapid_application_development) (RAD). Some of these include [CakePHP](http://en.wikipedia.org/wiki/CakePHP), [Symfony](http://en.wikipedia.org/wiki/Symfony), [CodeIgniter](http://en.wikipedia.org/wiki/CodeIgniter), and [Zend Framework](http://en.wikipedia.org/wiki/Zend_Framework), offering features similar to other [web application frameworks](http://en.wikipedia.org/wiki/List_of_web_application_frameworks).

**HTML**

**HTML**, which stands for **Hyper Text Markup Language**, is the predominant [markup language](http://en.wikipedia.org/wiki/Markup_language) for [web pages](http://en.wikipedia.org/wiki/Web_page). It provides a means to create [structured documents](http://en.wikipedia.org/wiki/Structured_document) by denoting structural [semantics](http://en.wikipedia.org/wiki/Semantic) for text such as headings, paragraphs, lists etc. as well as for links, quotes, and other items. It allows [images and objects](http://en.wikipedia.org/wiki/HTML_element#Images_and_objects) to be embedded and can be used to create [interactive forms](http://en.wikipedia.org/wiki/HTML_element#Forms). It is written in the form of [HTML elements](http://en.wikipedia.org/wiki/HTML_element) consisting of "tags" surrounded by [angle brackets](http://en.wikipedia.org/wiki/Brackets#Angle_brackets_or_chevrons_.3C_.3E) within the web page content. It can include or can load [scripts](http://en.wikipedia.org/wiki/Scripting_language) in languages such as [JavaScript](http://en.wikipedia.org/wiki/JavaScript) which affect the behavior of HTML processors like [Web browsers](http://en.wikipedia.org/wiki/Web_browser); and [Cascading Style Sheets](http://en.wikipedia.org/wiki/Cascading_Style_Sheets) (CSS) to define the appearance and layout of text and other material. The [W3C](http://en.wikipedia.org/wiki/W3C), maintainer of both HTML and CSS standards, encourages the use of CSS over explicit presentational markup.

Hyper Text Markup Language (HTML) is the encoding scheme used to create and format a web document. A user need not be an expert programmer to make use of HTML for creating hypertext documents that can be put on the internet.

Most graphical [e-mail](http://en.wikipedia.org/wiki/E-mail) clients allow the use of a subset of HTML (often ill-defined) to provide formatting and [semantic](http://en.wikipedia.org/wiki/Semantic_web) markup not available with [plain text](http://en.wikipedia.org/wiki/Plain_text). This may include typographic information like colored headings, emphasized and quoted text, inline images and diagrams. Many such clients include both a [GUI](http://en.wikipedia.org/wiki/GUI) editor for composing HTML e-mail messages and a rendering engine for displaying them. Use of HTML in e-mail is controversial because of compatibility issues, because it can help disguise [phishing](http://en.wikipedia.org/wiki/Phishing) attacks, because it can confuse [spam](http://en.wikipedia.org/wiki/E-Mail_spam) filters and because the message size is larger than plain text.

NAMING CONVENTIONS

The most common [filename extension](http://en.wikipedia.org/wiki/Filename_extension) for [files](http://en.wikipedia.org/wiki/Computer_file) containing HTML is .html. A common abbreviation of this is .htm, which originated because some early operating systems and file systems, such as [DOS](http://en.wikipedia.org/wiki/DOS) and [FAT](http://en.wikipedia.org/wiki/File_Allocation_Table), limited file extensions to [three letters](http://en.wikipedia.org/wiki/8.3_filename).

**HTML APPLICATION**

An HTML Application is a [Microsoft Windows](http://en.wikipedia.org/wiki/Microsoft_Windows) application that uses HTML and Dynamic HTML in a browser to provide the application's graphical interface. A regular HTML file is confined to the security model of the web browser, communicating only to web servers and manipulating only webpage objects and [site cookies](http://en.wikipedia.org/wiki/HTTP_cookie). An HTA runs as a fully trusted application and therefore has more privileges, like creation/editing/removal of files and [Windows Registry](http://en.wikipedia.org/wiki/Windows_Registry) entries. Because they operate outside the browser's security model, HTAs cannot be executed via HTTP, but must be downloaded (just like an [EXE file](http://en.wikipedia.org/wiki/EXE)) and executed from local file system

**JAVASCRIPT**

JavaScript is an [object-oriented](http://en.wikipedia.org/wiki/Object-oriented) [scripting language](http://en.wikipedia.org/wiki/Scripting_language) used to enable [programmatic](http://en.wikipedia.org/wiki/Computer_programming) access to objects within both the [client application](http://en.wikipedia.org/wiki/Client_(computing)) and other [applications](http://en.wikipedia.org/wiki/Application_software). It is primarily used in the form of [client-side JavaScript](http://en.wikipedia.org/wiki/Client-side_JavaScript), implemented as an integrated component of the [web browser](http://en.wikipedia.org/wiki/Web_browser), allowing the development of enhanced [user interfaces](http://en.wikipedia.org/wiki/User_interface) and dynamic [websites](http://en.wikipedia.org/wiki/Website). JavaScript is a [language](http://en.wikipedia.org/wiki/Programming_language_dialect) of the [ECMAScript](http://en.wikipedia.org/wiki/ECMAScript) standard and is characterized as a [dynamic](http://en.wikipedia.org/wiki/Dynamic_language), [weakly typed](http://en.wikipedia.org/wiki/Weak_typing), [prototype-based](http://en.wikipedia.org/wiki/Prototype-based_programming) language with [first-class functions](http://en.wikipedia.org/wiki/First-class_function). JavaScript was influenced by many languages and was designed to look like [Java](http://en.wikipedia.org/wiki/Java_(programming_language)), but to be easier for non-programmers to work with.

**PROTOTYPE-BASED**

JavaScript uses [prototypes](http://en.wikipedia.org/wiki/Prototype-based_programming) instead of [classes](http://en.wikipedia.org/wiki/Class_(computer_science)) for [inheritance](http://en.wikipedia.org/wiki/Inheritance_(computer_science)). It is possible to simulate many class-based features with prototypes in JavaScript.

Functions double as object constructors along with their typical role. Prefixing a function call with new creates a new object and calls that function with its local this keyword bound to that object for that invocation. The constructor's prototype property determines the object used for the new object's internal prototype. JavaScript's built-in constructors, such as Array, also have prototypes that can be modified.

Unlike many object-oriented languages, there is no distinction between a function definition and a [method](http://en.wikipedia.org/wiki/Method_(computer_science)) definition. Rather, the distinction occurs during function calling; a function can be called as a method. When a function is called as a method of an object, the function's local this keyword is bound to that object for that invocation.

**USAGE**

The primary use of JavaScript is to write functions that are embedded in or included from [HTML](http://en.wikipedia.org/wiki/HTML) pages and interact with the [Document Object Model](http://en.wikipedia.org/wiki/Document_Object_Model) (DOM) of the page. Because JavaScript code can run locally in a user's browser (rather than on a remote server) it can respond to user actions quickly, making an application feel more responsive. Furthermore, JavaScript code can detect user actions which HTML alone cannot, such as individual keystrokes. Applications such as [Gmail](http://en.wikipedia.org/wiki/Gmail) take advantage of this: much of the user-interface logic is written in JavaScript, and JavaScript dispatches requests for information (such as the content of an e-mail message) to the server. The wider trend of [Ajax programming](http://en.wikipedia.org/wiki/AJAX) similarly exploits this strength.

A [JavaScript engine](http://en.wikipedia.org/wiki/JavaScript_engine) (also known as *JavaScript interpreter* or *JavaScript implementation*) is an [interpreter](http://en.wikipedia.org/wiki/Interpreter_(computing)) that interprets JavaScript [source code](http://en.wikipedia.org/wiki/Source_code) and executes the [script](http://en.wikipedia.org/wiki/Computer_program) accordingly.

The first JavaScript engine was created by [Brendan Eich](http://en.wikipedia.org/wiki/Brendan_Eich) at Netscape Communications Corporation, for the [Netscape Navigator](http://en.wikipedia.org/wiki/Netscape_Navigator) [web browser](http://en.wikipedia.org/wiki/Web_browser). A web browser is by far the most common host environment for JavaScript. Web browsers typically use the public [API](http://en.wikipedia.org/wiki/Application_programming_interface) to create "host objects" responsible for reflecting the [DOM](http://en.wikipedia.org/wiki/Document_Object_Model) into JavaScript.

**ABOUT MySQL**

## MySQL Introduction

## There are a large number of database management systems currently available, some commercial and some free. Some of them: Oracle, Microsoft Access, MySQL and PostgreSQL.

## These database systems are powerful, feature-rich software, capable of organizing and searching millions of records at very high speeds.

### Understanding Databases, Records, and Primary Keys

Every Database is composed of one or more tables. These Tables, which structure data into rows and columns, impose organization on the data. The records in a table (below) are not arranged in any particular order. To make it easy to identify a specific record, therefore, it becomes necessary.

### Standing Relationships and Foreign Keys (RDBMS)

You already know that a single database can hold multiple tables. In a Relational database management system (RDBMS), these tables can be linked to each other by one or more common fields, called ****foreign keys****.

### What is Database administrator (DBA)?

### Database administrator is the super user of database, he has unrestricted rights and privileges to access database, grant permission to other database users.

### What is Database user (DBU)?

### Database user is the person who uses the database in a restricted privileges, provided by database administrator.

### Download MySQL Database

### If you have installed PHP’s WAMP or XAMPP server, then MySQL database already exists. If you don’t have then download MySQL database from here [http://www.mysql.com](http://www.phptpoint.com/mysql/)

**DATABASE TABLES:**

**User Registration Table**

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Type | Null | Key |
| id | int (30) | No | PRIMARY KEY |
| name | Varchar (60) | Yes |  |
| email | Varchar (60) | Yes |  |
| password | Varchar (60) | Yes |  |

**Admin Table**

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Type | Null | Key |
| id | int (30) | No | PRIMARY KEY |
| username | Varchar (60) | Yes |  |
| password | Varchar (60) | Yes |  |

**Discipline Table**

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Type | Null | Key |
| id | int (30) | No | PRIMARY KEY |
| name | Varchar (60) | Yes |  |
| roll | int (30) | Yes |  |
| cnic | Varchar (60) | Yes |  |
| address | Varchar (60) | Yes |  |
| cntct | Varchar (60) | Yes |  |
| session | Varchar (60) | Yes |  |
| fname | Varchar (60) | Yes |  |
| fcntct | Varchar (60) | Yes |  |
| nationality | text | Yes |  |
| religion | text | Yes |  |

**Course Table**

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Type | Null | Key |
| id | int (30) | No | PRIMARY KEY |
| course\_id | Varchar (60) | Yes |  |
| name | Varchar (60) | Yes |  |
| time | Varchar (60) | Yes |  |
| date | Varchar (60) | Yes |  |
| duration | Varchar (60) | Yes |  |
| fee | Varchar (60) | Yes |  |
| instructor | Varchar (60) | Yes |  |
| room | Varchar (60) | Yes |  |
| lec | Varchar (60) | Yes |  |

**Faculty Table**

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Type | Null | Key |
| id | int (30) | No | PRIMARY KEY |
| name | Text | Yes |  |
| cnic | Varchar (60) | Yes |  |
| addr | Varchar (60) | Yes |  |
| contact | Varchar (60) | Yes |  |
| email | Varchar (60) | Yes |  |
| quali | Varchar (60) | Yes |  |
| relig | Text | Yes |  |
| nationality | Text | Yes |  |
| joining | Varchar (60) | Yes |  |
| sal | Varchar (60) | Yes |  |
| post | Varchar (60) | Yes |  |

**Employee Table**

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Type | Null | Key |
| id | int (30) | No | PRIMARY KEY |
| name | Text | Yes |  |
| cnic | Varchar (60) | Yes |  |
| addr | Varchar (60) | Yes |  |
| contact | Varchar (60) | Yes |  |
| email | Varchar (60) | Yes |  |
| desig | Varchar (60) | Yes |  |
| hours | Varchar (60) | Yes |  |
| year | Varchar (60) | Yes |  |
| salary | Varchar (60) | Yes |  |
| religion | Text | Yes |  |
| nationality | Text | Yes |  |
| assistant\_to | Text | Yes |  |
| shift | Text | Yes |  |

**Admission Table**

|  |  |  |  |
| --- | --- | --- | --- |
| Column | Type | Null | Key |
| id | int (30) | No | PRIMARY KEY |
| surname | Varchar (60) | Yes |  |
| name | Varchar (60) | Yes |  |
| fname | Varchar (60) | Yes |  |
| prof | Varchar (60) | Yes |  |
| gender | Varchar (60) | Yes |  |
| cnic | Varchar (60) | Yes |  |
| email | Varchar (60) | Yes |  |
| day | int (30) | Yes |  |
| month | Varchar (60) | Yes |  |
| year | int (30) | Yes |  |
| address | Varchar (60) | Yes |  |
| city | Varchar (60) | Yes |  |
| phone | Varchar (60) | Yes |  |
| mobile | Varchar (60) | Yes |  |
| degree | Varchar (60) | Yes |  |
| marks | Varchar (60) | Yes |  |
| percent | Varchar (60) | Yes |  |
| college | Varchar (60) | Yes |  |
| passing | Varchar (60) | Yes |  |
| nat | Varchar (60) | Yes |  |
| amount | Varchar (60) | Yes |  |
| deposit | Varchar (60) | Yes |  |
| bank | Varchar (60) | Yes |  |
| details | Varchar (60) | Yes |  |

**FEASIBILITY STUDY:**

The data collection that occurs during preliminary investigations examines system feasibility, the likelihood that the system will be beneficial to the organization. Three tests of feasibility studies are: technical, economical and operational. All are equally important.

The system has been tested for feasibility in the following points.

1. Technical Feasibility

2. Economical Feasibility

3. Operational Feasibility.

1. **Technical Feasibility:**

It involves determining whether or not a system can actually be constructed to solve the problem at hand. Some users expect too much of computers, assuming that computers can accurately predict the future, immediately reflect all information in an organization, easily understand speech, or figure out how to handle difficult problems. Such systems, even if they exist, are not yet available for widespread use.

The technical issues raised during the feasibility stage of the investigation are:

1. Does the necessary technology exist (can it be acquired) to do what is suggested?
2. Does the proposed equipment have the technical capacity to hold the data required to use the new system?
3. Will the proposed system and components provide adequate responses to inquiries, regardless of the number or location of users?
4. Can the system be expanded, if developed?
5. Are there technical guarantees of accuracy, reliability, ease of access and data security

For example, if the proposal includes a printer that prints at the rate of 2,000 lines per minute, a brief search shows that this is technically feasible. Whether it should be included in the configuration because of its cost is an economic decision. On the other hand, if a user is requesting audio input to write, read, and change stored data, the proposal may not be technically feasible.

1. **Economical Feasibility:**

It involves estimating benefits and costs. These benefits and costs may be tangible or intangible. Because of confusion between the types of costs, it is sometimes very difficult to decide if the benefits compensate the costs.

Tangible benefits may include decreasing salary costs (by automating manual procedures), preventing costly but frequent errors, sending bills earlier in the month, and increasing control over inventory levels. Such benefits may be directly estimated in rupees without much trouble. Intangible benefits may include increasing quality of goods produced, upgrading or creating new customer services, reducing repetitive or monotonous work for employees, and developing a better understanding of the market. Such benefits may be much more important than tangible benefits, but they may be ignored because estimating their rupee values involves pure guesswork.

Tangible costs are easily estimated. They include the one-time cost of developing the system and the continuous costs of operating the system. Examples of development costs are the salaries of programmers and` analysts, the prices of the computer equipment, and the expenses connected with user training. Operating costs include the salaries of computer operators and the costs of computer time and computer supplies. Intangible costs are usually not discussed because they are rarely large. Examples of such costs include those associated with early user dissatisfaction and with the problems of converting to the new system.

A system that can be developed technically and will be used if installed must still be a good investment. That is, financial benefits must equal or exceed the financial costs. The economic and financial questions raised by analysts during the preliminary investigation seek estimates of:

1. The cost to conduct a full systems investigation.
2. The cost of hardware and software for the class of application being considered.
3. The benefits in the form of reduced costs or fewer costly errors.
4. The cost if nothing changes (the system is not developed)

Cost and benefit estimates on each project provide a basis for determining which projects are most worthy of consideration. Each estimate can be analyzed to determine how rapidly costs are recovered by benefits, to calculate both the absolute and interest-adjusted amounts of excess benefits, and to establish the ratio of benefits to costs. All of these factors are considered when developing an overall sense of the projects’ economic feasibility.

To be judged feasible, a project proposal must pass all these tests. Otherwise, it is not a feasible project. For example, a personnel record system that is financially feasible and operational attractive, is not feasible if the necessary technology does not exist. Or a medical system which can be developed at reasonable cost but which nurses will avoid using cannot be judged operationally feasible.

1. **Operational Feasibility:**

Proposed projects are of course beneficial only if they can be turned into information systems that will meet the organization's operation requirements. Simply stated, this test of feasibility asks if the system will work when developed and installed. Are there major barriers to implementation? Here are questions that will help test the operational feasibility of a project:

1. Is there sufficient support for the project from the management and from users? If the current system is well liked and used to the extent that persons will not see reasons for a change, there may be resistance.
2. Are current business methods acceptable to the user? If they are not, user may welcome a change that will bring about a more operational and useful system.
3. Have the users been involved in the planning and development of the project? Early involvement reduces the chances of resistance to the system and change in general, and increases the likelihood of successful projects.
4. Will the proposed system cause harm? The following questions are related to this issue:

* Will the system produce result in any respect or area?
* Will loss of control result in any area?
* Will accessibility of information be lost?
* Will individual performance be poorer after implementation than before?
* Will customers be affected in an undesirable way?
* Will it slow performance in any areas?

Operational feasibility is a measure of how people are able to work with the system. For example, a system may require managers to write BASIC, COBOL, or FORTRAN programs to access data. However, managers probably receive the greatest help from a system when they can concentrate on the problems to solve rather than on how programs should be constructed to solve them.

### 

### Implementation

**Implementation:**

Implementation is the stage where the theoretical design is turned into a working system. The most crucial stage in achieving a new successful system and in giving confidence on the new system for the users that it will work efficiently and effectively.

The system can be implemented only after thorough testing is done and if it is found to work according to the specification.

It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the changeover and an evaluation of change over methods a part from planning. Two major tasks of preparing the implementation are education and training of the users and testing of the system.

The more complex the system being implemented, the more involved will be the systems analysis and design effort required just for implementation.

The implementation phase comprises of several activities. The required hardware and software acquisition is carried out. The system may require some software to be developed. For this, programs are written and tested. The user then changes over to his new fully tested system and the old system is discontinued.

### 

### Testing

**SYSTEM TESTING**

It brings all the pieces together into a special testing environment, then checks for errors, bugs and interoperability. Software testing is the process of testing the software product. Effective software testing will contribute to the delivery of higher quality software products, more satisfied users, lower maintenance costs, more accurate, and reliable results. However, ineffective testing will lead to the opposite results; low quality products, unhappy users, increased maintenance costs, unreliable and inaccurate results.

Testing is the major quality control measure used during software development. Its basic function is to detect errors in the software. It is a very expensive process and consumes one-third to one-half of the cost of a typical development project. It is the process of executing program (or a part of a program) with the intention of finding the errors, however, testing cannot show the absence of errors it can show that errors are present.

“Errors are present within the software under test”. This cannot be the aim of software designers they must have designed the software with the aim of producing it with zero errors. Software testing is becoming increasingly important in the earlier part of the software development life cycle, aiming to discover errors before they are deeply embedded within systems. In the software development life cycle the earlier the errors are discovered and removed, the lower is the cost of their removal. The most damaging errors are those, which are not discovered during the testing process and therefore remain when the system ‘goes live’.

The testing requires the developers to find errors from their software. It is very difficult for software developer to point out errors from own creations. A good test is one that has a high probability of finding an as yet undiscovered error. A successful test case unearths an undiscovered error. This implies that testing not only has to uncover errors introduced during coding, but also errors introduced during the previous phases. The goal of testing is to uncover requirement, design, and coding errors in the programs. Different levels of testing are used:

**Unit testing:**

A module is tested separately and is often performed by the coder himself simultaneously along with the coding of the module. The purpose is to exercise the different parts of the modules code to detect coding errors.

**Integration Testing:**

The modules are gradually integrated into subsystems, which are then integrated to eventually from the entire system. Integration testing is performed to detect design errors by focusing on testing the interconnection between modules.

**System Testing:**

After the system is put together, it is performed. The system is tested against the system requirement to see if the entire requirement are met and if the system performs as specified by the requirement.

**Acceptance Testing:**

The final stage of initial development, where the software is put into production and runs actual business. It is performed to demonstrate to the client, on the real life data of the client, the operation of the system.

Testing is an extremely critical and time-consuming activity. It requires proper planning of the overall testing process. The test plan specifies conditions that should be tested, different units to be tested, and the manner in which the modules will be integrated together. The final output of the testing phase is the test report and the error report, or a set of such reports (one for each unit tested).

The importance of software testing and its implications with respect to S/W Quality cannot be overemphasized. Because of this importance & the large amount of project effort associated with the system development, it becomes quite necessary to become well planned and through testing. Inadequate testing & no-adequate testing lead's to errors that may be costly when they appear months later. Effective testing translates into cost savings from reduced errors & saves a lot of project efforts. It follows major factors that decide the occurrences of errors in a new design from the very early stage of the development.

1. **Communication between the user & the designer**

This factor is handled by frequently communicating with the finance department and the gate entry.

1. **The Time factor for the design**

This factor is handled by giving comparatively more time to the designing of the system.

**Objectives of System Testing**

Once a system has been designed, it is necessary to undergo an exhaustive testing before installing the system. This is important because in some cases a small error, not detected and corrected early before installation, may explode into a much large problem later on. Testing is being performed when users are asked to assist in identifying all possible situations. That might arise as regards the factor that efforts were put to tackle the problem under consideration. A plan was decided to be followed for testing the system. The complete testing procedure was divided into several steps, to be performed at different stages. Tests were to be done as follows: -

**Testing Criteria**

**A. White Box Testing**

1. **Transaction path Testing**

In this phase each and every condition within a unit program were tested. As and when a loop or condition statement was incorporated into a unit the loops were tested for correctness, for foundry conditions and for not getting into infinite execution cycle. The data used was whatever necessary at that instance. The path of each transaction from origin to destination was tested for reliable results.

1. **Module Testing**

This was carried out during the programming stage itself. Individual programs were tested at the time of coding and necessary changes are made there on to make sure that the modules in the form program, is working satisfactory as regards the expected output from the module. All aspects of the program viz. All choices available were properly tested.

1. **String Testing**

After loading all individual program string was performed for each one of programs where the output generated by one program is used as input by another program. This step was completed after making necessary changes wherever required.

**B. Black Box Testing**

1. **System Testing**

After module and string testing, the systems were tested as a whole system Tests were undertaken to check bundled modules for errors. The errors found in the couple system as a whole was corrected. A testing on the Actual data of the company followed this. During this phase the existing System and this package was running in parallel to enable us to verify and compare the result sets. The following criteria were used while testing the system.

1. **Output Testing**

No systems could be useful if it does not produced the required operation in the required format. The outputs generated or displayed by the system under consider was tested by asking the format required by them.

1. **User Acceptance Testing**

User acceptance of a system is a key factor for the success of any system. The system under consideration was tested for user acceptance by constantly keeping in touch with the prospected system users at the time of developing and making changes.

Wherever required this was done in regard to the user satisfaction.

**Testing Procedure**

Different type of checks like duplicate checks, completeness check, validity, checks etc. are incorporated in this system, as the data has to be entered in different forms.

The user is not familiar with new system the data entry screens are designed in such a way that they are

* Consistent
* Compatible
* Easy to use
* Had quick response

The following conventions are used while designing of the various screens to make the system user friendly

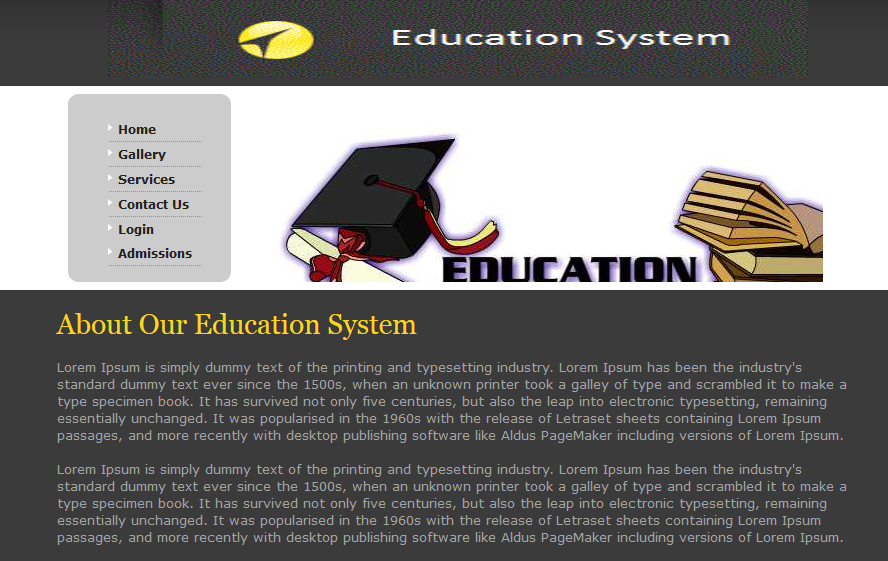
* All the items that are logically related are together.
* Error and validation messages are provided wherever required.
* System testing is against its initial objectives, it is done in a simulated environment.

**Test Review**

Test review is the process, which ensures that testing is carried out, as planned test review decides whether or not the program is ready to ship out for the implementation.

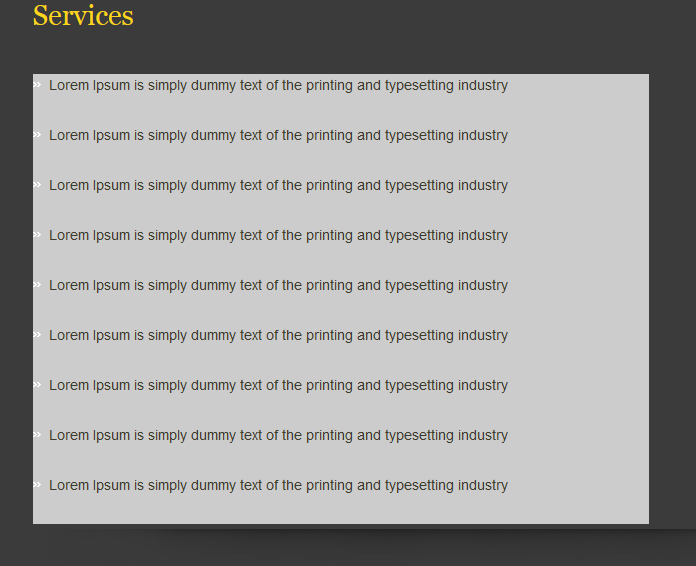
### 

### Output Screen

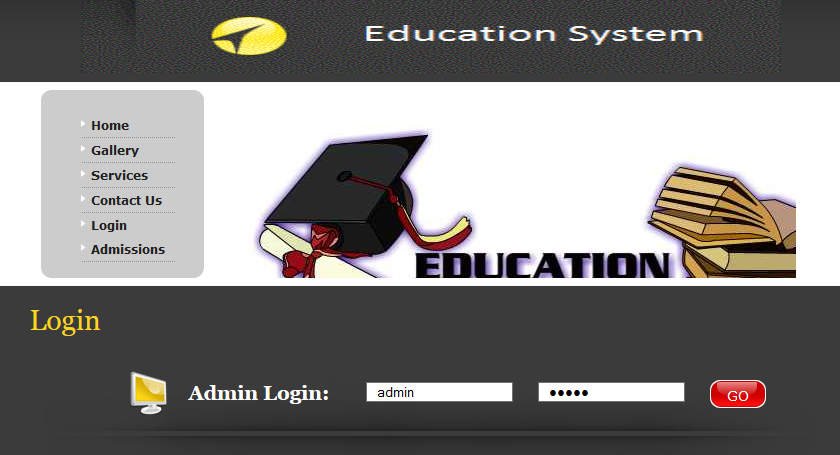


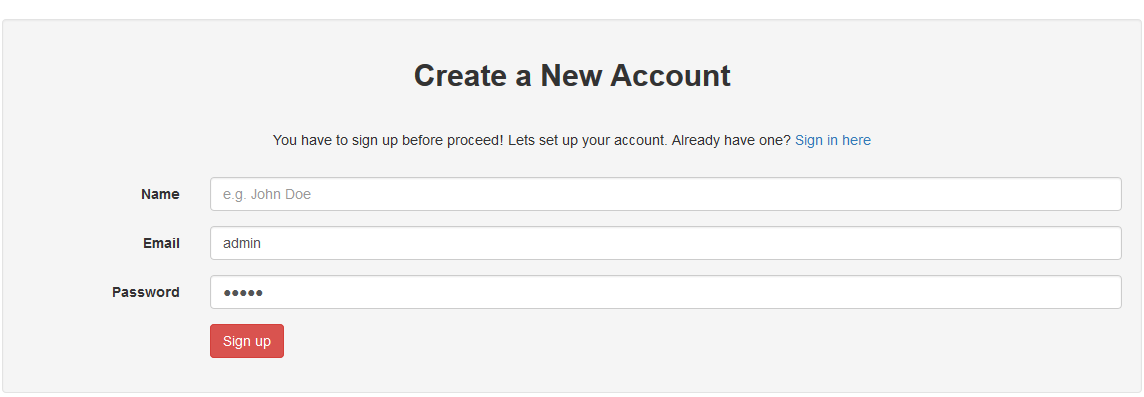


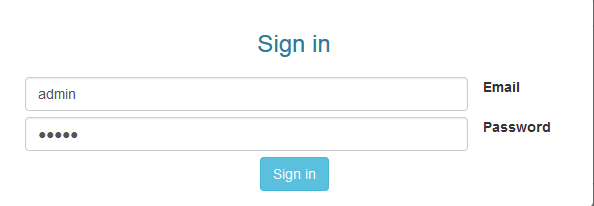


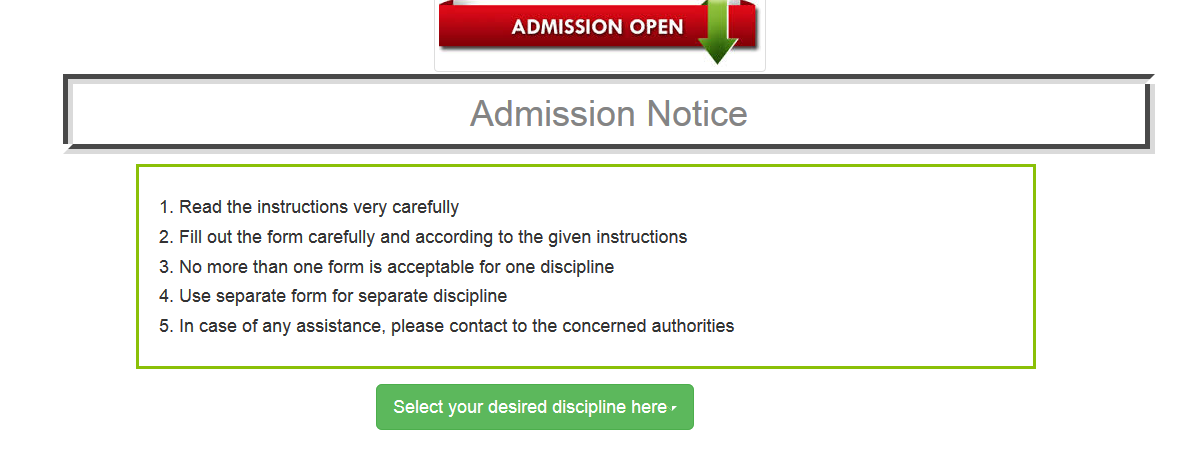


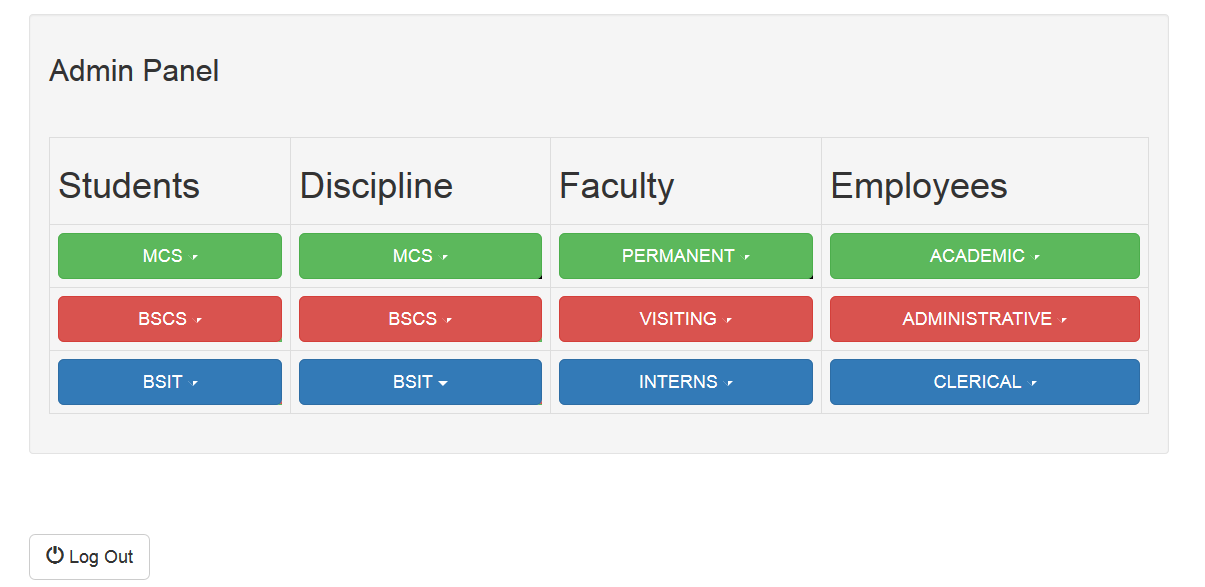


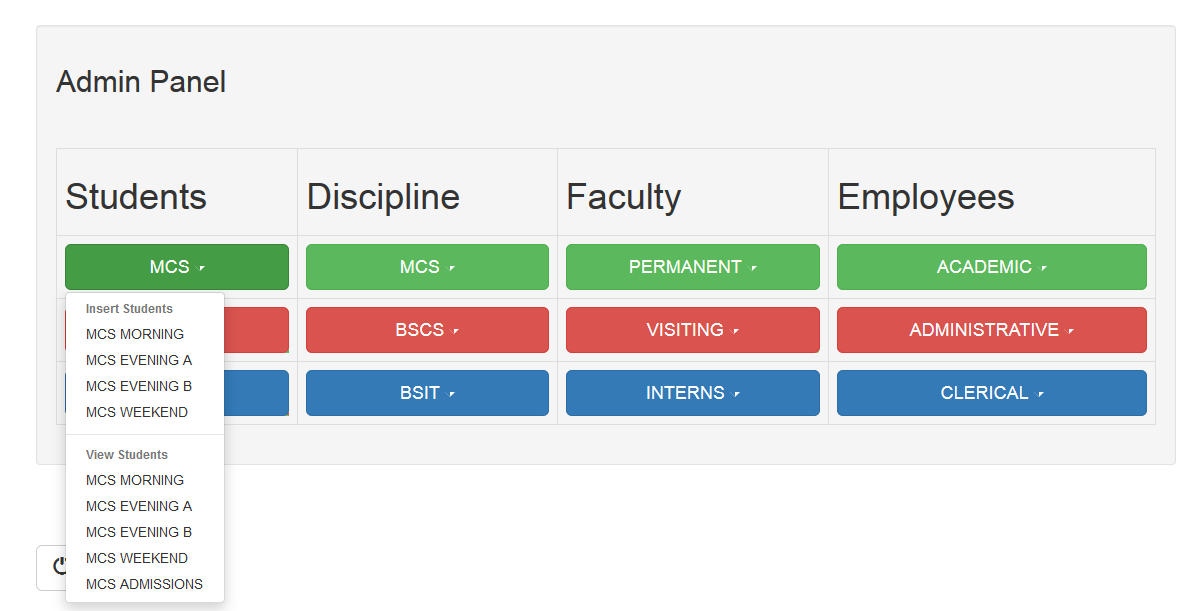


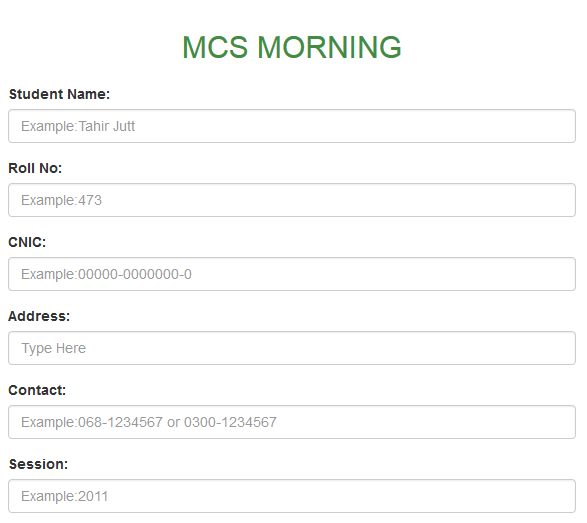


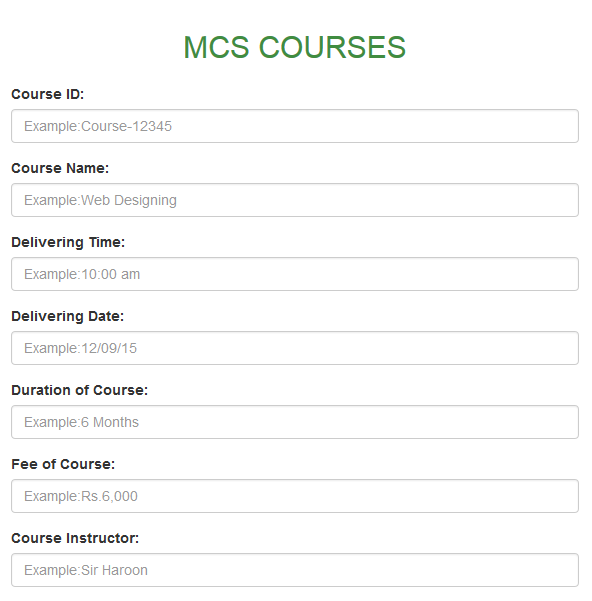


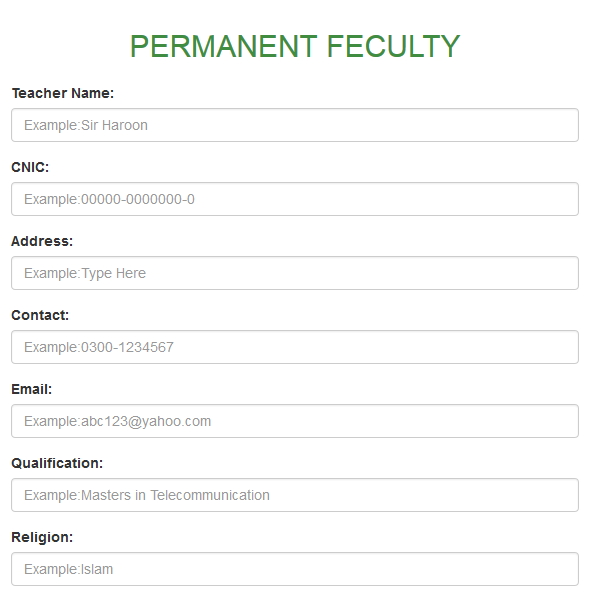


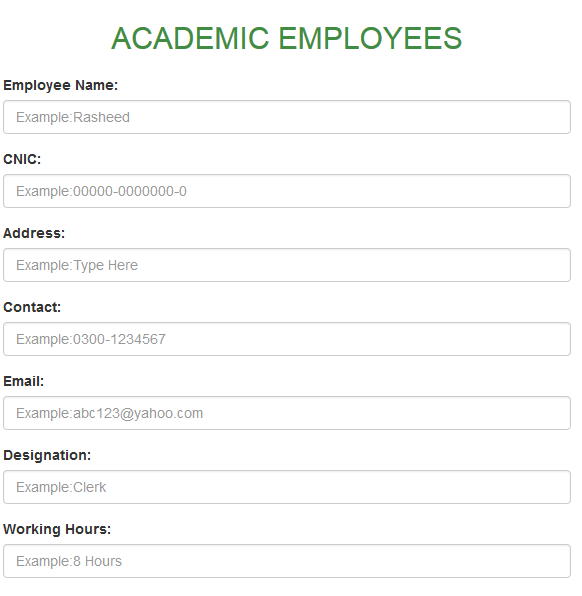


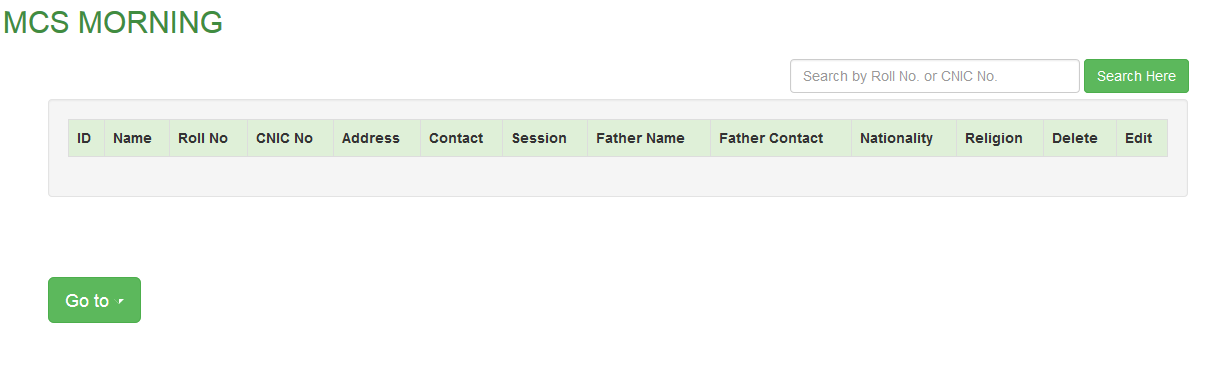


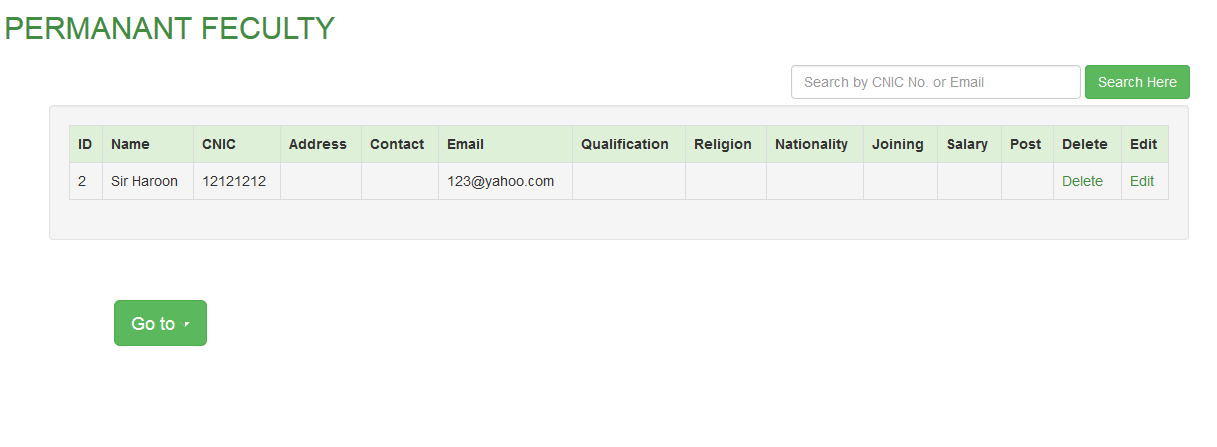


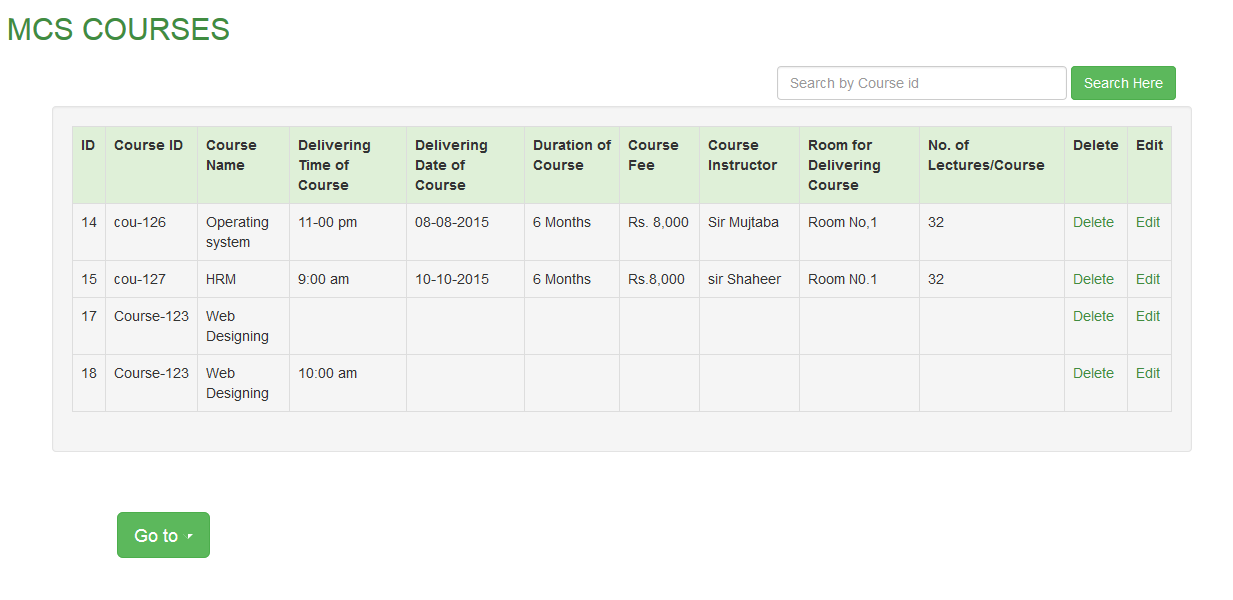


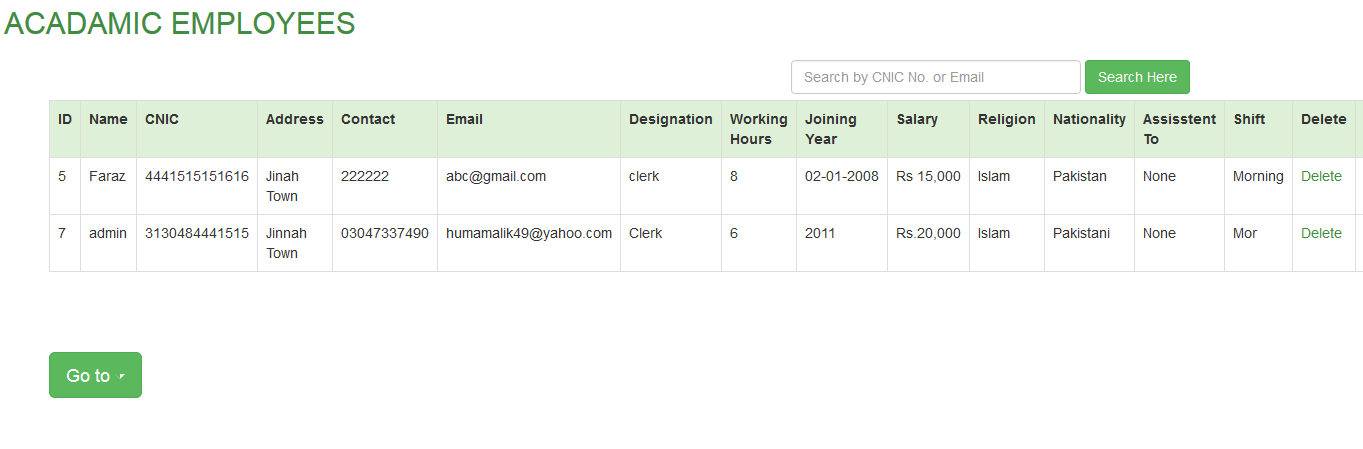












### 

### Conclusion

**CONCLUSION:**

The package was designed in such a way that future modifications can be done easily. The following conclusions can be deduced from the development of the project.

* Automation of the entire system improves the efficiency
* It provides a friendly graphical user interface which proves to be better when compared to the existing system.
* It gives appropriate access to the authorized users depending on their permissions.
* It effectively overcomes the delay in communications.
* Updating of information becomes so easier.
* System security, data security and reliability are the striking features.
* The System has adequate scope for modification in future if it is necessary.

### 

### Future

### Enhancement

**FUTURE ENHANCEMENTS:**

This application avoids the manual work and the problems concern with it. It is an easy way to obtain the information regarding the different scheduled examinations information that are currently issued.

Well I and my team members have worked hard in order to present an improved website better than the existing one’s regarding the information about the various activities. Still, we found out that the project can be done in a better way. Primarily, when we request information about a particular schedules it just shows the exam date and platform. So, after getting the information we can get access to the online exam.

The enhancement that we can add the searching option. We can directly search to the particular student details from this site.

### 

### Bibliography

**BIBLIOGRAPHY**

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

* **Books Referred:**
* BEGINNING PHP 5 ---DAVE MERCER
* BLACK BOOK HTML ---WILEY DREAMTECH
* PHP AND MYSQL WEB DEVELOPMENT --- LUKEWELLING,LAURA
* MICROSOFT SQL SERVER-2000 ---RANKIN, PAUL & JENSEN
* SQL SERVER-2000 ---DUSAN PETKOVIC
* PHP IN A NUTSHELL --- PAUL HUDSON
* **Websites Referred:**

http:// http://www.w3schools.com