CONTENTS

1. INTRODUCTION
2. MOTIVATION

* Objectives
* Scope

1. TECHNOLOGIES USED
2. MODULES USED

* Admin
* Teacher
* Student

1. SYSTEM DESIGN

* Data Flow Diagram
  + O-Level DFD
  + 1-Level DFD
* E-R Diagram

1. FUNCTIONAL REQUIREMENT
2. TESTING and VALIDATION
3. SCREEN LAYOUT
4. TABLES
5. CODING
6. IMPLEMENTATION
7. CONCLUSION
8. REFERENCE
9. BIBLIOGRAPHY

INTRODUCTION

This software has developed for daily attendance of students. It made easy to access the attendance information of a particular student. The information is stored through operators, and provided by teacher for related class. This software is helpful in evaluating the attendance eligibility of a student. Its purpose was to computerize the tradition way of taking attendance and generating of report automatically at the end or between of the session. This project has developed as a desktop application for a specific institute.

MOTIVATION

In most educational institutions the attendance is taken manually. It is not only time consuming, but it is also unsecure and unreliable and it can be lost. Some institutions are using punch card for attendance while this will be difficult for teachers to keep track of the large number of students because by using punch card, a student can help the other students or his/her friend to punch their card even the other student may be absent or come late in class, so it is not reliable. To overcome these problems I have developed a better system which is Web based; it is fully responsive where a user can use in mobile, tablets and different computer systems. In this system records are kept safe and secure and the attendance information of particular or all students of particular class can be accessed easily and without time consuming, the report is generated automatically.

OBJECTIVES

* Eliminate duplicate data entry and errors in time and attendance entries.
* Eliminate paperwork and save time.
* Automatic calculation of attendance
* To Increase security.

FEATURES

As soon as a teacher or a student joins the institution, he/she is requested to sign up to the online attendance management system. After this, admin is required to review and approve their account activation.  
As their account gets approved, they can access to their account.

SCOPE

The **scope** of the project is the **system** on which the **software** is installed, i.e. the project is developed as a desktop application, and it will work for a particular institute.

**TECHNOLOGY USED**

There are many technologies which we can use for developing software but we have used following:

**Project Category : Multimedia**

**Used Operating System : Windows 8**

**Front End :ASP.NET 4.5**

**Back End :SQL-SERVER 2005**

**Software & Hardware Requirement Specification**

**Software Requirement:**

**Server**

Browser : IE 6.0 or later

Database : MS SQL Server 2005

Web server : Internet Information Server (IIS) 7.5

Operating System : Windows

**Client**

Browser : IE 6.0 or any browser

Operating System : Any O.S. Windows/Linux/Solaris

**Developer**

Browser : IE 6.0 or any browser

IDE : Visual Studio 2010

Database : MS SQL Server 2005

Operating System : Windows 7

Web server : Internet Information Server (IIS) 7.5

Documentation tool : MS-Word, MS-PowerPoint

Designing tool : VS

Scripting language : C#

**Hardware Specification:**

**Server:**

Processor : 1 .0 (GHz) Pentium processor

RAM : 1 GB

HDD : 80GB

Display : 1024 x 768 High color-32-bit

**Client:**

Processor : P3 866 MHz or later

RAM : 512 MB

HDD : 40GB

Display : 1024 x 768 High color-32-bit SOFTWARE

**Developer:**

Processor : 1 .0 (GHz) Pentium processor

RAM : 1 GB

Display : 1024 x 768 High color-32-bit SOFTWARE

MODULES USED

ADMIN

* He/she is a super user who enjoys all the privileges .
* Add teacher account .
* View attendance of each student.
* Update Records

TEACHER

* Teacher can Choose/assigned Multiple lectures.
* Teacher can view students Attendance.
* View the list of students assigned to them.
* Mark attendance after each lecture
* Update Profile
* Upload Profile Pic
* Update Password(encrypted format)

STUDENT

* He/she needs to create an account.
* Students can Check Attendance.
* Can view their attendance from anywhere around the globe.
* Update Profile
* Upload Profile Pic
* Update Password(encrypted format)

**SYSTEM DESIGN**

DATA FLOW DIAGRAM

## Data flow diagram (DFD) was first develop by LARRY CONSTANTINE as way representation system requirements in a graphical form; this lead to modular design.

## A DFD describes what data flow (logical) rather than how they are processed , so it does not depend on hardware , software , data structure or file organization. It is also known as ‘bubble chart’. A data flow diagram is a structured analysis and design tool that can be used for flowcharting in place of, or in association with, information oriented and process oriented systems flowchart.

## The DFD serves two purposes:-

## To provide an indication of how data are transformed as they move through the data flow.

## To depict the function (and sub function) that transforms the data flow.

## A data flow diagram is graphical representation that depicts information flow and the transform that are applied as data move from input to output. The basic form of a data flow diagram ; also known as data flow graph or a bubble chart.

SYSMBOLS OF DFD

INPUT/OUTPUT

It is use for input/output process.

PROCESS

It is use for process of data.

DATABASE

It is use to store the data.

FLOW

It tells the flow of data.

EXTERNAL ENTITY

It is use to terminators.

0-LEVEL DFD

Login Login

Student

Admin

Teacher

Response Response

Login Response

1-LEVEL DFD

Check Details

Login

Student

Student

Reply

Reference

Request to view

Attendance

Display Data

Request to view

Response

Insert new password

Request to new password

Student

Reply

Response

E-R DIAGRAM

An Entity Relationship diagram is a visual representation of different data using conventions that describes how these data are related to each other. ER Diagram express overall logical structure of database graphically. It shows are relationship between different entities. The entities can have composite, multivolume or derived attributes.

Three main components of an ERD are the [entities](http://whatis.techtarget.com/definition/entity), which are objects or concepts that can have data stored about them, the relationship between those entities, and the [cardinality](http://whatis.techtarget.com/definition/cardinality), which defines that relationship in terms of numbers.

The three main cardinal relationships are:

* **One-to-one (1:1).** For example, if each customer in a database is associated with one mailing address.
* **One-to-many (1:M).** For example, a single customer might place an order for multiple products. The customer is associated with multiple entities, but all those entities have a single connection back to the same customer.
* **Many-to-many (M:N).** For example,at a company where all call center agents work with multiple customers, each agent is associated with multiple customers, and multiple customers might also be associated with multiple agents.

SYMBOLS OF E-R DIAGRAM

Entity

Weak Entity

Relationship

Weak Relationship

Attribute

have

ADMIN

BRANCH

has

has

Teacher

Student

has

Attendance

have

**FEASIBILITY STUDY**

A feasibility study is conducted to select the best system that meets performance requirement. This entails an identification description, an evaluation of candidate system and the selection of best system for the job. The system required performance is defined by a statement of constraints, the identification of specific system objective and a description of outputs.

The key consideration in feasibility analysis is:

1. **Economic Feasibility**
2. **Technical Feasibility**
3. **Operational Feasibility**
4. **Behavioral Feasibility**

**ECONOMIC FEASIBILITY**

Here we analyze the costs of specific software, hardware, personnel office space and so forth for each implementation alternatives for website. Each costs and benefits estimates can be analyzed to determine how rapidly costs are recovered by benefits, to calculate both the absolute and interest adjustment amount of excess benefits and to establish the ratio of benefits of costs for the software. More commonly known as cost- benefits analysis; procedure is to determine the benefits and savings that are expected from a proposed system and compared them with costs.

**TECHNICAL FEASIBILITY**

In this phase alternatives for hardware, software and general design approach of website are determined. Here we must consider the following points:

* Does the necessary technology for the website exist?
* Does the proposed equipment have the technical capacity to hold the data required to use the website?
* Can this system be expended, if developed?
* Are there technical guarantees of accuracy, reliability, easy of access and data security?

**OPERATIONAL FEASIBILITY**

Operational feasibility is a measure of how people are able to work with the software. This type of feasibility demands if the system will work if developed and installed. Here following points must be considered:

* Is there sufficient support for the website from the management and the users?
* Have the users been involved in the planning and development of website?
* Will this system cause any harm?

The following questions are related to this point:

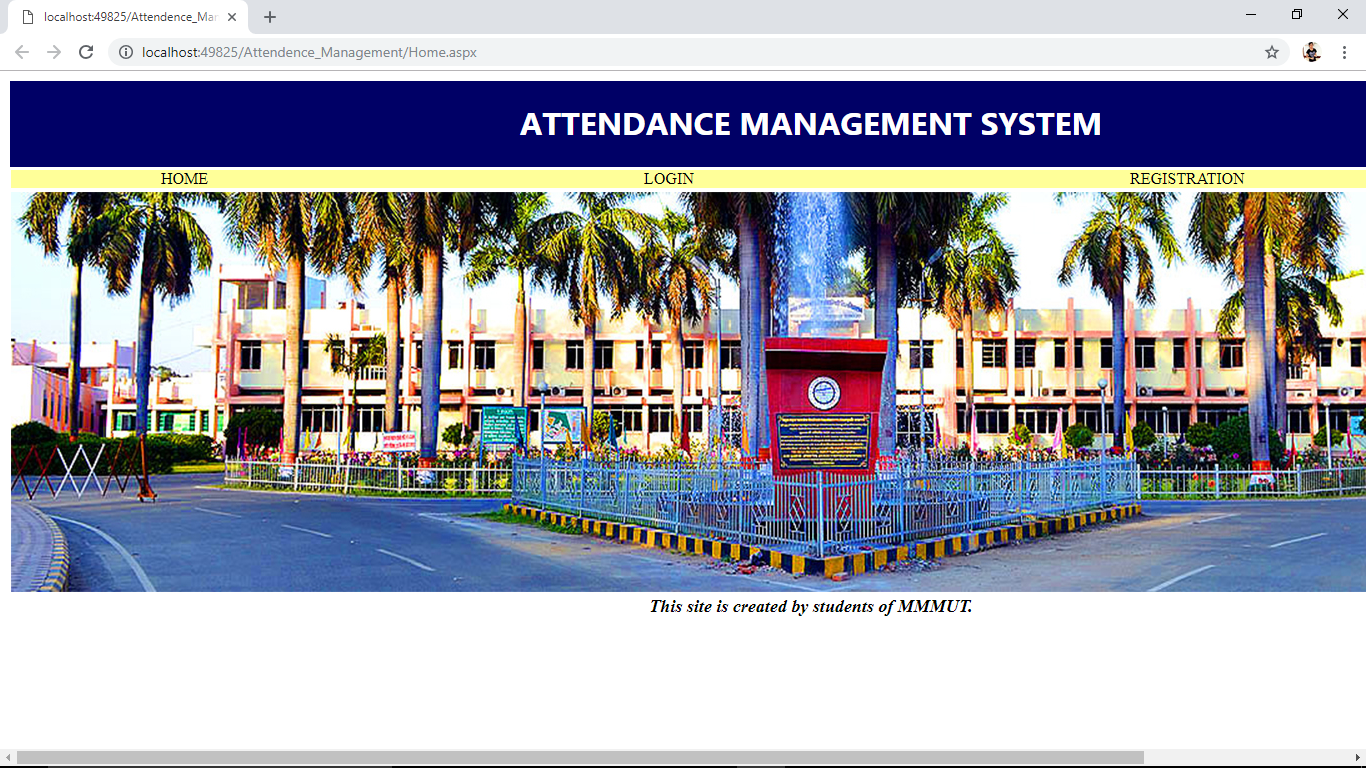
* Will loss of control result in any area?
* Will accessibility of information be lost?
* Will customer be affected in an undesirable way?
* Will its slow performance in any area?

**BEHAVIORAL FEASIBILITY**

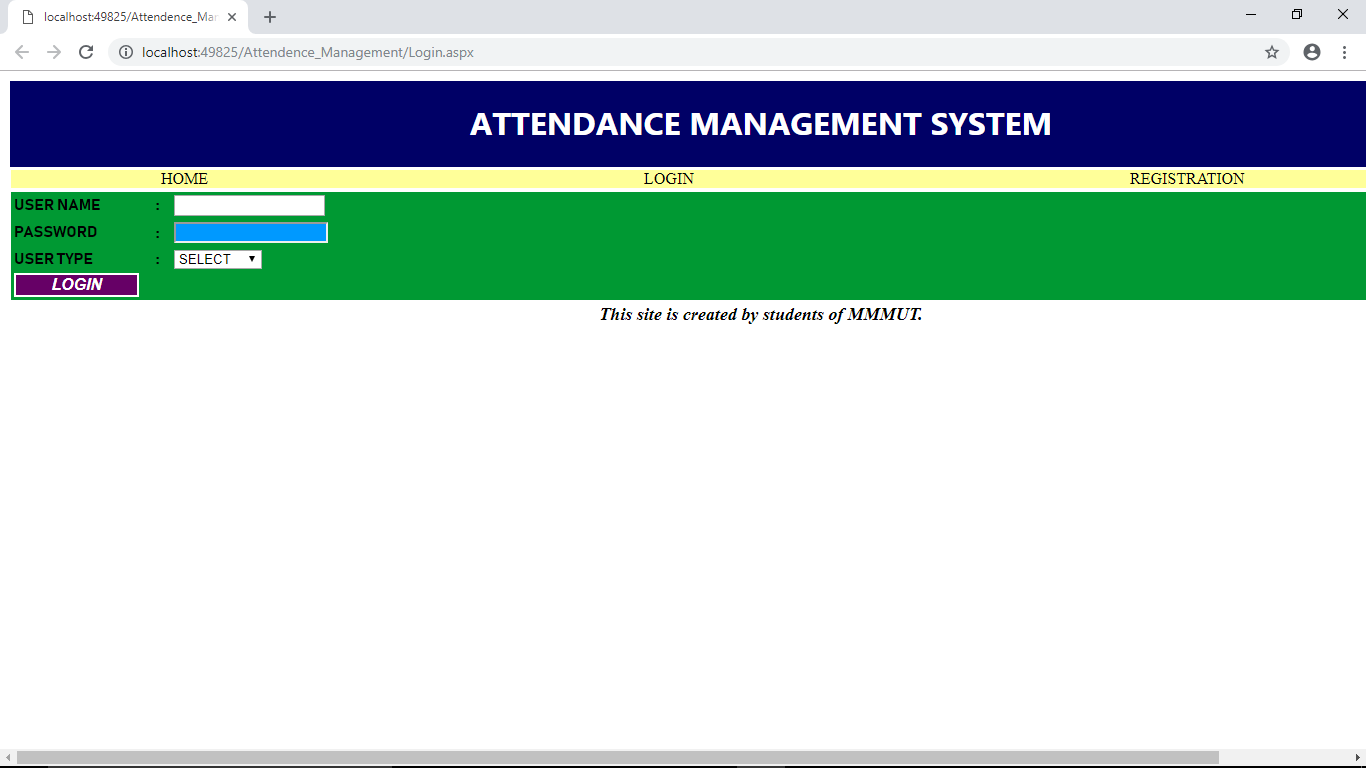
People are inherent to change. In this type of feasibility check, we come to know if the newly developed system will be taken and accepted by the working force i.e. the people who will use it.

**SCREEN SHOTS**

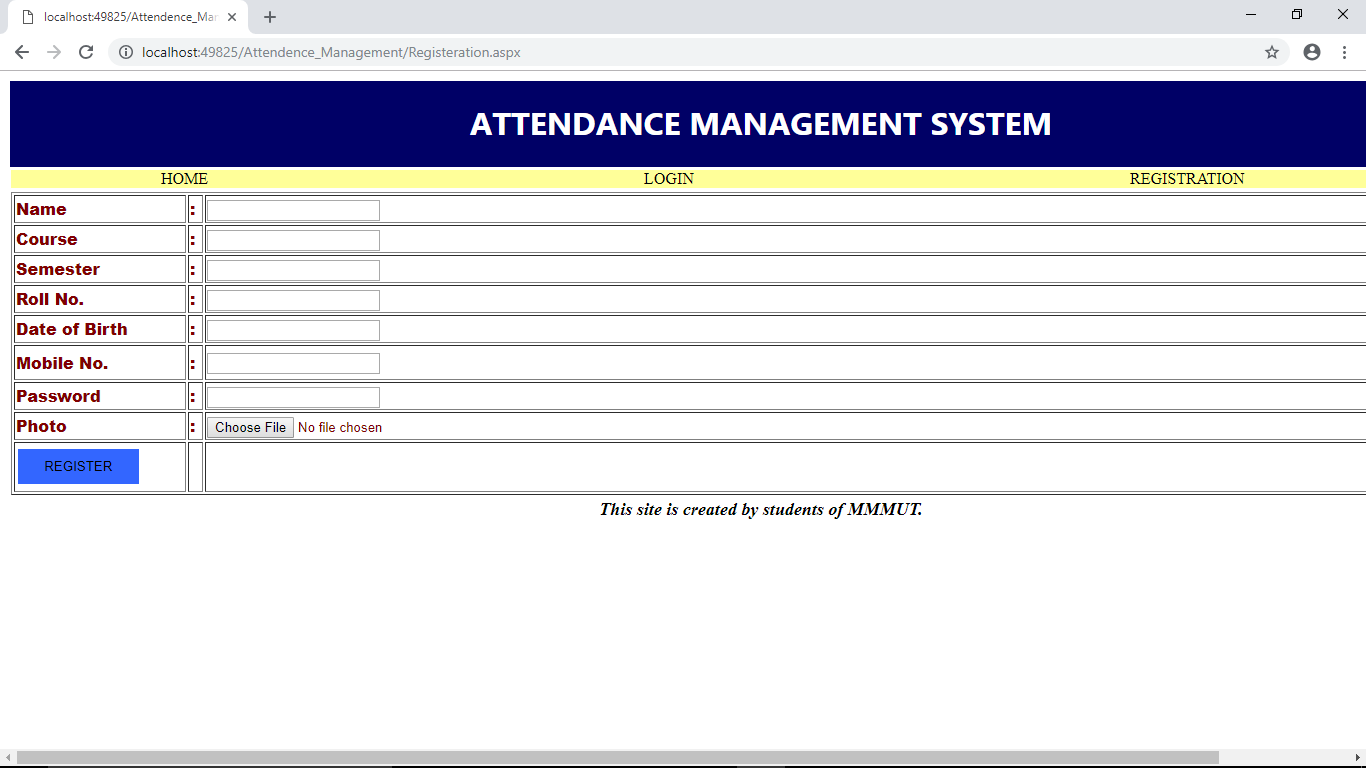
**Home :**



**Login :**

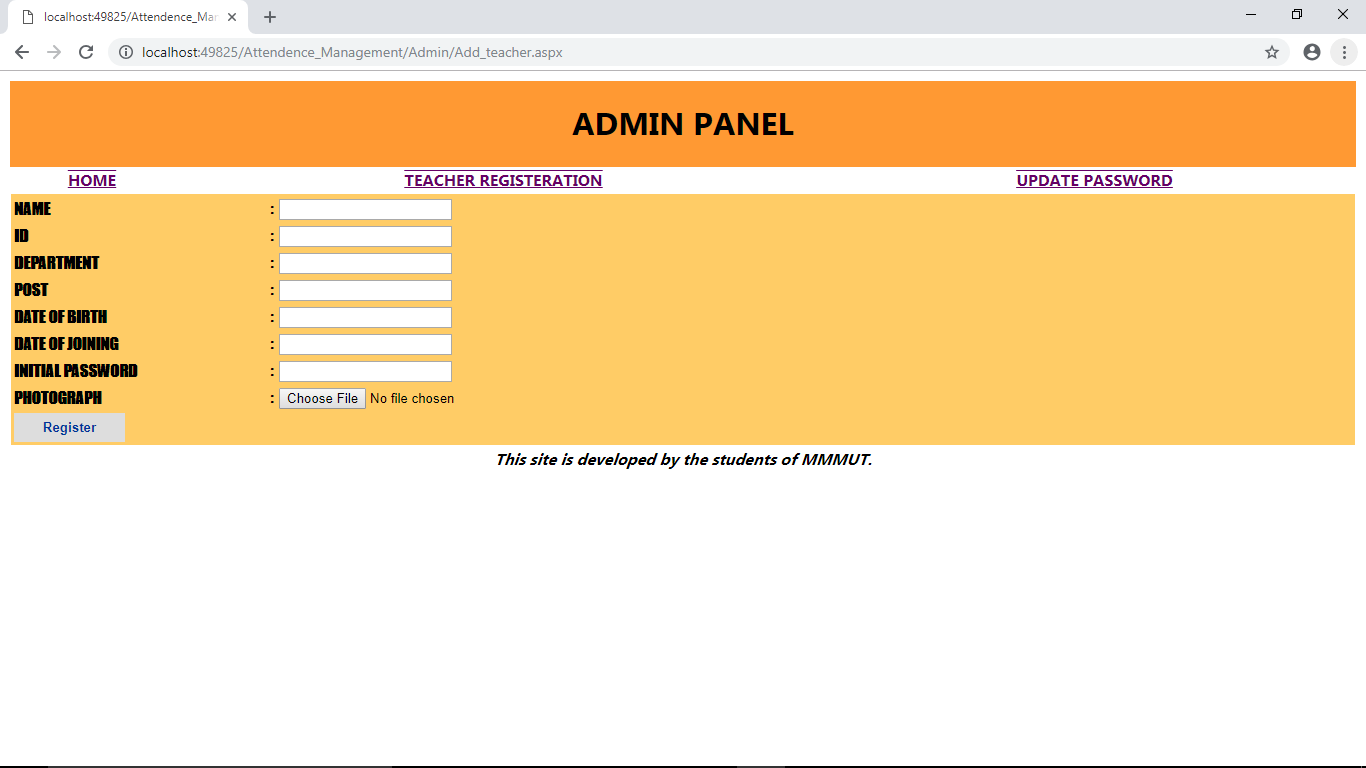


**Registration :**

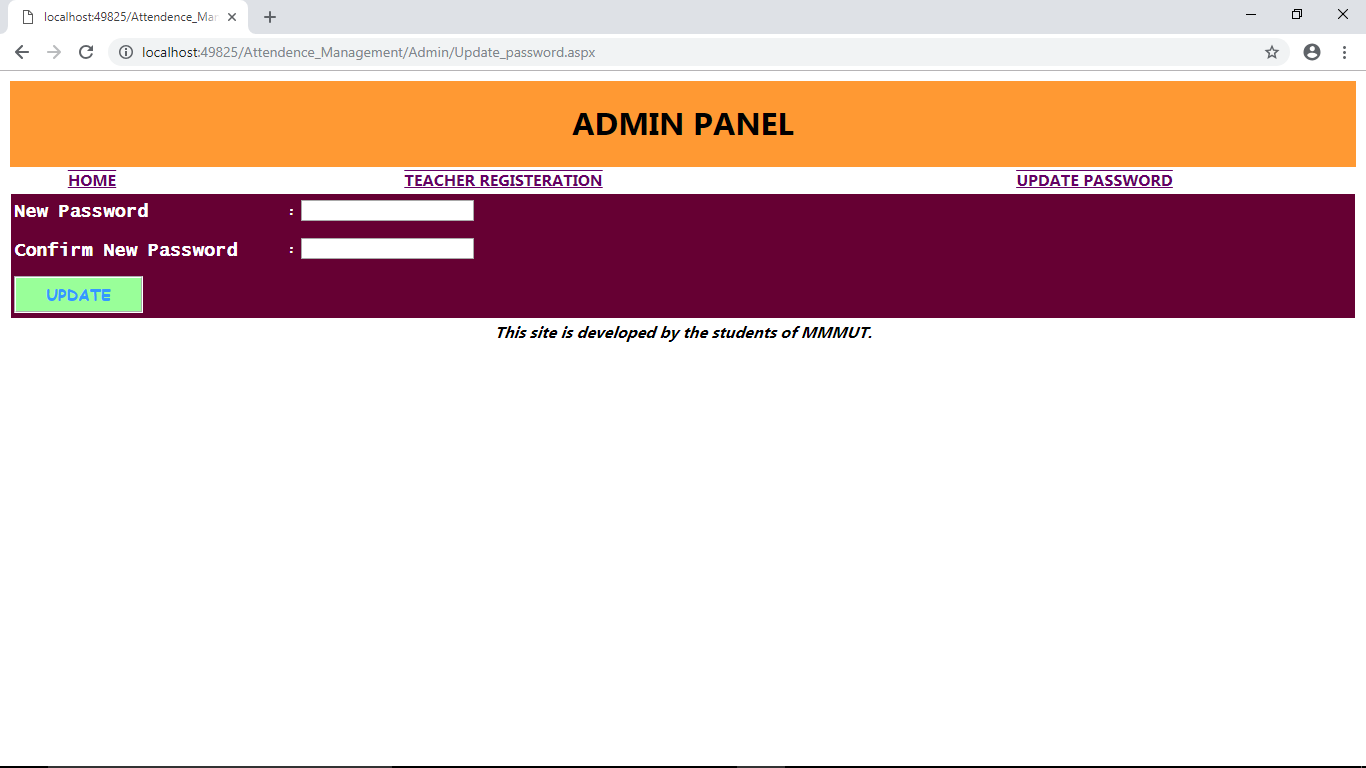


**Admin**

**\*Teacher Registration :**

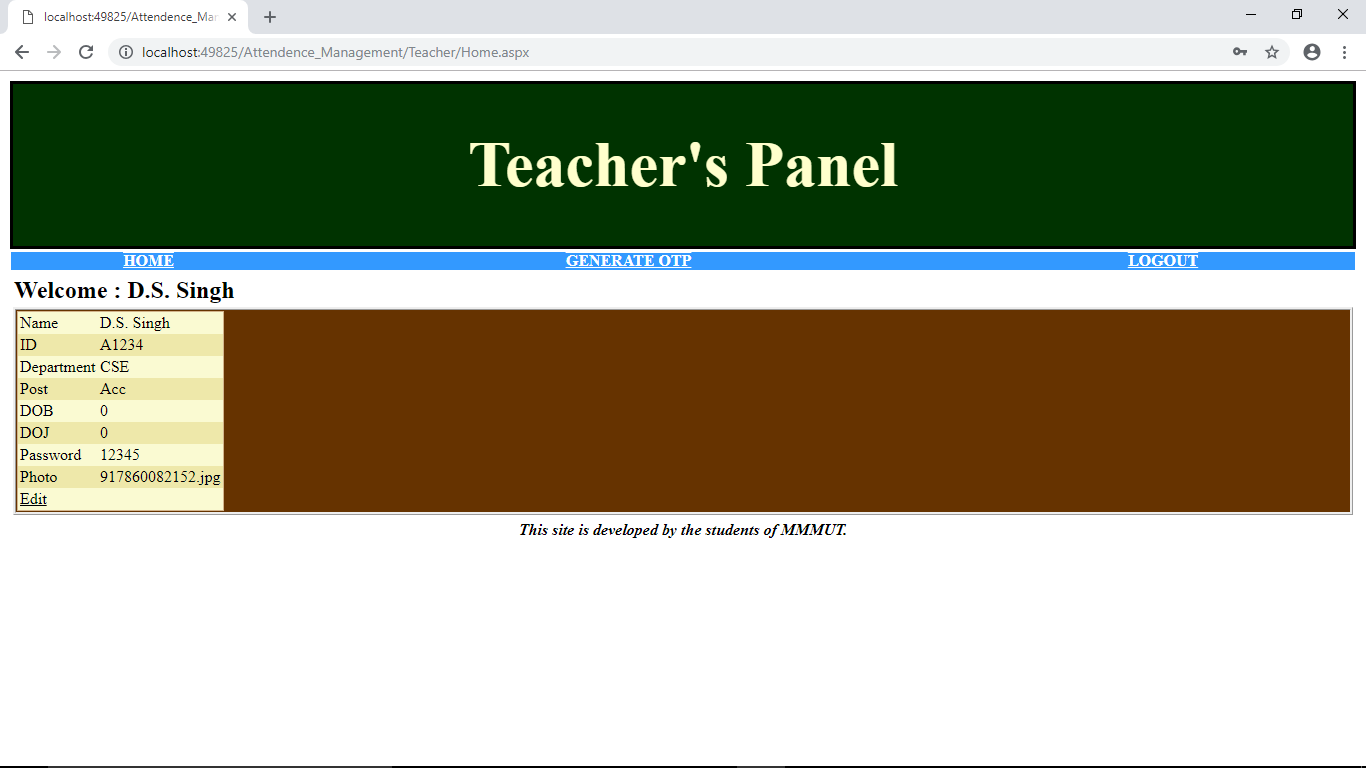


**\*Update Password :**

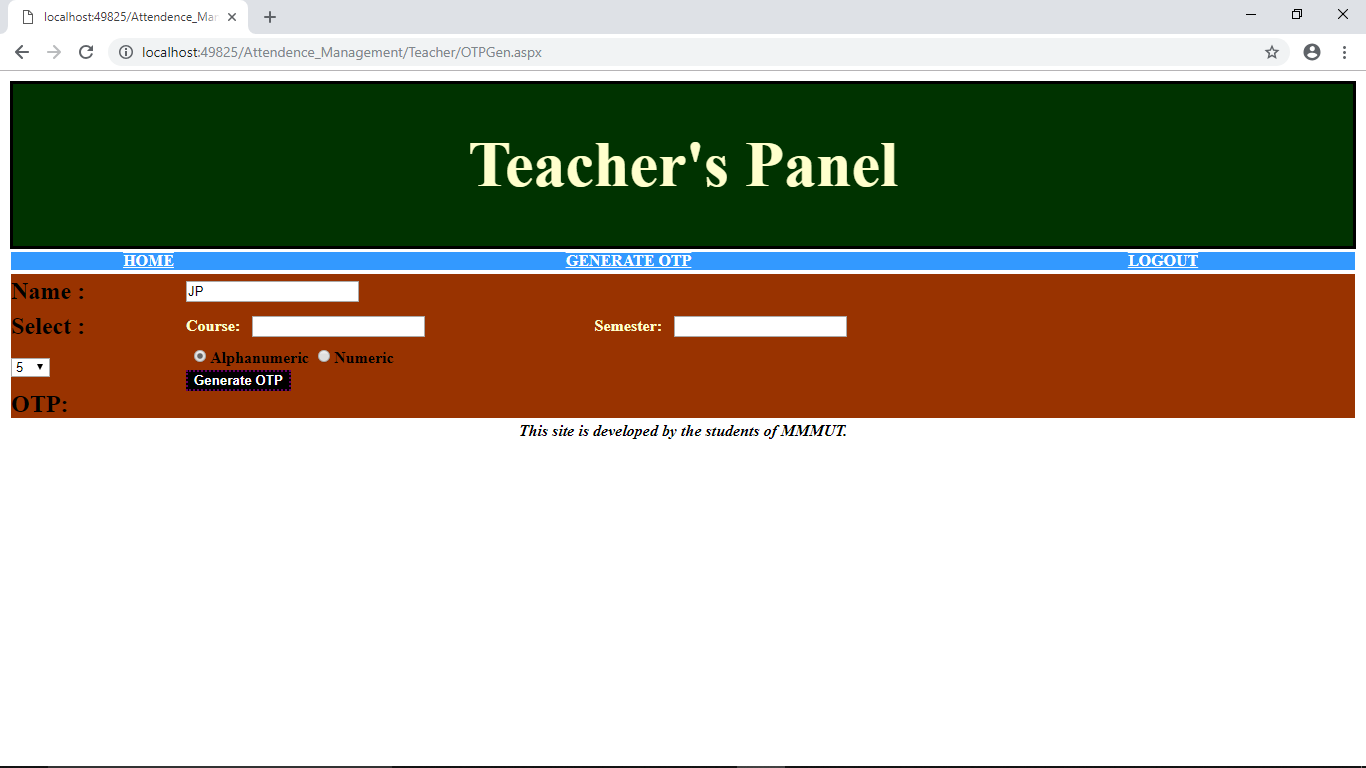


**Teacher’s Panel**

**\*Home:**

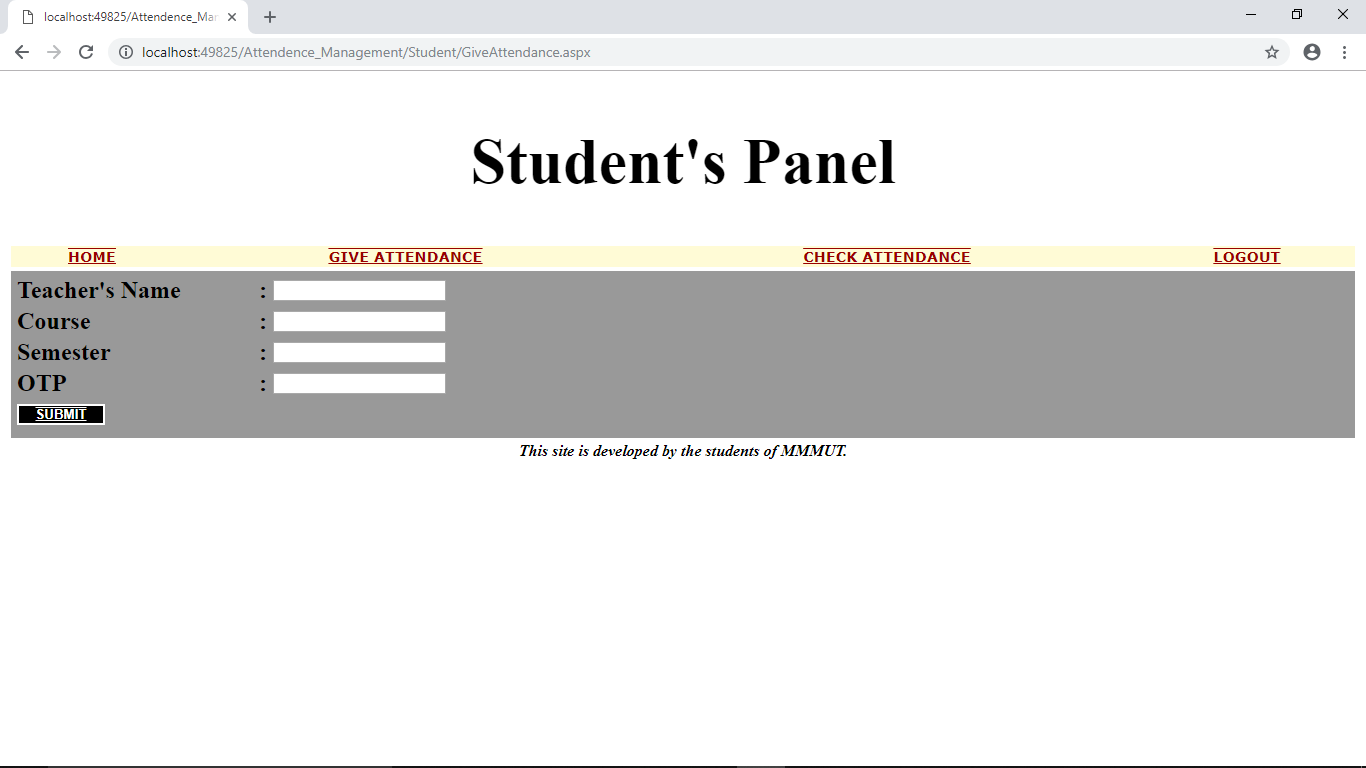


**\*Generate OTP:**



**Student’s Panel00:**

**\*Give attendance**



**TABLES**

**Login:**

|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| **UserName** | **Nvarchar(50)** |
| **Password** | **Nvarchar(50)** |
| **Type** | **Nvarchar(50)** |

**Student:**

|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| **Name** | **Nvarchar(50)** |
| **Course** | **Nvarchar(50)** |
| **Sem** | **Int** |
| **Roll** | **Nvarchar(50)** |
| **DOB** | **Int** |
| **Mob** | **Numeric(18,0)** |
| **Password** | **Nvarchar(50)** |
| **Photo** | **Nvarchar(MAX)** |

**Teacher:**

|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| **Name** | **Nvarchar(50)** |
| **ID** | **Nvarchar(50)** |
| **Department** | **Nvarchar(50)** |
| **Post** | **Nvarchar(50)** |
| **DOB** | **Int** |
| **DOJ** | **Int** |
| **Password** | **Nvarchar(50)** |
| **Photo** | **Nvarchar(MAX)** |

**Attendance:**

|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| **Name** | **Nvarchar(50)** |
| **Course** | **Nvarchar(50)** |
| **Sem** | **Int** |
| **TName** | **Nvarchar(50)** |
| **COTP** | **Int** |
| **CATT** | **Int** |

**OTP:**

|  |  |
| --- | --- |
| **Column Name** | **Data Type** |
| **Course** | **Nvarchar(50)** |
| **Sem** | **Int** |
| **TName** | **Nvarchar(50)** |
| **OTP** | **Int** |
| **COTP** | **Int** |

**CODING**

**\*Login:**

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Web;

usingSystem.Web.UI;

usingSystem.Web.UI.WebControls;

usingSystem.Data;

usingSystem.Data.SqlClient;

publicpartialclassLogin :System.Web.UI.Page

{

SqlConnectioncn = Class1.getconnection();

SqlCommandcmd = newSqlCommand();

SqlDataAdapteradp = newSqlDataAdapter();

SqlDataReaderdr;

protectedvoidPage\_Load(object sender, EventArgs e)

{

cmd.Connection = cn;

}

protectedvoid Button1\_Click(object sender, EventArgs e)

{

try

{

cn.Open();

string s = "";

cmd = newSqlCommand("select \* from Login where UserName='" + TextBox1.Text + "'and password='" + TextBox2.Text + "'", cn);

dr = cmd.ExecuteReader();

if (dr.Read())

{

s = dr["Type"].ToString();

if (s == "Admin")

{

Session["UserName"] = TextBox1.Text;

Response.Redirect("Admin/Home.aspx");

}

elseif (s == "stu")

{

Session["UserName"] = TextBox1.Text;

Response.Redirect("Student/Home.aspx");

}

elseif (s == "Teacher")

{

Session["UserName"] = TextBox1.Text;

Response.Redirect("Teacher/Home.aspx");

}

else

{

Label1.Text = "Please enter valid user name and password";

}

}

else

{

Label1.Text = "Please enter valid user name and password";

}

TextBox1.Text = "";

TextBox2.Text = "";

}

catch (Exception ex)

{

Label1.Text = "You are not Registered";

}

}

}

**\*Registration:**

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Web;

usingSystem.Web.UI;

usingSystem.Web.UI.WebControls;

usingSystem.Data;

usingSystem.Data.SqlClient;

publicpartialclassRegisteration :System.Web.UI.Page

{

SqlConnectioncn = Class1.getconnection();

SqlCommandcmd = newSqlCommand();

SqlDataAdapteradp = newSqlDataAdapter();

protectedvoidPage\_Load(object sender, EventArgs e)

{

cmd.Connection = cn;

}

protectedvoid Button1\_Click(object sender, EventArgs e)

{

cn.Open();

FileUpload1.SaveAs(Server.MapPath("../Attendence\_Management/StuPics")+"/"+FileUpload1.FileName);

Literal1.Text = "../Attendence\_Management/StuPics" + "/"+FileUpload1.FileName;

cmd.CommandText = "insert into Student values('" + TextBox1.Text + "','" + TextBox2.Text + "'," + TextBox3.Text + ",'" + TextBox4.Text + "'," + TextBox5.Text + "," + TextBox6.Text + ",'" + TextBox7.Text + "','" + FileUpload1.FileName + "')";

cmd.ExecuteNonQuery();

cmd.CommandText = "insert into Login values('" + TextBox1.Text.Replace("'", "''") + "','" + TextBox7.Text.Replace("'", "''") + "','stu')";

cmd.ExecuteNonQuery();

Literal1.Text = "Record Saved";

cn.Close();

TextBox1.Text = "";

TextBox2.Text = "";

TextBox3.Text = "";

TextBox4.Text = "";

TextBox5.Text = "";

TextBox6.Text = "";

TextBox7.Text = "";

}

}

**\*Add Teacher:**

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Web;

usingSystem.Web.UI;

usingSystem.Web.UI.WebControls;

usingSystem.Data;

usingSystem.Data.SqlClient;

publicpartialclassAdmin\_Add\_teacher :System.Web.UI.Page

{

SqlConnectioncn = Class1.getconnection();

SqlCommandcmd = newSqlCommand();

SqlDataAdapteradp = newSqlDataAdapter();

protectedvoidPage\_Load(object sender, EventArgs e)

{

cmd.Connection = cn;

}

protectedvoid Button1\_Click(object sender, EventArgs e)

{

cn.Open();

FileUpload1.SaveAs(Server.MapPath("../TeacherPics") + "/" + FileUpload1.FileName);

Literal1.Text = "../TeacherPics" + "/" + FileUpload1.FileName;

cmd.CommandText = "insert into Teacher values('" + TextBox1.Text + "','" + TextBox2.Text + "','" + TextBox3.Text + "','" + TextBox4.Text + "'," + TextBox5.Text + "," + TextBox6.Text + ",'" + TextBox7.Text + "','" + FileUpload1.FileName + "')";

cmd.ExecuteNonQuery();

cmd.CommandText = "insert into Login values('" + TextBox1.Text.Replace("'", "''") + "','" + TextBox7.Text.Replace("'", "''") + "','Teacher')";

cmd.ExecuteNonQuery();

Literal1.Text = "Record Saved";

cn.Close();

TextBox1.Text = "";

TextBox2.Text = "";

TextBox3.Text = "";

TextBox4.Text = "";

TextBox5.Text = "";

TextBox6.Text = "";

TextBox7.Text = "";

}

}

**\*Update Password:**

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Web;

usingSystem.Web.UI;

usingSystem.Web.UI.WebControls;

usingSystem.Data;

usingSystem.Data.SqlClient;

publicpartialclassAdmin\_Update\_password :System.Web.UI.Page

{

SqlConnectioncn = Class1.getconnection();

SqlCommandcmd = newSqlCommand();

SqlDataAdapteradp = newSqlDataAdapter();

protectedvoidPage\_Load(object sender, EventArgs e)

{

Label1.Text = Session["UserName"].ToString();

cmd.Connection = cn;

}

protectedvoid Button1\_Click(object sender, EventArgs e)

{

cn.Open();

cmd.CommandText = "update Login set Password='"+TextBox1.Text+"' where UserName='"+Label1.Text+"'";

cmd.ExecuteNonQuery();

Literal1.Text = "Password Updated";

cn.Close();

TextBox1.Text = "";

TextBox2.Text = "";

}

}

**\*OTP Generation:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data;

using System.Data.SqlClient;

public partial class Teacher\_OTPGen : System.Web.UI.Page

{

int countotp = 0;

SqlConnection cn = Class1.getconnection();

SqlCommand cmd = new SqlCommand();

SqlDataAdapter adp = new SqlDataAdapter();

protected void Page\_Load(object sender, EventArgs e)

{

TextBox3.Text = Session["UserName"].ToString();

cmd.Connection = cn;

}

protected void GenerateOTP(object sender, EventArgs e)

{

string alphabets = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

string small\_alphabets = "abcdefghijklmnopqrstuvwxyz";

string numbers = "1234567890";

string characters = numbers;

if (rbType.SelectedItem.Value == "1")

{

characters += alphabets + small\_alphabets + numbers;

}

int length = int.Parse(ddlLength.SelectedItem.Value);

string otp = string.Empty;

for (int i = 0; i < length; i++)

{

string character = string.Empty;

do

{

int index = new Random().Next(0, characters.Length);

character = characters.ToCharArray()[index].ToString();

} while (otp.IndexOf(character) != -1);

otp += character;

}

lblOTP.Text = otp;

countotp++;

cn.Open();

cmd.CommandText = "insert into [OTP] values('" + TextBox1.Text + "','" + TextBox2.Text + "', '" + TextBox3.Text + "','" + otp + "','" + countotp + "')";

cmd.ExecuteNonQuery();

cn.Close();

}

}

**\*Give Attendance:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data;

using System.Data.SqlClient;

public partial class Student\_GiveAttendance : System.Web.UI.Page

{

int catt = 0,f=1;

SqlConnection cn = Class1.getconnection();

SqlCommand cmd = new SqlCommand();

SqlDataAdapter adp = new SqlDataAdapter();

SqlDataReader dr;

protected void Page\_Load(object sender, EventArgs e)

{

cmd.Connection = cn;

}

protected void Button2\_Click(object sender, EventArgs e)

{

string s = Session["Username"].ToString();

cn.Open();

cmd = new SqlCommand("select \* from OTP where Course='" + TextBox2.Text + "'and Sem='" + TextBox3.Text + "'and TName='" +TextBox5.Text + "'and OTP='" + TextBox4.Text + "'", cn);

dr = cmd.ExecuteReader();

if (dr.Read())

{

Label1.Text = "Submitted";

TextBox4.Text = "";

TextBox2.Text = "";

TextBox3.Text = "";

TextBox5.Text = "";

catt++;

if (f == 1)

{

f++;

cmd.CommandText = "insert into Attendance values('" + s + "','" + TextBox2.Text + "','" + TextBox3.Text + "','" + TextBox5.Text + "','" + catt + "','"+ f +"')";

cmd.ExecuteNonQuery();

}

else

{

cmd = new SqlCommand("select \* from Attendance where TName='" + TextBox5.Text + "'",cn);

cmd.CommandText="update main Attendance values('"+catt+"')";

}

}

else

{

Label1.Text = "Either the OTP is wrong or the details provided are wrong";

TextBox4.Text = "";

TextBox2.Text = "";

TextBox3.Text = "";

TextBox5.Text = "";

}

cn.Close();

}

}

**TESTING AND VALIDATIONS**

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding testing requires that the developer discard preconceived notions of the “correctness” of software just developed and overcome a conflict of interest that occurs when errors are uncovered.

Testing is a process of executing a program with the intent of finding an error. Also testing is one element of a broader topic often referred as **Validation & Verification.**

There are mainly 2 types of testing:-

* **White box testing**
* **black box testing**

**WHITE BOX TESTING**

Sometimes it is also called as glass box testing this is a test case design method that uses the control structure of procedural design to derive the test. Using this guarantees that

(1) All independent paths within a module have been exercised at least once.

(2) All logical decisions were exercised to their true and false side.

(3) Executed all it tests to their boundaries within their operational bounds

(4) Exercised internal data structures to ensure their validity.

**BLACK BOX TESTING**

This focuses on functional requirements of the software. It finds errors for:-

(1) Incorrect or missing functions

(2) Interface errors

(3) Errors in Data Structures or external data base access.

(3) Performance errors

(4) Initialization and termination errors.

Test Case considerations

**Unit Testing**

Starting from the innermost part i.e. Code the testing applied was *Unit Testing*. Unit testing makes heavy use of white Box testing techniques. Here each module structure was tested to cover maximum error detection.

1. Number of input parameters equal to number of arguments (e.g. as in case of function call to upload teacher code in combo box parameter supplied)
2. Parameter and arguments of attribute match
3. Parameter and arguments of system units match
4. Number attributes and order of arguments to built in functions are correct
5. Any references to parameters not associated with current point of entry
6. Global variables definitions is consistent across modules
7. Are file attributes correct?
8. Format specification matches with I/O statement?
9. End of file conditions are handled?
10. Files are opened before use?
11. Is there erroneous initialization?
12. Are variable names correct?
    * The module and forms were checked to ensure that they were actually supplying the correct information as desired.
    * All errors reported by SQL Server, due to applied constraints were displayed properly and were trapped.
    * All the dates were captured properly and validation checks applied were working as required.
    * Similarly data Reports were checked individually to display the information as desired.

**Integration Testing**

* It should be tested that when integrated data is not lost across an interface
* They don’t have adverse effect on each other
* They produce desired main function
* Global data structures do not produce problems
* The entire program should be tested as a whole. Technique used here was incremental approach where each integration was made and tested to get final result.

**System Testing**

1. The software was incorporated with documentation
2. The executable version was produced for deployment
3. a series of bad data tests were made to check any unearthed errors

**VALIDATION**

On the basis of specified task, function security and privilege levels, we can divide whole system into following users-

**System Administrator-**

* + Owner that takes care of maintenance for the online movie buying.

**User-**

* + One of significant user of this movie buying site who impart efficient participation in communication through forum . They can also enjoy whole availability of movie materials .

1. No record can be saved till all the necessary entries are done.
2. Only administrator can perform sophisticated tasks like registration of STAFF, controlling.

CONCLUSION

In this work , the web based attendance management system is developed using C# server-side scripting language and CSS,HTML for designing which is fully meet the system’s goals.

This system overcome many limitations incorporated in attendance, this system saves a great amount of time and reduces errors which may occur during attendance calculation. The system I have developed is fully responsive which can be used in mobile, tablets and different operating systems. Some other benefits are,

Automated and web-based for easy accessibility

1. It is a dynamic and flexible system
2. It excludes paperwork and the possibility of making mistakes while using paper for taking attendance .
3. It is very user friendly and handy
4. The records of current and previous can be available in prompt and an immediate.

**REFERENCES**

1. Mastering Asp.Net with c# Black Book.

2. Mastering Wrox Book for asp.net with C#.

3. Software Engineering ‘A practitioner’s approach’ – Roger S. Pressman, (McGraw- Hill International Edition)