### 

##### A Software Engineering Project

##### On

E-Assignment

as the partial fulfillment of

**Semester -V study**

for the degree of

**Bachelor of Science (Computer Science)**

## (Affiliated to Veer Narmad South Gujarat University)

during the academic year 2019-2020.

### 

**Developed By: Guided By:**

1)Patil Utkarsh Ms.Shagufta Khan



Sarvajanik Education Society

## Shree Ramkrishna Institute of Computer Education and Applied Sciences

## Behind P.T Science College, M.T.B. College Campus, Athwalines, Surat-395 001.

### 

### Sarvajanik Education Society

## Shree Ramkrishna Institute of Computer Education and Applied Sciences

# Behind P.T. Science College, M.T.B College Campus, Athwalines, Surat-395 001.

### CERTIFICATE

## DEPARTMENT OF COMPUTER SCIENCE

This is to certify that

Student name Exam no

1) Utkarsh Patil 4246

has / have successfully completed his / her / their software engineering project work entitled

E Assignment

as the partial fulfillment of

**Semester -V study**

for the degree of

**Bachelor of Science (Computer Science)**

## (Affiliated to Veer Narmad South Gujarat University)

during the academic year 2019-2020.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#### Internal Project Guide HOD

#### Dept. of Computer Science

#### Date: \_\_\_\_\_\_\_\_\_\_\_\_ Place: Surat

### University Examination:

**Examination No.: \_\_\_\_\_\_\_\_\_\_\_\_**

**Signature of External Examiners: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**INDEX for Project Documentation**

|  |  |  |  |
| --- | --- | --- | --- |
| **SR.NO** | **CONTENTS** | | **Page. No** |
| **1.** | **Introduction** | |  |
|  |  | Project Profile | 4 |
| **2.** | **System Introduction** | |  |
|  |  | System Definition, Objective and scope | 5 |
|  |  | Hardware & Software Requirements | 5 |
| **3.** | **Requirement Analysis & Modeling** | |  |
|  |  | Expected working of system | 6 |
|  | Data Flow Diagrams (DFD) | 7 |
|  | Process Specification | 10 |
|  | Data Dictionary | 13 |
| **4.** | **Design** | |  |
|  |  | ER Diagram & Database Design | 16 |
|  | System Architecture | 17 |
|  | System Flowchart | 21 |
|  | Form & Report Designs (Screenshots) Layouts | 23 |
| **5.** | **Coding** | |  |
|  |  | Coding Approach/style | 27 |
|  | Code Snippets | 27 |
| **6.** | **Testing** | |  |
|  |  | Testing Data Inputs (Data Validations) | 42 |
|  | Testing Operations (Actions on each screen) | 42 |
| **7.** | **New Tools/ Technologies learned/used** | | 43 |
| **8.** | **System Limitations/Restrictions and Dependencies/Constraints** | | 44 |
| **9.** | **Future Enhancement & Opportunities** | | 45 |
| **10.** | **Bibliography & References** | | 46 |

**1. Introduction**

**1.1 Project Profile**

* **Project Name:** E-Assignment
* **Developed By:** Patil Utkarsh
* **Project Guide:** Shagufta Khan
* **Front-End Tool:** ASP.NET
* **Back-End Tool:** SQL Server Management Studio

**2. System Introduction**

**2.1 System Definition Scope & Objective**

* The E-Assignment provides an online forum for related activities.
* Users are administrator, student and mentor.
* The aim and objective of this project to develop online Assignment System to promote and encourage student to actively participate in the academic activities.
* This project replaces the conventional and inconvenient method of clearing doubts in classes which involves the faculty and student to be physically present at the same time.
* In this context arises the important of this project. This project bridges the existing gap between faculty and students due to the time constraints.
* Here faculty can upload assignment and student will be submit their assignment to given time.
* Student can post doubt and faculty give answer to the posted doubt and clear students doubt.

**2.2 Hardware and Software Requirements**

* **Hardware Requirements:**

* **Application Language:** ASP.NET

* **Back-End:** SQL Server Management Studio
* **Software Requirements:**
* **User:**
* ***Web Server****:* Any Browser
* ***Disk:*** 1 GB
* ***RAM:*** 2 GB
* **Admin:**
* Any kind of Server.

**3. Requirement Analysis and Modeling**

**3.1 Expected Working of System**

* Here admin add Faculty and Student can register them or admin can also add them.
* Faculty upload the assignment for the student but faculty must be login before uploading any type of assignment on website.
* Uploaded assignment submitted by student who under of that faculty but student also login before submitting any type of assignment on website.
* If any assignment related doubt occupied in student mind then they can post their problem.
* Faculty see the doubt of student and solve their doubt and other login student who know doubt post related reply then they can also solve that doubt.
* After registration student can’t get Registration successful Email.
* **Security Module**
* Before uploading and submitting assignment or any type of post on website user must be login.
* If any user tries to login and attempt all 4 login attempts then account will be lock for 24 hours due to prevent anonymous activity of login.
* User can change password via using username of him and link sent on his registered E-mail address and user can change the password.
* If user change forget link from given E-mail after open browser then link will be converting into invalid link and then user can’t reset password. After invalid of link user again do that reset password process to change password.
* User can’t access after locking account before 24 hour and after reset password also can’t access account.
* Before 4 attempts user can reset password and login their account with new password.
* User account will be unlocking automatically after 24 hours completed.
* User can change password after login using current password.

**3.2. Data Flow Diagram**

* **Context Level DFD (Zero Level):**

Admin

E-

Assignment

Faculty

Student

Login

Registered Student,Teacher

Register Subject

Assign subject

Response

Login

Assignment

Doubt

Reply

Register

Login

Doubt

Reply

Assignment

* **First Level DFD:**
* **Admin Module:**

1.1

. Login

Process

Admin

. Register

1.2

Faculty

Process

tblAdmin

tblFaculty

tblSubject

1.3

. Register

Subject

Process

Username, Password

Username, Password

Faculty Details

Subject Details

Permission

Permission

Permission

Permission

Faculty Details

Request

Permission

Subject Details

* **Faculty Module:**

2.1

. Login

Process

Faculty

2.2.

Assignment

Process

tblFaculty

tblPostAssignment

tblDoubt

2.3

. Doubt

Process

Username, Password

Username, Password

Asnwer

Doubt

Permission

Permission

Permission

Doubt

Answer

Request

Permission

Request

tblReply

Permission

Reply

* **Student Module:**

3.2

. Login

Process

Student

tblStudent

3.1.

Registration

Process

3.3.

Assignment

Process

tblStudent

tblSubmitAssignmnet

tblDoubt

. Doubt

3.4

Process

Username, Password

Student Details

Username, Password

Submit

Doubt

Permission

Permission

Permission

Student Details

Student Details

Permission

Submit

Doubt

Request

Permission

**3.3 Process Specification**

|  |  |
| --- | --- |
| **Process Name** | **1.1 Login Process** |
| **Entity** | Admin |
| **Database** | tblAdmin |
| **Input** | Username, Password |
| **Output** | Login Successful message. |

|  |  |
| --- | --- |
| **Process Name** | **1.2 Register Faculty Process** |
| **Entity** | Admin |
| **Database** | tblFaculty |
| **Input** | Faculty details. |
| **Output** | Registration Successful message. |

|  |  |
| --- | --- |
| **Process Name** | **1.3 Register Subject Process** |
| **Entity** | Admin |
| **Database** | tblSubject |
| **Input** | Subject details. |
| **Output** | Registration Successful message. |

|  |  |
| --- | --- |
| **Process Name** | **2.1 Login Process** |
| **Entity** | Faculty |
| **Database** | tblFaculty |
| **Input** | Username, Password |
| **Output** | Login Successful message. |

|  |  |
| --- | --- |
| **Process Name** | **2.2 Assignment Process** |
| **Entity** | Faculty |
| **Database** | tblPostAssignment |
| **Input** | Assignment details. |
| **Output** | Successful message. |

|  |  |
| --- | --- |
| **Process Name** | **2.3 Doubt Process** |
| **Entity** | Faculty |
| **Database** | tblDoubt, tblReply |
| **Input** | Doubt details. |
| **Output** | Successful message. |

|  |  |
| --- | --- |
| **Process Name** | **3.1 Registration Process** |
| **Entity** | Student |
| **Database** | tblStudent |
| **Input** | Student details. |
| **Output** | Registration Successful message. |

|  |  |
| --- | --- |
| **Process Name** | **3.2 Login Process** |
| **Entity** | Student |
| **Database** | tblStudent |
| **Input** | Username, Password |
| **Output** | Login Successful message. |

|  |  |
| --- | --- |
| **Process Name** | **3.3 Assignment Process** |
| **Entity** | Student |
| **Database** | tblStudent |
| **Input** | Assignment details. |
| **Output** | Successful message. |

|  |  |
| --- | --- |
| **Process Name** | **3.4 Doubt Process** |
| **Entity** | Student |
| **Database** | tblStudent |
| **Input** | Doubt details. |
| **Output** | Successful message. |

**3.4 Data Dictionary**

* **Student Registration**

|  |  |
| --- | --- |
| **Name** | **Student Registration** |
| **Alias** | None |
| **Where used** | 3.1 Registration Process |
| **Contain Id** | StudId, FName, LName, Username, Password, Email, Mobile, GraduationType, Course, Class, ImageName, Size, ImageData, DOB, RetryAttempt, IsLocked, LockedDateTime |
| **Type** | Input |

* **Faculty Registration**

|  |  |
| --- | --- |
| **Name** | **Faculty Registration** |
| **Alias** | None |
| **Where used** | 1.2 Register Faculty Process |
| **Contain Id** | FacultyId, FName, LName, Username, Password, Email, Mobile , GraduationType, Course, Class, ImageName, Size, ImageData, DOB, RetryAttempt, IsLocked, LockedDateTime |
| **Type** | Input |

* **Login**

|  |  |
| --- | --- |
| **Name** | **Login** |
| **Alias** | None |
| **Where used** | (1.1, 2.1, 3.2) Login Process |
| **Contain Id** | IsLocked, Username, Password |
| **Type** | Output, Input |

* **Subject**

|  |  |
| --- | --- |
| **Name** | **Subject Registration** |
| **Alias** | None |
| **Where used** | 1.3 Register Subect Process |
| **Contain Id** | SubjectId, SubjectName, FacultyId |
| **Type** | Input, Output |

* **Assignment**

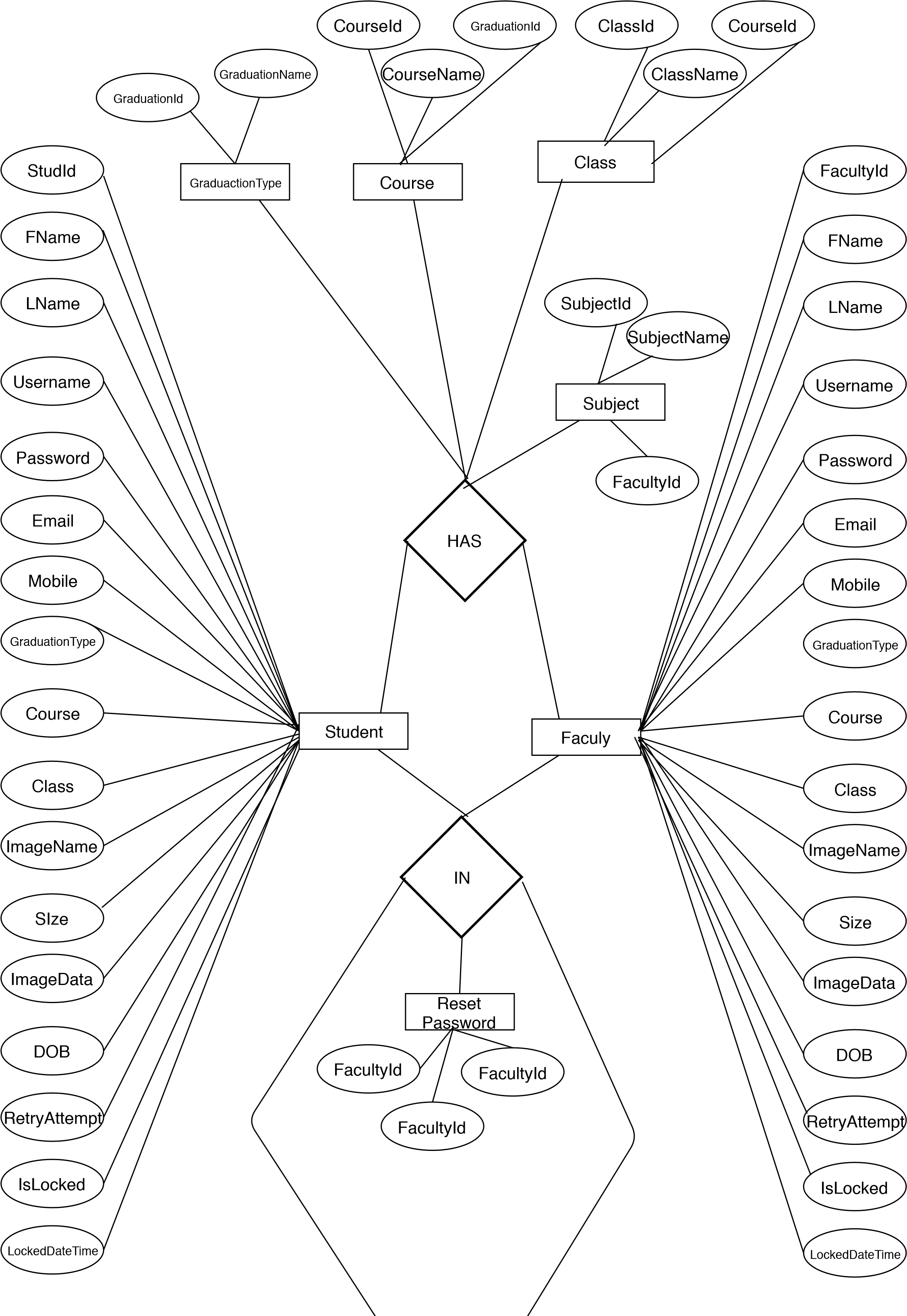
|  |  |
| --- | --- |
| **Name** | **Assignment** |
| **Alias** | Post Assignment, Submit Assigment |
| **Where used** | (2.2, 3.3) Assignment Process |
| **Contain Id** | PAId, FacultyId, AssignmentData, AssignmentDateTime  SAId, StudId, AssignmentData, AssignmentDateTime |
| **Type** | Input, Output |

* **Doubt**

|  |  |
| --- | --- |
| **Name** | **Doubt** |
| **Alias** | Post Doubt, Reply Doubt |
| **Where used** | (2.3, 3.4) Doubt Process |
| **Contain Id** | DoubtId, StudId, DoubtData, DoubtDateTime  ReplyId, FacultyId, StudId, ReplyData, ReplyDateTime |
| **Type** | Input, Output |

**4. Design**

**4.1 ER Diagram**



Post

Assignment

Reply

Submit

Assignment

PAId

FacultyId

AssignmentData

AssignmentDateTime

AssignmentDateTime

AssignmentData

StudId

SAId

IN

Doubt

DoubtDateTime

DoubtData

DoubtId

StudId

StudId

ReplyDateTime

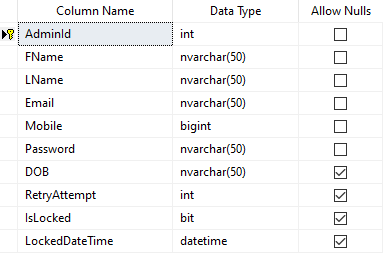
ReplyData

FacultyId

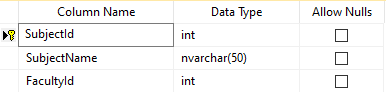
ReplyId

**4.2. Database Design**

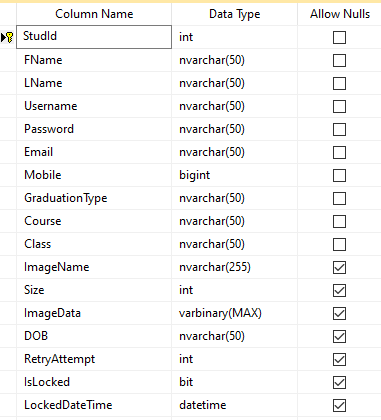
* **Admin Database**

****

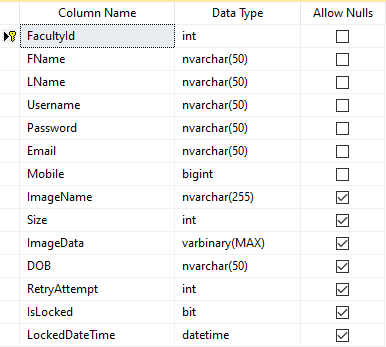
* **Subject Database**

****

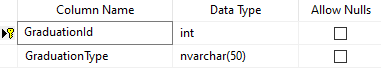
* **Student Registration Database**

****

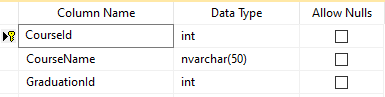
* **Faculty Registration Database**

****

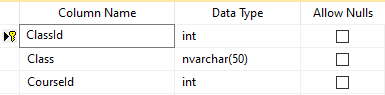
* **Graduation Type Database**

****

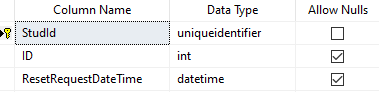
* **Course Database**

****

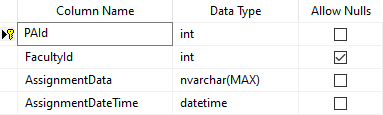
* **Class Database**

****

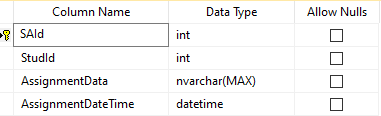
* **Reset Password Request Database**

****

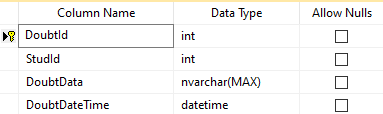
* **Post Assignment Database**

****

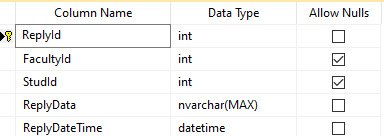
* **Submit Assignment Database**

****

* **Doubt Database**

****

* **Reply Database**

****

**4.3 System Flowchart**

Start

End

Register

Login

Reset

Password

Verified

Login

Forget

Password

Post

Doubt/Submit

Assignment

Reply Doubt

Post

Assignment

Logout

IsLocked

Yes

No

Yes

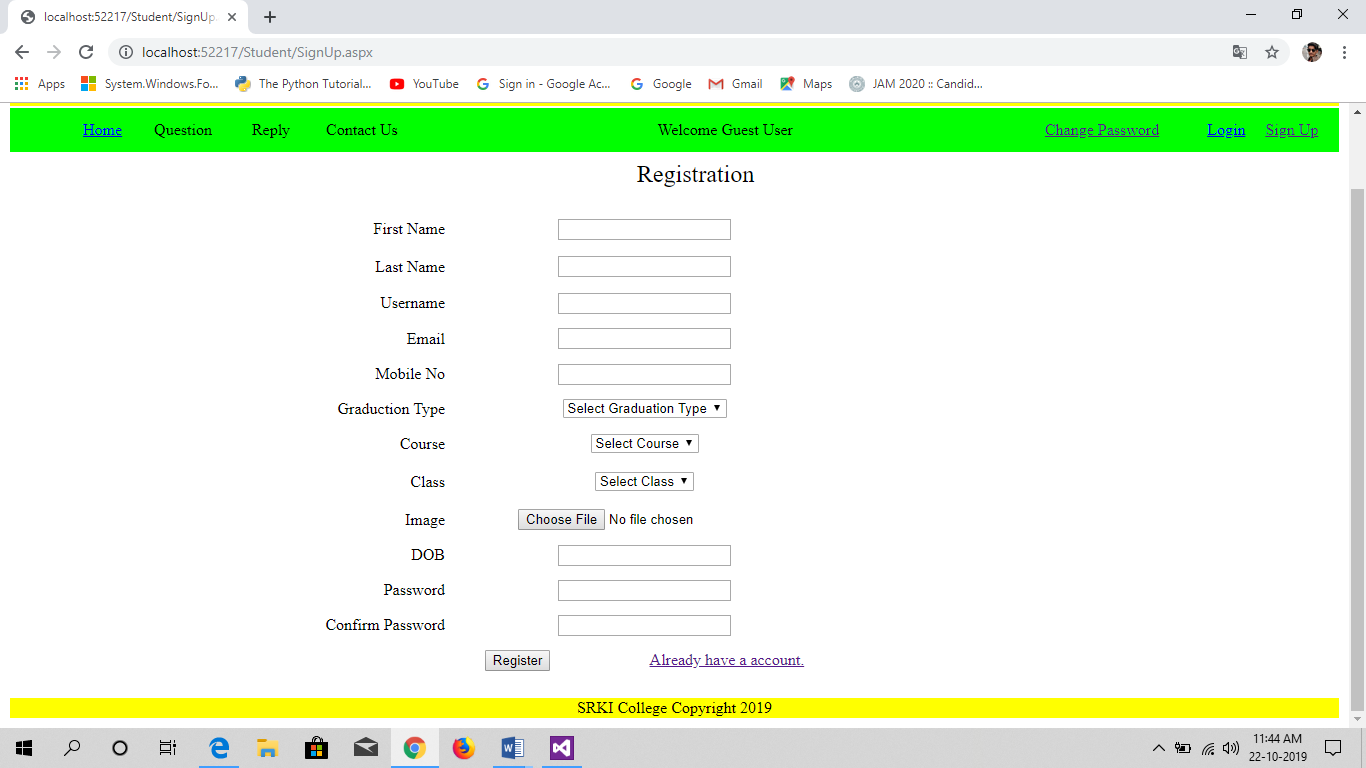
No

Yes

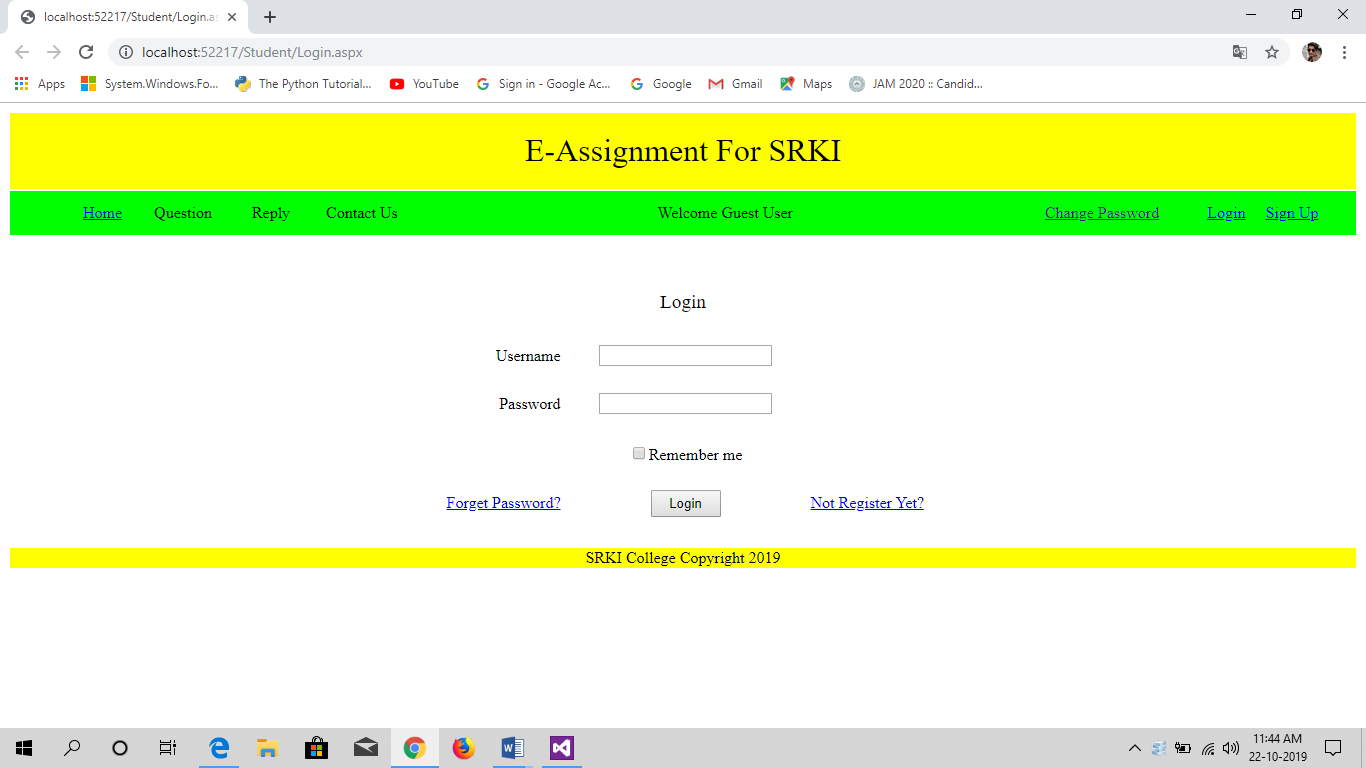
No

**4.4 Form and Report Design**

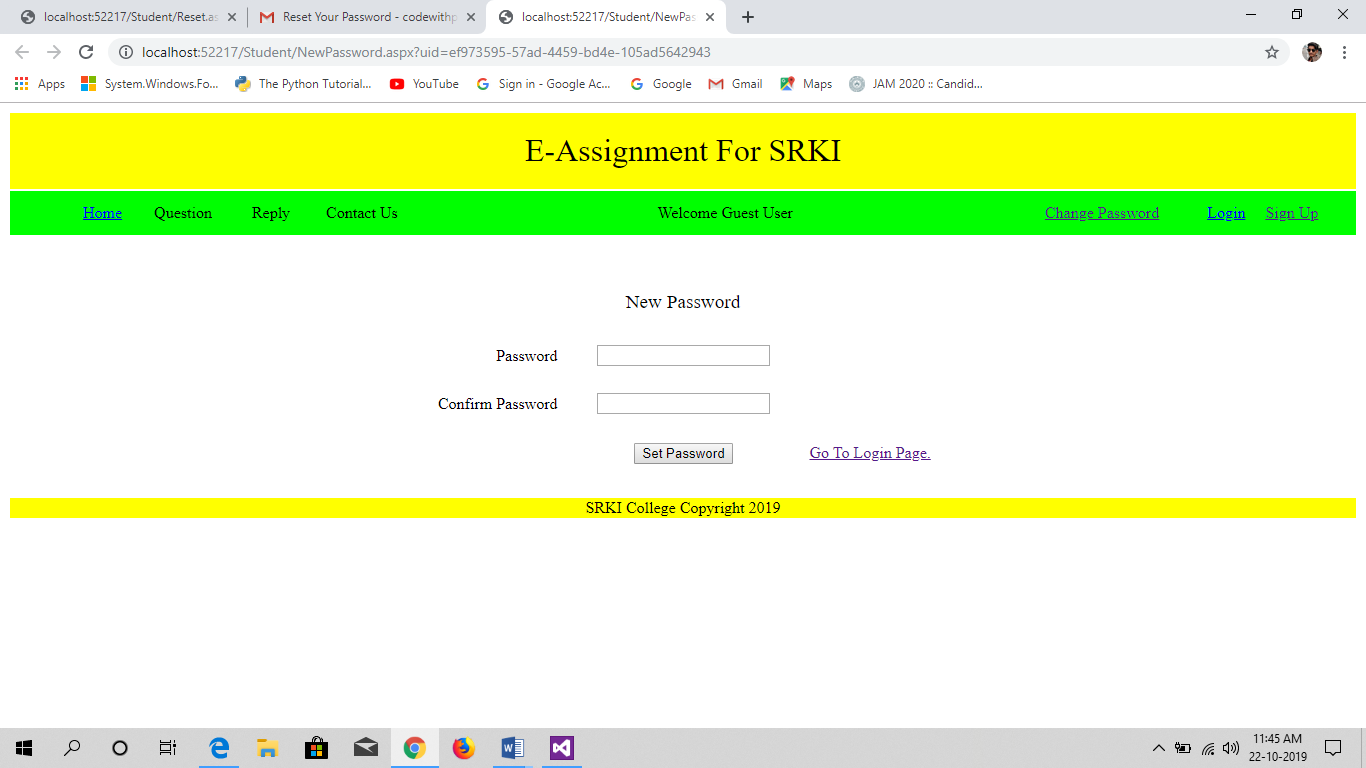
* **Student Registration Form**

****

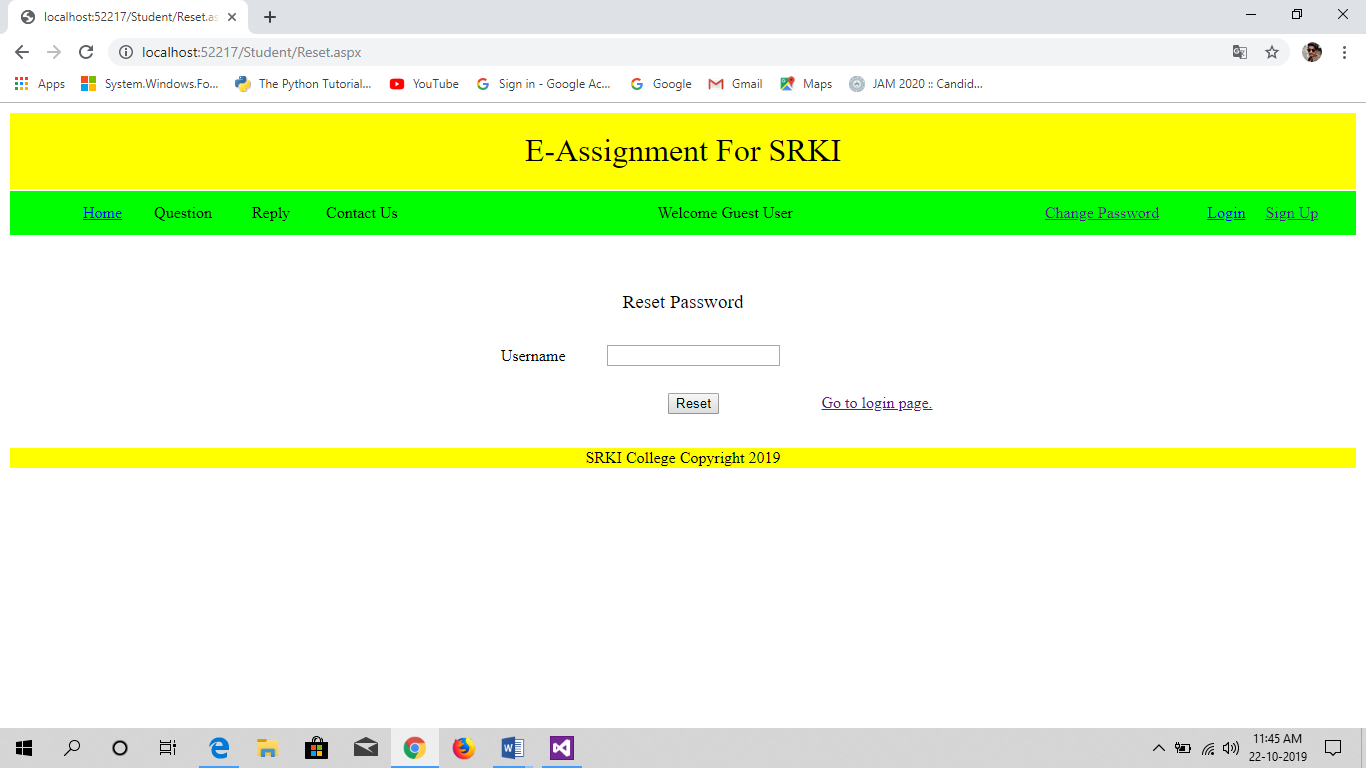
* **Login Form for Users**

****

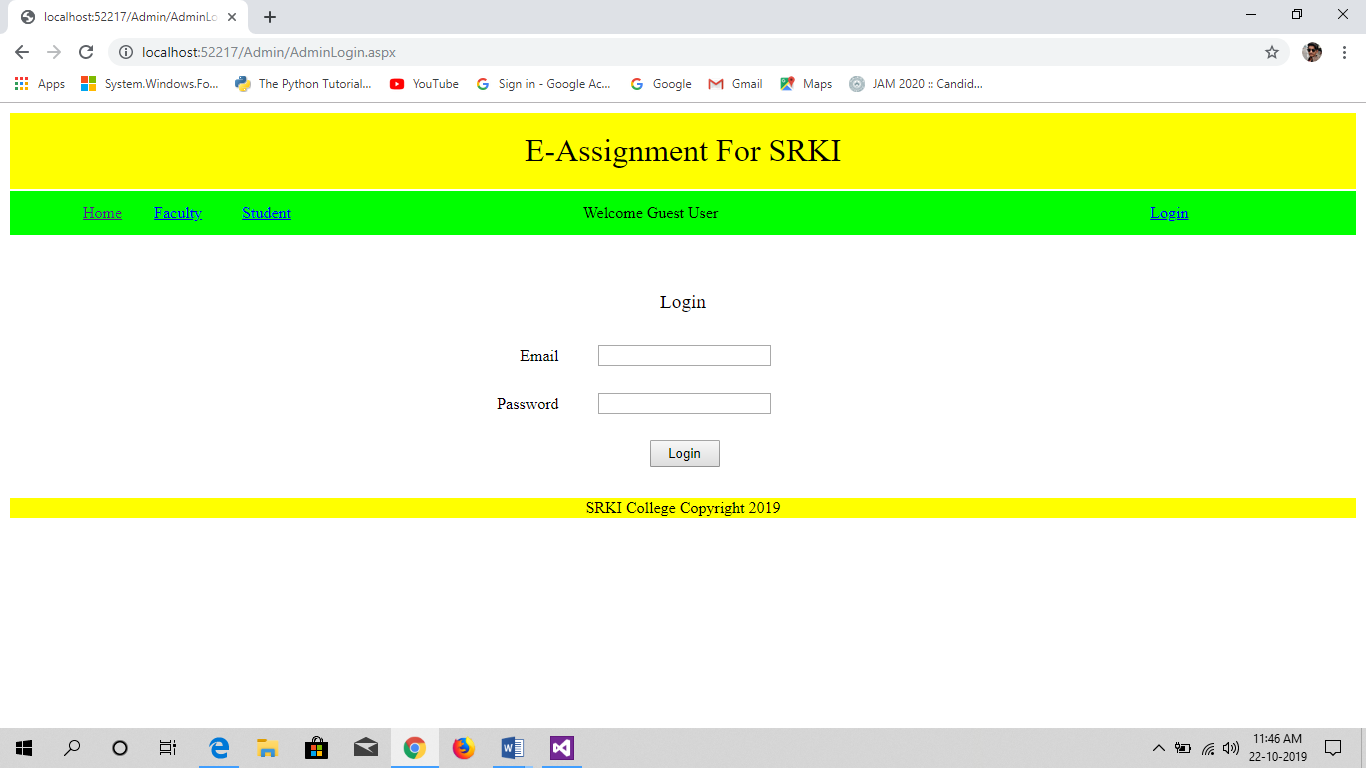
* **New Password**

****

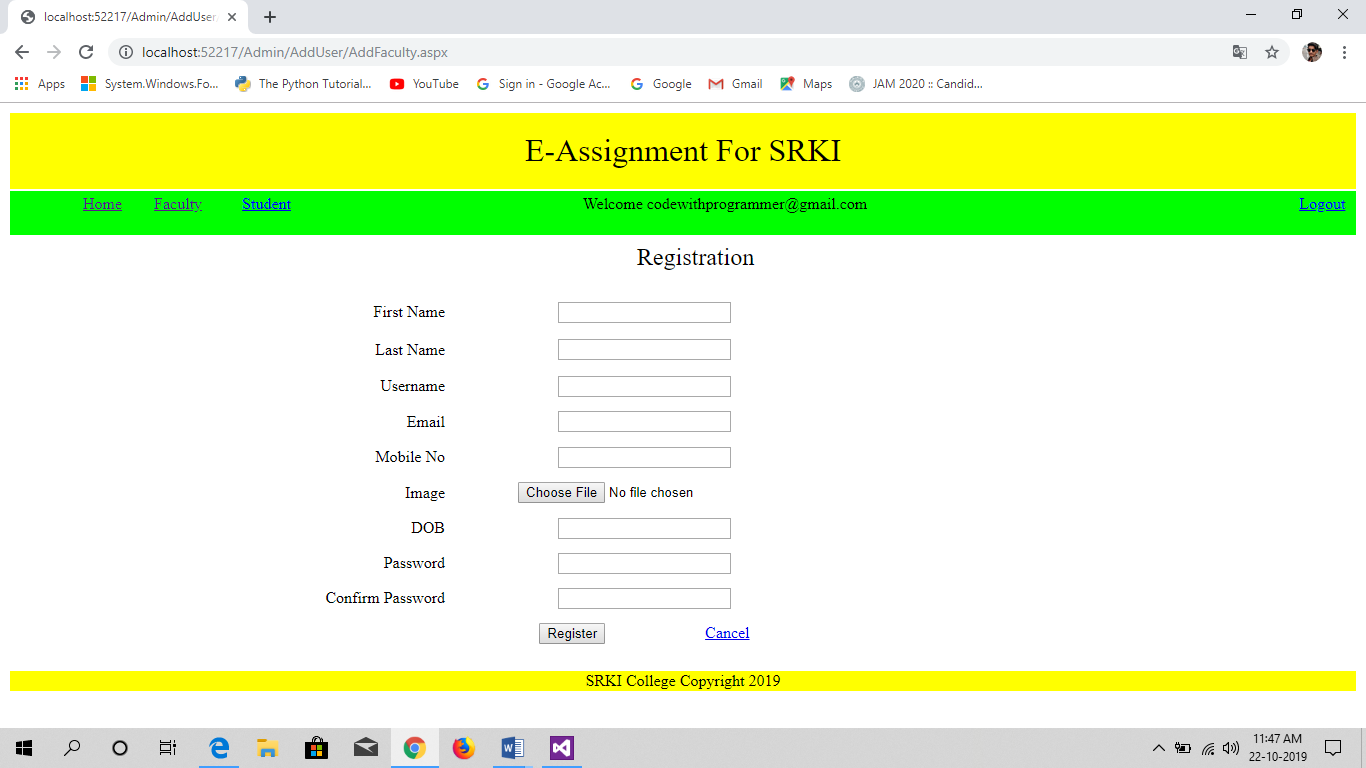
* **Reset Password**

****

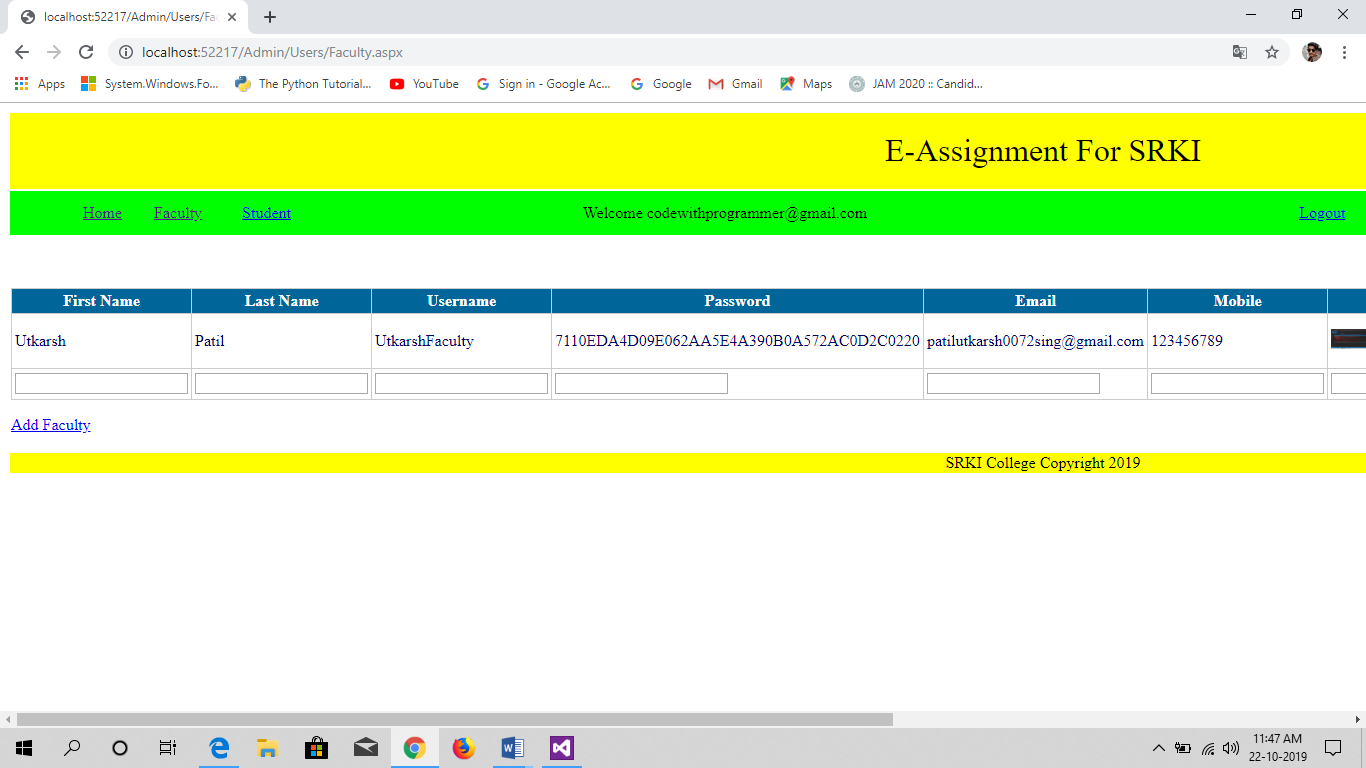
* **Admin Login**

****

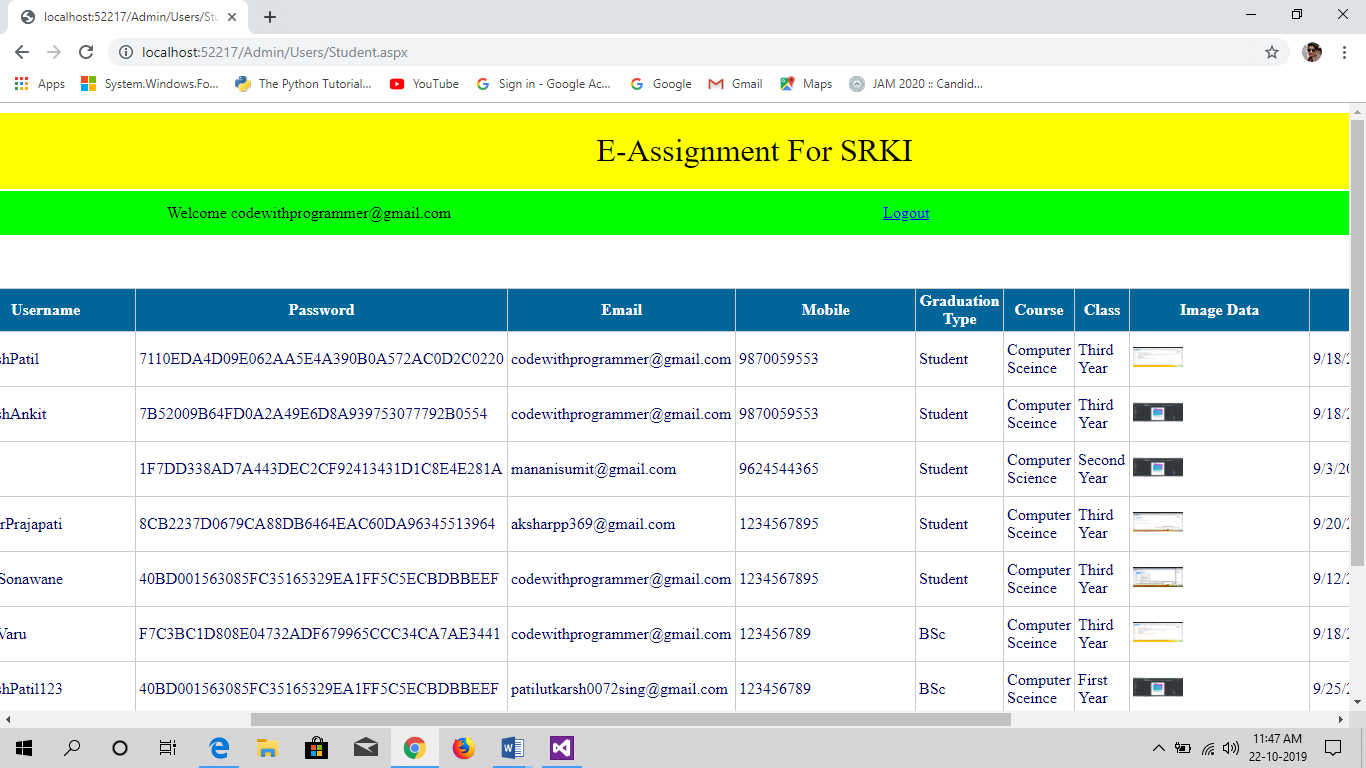
* **Faculty Registration Form**

****

* **Admin Panel for Faculty**

****

* **Admin Panel for Student**

****

**5. Coding**

**5.1 Coding styles / Code snippets**

* **Student Registration**

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;

using System.Configuration;

using System.IO;

using System.Drawing;

using System.Net.Mail;

using System.Web.Security;

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

{

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

{

if (!IsPostBack)

{

string CS = ConfigurationManager.ConnectionStrings["EMSConnectionString"].ConnectionString;

using (SqlConnection con = new SqlConnection(CS))

{

SqlCommand cmd = new SqlCommand("Select \* from tblGraduation", con);

con.Open();

SqlDataReader rdr = cmd.ExecuteReader();

ddlSType.DataSource = rdr;

ddlSType.DataValueField = "GraduationId";

ddlSType.DataTextField = "GraduationType";

ddlSType.DataBind();

ListItem liDefaultType = new ListItem("Select Graduation Type", "-1");

ddlSType.Items.Insert(0, liDefaultType);

}

}

}

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

{

if (Page.IsValid)

{

HttpPostedFile postedFile = fupSImage.PostedFile;

string filename = Path.GetFileName(postedFile.FileName);

string fileExtension = Path.GetExtension(filename);

int fileSize = postedFile.ContentLength;

if (fileExtension.ToLower() == ".jpg" || fileExtension.ToLower() == ".gif"

|| fileExtension.ToLower() == ".png" || fileExtension.ToLower() == ".bmp")

{

if (fileSize < 2097152)

{

Stream stream = postedFile.InputStream;

BinaryReader binaryReader = new BinaryReader(stream);

Byte[] bytes = binaryReader.ReadBytes((int)stream.Length);

string ConnectionString = ConfigurationManager.ConnectionStrings["EMSConnectionString"].ConnectionString;

using (SqlConnection con = new SqlConnection(ConnectionString))

{

SqlCommand cmd = new SqlCommand("spRegisterUser", con);

cmd.CommandType = CommandType.StoredProcedure;

string encryptedpassword = FormsAuthentication.HashPasswordForStoringInConfigFile(txtSPass.Text, "SHA1");

cmd.Parameters.AddWithValue("@FName", txtSFName.Text);

cmd.Parameters.AddWithValue("@LName", txtSLName.Text);

cmd.Parameters.AddWithValue("@Username", txtSUsername.Text);

cmd.Parameters.AddWithValue("@Password", encryptedpassword);

cmd.Parameters.AddWithValue("@Email", txtSEmail.Text);

cmd.Parameters.AddWithValue("@Mobile", txtSMobile.Text);

cmd.Parameters.AddWithValue("@GraduationType", ddlSType.SelectedItem.Text);

cmd.Parameters.AddWithValue("@Course", ddlSCourse.SelectedItem.Text);

cmd.Parameters.AddWithValue("@Class", ddlSClass.SelectedItem.Text);

cmd.Parameters.AddWithValue("@DOB", txtSDOB.Text);

SqlParameter paramImageName = new SqlParameter()

{

ParameterName = @"ImageName",

Value = filename

};

cmd.Parameters.Add(paramImageName);

SqlParameter paramSize = new SqlParameter()

{

ParameterName = "@Size",

Value = fileSize

};

cmd.Parameters.Add(paramSize);

SqlParameter paramImageData = new SqlParameter()

{

ParameterName = "@ImageData",

Value = bytes

};

cmd.Parameters.Add(paramImageData);

con.Open();

int ReturnCode = (int)cmd.ExecuteScalar();

if (ReturnCode == -1)

{

lblMessage.Text = "User Name already in use, please choose another user name";

}

else

{

cmd.ExecuteNonQuery();

MailMessage mailMessage = new MailMessage("codewithprogrammer@gmail.com", txtSEmail.Text);

mailMessage.Body = "Hello, " + txtSFName.Text + " " + txtSLName.Text + "\n\tYou are successfully registration on E mentoring system.\n\nWith Regards,\ncodewithprogrammer.";

mailMessage.Subject = "E Mentoring System Registration";

SmtpClient smtpClient = new SmtpClient();

smtpClient.Send(mailMessage);

lblMessage.Text = "Insert Successfully and check your Email.";

}

}

}

else

{

lblMessage.ForeColor = Color.Red;

lblMessage.Text = "Image size more than 2 MB not allowed.";

}

}

else

{

lblMessage.ForeColor = Color.Green;

lblMessage.Text = "Image file not having .png, .jpg, .bmp, .gif extension.";

}

}

}

protected void ddlSType\_SelectedIndexChanged(object sender, EventArgs e)

{

string CS = ConfigurationManager.ConnectionStrings["EMSConnectionString"].ConnectionString;

using (SqlConnection con = new SqlConnection(CS))

{

SqlCommand cmd = new SqlCommand("Select \* from tblCourse where GraduationId = '" + ddlSType.SelectedValue + "'", con);

con.Open();

SqlDataReader rdr = cmd.ExecuteReader();

ddlSCourse.DataSource = rdr;

ddlSCourse.DataValueField = "CourseId";

ddlSCourse.DataTextField = "CourseName";

ddlSCourse.DataBind();

ListItem liDefault = new ListItem("Select Course", "-1");

ddlSCourse.Items.Insert(0, liDefault);

}

}

protected void ddlSCourse\_SelectedIndexChanged(object sender, EventArgs e)

{

string CS = ConfigurationManager.ConnectionStrings["EMSConnectionString"].ConnectionString;

using (SqlConnection con = new SqlConnection(CS))

{

SqlCommand cmd = new SqlCommand("Select \* from tblClass where CourseId = '" + ddlSCourse.SelectedValue + "'", con);

con.Open();

SqlDataReader rdr = cmd.ExecuteReader();

ddlSClass.DataSource = rdr;

ddlSClass.DataValueField = "ClassId";

ddlSClass.DataTextField = "Class";

ddlSClass.DataBind();

ListItem liDefault = new ListItem("Select Class", "-1");

ddlSClass.Items.Insert(0, liDefault);

}

}

}

* **Reset Password**

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;

using System.Configuration;

using System.IO;

using System.Drawing;

using System.Net.Mail;

using System.Web.Security;

using System.Text;

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

{

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

{

}

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

{

string CS = ConfigurationManager.ConnectionStrings["EMSConnectionString"].ConnectionString;

using (SqlConnection con = new SqlConnection(CS))

{

SqlCommand cmd = new SqlCommand("spResetPassword", con);

cmd.CommandType = CommandType.StoredProcedure;

SqlParameter paramUsername = new SqlParameter("@Username", txtForgetEmail.Text);

cmd.Parameters.Add(paramUsername);

con.Open();

SqlDataReader dr = cmd.ExecuteReader();

while (dr.Read())

{

if (Convert.ToBoolean(dr["ReturnCode"]))

{

SendPasswordResetEmail(dr["Email"].ToString(), txtForgetEmail.Text, dr["UniqueId"].ToString());

lblForgetMessage.Text = "An email with instructions to reset your password is sent to your registered email";

}

else

{

lblForgetMessage.ForeColor = System.Drawing.Color.Red;

lblForgetMessage.Text = "Username not found!";

}

}

}

}

private void SendPasswordResetEmail(string ToEmail, string UserName, string UniqueId)

{

// MailMessage class is present is System.Net.Mail namespace

MailMessage mailMessage = new MailMessage("codewithprogrammer@gmail.com", ToEmail);

// StringBuilder class is present in System.Text namespace

StringBuilder sbEmailBody = new StringBuilder();

sbEmailBody.Append("Dear " + UserName + ",<br/><br/>");

sbEmailBody.Append("Please click on the following link to reset your password");

sbEmailBody.Append("<br/>"); sbEmailBody.Append("http://localhost:52217/Student/NewPassword.aspx?uid=" + UniqueId);

sbEmailBody.Append("<br/><br/>");

sbEmailBody.Append("<b>By Utkarsh Patil.</b>");

mailMessage.IsBodyHtml = true;

mailMessage.Body = sbEmailBody.ToString();

mailMessage.Subject = "Reset Your Password";

SmtpClient smtpClient = new SmtpClient();

smtpClient.Send(mailMessage);

}

}

* **New Password**

using System;

using System.Collections.Generic;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

using System.Linq;

using System.Web;

using System.Web.Security;

using System.Web.UI;

using System.Web.UI.WebControls;

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

{

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

{

if (!IsPostBack)

{

if (!IsPasswordResetLinkValid())

{

lblLinkError.ForeColor = System.Drawing.Color.Red;

lblLinkError.Text = "Password Reset link has expired or is invalid";

}

}

}

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

{

if (ChangeUserPassword())

{

lblLinkError.Text = "Password Changed Successfully!";

}

else

{

lblLinkError.ForeColor = System.Drawing.Color.Red;

lblLinkError.Text = "Password Reset link has expired or is invalid";

}

}

private bool ExecuteSP(string SPName, List<SqlParameter> SPParameters)

{

string CS = ConfigurationManager.ConnectionStrings["EMSConnectionString"].ConnectionString;

using (SqlConnection con = new SqlConnection(CS))

{

SqlCommand cmd = new SqlCommand(SPName, con);

cmd.CommandType = CommandType.StoredProcedure;

foreach (SqlParameter parameter in SPParameters)

{

cmd.Parameters.Add(parameter);

}

con.Open();

return Convert.ToBoolean(cmd.ExecuteScalar());

}

}

private bool IsPasswordResetLinkValid()

{

List<SqlParameter> paramList = new List<SqlParameter>()

{

new SqlParameter()

{

ParameterName = "@GUID",

Value = Request.QueryString["uid"]

}

};

return ExecuteSP("spIsPasswordResetLinkValid", paramList);

}

private bool ChangeUserPassword()

{

List<SqlParameter> paramList = new List<SqlParameter>()

{

new SqlParameter()

{

ParameterName = "@GUID",

Value = Request.QueryString["uid"]

},

new SqlParameter()

{

ParameterName = "@Password",

Value = FormsAuthentication.HashPasswordForStoringInConfigFile(txtPass.Text, "SHA1")

}

};

return ExecuteSP("spChangePassword", paramList);

}

}

* **Login**

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;

using System.Configuration;

using System.Web.Security;

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

{

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

{

}

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

{

AuthenticateUser(txtLoginUsername.Text, txtLoginPass.Text);

}

private void AuthenticateUser(string username, string password)

{

// ConfigurationManager class is in System.Configuration namespace

string CS = ConfigurationManager.ConnectionStrings["EMSConnectionString"].ConnectionString;

// SqlConnection is in System.Data.SqlClient namespace

using (SqlConnection con = new SqlConnection(CS))

{

SqlCommand cmd = new SqlCommand("spAuthenticateUser", con);

cmd.CommandType = CommandType.StoredProcedure;

//Formsauthentication is in system.web.security

//sqlparameter is in System.Data namespace

SqlParameter paramUsername = new SqlParameter("@Username", username);

string encryptedpassword = FormsAuthentication.HashPasswordForStoringInConfigFile(password, "SHA1");

SqlParameter paramPassword = new SqlParameter("@Password", encryptedpassword);

cmd.Parameters.Add(paramUsername);

cmd.Parameters.Add(paramPassword);

con.Open();

SqlDataReader dr = cmd.ExecuteReader();

while (dr.Read())

{

int RetryAttempt = Convert.ToInt32(dr["RetryAttempt"]);

if (Convert.ToBoolean(dr["AccountLocked"]))

{

lblLoginError.Text = "Account locked. Please contact administrator";

}

else if (RetryAttempt > 0)

{

int AttemptsLeft = (4 - RetryAttempt);

lblLoginError.Text = "Invalid user name and/or password. " +

AttemptsLeft.ToString() + "attempt(s) left";

}

else if (Convert.ToBoolean(dr["Authenticated"]))

{

Session["Username"] = txtLoginUsername.Text;

Response.Cookies["Username"].Value = txtLoginUsername.Text;

Response.Cookies["Password"].Value = txtLoginPass.Text;

Response.Cookies["Username"].Expires = DateTime.Now.AddDays(1);

Response.Cookies["Password"].Expires = DateTime.Now.AddDays(1);

FormsAuthentication.RedirectFromLoginPage(txtLoginUsername.Text, chkRemember.Checked);

Response.Redirect("~/Default.aspx");

}

else

{

lblLoginError.Text = "Invalid User Name and/or Pasword";

}

}

}

}

}

* **Change Password**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Web.Security;

using System.Data.SqlClient;

using System.Configuration;

using System.Data;

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

{

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

{

if (Request.QueryString["uid"] == null && User.Identity.Name == "")

{

Response.Redirect("~/Student/Login.aspx");

}

if (!IsPostBack)

{

if (Request.QueryString["uid"] != null)

{

if (!IsPasswordResetLinkValid())

{

lblError.ForeColor = System.Drawing.Color.Red;

lblError.Text = "Password Reset link has expired or is invalid";

}

//trCurrentPassword.Visible = false;

}

else if (User.Identity.Name != "")

{

//trCurrentPassword.Visible = true;

}

}

}

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

{

if ((Request.QueryString["uid"] != null && ChangeUserPassword()) ||

(User.Identity.Name != "" && ChangeUserPasswordUsingCurrentPassword()))

{

lblError.Text = "Password Changed Successfully!";

}

else

{

lblError.ForeColor = System.Drawing.Color.Red;

//if (trCurrentPassword.Visible)

//{

// lblLoginError.Text = "Invalid Current Password!";

//}

//else

//{

// lblLoginError.Text = "Password Reset link has expired or is invalid";

//}

}

}

private bool ExecuteSP(string SPName, List<SqlParameter> SPParameters)

{

string CS = ConfigurationManager.ConnectionStrings["EMSConnectionString"].ConnectionString;

using (SqlConnection con = new SqlConnection(CS))

{

SqlCommand cmd = new SqlCommand(SPName, con);

cmd.CommandType = CommandType.StoredProcedure;

foreach (SqlParameter parameter in SPParameters)

{

cmd.Parameters.Add(parameter);

}

con.Open();

return Convert.ToBoolean(cmd.ExecuteScalar());

}

}

private bool IsPasswordResetLinkValid()

{

List<SqlParameter> paramList = new List<SqlParameter>()

{

new SqlParameter()

{

ParameterName = "@GUID",

Value = Request.QueryString["uid"]

}

};

return ExecuteSP("spIsPasswordResetLinkValid", paramList);

}

private bool ChangeUserPassword()

{

List<SqlParameter> paramList = new List<SqlParameter>()

{

new SqlParameter()

{

ParameterName = "@GUID",

Value = Request.QueryString["uid"]

},

new SqlParameter()

{

ParameterName = "@Password",

Value = FormsAuthentication.HashPasswordForStoringInConfigFile(txtPass.Text, "SHA1")

}

};

return ExecuteSP("spChangePassword", paramList);

}

private bool ChangeUserPasswordUsingCurrentPassword()

{

List<SqlParameter> paramList = new List<SqlParameter>()

{

new SqlParameter()

{

ParameterName = "@Username",

Value = User.Identity.Name

},

new SqlParameter()

{

ParameterName = "@CurrentPassword",

Value = FormsAuthentication.HashPasswordForStoringInConfigFile(txtCurrentPass.Text, "SHA1")

},

new SqlParameter()

{

ParameterName = "@NewPassword",

Value = FormsAuthentication.HashPasswordForStoringInConfigFile(txtPass.Text, "SHA1")

}

};

return ExecuteSP("spChangePasswordUsingCurrentPassword", paramList);

}

}

* **Admin Login**

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;

using System.Configuration;

using System.Web.Security;

public partial class Admin\_AdminLogin : System.Web.UI.Page

{

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

{

if (!IsPostBack)

{

Session.Abandon();

}

}

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

{

AuthenticateAdmin(txtLoginEmail.Text, txtLoginPass.Text);

}

private void AuthenticateAdmin(string username, string password)

{

// ConfigurationManager class is in System.Configuration namespace

string CS = ConfigurationManager.ConnectionStrings["EMSConnectionString"].ConnectionString;

// SqlConnection is in System.Data.SqlClient namespace

using (SqlConnection con = new SqlConnection(CS))

{

SqlCommand cmd = new SqlCommand("spAuthenticateAdmin", con);

cmd.CommandType = CommandType.StoredProcedure;

//Formsauthentication is in system.web.security

//sqlparameter is in System.Data namespace

SqlParameter paramUsername = new SqlParameter("@Email", username);

string encryptedpassword = FormsAuthentication.HashPasswordForStoringInConfigFile(password, "SHA1");

SqlParameter paramPassword = new SqlParameter("@Password", encryptedpassword);

cmd.Parameters.Add(paramUsername);

cmd.Parameters.Add(paramPassword);

con.Open();

SqlDataReader dr = cmd.ExecuteReader();

while (dr.Read())

{

int RetryAttempt = Convert.ToInt32(dr["RetryAttempt"]);

if (Convert.ToBoolean(dr["AccountLocked"]))

{

lblLoginError.Text = "Account locked. Please contact administrator";

}

else if (RetryAttempt > 0)

{

int AttemptsLeft = (4 - RetryAttempt);

lblLoginError.Text = "Invalid user name and/or password. " +

AttemptsLeft.ToString() + "attempt(s) left";

}

else if (Convert.ToBoolean(dr["Authenticated"]))

{

Session["Email"] = txtLoginEmail.Text;

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

}

else

{

lblLoginError.Text = "Invalid User Name and/or Pasword";

}

}

}

}

}

* **Add Faculty**

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;

using System.Configuration;

using System.IO;

using System.Drawing;

using System.Net.Mail;

using System.Web.Security;

public partial class Admin\_AddUser\_AddFaculty : System.Web.UI.Page

{

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

{

}

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

{

if (Page.IsValid)

{

HttpPostedFile postedFile = fupSImage.PostedFile;

string filename = Path.GetFileName(postedFile.FileName);

string fileExtension = Path.GetExtension(filename);

int fileSize = postedFile.ContentLength;

if (fileExtension.ToLower() == ".jpg" || fileExtension.ToLower() == ".gif"

|| fileExtension.ToLower() == ".png" || fileExtension.ToLower() == ".bmp")

{

if (fileSize < 2097152)

{

Stream stream = postedFile.InputStream;

BinaryReader binaryReader = new BinaryReader(stream);

Byte[] bytes = binaryReader.ReadBytes((int)stream.Length);

string ConnectionString = ConfigurationManager.ConnectionStrings["EMSConnectionString"].ConnectionString;

using (SqlConnection con = new SqlConnection(ConnectionString))

{

SqlCommand cmd = new SqlCommand("spRegisterFaculty", con);

cmd.CommandType = CommandType.StoredProcedure;

string encryptedpassword = FormsAuthentication.HashPasswordForStoringInConfigFile(txtSPass.Text, "SHA1");

cmd.Parameters.AddWithValue("@FName", txtSFName.Text);

cmd.Parameters.AddWithValue("@LName", txtSLName.Text);

cmd.Parameters.AddWithValue("@Username", txtSUsername.Text);

cmd.Parameters.AddWithValue("@Password", encryptedpassword);

cmd.Parameters.AddWithValue("@Email", txtSEmail.Text);

cmd.Parameters.AddWithValue("@Mobile", txtSMobile.Text);

cmd.Parameters.AddWithValue("@DOB", txtSDOB.Text);

SqlParameter paramImageName = new SqlParameter()

{

ParameterName = @"ImageName",

Value = filename

};

cmd.Parameters.Add(paramImageName);

SqlParameter paramSize = new SqlParameter()

{

ParameterName = "@Size",

Value = fileSize

};

cmd.Parameters.Add(paramSize);

SqlParameter paramImageData = new SqlParameter()

{

ParameterName = "@ImageData",

Value = bytes

};

cmd.Parameters.Add(paramImageData);

con.Open();

int ReturnCode = (int)cmd.ExecuteScalar();

if (ReturnCode == -1)

{

lblMessage.Text = "User Name already in use, please choose another user name";

}

else

{

cmd.ExecuteNonQuery();

/\*

MailMessage mailMessage = new MailMessage("codewithprogrammer@gmail.com", txtSEmail.Text);

mailMessage.Body = "Hello, " + txtSFName.Text + " " + txtSLName.Text + "\n\tYou are successfully registration on E mentoring system.\n\nWith Regards,\ncodewithprogrammer.";

mailMessage.Subject = "E Mentoring System Registration";

SmtpClient smtpClient = new SmtpClient();

smtpClient.Send(mailMessage);\*/

lblMessage.Text = "Successfully added.";

}

}

}

else

{

lblMessage.ForeColor = Color.Red;

lblMessage.Text = "Image size more than 2 MB not allowed.";

}

}

else

{

lblMessage.ForeColor = Color.Green;

lblMessage.Text = "Image file not having .png, .jpg, .bmp, .gif extension.";

}

}

}

}

* **Add Student**

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;

using System.Configuration;

using System.IO;

using System.Drawing;

using System.Net.Mail;

using System.Web.Security;

public partial class Admin\_AddUser\_AddStudent : System.Web.UI.Page

{

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

{

if (!IsPostBack)

{

string CS = ConfigurationManager.ConnectionStrings["EMSConnectionString"].ConnectionString;

using (SqlConnection con = new SqlConnection(CS))

{

SqlCommand cmd = new SqlCommand("Select \* from tblGraduation", con);

con.Open();

SqlDataReader rdr = cmd.ExecuteReader();

ddlSType.DataSource = rdr;

ddlSType.DataValueField = "GraduationId";

ddlSType.DataTextField = "GraduationType";

ddlSType.DataBind();

ListItem liDefaultType = new ListItem("Select Graduation Type", "-1");

ddlSType.Items.Insert(0, liDefaultType);

}

}

}

protected void ddlSType\_SelectedIndexChanged(object sender, EventArgs e)

{

string CS = ConfigurationManager.ConnectionStrings["EMSConnectionString"].ConnectionString;

using (SqlConnection con = new SqlConnection(CS))

{

SqlCommand cmd = new SqlCommand("Select \* from tblCourse where GraduationId = '" + ddlSType.SelectedValue + "'", con);

con.Open();

SqlDataReader rdr = cmd.ExecuteReader();

ddlSCourse.DataSource = rdr;

ddlSCourse.DataValueField = "CourseId";

ddlSCourse.DataTextField = "CourseName";

ddlSCourse.DataBind();

ListItem liDefault = new ListItem("Select Course", "-1");

ddlSCourse.Items.Insert(0, liDefault);

}

}

protected void ddlSCourse\_SelectedIndexChanged(object sender, EventArgs e)

{

string CS = ConfigurationManager.ConnectionStrings["EMSConnectionString"].ConnectionString;

using (SqlConnection con = new SqlConnection(CS))

{

SqlCommand cmd = new SqlCommand("Select \* from tblClass where CourseId = '" + ddlSCourse.SelectedValue + "'", con);

con.Open();

SqlDataReader rdr = cmd.ExecuteReader();

ddlSClass.DataSource = rdr;

ddlSClass.DataValueField = "ClassId";

ddlSClass.DataTextField = "Class";

ddlSClass.DataBind();

ListItem liDefault = new ListItem("Select Class", "-1");

ddlSClass.Items.Insert(0, liDefault);

}

}

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

{

if (Page.IsValid)

{

HttpPostedFile postedFile = fupSImage.PostedFile;

string filename = Path.GetFileName(postedFile.FileName);

string fileExtension = Path.GetExtension(filename);

int fileSize = postedFile.ContentLength;

if (fileExtension.ToLower() == ".jpg" || fileExtension.ToLower() == ".gif"

|| fileExtension.ToLower() == ".png" || fileExtension.ToLower() == ".bmp")

{

if (fileSize < 2097152)

{

Stream stream = postedFile.InputStream;

BinaryReader binaryReader = new BinaryReader(stream);

Byte[] bytes = binaryReader.ReadBytes((int)stream.Length);

string ConnectionString = ConfigurationManager.ConnectionStrings["EMSConnectionString"].ConnectionString;

using (SqlConnection con = new SqlConnection(ConnectionString))

{

SqlCommand cmd = new SqlCommand("spRegisterUser", con);

cmd.CommandType = CommandType.StoredProcedure;

string encryptedpassword = FormsAuthentication.HashPasswordForStoringInConfigFile(txtSPass.Text, "SHA1");

cmd.Parameters.AddWithValue("@FName", txtSFName.Text);

cmd.Parameters.AddWithValue("@LName", txtSLName.Text);

cmd.Parameters.AddWithValue("@Username", txtSUsername.Text);

cmd.Parameters.AddWithValue("@Password", encryptedpassword);

cmd.Parameters.AddWithValue("@Email", txtSEmail.Text);

cmd.Parameters.AddWithValue("@Mobile", txtSMobile.Text);

cmd.Parameters.AddWithValue("@GraduationType", ddlSType.SelectedItem.Text);

cmd.Parameters.AddWithValue("@Course", ddlSCourse.SelectedItem.Text);

cmd.Parameters.AddWithValue("@Class", ddlSClass.SelectedItem.Text);

cmd.Parameters.AddWithValue("@DOB", txtSDOB.Text);

SqlParameter paramImageName = new SqlParameter()

{

ParameterName = @"ImageName",

Value = filename

};

cmd.Parameters.Add(paramImageName);

SqlParameter paramSize = new SqlParameter()

{

ParameterName = "@Size",

Value = fileSize

};

cmd.Parameters.Add(paramSize);

SqlParameter paramImageData = new SqlParameter()

{

ParameterName = "@ImageData",

Value = bytes

};

cmd.Parameters.Add(paramImageData);

con.Open();

int ReturnCode = (int)cmd.ExecuteScalar();

if (ReturnCode == -1)

{

lblMessage.Text = "User Name already in use, please choose another user name";

}

else

{

cmd.ExecuteNonQuery();

/\*

MailMessage mailMessage = new MailMessage("codewithprogrammer@gmail.com", txtSEmail.Text);

mailMessage.Body = "Hello, " + txtSFName.Text + " " + txtSLName.Text + "\n\tYou are successfully registration on E mentoring system.\n\nWith Regards,\ncodewithprogrammer.";

mailMessage.Subject = "E Mentoring System Registration";

SmtpClient smtpClient = new SmtpClient();

smtpClient.Send(mailMessage);\*/

lblMessage.Text = "Successfully added.";

}

}

}

else

{

lblMessage.ForeColor = Color.Red;

lblMessage.Text = "Image size more than 2 MB not allowed.";

}

}

else

{

lblMessage.ForeColor = Color.Green;

lblMessage.Text = "Image file not having .png, .jpg, .bmp, .gif extension.";

}

}

}

}

**6. Testing, Validation**

|  |  |  |
| --- | --- | --- |
| **Test No.** | **Test case Step** | **Expected Result** |
| **1.** | Input all field in Registration form (Require validator) | User register on it |
| **2.** | If any field will empty | Field is required. |
| **3.** | Input number in name field | Only alphabet allowed |
| **4.** | Input valid email id and valid  Password | Login successful |
| **5.** | Input invalid email id and password | Login not successful |
| **6.** | Password and confirm password Compare by compare validator in Registration | Confirm password should match otherwise it throw error message |
| **7.** | Email id should be in proper email format in Registration and login by using textbox email property | Email should be in the form of abcde@xyz.com otherwise it throws error |
| **8.** | Mobile no. should be in digit in all register form | Only number allowed not alphabet |
| **9.** | Upload invalid image file | Image file not having png, jpg, bmp, gif extension |
| **10.** | Image size more than 2 MB | Image size more than 2 MB not allowed |

**7.New Tools and Technology**

* **Bootstrap**
* Bootstrap is responsive, mobile-first, prevailing, and frontend framework, which is developed along with CSS, JavaScript and HTML. Bootstrap has many benefits from Scratch for every web development project, and such reason is the huge number for resources accessible for bootstrap.
* **Ajax Toolkit**
* It is a joint effort between Microsoft and the ASP.NET AJAX community that provides a powerful infrastructure to write reusable, customizable and extensible ASP.NET AJAX extenders and controls, as well as a rich array of controls that can be used out of the box to create an interactive web experience.

**8.System Limitations**

* Faculty can’t register itself.
* User can’t delete any other person account.
* User can’t access account after lock account.
* User can’t access account after lock account.
* Student can’t delete any uploaded assignment.

**9. Future Enhancement & Opportunities**

* Student get notification after faculty uploaded assignment related to student subject.
* Faculty also get notification after student submitted assignment.
* Faculty can give grade to the student according their work.
* Generate top 3 student who having highest grade.

**10. Bibliography and References**

* [www.getbootstrap.com](http://www.getbootstrap.com)
* [www.nuget.com](http://www.nuget.com)
* [www.stackoverflow.com](http://www.stackoverflow.com)
* [www.aspsnippets.com](http://www.aspsnippets.com)
* [www.youtube.com](http://www.youtube.com)