

Using LINQ to SQl with Multiple Tables

create table Designation(DesignationId int Constraint Did\_PK Primary key identity(10,10),Dname varchar(50))

create table Qualification(QualificationId int Constraint Qid\_PK Primary Key identity(5,5),Qname varchar(50))

create table Employee(Eid int Constraint Eid\_PK Primary Key Identity(1001,1),Ename varchar(50),Salary Money,Email varchar(50),Mobile varchar(20),Manager varchar(20), DesignationId Int Constraint Did\_FK References Designation(DesignationId), QualificationId Int Constraint Qid\_FK References Qualification(QualificationId),Status Bit Not Null Default 1)

insert into Designation values('CEO');

insert into Designation values('Trainee');

insert into Designation values('SDE');

insert into Designation values('HR');

insert into Designation values('Analyst');

insert into Qualification values('MBA')

insert into Qualification values('BE')

insert into Qualification values('CA')

insert into Qualification values('BSC'

Create Procedure Get\_Designation

as

begin

Select DesignationId ,Dname from Designation

End

Create Procedure Get\_Qualification

as

begin

Select QualificationId ,Qname from Qualification

End

create Procedure Employee\_Select(@Eid int=null,@Status Bit=Null)

As

Begin

if @Eid Is Null And @Status Is Null

select Eid,Ename,Salary,Email,Mobile,Manager,Did,Qid from Employee Order By Eid

Else if @Eid Is Null And @Status Is Not Null

Select Eid,Ename,Salary,Email,Mobile,Manager, Did,Qid from Employee Where Status=@Status Order by Eid

Else if @Eid Is Not Null And @Status Is Null

select Eid,Ename,Salary,Email,Mobile,Manager, Did,Qid from Employee where Eid=@Eid

Else if @Eid Is Not Null And @Status Is Not Null

select Eid,Ename,Salary,Email,Mobile,Manager, Did,Qid from Employee where Eid=@Eid And Status=@Status

End;

create Procedure Employee\_Update(@Eid int,@Ename varchar(50),@Salary Money,@Email Varchar(50),@Mobile varchar(20),@Manager varchar(20),@Did int,@Qid int)

As

Update Employee Set Ename=@Ename,Salary=@Salary,Email=@Email,Mobile=@Mobile,Manager=@Manager,Did=@Did,Qid=@Qid where Eid=@Eid

create Procedure Employee\_Insert(@Eid int,@Ename varchar(50),@Salary Money,@Email Varchar(50),@Mobile varchar(20),@Manager varchar(20),@Did int,@Qid int)

As

Insert Into Employee(Eid,Ename,Salary,Email,Mobile,Manager,Did,Qid) values(@Eid,@Ename,@Salary,@Email,@Mobile,@Manager,@Did,@Qid)

create Procedure Employee\_Delete(@Eid int)

as

Update Employee Set Status=0 Where Eid=@Eid;

Add Linq to sql in Model folder drag and drop tables left Side of OR Table.

Add class under model folder name it as EmpDQ

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.Linq;

using System.Web;

using System.Web.Mvc;

namespace MVCWithLinq3.Models

{

public class EmpDQ

{

public int Eid { get; set; }

[Required(ErrorMessage ="Can't left empty.")]

public string Ename { get; set; }

public decimal? Salary { get; set; }

public string Email { get; set; }

public string Mobile { get; set; }

public string Manager { get; set; }

public int DesignationId { get; set; }

public string Dname { get; set; }

public int QualificationId { get; set; }

public string Qname { get; set; }

public List<SelectListItem> Designations { get; set; }

public List<SelectListItem> Qualifications { get; set; }

}

}

Add EmloyeeDAL in model folder

using System;

using System.Collections.Generic;

using System.Configuration;

using System.Linq;

using System.Web;

using System.Web.Mvc;

namespace MVCWithLinq3.Models

{

public class EmployeeDAL

{

LINQDataContext dc = new LINQDataContext(ConfigurationManager.ConnectionStrings["LINQConnectionString"].ConnectionString);

public List<SelectListItem> GetDesignation()

{

List<SelectListItem> Designations = new List<SelectListItem>();

foreach (var item in dc.Designations)

{

SelectListItem li = new SelectListItem { Text=item.Dname,Value=item.DesignationId.ToString()};

Designations.Add(li);

}

return Designations;

}

public List<SelectListItem> GetQualification()

{

List<SelectListItem> Qualifications=new List<SelectListItem>();

foreach(var item in dc.Qualifications)

{

SelectListItem li = new SelectListItem {Text=item.Qname,Value=item.QualificationId.ToString() };

Qualifications.Add(li);

}

return Qualifications;

}

public List<EmpDQ> GetEmployees()

{

var Records = from E in dc.Employees

join D in dc.Designations on E.DesignationId equals D.DesignationId

join Q in dc.Qualifications on E.QualificationId equals Q.QualificationId

where E.Status == true

select new { E.Eid, E.Ename, E.Salary, E.Email, E.Mobile, E.Manager, D.DesignationId, D.Dname, Q.QualificationId, Q.Qname };

List<EmpDQ> Emps = new List<EmpDQ>();

foreach(var Record in Records)

{

EmpDQ emp = new EmpDQ

{

Eid=Record.Eid,

Ename=Record.Ename,

Salary=Record.Salary,

Email=Record.Email,

Mobile=Record.Mobile,

Manager=Record.Manager,

DesignationId=Record.DesignationId,

Dname=Record.Dname,

QualificationId=Record.QualificationId,

Qname=Record.Qname

};

Emps.Add(emp);

}

return Emps;

}

public EmpDQ GetEmployee(int Eid)

{

var Record = (from E in dc.Employees

join D in dc.Designations on E.DesignationId equals D.DesignationId

join Q in dc.Qualifications on E.QualificationId equals Q.QualificationId

where E.Eid == Eid

select new

{

E.Eid,

E.Ename,

E.Salary,E.Email,E.Mobile,E.Manager,D.DesignationId,D.Dname,Q.QualificationId,Q.Qname

}).Single();

EmpDQ emp = new EmpDQ

{

Eid=Record.Eid,

Ename=Record.Ename,

Salary=Record.Salary,

Email=Record.Email,

Mobile=Record.Mobile,

Manager=Record.Manager,

DesignationId=Record.DesignationId,

Dname=Record.Dname,

QualificationId=Record.QualificationId,

Qname=Record.Qname

};

return emp;

}

public void InsertEmployee(EmpDQ employee)

{

Employee emp = new Employee

{

Ename=employee.Ename,

Salary=employee.Salary,

Email=employee.Email,

Mobile=employee.Mobile,

Manager=employee.Manager,

DesignationId=employee.DesignationId,

QualificationId= employee.QualificationId,

Status=true

};

dc.Employees.InsertOnSubmit(emp);

dc.SubmitChanges();

}

public void UpdateEmployee(EmpDQ newValues)

{

Employee oldValues = dc.Employees.Single(E => E.Eid == newValues.Eid);

oldValues.Ename= newValues.Ename;

oldValues.Salary = newValues.Salary;

oldValues.Email = newValues.Email;

oldValues.Mobile = newValues.Mobile;

oldValues.Manager = newValues.Manager;

oldValues.DesignationId = newValues.DesignationId;

oldValues.QualificationId = newValues.QualificationId;

dc.SubmitChanges();

}

public void DeleteEmployee(int Eid)

{

Employee oldValue = dc.Employees.Single(E => E.Eid == Eid);

oldValue.Status = false;

dc.SubmitChanges();

}

}

}

Add a Employee Controller in Controller Folder

using MVCWithLinq3.Models;

using System;

using System.Collections.Generic;

using System.Configuration;

using System.Linq;

using System.Web;

using System.Web.Mvc;

namespace MVCWithLinq3.Controllers

{

public class EmployeeController : Controller

{

EmployeeDAL obj =new EmployeeDAL();

public ViewResult GetAll()

{

var emps = obj.GetEmployees();

return View(emps);

}

public ViewResult GetById(int Eid)

{

var emp = obj.GetEmployee(Eid);

return View(emp);

}

[HttpGet]

public ViewResult AddEmployee()

{

EmpDQ emp = new EmpDQ();

if (ModelState.IsValid)

{

emp.Designations = obj.GetDesignation();

emp.Qualifications = obj.GetQualification();

}

return View(emp);

}

[HttpPost]

public RedirectToRouteResult AddEmployee(EmpDQ emp)

{

obj.InsertEmployee(emp);

return RedirectToAction("GetAll");

}

public ViewResult EditEmployee(int Eid)

{

EmpDQ emp = obj.GetEmployee(Eid);

emp.Designations = obj.GetDesignation();

emp.Qualifications = obj.GetQualification();

return View(emp);

}

[HttpPost]

public RedirectToRouteResult UpdateEmployee(EmpDQ emp)

{

obj.UpdateEmployee(emp);

return RedirectToAction("GetAll");

}

public RedirectToRouteResult DeleteEmployee(int Eid)

{

obj.DeleteEmployee(Eid);

return RedirectToAction("GetAll");

}

}

}

@model IEnumerable<MVCWithLinq3.Models.EmpDQ>

<table border="1" align="center">

<tr>

<th>

@Html.DisplayNameFor(E => E.Eid)

</th>

<th>

@Html.DisplayNameFor(E => E.Ename)

</th>

<th>

@Html.DisplayNameFor(E => E.DesignationId)

</th>

<th>

@Html.DisplayNameFor(E => E.Dname)

</th>

<th>

@Html.DisplayNameFor(E => E.Email)

</th>

<th>

@Html.DisplayNameFor(E => E.Salary)

</th>

<th>

@Html.DisplayNameFor(E => E.QualificationId)

</th>

<th>

@Html.DisplayNameFor(E => E.Qname)

</th>

<th>

@Html.DisplayNameFor(E => E.Manager)

</th>

<th>

@Html.DisplayNameFor(E => E.Mobile)

</th>

<th>Actions</th>

</tr>

@foreach(var employee in Model)

{

<tr>

<td>

@Html.DisplayFor(E => employee.Eid)

</td>

<td>

@Html.DisplayFor(E => employee.Ename)

</td>

<td>

@Html.DisplayFor(E => employee.DesignationId)

</td>

<td>

@Html.DisplayFor(E => employee.Dname)

</td>

<td>

@Html.DisplayFor(E => employee.Email)

</td>

<td>

@Html.DisplayFor(E => employee.Salary)

</td>

<td>

@Html.DisplayFor(E => employee.QualificationId)

</td>

<td>

@Html.DisplayFor(E => employee.Qname)

</td>

<td>

@Html.DisplayFor(E => employee.Manager)

</td>

<td>

@Html.DisplayFor(E => employee.Mobile)

</td>

<td>

@Html.ActionLink("View", "GetById", new {Eid=employee.Eid})

@Html.ActionLink("Edit", "EditEmployee", new {Eid=employee.Eid})

@Html.ActionLink("Delete", "DeleteEmployee", new { Eid = employee.Eid },

new {onclick="return confirm('Are you sure of deleting recird?')"})

</td>

</tr>

}

<tr>

<td colspan="11" align="center">@Html.ActionLink("Add new emp","AddEmployee")</td>

</tr>

</table>

@model MVCWithLinq3.Models.EmpDQ

@{

ViewBag.Title = "GetById";

}

<table boredr="1" align="center">

<tr>

<td>Eid:</td>

<td>@Model.Eid</td>

</tr>

<tr>

<td>Ename:</td>

<td>@Model.Ename</td>

</tr>

<tr>

<td>DesignationId:</td>

<td>@Model.DesignationId</td>

</tr>

<tr>

<td>Designation:</td>

<td>@Model.Dname</td>

</tr>

<tr>

<td>Salary:</td>

<td>@Model.Salary</td>

</tr>

<tr>

<td>Email:</td>

<td>@Model.Email</td>

</tr>

<tr>

<td>Mobile:</td>

<td>@Model.Mobile</td>

</tr>

<tr>

<td>QualificationId:</td>

<td>@Model.QualificationId</td>

</tr>

<tr>

<td>Qualification:</td>

<td>@Model.Qname</td>

</tr>

<tr>

<td>Manager:</td>

<td>@Model.Manager</td>

</tr>

<tr>

<td colspan="2" align="center">

@Html.ActionLink("Back to employee details", "GetAll")

</td>

</tr>

</table>

@model MVCWithLinq3.Models.EmpDQ

@{

ViewBag.Title = "AddEmployee";

}

<style>

.error {

color: red;

}

</style>

<script src="~/Scripts/jquery-3.6.0.min.js"></script>

<script src="~/Scripts/jquery.validate.min.js"></script>

<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>

@using (Html.BeginForm("AddEmployee", "Employee", FormMethod.Post))

{

<div>

@Html.LabelFor(E => E.Ename)<br />

@Html.TextBoxFor(E => E.Ename)

@Html.ValidationMessageFor(E => E.Ename, "", new { @class = "error" })

</div>

<div>

@Html.LabelFor(E => E.Dname)<br />

@Html.DropDownListFor(E => E.DesignationId, Model.Designations, "-Select Designation-")

</div>

<div>

@Html.LabelFor(E => E.Email)<br />

@Html.TextBoxFor(E => E.Email)

</div>

<div>

@Html.LabelFor(E => E.Salary)<br />

@Html.TextBoxFor(E => E.Salary)

</div>

<div>

@Html.LabelFor(E => E.Qname)<br />

@Html.DropDownListFor(E => E.QualificationId, Model.Qualifications, "-Select Qualification-")

</div>

<div>

@Html.LabelFor(E => E.Mobile)<br />

@Html.TextBoxFor(E => E.Mobile)

</div>

<div>

@Html.LabelFor(E => E.Manager)<br />

@Html.TextBoxFor(E => E.Manager)

</div>

<div>

<input type="submit" value="Submit" name="btnSave" />

<input type="reset" value="Reset" name="btnReset" />

</div>

}

@model MVCWithLinq3.Models.EmpDQ

@{

ViewBag.Title = "EditEmployee";

}

@using (Html.BeginForm("UpdateEmployee", "Employee", FormMethod.Post))

{

<div>

@Html.LabelFor(E => E.Ename)<br />

@Html.TextBoxFor(E=>E.Eid, new { @readonly = "true" })

</div>

<div>

@Html.LabelFor(E => E.Ename)<br />

@Html.TextBoxFor(E => E.Ename)

</div>

<div>

@Html.LabelFor(E => E.Dname)<br />

@Html.DropDownListFor(E => E.DesignationId, Model.Designations, "-Select Designation-")

</div>

<div>

@Html.LabelFor(E => E.Email)<br />

@Html.TextBoxFor(E => E.Email)

</div>

<div>

@Html.LabelFor(E => E.Salary)<br />

@Html.TextBoxFor(E => E.Salary)

</div>

<div>

@Html.LabelFor(E => E.Qname)<br />

@Html.DropDownListFor(E => E.QualificationId, Model.Qualifications, "-Select Qualification-")

</div>

<div>

@Html.LabelFor(E => E.Mobile)<br />

@Html.TextBoxFor(E => E.Mobile)

</div>

<div>

@Html.LabelFor(E => E.Manager)<br />

@Html.TextBoxFor(E => E.Manager)

</div>

<div>

<input type="submit" value="Update" name="btnUpdate" />

@Html.ActionLink("Cancel","GetAll")

</div>

}

Using LINQ to Stored Procedure

Now drag and drop the 6 Stored Procedures we have defined above on to the RHS Panel of “OR Designer”, which will generate the required methods in “LINQDataContext”

Add A Controller

using MVCWithLinq2.Models;

using System;

using System.Collections.Generic;

using System.Configuration;

using System.Linq;

using System.Web;

using System.Web.Mvc;

namespace MVCWithLinq2.Controllers

{

public class EmployeeController : Controller

{

LINQDataContext dc = new LINQDataContext(ConfigurationManager.ConnectionStrings["LINQConnectionString"].ConnectionString);

public ViewResult GetAll()

{

List<Employee\_SelectResult> employees = dc.Employee\_Select(null, true).ToList();

return View(employees);

}

public ViewResult GetById(int Eid)

{

Employee\_SelectResult employee = dc.Employee\_Select(Eid, true).ToList()[0];

return View(employee);

}

public ViewResult AddEmployee()

{

Employee\_SelectResult employee=new Employee\_SelectResult();

ViewBag.DesignationId = new SelectList(dc.Get\_Designation(), "DesignationId", "Dname");

ViewBag.QualificationId = new SelectList(dc.Get\_Qualification(), "QualificationId", "Qname");

return View(employee);

}

[HttpPost]

public RedirectToRouteResult AddEmployee(Employee\_SelectResult employee)

{

dc.Employee\_Insert(employee.Eid,employee.Ename,employee.Email,employee.Salary,employee.Mobile,employee.Manager,employee.DesignationId,employee.QualificationId);

return RedirectToAction("GetAll");

}

public ViewResult EditEmployee(int Eid)

{

Employee\_SelectResult emp = dc.Employee\_Select(Eid, true).ToList()[0];

ViewBag.DesignationId = new SelectList(dc.Get\_Designation(), "DesignationId", "Dname");

ViewBag.QualificationId = new SelectList(dc.Get\_Qualification(), "QualificationId", "Qname");

return View(emp);

}

public RedirectToRouteResult UpdateEmployee(Employee\_SelectResult employee)

{

dc.Employee\_Update(employee.Eid, employee.Ename, employee.Email, employee.Salary, employee.Mobile, employee.Manager, employee.DesignationId, employee.QualificationId);

return RedirectToAction("GetAll");

}

public RedirectToRouteResult DeleteEmployee(int Eid)

{

dc.Employee\_Delete(Eid);

return RedirectToAction("GetAll");

}

}

}

@model IEnumerable<MVCWithLinq2.Models.Employee\_SelectResult>

@{

ViewBag.Title = "GetAll";

}

<table border="1" align="center">

<tr>

<th>

@Html.DisplayNameFor(E => E.Eid)

</th>

<th>

@Html.DisplayNameFor(E => E.Ename)

</th>

<th>

@Html.DisplayNameFor(E => E.DesignationId)

</th>

<th>

@Html.DisplayNameFor(E => E.Email)

</th>

<th>

@Html.DisplayNameFor(E => E.Salary)

</th>

<th>

@Html.DisplayNameFor(E => E.QualificationId)

</th>

<th>

@Html.DisplayNameFor(E => E.Manager)

</th>

<th>

@Html.DisplayNameFor(E => E.Mobile)

</th>

<th>Actions</th>

</tr>

@foreach (var employee in Model)

{

<tr>

<td>

@Html.DisplayFor(E => employee.Eid)

</td>

<td>

@Html.DisplayFor(E => employee.Ename)

</td>

<td>

@Html.DisplayFor(E => employee.DesignationId)

</td>

<td>

@Html.DisplayFor(E => employee.Email)

</td>

<td>

@Html.DisplayFor(E => employee.Salary)

</td>

<td>

@Html.DisplayFor(E => employee.QualificationId)

</td>

<td>

@Html.DisplayFor(E => employee.Manager)

</td>

<td>

@Html.DisplayFor(E => employee.Mobile)

</td>

<td>

@Html.ActionLink("View", "GetById", new { Eid = employee.Eid })

@Html.ActionLink("Edit", "EditEmployee", new { Eid = employee.Eid })

@Html.ActionLink("Delete", "DeleteEmployee", new { Eid = employee.Eid },

new { onclick = "return confirm('Are you sure of deleting recird?')" })

</td>

</tr>

}

<tr>

<td colspan="9" align="center">@Html.ActionLink("Add new emp", "AddEmployee")</td>

</tr>

</table>

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.Linq;

using System.Web;

namespace WebApplication9.Models

{

public class EmployeeModel

{

public int Eid { get; set; }

[Required(ErrorMessage ="Can't leave this field empty")]

public string Ename { get; set; }

public Nullable<int> Did { get; set; }

[Required(ErrorMessage = "Can't leave this field empty")]

public string Salary { get; set; }

[Required(ErrorMessage = "Can't leave this field empty")]

public string Email { get; set; }

public string Mobile { get; set; }

public Nullable<int> Qid { get; set; }

public string Manager { get; set; }

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using WebApplication9.Models;

namespace WebApplication9.Controllers

{

public class EmployeeController : Controller

{

NOVAEntities dc=new NOVAEntities();

public ActionResult DisplayEmployees()

{

var Emps = dc.Employees.Where(E => E.Status == true);

return View(Emps);

}

public ActionResult DisplayEmployee(int Eid)

{

var emp = dc.Employees.Find(Eid);

return View(emp);

}

public ActionResult AddEmployee()

{

ViewBag.Did = new SelectList(dc.Designations, "Did", "Dname");

ViewBag.Qid = new SelectList(dc.Qualifications, "Qid", "Qname");

return View();

}

[HttpPost]

public ActionResult AddEmployee(EmployeeModel emp)

{

if (!ModelState.IsValid)

{

return View(emp);

}

else

{

Employee emps = new Employee

{

Ename = emp.Ename,

Did=emp.Did,

Salary=emp.Salary,

Email=emp.Email,

Mobile=emp.Mobile,

Qid=emp.Qid,

Manager=emp.Manager,

Status=true

};

dc.Employees.Add(emps);

dc.SaveChanges();

return RedirectToAction("DisplayEmployees");

}

}

public ActionResult EditEmployee(int Eid)

{

Employee emp = dc.Employees.Find(Eid);

ViewBag.Did = new SelectList(dc.Designations, "Did", "Dname", emp.Did);

ViewBag.Qid = new SelectList(dc.Qualifications, "Qid", "Qname", emp.Qid);

return View(emp);

}

[HttpPost]

public ActionResult UpdateEmployee(Employee emp)

{

emp.Status = true;

dc.Entry(emp).State = System.Data.Entity.EntityState.Modified;

dc.SaveChanges();

return RedirectToAction("DisplayEmployees");

}

public ViewResult DeleteEmployee(int Eid)

{

Employee emp = dc.Employees.Find(Eid);

return View(emp);

}

[HttpPost]

public RedirectToRouteResult DeleteEmployee(Employee emp)

{

dc.Entry(emp).State = System.Data.Entity.EntityState.Modified;

dc.SaveChanges();

return RedirectToAction("DisplayEmployees");

}

}

}

@model IEnumerable<WebApplication9.Models.Employee>

@{

ViewBag.Title = "DisplayEmployees";

}

<table border="1" align="center">

<tr>

<th>@Html.DisplayNameFor(E => E.Eid)</th>

<th>@Html.DisplayNameFor(E => E.Ename)</th>

<th>@Html.DisplayNameFor(E => E.Did)</th>

<th>@Html.DisplayNameFor(E => E.Designation.Dname)</th>

<th>@Html.DisplayNameFor(E => E.Salary)</th>

<th>@Html.DisplayNameFor(E => E.Email)</th>

<th>@Html.DisplayNameFor(E => E.Mobile)</th>

<th>@Html.DisplayNameFor(E => E.Qid)</th>

<th>@Html.DisplayNameFor(E => E.Qualification.Qname)</th>

<th>@Html.DisplayNameFor(E => E.Manager)</th>

<th>Actions</th>

</tr>

@foreach(var Employee in Model)

{

<tr>

<td>

@Html.DisplayFor(E => Employee.Eid)

</td>

<td>

@Html.DisplayFor(E=>Employee.Ename)

</td>

<td>

@Html.DisplayFor(E => Employee.Designation.Did)

</td>

<td>

@Html.DisplayFor(E => Employee.Designation.Dname)

</td>

<td>

@Html.DisplayFor(E => Employee.Salary)

</td>

<td>

@Html.DisplayFor(E => Employee.Email)

</td>

<td>

@Html.DisplayFor(E => Employee.Mobile)

</td>

<td>

@Html.DisplayFor(E => Employee.Qualification.Qid)

</td>

<td>

@Html.DisplayFor(E => Employee.Qualification.Qname)

</td>

<td>

@Html.DisplayFor(E => Employee.Manager)

</td>

<td>

@Html.ActionLink("View", "DisplayEmployee", new { Eid = Employee.Eid })

@Html.ActionLink("Edit", "EditEmployee", new { Eid = Employee.Eid })

@Html.ActionLink("Delete", "DeleteEmployee", new { Eid = Employee.Eid })

</td>

</tr>

}

<tr>

<td colspan="11" align="center">@Html.ActionLink("Add New Employee","AddEmployee")</td>

</tr>

</table>

@model WebApplication9.Models.EmployeeModel

@{

ViewBag.Title = "AddEmployee";

}

@using (Html.BeginForm("AddEmployee", "Employee"))

{

<div>

@Html.LabelFor(E => E.Ename)<br />

@Html.TextBoxFor(E => E.Ename)

@Html.ValidationMessageFor(E => E.Ename, "", new { @class = "text-danger" })

</div>

<div>

<label>Designation</label><br />

@Html.DropDownList("Did", "-Select Designation-")

</div>

<div>

@Html.LabelFor(E => E.Salary)<br />

@Html.TextBoxFor(E => E.Salary)

@Html.ValidationMessageFor(E => E.Salary, "", new { @class = "text-danger" })

</div>

<div>

@Html.LabelFor(E => E.Email)<br />

@Html.TextBoxFor(E => E.Email)

@Html.ValidationMessageFor(E => E.Email, "", new { @class = "text-danger" })

</div>

<div>

@Html.LabelFor(E => E.Mobile)<br />

@Html.TextBoxFor(E => E.Mobile)

</div>

<div>

<label>Qualificaton</label><br />

@Html.DropDownList("Qid", "-Select Qualification-")

</div>

<div>

@Html.LabelFor(E => E.Manager)<br />

@Html.TextBoxFor(E => E.Manager)

</div>

<div>

<input type="submit" value="Save" name="btnSave" />

<input type="reset" value="Reset" name="btnReset" />

</div>

}

@Html.ActionLink("Back to Employee Details", "DisplayEmployees")

<script src="~/Scripts/jquery-3.4.1.min.js"></script>

<script src="~/Scripts/jquery.validate.min.js"></script>

<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>

<link href="~/Content/bootstrap.min.css" rel="stylesheet" />

@model WebApplication9.Models.Employee

@{

ViewBag.Title = "DeleteEmployee";

}

@using (Html.BeginForm("DeleteEmployee", "Employee"))

{

<table border="1">

<tr>

<td>Eid:</td>

<td>@Html.TextBoxFor(E=>E.Eid,new { @readonly="true"})</td>

</tr>

<tr>

<td>Ename:</td>

<td>@Html.TextBoxFor(E => E.Ename, new {@readonly="true"})</td>

</tr>

<tr>

<td>Designation:</td>

<td>@Html.TextBoxFor(E => E.Did, new {@readonly="true"})</td>

</tr>

<tr>

<td>Salary:</td>

<td>@Html.TextBoxFor(E => E.Salary, new {@readonly="true"})</td>

</tr>

</table>

<font color="red">Are you sure of deleting the current record?</font>

<input type="submit" value="Yes"/><br/>

@Html.ActionLink("Cancel","DisplayEmployees")

}

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.Linq;

using System.Web;

namespace WebApplication10.Models

{

public class Employee

{

public int? Eid { get; set; }

[Required(ErrorMessage ="Can't leave this Field Empty")]

public string Ename { get; set; }

public int? Did { get; set; }

public decimal? Salary { get; set; }

[Required(ErrorMessage = "Can't leave this Field Empty")]

public string Email { get; set; }

[Required(ErrorMessage = "Can't leave this Field Empty")]

public string Mobile { get; set; }

public int? Qid { get; set; }

public string Manager { get; set; }

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace WebApplication10.Models

{

public class Designation

{

public int Did { get; set; }

public string Dname { get; set; }

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace WebApplication10.Models

{

public class Qualification

{

public int Qid { get; set; }

public string Qname { get; set; }

}

}

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Web;

using System.Data;

using System.Configuration;

namespace WebApplication10.Models

{

public class EmployeeDAL

{

SqlCommand cmd;

SqlConnection con;

public EmployeeDAL()

{

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

con = new SqlConnection(ConStr);

cmd = new SqlCommand();

cmd.Connection = con;

cmd.CommandType = CommandType.StoredProcedure;

}

public List<Employee> GetEmployees(bool? Status)

{

List<Employee> employees = new List<Employee>();

try

{

cmd.CommandText = "Employee\_Select";

cmd.Parameters.Clear();

if (Status != null)

{

cmd.Parameters.AddWithValue("@Status", Status);

con.Open();

SqlDataReader dr = cmd.ExecuteReader();

while (dr.Read())

{

Employee employee = new Employee

{

Eid = Convert.ToInt32(dr["Eid"]),

Ename = dr["Ename"].ToString(),

Did = Convert.ToInt32(dr["Did"]),

Salary = Convert.ToDecimal(dr["Salary"]),

Email = dr["Email"].ToString(),

Mobile = dr["Mobile"].ToString(),

Qid = Convert.ToInt32(dr["Qid"]),

Manager = dr["Manager"].ToString()

};

employees.Add(employee);

}

}

}

catch(Exception ex)

{

throw ex;

}

finally

{

con.Close();

}

return employees;

}

public List<Designation> GetDesignations()

{

List<Designation> designations=new List<Designation>();

try

{

cmd.CommandText = "Get\_Designation";

cmd.Parameters.Clear();

con.Open();

SqlDataReader dr = cmd.ExecuteReader();

while (dr.Read())

{

Designation des = new Designation

{

Did = Convert.ToInt32(dr["Did"]),

Dname = dr["Dname"].ToString()

};

designations.Add(des);

}

}

catch(Exception ex)

{

throw ex;

}

finally

{

con.Close();

}

return designations;

}

public List<Qualification> GetQualifications()

{

List<Qualification> qualifications = new List<Qualification>();

try

{

cmd.CommandText = "Get\_Qualification";

cmd.Parameters.Clear();

con.Open();

SqlDataReader dr = cmd.ExecuteReader();

while (dr.Read())

{

Qualification ques = new Qualification

{

Qid = Convert.ToInt32(dr["Qid"]),

Qname = dr["Qname"].ToString()

};

qualifications.Add(ques);

}

}

catch(Exception ex)

{

throw ex;

}

finally

{

con.Close();

}

return qualifications;

}

public Employee GetEmployee(int Eid ,bool? Status)

{

Employee employee = null;

try

{

cmd.CommandText = "Employee\_Select";

cmd.Parameters.Clear();

cmd.Parameters.AddWithValue("@Eid", Eid);

if (Status != null)

{

cmd.Parameters.AddWithValue("@Status", Status);

}

con.Open();

SqlDataReader dr = cmd.ExecuteReader();

if (dr.Read())

{

employee = new Employee();

employee.Eid = Convert.ToInt32(dr["Eid"]);

employee.Ename = dr["Ename"].ToString();

employee.Did = Convert.ToInt32(dr["Did"]);

employee.Salary = Convert.ToDecimal(dr["Salary"]);

employee.Email = dr["Email"].ToString();

employee.Mobile = dr["Mobile"].ToString();

employee.Qid = Convert.ToInt32(dr["Qid"]);

employee.Manager = dr["Manager"].ToString();

}

}

catch(Exception ex)

{

}

finally

{

con.Close();

}

return employee;

}

public int InsertEmployee(Employee emp)

{

int Count = 0;

try

{

cmd.CommandText = "Insert\_Employee";

cmd.Parameters.Clear();

cmd.Parameters.AddWithValue("@Eid", emp.Eid);

cmd.Parameters.AddWithValue("@Ename", emp.Ename);

cmd.Parameters.AddWithValue("@Did", emp.Did);

cmd.Parameters.AddWithValue("@Salary", emp.Salary);

cmd.Parameters.AddWithValue("@Email", emp.Email);

cmd.Parameters.AddWithValue("@Mobile", emp.Mobile);

cmd.Parameters.AddWithValue("@Qid", emp.Qid);

cmd.Parameters.AddWithValue("@Manager", emp.Manager);

con.Open();

Count = cmd.ExecuteNonQuery();

}

catch(Exception ex)

{

throw ex;

}

finally

{

con.Close();

}

return Count;

}

public int UpdateEmployee(Employee emp)

{

int Count = 0;

try

{

cmd.CommandText = "Update\_Employee";

cmd.Parameters.Clear();

cmd.Parameters.AddWithValue("@Eid", emp.Eid);

cmd.Parameters.AddWithValue("@Ename", emp.Ename);

cmd.Parameters.AddWithValue("@Did", emp.Did);

cmd.Parameters.AddWithValue("@Salary", emp.Salary);

cmd.Parameters.AddWithValue("@Email", emp.Email);

cmd.Parameters.AddWithValue("@Mobile", emp.Mobile);

cmd.Parameters.AddWithValue("@Qid", emp.Qid);

cmd.Parameters.AddWithValue("@Manager", emp.Manager);

con.Open();

Count = cmd.ExecuteNonQuery();

}

catch (Exception ex)

{

throw ex;

}

finally

{

con.Close();

}

return Count;

}

public int DeleteEmployee(int Eid)

{

int Count = 0;

try

{

cmd.CommandText = "Employee\_Delete";

cmd.Parameters.Clear();

cmd.Parameters.AddWithValue("@Eid", Eid);

con.Open();

Count = cmd.ExecuteNonQuery();

}

catch (Exception ex)

{

throw ex;

}

finally

{

con.Close();

}

return Count;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using WebApplication10.Models;

namespace WebApplication10.Controllers

{

public class EmployeeController : Controller

{

EmployeeDAL dc = new EmployeeDAL();

public ActionResult DisplayEmployees()

{

var emps = dc.GetEmployees(true);

return View(emps);

}

public ActionResult DisplayEmployee(int Eid)

{

var emp = dc.GetEmployee(Eid, true);

return View(emp);

}

public ActionResult AddEmployee()

{

ViewBag.Did = new SelectList(dc.GetDesignations(), "Did", "Dname");

ViewBag.Qid = new SelectList(dc.GetQualifications(), "Qid", "Qname");

return View();

}

[HttpPost]

public ActionResult AddEmployee(Employee emp)

{

dc.InsertEmployee(emp);

return RedirectToAction("DisplayEmployees");

}

public ActionResult EditEmployee(int Eid)

{

Employee emp = dc.GetEmployee(Eid, true);

ViewBag.Did = new SelectList(dc.GetDesignations(), "Did", "Dname");

ViewBag.Qid = new SelectList(dc.GetQualifications(), "Qid", "Qname");

return View(emp);

}

[HttpPost]

public ActionResult UpdateEmployee(Employee Emp)

{

dc.UpdateEmployee(Emp);

return RedirectToAction("DisplayEmployees");

}

public ActionResult DeleteEmployee(int Eid)

{

dc.DeleteEmployee(Eid);

return RedirectToAction("DisplayEmployees");

}

}

}

@model WebApplication10.Models.Employee

@{

ViewBag.Title = "AddEmployee";

}

@using (Html.BeginForm("AddEmployee", "Employee"))

{

<div>

@Html.LabelFor(E => E.Eid)<br />

@Html.TextBoxFor(E => E.Eid)

</div>

<div>

@Html.LabelFor(E => E.Ename)<br />

@Html.TextBoxFor(E => E.Ename)

@Html.ValidationMessageFor(E => E.Ename, "", new { @class = "text-danger" })

</div>

<div>

<label>Designation</label><br />

@Html.DropDownList("Did", "-Select Designation-")

</div>

<div>

@Html.LabelFor(E => E.Salary)<br />

@Html.TextBoxFor(E => E.Salary)

</div>

<div>

@Html.LabelFor(E => E.Email)<br />

@Html.TextBoxFor(E => E.Email)

@Html.ValidationMessageFor(E => E.Email, "", new { @class = "text-danger" })

</div>

<div>

@Html.LabelFor(E => E.Mobile)<br />

@Html.TextBoxFor(E => E.Mobile)

@Html.ValidationMessageFor(E => E.Email, "", new { @class = "text-danger" })

</div>

<div>

<label>Qualificaton</label><br />

@Html.DropDownList("Qid", "-Select Qualification-")

</div>

<div>

@Html.LabelFor(E => E.Manager)<br />

@Html.TextBoxFor(E => E.Manager)

</div>

<div>

<input type="submit" value="Save" name="btnSave" />

<input type="reset" value="Reset" name="btnReset" />

</div>

}

@Html.ActionLink("Back to Employee Details", "DisplayEmployees")

<script src="~/Scripts/jquery-3.4.1.min.js"></script>

<script src="~/Scripts/jquery.validate.min.js"></script>

<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>

<link href="~/Content/bootstrap.min.css" rel="stylesheet" />