**Model**

Registration.cs FILE

public class Registration {

public int Id { get; set; }

[Required(ErrorMessage = "Name is required")]

[StringLength(100, ErrorMessage = "Name cannot be longer than 100 characters")]

[RegularExpression(@"^[a-zA-Z\s]\*$", ErrorMessage = "Name can only contain alphabets")]

public string Name { get; set; }

[Required(ErrorMessage = "Date of Birth is required")]

[DataType(DataType.Date)]

[DisplayFormat(DataFormatString = "{0:yyyy-MM-dd}", ApplyFormatInEditMode = true)]

public DateTime Dob { get; set; }

[Required(ErrorMessage = "Age is required")]

[Range(1, 150, ErrorMessage = "Age must be between 1 and 150")]

public int Age { get; set; }

[Required(ErrorMessage = "Gender is required")]

public string Gender { get; set; }

[Required(ErrorMessage = "Phone number is required")]

[RegularExpression(@"^\d{10}$", ErrorMessage = "Invalid phone number format. Please enter a 10-digit phone number.")]

public string Phone { get; set; }

[Required(ErrorMessage = "Email is required")]

[EmailAddress(ErrorMessage = "Invalid email address")]

public string Email { get; set; }

[Required(ErrorMessage = "State is required")]

public string State { get; set; }

[Required(ErrorMessage = "City is required")]

public string City { get; set; }

[Required(ErrorMessage = "Username is required")]

[StringLength(50, ErrorMessage = "Username cannot be longer than 50 characters")]

public string Username { get; set; }

[Required(ErrorMessage = "Password is required")]

[StringLength(100, MinimumLength = 8, ErrorMessage = "Password must be at least 8 characters long")]

public string Password { get; set; }

}

**Database/Stored Procedure**

Microsoft Sql Server

Creating table

CREATE TABLE Registration (

id INT PRIMARY KEY IDENTITY(1,1),

Name VARCHAR(255) NOT NULL,

Dob DATE NOT NULL,

Age INT NOT NULL,

Gender VARCHAR(10) NOT NULL,

Phone VARCHAR(15) NOT NULL,

Email VARCHAR(255) NOT NULL,

State VARCHAR(50) NOT NULL,

City VARCHAR(50) NOT NULL,

Username VARCHAR(50) NOT NULL,

Password VARCHAR(255) NOT NULL

);

Inserting in table

CREATE PROCEDURE SPI\_Registration

@name VARCHAR(255),

@dob DATE,

@age INT,

@gender CHAR(1),

@phone VARCHAR(15),

@email VARCHAR(255),

@state VARCHAR(50),

@city VARCHAR(50),

@username VARCHAR(50),

@password VARCHAR(255)

AS

BEGIN

INSERT INTO Registration (Name, Dob, Age, Gender, Phone, Email, State, City, Username, [password])

VALUES (@name, @dob, @age, @gender, @phone, @email, @state, @city, @username, @password);

END;

EXEC InsertRegistration

@name = 'John Doe',

@dob = '1990-05-15',

@age = 32,

@gender = 'M',

@phone = '123-456-7890',

@email = 'johndoe@example.com',

@state = 'California',

@city = 'Los Angeles',

@username = 'johndoe123',

@password = 'securepassword';

Updating data

CREATE PROCEDURE SPU\_Registration

@id INT,

@name VARCHAR(255),

@dob DATE,

@age INT,

@gender CHAR(1),

@phone VARCHAR(15),

@email VARCHAR(255),

@state VARCHAR(50),

@city VARCHAR(50),

@username VARCHAR(50),

@password VARCHAR(255)

AS

BEGIN

UPDATE Registration

SET

Name = @name,

Dob = @dob,

Age = @age,

Gender = @gender,

Phone = @phone,

Email = @email,

State = @state,

City = @city,

Username = @username,

[Password] = @password

WHERE

Id = @id;

END;

EXEC UpdateRegistration

@id = 1,

@name = 'Updated Name',

@dob = '1995-08-20',

@age = 27,

@gender = 'F',

@phone = '555-555-5555',

@email = 'updated@example.com',

@state = 'New State',

@city = 'New City',

@username = 'updateduser',

@password = 'updatedpassword';

Deleting data

CREATE PROCEDURE SPD\_Registration

@id INT

AS

BEGIN

DELETE FROM Registration

WHERE Id = @id;

END;

Selecting data

CREATE PROCEDURE SPS\_Registration

(

@UserId INT

)

AS

BEGIN

SELECT \*

FROM Registration

WHERE id = @UserId;

END;

**Repository**

RegistrationRepository.cs

public class RegistrationRepository

{

SqlConnection connectionLink = new SqlConnection(ConfigurationManager.ConnectionStrings["cs"].ConnectionString);

SqlCommand command;

SqlDataAdapter adapter;

DataTable table;

public List<Registration> GetUsers()

{

command = new SqlCommand("sp\_SelectAllRegistration", connectionLink);

command.CommandType = CommandType.StoredProcedure;

adapter = new SqlDataAdapter(command);

table = new DataTable();

adapter.Fill(table);

List<Registration> list = new List<Registration>();

foreach (DataRow table in table.Rows)

{

list.Add(new Registration

{

Id = Convert.ToInt32(table["Id"]),

Name = table["Name"].ToString(),

Dob = DateTime.Parse(table["Dob"].ToString()),

Age = Convert.ToInt32(table["Age"]),

Gender = table["Gender"].ToString(),

Phone = table["Phone"].ToString(),

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

State = table["State"].ToString(),

City = table["City"].ToString(),

Username = table["Username"].ToString(),

Password = table["Password"].ToString(),

});

}

return list;

}

public bool InsertUser(Registration registration)

{

try

{

command = new SqlCommand("sp\_InsertRegistration", connectionLink);

command.CommandType = CommandType.StoredProcedure;

command.Parameters.AddWithValue("@name", registration.Name);

command.Parameters.AddWithValue("@dob", registration.Dob);

command.Parameters.AddWithValue("@age", registration.Age);

command.Parameters.AddWithValue("@gender", registration.Gender);

command.Parameters.AddWithValue("@phone", registration.Phone);

command.Parameters.AddWithValue("@email", registration.Email);

command.Parameters.AddWithValue("@state", registration.State);

command.Parameters.AddWithValue("@city", registration.City);

command.Parameters.AddWithValue("@username", registration.Username);

command.Parameters.AddWithValue("@password", registration.Password);

connectionLink.Open();

int r = command.ExecuteNonQuery();

if (r > 0)

{

return true;

}

else

return false;

}

catch (Exception)

{

throw;

}

}

public bool UpdateUser(Registration registration)

{

try

{

command = new SqlCommand("sp\_UpdateRegistration", connectionLink);

command.CommandType = CommandType.StoredProcedure;

command.Parameters.AddWithValue("@name", registration.Name);

command.Parameters.AddWithValue("@dob", registration.Dob);

command.Parameters.AddWithValue("@age", registration.Age);

command.Parameters.AddWithValue("@gender", registration.Gender);

command.Parameters.AddWithValue("@phone", registration.Phone);

command.Parameters.AddWithValue("@email", registration.Email);

command.Parameters.AddWithValue("@state", registration.State);

command.Parameters.AddWithValue("@city", registration.City);

command.Parameters.AddWithValue("@username", registration.Username);

command.Parameters.AddWithValue("@password", registration.Password);

command.Parameters.AddWithValue("@id", registration.Id);

connectionLink.Open();

int r = command.ExecuteNonQuery();

if (r > 0)

{

return true;

}

else

return false;

}

catch (Exception)

{

throw;

}

}

public int DeleteUser(int id)

{

try

{

command = new SqlCommand("sp\_DeleteRegistration", connectionLink);

command.CommandType = CommandType.StoredProcedure;

command.Parameters.AddWithValue("@id", id);

connectionLink.Open();

return command.ExecuteNonQuery();

}

catch (Exception)

{

throw;

}

}

}

**Controller**

RegistrationController.cs

public class RegistrationController : Controller

{

RegistrationRepository registrationRepository = new RegistrationRepository();

public ActionResult List()

{

var data = registrationRepository.GetUsers();

return View(data);

}

public ActionResult Create()

{

return View();

}

[HttpPost]

public ActionResult Create(Registration registration)

{

if (ModelState.IsValid)

{

if (registrationRepository.InsertUser(registration))

{

TempData["InsertMsg"] = "<script>alert('User saved successfully.')</script>";

return RedirectToAction("List");

}

else

{

TempData["InsertErrorMsg"] = "<script>alert('Error')</script>";

}

}

return View(registration);

}

public ActionResult Edit(int id)

{

var data = registrationRepository.GetUsers().Find(a => a.Id == id);

if (data != null)

{

data.Dob = data.Dob.Date;

return View(data);

}

else

{

return HttpNotFound();

}

}

[HttpPost]

public ActionResult Edit(Registration registration)

{

if (ModelState.IsValid)

{

if (registrationRepository.UpdateUser(registration))

{

TempData["UpdateMsg"] = "<script>alert('User updated successfully.')</script>";

return RedirectToAction("List");

}

else

{

TempData["UpdateErrorMsg"] = "<script>alert('Error')</script>";

}

}

return View(registration);

}

public ActionResult Delete(int id)

{

int result = registrationRepository.DeleteUser(id);

if (result > 0)

{

TempData["DeleteMsg"] = "<script>alert('User Deleted sucessfully.....')</script>";

return RedirectToAction("List");

}

else

{

TempData["DeleteErrorMsg"] = "<script>alert('Error Deleting')</script>";

}

return View();

}

}

**Views**

List.cshtml

@model IEnumerable<WatchWorld.Models.Registration>

@{

ViewBag.Title = "List";

Layout = "~/Views/HomeShared/HomeLayout.cshtml";

}

<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width" />

<title>Index</title>

<link href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css" rel="stylesheet">

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

</head>

<body>

<div class="container">

<h2>List</h2>

@Html.Raw(TempData["InsertMsg"])

@Html.Raw(TempData["UpdateMsg"])

@Html.Raw(TempData["DeleteMsg"])

@Html.Raw(TempData["DeleteErrorMsg"])

<p>

@Html.ActionLink("Create New", "Create", null, new { @class = "btn btn-primary" })

</p>

<table class="table table-striped">

<thead>

<tr>

<th>@Html.DisplayNameFor(model => model.Name)</th>

<th>@Html.DisplayNameFor(model => model.Dob)</th>

<th>@Html.DisplayNameFor(model => model.Age)</th>

<th>@Html.DisplayNameFor(model => model.Gender)</th>

<th>@Html.DisplayNameFor(model => model.Phone)</th>

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

<th>@Html.DisplayNameFor(model => model.State)</th>

<th>@Html.DisplayNameFor(model => model.City)</th>

<th>@Html.DisplayNameFor(model => model.Username)</th>

<th>Password</th>

<th>Actions</th>

</tr>

</thead>

<tbody>

@foreach (var item in Model)

{

<tr>

<td>@Html.DisplayFor(modelItem => item.Name)</td>

<td>@Html.DisplayFor(modelItem => item.Dob)</td>

<td>@Html.DisplayFor(modelItem => item.Age)</td>

<td>@Html.DisplayFor(modelItem => item.Gender)</td>

<td>@Html.DisplayFor(modelItem => item.Phone)</td>

<td>@Html.DisplayFor(modelItem => item.Email)</td>

<td>@Html.DisplayFor(modelItem => item.State)</td>

<td>@Html.DisplayFor(modelItem => item.City)</td>

<td>@Html.DisplayFor(modelItem => item.Username)</td>

<td>

@Html.Password("Password", new string('•', item.Password.Length), new { @readonly = "readonly", @class = "form-control" })

</td>

<td class="buttons">

@Html.ActionLink("Edit", "Edit", new { id = item.Id }, new { @class = "button btn btn-warning btn-sm" })

@Html.ActionLink("Delete", "Delete", new { id = item.Id }, new { @class = "button btn btn-danger btn-sm" })

</td>

</tr>

}

</tbody>

</table>

</div>

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/2.5.3/umd/popper.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

</body>

</html>

Create.cshtml

@model WatchWorld.Models.Registration

@{

ViewBag.Title = "Create";

Layout = "~/Views/HomeShared/HomeLayout.cshtml";

}

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Edit</title>

<link href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css" rel="stylesheet">

<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/2.5.3/umd/popper.min.js"></script>

</head>

<body>

<h2>Create</h2>

@Html.Raw(TempData["InsertErrorMsg"])

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="container">

<h4>Registration</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group row">

@Html.LabelFor(model => model.Name, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

@Html.EditorFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })

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

</div>

</div>

<div class="form-group row">

@Html.LabelFor(model => model.Dob, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

@{

var maxDate = DateTime.Today.ToString("yyyy-MM-dd");

}

@Html.TextBoxFor(model => model.Dob, new { @type = "date", @id = "datepicker", @class = "form-control", @max = maxDate })

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

</div>

</div>

<div class="form-group row">

@Html.LabelFor(model => model.Age, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

@Html.EditorFor(model => model.Age, new { htmlAttributes = new { @class = "form-control" } })

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

</div>

</div>

<div class="form-group row">

@Html.LabelFor(model => model.Gender, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

<label class="form-check-inline">

@Html.RadioButtonFor(model => model.Gender, "M", new { @class = "form-check-input" }) Male

</label>

<label class="form-check-inline">

@Html.RadioButtonFor(model => model.Gender, "F", new { @class = "form-check-input" }) Female

</label>

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

</div>

</div>

<div class="form-group row">

@Html.LabelFor(model => model.Phone, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

@Html.EditorFor(model => model.Phone, new { htmlAttributes = new { @class = "form-control" } })

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

</div>

</div>

<div class="form-group row">

@Html.LabelFor(model => model.Email, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

@Html.EditorFor(model => model.Email, new { htmlAttributes = new { @class = "form-control" } })

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

</div>

</div>

<div class="form-group row">

@Html.LabelFor(model => model.State, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

@Html.DropDownListFor(model => model.State, new SelectList(

new List<SelectListItem>

{

new SelectListItem { Text = "Kerala", Value = "Kerala" },

new SelectListItem { Text = "Karnataka", Value = "Karnataka" },

new SelectListItem { Text = "Tamil Nadu", Value = "Tamil Nadu" },

new SelectListItem { Text = "Andhra", Value = "Andhra" }

}, "Value", "Text"),

new { @class = "form-control" })

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

</div>

</div>

<div class="form-group row">

@Html.LabelFor(model => model.City, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

@Html.DropDownListFor(model => model.City, new SelectList(

new List<SelectListItem>

{

new SelectListItem { Text = "Kochi", Value = "Kochi" },

new SelectListItem { Text = "Bangalore", Value = "Bangalore" },

new SelectListItem { Text = "Chennai", Value = "Chennai" },

new SelectListItem { Text = "Vizag", Value = "Vizag" }

}, "Value", "Text"),

new { @class = "form-control" })

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

</div>

</div>

<div class="form-group row">

@Html.LabelFor(model => model.Username, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

@Html.EditorFor(model => model.Username, new { htmlAttributes = new { @class = "form-control" } })

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

</div>

</div>

<div class="form-group row">

@Html.LabelFor(model => model.Password, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

@Html.PasswordFor(model => model.Password, new { htmlAttributes = new { @class = "form-control" } })

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

</div>

</div>

<div class="form-group row">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Create" class="btn btn-primary" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "List", "Registration")

</div>

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/2.5.3/umd/popper.min.js"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

</body>

</html>

Edit.cshtml

@model WatchWorld.Models.Registration

@{

ViewBag.Title = "Edit";

Layout = "~/Views/HomeShared/HomeLayout.cshtml";

}

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Edit</title>

<link href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css" rel="stylesheet">

<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/2.5.3/umd/popper.min.js"></script>

</head>

<body>

<h2>Edit</h2>

@Html.Raw(TempData["UpdateErrorMsg"])

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="container">

<h4>Registration</h4>

<hr />

@Html.HiddenFor(model => model.Id)

<div class="form-group row">

@Html.LabelFor(model => model.Name, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

@Html.EditorFor(model => model.Name, new { htmlAttributes = new { @class = "form-control" } })

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

</div>

</div>

<div class="form-group row">

@Html.LabelFor(model => model.Dob, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

<input type="date" id="datepicker" name="Dob" value="@Model.Dob.ToString("yyyy-MM-dd")" class="form-control" />

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

</div>

</div>

<div class="form-group row">

@Html.LabelFor(model => model.Age, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

@Html.EditorFor(model => model.Age, new { htmlAttributes = new { @class = "form-control" } })

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

</div>

</div>

<div class="form-group row">

@Html.LabelFor(model => model.Gender, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

<label class="form-check-inline">

@Html.RadioButtonFor(model => model.Gender, "M", new { @class = "form-check-input" }) Male

</label>

<label class="form-check-inline">

@Html.RadioButtonFor(model => model.Gender, "F", new { @class = "form-check-input" }) Female

</label>

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

</div>

</div>

<div class="form-group row">

@Html.LabelFor(model => model.Phone, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

@Html.EditorFor(model => model.Phone, new { htmlAttributes = new { @class = "form-control" } })

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

</div>

</div>

<div class="form-group row">

@Html.LabelFor(model => model.Email, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

@Html.EditorFor(model => model.Email, new { htmlAttributes = new { @class = "form-control" } })

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

</div>

</div>

<div class="form-group row">

@Html.LabelFor(model => model.State, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

@Html.DropDownListFor(model => model.State, new SelectList(

new List<SelectListItem>

{

new SelectListItem { Text = "Kerala", Value = "Kerala" },

new SelectListItem { Text = "Karnataka", Value = "Karnataka" },

new SelectListItem { Text = "Tamil Nadu", Value = "Tamil Nadu" },

new SelectListItem { Text = "Andhra", Value = "Andhra" }

}, "Value", "Text"),

new { @class = "form-control" })

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

</div>

</div>

<div class="form-group row">

@Html.LabelFor(model => model.City, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

@Html.DropDownListFor(model => model.City, new SelectList(

new List<SelectListItem>

{

new SelectListItem { Text = "Kochi", Value = "Kochi" },

new SelectListItem { Text = "Bangalore", Value = "Bangalore" },

new SelectListItem { Text = "Chennai", Value = "Chennai" },

new SelectListItem { Text = "Vizag", Value = "Vizag" }

}, "Value", "Text"),

new { @class = "form-control" })

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

</div>

</div>

<div class="form-group row">

@Html.LabelFor(model => model.Username, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

@Html.EditorFor(model => model.Username, new { htmlAttributes = new { @class = "form-control" } })

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

</div>

</div>

<div class="form-group row">

@Html.LabelFor(model => model.Password, htmlAttributes: new { @class = "col-md-2 col-form-label" })

<div class="col-md-10">

@Html.PasswordFor(model => model.Password, new { htmlAttributes = new { @class = "form-control" } })

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

</div>

</div>

<div class="form-group row">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Save" class="btn btn-primary" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "List")

</div>

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/2.5.3/umd/popper.min.js"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

</body>

</html>

Web.config

<connectionStrings>

<add name="cs" connectionString="Data Source=DESKTOP-10MH51F\SQLEXPRESS;Initial Catalog=MvcSp;Integrated Security=True" providerName="System.Data.SqlClient" />

</connectionStrings>