For this single page repository necessary code and tools

VS Studio 2022

Sql server

Nuget required

Microsoft.EntityFrameworkCore

Microsoft.EntityFrameworkCore.Design

Microsoft.EntityFrameworkCore.SqlServer

Microsoft.EntityFrameworkCore.Tools

Create class Students

public class Students

{

[Key]

public int Id { get; set; }

public string studentCode { get; set; }

public string studentName { get; set; }

public string studentDepartment { get; set; }

public string studentFee { get; set; }

}

Create Dbcontext

public class AppDbContext:DbContext

{

public AppDbContext(DbContextOptions<AppDbContext> options) : base(options)

{

}

public DbSet<Students> Students { get; set; }

}

Programm class code

// Add services to the container.

builder.Services.AddControllersWithViews();

builder.Services.AddRazorPages();

builder.Services.AddDbContext<AppDbContext>(options => options.UseSqlServer(builder.Configuration.GetConnectionString("DbConnection")));

builder.Services.AddTransient<IStudent, StudentService>();

var app = builder.Build();

// Configure the HTTP request pipeline.

if (!app.Environment.IsDevelopment())

{

app.UseExceptionHandler("/Home/Error");

// The default HSTS value is 30 days. You may want to change this for production scenarios, see https://aka.ms/aspnetcore-hsts.

app.UseHsts();

}

app.UseHttpsRedirection();

app.UseStaticFiles();

app.UseRouting();

app.UseAuthorization();

app.MapRazorPages();

app.MapControllerRoute(

name: "default",

pattern: "{controller=Home}/{action=Index}/{id?}");

app.MapControllerRoute(

name: "default",

pattern: "{area}/{controller=Student}/{action=Index}/{id?}",

defaults: new { area = "Student" }); // specify the default area

app.Run();

Creating Repository

public interface IStudent

{

Task<int> SaveStudent(Students student);

Task<IEnumerable<Students>> GetStudent();

Task<Students> GetStudentById(int id);

Task<string> CheckStuCode(string stuCode);

Task<bool> DeleteStudentById(int id);

}

public class StudentService : IStudent

{

private readonly AppDbContext \_context;

public StudentService(AppDbContext context)

{

\_context = context;

}

//Student Info

public async Task<int> SaveStudent(Students student)

{

if (student.Id != 0)

\_context.Students.Update(student);

else

\_context.Students.Add(student);

await \_context.SaveChangesAsync();

return student.Id;

}

public async Task<IEnumerable<Students>> GetStudent()

{

return await \_context.Students.AsNoTracking().ToListAsync();

}

public async Task<Students> GetStudentById(int id)

{

return await \_context.Students.FindAsync(id);

}

public async Task<string> CheckStuCode(string stuCode)

{

var result = await \_context.Students.Where(x => x.studentCode == stuCode).Select(x => x.studentCode).FirstOrDefaultAsync();

if (result != null)

{

result = "duplicate";

return result;

}

return result;

}

public async Task<bool> DeleteStudentById(int id)

{

\_context.Students.Remove(\_context.Students.Find(id));

return 1 == await \_context.SaveChangesAsync();

}

}

}

Create viewmodel

public class StudentViewModels

{

public int studentId { get; set; }

public string studentCode { get; set; }

public string studentName { get; set; }

public string studentDepartment { get; set; }

public string studentFee { get; set; }

public IEnumerable<Students> students;

}

@model StudentViewModels

@{

ViewData["Title"] = "Index";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<div class="card" style="width: 100%;">

<div class="card-body">

<h5 class="card-title" style="color: black">Student Information</h5>

<hr>

<div class="container">

<**form** **asp-area**="Student" id="FormId" **asp-controller**="Student" **asp-action**="Index" method="post" data-parsley-validate="parsley">

<**div** **asp-validation-summary**="All" class="text-danger"></**div**>

<input type="hidden" name="studentId" id="studentId" value="0" />

<div class="form-group row">

<label for="studentCode" class="col-sm-2 col-form-label">Student Code</label>

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

<input type="text" class="form-control" maxlength="4" name="studentCode" id="studentCode" onchange="CheckStuCode()" required>

</div>

</div>

<div class="form-group row">

<label for="studentName" class="col-sm-2 col-form-label">Student Name </label>

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

<input type="text" data-parsley-required="true" data-parsley-trigger="keyup" class="form-control" name="studentName" id="studentName" required>

</div>

</div>

<div class="form-group row">

<label for="studentDepartment" class="col-sm-2 col-form-label">Department</label>

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

<input type="text" class="form-control" name="studentDepartment" id="studentDepartment" required>

</div>

</div>

<div class="form-group row">

<label for="studentFee" class="col-sm-2 col-form-label">Admission Fee</label>

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

<input type="number" class="form-control" name="studentFee" id="studentFee" required>

</div>

</div>

<hr />

<button type="submit" id="submit" value="Submit" class="btn btn-success btn-lg" style="float:right; margin-top:5px;">Save</button>

</**form**>

</div>

</div>

</div>

<br />

<hr />

<div class="row clearfix">

<div class="col-12">

<div class="card mb-4">

<div class="card-body">

<table class="table table-striped table-bordered" id="example">

<thead>

<tr>

<th>SL. No</th>

<th>Student Code</th>

<th>Student Name</th>

<th>Department</th>

<th>Admission Fee</th>

<th>Action</th>

</thead>

<tbody>

@{

var i = 1;

@foreach (var data in Model.students)

{

<tr>

<td>@i</td>

<td>@data.studentCode</td>

<td>@data.studentName</td>

<td>@data.studentDepartment</td>

<td>@data.studentFee</td>

<td>

<a class="btn btn-success" onclick="Edit(@data.Id, '@data.studentCode', '@data.studentName', '@data.studentDepartment', '@data.studentFee')" href="#">Edit</a>

<a class="btn btn-danger" onclick="Remove(@data.Id)" href="#">Delete</a>

</td>

</tr>

i++;

}

}

</tbody>

</table>

</div>

</div>

</div>

</div>

<script type="text/javascript">

$(document).ready(function () {

//$('#example').DataTable();

});

function Edit(Id, studentCode, studentName, studentDepartment, studentFee) {

$("#studentId").val(Id);

$("#studentCode").val(studentCode);

$("#studentName").val(studentName);

$("#studentDepartment").val(studentDepartment);

$("#studentFee").val(studentFee);

}

function Remove(Id) {

location.replace("/Student/Student/Delete/" + Id);

}

function CheckStuCode() {

var stuCode = $('#studentCode').val();

if (stuCode != "") {

$.ajax({

url: '/Student/Student/CheckEmpCode?empCode=' + stuCode,

type: "GET",

success: function (response) {

if (response == 'duplicate') {

alert('This Student Code is Already Exist!');

$('#studentCode').val('');

return false;

}

},

failure: function () {

alert('failed');

}

});

} }

</script>