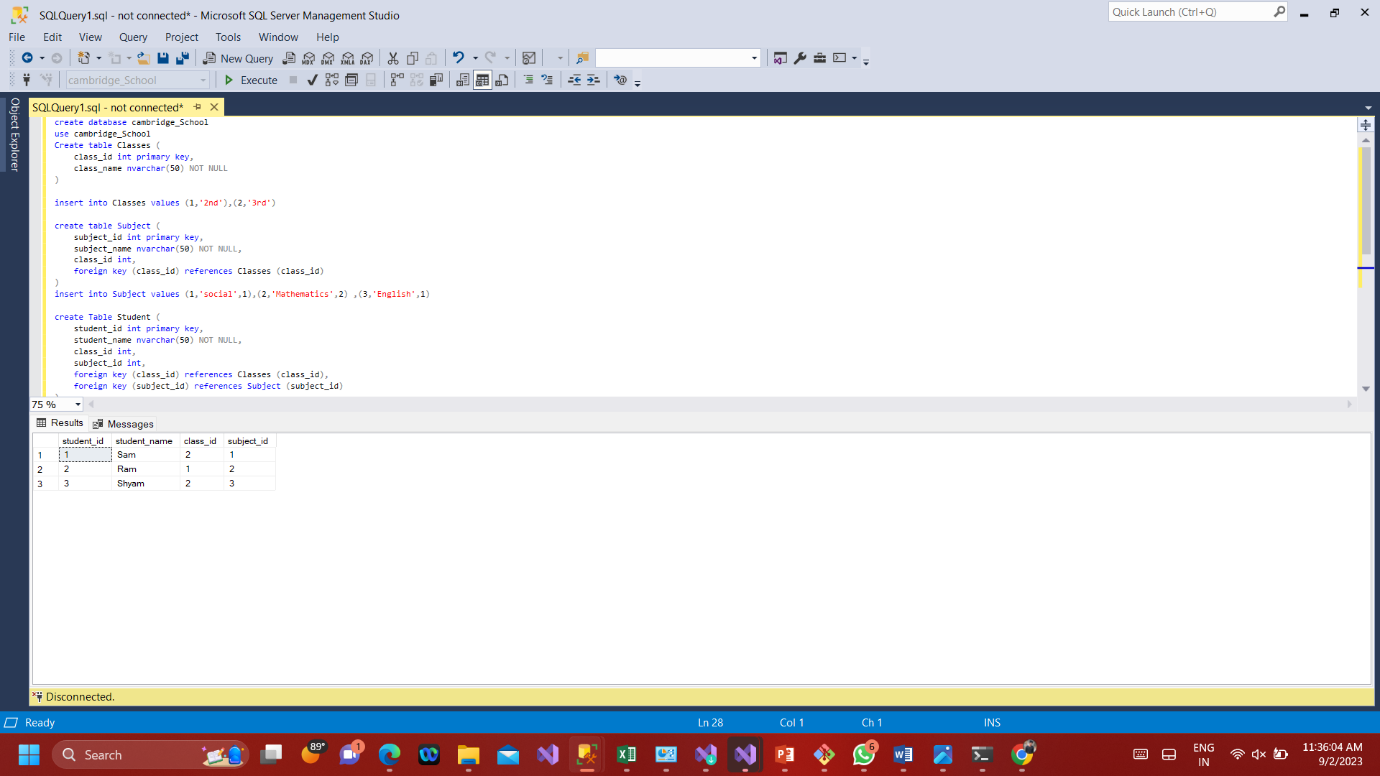
**Sql-query**

****

**A computer screen shot of a computer screen

Description automatically generatedVisual studio 2022**

**ClassesController.cs 🡺Code**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.Rendering;

using Microsoft.EntityFrameworkCore;

using SchooldataDb.Models;

namespace SchooldataDb.Controllers

{

public class ClassesController : Controller

{

private readonly CambridgeSchoolContext \_context;

public ClassesController(CambridgeSchoolContext context)

{

\_context = context;

}

// GET: Classes

public async Task<IActionResult> Index()

{

return \_context.Classes != null ?

View(await \_context.Classes.ToListAsync()) :

Problem("Entity set 'CambridgeSchoolContext.Classes' is null.");

}

// GET: Classes/Details/5

public async Task<IActionResult> Details(int? id)

{

if (id == null || \_context.Classes == null)

{

return NotFound();

}

var @class = await \_context.Classes

.FirstOrDefaultAsync(m => m.ClassId == id);

if (@class == null)

{

return NotFound();

}

return View(@class);

}

// GET: Classes/Create

public IActionResult Create()

{

return View();

}

// POST: Classes/Create

// To protect from overposting attacks, enable the specific properties you want to bind to.

// For more details, see http://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Create([Bind("ClassId,ClassName")] Class @class)

{

if (ModelState.IsValid)

{

\_context.Add(@class);

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

return View(@class);

}

// GET: Classes/Edit/5

public async Task<IActionResult> Edit(int? id)

{

if (id == null || \_context.Classes == null)

{

return NotFound();

}

var @class = await \_context.Classes.FindAsync(id);

if (@class == null)

{

return NotFound();

}

return View(@class);

}

// POST: Classes/Edit/5

// To protect from overposting attacks, enable the specific properties you want to bind to.

// For more details, see http://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Edit(int id, [Bind("ClassId,ClassName")] Class @class)

{

if (id != @class.ClassId)

{

return NotFound();

}

if (ModelState.IsValid)

{

try

{

\_context.Update(@class);

await \_context.SaveChangesAsync();

}

catch (DbUpdateConcurrencyException)

{

if (!ClassExists(@class.ClassId))

{

return NotFound();

}

else

{

throw;

}

}

return RedirectToAction(nameof(Index));

}

return View(@class);

}

// GET: Classes/Delete/5

public async Task<IActionResult> Delete(int? id)

{

if (id == null || \_context.Classes == null)

{

return NotFound();

}

var @class = await \_context.Classes

.FirstOrDefaultAsync(m => m.ClassId == id);

if (@class == null)

{

return NotFound();

}

return View(@class);

}

// POST: Classes/Delete/5

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public async Task<IActionResult> DeleteConfirmed(int id)

{

if (\_context.Classes == null)

{

return Problem("Entity set 'CambridgeSchoolContext.Classes' is null.");

}

var @class = await \_context.Classes.FindAsync(id);

if (@class != null)

{

\_context.Classes.Remove(@class);

}

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

private bool ClassExists(int id)

{

return (\_context.Classes?.Any(e => e.ClassId == id)).GetValueOrDefault();

}

}

}

**SubjectController.cs 🡺Code**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc.Rendering;

using Microsoft.EntityFrameworkCore;

using SchooldataDb.Models;

namespace SchooldataDb.Controllers

{

public class SubjectsController : Controller

{

private readonly CambridgeSchoolContext \_context;

public SubjectsController(CambridgeSchoolContext context)

{

\_context = context;

}

// GET: Subjects

public async Task<IActionResult> Index()

{

var cambridgeSchoolContext = \_context.Subjects.Include(s => s.Class);

return View(await cambridgeSchoolContext.ToListAsync());

}

// GET: Subjects/Details/5

public async Task<IActionResult> Details(int? id)

{

if (id == null || \_context.Subjects == null)

{

return NotFound();

}

var subject = await \_context.Subjects

.Include(s => s.Class)

.FirstOrDefaultAsync(m => m.SubjectId == id);

if (subject == null)

{

return NotFound();

}

return View(subject);

}

// GET: Subjects/Create

public IActionResult Create()

{

ViewData["ClassId"] = new SelectList(\_context.Classes, "ClassId", "ClassId");

return View();

}

// POST: Subjects/Create

// To protect from overposting attacks, enable the specific properties you want to bind to.

// For more details, see http://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Create([Bind("SubjectId,SubjectName,ClassId")] Subject subject)

{

if (ModelState.IsValid)

{

\_context.Add(subject);

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

ViewData["ClassId"] = new SelectList(\_context.Classes, "ClassId", "ClassId", subject.ClassId);

return View(subject);

}

// GET: Subjects/Edit/5

public async Task<IActionResult> Edit(int? id)

{

if (id == null || \_context.Subjects == null)

{

return NotFound();

}

var subject = await \_context.Subjects.FindAsync(id);

if (subject == null)

{

return NotFound();

}

ViewData["ClassId"] = new SelectList(\_context.Classes, "ClassId", "ClassId", subject.ClassId);

return View(subject);

}

// POST: Subjects/Edit/5

// To protect from overposting attacks, enable the specific properties you want to bind to.

// For more details, see http://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Edit(int id, [Bind("SubjectId,SubjectName,ClassId")] Subject subject)

{

if (id != subject.SubjectId)

{

return NotFound();

}

if (ModelState.IsValid)

{

try

{

\_context.Update(subject);

await \_context.SaveChangesAsync();

}

catch (DbUpdateConcurrencyException)

{

if (!SubjectExists(subject.SubjectId))

{

return NotFound();

}

else

{

throw;

}

}

return RedirectToAction(nameof(Index));

}

ViewData["ClassId"] = new SelectList(\_context.Classes, "ClassId", "ClassId", subject.ClassId);

return View(subject);

}

// GET: Subjects/Delete/5

public async Task<IActionResult> Delete(int? id)

{

if (id == null || \_context.Subjects == null)

{

return NotFound();

}

var subject = await \_context.Subjects

.Include(s => s.Class)

.FirstOrDefaultAsync(m => m.SubjectId == id);

if (subject == null)

{

return NotFound();

}

return View(subject);

}

// POST: Subjects/Delete/5

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public async Task<IActionResult> DeleteConfirmed(int id)

{

if (\_context.Subjects == null)

{

return Problem("Entity set 'CambridgeSchoolContext.Subjects' is null.");

}

var subject = await \_context.Subjects.FindAsync(id);

if (subject != null)

{

\_context.Subjects.Remove(subject);

}

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

private bool SubjectExists(int id)

{

return (\_context.Subjects?.Any(e => e.SubjectId == id)).GetValueOrDefault();

}

}

}

**StudentController.cs 🡺Code**

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.Entity;

using System.Linq;

using System.Net;

using System.Web;

using System.Web.Mvc;

using Application\_to\_Manage\_Data\_for\_School\_Application.Models;

namespace Application\_to\_Manage\_Data\_for\_School\_Application.Controllers

{

public class StudentsController : Controller

{

private Rain\_SchoolEntities db = new Rain\_SchoolEntities();

// GET: Students

public ActionResult Index()

{

var students = db.Students.Include(s => s.Class).Include(s => s.Subject);

return View(students.ToList());

}

// GET: Students/Details/5

public ActionResult Details(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Student student = db.Students.Find(id);

if (student == null)

{

return HttpNotFound();

}

return View(student);

}

// GET: Students/Create

public ActionResult Create()

{

ViewBag.class\_id = new SelectList(db.Classes, "class\_id", "class\_name");

ViewBag.subject\_id = new SelectList(db.Subjects, "subject\_id", "subject\_name");

return View();

}

// POST: Students/Create

// To protect from overposting attacks, enable the specific properties you want to bind to, for

// more details see https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Create([Bind(Include = "student\_id,student\_name,class\_id,subject\_id")] Student student)

{

if (ModelState.IsValid)

{

db.Students.Add(student);

db.SaveChanges();

return RedirectToAction("Index");

}

ViewBag.class\_id = new SelectList(db.Classes, "class\_id", "class\_name", student.class\_id);

ViewBag.subject\_id = new SelectList(db.Subjects, "subject\_id", "subject\_name", student.subject\_id);

return View(student);

}

// GET: Students/Edit/5

public ActionResult Edit(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Student student = db.Students.Find(id);

if (student == null)

{

return HttpNotFound();

}

ViewBag.class\_id = new SelectList(db.Classes, "class\_id", "class\_name", student.class\_id);

ViewBag.subject\_id = new SelectList(db.Subjects, "subject\_id", "subject\_name", student.subject\_id);

return View(student);

}

// POST: Students/Edit/5

// To protect from overposting attacks, enable the specific properties you want to bind to, for

// more details see https://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult Edit([Bind(Include = "student\_id,student\_name,class\_id,subject\_id")] Student student)

{

if (ModelState.IsValid)

{

db.Entry(student).State = EntityState.Modified;

db.SaveChanges();

return RedirectToAction("Index");

}

ViewBag.class\_id = new SelectList(db.Classes, "class\_id", "class\_name", student.class\_id);

ViewBag.subject\_id = new SelectList(db.Subjects, "subject\_id", "subject\_name", student.subject\_id);

return View(student);

}

// GET: Students/Delete/5

public ActionResult Delete(int? id)

{

if (id == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Student student = db.Students.Find(id);

if (student == null)

{

return HttpNotFound();

}

return View(student);

}

// POST: Students/Delete/5

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public ActionResult DeleteConfirmed(int id)

{

Student student = db.Students.Find(id);

db.Students.Remove(student);

db.SaveChanges();

return RedirectToAction("Index");

}

protected override void Dispose(bool disposing)

{

if (disposing)

{

db.Dispose();

}

base.Dispose(disposing);

}

}

}

**\_Layout.cshtml 🡺Code**

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

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

<title>@ViewBag.Title - My ASP.NET Application</title>

@Styles.Render("~/Content/css")

@Scripts.Render("~/bundles/modernizr")

</head>

<body>

<nav class="navbar navbar-expand-sm navbar-toggleable-sm navbar-dark bg-dark">

<div class="container">

@Html.ActionLink("Application name", "Index", "Home", new { area = "" }, new { @class = "navbar-brand" })

<button type="button" class="navbar-toggler" data-bs-toggle="collapse" data-bs-target=".navbar-collapse" title="Toggle navigation" aria-controls="navbarSupportedContent"

aria-expanded="false" aria-label="Toggle navigation">

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse d-sm-inline-flex justify-content-between">

<ul class="navbar-nav flex-grow-1">

<li>@Html.ActionLink("Home", "Index", "Home", new { area = "" }, new { @class = "nav-link" })</li>

<li>@Html.ActionLink("About", "About", "Home", new { area = "" }, new { @class = "nav-link" })</li>

<li>@Html.ActionLink("Contact", "Contact", "Home", new { area = "" }, new { @class = "nav-link" })</li>

<li>@Html.ActionLink("Class", "Index", "Classes", new { area = "" }, new { @class = "nav-link" })</li>

<li>@Html.ActionLink("Subject", "Index", "Subjects", new { area = "" }, new { @class = "nav-link" })</li>

<li>@Html.ActionLink("Students Details", "Index", "Students", new { area = "" }, new { @class = "nav-link" })</li>

</ul>

</div>

</div>

</nav>

<div class="container body-content">

@RenderBody()

<hr />

<footer>

<p>&copy; @DateTime.Now.Year - My ASP.NET Application</p>

</footer>

</div>

@Scripts.Render("~/bundles/jquery")

@Scripts.Render("~/bundles/bootstrap")

@RenderSection("scripts", required: false)

</body>

</html>