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 PracticeProjectManageSchoolData.Models;

namespace PracticeProjectManageSchoolData.Controllers

{

public class StudentsController : Controller

{

private readonly SchoolContext \_context;

public StudentsController(SchoolContext context)

{

\_context = context;

}

// GET: Students

public async Task<IActionResult> Index()

{

return \_context.Student != null ?

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

Problem("Entity set 'SchoolContext.Student' is null.");

}

// GET: Students/Details/5

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

{

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

{

return NotFound();

}

var student = await \_context.Student

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

if (student == null)

{

return NotFound();

}

return View(student);

}

// GET: Students/Create

public IActionResult Create()

{

return View();

}

// POST: Students/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("Id,Name,Address,Email,Class")] Student student)

{

if (ModelState.IsValid)

{

\_context.Add(student);

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

return View(student);

}

// GET: Students/Edit/5

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

{

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

{

return NotFound();

}

var student = await \_context.Student.FindAsync(id);

if (student == null)

{

return NotFound();

}

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 http://go.microsoft.com/fwlink/?LinkId=317598.

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Edit(int id, [Bind("Id,Name,Address,Email,Class")] Student student)

{

if (id != student.Id)

{

return NotFound();

}

if (ModelState.IsValid)

{

try

{

\_context.Update(student);

await \_context.SaveChangesAsync();

}

catch (DbUpdateConcurrencyException)

{

if (!StudentExists(student.Id))

{

return NotFound();

}

else

{

throw;

}

}

return RedirectToAction(nameof(Index));

}

return View(student);

}

// GET: Students/Delete/5

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

{

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

{

return NotFound();

}

var student = await \_context.Student

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

if (student == null)

{

return NotFound();

}

return View(student);

}

// POST: Students/Delete/5

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public async Task<IActionResult> DeleteConfirmed(int id)

{

if (\_context.Student == null)

{

return Problem("Entity set 'SchoolContext.Student' is null.");

}

var student = await \_context.Student.FindAsync(id);

if (student != null)

{

\_context.Student.Remove(student);

}

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

private bool StudentExists(int id)

{

return (\_context.Student?.Any(e => e.Id == id)).GetValueOrDefault();

}

}

}