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 ClassesController : Controller

{

private readonly SchoolContext \_context;

public ClassesController(SchoolContext context)

{

\_context = context;

}

// GET: Classes

public async Task<IActionResult> Index()

{

return \_context.Class != null ?

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

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

}

// GET: Classes/Details/5

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

{

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

{

return NotFound();

}

var @class = await \_context.Class

.FirstOrDefaultAsync(m => m.Id == 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("Id,Name")] 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.Class == null)

{

return NotFound();

}

var @class = await \_context.Class.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("Id,Name")] Class @class)

{

if (id != @class.Id)

{

return NotFound();

}

if (ModelState.IsValid)

{

try

{

\_context.Update(@class);

await \_context.SaveChangesAsync();

}

catch (DbUpdateConcurrencyException)

{

if (!ClassExists(@class.Id))

{

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.Class == null)

{

return NotFound();

}

var @class = await \_context.Class

.FirstOrDefaultAsync(m => m.Id == 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.Class == null)

{

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

}

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

if (@class != null)

{

\_context.Class.Remove(@class);

}

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

private bool ClassExists(int id)

{

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

}

}

}