using Microsoft.EntityFrameworkCore; using Microsoft.Extensions.DependencyInjection; using PracticeProblem_4.Data; var builder = WebApplication.CreateBuilder(args); builder.Services.AddDbContext<PlayersDb2Context>(options => options.UseSqlServer(builder.Configuration.GetConnectionString("PlayersDb2Context") ?? throw new InvalidOperationException("Connection string 'PlayersDb2Context' not found."))); // Add services to the container. builder.Services.AddControllersWithViews(); var app = builder.Build(); // Configure the HTTP request pipeline. if (!app.Environment.IsDevelopment()) { app.UseExceptionHandler("/Home/Error"); } app.UseStaticFiles(); app.UseRouting(); app.UseAuthorization(); app.MapControllerRoute(name: "default", pattern: "{controller=Home}/{action=Index}/{id?}"); app.Run();

{ "Logging": { "LogLevel": { "Default": "Information", "Microsoft.AspNetCore": "Warning" } }, "AllowedHosts": "*", "ConnectionStrings": { "PlayersDb2Context": "Server=tcp:pavan123server.database.windows.net,1433;Initial Catalog=mydbs123;Persist Security Info=False;User ID=pavanadmin;Password=123456@Pavan;MultipleActiv Timeout=30;" } }

<!DOCTYPE html> <html lang="en"> <head> <meta charset="utf-8" /> <meta name="viewport" content="width=device-width, initial-scale=1.0" /> <title>@ViewData["Title"] - PracticeProblem_4</title> <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" /> <link rel="stylesheet" href="~/css/site.css" asp-append-version="true" /> <link rel="stylesheet" href="~/PracticeProblem_4.styles.css" asp-append-version="true" /> </head> <body> <header> <nav class="navbar navbar-expand-sm navbar-toggleable-sm navbar-light bg-white border-bottom box-shadow mb-3"> PracticeProblem_4 <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target=".navbar-collapse" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation"> </button> <ul class="navbar-nav flex-grow-1"> <li class="nav-item"> Home <li class="nav-item"> Privacy <li class="nav-item"> Players   </nav>

</header> <main role="main" class="pb-3"> @RenderBody() </main>

<footer class="border-top footer text-muted"> © 2024 - PracticeProblem_4 - <a asp-area="" asp-controller="Home" asp-action="Privacy">Privacy </footer> <script src="~/lib/jquery/dist/jquery.min.js"></scr <script src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script> <script src="~/js/site.js" asp-append-version="true"></script> @await RenderSectionAsync("Scripts", required: false) </body> </html>

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 PracticeProblem_4.Data; using PracticeProblem_4.Models;

namespace PracticeProblem_4.Controllers { public class PlayersController : Controller { private readonly PlayersDb2Context _context;

public PlayersController(PlayersDb2Context context) { _context = context; }

// GET: Players public async Task<IActionResult> Index() { return _context.Player != null ? View(await _context.Player.ToListAsync()) : Problem("Entity set 'PlayersDb2Context.Player' is null."); }

// GET: Players/Details/5 public async Task<IActionResult> Details(int? id) { if (id == null || _context.Player == null) { return NotFound(); }

var player = await _context.Player .FirstOrDefaultAsync(m => m.Id == id); if (player == null) { return NotFound(); }

return View(player); }

// GET: Players/Create public IActionResult Create() { return View(); }

// POST: Players/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,Salary,Designation")] Player player) { if (ModelState.IsValid) { _context.Add(player); await _context.SaveChangesAsync(); return RedirectToAction(nameof(Index)); } return View(player); }

// GET: Players/Edit/5 public async Task<IActionResult> Edit(int? id) { if (id == null || _context.Player == null) { return NotFound(); }

var player = await _context.Player.FindAsync(id); if (player == null) { return NotFound(); } return View(player); }

// POST: Players/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,Salary,Designation”)] Player player) { if (id != player.Id) { return NotFound(); }

if (ModelState.IsValid) { try { _context.Update(player); await _context.SaveChangesAsync(); } catch (DbUpdateConcurrencyException) { if (!PlayerExists(player.Id)) { return NotFound(); } else { throw; } } return RedirectToAction(nameof(Index)); } return View(player); }

// GET: Players/Delete/5 public async Task<IActionResult> Delete(int? id) { if (id == null || _context.Player == null) { return NotFound(); }

var player = await _context.Player .FirstOrDefaultAsync(m => m.Id == id); if (player == null) { return NotFound(); }

return View(player); }

// POST: Players/Delete/5 [HttpPost, ActionName(”Delete”)] [ValidateAntiForgeryToken] public async Task<IActionResult> DeleteConfirmed(int id) { if (_context.Player == null) { return Problem(”Entity set ’PlayersDb2Context.Player’ is null.”); } var player = await _context.Player.FindAsync(id); if (player != null) { _context.Player.Remove(player); }

await _context.SaveChangesAsync(); return RedirectToAction(nameof(Index)); }

private bool PlayerExists(int id) { return (_context.Player?.Any(e => e.Id == id)).GetValueOrDefault(); } } }

using System.ComponentModel.DataAnnotations; using System.ComponentModel.DataAnnotations.Schema;

namespace PracticeProblem_4.Models { [Table(”Player”)] public class Player { [Key] public int Id { get; set; } [Required] [StringLength(100)] public string Name { get; set; } public double Salary { get; set; } [Required] [StringLength(100)] public string Designation { get; set; } } }