Model Examen.pdf

Visual Studio Code 2022 - New Project -> ASP.NET Core Web App (Razor Pages) -> .NET 8

Instalam din NuGet:

- Microsoft.EntityFrameworkCore
- Microsoft.EntityFrameworkCore.SqlServer
- Microsoft.EntityFrameworkCore.Tools

Organizăm Proiectul în foldere: Models, Data, Pages

O clasă model exemplu:

Models/Student.cs

```
using Microsoft.AspNetCore.Mvc;

namespace PregatireExamenDAW.Models
{
    public class Student
    {

        public int StudentId { get; set; }
        public string Nume { get; set; }
        public string Prenume { get; set; }
        public string Email { get; set; }
        public DateOnly DataNasterii { get; set; }

        public int GrupaId {  get; set; }
        public Grupa Grupa { get; set; }

    }
}
```

Models/Grupa.cs

```
namespace PregatireExamenDAW.Models
{
    public class Grupa
    {
        public int GrupaId { get; set; }
        public string Denumire { get; set; }
        public string Domeniu { get; set; }

        public ICollection<Student> Studenti { get; set; }
    }
}
```

Realizăm conexiunea cu serverul SQL după crearea database-ului astfel:

appsettings.json

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "DefaultConnection": "Data Source=(localdb)\\ProjectModels;Initial
Catalog=PregatireDAW;Integrated Security=True;Connect
Timeout=30;Encrypt=False;Trust Server Certificate=False;Application
Intent=ReadWrite;Multi Subnet Failover=False"
    }
  }
```

Data/ApplicationDbContext.cs

```
using Microsoft.EntityFrameworkCore;
using PregatireExamenDAW.Models;

namespace PregatireExamenDAW.Data
{
    public class ApplicationDbContext : DbContext
    {
        public ApplicationDbContext(DbContextOptions<ApplicationDbContext>
options) : base(options)
        {

        }
        public DbSet<Student> Studenti { get; set; }
    }
}
```

Program.cs

```
builder.Services.AddDbContext<ApplicationDbContext>(options =>
options.UseSqlServer(
        builder.Configuration.GetConnectionString("DefaultConnection")
    ));
```

Pentru a updata baza de date cu tabelele noi create:

Tools > NuGet Package Manager > Package Manager Console

```
add-migration NumeMigration
update-database
```

Introducem manual date în tabele prin View -> SQL Server Object Explorer

Pentru a vizualiza toți studenții pe prima pagină, modificăm:

Index.cshtml.cs

```
namespace PregatireExamenDAW.Pages
{
    public class IndexModel : PageModel
    {
        private readonly ILogger<IndexModel> _logger;
        private readonly ApplicationDbContext _context;

        public IEnumerable<Student> Studenti { get; set; }

        public IndexModel(ILogger<IndexModel> logger, ApplicationDbContext db)
        {
            _logger = logger;
            _context = db;
        }

        public void OnGet()
        {
            Studenti = _context.Studenti.Include( s => s.Grupa);
        }
    }
}
```

Index.cshtml

```
@page
@model IndexModel
@{
    ViewData["Title"] = "Home page";
}

<div class="container">
    <h2>Lista Studentilor</h2>
    <table class="table table-bordered table-striped" style="width:100%">
        <thead>
```

```
      <tr>
        <th>Nume</th>
        <th>Prenume</th>
        <th>E-Mail</th>
        <th>Data Nasterii</th>
        <th>Grupa</th>
      </tr>
    </thead>
    <tbody>
      @foreach(var obj in Model.Studenti)
      {
        <tr>
          <td width="20%">@obj.Nume</td>
          <td width="20%">@obj.Prenume</td>
          <td width="20%">@obj.Email</td>
          <td width="20%">@obj.DataNasterii</td>
          <td width="20%">@obj.Grupa.Denumire</td>
        </tr>
      }
    </tbody>
  </table>
</div>
```

Pentru a adauga studenti noi, adaugam un nou Razor Page numit *AddStudent*

AddStudent.cshtml.cs

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.RazorPages;
using PregatireExamenDAW.Data;
using PregatireExamenDAW.Models;

namespace PregatireExamenDAW.Pages
{
  public class AddStudentModel : PageModel
  {
    private readonly ApplicationDbContext _context;
    [BindProperty]
    public Student Student {  get; set; }

        public AddStudentModel(ApplicationDbContext context)
        {
            _context = context;
        }

        public void OnGet()
    {
    }
```

```
        public async Task<IActionResult> OnPost()
        {
            await _context.Studenti.AddAsync(Student);
            await _context.SaveChangesAsync();
            return RedirectToPage("Index");
        }
    }
}
```

AddStudent.cshtml

```
@page
@model PregatireExamenDAW.Pages.AddStudentModel
@{
}

<form method="post">
        <div class="border p-3 mt-4">
                <div class="row pb-2">
                        <h2>Adaugare Student</h2>
                        <hr />
                </div>
                <div class="mb-3">
                        <label asp-for="Student.Nume"></label>
                        <input type="text" asp-for="Student.Nume" class="form-
control" />
                </div>
                <div class="mb-3">
                        <label asp-for="Student.Prenume"></label>
                        <input type="text" asp-for="Student.Prenume" class="form-
control" />
                </div>
                <div class="mb-3">
                        <label asp-for="Student.Email"></label>
                        <input type="text" asp-for="Student.Email" class="form-
control" />
                </div>
                <div class="mb-3">
                        <label asp-for="Student.DataNasterii"></label>
                        <input type="date" asp-for="Student.DataNasterii"
class="form-control" />
                </div>
                <div class="mb-3">
                        <label asp-for="Student.GrupaId"></label>
                        <input type="number" asp-for="Student.GrupaId"
class="form-control" />
                </div>
                <button type="submit" class="btn btn-primary">Adaugare</button>
        </div>
</form>
```

Adăugăm pagina nouă în _Layout.cshtml în navbar

_Layout.cshtml

```
(...)
                <div class="navbar-collapse collapse d-sm-inline-flex justify-content-between">
                    <ul class="navbar-nav flex-grow-1">
                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-area="" asp-page="/Index">Home</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-area="" asp-page="/Addstudent">Add Student</a>
                        </li>
                    </ul>
                </div>
(...)
```

Pentru a crea butoanele de Edit și Delete, vom edita pagina Index astfel:

Index.cshtml

```
(...)
        <tbody>
            @foreach(var obj in Model.Studenti)
            {
                <tr>
                    <td width="20%">@obj.Nume</td>
                    <td width="20%">@obj.Prenume</td>
                    <td width="20%">@obj.Email</td>
                    <td width="20%">@obj.DataNasterii</td>
                    <td width="20%">@obj.Grupa.Denumire</td>
                    <td>
                        <div class="w-75 btn-group" role="group">
                        <a asp-page="EditStudent" asp-route-id="@obj.StudentId" class="btn btn-primary mx-2">
                            Edit
                        </a>
                        <a asp-page="Delete" class="btn btn-danger mx-2">
                            Delete
                        </a>
                        </div>
                    </td>
                </tr>
            }
        </tbody>
(...)
```

Pentru că pagina de Edit va avea același format ca cea de AddStudent, putem duplica fișierul AddStudent.cshtml și redenumi corespunzător interiorul ei din AddStudentModel în EditStudentModel

Pentru logică, vom avea:

EditStudent.cshtml.cs

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.RazorPages;
using PregatireExamenDAW.Data;
using PregatireExamenDAW.Models;

namespace PregatireExamenDAW.Pages
{
    public class EditStudentModel : PageModel
    {
        private readonly ApplicationDbContext _context;
        [BindProperty]
        public Student Student {  get; set; }

                public EditStudentModel(ApplicationDbContext context)
                {
                        _context = context;
                }

                public void OnGet(int id)
        {
            Student = _context.Studenti.FirstOrDefault(u => u.StudentId == id);
        }

        public async Task<IActionResult> OnPost()
        {
             _context.Studenti.Update(Student);
            await _context.SaveChangesAsync();
            return RedirectToPage("Index");
        }
    }
}
```

Iar pentru Razor page vom face urmatoarele modificări:

EditStudent.cshtml

```
@page
@model PregatireExamenDAW.Pages.EditStudentModel
@{
}

<form method="post">
        <input hidden asp-for="Student.StudentId" />
        <div class="border p-3 mt-4">
                <div class="row pb-2">
                        <h2>Editare Student</h2>
                        <hr />
                </div>
                <div class="mb-3">
                        <label asp-for="Student.Nume"></label>
                        <input type="text" asp-for="Student.Nume" class="form-control" />
                </div>
                <div class="mb-3">
                        <label asp-for="Student.Prenume"></label>
                        <input type="text" asp-for="Student.Prenume" class="form-control" />
                </div>
                <div class="mb-3">
                        <label asp-for="Student.Email"></label>
                        <input type="text" asp-for="Student.Email" class="form-control" />
                </div>
                <div class="mb-3">
                        <label asp-for="Student.DataNasterii"></label>
                        <input type="date" asp-for="Student.DataNasterii" class="form-control" />
                </div>
                <div class="mb-3">
                        <label asp-for="Student.GrupaId"></label>
                        <input type="number" asp-for="Student.GrupaId" class="form-control" />
                </div>
                <button type="submit" class="btn btn-primary">Update</button>
        </div>
</form>
```

Analog duplicării de mai sus, facem același lucru pentru pagina DeleteStudent, cu excepția faptului că DeleteStudent.cshtml nu conține nimic.

DeleteStudent.cshtml.cs

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.RazorPages;
using PregatireExamenDAW.Data;
using PregatireExamenDAW.Models;

namespace PregatireExamenDAW.Pages
{
    public class DeleteStudentModel : PageModel
    {
        private readonly ApplicationDbContext _context;
        [BindProperty]
        public Student Student {  get; set; }

                public DeleteStudentModel(ApplicationDbContext context)
                {
                        _context = context;
                }

                public async Task<IActionResult> OnGet(int id)
        {
            var studentFromDb = _context.Studenti.FirstOrDefault(u => u.StudentId
== id);
            if (studentFromDb != null)
            {
                _context.Studenti.Remove(studentFromDb);
                await _context.SaveChangesAsync();

            }
            return RedirectToPage("Index");

        }
    }
}
```

Pentru a adauga o lista cu grupele și numărul membrilor fiecăruia, vom face câteva modificări mici în pagina Index:

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.RazorPages;
using Microsoft.EntityFrameworkCore;
using PregatireExamenDAW.Data;
using PregatireExamenDAW.Models;

namespace PregatireExamenDAW.Pages
{
    public class IndexModel : PageModel
    {
        private readonly ILogger<IndexModel> _logger;
        private readonly ApplicationDbContext _context;

        public IEnumerable<Student> Studenti { get; set; }
        // Adăugăm grupele
        public IEnumerable<Grupa> Grupe { get; set; }


        public IndexModel(ILogger<IndexModel> logger, ApplicationDbContext
context)
        {
            _logger = logger;
            _context = context;
        }

        public void OnGet()
        {
            Studenti = _context.Studenti.Include( s => s.Grupa);
            // Și le pasăm valori aici
            Grupe = _context.Grupe.Include(s => s.Studenti);
        }
    }
}
```

```
@page
@model IndexModel
@{
    ViewData["Title"] = "Home page";
}

<div class="container">
    <h2>Lista Studentilor</h2>
    <table class="table table-bordered table-striped" style="width:100%">
        <thead>
            <tr>
                <th>Nume</th>
                <th>Prenume</th>
                <th>E-Mail</th>
                <th>Data Nasterii</th>
                <th>Grupa</th>
            </tr>
        </thead>
        <tbody>
            @foreach(var obj in Model.Studenti)
            {
                <tr>
                    <td width="20%">@obj.Nume</td>
                    <td width="20%">@obj.Prenume</td>
                    <td width="20%">@obj.Email</td>
                    <td width="20%">@obj.DataNasterii</td>
                    <td width="20%">@obj.Grupa.Denumire</td>
                    <td>
                        <div class="w-75 btn-group" role="group">
                        <a asp-page="EditStudent" asp-route-id="@obj.StudentId"
class="btn btn-primary mx-2">
                            Edit
                        </a>
                        <a asp-page="DeleteStudent" asp-route-id="@obj.StudentId"
class="btn btn-danger mx-2"
                               onclick="return confirm('Sigur vreti sa stergeti
acest student?');">
                            Delete
                        </a>
                        </div>
                    </td>
                </tr>
            }
        </tbody>
    </table>
<!-- Modificarile incep de aici -->
    <div>
        <h2>Numarul studentilor din fiecare grupa</h2>
        <ul>
            @foreach(var grupa in Model.Grupe)
            {
                <li>
                    @grupa.Denumire: @grupa.Studenti.Count
                </li>
            }
        </ul>
    </div>
</div>
```