shorturl.at/ghpZ8

29/08/2022

IProductService.cs

```
using ProductManager.Models;

namespace ProductManager.Services
{
    public interface IProductService
    {
       List<Product> GetProducts();
       Product? GetProductById(int id);
       void CreateProduct(Product product);
       void UpdateProduct(Product product);
       void DeleteProduct(int id);
       List<Category> GetCategories();
    }
}
```

ProductService.cs

```
using ProductManager.Models;
using Microsoft.EntityFrameworkCore;

namespace ProductManager.Services
{
    public class ProductService : IProductService
    {
        private readonly DataContext _context;
        public ProductService(DataContext context)
        {
            _context = context;
```

```
}

public void CreateProduct(Product product)
{
    _context.Products.Add(product);
    _context.SaveChanges();
}

public void DeleteProduct(int id)
{
    var existedProduct = GetProductById(id);
    if (existedProduct == null) return;
    _context.Products.Remove(existedProduct);
    _context.SaveChanges();
}

public List<Category> GetCategories()
{
    return _context.Categories.ToList();
}

public Product? GetProductById(int id)
{
    return _context.Products.FirstOrDefault(p => p.Id == id);
}

public List<Product> GetProducts()
{
    return _context.Products
        .Include(p => p.Category)
        .ToList();
}
```

```
        public void UpdateProduct(Product product)
        {
            var existedProduct = GetProductById(product.Id);
            if (existedProduct == null) return;
            existedProduct.Name = product.Name;
            existedProduct.Slug = product.Slug;
            existedProduct.Price = product.Price;
            existedProduct.Quantity = product.Quantity;
            existedProduct.CategoryId = product.CategoryId;
            _context.Products.Update(existedProduct);
            _context.SaveChanges();
        }
    }
}
```

Create.cshtml

```
@model List<Category>;
@{
    ViewData["Title"] = "Product Create Page";
    var categories = Model;
}
<h1 class="display-4">Product Create Page</h1>
<form action="/Product/Save" method="POST">
    <div class="form-group">
        <label for="Name">Name</label>
        <input type="text" class="form-control" id="Name" name="Name">
    </div>
    <div class="form-group">
        <label for="Slug">Slug</label>
        <input type="text" class="form-control" id="Slug" name="Slug">
    </div>
```

```
    <div class="form-group">
        <label for="Price">Price</label>
        <input type="number" class="form-control" id="Price" name="Price">
    </div>
    <div class="form-group">
        <label for="Quantity">Quantity</label>
        <input type="number" class="form-control" id="Quantity" name="Quantity">
    </div>
    <div class="form-group">
        <label for="Category">Category</label>
        <select class="form-control" id="CategoryId" name="CategoryId">
            @foreach (var category in categories)
            {
                <option value="@category.Id">@category.Name</option>
            }
        </select>
    </div>
    <button type="submit" class="btn btn-primary">Submit</button>
</form>
```

Update.cshtml

```
@model List<Category>;
@{
    ViewData["Title"] = "Product Create Page";
    var categories = Model;
    var product = ViewBag.Product;
}
<h1 class="display-4">Product Create Page</h1>
<form action="/Product/Save" method="POST">
    <div class="form-group">
        <label for="Id">Id</label>
```

```
        <input type="text" class="form-control" id="Id" name="Id" value="@product.Id" readonly="readonly">
    </div>
    <div class="form-group">
        <label for="Name">Name</label>
        <input type="text" class="form-control" id="Name" name="Name" value="@product.Name">
    </div>
    <div class="form-group">
        <label for="Slug">Slug</label>
        <input type="text" class="form-control" id="Slug" name="Slug" value="@product.Slug">
    </div>
    <div class="form-group">
        <label for="Price">Price</label>
        <input type="number" class="form-control" id="Price" name="Price" value="@product.Price">
    </div>
    <div class="form-group">
        <label for="Quantity">Quantity</label>
        <input type="number" class="form-control" id="Quantity" name="Quantity" value="@product.Quantity">
    </div>
    <div class="form-group">
        <label for="Category">Category</label>
        <select class="form-control" id="CategoryId" name="CategoryId">
            @foreach (var category in categories)
            {
                @if (category.Id == product.CategoryId)
                {
                    <option value="@category.Id" selected="selected">@category.Name</option>
                }
                else
                {
                    <option value="@category.Id">@category.Name</option>
                }
            }
```

```
        </select>
    </div>
    <button type="submit" class="btn btn-primary">Submit</button>
</form>
```

Index.cshtml

```
@model List<Product>;
@{
    ViewData["Title"] = "Product Page";
    var products = Model;
    <!-- var categories = ViewBag.Categories; -->
    var categories = ViewData["Categories"] as List<string>;
}

<div class="text-center">
    <h1 class="display-4">Welcome ProductManager</h1>
    <table class="table table-striped table-hover">
        <thead>
            <th>ID</th>
            <th>Name</th>
            <th>Price</th>
            <th>Quantity</th>
            <th>Category</th>
            <th>
                <a href="/Product/Create" class="btn btn-primary">Create</a>
            </th>
        </thead>
        <tbody>
            @foreach (var product in products)
            {
                <tr>
```

```
                    <td>@product.Id</td>
                    <td>@product.Name</td>
                    <td>@product.Price</td>
                    <td>@product.Quantity</td>
                    <td>@product.Category.Name</td>
                    <td>
                        <a href="/Product/Update?id=@product.Id" class="btn btn-success">Update</a>
                        <a href="/Product/Delete?id=@product.Id" class="btn btn-danger">Delete</a>
                    </td>
                </tr>
            }
        </tbody>
    </table>
</div>
```

ProductController.cs

```
using Microsoft.AspNetCore.Mvc;
using ProductManager.Models;
using ProductManager.Services;

namespace ProductManager.Controllers
{
    public class ProductController : Controller
    {
        private readonly IProductService _productService;
        public ProductController(IProductService productService)
        {
            _productService = productService;
        }

        public IActionResult Index()
        {
            var products = _productService.GetProducts();
            return View(products);
        }

        public IActionResult Create()
        {
            var categories = _productService.GetCategories();
            return View(categories);
        }

        public IActionResult Update(int id)
        {
            var product = _productService.GetProductById(id);
            if (product == null) return RedirectToAction("Create");
```

```
            var categories = _productService.GetCategories();
            ViewBag.Product = product;
            return View(categories);
        }

        public IActionResult Delete(int id)
        {
            _productService.DeleteProduct(id);
            return RedirectToAction("Index");
        }

        public IActionResult Save(Product product)
        {
            if (product.Id == 0)
            {
                _productService.CreateProduct(product);
            }
            else
            {
                _productService.UpdateProduct(product);
            }
            return RedirectToAction("Index");
        }

    }
}
```

28/08/2022

ViewBag & ViewData

- dotnet add package Microsoft.EntityFrameworkCore
- dotnet add package Microsoft.EntityFrameworkCore.Design
- dotnet add package Microsoft.EntityFrameworkCore.SqlServer
- dotnet add package Pomelo.EntityFrameworkCore.MySql
- dotnet tool install --global dotnet-ef
- dotnet-ef migrations add InitialDb
- dotnet-ef database update

- dotnet-ef migrations add AddSlugIntoProduct
- dotnet-ef database update
- **dotnet-ef migrations add AddTableCategory**
- **dotnet-ef database update**

DataContext.cs

```
using Microsoft.EntityFrameworkCore;

namespace ProductManger.Models
{
    public class DataContext : DbContext
    {
        public DataContext(DbContextOptions<DataContext> options) : base(options) { }
        public DbSet<Product> Products { get; set; }
        public DbSet<Category> Category { get; set; }

        protected override void OnModelCreating(ModelBuilder modelBuilder)
        {
            modelBuilder.Entity<Product>()
                .HasOne(p => p.Category)
                .WithMany(c => c.Products)
                .HasForeignKey(p => p.CategoryId);

            base.OnModelCreating(modelBuilder);
        }
    }
}
```

Category.cs

```
using System.ComponentModel.DataAnnotations;

namespace ProductManger.Models
{
    public class Category
    {
        [Key]
        public int Id { get; set; }

        [MaxLength(256)]
        public string Name { get; set; }

        public List<Product> Products { get; set; }
    }
}
```

Product.cs

```
using System.ComponentModel.DataAnnotations;

namespace ProductManger.Models
{
    public class Product
    {
        [Key]
        public int Id { get; set; }

        [MaxLength(256)]
        public string Name { get; set; }

        public int Price { get; set; }

        public int Quantity { get; set; }

        [MaxLength(256)]
        public string Slug { get; set; }

        public int CategoryId { get; set; }

        public Category Category { get; set; }
    }
}
```

appsettings.json

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "Default": "Server=localhost;Database=ProductDb14;Trusted_Connection=True;"
  }
}
```

MySQL

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "Default": "server=localhost;database=ProductDb14;user=user;password=password;"
  }
}
```

Program.cs

```
using Microsoft.EntityFrameworkCore;
using ProductManger.Models;

var builder = WebApplication.CreateBuilder(args);
var services = builder.Services;
var connectionString = builder.Configuration.GetConnectionString("Default");

// Add services to the container.
services.AddControllersWithViews();
services.AddDbContext<DataContext>(options =>
    options.UseSqlServer(connectionString));
services.AddTransient<IProductService, ProductService>();

var app = builder.Build();
```

MySQL

```
using Microsoft.EntityFrameworkCore;
using ProductManger.Models;

var builder = WebApplication.CreateBuilder(args);
var services = builder.Services;
var connectionString = builder.Configuration.GetConnectionString("Default");

// Add services to the container.
services.AddControllersWithViews();
var serverVersion = new MySqlServerVersion(new Version(8, 0, 29));
services.AddDbContext<DataContext>(
    dbContextOptions => dbContextOptions
        .UseMySql(connectionString, serverVersion)
        .LogTo(Console.WriteLine, LogLevel.Information)
        .EnableSensitiveDataLogging()
        .EnableDetailedErrors()
);

var app = builder.Build();
```

IProductService.cs

```
using ProductManger.Models;

namespace ProductManger.Services
{
    public interface IProductService
    {
        List<Product> GetProducts();
    }
}
```

ProductService.cs

```
using ProductManger.Models;

namespace ProductManger.Services
{
    public class ProductService : IProductService
    {
        private readonly DataContext _context;
        public ProductService(DataContext context)
        {
            _context = context;
        }

        public List<Product> GetProducts()
        {
            return _context.Products.ToList();
        }
    }
}
```

ProductController.cs

```
using Microsoft.AspNetCore.Mvc;
using ProductManger.Services;

namespace ProductManger.Controllers
{
    public class ProductController : Controller
    {
        private readonly IProductService _productService;
        public ProductController(IProductService productService)
        {
            _productService = productService;
        }

        public IActionResult Index()
        {
            var products = _productService.GetProducts();
            return View(products);
        }
    }
}
```