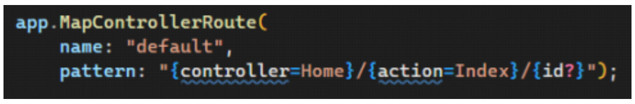
asp.net

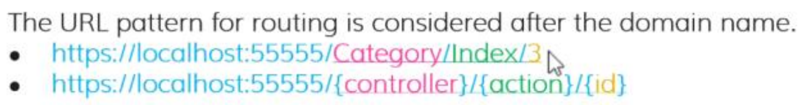
16 de janeiro de 2024 22:43

static -> goes into wwwroot folder
appsetting -> place for the secret keys (connectionStrings)
launchsettings -> environment variables, port numbers, protocol used, http, https

arquitetura utilizada: MVC (model-view-controller)

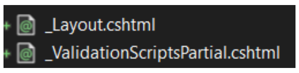


```
app.MapControllerRoute(
    name: "default",
    pattern: "{controller=Home}/{action=Index}/{id?}");
```



The URL pattern for routing is considered after the domain name.
- https://localhost:55555/Category/Index/3
- https://localhost:55555/{controller}/{action}/{id}

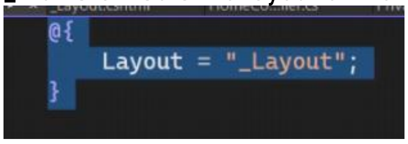
views -> shared -> "master page" (render body -- built-in helper)



_Layout.cshtml
_ValidationScriptsPartial.cshtml

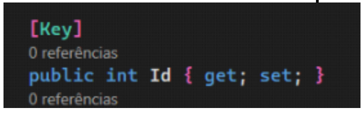
"_" ⇒ means it's used throughout the aplication

_ViewStart sets the layout of the master page



```
@{
    Layout = "_Layout";
}
```

data annotation : Id is the primary key



```
[Key]
0 referências
public int Id { get; set; }
0 referências
```

NuGet Packages
Microsoft.EntityFrameworkCore
Microsoft.EntityFrameworkCore.SqlServer
Microsoft.EntityFrameworkCore.Tools

```
<PackageReference Include="Microsoft.EntityFrameworkCore" Version="8.0.1" />
<PackageReference Include="Microsoft.EntityFrameworkCore.SqlServer" Version="8.0.1" />
<PackageReference Include="Microsoft.EntityFrameworkCore.Tools" Version="8.0.1">
```

Nova pasta "Data" com nova classe:

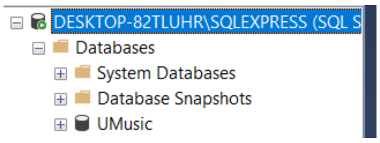
```
public class ApplicationDbContext : DbContext // extends DbContext
{
    public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options) : base(options) { }
}
```

Adicionar serviço ao container: (em program.cs)

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

Update database:

```
PM> update-database
Build started...
Build succeeded.
```



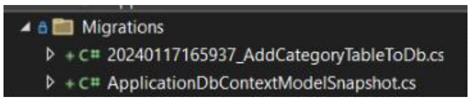
```
DESKTOP-82TLUHR\SQLEXPRESS (SQL S
  Databases
    System Databases
    Database Snapshots
    UMusic
```

Database criada

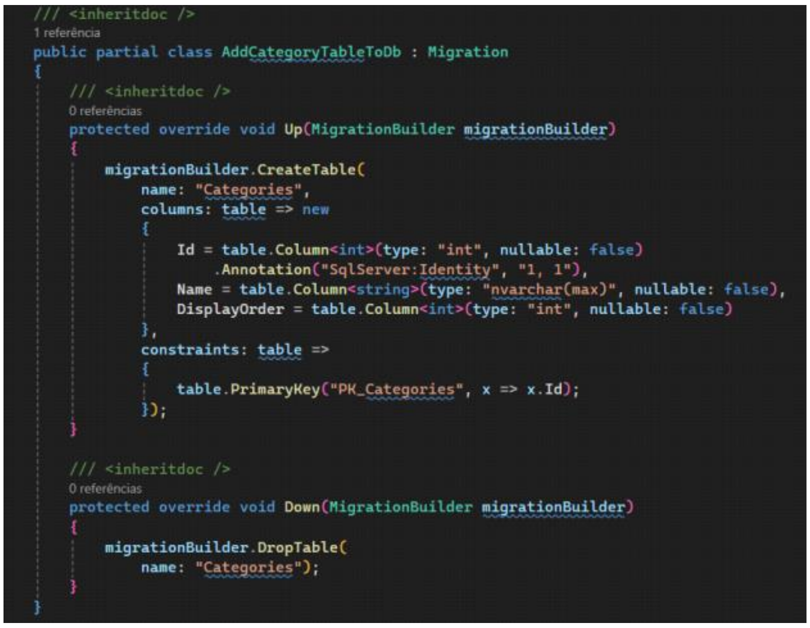
Adicionar tabela (neste caso, a classe Category)

execute command:
PM> add-migration AddCategoryTableToDb

new file created: (automatically) (lesgooo, o DAO é feito automaticamente)

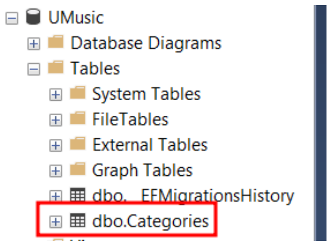


```
Migrations
  C# 20240117165937_AddCategoryTableToDb.cs
  C# ApplicationDbContextModelSnapshot.cs
```



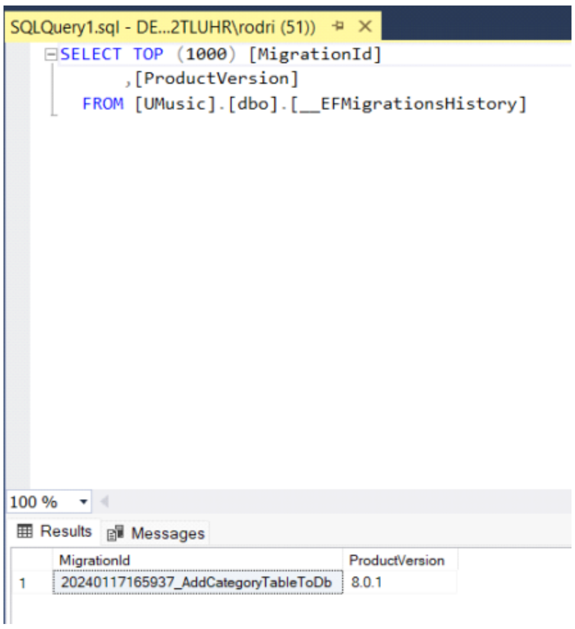
```
/// <inheritdoc />
1 referência
public partial class AddCategoryTableToDb : Migration
{
    /// <inheritdoc />
    0 referências
    protected override void Up(MigrationBuilder migrationBuilder)
    {
        migrationBuilder.CreateTable(
            name: "Categories",
            columns: table => new
            {
                Id = table.Column<int>(type: "int", nullable: false)
                    .Annotation("SqlServer:Identity", "1, 1"),
                Name = table.Column<string>(type: "nvarchar(max)", nullable: false),
                DisplayOrder = table.Column<int>(type: "int", nullable: false)
            },
            constraints: table =>
            {
                table.PrimaryKey("PK_Categories", x => x.Id);
            });
    }

    /// <inheritdoc />
    0 referências
    protected override void Down(MigrationBuilder migrationBuilder)
    {
        migrationBuilder.DropTable(
            name: "Categories");
    }
}
```

execute command:
update-database



```
UMusic
  Database Diagrams
  Tables
    System Tables
    FileTables
    External Tables
    Graph Tables
    dbo.__EFMigrationsHistory
    dbo.Categories
```

migration history:

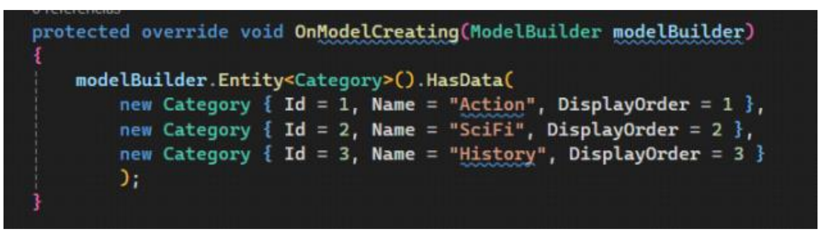

SQLQuery1.sql - DE...2TLUHR\rodri (51))

```
SELECT TOP (1000) [MigrationId]
      ,[ProductVersion]
  FROM [UMusic].[dbo].[__EFMigrationsHistory]
```

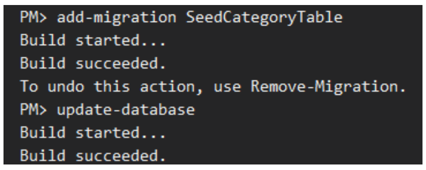
100 %

Results | Messages

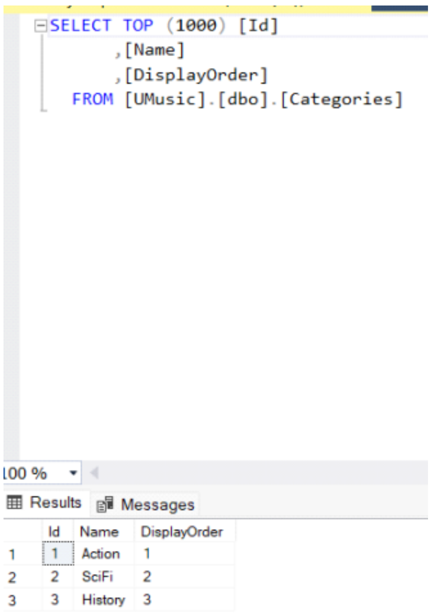
	MigrationId	ProductVersion
1	20240117165937_AddCategoryTableToDb	8.0.1



```
protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    modelBuilder.Entity<Category>().HasData(
        new Category { Id = 1, Name = "Action", DisplayOrder = 1 },
        new Category { Id = 2, Name = "SciFi", DisplayOrder = 2 },
        new Category { Id = 3, Name = "History", DisplayOrder = 3 }
        );
}
```



```
PM> add-migration SeedCategoryTable
Build started...
Build succeeded.
To undo this action, use Remove-Migration.
PM> update-database
Build started...
Build succeeded.
```

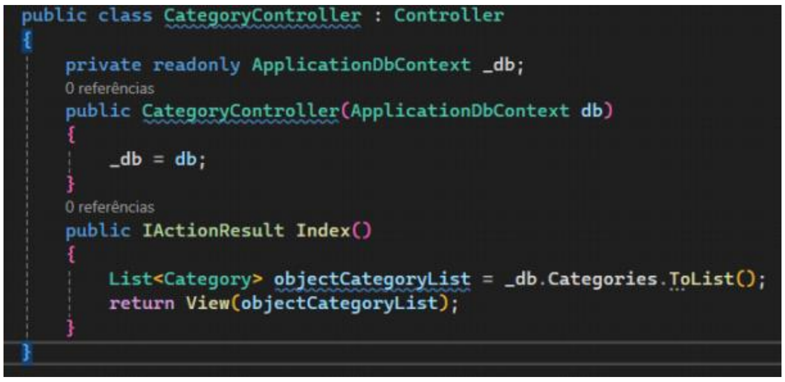


```
SELECT TOP (1000) [Id]
      ,[Name]
      ,[DisplayOrder]
  FROM [UMusic].[dbo].[Categories]
```

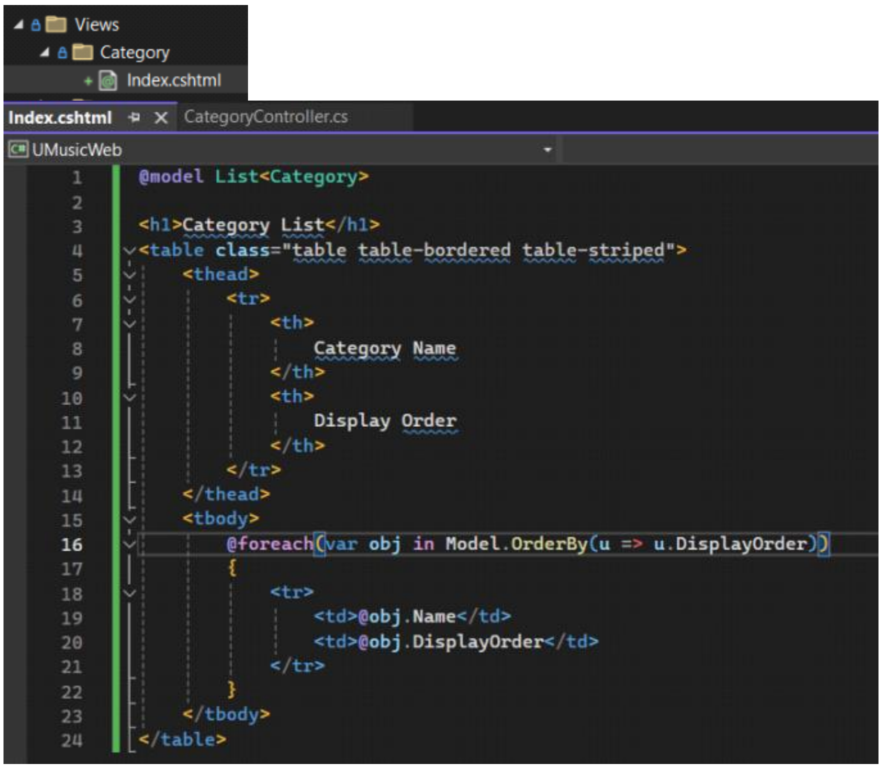
100 %

Results | Messages

	Id	Name	DisplayOrder
1	1	Action	1
2	2	SciFi	2
3	3	History	3



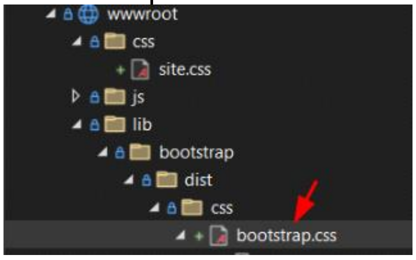
```
public class CategoryController : Controller
{
    private readonly ApplicationDbContext _db;
    0 referências
    public CategoryController(ApplicationDbContext db)
    {
        _db = db;
    }
    0 referências
    public IActionResult Index()
    {
        List<Category> objectCategoryList = _db.Categories.ToList();
        return View(objectCategoryList);
    }
}
```



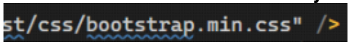
```
@model List<Category>

<h1>Category List</h1>
<table class="table table-bordered table-striped">
    <thead>
        <tr>
            <th>
                Category Name
            </th>
            <th>
                Display Order
            </th>
        </tr>
    </thead>
    <tbody>
        @foreach(var obj in Model.OrderBy(u => u.DisplayOrder))
        {
            <tr>
                <td>@obj.Name</td>
                <td>@obj.DisplayOrder</td>
            </tr>
        }
    </tbody>
</table>
```

https://bootswatch.com/ --- free themes for bootstrap

substituir aqui:



retirar .min no css do _Layout

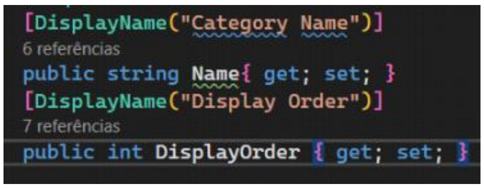


```
st/css/bootstrap.min.css" />
```

+ mudar outras coisas no _Layout (por exemplo de light para dark, etc.)

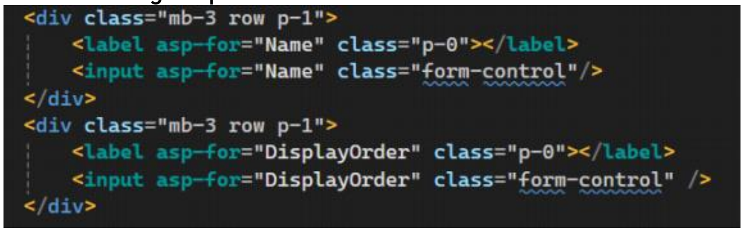
https://icons.getbootstrap.com/ --- icons
adicionar cdn:
<linkrel="stylesheet"href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css">
De < https://icons.getbootstrap.com/#install>
ao _Layout

data anotations



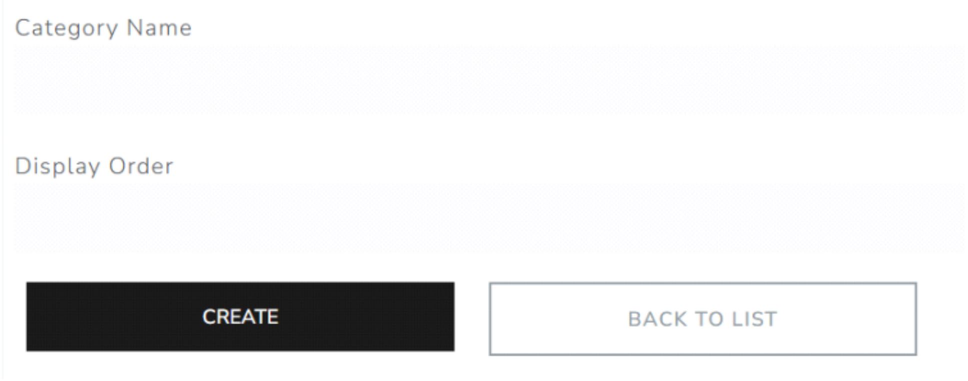
```
[DisplayName("Category Name")]
6 referências
public string Name{ get; set; }
[DisplayName("Display Order")]
7 referências
public int DisplayOrder { get; set; }
```

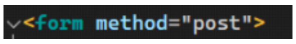
⇒ utilizar tag helpers



```
<div class="mb-3 row p-1">
    <label asp-for="Name" class="p-0"></label>
    <input asp-for="Name" class="form-control"/>
</div>
<div class="mb-3 row p-1">
    <label asp-for="DisplayOrder" class="p-0"></label>
    <input asp-for="DisplayOrder" class="form-control" />
</div>
```

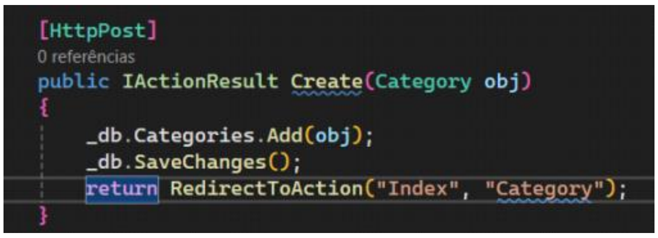
Result:





```
<form method="post">
```

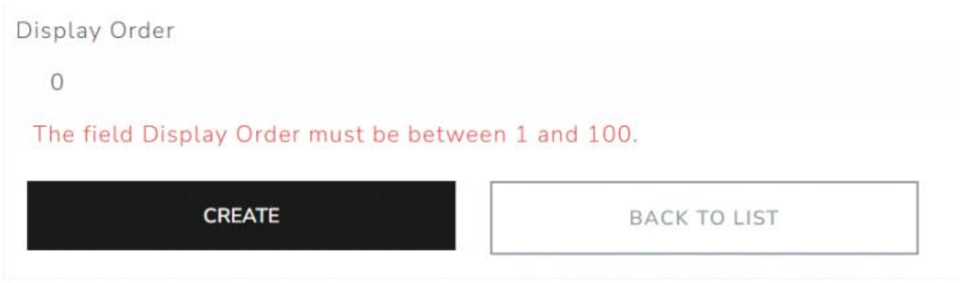
⇒



```
[HttpPost]
0 referências
public IActionResult Create(Category obj)
{
    _db.Categories.Add(obj);
    _db.SaveChanges();
    return RedirectToAction("Index", "Category");
}
```

No sql queries needed

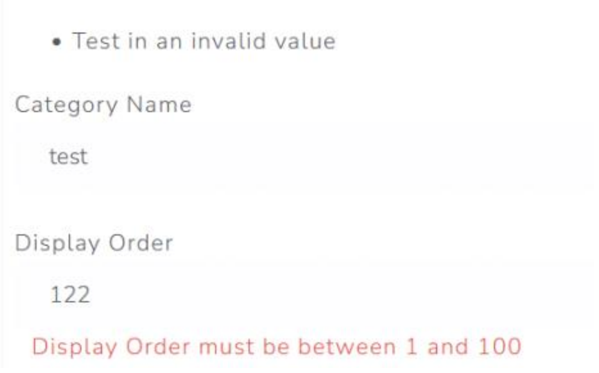
é necessário adicionar validação de dados (por exemplo display order > 0)
sinalizar erro:
<span asp-validation-for="Name" class="text-danger"></span>
(mto facil com tag helpers)



lesgoooo

<div asp-validation-summary="ModelOnly"></div>
if (obj.Name.ToLower() == "test")
{
    ModelState.AddModelError("", "Test in an invalid value");
}
          ^ not attached to an atribute

result:



- Test in an invalid value

Category Name

test

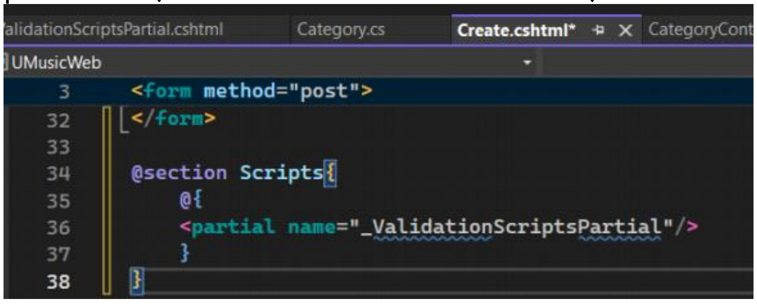
Display Order

122

Display Order must be between 1 and 100

no validation summary aparecem apenas os erros que não estejam ligados aos atributes

partial view (default location is the shared folder)



```
<form method="post">
</form>

@section Scripts{
    @{
    <partial name="_ValidationScriptsPartial"/>
    }
}
```

with this the validation is first client side (mais eficiente, para não ter de comunicar sempre com o servidor) para além disso pode dar display automatico de mensagens de erro sem se ter de clicar no botao de submit

edit
<a asp-controller="Category" asp-action="Edit" asp-route-id="@obj.Id" class="btn btn-secondary mx-2">
    <i class="bi bi-pencil-square"></i> Edit
</a>

no controller:

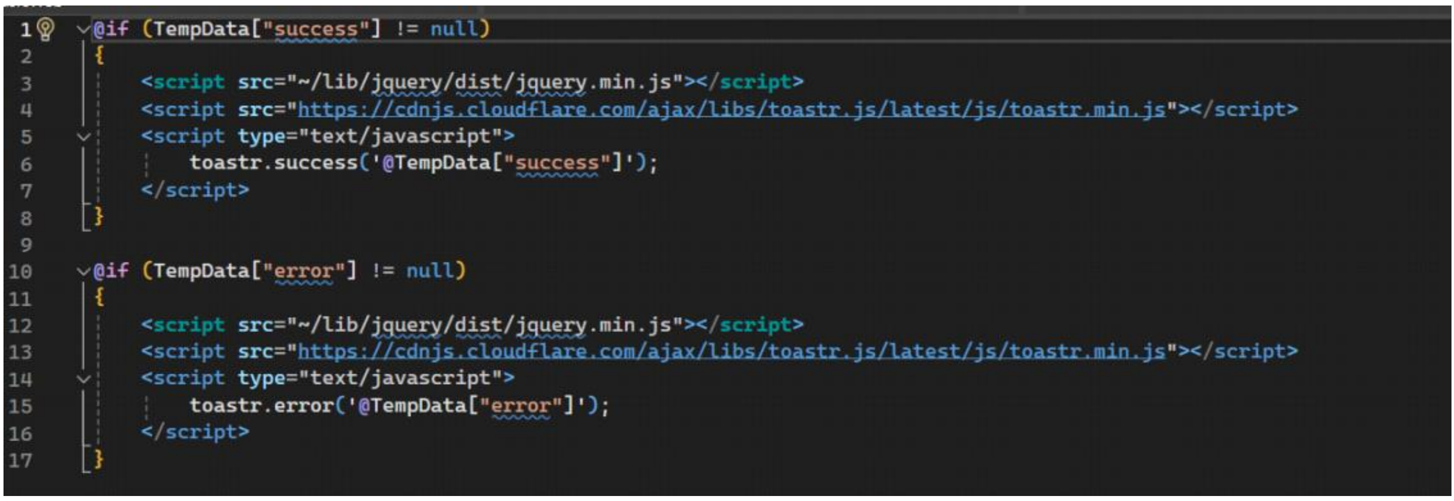
```
public IActionResult Edit(int? id)
{
    if (id == null || id == 0)
    {
        return NotFound();
    }

    Category? category = _db.Categories.Find(id);

    //Category? c2 = _db.Categories.FirstOrDefault(u => u.Id == id);
    //Category? c3 = _db.Categories.Where(u => u.Id == id).FirstOrDefault();

    if (category == null)
    {
        return NotFound();
    }

    return View(category);
}

[HttpPost]
public IActionResult Edit(Category obj)
{
    if (ModelState.IsValid)
    {
        _db.Categories.Update(obj);
        _db.SaveChanges();
        return RedirectToAction("Index", "Category");
    }
    return View();
}
```

(parecido para o delete)

notificaçao com animaçao para as operaçoes
https://codeseven.github.io/toastr/

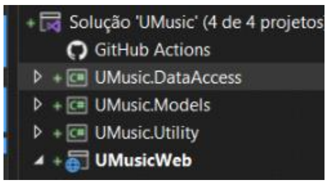
adicionar ao _Layout
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/css/toastr.min.css" />

adicionar à main do _Layout
<partial name="_Notification" />



```
@if (TempData["success"] != null)
{
    <script src="~/lib/jquery/dist/jquery.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/js/toastr.min.js"></script>
    <script type="text/javascript">
        toastr.success('@TempData["success"]');
    </script>
}

@if (TempData["error"] != null)
{
    <script src="~/lib/jquery/dist/jquery.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/toastr.js/latest/js/toastr.min.js"></script>
    <script type="text/javascript">
        toastr.error('@TempData["error"]');
    </script>
}
```

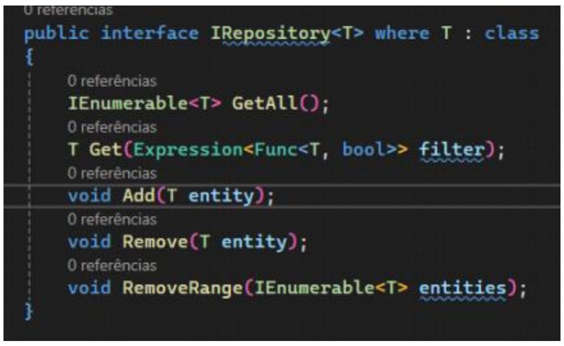
criar class libraries (afinal n fiz isto, dá alguns erros, preguiça de resolver, fica tudo em pastas msm)



```
Solução 'UMusic' (4 de 4 projetos
    GitHub Actions
  UMusic.DataAccess
  UMusic.Models
  UMusic.Utility
  UMusicWeb
```

(dependency injection lifetimes
- transient
- scoped
- singleton)

nova interface

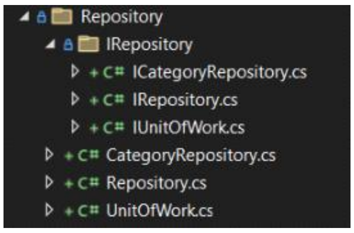


```
0 referências
public interface IRepository<T> where T : class
{
    0 referências
    IEnumerable<T> GetAll();
    0 referências
    T Get(Expression<Func<T, bool>> filter);
    0 referências
    void Add(T entity);
    0 referências
    void Remove(T entity);
    0 referências
    void RemoveRange(IEnumerable<T> entities);
}
```

public class Repository<T> : IRepository<T> where T : class
{
  private readonly ApplicationDbContext _db;
  internal DbSet<T> dbSet;
  public Repository(ApplicationDbContext db)
  {
    _db = db;
    this.dbSet = _db.Set<T>();
  }

Aqui _db.Set<T>() seria por exemplo igual a _db.Categories (caso T == Category)

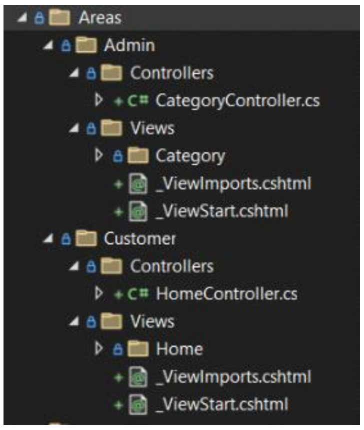
Adicionar UnitOfWork (with a Save method)



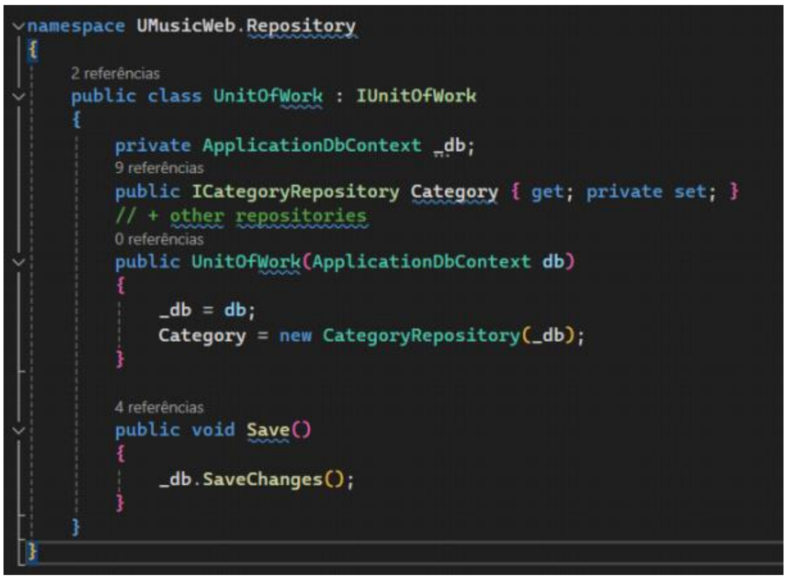
```
Repository
  IRepository
    ICategoryRepository.cs
    IRepository.cs
    IUnitOfWork.cs
  CategoryRepository.cs
  Repository.cs
  UnitOfWork.cs
```

builder.Services.AddScoped<IUnitOfWork, UnitOfWork>();

Adicionar áreas: Admin e Customer e mover os controllers e views para lá



```
Areas
  Admin
    Controllers
      CategoryController.cs
    Views
      Category
      _ViewImports.cshtml
      _ViewStart.cshtml
  Customer
    Controllers
      HomeController.cs
    Views
      Home
      _ViewImports.cshtml
      _ViewStart.cshtml
```

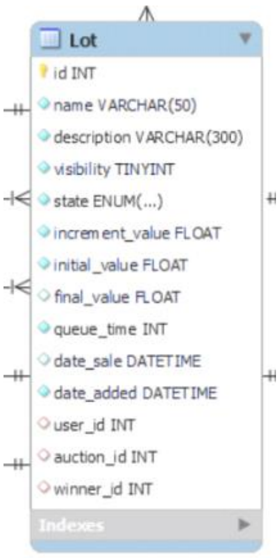


```
namespace UMusicWeb.Repository
{
    2 referências
    public class UnitOfWork : IUnitOfWork
    {
        private ApplicationDbContext _db;
        9 referências
        public ICategoryRepository Category { get; private set; }
        // + other repositories
        0 referências
        public UnitOfWork(ApplicationDbContext db)
        {
            _db = db;
            Category = new CategoryRepository(_db);
        }

        4 referências
        public void Save()
        {
            _db.SaveChanges();
        }
    }
}
```

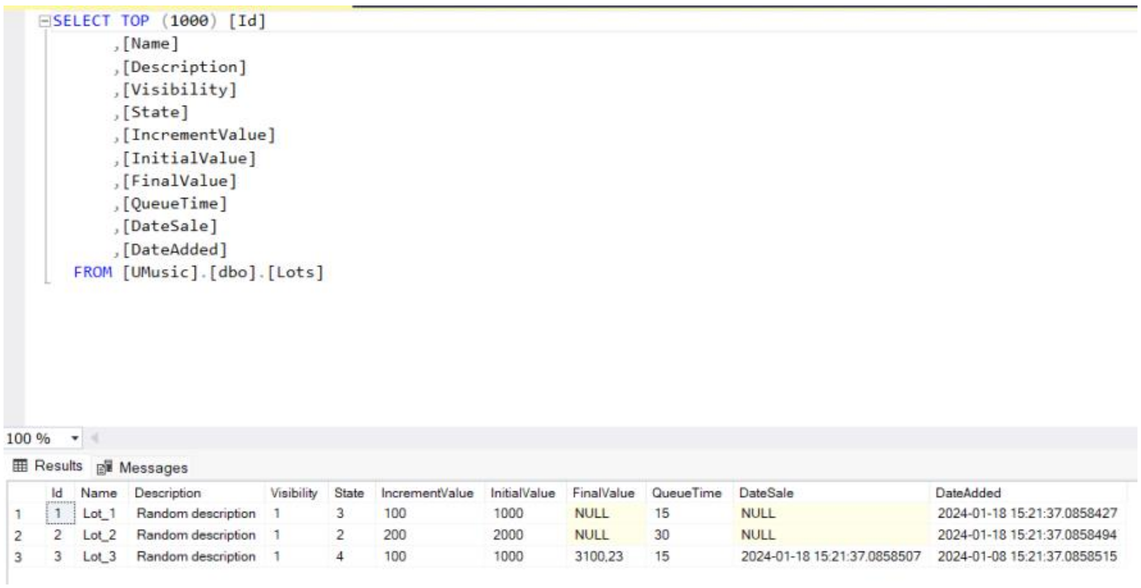
```
app.MapControllerRoute(
    name: "default",
    pattern: "{area=Customer}/{controller=Home}/{action=Index}/{id?}");
```

---

criação de novas entidades:
Lot (ainda sem chaves estrangeiras)



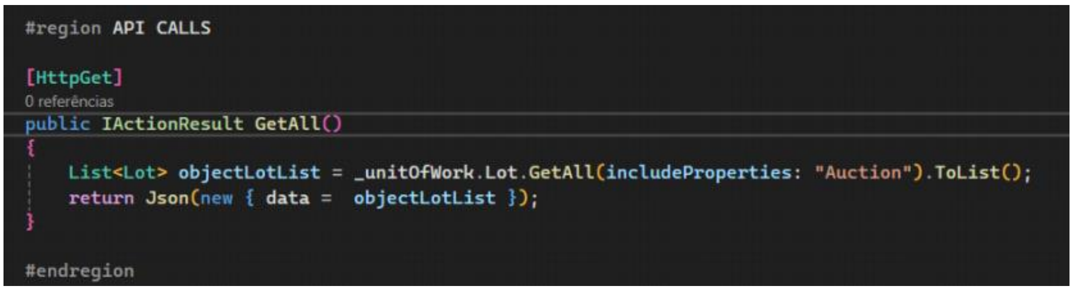
```
Lot
id INT
name VARCHAR(50)
description VARCHAR(300)
visibility TINYINT
state ENUM(...)
increment_value FLOAT
initial_value FLOAT
final_value FLOAT
queue_time INT
date_sale DATETIME
date_added DATETIME
user_id INT
auction_id INT
winner_id INT
```



```sql
SELECT TOP (1000) [Id]
      ,[Name]
      ,[Description]
      ,[Visibility]
      ,[State]
      ,[IncrementValue]
      ,[InitialValue]
      ,[FinalValue]
      ,[QueueTime]
      ,[DateSale]
      ,[DateAdded]
  FROM [UMusic].[dbo].[Lots]
```

	Id	Name	Description	Visibility	State	IncrementValue	InitialValue	FinalValue	QueueTime	DateSale	DateAdded
1	1	Lot_1	Random description	1	3	100	1000	NULL	15	NULL	2024-01-18 15:21:37.0858427
2	2	Lot_2	Random description	1	2	200	2000	NULL	30	NULL	2024-01-18 15:21:37.0858494
3	3	Lot_3	Random description	1	4	100	1000	3100,23	15	2024-01-18 15:21:37.0858507	2024-01-08 15:21:37.0858515

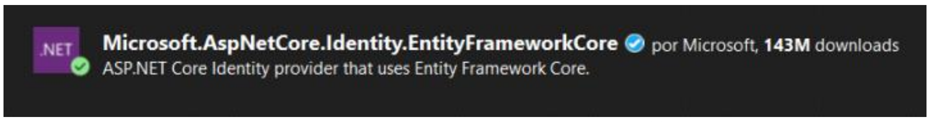
https://datatables.net/



```csharp
#region API CALLS

[HttpGet]
0 referências
public IActionResult GetAll()
{
    List<Lot> objectLotList = _unitOfWork.Lot.GetAll(includeProperties: "Auction").ToList();
    return Json(new { data =  objectLotList });
}

#endregion
```

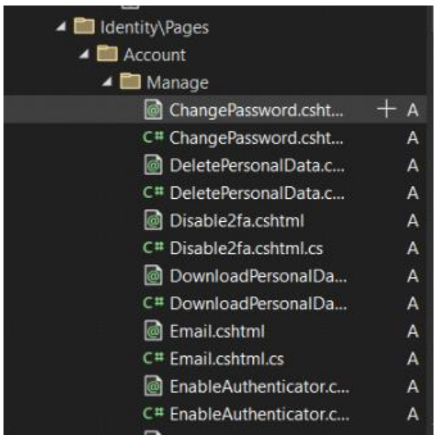
https://www.tiny.cloud/ --- para a textarea

- adicionar uma página de visualizaçao de produto
    - esta pagina terá um botao para adicionar o lote aos favoritos
- adicionar registo, login e gestao de permissoes



Microsoft.AspNetCore.Identity.EntityFrameworkCore por Microsoft, **143M** downloads
ASP.NET Core Identity provider that uses Entity Framework Core.

pages added automatically (razor pages (infelizmente))


Identity\Pages
Account
Manage
ChangePassword.csht... A
ChangePassword.csht... A
DeletePersonalData.c... A
DeletePersonalData.c... A
Disable2fa.cshtml A
Disable2fa.cshtml.cs A
DownloadPersonalDa... A
DownloadPersonalDa... A
Email.cshtml A
Email.cshtml.cs A
EnableAuthenticator.c... A
EnableAuthenticator.c... A