Laboratório 2 – Acesso a dados

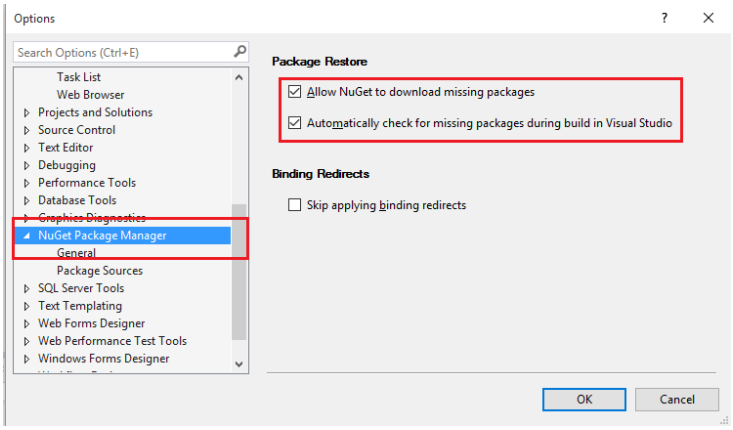
Neste laboratório é apresentada a criação de um repositório de dados utilizando a abordagem código primeiro (code first).

1 Criando um repositório de dados no modelo código primeiro (code first)

1.1 Abra o Visual Studio e crie um novo projeto. Na caixa de diálogo **New Project** selecione o template **Visual C# | Web** e a seguir selecione **ASP.NET Web Application.** Defina o nome do projeto como **MvcMovie**, selecione uma pasta de trabalho e tecle **OK**.

1.2 Na caixa de diálogo **New ASP.NET Project**, selecione (marque) o template **MVC**, desmarque as opções **Add unit tests** e **Host in the cloud.** Clique em **Change Authentication** e selecione a opção **No Authentication**. Tecle **OK** para criar o projeto.

1.3 Verifique (e habilite se for o caso) o ***NuGet*** para baixar automaticamente os pacotes necessários para poder compilar o projeto. Selecione a opção de menu **Tools** | **Options | Package Manager** e habilite a opção **Allow NuGet to download missing packages during build.** Tecle **OK** e faça um **Build** da solução **(**menu **Build | Build Solution).**



1.4 Execute e verifique que este *template* já possui algumas funcionalidades implementadas. Teste.

1.5 Inicialmente acrescente classes POCO para representar as entidades (classes de modelo). Selecione a pasta **Models** e com o botão direito selecione a opção **Add** | **Class**, adicione as classes **Movie.cs** e **Genre.cs.**

1.6 Para classe **Movie** utilize o código abaixo:

```
public class Movie
{
    public int ID { get; set; }
    public string Title { get; set; }
    public string Director { get; set; }
    public DateTime ReleaseDate { get; set; }
    public decimal Gross { get; set; }
    public double Rating { get; set; }

    public int GenreID { get; set; }
    public virtual Genre Genre { get; set; }
  }
```

Para a classe **Genre** use o código abaixo:

```
public class Genre
{
    public int GenreID { get; set; }
    public string Name { get; set; }
    public string Description { get; set; }

    public virtual ICollection<Movie> Movies { get; set; }
}
```

1.7 Analise o código destas classes, em especial as propriedades **ID**, as "chaves estrangeiras", e a navegação bidirecional entre as classes **Movie** e **Genre**. Isto não seria implementado em uma abordagem OO, mas permite ao EF o mapeamento automático e otimização das consultas.

1.8 Para realizar a persistência dos dados instale no seu projeto o **Entity Framework** (EF). Selecione a opção de menu **Tools** | **Package Manager** | **Package Manager Console** (esta operação também é possível realizar utilizando a interface visual do Nuget: **Manage Nuget Packages for Solution)**. Na console digite o seguinte comando**:**

```
Install-Package EntityFramework
```

1.9 Adicione um contexto do Entity Framework ao projeto para fazer a ligação entre os objetos da aplicação e o banco de dados. Na pasta **Models** adicione uma nova classe de nome **MovieDBContext.cs** e acrescente o código abaixo (será necessário acrescentar o namespace `System.Data.Entity` ao código):

```
public class MovieDBContext : DbContext
{
    public MovieDBContext() : base("MovieDbContext")
    {
    }

    public DbSet<Movie> Movies { get; set; }
    public DbSet<Genre> Genres { get; set; }
}
```

1.10 Adicione a seguinte string de conexão que define qual será o provedor de banco de dados e onde serão armazenados os dados da aplicação. Abra o aquivo **Web.config** e acrescente a tag abaixo imediatamente antes da tag `<appSettings>`**:**

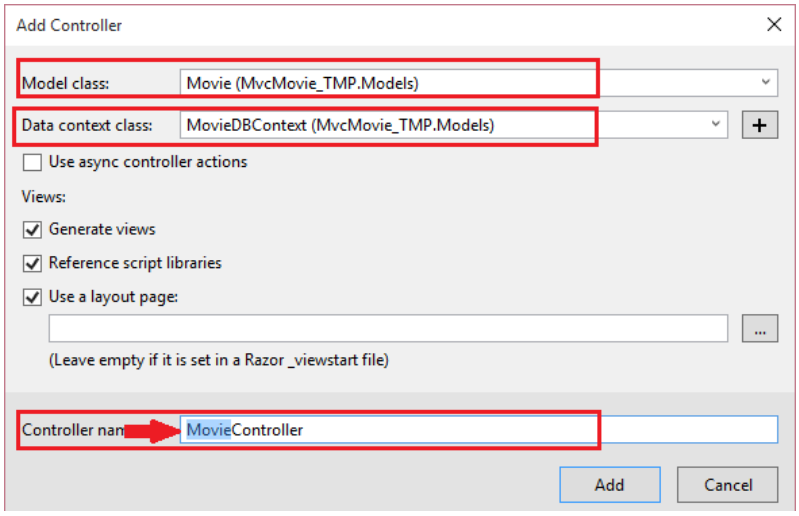
```
</configSections>
  <connectionStrings>
    <add name="MovieDbContext" connectionString="Data
Source=(LocalDb)\MSSQLLocalDB;initial
catalog=DbMovies;Integrated
Security=SSPI;AttachDBFilename=|DataDirectory|\DbMovies.mdf"
providerName="System.Data.SqlClient" />
  </connectionStrings> >
```

1.11 Selecione a pasta **Models** e com o botão direito selecione a opção **Add** | **Existing Item**, selecione o arquivo **MovieInitializer.cs** disponível na pasta **labs | recursos**. Analise cuidadosamente o código disponibilizado neste arquivo.

1.12 Abra o arquivo Global.asax e acrescente ao final do método **Application_Start()** o código abaixo para ativar a inicialização do banco de dados (será necessário acrescentar os namespaces `System.Data.Entity` e `MvcMovie.Models` ao código).

```
Database.SetInitializer(new MovieInitializer());
```

1.13 Para poder testar o modelo, é necessário fornecer ao sistema classes controladoras e views. Por hora vamos utilizar o sistema de Scaffolding disponibilizado pelo MVC. Selecione a pasta **Controllers** e com o botão direito selecione a opção **Add | Controller**. Selecione a opção **MVC 5 Controller with views Framework** Adicione um controlador com o nome MusicControler e configure as propriedades conforme a figura abaixo:



1.14 Olhe “rapidamente” o código do controlador e das *views* que foram criadas (na pasta **Views | Movie**).

1.15 Teste a aplicação. Execute o projeto e digite na *URL* do navegador o endereço:

```
http://localhost:<porta>/Store
```

teste as opções para criar, editar, apagar e ver detalhes disponibilizadas.

1.16 Altere a class de modelo **Movie.cs** adicionado ao modelo anotações de apresentação e validação:

- Título e diretor são obrigatórios e devem ter um comprimento entre 5 e 40 caracteres
- Ranking deve ser um valor entre 0 e 10
- Apresentar faturamento (Goss) em formato monetário, com separador de milhar.
- Data de lançamento deve ser no formato brasileiro (dia/mês/ano) e usado, também, para entrada/atualização de dados

1.17 Volte a testar e veja o efeito nas operações.