Exercício de Laboratório

1. Inicie o Visual Studio;
2. Inicie um novo projeto, selecione “Other Project Types”, “Visual Studio Solutions”, “Blank Solution”;
3. Nomeie seu projeto como “com.sakila” e clique OK;
4. No ambiente do Visual Studio, na árvore do projeto, clique com o botão direito na “Solution” e inclua mais três projetos:

   a. com.sakila.web (tipo C# Asp.Net WebAPi, lembre de desconectar do azure);
   b. com.sakila.negocio (tipo class library);
   c. com.sakila.entidade (tipo class library);

5. Clique com o botão direito na camada web, se não estiver em negrito e escolha “Set as StartUp Project”;
6. Clique com o botão direito na camada entidade, escolha “Manage NuGet Package”;
7. Do lado esquerdo da tela escolha a opção “OnLine”, e no campo de pesquisa procure por “MySQL”;
8. Selecione, o pacote “MySQL.Data.Entity” e adicione-o ao seu projeto utilizando o botão “Install”;
9. Marque “I Accept” na licença de uso;
10. Após instalar os pacotes, escolha “Close”;
11. Clique com o botão direito na camada Entidade, escolha “New”, e “Class” e crie uma classe chamada “Actor”;

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace com.sakila.entidade
{
    public class Actor
    {
        public int ID { get; set; }
        public string FirstName { get; set; }
        public string LastName { get; set; }
        public DateTime LastUpdate { get; set; }
    }
}
```

12. A estrutura da Classe é a que segue, análoga à tabela:

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

13. Inclua nas cláusulas “Using” da classe “Actor” os seguintes pacotes que servirão para configurar o mapeamento com as tabelas:

```
{
    [Table("actor", Schema = "sakila")]
    public class Actor
```

```
{
```

14. Após incluir os pacotes, antes do nome da classe, vamos indicar o nome da tabela no banco onde os dados serão salvos, inclua a linha abaixo:

```
[Key]
[DatabaseGenerated(DatabaseGeneratedOption.Identity)]
[Column("actor_id")]
 public int ID { get; set; }
```

15. Para o campo ID, vamos configurar a chave primária da tabela:

```
[Column("first_name")]
public string FirstName { get; set; }
```

16. Agora, antes de cada propriedade da classe, vamos indicar o campo da tabela relacionado:
17. Abra o MySQL Workbench, abra a instância atual do MySQL, e essa instância, procure pelo “Schema sakila”. Nas tabelas do “schema”, procure a tabela “Actor” e mapeie os demais campos da tabela com a classe;
18. Faça o mapeamento da tabela “Film” e “Language”;
19. Agora crie uma classe, na camada de Entidade chamada “SakilaDbContext”;
20. Ponha nessa classe o seguinte código:

```
using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Diagnostics;

namespace com.sakila.entidade
{
    public class SakilaDbContext : DbContext
    {
        public SakilaDbContext() : base("SkDbContext")
        {
            Database.SetInitializer<SakilaDbContext>(null);
            // Database.Log = s => Debug.Write(s);
        }
        public DbSet<Actor> Actor { get; set; }
    }
}
```

21. Abra, na camada Web o arquivo “Web.Config” e procure por “ConnectionStrings”, no bloco “ConnectionStrings”, inclua a seguinte “string” de conexão que será usada pelo nosso “dbContext”;

```
<add name="SkDbContext" connectionString="server=127.0.0.1;database=sakila;user id=root;password=1234;pooling=false;" providerName="MySql.Data.MySqlClient" />
```

Lembre-se que deverá colocar sua senha de acesso ao MySQL em password e o usuário em user id

22. Use o “NuGet” para fazer referência ao “MySQL.Data” e ao “MySQL.Data.Entity” na camada WEB;
23. Agora, iremos criar uma página com um controlador que acesse nossa classe de entidades: primeiro, na camada Web faça referência à camada de entidades clicando com o botão direito na camada Web, escolhendo “Add”, “Reference” e na janela de referências escolhendo “Projects” que está dentro do grupo “Solution”. Marque a camada de entidades e dê ok;
24. Clique com o botão direito na pasta “Views”, e escolha “Add” e “New Folder”. Chame a pasta de “Actor”;
25. Clique com o botão direito na pasta “Actor”, escolha “Add” e depois “View”. Chame sua nova “View” de “Index” e clique “OK”;
26. Clique com o botão direito na pasta “controller” e selecione “Add” e “Controller”. Escolha um “controller MVC5 Empty”;
27. Abra o arquivo “_Layout.cshtml” que está em “View / Shared” e no menu inclua a seguinte opção:

    *
@Html.ActionLink("Actor", "Index", "Actor", new { area = "" }, null)
*

    O primeiro Produto é o texto do link e o segundo o nome do controller.

28. Teste se a página está abrindo;
29. Vamos testar se estamos acessando o banco de dados. Inicialmente inclua as seguintes linhas no método “index” do “ActorController”:

    SakilaDbContext db = new SakilaDbContext();
    var actors = db.Actor.ToList();

    Se a página abrir, basicamente acessamos o banco, senão, veremos o problema. Você pode colocar um breakpoint pra ver se os dados foram carregados.

30. Vamos retornar à nossa visão “Index.cshtml” que está na pasta “Actor”, nas “Views” e incluir o código para listar esses atores em uma tabela;

```
@using com.sakila.entidade;
@model IEnumerable<Actor>
@{
        ViewBag.Title = "Index";
}
<h2>List of Actor's</h2>
<div class="row">
        @Html.ActionLink("Novo","Create","Actor", new { area = ""},null)
         <table class="table table-striped table-hover dataTable">
        <thead>
                <tr class="heading">
                        <th>ID</th>
                        <th>First Name</th>
                        <th>Last Name</th>
                        <th>Last Update</th>
                </tr>
        </thead>
        <tfoot>
```

```
<tr>
        <td></td>
        <td></td>
        <td></td>
        <td></td>
</tr>
</tfoot>
<tbody>
@if(TempData["error"] != null)
{
<tr>
        <td colspan="4" class="text-center">
        <span class="label label-danger">@TempData["error"]</span>
        </td>
</tr>
}
@foreach(var dados in this.Model)
{
        if(dados != null)
        {
        <tr>
                <td>@dados.ID</td>
                <td>@dados.FirstName</td>
                <td>@dados.LastName</td>
                <td>@dados.LastUpdate</td>
        </tr>
        }
}
</tbody>
</table>
</div> <!--/.row-->
```

31. Se você executar o código agora, verá um erro dizendo que o objeto não foi instanciado, isso se dá porque nosso model está vazio. Precisamos passar os dados para a requisição. Vá ao “Actor” “controller” e faça a seguinte mudança na linha final:

```
public ActionResult Index()
{
    var actors = db.Actor.ToList();
    return View(actors);
}
```

Consideração Final

Talvez não seja tão interessante acessar da “view” a classe das entidades para usar como “Model”, apesar de possível. Podemos passar essa responsabilidade para o pacote “Model”.