

Curso .NET

Unidade Curricular: Laboratório Web

Docente: David Jardim

FICHA DE TRABALHO 13

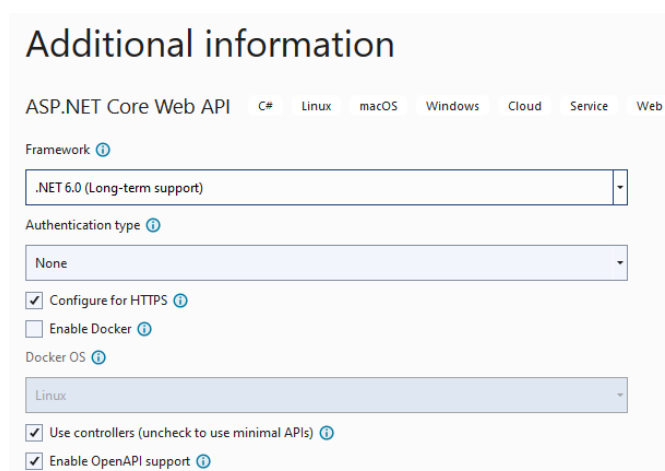
Exercícios:

1. MySQLWorkBench

- Faça download da BD Sakila no seguinte url: <https://downloads.mysql.com/docs/sakila-db.zip>
- Extraia o .zip para uma pasta
- No MySQLWorkBench instale o schema da BD a partir do ficheiro **sakila-schema.sql**
- No MySQLWorkBench insira os dados na BD a partir do ficheiro **sakila-data.sql**
- Confirme que as tabelas foram criadas e os dados inseridos

2. Projeto (*API with controllers*)

- Crie um projeto do tipo *Web API* denominado por Ficha13 com as seguintes definições:



Additional information

ASP.NET Core Web API C# Linux macOS Windows Cloud Service Web

Framework ⓘ

.NET 6.0 (Long-term support)

Authentication type ⓘ

None

☒ Configure for HTTPS ⓘ

☐ Enable Docker ⓘ

Docker OS ⓘ

Linux

☒ Use controllers (uncheck to use minimal APIs) ⓘ

☒ Enable OpenAPI support ⓘ

3. Pacotes

- Instale a **versão 5** dos seguintes pacotes utilizando o Nuget Packet manager
 - Microsoft.EntityFrameworkCore
 - Microsoft.EntityFrameworkCore.Tools
 - MySQL.EntityFrameworkCore

4. Scaffolding

- Abra o Package Manager Console
 - Execute o comando necessário para efetuar o scaffolding da BD Sakila
 - Confirme que o DbContext e as classes de domínio foram criadas na pasta Models
- Atualize os pacotes instalados anteriormente para a última versão (6)

5. Serviços

- a. Crie uma pasta denominada por *Services*
- b. Crie uma interface denominada por *IActorsService* e defina os seguintes métodos abstratos tendo em conta os seus parâmetros e tipo a devolver
 - i. *GetAll*
 - ii. *Create*
 - iii. *DeleteById*
 - iv. *GetById*
 - v. *Update*
 - vi. *GetFilms*
- c. Crie uma classe denominada por *ActorsService* que implementa a interface anterior
 - i. Implemente o construtor por parâmetros que receberá como parâmetro uma referência para a classe *SakilaContext*
 - ii. Implemente os métodos abstratos que utilizarão a referência para a classe *SakilaContext* de forma a realizar as operações de seleção e modificação na base de dados (consulte o material disponibilizado)

6. Controllers

- a. Tendo em conta a Tabela 1, na pasta *Controllers* crie um controller *ActorsController*
- b. Adicione um atributo privado do tipo *IActorsService*
- c. Implemente o construtor por parâmetros, onde o parâmetro é do tipo *IActorsService*
- d. Implemente os seguintes endpoints tendo em conta as valorações necessárias e respostas adequadas e utilize o serviço para :
 - i. Listar todos os atores existentes no ficheiro e devolver os mesmos na resposta
 - ii. Adicionar um novo ator, o ID do novo ator deve ser devolvido na resposta
 - iii. Apagar um ator pelo seu ID. O ID do ator removido deve ser devolvido na resposta
 - iv. Selecionar apenas um ator pelo seu ID e devolver esse mesmo ator na resposta
 - v. Alterar os detalhes de um determinado ator pelo seu ID e devolver esse mesmo ator atualizado na resposta
 - vi. Listar todos os filmes por ator

7. Program

- a. Aplicando *dependency injection* adicione à lista de serviços da aplicação um contexto para a base de dados do tipo *SakilaContext* (consulte o material disponibilizado)
- b. Aplicando *dependency injection* adicione à lista de serviços da aplicação um serviço com tempo de vida *scoped* para a base de dados do tipo *IActorsService* com implementação *ActorsService* (consulte o material disponibilizado)

URI	Método HTTP	Body do POST	Resultado
/actors	GET	empty	List all actors
/actors	POST	JSON String	Add details of a new actor
/actors/{id}	DELETE	empty	Delete an existing actor
/actors/{id}	GET	empty	Show details of an actor
/actors/{id}	PUT	JSON String	Update details of an actor
/actors/{id}/films	GET	empty	Show films from an actor

Tabela 1 - Métodos a implementar