

Processo Seletivo
Eduardo Heitor Soares Agostinho

Desafio Técnico DOTNET
TaskManagementAPI

1. Introdução
2. Configurando a ConnectionString com o SQL Server
3. Executando as Migrations
4. Utilizando o Swagger
5. Utilizando Endpoints Explorer do Visual Studio 2022
6. Efetuando testes de Integração xUnit

1. Introdução

O projeto foi desenvolvido utilizando o Visual Studio 2022 e visando criar uma API mais robusta e com uma implementação mais abrangente, foi utilizado o banco de dados SQL Server 2019 (RTM-GDR) (KB5040986) - 15.0.2116.2 (X64) para persistencia dos dados e autenticação JWT com Identity.

Para criação da estrutura das tabelas no banco de dados foi utilizado Migrations, portanto com passos simples o banco de dados será configurado.

Nos testes de integração, foi utilizado xUnit, além de ser possível efetuar testes pelo Swagger, Endpoint Explorer outra ferramentas como por exemplo o Postman.

Este documento, contém instruções de como configurar o banco de dados através das migrations, como utilizar a API e efetuar os testes pelo Swagger, Endpoint Explorer e os testes de integração com xUnit.

Espero ter atingido as expectativas em mim depositadas.

Desde já agradeço a oportunidade.

EDUARDO AGOSTINHO

2. Configurando a ConnectionString com o SQL Server

A primeira configuração a ser feita é a conexão com o banco de dados SQL Server, para isso na Solution Explorer do Visual Studio 2022 abra o arquivo *appsettings.json* e localize na linha 3 a chave *"DefaultConnection"* e altere o nome do *"server\\instância"* para a configuração do seu SQL Server.

```
"ConnectionStrings": {  
  "DefaultConnection": "Server=localhost\\SQLServer;"  
},
```

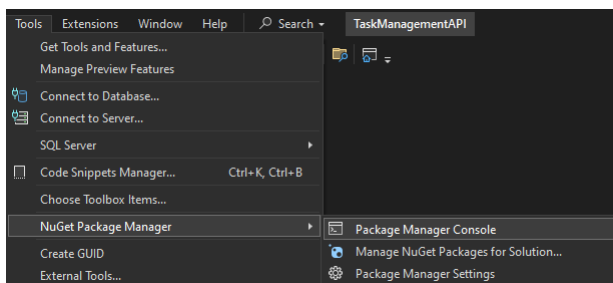
- **localhost**: Nome do host onde está instalado seu SQL Server;
- **SQLServer**: Nome da Instância do seu SQL Server;

Salve o arquivo *appsettings.json* e pode fechar.

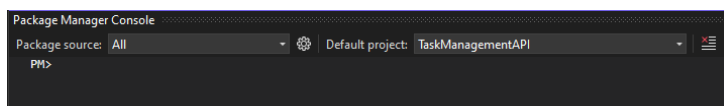
3. Executando a Migrations

O próximo passo é executar a Migrations (code first) para que as tabelas sejam criadas conforme o modelo do projeto. Siga os passos abaixo:

No Visual Studio 2022, clique no menu Tools => Nuget Package Manager => Package Manager Console (veja a imagem abaixo).



Na parte inferior da tela será aberto Console



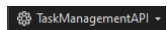
Digite os seguintes comandos:


```
add-migration 'Criacao' <ENTER>  
update-database <ENTER>
```

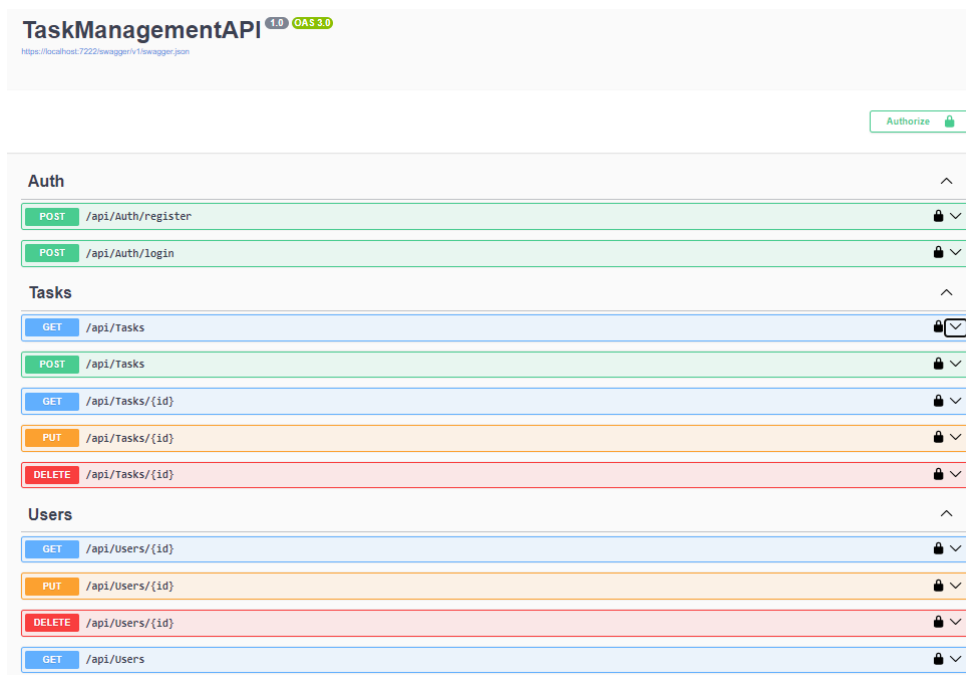
Neste momento o banco de dados TaskManagementDb e as tabelas serão criadas.

4. Utilizando o Swagger

Antes de executar a API, certifique de ter selecionado o projeto correto conforme a imagem





Clique em , para executar a API... Será aberto o navegador com a interface do Swagger.






Auth: Contém os endpoints para registrar e efetuar login do usuário;

Tasks: Contém os endpoints referente ao CRUD das Tasks;

Users: Contém os endpoints referente CRUD dos usuários, exceto o create que é feito através do endpoint register em Auth.

A primeira coisa a ser feita, é cadastrar um usuário através do endpoint `/api/Auth/register`. Clique em  `/api/Auth/register` e em seguida em , insira os dados solicitados no formato JSON e clique no botão EXECUTE. O status code de retorno deve ser 200 e a mensagem de usuário cadastrado com sucesso.

Agora para obter o token, devemos fazer login através do endpoint `/api/Auth/login`. Clique em  `/api/Auth/login` e em seguida em , insira os dados do usuário que foi cadastrado no passo anterior no formato JSON e clique no botão EXECUTE. O status code de retorno deve ser 200 e o token. **Copie o token que foi retornado.**

Clique em , será aberta a janela abaixo:

Available authorizations

Bearer (apiKey)

Token JWT

Name: Authorization

In: header

Value:

bearer eyJhbGciOiJIUzI1NiIs

Authorize

Close

Digite “*bearer*” e cole o token copiado no passo anterior; para finalizar clique em **Authorize**, feche a janela clicando em Close.

A partir deste momento você pode utilizar qualquer um dos endpoints de Tasks como:

Post /api/Tasks: Cria uma nova task;

Put /api/Tasks/{id}: Altera uma task existente pelo id;

Delete /api/Tasks/{id}: Exclui uma task existente pelo id;

Get /api/Tasks/{id}: Retorna a task pelo id;

Get /api/Tasks: Retorna todas as Tasks;

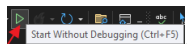
O mesmo critério é válido para Users.

6. Efetuando testes de integração xUnit

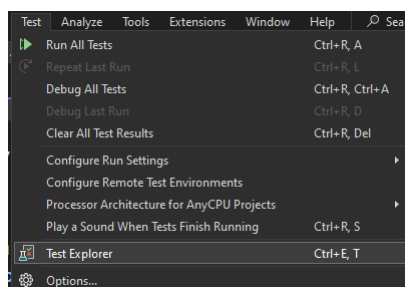
Por fim, o teste de integração com xUnit.

Este já possui alguns testes configurados basta executar através da interface Test Explorer do Visual Studio 2022.

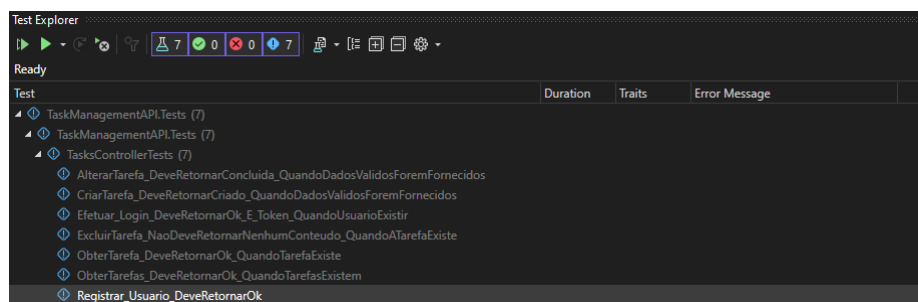
Para executar os testes, a API deve ser executada sem o modo debug clicando no botão



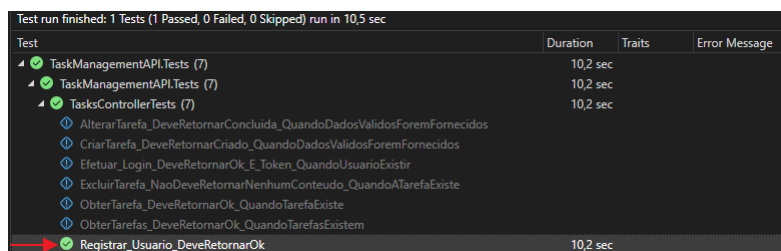
Após sua execução clique no menu Test => Test Explorer:



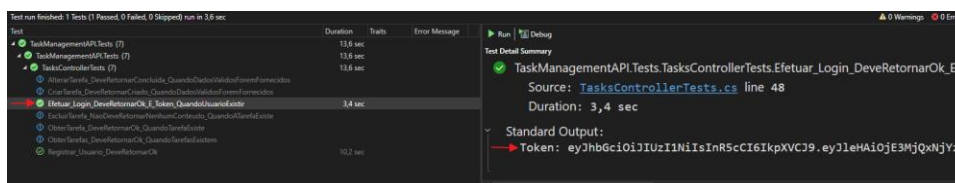
Será aberta a janela Test Explorer:



O primeiro teste que devemos executar é o teste Registrar_Usuario_DeveRetornarOk, para isso devemos informar no Arrange os dados desse novo usuário. Clique 2x em Registrar_Usuario_DeveRetornarOk e será exibido o teste, altere os dados do Arrange com os dados do usuário que será cadastrado, clique o botão direito sobre Registrar_Usuario_DeveRetornarOk e clique em “Run” para executar o teste. Veja na imagem abaixo que o teste passou com sucesso.



Agora devemos efetuar o login para obter o token que será utilizado nos testes das Tasks. Clique 2x em Efetuar_Login_DeveRetornarOk_E_Token_QuandoUsuarioExistir e altere os dados do do Arrange com os dados do usuário cadastrado no passo anterior, clique o botão direito sobre Efetuar_Login_DeveRetornarOk_E_Token_QuandoUsuarioExistir e clique em “Run” para executar o teste. Veja na imagem abaixo que o teste passou com sucesso e retornou o Token do lado direito. **Copie o valor retornado do token.**



Para finalizar, vamos passar o token atual para ser utilizado nos endpoints protegidos. Abra o arquivo TaskControllerTests.cs se o mesmo não estiver aberto e vá para a linha 14, e cole o token copiado no passo anterior na atribuição da variável Token.

```
14 private readonly string Token = "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOiE3MjQxNjYzNjYsImIzcyI6Imh0dHBzOi8vbG9j"
```

Agora para finalizar, podemos executar os outros testes que exigem autenticação:

- CriarTarefa_DeveRetornarCriado_QuandoDadosValidosForemFornecidos;
- AlterarTarefa_DeveRetornarConcluida_QuandoDadosValidosForemFornecidos;
- ObterTarefa_DeveRetornarOk_QuandoTarefaExiste;
- ObterTarefas_DeveRetornarOk_QuandoTarefasExistem;
- ExcluirTarefa_NaoDeveRetornarNenhumConteudo_QuandoATarefaExiste;