# Test technique d’intégration dans l’équipe MCO - 2022

## Introduction

Bonjour cher collègue !

J’ai essayé de faire un test technique qui se rapproche le plus possible de l’environnement de travail qu’on essaie d’améliorer au quotidien. Voici quelques difficultés que tu pourras rencontrer :

* Peu (voir pas du tout) de documentation
* Les sachants sur nos applicatifs ont quitté la société. Il faut donc tenter de comprendre le fonctionnement (les règles métiers) tout seul en lisant le code.
* Le legacy peut être très lourd. Il y a beaucoup de portions de code difficile à comprendre.

S’il manque des informations, des précisions dans l’énoncé du test, c’est voulu. Ce sont (malheureusement) les conditions dans lesquelles, l’équipe travaille. Bref, bon courage pour le test ! 😊

## Prérequis

Voici la configuration qui m’a permis de créer le test :

* Solution créée sous Visual Studio 2022 (v. 17.3.3) avec composant pour gérer une base de données  
  (dans le cadre du test, nous utiliserons un fichier .mdf)
* Projet Console App sur NET 6

Avec une version ultérieure de Visual Studio ou du framework .NET, le test doit être réalisable mais je n’ai pas essayé. Il est donc préférable d’avoir la même configuration.

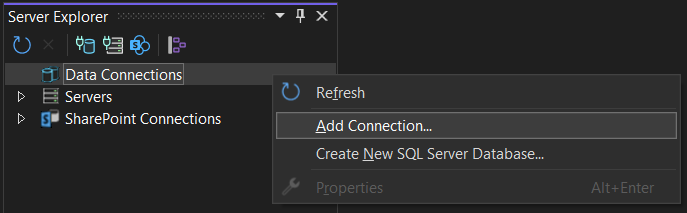
## Installation

Dans la solution, il y a un projet OfferExporter où il manque un fichier OFFERS.mdf

Une image contenant texte

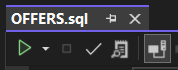
Description générée automatiquement

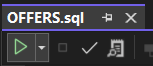
Avant de commencer, il faut créer le fichier OFFERS.mdf. Pour cela, on va le faire manuellement :

* Il faut ouvrir la vue Server Explorer (Menu View / Server Explorer) puis cliquer sur Add Connection…  
  
* Dans la fenêtre, renseigner les champs ci-dessous, puis cliquer sur OK.  
  > DataSource : Microsoft SQL Server Database File (SqlClient)  
  > Database file name (new or existing) : {local path on your computer}\OfferExporter\\_DB\OFFERS.mdf  
  Une image contenant texte

  Description générée automatiquement
* Une fenêtre de confirmation s’ouvrira, cliquer sur OK pour créer le fichier OFFERS.mdf
* A présent dans le dossier \_DB, il devrait y avoir les fichiers créés suivant :  
  Une image contenant texte

  Description générée automatiquement
* Maintenant, il faut ouvrir le fichier OFFERS.sql qui contient le script d’initialisation de la base :  
  Une image contenant texte, capture d’écran, écran, moniteur

  Description générée automatiquement
* Pour lancer le script, il faut se connecter à la base. Pour cela, cliquer sur le bouton de connexion en haut :  
  
* Dans la fenêtre de connexion, renseigner les champs suivant :  
  > Server Name : (localdb)\MSSQLLocalDB  
  > Authentication : Windows Authentication  
  > User Name : {your domain}\{your login}  
  > Database Name : {absolute path to OFFERS.mdf}  
  Une image contenant texte

  Description générée automatiquement
* Une fois connecté, lancer le script avec le bouton d’exécution en haut à gauche :  
  
* En output, il devrait y avoir les messages suivants :  
  Une image contenant texte, plaque, capture d’écran

  Description générée automatiquement

Cette procédure te sera utile si tu souhaites faire des modifications en base.

## Description du projet

Le projet OfferExporter est un batch qui exporte pour chaque produit d’un référentiel donné, toutes les offres du catalogue produits. Les exports sont des pseudos-fichiers JSON (une ligne = un JSON d’un produit) zippés au format GZ.

En lançant le projet en mode Debug, on a l’output suivant :

Une image contenant texte

Description générée automatiquement

Les exports sont générés dans un dossier \_OUTPUT où se trouve l’exe :

Une image contenant texte

Description générée automatiquement

Dans le projet, il y a un dossier \_ExpectedExports qui contient une version des exports attendus après les évolutions que vous allez apporter à l’application. Vous pourrez comparer les hash MD5 et même dézipper les fichiers pour voir l’attendu.

Une image contenant texte

Description générée automatiquement

## Evolution à réaliser

C’est parti ! Ci-dessous, voici les tickets à traiter pour le test technique. Tu peux les traiter dans l’ordre que tu préfères.

### Ticket 1

* Application : OfferExporter
* Demandeur : Les consommateurs des exports
* Criticité : Ultra Critical
* Difficulté : Low
* **Description** : Actuellement dans les exports, on retrouve des champs null. Par exemple, lorsqu’une offre d’un vendeur n’a pas de promo, on retrouve tout de même le champs reducedPrice dans le JSON :  
    
  Il faudrait faire en sorte que tous les champs null ne soient plus sérialisés :   
  

### Ticket 2

* Application : OfferExporter
* Demandeur : Les consommateurs des exports
* Criticité : Ultra Critical
* Difficulté : Medium
* **Description** : Actuellement dans les exports, on retrouve un champ reducedPrice qui correspond au prix promo d’une l’offre. En revanche, on ne précise pas quelle clientèle cette promotion s’applique. Il faut donc dans l’export de chaque offre, ajouter un champ discountFor pour indiquera la cible de la promo (Public, Member, Company (B2B))  
  
* Commentaire : Dans l’équipe, personne ne connait ce batch, ni où se trouve les données demandées.  
  Bon courage !

## Ticket 3

* Application : OfferExporter
* Demandeur : Tech Lead
* Criticité : Low
* Difficulté : Low
* **Description** :
  + Supprimer le Quick-fix: Exclude Darty referential (ID 2)
  + Mettre à jour le champs IsExportable de Darty dans le fichier OFFERS.sql
  + Réexécuter le script SQL pour mettre à jour OFFERS.mdf

## Ticket 4

* Application : OfferExporter
* Demandeur : Tech Lead
* Criticité : Medium
* Difficulté : Very High
* **Description** : Le code est lent, consomme trop de mémoire et n’est pas maintenable. Je te laisse refactoriser le code pour améliorer tout ça. Lors du code review, tu me présenteras les améliorations apportées. Il ne faut plus qu’on ait ces Quick-fix partout…