Exercice « menoum »

L'ensemble des parties de l'exercice « menoum » compte pour 5 % de la note finale.

Partie 1

Remise **mardi 1er septembre minuit**

À partir du projet « menoum »

1. Complétez les tests d'intégration des classes :

   - WriterRepositoryTests.cs
   - TagRepositoryTests.cs

2. Faites passer le test de la classe **HomeControllerIndexTests** en écrivant le code de la méthode **Index** (http get) de la classe **HomeController** afin d'afficher la liste des restaurants (nom, ville et pays). Suivez les consignes suivantes :

   - Ne vous souciez de la mise en forme (css).
   - Utilisez le repository de Restaurant pour aller chercher les restaurants dans la BD.
   - Pour peupler la BD, appeler la méthode CreateRestaurant de HomeController à partie de l'URL.
   - Vous devez envoyer à la vue une liste de **RestaurantIndexViewModel** et non une liste de **Restaurant**. La vue reçoit donc un **IEnumerable** de la classe **RestaurantIndexViewModel**. La première ligne de la vue devrait ressembler à :

     ```
     @model IEnumerable<Menoum.ViewModels.RestaurantIndexViewModel>
     ```

3. Vérifiez que les restaurants s'affichent dans le navigateur.

4. Répondez à la question qui se trouve dans le test **index_should_return_view_with_restaurants** de la classe **HomeControllerIndexTests**.

Partie 2

Remise **mercredi 2 septembre minuit**

Pour l'ensemble des questions :

- Ne vous souciez de la mise en forme (css).
- N'oubliez pas de réusiner le code, surtout pour les tests.
- Quelques questions à vous poser pendant l'écriture du code (tests et application):
  - Est-ce que les noms des variables, constantes et fonctions sont significatifs ?
  - Est-ce qu'il y'a de la répétition de code ?
  - Le code est-il facile à lire et à comprendre ?

1. Faites en sorte que le test dans **HomeControllerIndexTests** ne soit plus dépendant de la BD. Après les modifications, testez manuellement l'interface web dans le navigateur (nous verrons plus tard comment automatiser les tests de l'interface).

2. Écrivez et faites passer tous les tests de la classe **RestaurantControllerEditTests**. Créez un viewModel **RestaurantEditViewModel** afin de passer l'information d'un restaurant à la vue. Testez manuellement l'interface web dans le navigateur.

3. Écrivez et faites passer tous les tests de la classe **RestaurantControllerDeleteTests**. Créez un viewModel **RestaurantDeleteViewModel** afin de passer l'information d'un restaurant à la vue. Testez manuellement l'interface web dans le navigateur.

4. Écrivez et faites passer tous les tests de la classe **RestaurantControllerCreateTests**. Créez un viewModel **RestaurantCreateViewModel** afin de passer l'information d'un restaurant à la vue. Testez manuellement l'interface web dans le navigateur.

Partie 3

Remise **lundi 7 septembre minuit**

1. Modifiez l'ensemble du code de l'application pour utiliser **T4MVC**.

2. Modifiez l'ensemble des tests afin d'utiliser des Mocks (avec **Nsubstitute**) au lieu de « Fakes ».

3. Modifiez le code pour utiliser **Ninject** dans le projet au lieu d'avoir deux constructeurs dans les contrôleurs.

4. Modifiez l'ensemble du code pour utiliser **Automapper.**

5. Refactorisez vos tests pour qu'ils soient lisibles et évitez la répétition.