Technologies .NET - TP 1

Simulateur de crédit immobilier (1ère Partie)

L'objectif de premier TP est la découverte des technologies .NET et du MVC à travers une application de gestion de crédit immobilier.

1. Pré-Requis

- Avoir installé le framework .NET 4.5 et le MVC 5 sur votre machine

- Avoir installé Visual Studio 2015 Community sur votre machine

2. Création du projet

- Créez une application MVC, Fichier – Nouveau Projet – Menu Visual C# - Application ASP.NET MVC 4 – Nommez le "TP1-CreditImmobilier".

- Sélectionnez "Application Internet" afin de disposer d'un template de base pour pouvoir travailler.

3. Définition du modèle

Pour commencer nous allons définir le modèle de notre application, il va donc falloir créer les classes permettant de gérer la simulation et l'affichage du tableau de simulation.

3.1. La simulation du crédit

- Créez une classe Simulation avec les propriétés suivantes :
    - Un nom de client
    - Une date
    - le capital emprunté
    - la durée du crédit en mois
    - le nombre d'échéances par an
    - le montant des échéances,
    - le taux d'intérêt annuel,
    - et le taux d'assurance
    - Le taux d'intérêt périodique = Taux d'intérêt annuel / Nombre d'échéances par an,
    - Le taux périodique d'assurance = Taux d'assurance / Nombre d'échéances par an,
    - Le montant des échéances d'assurance = Capital * Taux périodique d'assurance.

- Faîtes en sorte que les champs suivants soient obligatoires pour la validation du modèle : Capital, Nombre d'échéances, montant des échéances ou durée du crédit, taux d'intérêt, taux d'assurance.

3.2. Le tableau d'amortissement du crédit

- Créer deux autres classes pour le tableau d'amortissement du crédit :
    - Une classe pour décrire une ligne du tableau d'amortissement :
        - Le capital restant dû avant l'échéance
        - Les intérêts à l'échéance
        - La partie de capital remboursé à l'échéance, i.e. l'amortissement

        - Les frais à l'échéance
        - Le montant réel de l'échéance
    - Une classe pour décrire le résultat d'une simulation :
        - Une simulation
        - Une liste de lignes d'amortissement
        - Le coût total des intérêts
        - Le coût total de l'assurance
        - Le coût total du prêt.

4. Création d'une simulation

Bon, on vient de définir le modèle de notre application Web, nous allons maintenant nous en servir à travers un contrôleur pour pouvoir saisir les informations de la simulation.

- Avant toute chose, vous allez ajouter pour chaque propriété des trois modèles, une annotation permettant de définir le label affiché dans la vue.

- Créez un nouveau contrôleur : SimulationController.

- Ajoutez une action en GET « Create », qui doit créer une nouvelle instance de la classe Simulation, en renseignant par le nom de l'emprunteur (l'utilisateur courant) et la date de la simulation qui est la date du jour. Cette action doit retourner une vue avec ce modèle.

- Il faut maintenant créer la vue correspondant au formulaire de création de la simulation. Pour cela, faîtes un clic droit sur la méthode « View » et cliquez sur « Add View ». Donnez-lui le même nom que l'action « Create », et indiquez que vous souhaitez une vue vide « Empty View ». Visual Studio vous créé une vue cshtml dans le répertoire « Simulation » du répertoire « Views ».

- Dans cette vue, définissez sur la première ligne le modèle que vous souhaitez utiliser dans la vue (ici une simulation…).

- Il s'agit maintenant de créer le formulaire de génération de la simulation. Pour cela, vous allez vous aider des deux HtmlHelpers que sont « Html.BeginForm » et « Html.EndForm ». Ces deux HtmlHelpers correspondent aux balises d'ouverture et de fermeture « form ». Dans le « BeginForm », indiquez la méthode POST qui va correspondre à la soumission du formulaire.

- Dans le formulaire, affichez en lecture seule le nom du client et la date de la simulation. Arrangez-vous pour que ces deux champs ne soient pas « perdus » lors de la soumission du formulaire (Regardez du côté des Helpers disponibles…)

- Ajoutez ensuite l'ensemble des éléments du formulaire qui sont éditables grâce aux Helpers (LabelFor, TextBoxFor, DropDownListFor)

- Ajoutez pour chaque champ obligatoire du formulaire, un Helper « ValidationMessageFor ». Cela permettra d'afficher un message d'erreur si le champ n'est pas valide.

- Le montant des échéances doit être un champ en lecture seule comme le nom du client et la date. Ajoutez un événement JavaScript sur la modification des champs (Capital, Taux Annuel, Durée en Mois) qui va permettre de calculer automatiquement le montant des échéances en fonction de

ces trois champs. Il faut bien évidemment contrôler que les champs sont renseignés et que ce sont bien des nombres décimaux.

- Faîtes de même avec l'échéance de l'assurance.

- Ajoutez un bouton de soumission du formulaire qui permettra de lancer la simulation

5. Tableau d'Amortissement

5.1. Création du Tableau D'Amortissement

Nous allons maintenant soumettre notre formulaire afin de calculer le tableau d'amortissement et de l'afficher dans une vue spécifique.

- Créez une nouvelle méthode qui porte le même nom que celui que vous avez donné dans la méthode « BeginForm ». Notez la en requête POST. Donnez-lui comme paramètre un objet de la classe « Simulation ».

- La première chose à faire quand on est dans une soumission de formulaire est de contrôler que le modèle transmis est bien valide. Pour cela, il vous faut utiliser la propriété « IsValid » de l'objet « ModelState » du contrôleur qui permet de savoir si le modèle est valide ou non. Si le modèle n'est pas valide, vous devez afficher la vue de création avec le modèle associé afin que l'utilisateur corrige ses erreurs. Faîtes un test pour bien vérifier que vous êtes redirigé vers le formulaire si le modèle n'est pas valide.

- Si le modèle est valide, alors vous devez créer le résultat de la simulation. Pour cela vous allez créer une méthode dans le contrôleur qui vous calculera votre résultat. Dans cette fonction, vous allez devoir créer une nouvelle instance de la classe ResultatSimulation, et l'enrichir. Il faut donc associer la simulation au résultat et créer les lignes de facturation, le tout en mettant à jour les totaux du résultat. L'algorithme de calcul des lignes de facturation est le suivant.

```
for (int ech = 0; ech < Duree; ech++)
{
    LigneAmortissement la = new LigneAmortissement();
    // Le CRD
    if (ech == 0)
    {
        // Première échéance, le restant dû est le capital
        la.CRD = Capital;
    }
    else
    {
        // On récupère la ligne précédente pour calculer le restant dû (arrondi à 2 chiffres après la virgule,
        //  puisque le centime est la plus petite unité payable dans le système monétaire français)
        LigneAmortissement laPrec = TableauAmortissement.Lignes[ech - 1];
        la.CRD = Math.Round(laPrec.CRD - laPrec.Amortissement, Simulation.NbDecimal);
    }
    // Les intérêts (arrondis également)
    la.Interets = Math.Round(la.CRD * TauxPeriodique, Simulation.NbDecimal);
    TableauAmortissement.CoutTotalInterets += la.Interets;
    // L'armortissement (arrondi également)
    if (ech < Duree - 1)
        la.Amortissement = Math.Round(Echeance - la.Interets, Simulation.NbDecimal);
    else
    {
        // Dernière échéance
        la.Amortissement = la.CRD;
    }
    // Les frais : l'assurance ! (arrondis également)
    la.Frais = Math.Round(EcheanceAssurance, Simulation.NbDecimal);
    TableauAmortissement.CoutTotalFrais += la.Frais;
    // L'échéance
    la.Echeance = la.Interets + la.Amortissement + la.Frais;
    TableauAmortissement.CoutTotalCredit += la.Echeance;
    TableauAmortissement.Lignes.Add(la);
}
```

- Faîtes ensuite appel à cette fonction dans votre POST, pour récupérer le modèle de résultat, et retournez le dans une vue appelée « Resultat ».

5.2. Affichage du Tableau d'Amortissement

- Créez une nouvelle vue « Resultat ».

- Dans cette vue, vous allez d'abord afficher dans un premier bloc un résumé des informations qui ont été saisies par l'utilisateur (Toutes les propriétés de la propriété « Simulation » du modèle)

- Affichez dans un deuxième bloc le tableau d'amortissement de la simulation, avec les différents totaux.

- Travaillez sur le style de la page pour qu'elle soit un minimum « User-Friendly »…