Gestion du Prêt

Préalable

Finir les écrans Introduction puis saisieBien, avec le client en c#.
Sauvez son projet avant de continuer le projet pour avoir un point de phase.

Cahier des charges

Ce document est sujet à évolution en fonction des besoins (V1.3).
Ce cahier des charges vous propose de partir de la description de la classe et de créer l'écran associé.
L'utilisateur peut renseigner le montant à financer, le taux annuel, le nombre de mensualités ou le montant de la mensualité.
Pour un montant de 25000 euros, il faudra rembourser 450 euros sur 60 mois au taux de 3.5 % sur l'année (exemple non contractuel).
Si un des champs n'est pas renseigné sur l'écran, il faudra en déduire que c'est le champ à calculer.

Pour les mensualités, au maximum, on travaillera sur 6 ans
Pour le taux, on prendra un taux proportionnel (voir math financière)
Vous pourrez facilement vérifier vos calculs sur Internet.

Attention, dans votre projet il ne faudra pas instancié un objet prêt mais un objet crédit.

Au niveau de l'interface Homme-Machine, il faudra respecter la charte graphique de l'entreprise.
Il faudra aussi suivre les règles de nommage propre à l'entreprise.
Par exemple, il faudra également prévoir l'appel des autres écrans du projet.

Description de la classe de base

Création de la classe Prêt avec ses attributs
A écrire en C#

Abstrac classe pret // c'est une classe générique --> à mettre en classe abstraite (plus tard)

private monMontant as double ' montant financé
private monDuree as integer ' durée du prêt
private monMensualité as double ' mensualité
private monTaux as double ' taux du prêt

sub new() ????
// constructeur
end sub

```
sub new(montant, duree, mensualite, taux)
        monMontant = montant;
        monDuree = duree;
        monMensualité = mensualite;
        monTaux = taux;       // les 4 paramètres sont donnés, est-ce possible ?
end sub
```

```
sub new(duree, mensualite, taux)
        monDuree = duree;
        monMensualité = mensualite;
        monTaux = taux;
        // ? monMontant = 0;      // dans ce cas, le montant à financer sera calculée
end sub

sub new(montant, duree, taux)
        monMontant = montant;
        monDuree = duree;
        monMensualité = ??  // dans ce cas, la mensualité sera calculée
        monTaux = taux;
end sub

function getMensualités() as double
    return monMensualité    // retourne une mensualité déjà calculée
end function

function getMensualités(montant, durée, taux) as double

        ' taux --> proportionnel    // en déduit qu'il faut la calculer, vu que les montants sont transmis

        return monMensualité

end function


fin classe   /// Vérifier comment créer les méthodes
```

Description de la classe Crédit

C'est la classe à utiliser pour votre projet, pour l'instant.
Attention, la classe prêt permet aussi de créer de la LOA, de la LLD, de la LLD-OA, …
Vous pouvez rapatrier des méthodes de la classe prêt si vous le juger nécessaire. En effet, les calculs de mensualités ne sont pas identiques pour la LOA et un Crédit classique.

```
classe credit hérite de pret

……  // a utiliser dans le projet

// écrire les mêmes méthodes dans la classe prêt et la classe crédit
```

- ➔ Classe Prêt : virtual ? abstract ??
- ➔ Classe Credit : override ??

```
fin classe
```

Description de la classe LOA

// pas à instancier pour l'instant.

Classe LOA hérite de Pret

private monLoyerInitial as double
private monBouquetFinal as double

LOA()

LOA(LoyerInitial)
{
monloyerInital = LoyerInital;
}

LOA(LoyerInitial,BouquetFinal, Duree, Mensualite, taux)
{
monloyerInital = LoyerInital;
monBouquetFinal = BouquetFinal;
pret(Duree, Mensualite, taux);
}