LIEN POUR LE PROJET :

http://s183418.php2.hec.ulg.ac.be/projet-groupe63/accueil.php

Démarche générale du travail :

Pour commencer, nous avons créé tous nos fichiers qui allaient constituer notre site, c'est-à-dire chaque page du site. La première étape était d'insérer notre logo dans le header. Quant au format de cette image que nous appellerons « logo.png », nous l'avons défini dans un fichier CSS. Ensuite vient la barre de navigation. Elle est la même pour toutes les pages, sans exception. Grâce au CSS, nous avons pu la modeler et la rendre un peu plus agréable et ergonomique. Tous les liens (a href) dans la barre de navigation sont indiqués par chemins relatifs, afin que cela fonctionne sur n'importe quelle machine.

De manière générale, les éléments cités plus haut sont les mêmes pour chaque page. Ce qui change est le contenu de chacune.

Algorithme simulateur 1 (mensualité) :

Au tout début du programme, on vérifie si le cookie « count » est initialisé. Etant donné qu'il n'existe pas, on le crée, on le nomme « count » et on lui assigne la variable $cookie comme valeur. Nous expliquerons le principe plus tard dans ce travail. On passe la partie html qui est similaire aux autres simulateurs pour arriver à la partie php. On commence le php avec un if qui vérifie si l'utilisateur a appuyé sur le bouton « valider » pour lancer la simulation. Ceci permet d'éviter des erreurs, l'algorithme ne se déclenche que lorsque l'utilisateur lance la simulation. Ensuite, nous avons décidé de vérifier si l'utilisateur avait bien rentré toutes les informations nécessaires au calcul. Dans le cas contraire, le message « vous n'avez pas fourni toutes les données nécessaires ! » s'affiche. Après cela, nous vérifions si l'utilisateur a correctement choisi un taux dans la liste déroulante. De nouveau, dans le cas contraire, le message « vous n'avez pas choisi un taux d'intérêt ! » s'affiche. Cela étant juste dans un but d'optimisation, afin d'éviter des erreurs.

Une fois que l'utilisateur a rentré toutes les données correctement, l'algorithme se lance. Au début, on initialise les variables $montant, $duree et $tauxchoisi en leur attribuant la valeur que l'utilisateur a encodé grâce à $_POST. Après cela vient le switch. Il vérifie le taux choisi par l'utilisateur et, en fonction de celui-ci, calcule :

- Le taux : il est fixé en premier lieu. Il servira au calcul de la mensualité.
- La mensualité : il s'agit de la formule affichée dans les consignes du projet qui prend en compte la variable $tx préalablement définie. Le montant est arrondi à deux décimales afin d'afficher un montant au format monétaire.
- La somme réellement remboursée : c'est le produit des deux variables $mensualite et $duree

- Les intérêts totaux de l'emprunt : c'est la différence entre la somme réellement remboursée et le montant emprunté.

Ensuite, les variables $interets et $somme_rem sont arrondies à 2 décimales et affichées avec un séparateur de milliers (un point) afin de les rendre plus agréables à lire grâce au format monétaire. A la fin du switch, on affiche un message récapitulatif reprenant toutes les données que l'utilisateur a entré dans le formulaire, avec le montant de la mensualité, la somme totale à rembourser ainsi que le montant des intérêts. Pour finir, un if vérifie si la valeur du cookie est égale à 5. Le cas échéant, l'utilisateur est redirigé vers la page « demande de devis », la valeur de la variable $cookie est remise à 0 et réinsérée dans le cookie « count », pour redémarrer le compteur. A la fin de la première simulation, le programme se relance et, au début, incrémente la valeur de la variable $cookie puis la réinsère dans le cookie « count ». Après la 1ère simulation, la variable $cookie a donc la valeur 1.

Algorithme simulateur 2 (calcul de la capacité d'emprunt) :

Ici, le principe est très similaire à celui du premier simulateur. La partie avec le cookie est exactement la même que celle citée ci-dessus. Ici, on demande à l'utilisateur de fournir 2 données et de choisir un taux d'intérêt dans la liste prédéfinie, qui est la même que celle du premier simulateur. On vérifie de nouveau si une mensualité et une durée sont renseignées puis on vérifie si l'utilisateur a bien choisi un taux d'intérêt. Le cas échéant, l'algorithme commence. Les variables $mensualite, $duree et $tauxchoisi sont initialisées et prennent comme valeur les entrées de l'utilisateur. Ensuite, on calcule la somme réellement remboursée grâce à la variable $somme_rem qui est le produit des variables $mensualite et $duree. Après cela vient le switch, qui va tester la valeur de la variable $tauxchoisi récupérée. Le switch suit la procédure suivante :

- Le taux est fixé grâce à la variable $tx
- La capacité d'emprunt est calculée et stockée dans la variable $montant. Il s'agit de la formule donnée dans les consignes du projet dans laquelle on a isolé le montant empruntable, le c dans la formule. Le montant est arrondi à 2 décimales grâce à la fonction round()
- La variable $interets représentent le montant d'intérêt total qui sera payé. Il est calculé en soustrayant le montant emprunté $montant de la somme réellement remboursée $somme_rem
- Pour finir, les 3 variables $montant, $interets et $somme_rem sont mise en forme grâce à la fonction number_format(), afin d'avoir un format monétaire plus agréable à la lecture, comme dans le premier simulateur.

Après le switch, on demande au programme d'afficher la capacité d'emprunt préalablement calculée ($montant) ainsi que les intérêts ($interets) et le montant total à rembourser ($somme_rem)

Enfin, on vérifie la valeur de la variable $cookie stockée dans le cookie « count ». Si elle est égale à 5, l'utilisateur est redirigée vers la page « demande de devis » grâce à la fonction header(), la variable $cookie est remise à 0 puis stockée dans le cookie « count » afin que le compteur redémarre (de 0 à 5). Ensuite, le code redémarre. L'algorithme de ce deuxième simulateur est calqué sur le premier. Le principe est le même, la différence réside dans les montants à calculer ainsi que les variables employées.

Algorithme simulateur 3 (crédit hypothécaire) :

Ce simulateur est un peu plus complexe que les autres car il s'agit ici de calcul un crédit hypothécaire et d'afficher tout le tableau d'amortissement y afférant. Le code est décomposé en 2 parties : la première, semblable aux deux précédents, qui va calculer la mensualité, la somme réellement remboursée et les intérêts (comme dans le 1er simulateur) et la deuxième partie qui va afficher toutes les données dans un tableau afin de générer le tableau d'amortissement du crédit simulé.

La première partie est totalement identique au 1er simulateur. Le mécanisme avec le cookie est aussi le même. La seule différence est que, dans ce simulateur, la durée n'est pas fournie par l'utilisateur mais associée au taux d'intérêt choisi. De nouveau, on affiche un message récapitulatif reprenant le montant emprunté, le taux d'intérêt choisi et la durée qui y est associée, la mensualité, l'annuité (mensualité * 12) ainsi que la somme totale à rembourser et les intérêts payés au total. Dans le switch, on fixe dans chaque cas la durée que l'on stocke dans la variable $duree. Cela peut paraître inutile mais nous nous en servirons plus tard.

La deuxième partie, quant à elle, est inédite. Tout d'abord, on affiche grâce à l'HTML les titres des colonnes du tableau d'amortissement. Maintenant qu'on a les titres de nos colonnes, on va remplir ces colonnes de manière dynamique, avec les montants correspondants. En premier lieu, on teste si l'utilisateur a bien appuyé sur « générer le tableau d'amortissement » puis s'il a bien rempli la case « montant » et choisi un taux d'intérêt et une durée. Le cas échéant, le programme commence. Les variables suivantes sont initialisées :

- $num_versement servira à calculer le numéro du versement.
- $srd qui correspond au solde restant dû est transformé en un tableau, vide pour l'instant.
- Le premier élément du tableau $srd sera le montant que l'utilisateur a entré, la variable $montant.
- $i est un compteur qui nous permet d'afficher les années afin de rendre le tableau plus clair et plus facile à lire.
- $j est aussi un compteur qui nous donnera le dernier solde restant dû, afin de calculer les intérêts de chaque versement sur ce solde.
- $a est le compteur des années qui s'affiche toutes les douze lignes pour rendre le tableau plus clair.

Afin de répéter les calculs d'une ligne sur toutes les lignes, on va utiliser une boucle. Dans ce cas-ci, nous utilisons un while() qui teste, avant de s'exécuter, si le numéro du versement est plus petit que la durée. De cette manière, pour une durée de 10 ans par exemple, lorsque la variable $num_versement aura 120 comme valeur, la condition ne sera plus remplie et la boucle s'arrêtera. Dans la boucle, la première instruction consiste à calculer la quote-part intérêts de la 1ère mensualité. Pour cela, on utilise la variable array $srd avec comme index la variable $j (qui vaut 0 pour le moment, car le premier élément dans un array est le 0ème élément) qu'on multiplie par le taux divisé en 12 pour avoir le taux mensuel. La deuxième instruction, elle, calcule la quote-part capital de la mensualité qui est la différence entre la mensualité et la quote-part intérêts. Après cela vient le if qui va tester si la division euclidienne du compteur $i est égale à 0. Le cas échéant, cela veut dire que la valeur de $i est égale à 0 ou un multiple de 12, et que l'on vient donc de finir une année dans le tableau et qu'on en commence une autre. On va donc incrémenter le compteur $a et afficher une nouvelle ligne sur laquelle sera indiqué le numéro de l'année. (NB : Année 1 est la toute première ligne car 0 % 12 = 0). Si la condition n'est pas respectée, on crée une nouvelle ligne sur laquelle on va

remplir chaque cellule avec le montant correspondant. La toute première cellule est celle du numéro de versement. Il s'agit de la variable $num_versement que l'on va incrémenter (car elle vaut 0 actuellement) et afficher. La cellule suivante reprend la mensualité que l'on affiche arrondie à 2 décimales grâce à la fonction round(). La 3ème est la quote-part intérêts préalablement calculée. Il s'agit du taux d'intérêt divisé par 12 et multiplié par le solde restant dû de la ligne antérieure (et du montant emprunté pour la toute 1ère ligne). Après avoir affiché le montant des intérêts aussi arrondi à 2 décimales, on incrémente la variable $j, afin de définir le « dernier solde restant dû » comme celui de la ligne actuelle. La prochaine cellule affiche la quote-part capital préalablement calculée arrondie à 2 décimales. La dernière cellule renvoie le solde restant dû APRÈS déduction de la quote-part capital. Pour cela, on stock dans la variable $srd de type array() la différence entre le $ième solde restant dû (ici, pour la première itération de la boucle, comme $i vaut 0, il s'agit du premier élément du tableau, qui est donc la variable $montant qu'on a stocké au moment d'initialiser les variables) et la quote-part capital. Cette valeur sera donc la 1ème valeur dans le tableau (NB : la 2ème vu que ça commence à 0) et on va l'afficher en prenant le $ième +1 élément (celui qu'on vient de calculer), arrondi à 2 décimales. Finalement, on ferme la ligne et la boucle recommence jusqu'à ce que la condition ne soit plus respectée.