**TP 1 ARCHITECTURE MVC PHP**

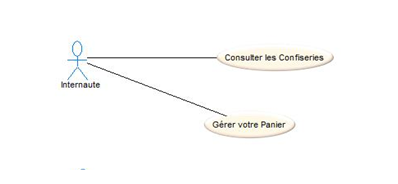
L’application Vanille permet de gérer un site en ligne de vente de confiseries de différentes catégories.

Une partie de cette application a été développée en architecture MVC.

Le Dossier **Vanille\_TP1** contient les éléments de cette partie du site.

Le fichier **vanille.sql** contient le script d’implantation dans Mysql de la base de données utilisée, nommée ***vanille***. Elle contient les produits et les catégories de produits en vente.

Pour l’instant la partie de l’application développée propose les 2 cas d’utlisation suivants :



# Première partie : Implantation de l’application

Installation de l’application web sur votre plateforme

**Travail à faire**

1 - Installer les fichiers de l’application, ainsi que la base de données sur la plateforme web choisie ( easyphp, wampserver, .. )

2 – Apporter les éventuelles modification afin de rendre le site fonctionnel

# Deuxième partie : Découverte de l’application

**Préalable : Organisation des fichiers en architecture MVC**

Ouvrez chaque dossier de l’application pour comprendre le contenu et la logique de l’arborescence des dossiers et fichiers.

Où est la partie modèle de l’architecture MVC ?

La partie MVC est dans le dossier « util ».

Dans cette application on retrouve pour l’instant les 2 cas d’utilisation suivants :

**Consulter les confiseries 🡪 c\_voirProduits.php**

**Gérer votre Panier 🡪 c\_gestionPanier.php**

Quel est le nom du contrôleur principal ?

Analysez son contenu

Le contrôleur principal est « index ».

**a ) Analyse du Cas d'utilisation *voirProduits***

|  |  |
| --- | --- |
| Ce cas propose deux vues externes:   * **Après l'appel du catalogue** * **Après le choix de la catégorie**   En fait ce cas d'utilisation propose 3 états les 2 précédents et un troisème correspondant au rechargement de la page après l'ajout d'un produit au panier    Analysez le code du contrôleur c\_voirProduits.php |  |

C'est la valeur de la variable $\_REQUEST['action'] qui permet d'aiguiller vers tel ou tel ***case***, c'est pourquoi il faut suivre **l'état applicatif** en passant de page en page **une variable action** qui est remplie pour construire un lien construit avec la bonne valeur, comme ci-dessous:

**<ul id="menu">**

**<li><a href="index.php?uc=accueil"> Accueil </a></li>**

**<li><a href="index.php?uc=voirProduits&action=voirCategories"> Voir le catalogue </a></li>**

**<li><a href="index.php?uc=gererPanier&action=voirPanier"> Voir son panier </a></li>**

Le paramètre **uc** fournit le cas d'utilisation, le paramètre **action** indique l'état dans le cas.

Idem pour le choix de visualisation des produits après sélection de la catégorie

**Analysez** **la vue** ***v\_categories.php***

Après le choix d’un produit, on peut s’apercevoir que si un produit est déjà dans le panier, un message lui est retourné

**Recherchez dans le code comment est effectué ce contrôle ?**

if(!$ok)

{

$message = "Ce produit est déjà dans le panier !!";

include("vues/v\_message.php");

}

**b) La gestion des erreurs**

Les erreurs liées à des saisies incorrectes sont signalées dans une sous-vue distincte (***v\_erreurs.php***), cette vue est incluse ou pas à la fin du formulaire de saisie reposté:

**Analysez la vue v\_erreurs.php**

**Recherchez comment est affiché le message « panier vide »**

if($n >0)

{

$desIdProduit = getLesIdProduitsDuPanier();

$lesProduitsDuPanier = $pdo->getLesProduitsDuTableau($desIdProduit);

$message = "Voici votre Panier";

include ("vues/v\_message.php");

include("vues/v\_panier.php");

}

else

{

**$message = "panier vide !!";**

**include ("vues/v\_message.php");**

}

**c) La gestion de la session**

Il faut comprendre que vous n'allez agir que sur un tableau $\_SESSION qui contient les produits sélectionnés dans le panier, à ce stade aucune interaction avec la base de données

**Recherchez comment est gérée la variable $session**

function initPanier()

{

if(!isset($\_SESSION['produits']))

{

$\_SESSION['produits']= array();

}

}

# Troisième partie : Amélioration de l’application

**TRAVAIL A FAIRE**

1. Améliorer l’affichage de l’application, notamment lors de l’accueil (configuration de l’entête, bandeau, pied, ……) en changeant les dimensions, couleurs, polices, textes ...
2. Ajouter la possibilité de supprimer l’ensemble du panier par l’utilisateur.

On utilisera la fonction php permettant de détruire la variable de session.

**Recherchez cette fonction php ?**

La suppression du panier se fera en 3 étapes :

* Créer la fonction **supprimerPanier** dans le fichier des fonctions (.inc)
* Modifier le contrôleur concerné pour ajouter la fonctionnalité
* Modifier la vue correspondante en ajoutant un bouton de commande dans la page concernée. On rajoutera la boite de dialogue de confirmation ( On Click…..) . Une image annulerPanier est fournie dans le dossier images de l’application

En complément, à l’issue de la suppression on affichera le message « panier supprimé » dans l’interface utilisateur

**Effectuez les modifications dans les fichiers correspondants**

1. Ajouter la possibilité de supprimer un produit choisi du panier.

On utilisera le même principe que pour la suppression totale du panier (création d’une fonction, modifier le contrôleur et la vue), mais il faudra sélectionner le produit à supprimer.

Pour la fonction, on utilisera la fonction php permettant de rechercher l’index du produit à supprimer dans le tableau **$\_SESSION['produits']**

On fera apparaitre un bouton ***retirer panier*** pour chaque produit du panier.

Après suppression du produit du panier, on fera apparaitre un message de confirmation de la suppression.