CFPT-I

Catal'info

TPI

SEEMULLER Julien

28/04/2015

Table des matières

Introduction

Dans le cadre de mon travail pratique individuel, le but est de réaliser un site web pour qu'un revendeur puisse fournir un catalogue en ligne des produits informatiques disponibles dans son établissement.

Les utilisateurs peuvent rechercher et visualiser les informations relatives aux composants ou produits informatique. Le catalogue informatique est géré par les administrateurs du site web et permet de fournir un maximum de documentation à l'utilisateur tel que des images, des descriptions du produits, le prix des produits ou encore des documentations sur les produits présentés.

Le site possède aussi différents moyens de classer les produits :

- Trier par popularité
- Trier par le choix des administrateurs
- Trier par date d'ajout

En plus d'une fonction de classement, il est aussi possible de rechercher des produits et de les afficher par catégories.

Etude d'opportunité

Il existe de nombreux sites de revendeurs de produits informatique, en voici quelques exemples :

① PRODIMEX

http://www.prodimex.ch/

Points positifs :

- Images représentatives du produit
- Recherche précise du produit
- Médias dans descriptif du produit
- Titre du produit possède une description
- Changement de devises CHF vers EUR et inversement
- Affichage disponibilité des produits à l'aide d'un icone

Points négatifs :

- Menu supérieur confus (trop de termes & catégories vagues)
- Redondance dans les menus
- Interface peu intuitive par moments
- Ne possède pas de site mobile

② 1000 ORDI

http://www.1000ordi.ch/

Points positifs :

- Interface intuitive
- Site disponible en deux langues différentes
- Menu supérieur propre
- Images représentatives du produit
- Recherche précise du produit
- Médias dans descriptif du produit
- Titre du produit possède une description
- Affichage disponibilité des produits à l'aide d'un icone
- Liste de comptabilité avec un produit en particulier

Points négatifs :

- Ne possède pas de site mobile

③ STEG

http://www.steg-electronics.ch/fr/Default.aspx

Points positifs :

- Site disponible en trois langues différentes
- Catégories claires
- Affichage disponibilité des produits à l'aide d'un icone
- Images représentatives du produit
- Recherche précise du produit
- Médias dans descriptif du produit

Points négatifs :

- Ne possède pas de site mobile
- Interface peu intuitive par moments
- Site « trop long » (information pas instantanément accessible)

Il existe bien sûr beaucoup d'autres sites de revendeurs de produits informatique, j'ai décidé de limiter mon étude d'opportunité sur des sites populaires suisses.

Le site web partage donc des similitudes avec ces différents sites web. Par exemple, le site web possède une interface intuitive à l'utilisateur, un système de recherche par mots-clés, un affichage graphique des produits ou encore un affichage de la disponibilité des produits à l'aide d'icônes.

Une des problématiques présente sur la totalité des sites de mon échantillon est le manque d'une interface disponible pour téléphone mobile. De nos jours, les utilisateurs utilisent de plus en plus leur smartphone pour accéder à des sites web. Je pense qu'il est donc important d'implémenter une interface adaptable au format mobile.

Analyse fonctionnelle

Généralités

Voici le fonctionnement général de Catal'info :

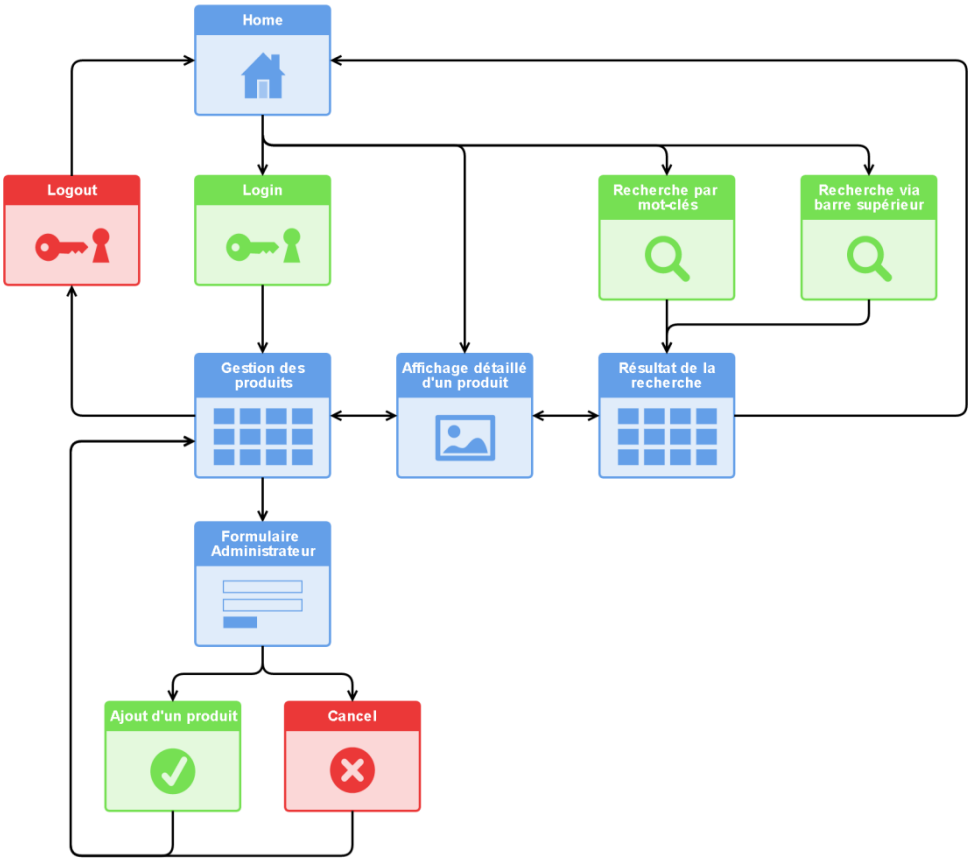
L'utilisateur est tout d'abord accueilli sur une page contenant les produits suggérés par les administrateurs du site, les produits les plus populaires auprès des utilisateurs et les produits ajoutés récemment.

Cette page possède aussi une barre de recherche permettant à l'utilisateur d'entrer des critères tels que le titre du produit, la marque ou encore des éléments de la description, puis on mène l'utilisateur sur une page contenant le résultat de sa recherche. Sur le côté gauche de la page, une liste des mots clés est affichée qui permet à l'utilisateur d'accéder rapidement à une catégorie (ex : RAM, Affiche une page avec la totalité des produits contenant le mot clé « RAM »).

Si un visiteur ou un administrateur clique sur un produit, on affiche les informations détaillées du produit selectionné.

Il existe deux types d'utilisateurs consultant le site. Le premier est le visiteur, il peut consulter les différentes informations des produits du site. Le deuxième est l'administrateur, il possède les mêmes privilèges que le visiteur mais peut aussi ajouter, modifier ou encore supprimer des produits. Quand un administrateur est connecté, il peut accéder à la modification ou suppression d'un produit en cliquant sur un icone d'engrenage se trouvant dans le coin de l'objet à modifier. Pour ajouter un produit, l'administrateur passe par une interface accessible via un lien dans la barre de navigation du site web.

Voici un schéma représentant les différentes pages du site web et les interractions entre ces dernières :



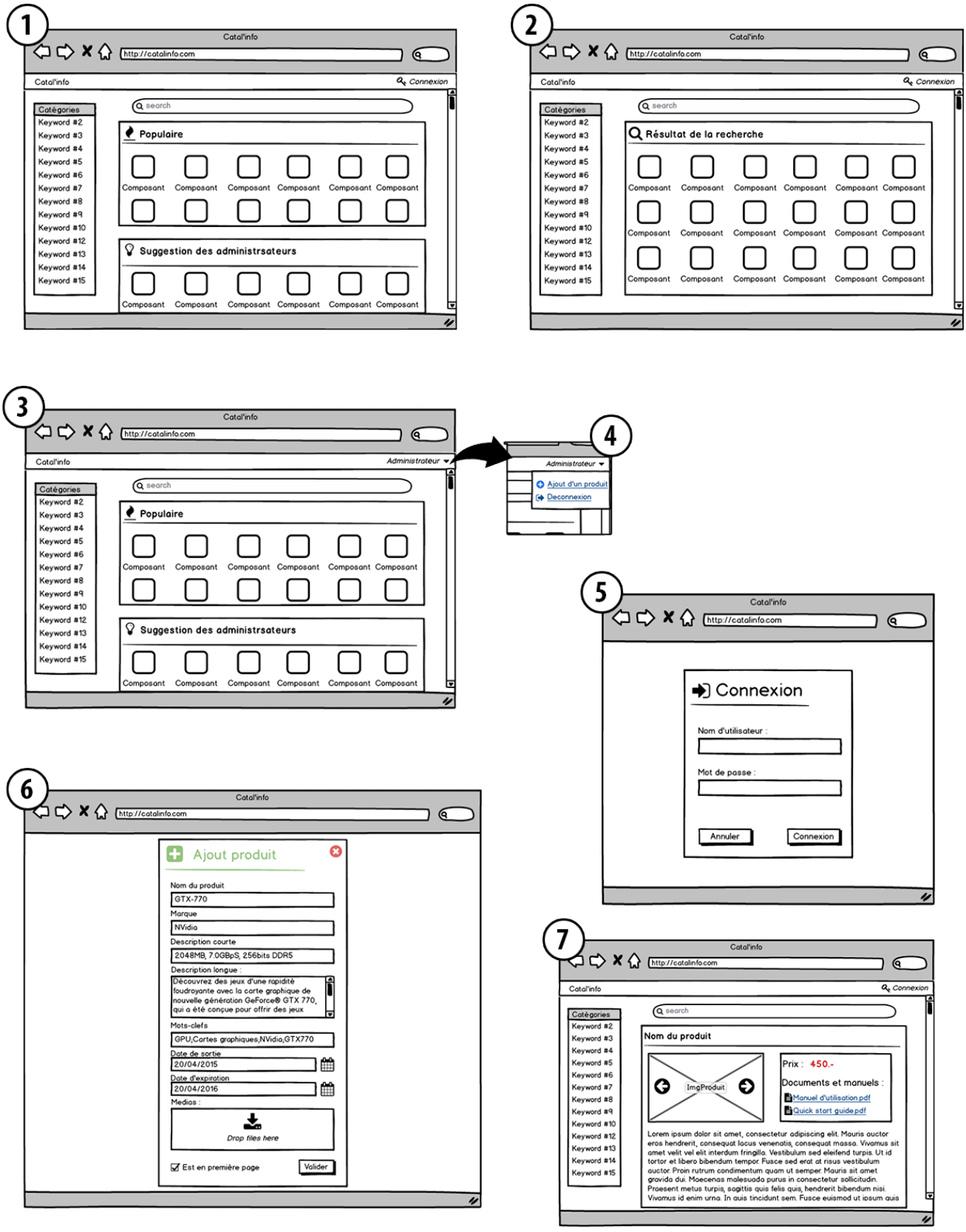
Description des fonctionnalités globales

Voici les fonctionnalités globales que possède Catal'info :

- Afficher les produits populaires et les sélections des administrateurs sur la page d'accueil
- Afficher la liste des produits selon une recherche de l'utilisateur
- Afficher la liste des produits selon un mot clé
- Les administrateurs peuvent :
    - Ajouter un produit et ses médias
    - Modifier un produit et ses médias
    - Supprimer un produit et ses médias
- Affichage détaillé des produits

Description des fonctionnalités particulières

Description détaillée de l'interface



① **Page principale (Visiteur)**

La page principale du site web est la page dont on accède en arrivant sur le site web, elle peut être résumée en cinq parties principales :

- L'affichage des produits les plus populaires
- L'affichage des produits recommandés par les administrateurs du site web
- Le bouton de connexion pour administrateur
- La barre de recherche
- La liste des mots clefs pour la recherche par mots clefs

Ces différents éléments sont tous interactifs en renvoient l'utilisateur à une partie différente du site web. Cliquer sur un des produits populaire ou recommandé par l'un des administrateurs ouvre les « ⑦ Détails du produit ». Cliquer sur le bouton de connexion renvoie l'utilisateur vers la page de « ⑤ Connexion administrateur ». Cliquer sur l'un des mots clefs présent sur la partie gauche du site web permet à l'utilisateur de filtrer la liste des produit selon un critère, il est renvoyé vers la page du « ② Résultat de recherche ». Finalement, si l'utilisateur valide une recherche, on le renvoie sur la page du « ② Résultat de recherche », puis on filtre les résultats affichés selon les mots entrés par l'utilisateur.

② **Résultat de recherche**

La page de résultat de recherche est accessible de deux façons, on y accède quand l'utilisateur clique sur l'un des mots clefs. Il est aussi possible d'y accéder après avoir validé une recherche à l'aide de la barre de recherche.

Le but principal de cette page est de permettre à l'utilisateur d'afficher des produits regroupés par critères tel que leur titre, leur description, leur marque ou encore à l'aide d'un des mots clefs présent sur la partie gauche de l'interface.

Il existe plusieurs éléments avec lesquels on peut interagir sur cette page. Cliquer sur l'un des produits affichés nous renvoie sur la page des « ⑦ Détails du produit ». Cliquer sur le texte « Catal'info » présent dans la barre supérieure renvoie l'utilisateur à la « ① Page principale (Visiteur) » ou à la « ③ Page principale (Administrateur) » en dépendant du fait que l'utilisateur soit connecté ou non.

③ **Page principale (Administrateur)**

La page principale de l'administrateur est uniquement accessible à un administrateur connecté à Catal'info. Cette dernière est identique à la « ① Page principale (Visiteur) » à l'exception de quelques éléments :

- La page possède un « ④ Menu déroulant en session administrateur »
- La page permet à l'administrateur de gérer des produits

Le but principal de cette page est de permettre à l'administrateur de modifier ou de supprimer rapidement des produits. L'interface est intuitive car elle est presque identique à la « ① Page principale (Visiteur) ».

Un icone d'engrenage placé dans le coin du cadre d'un produit permet à l'administrateur d'accéder au formulaire de modification du produit.

④ **Menu déroulant en session administrateur**

Le menu déroulant est accessible uniquement depuis une session administrateur et permet à l'administrateur de se déconnecter ou encore d'accéder à la page « ⑥ Ajout d'un produit / Modification produit ».

Pour accéder au menu déroulant, l'administrateur clique tout simplement sur son nom en haut à droite de la « ③ Page principale (Administrateur) ».

⑤ **Connexion administrateur**

On peut accéder à cette page depuis n'importe quelle autre page du site tant que l'on n'est pas connecté (statut visiteur). L'accès au formulaire est simple, il suffit de cliquer sur le lien « Connexion » se trouvant dans la barre supérieure de la majorité des pages du site.

Ce formulaire est uniquement utile à un administrateur de Catal'info, car il lui permet de se connecter et d'accéder aux formulaires de gestion de produits.

Après avoir validé son nom d'utilisateur et son mot de passe, l'administrateur est redirigé vers la « ③ Page principale (Administrateur) ». Si des informations incorrectes sont rentrées à l'intérieur des champs, on affiche un message d'erreur au visiteur et on lui propose d'entrer à nouveau ses informations.

⑥ **Ajout d'un produit / Modification produit**

Cette page est affichée après qu'un administrateur ait cliqué sur le bouton « Ajout d'un produit » dans le menu déroulant. Une version légèrement différente de cette page est affichée lorsque on modifie les informations d'un produit, le titre est changé à « Modification produit » et les informations du produit sélectionné sont préinscris dans les différents champs du formulaire. Un produit possède les informations suivantes et il est donc nécessaire de remplir tous les champs suivants :

- Un titre
- Une marque
- Une description courte du produit
- Une description longue du produit
- La date d'ajout au magasin du produit
  - La date ou le produit va être disponible dans le catalogue en ligne
- La date « d'expiration » du produit
  - La date ou le produit est retiré automatiquement du catalogue en ligne
- Une ou des images du produit
- Des documents téléchargeables (.pdf, .xls etc…)
  - Par exemple : Manuel d'utilisateur
- Recommandation des administrateurs (« Est en première page »)
  - Cocher ce champ pour afficher le produit dans les suggestions des administrateurs

Après avoir entré la totalité des champs ou après avoir modifié l'un de ces derniers, on peut cliquer sur valider pour ajouter le produit ou appliquer les changements potentiellement apportés.

⑦ **Détails du produit**

Les détails d'un produit sont affichés après qu'un utilisateur / administrateur ait cliqué sur l'image d'un produit. Le rôle principal des détails du produit est, comme son nom l'indique, d'afficher en détail les différentes informations du produit. Cette page reste identique pour un administrateur et un utilisateur.

Une image du produit est affichée dans le coin gauche de la boite du produit et si le produit possède différentes images, les images défilent les unes après les autres. Une description est aussi affichée en dessous de l'image du produit, des documents téléchargeables (.pdf, .xml etc…) sont aussi disponibles, voir « ⑥ Ajout d'un produit / Modification produit ».

Description des éléments de sécurité

Ici seront indiqués les différents éléments composant la sécurité de Catal'info.

Compte administrateur

Il est possible d'ajouter, de modifier ou encore de supprimer des produits mais uniquement à l'aide d'un compte administrateur protégé par une combinaison de nom d'utilisateur et de mot de passe.

Connexion en tant qu'Administrateur

Pour sécuriser la connexion à Catal'info, les mots de passes des administrateurs seront encryptés à l'aide d'une combinaison de « SHA-1 » et de « MD5 ». On s'assure que si le mot de passe est intercepté à mi-chemin avec le serveur, que la personne malveillante ne puisse rien faire avec le mot de passe.

Avertissement avant suppression d'un produit

Avant de supprimer un produit, une fenêtre pop-up demande à l'administrateur de confirmer son action, cela permet de réduire les chances de fausses manouvres et la suppression accidentelle d'un produit.

Analyse organique

Généralités

Le site web Catal'info peut être résumé en cinq parties principales :

1. La page d'accueil, pour afficher les produits les plus vus et les produits recommandés par les administrateurs.
2. Le détail produit, qui permet à une personne d'obtenir toutes les informations d'un produit. La page possède aussi un support visuel et des documents téléchargeables.
3. La partie recherche du site web qui permet à l'utilisateur de rechercher des produits à l'aide de mots clefs ou d'une barre de recherche multicritères.
4. L'identification utilisateur, qui permet à une personne de se connecter au site web
5. La partie réservée aux administrateurs pour ajouter, modifier ou supprimer des produits.

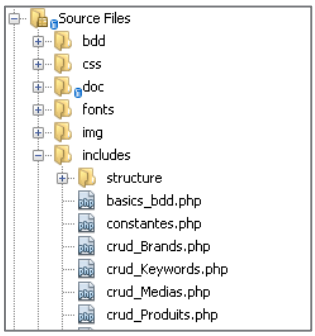
Les différentes fonctionnalités décrites seront donc classées par parties du site web. Pour plus d'informations sur chaque page de Catal'info, se rendre dans la « Description détaillée de l'interface »

Je parlerais aussi d'éléments importants tels que la base de données du site ou encore des fonctionnalités supplémentaires du site web.

Backups du site web

Structure du site web

Les fichiers du site web sont triés et structurés de manière à être auto-intuitifs à l'utilisateur et à permettre une organisation optimale du site web.



Des dossiers sont utilisés pour trier chaque partie individuellement. Les scripts PHP sont stockés à l'intérieur du dossier « includes », les feuilles de style css à l'intérieur du dossier « css » et ainsi de suite.

Cela permet une organisation du site web plus claire et la gestion des backups avec gitHub peut être plus organisée. [A MODIFIER PEU CLAIR]

Base de données

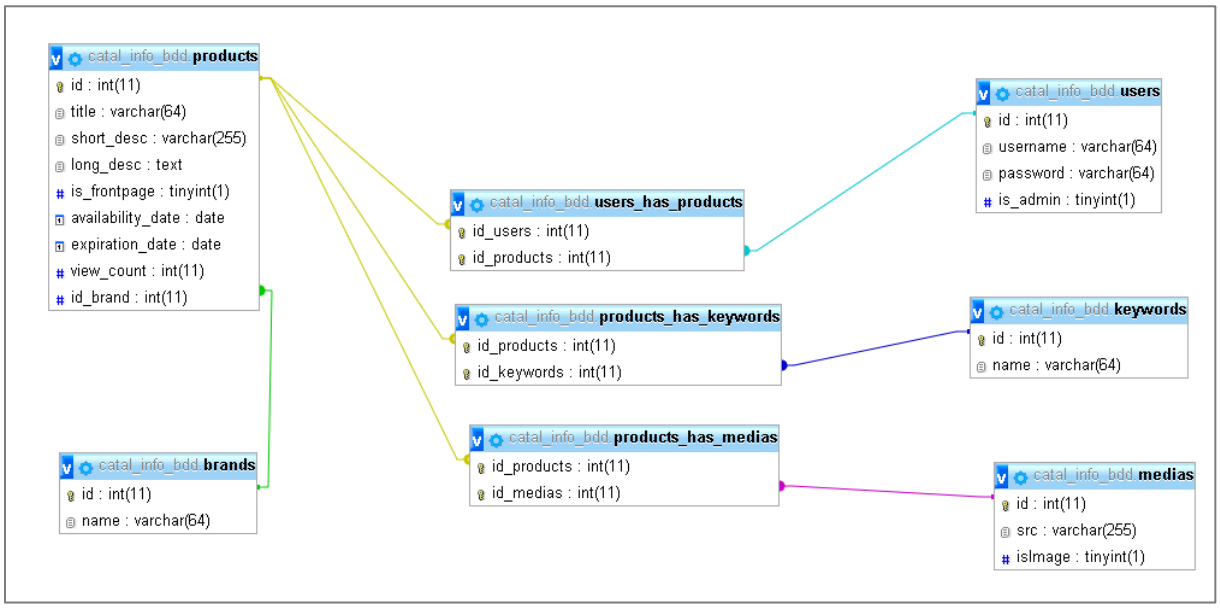
Introduction

Catal'info est un site de gestion de catalogue en ligne pour des revendeurs informatique, il est donc nécessaire de stocker et organiser les données dans une base de données. Pour la création de la base de données, j'ai utilisé « phpMyAdmin » (MySQL).

La base de données « catal_info_bdd » possède un total de huit tables permettant de gérer des produits, des utilisateurs, des mots clefs et même des medias.

Modèle relationnel de la base de données

Le modèle relationnel de la base a évolué tout au long du projet, voici la version finale du modèle relationnel de Catal'info :



Dictionnaire de données & Description des tables

Ci-dessous est décrit le contenu des tables et l'utilité de chaque champs pour Catal'info.

Table « brands »

La table brands contient les informations relatives aux marques des produits disponibles, elles sont identifiées par un id et possèdent un nom.

Exemple : « 1 – Asus »

Colonne	Type	Null	Défaut	Commentaires	MIME
id	int(11)	Non		L'identifiant unique de la marque.	
name	varchar(64)	Non		Le nom de la marque.	

Table « keywords »

La table des « keywords » contient les informations relatives aux mots clefs des produits ou aussi appelées « catégories ». Les mots clefs possèdent chacun un « id » pour les rendre unique et un nom.

Exemple : « 4 – Carte graphique »

Colonne	Type	Null	Défaut	Commentaires	MIME
id	int(11)	Non		L'identifiant unique du mot clef.	
name	varchar(64)	Non		Le mot clef.	

Table « medias »

La table des medias contient les informations relatives aux médias des produits (images, documents etc…). Un média possède un identifiant pour le rendre unique, une source indiquant ou se trouve le média sur le serveur et un champ « isImage » permettant de définir si le média concerné est une image.

Exemple : « 1 – up-content/img/test.png – 1 »

Colonne	Type	Null	Défaut	Commentaires	MIME
id	int(11)	Non		L'identifiant unique du média.	
src	varchar(255)	Non		Le chemin d'accès du média.	
isImage	tinyint(1)	Non		Décrit si le média est une image.	

Table « products »

La table principale de Catal'info. La table produit contient les informations relatives aux produits du site web. Chaque produit du site possède un identifiant unique (id), un titre, une description courte du produit, une description longue du produit, un champ permettant de dire si le produit doit être affiché en première page du site web, une date de disponibilité permettant de savoir quand le produit est disponible sur le catalogue, une date d'expiration permettant de savoir quand le produit va être retiré du catalogue, un nombre de vues, et une marque.

La marque du produit est en relation avec la table « brand » car un produit ne peut posséder qu'une seule marque et les marques sont des informations différentes des produits.

Exemple : « 1 – GeForce 770 – shortDesc – longDesc – 1 – 25.02.2005 – 14.03.16 – 99 – 1 »

Colonne	Type	Null	Défaut	Relié à	Commentaires	MIME
id	int(11)	Non			L'identifiant unique du produit	
title	varchar(64)	Non			Le titre du produit	
short_desc	varchar(255)	Non			Une courte description	
long_desc	text	Non			Une longue description	
is_frontpage	tinyint(1)	Non			Décrit si le produit est en première page	
availability_date	date	Non			Date de disponibilité du produit	
expiration_date	date	Non			Date d'expiration du produit	
view_count	int(11)	Non			Le nombre de vues du produit	
id_brand	int(11)	Non		brands -> id	L'id de la marque du produit (FK)	

Table « products_has_keywords »

Table intermédiaire permettant de faire une relation « many to many » entre les produits et leurs mots clefs (plusieurs produits possèdent plusieurs mots clefs).

Colonne	Type	Null	Défaut	Relié à	Commentaires	MIME
id_products	int(11)	Non		products -> id		
id_keywords	int(11)	Non		keywords -> id		

Table « products_has_medias »

Table intermédiaire permettant de faire une relation « many to many » entre les produits et leurs medias (plusieurs produits possèdent plusieurs medias).

Colonne	Type	Null	Défaut	Relié à	Commentaires	MIME
id_products	int(11)	Non		products -> id		
id_medias	int(11)	Non		medias -> id		

Table « users »

Table contenant les informations relatives aux utilisateurs et administrateurs de Catal'info. Chaque utilisateur possède un identifiant unique (id), un nom d'utilisateur utilisé pour se connecter, un mot de passe (encrypté en md5 & sha1), et un champ permettant de définir si l'utilisateur est un administrateur.

Exemple : 12 – Admin – 38758f53d77d5217d477433658b49a301f435feb – 1 »

Colonne	Type	Null	Défaut	Commentaires	MIME
id	int(11)	Non			
username	varchar(64)	Non			
password	varchar(64)	Non			
is_admin	tinyint(1)	Non			

Table « users_has_products »

Table intermédiaire permettant de faire une relation « many to many » entre les utilisateurs et leurs produits (plusieurs utilisateurs possèdent plusieurs produits). Cette table n'est actuellement pas utilisée, mais peut être utilisée pour faire un système de produits favoris ou de chariots.

Colonne	Type	Null	Défaut	Relié à	Commentaires	MIME
id_users	int(11)	Non		products -> id		
id_products	int(11)	Non		users -> id		

Script des fonctions basiques relatives à la base de données

Pour gérer la base de données, un script nommé « basics_bdd » a été créé. Ce dernier possède les fonctions basiques relatives à la base de données. Voici des exemples de fonctions que possède « basics_bdd » :

- Connexion à la base de données (PDO)
- Retourner le nombre d'enregistrements dans une table donnée
- Retourner un enregistrement par rapport à son « id »
- Retourner tous les enregistrements d'une table
- Supprimer un enregistrement d'une table par rapport à son « id »

« basics_bdd » permet aussi la simplification de la création de fonctions relatives à la base de données. Voici un exemple d'une fonction crée à l'aide de « basics_bdd », retournant le nombre de produits dans la base de données :

```php
function countProducts() {
    global $tableProducts;

    return countFields($tableProducts);
}
```

La taille de la fonction a été grandement réduite grâce à la création de fonctions basiques.

Fonctions principales relatives à la base de données

Ici seront décrites et analysées les fonctions principales se trouvant à l'intérieur du script « basics_bdd » :

Connexion à la base de données :

Voici le fonctionnement pour la connexion à la base de données : On essaye de se connecter à la base avec les informations relatives à la base contenues à l'intérieur de constantes.

Puis si cela réussit, on stocke le résultat dans une variable nommée « $bdd », sinon en renvoie l'erreur de PDO.

```php
function connection() {
    try {
        $bdd = new PDO('mysql:host=' . DB_HOST .
                ';dbname=' . DB_NAME, DB_LOGIN, DB_PASS,
                array(PDO::ATTR_PERSISTENT => true));

        return $bdd;
    } catch (PDOException $e) {
        print "Erreur !: " . $e->getMessage() . "<br/>";
        die();
    }
}
```

Récupération de tous les champs d'une table

Pour récupérer tous les champs d'une table, on commence par se connecter à la base de données. Puis on protège la chaine reçue contre l'injection SQL à l'aide de la fonction « quote ». Par la suite on crée une requête permettant de récupérer tous les champs de la table reçue en paramètre. On prépare la requête avant de l'exécuter.

Finalement on récupère les données de la table à l'intérieur d'un objet et on renvoie ces informations à l'aide de « return ».

```php
function getAllFields($table) {
    $dbc = connection();
    $dbc->quote($table);
    $req = "SELECT * FROM $table";

    $requPrep = $dbc->prepare($req);
    $requPrep->execute();
    $data = $requPrep->fetchAll(PDO::FETCH_OBJ);
    $requPrep->closeCursor();
    return $data;
}
```

Récupération d'un champ par son identifiant unique

Pour récupérer un champ par son identifiant, on commence par se connecter à la base de données, et comme précédemment, on protège la chaine reçue contre l'injection SQL avec la fonction « quote ». On crée notre requête, puis on la prépare avant de l'exécuter.

Finalement, on récupère le champ dans un type donnée en paramètre, le paramètre est surchargé, ce qui permet à la fonction d'être plus robuste. Par défaut, on renvoie un objet contenant les informations du champ récupéré.

```php
function getFieldById($id, $table, $type = PDO::FETCH_OBJ) {
    $dbc = connection();
    $dbc->quote($table);
    $req = "SELECT * FROM $table WHERE id=:id";
    $requPrep = $dbc->prepare($req);
    $requPrep->bindParam(':id', $id, PDO::PARAM_INT);
    $requPrep->execute();

    return $requPrep->fetch($type);
}
```

Compter le nombre de champs d'une table

Pour compter les champs d'une table, la procédure est très similaire à la récupération de tous les champs d'une table. On procède de la même manière en se connectant et en protégeant nos paramètres contre l'injection SQL.

Puis on crée notre requête en utilisant la fonction MySQL « COUNT » qui permet de compter les champs donnés en paramètre, dans notre cas, on compte tous les champs de la table. Finalement on prépare la requête, on l'envoie et on récupère les données au format « integer ».

```php
function countFields($table) {
    $dbc = connection();
    $dbc->quote($table);
    $req = "SELECT COUNT(*) FROM $table";

    $requPrep = $dbc->prepare($req);
    $requPrep->execute();

    $number = $requPrep->fetch();
    $requPrep->closeCursor();
    return $number[0];
}
```

Suppression d'un champ par son identifiant

Pour supprimer un champ à l'aide de son identifiant unique (id), on commence par se connecter à la base, puis on protège le paramètre reçu contre l'injection SQL.

On crée notre requête, ou l'on supprime un élément de la table donnée en paramètre où l'identifiant du champ à supprimer est égal à celui reçu en paramètre. Finalement on renvoie le résultat de la requête si tout s'est passé comme prévu.

```php
function deleteFieldById($id, $table) {
    $dbc = connection();
    $dbc->quote($table);
    $req = "DELETE FROM $table WHERE id=:id";

    $requPrep = $dbc->prepare($req);
    $requPrep->bindParam(':id', $id, PDO::PARAM_INT);
    $requPrep->execute();
    $data = $requPrep->fetchAll(PDO::FETCH_OBJ);
    $requPrep->closeCursor();
}
```

Page d'accueil

Affichage des produits les plus populaires
Pour afficher des produits par popularité, le fonctionnement est le suivant :

On utilise une fonction « getMostViewedProducts() » se trouvant dans le « crud_Produits ». La fonction, à l'aide d'une requête MySQL, récupère depuis la base de données la liste des produits triés par nombre de vues ainsi que leur medias. Il est important de récupérer les medias des produits pour pouvoir afficher un aperçu du produit par la suite. Les produits récupérés sont stockés dans un tableau d'objets. On s'assure aussi dans la requête MySQL que le media récupéré soit bien une image grâce au champ « isImage ». On limite aussi le nombre de produits affichés à l'aide d'une constante nommée « NUMBER_OF_TOP_PRODUCTS » qui sera passée en paramètre. Voici la fonction utilisée :

```php
function getMostViewedProducts($nbProductsShown) {
    global $tableProducts;

    $dbc = connection();
    $dbc->quote($tableProducts);
    $req = 'SELECT DISTINCT p.id AS idProduct, p.title as productTitle,
            . 'p.short_desc, MIN(m.src) AS mediaSource,'
            . ' m.isImage, p.view_count, p.is_frontpage '
            . 'FROM products AS p '
            . 'INNER JOIN products_has_medias AS pm ON p.id = pm.id_products '
            . 'INNER JOIN medias AS m ON pm.id_medias = m.id '
            . 'WHERE m.isImage = 1 '
            . 'AND NOW() > availability_date '
            . 'AND NOW() < expiration_date '
            . 'GROUP BY p.title '
            . 'ORDER BY view_count DESC '
            . 'LIMIT ' . $nbProductsShown;

    $requPrep = $dbc->prepare($req); // on prépare notre requête
    $requPrep->execute();
    $data = $requPrep->fetchAll(PDO::FETCH_OBJ);
    $requPrep->closeCursor();

    return $data;
}
```

Pour convertir la liste des produits triés vers un format HTML, on utilise la fonction « structMostViewedProducts() ». Cette fonction va tout d'abord récupérer la liste des produits à l'aide de « getMostViewedProducts() », puis, pour chaque produit, elle va ajouter un élément à la variable « $str ». On remplit la variable « $str » avec le code HTML pour l'affichage de chaque produits et on remplace certains éléments par les éléments récupérés de la base de données, tel que le titre du produit, et une courte description. Après avoir parcouru chaque produit, on renvoie la variable contenant le code HTML.

Voici a quoi pourrait ressembler la fonction « structMostViewedProducts() »

```
function structMostViewedProducts() {
    $products = getMostViewedProducts(NUMBER_OF_TOP_PRODUCTS);
    $str = '';

    foreach ($products as $product) {
        $str.= '<div class="col-xs-12 col-sm-6 col-md-4 col-lg-3">
                    <div class="panel panel-success">
                        <div class="panel-heading">
                            <a href="detail.php?id=' . $product->idProduct . '"></a>
                        </div>
                        <div class="panel-body">
                            <div class="product-container thumbnail">
                                <a href="detail.php?id=' . $product->idProduct . '"></a>
                            </div>
                            <hr/>
                            <div class="panel-footer" style="float: left;">
                                <p>
                                    ' . $product->short_desc . '
                                </p>
                            </div>
                        </div>
                    </div>
                </div>';
    }

    return $str;
}
```

Après avoir créé la fonction « structMostViewedProducts() », il suffit d'afficher le contenu

Voici à quoi pourrait ressembler la partie s'occupant de l'affichage des produits les plus vu dans la page principale du site web :

```
<div class="row">
    <?php
    echo structMostViewedProducts();
    ?>
</div>
```

Affichage des produits recommandés

Le fonctionnement derrière l'affichage des produits recommandés est très similaire à celui de l'affichage des produits les plus vus. Il est le suivant :

On utilise une fonction «getRecommendedProducts() » se trouvant dans le « crud_Produits ». La fonction, à l'aide d'une requête MySQL, récupère depuis la base de données la liste des produits recommandés par les administrateurs ainsi que leur medias. Comme indiqué précédemment, il est important de récupérer les medias des produits pour pouvoir afficher un aperçu du produit par la suite. Les produits récupérés sont stockés dans un tableau d'objets. Un produit recommandé par un administrateur est définit par un champ « is_frontpage » qui est égal à true. On s'assure aussi dans la requête MySQL que le media récupéré soit bien une image grâce au champ « isImage ». Comparé à l'affichage des produits les plus vus on ne limite pas le nombre de produits affichés, les administrateurs du site sont en charge de choisir quel produit mettre en première page.

Voici a quoi pourrait ressembler la fonction « getRecommendedProducts() » :

```php
function getRecommendedProducts() {
    global $tableProducts;

    $dbc = connection();
    $dbc->quote($tableProducts);

    $req = 'SELECT DISTINCT p.id AS idProduct, p.title as productTitle,
            . ' p.short_desc, MIN(m.src) AS mediaSource,'
            . ' m.isImage, p.view_count, p.is_frontpage '
            . 'FROM products AS p '
            . 'INNER JOIN products_has_medias AS pm ON p.id = pm.id_products '
            . 'INNER JOIN medias AS m ON pm.id_medias = m.id '
            . 'WHERE m.isImage = 1 '
            . 'AND is_frontpage=true '
            . 'AND NOW() > availability_date '
            . 'AND NOW() < expiration_date '
            . 'GROUP BY p.title '
            . 'ORDER BY p.title ASC ';

    $requPrep = $dbc->prepare($req);
    $requPrep->execute();
    $data = $requPrep->fetchAll(PDO::FETCH_OBJ);
    $requPrep->closeCursor();

    return $data;
}
```

Comme précédemment, pour convertir la liste des produits triés vers un format HTML, on utilise la fonction « structRecommendedProducts() ». Cette fonction va tout d'abord récupérer la liste des produits à l'aide de « getRecommendedProducts () », puis, pour chaque produit, elle va ajouter un élément à la variable « $str ». On remplit la variable « $str » avec le code HTML pour l'affichage de chaque produits et on remplace certains éléments par les éléments récupérés de la base de données, tel que le titre du produit, et une courte description. Après avoir parcouru chaque produit, on renvoie la variable contenant le code HTML.

Voici a quoi pourrait ressembler la fonction « structRecommendedProducts() » :

```
function structRecommendedProducts() {
    $products = getRecommendedProducts();
    $str = '';

    foreach ($products as $product) {
        $str.= '<div class="col-xs-12 col-sm-6 col-md-4 col-lg-3">
                    <div class="panel panel-success">
                        <div class="panel-heading">
                            <a href="detail.php?id=' . $product->idProduct . '"></a>
                        </div>
                        <div class="panel-body">
                            <div class="product-container thumbnail">
                                <a href="detail.php?id=' . $product->idProduct . '"></a>
                            </div>
                            <hr/>
                            <div class="panel-footer" style="float: left;">
                                <p>
                                    ' . $product->short_desc . '
                                </p>
                            </div>
                        </div>
                    </div>
                </div>';
    }

    return $str;
}
```

Après avoir créé la fonction « structRecommendedProducts () », il suffit d'afficher le contenu

Voici à quoi pourrait ressembler la partie s'occupant de l'affichage des produits recommandés dans la page principale du site web :

```
<div class="row">
    <?php
    echo structRecommendedProducts();
    ?>
</div>
```

Affichage en-tête dynamique

Le fonctionnement de l'en-tête dynamique est simple : On affiche de différents en-têtes en fonction de l'utilisateur connecté.

Pour cela, j'ai créé plusieurs fonctions me permettent de savoir si un utilisateur est connecté ou encore si l'utilisateur connecté est un administrateur.

Pour savoir si un utilisateur est connecté, on regarde si un identifiant a été initialisé dans la session de l'utilisateur. Après avoir effectué le test, on renvoie « true » si l'utilisateur est connecté, et « false » si l'utilisateur n'est actuellement pas connecté.

Voici un exemple du code que pourrait contenir la fonction « isConnected() » :

```
function isConnected() {
    return (isset($_SESSION['id']));
}
```

Pour déterminer si un utilisateur est un administrateur, le principe est le même, mais on teste plusieurs éléments.

Il faut vérifier que l'élément « admin » soit initialisé dans la session de l'utilisateur. Puis on regarde si cet élément est égal à 1. Si ces deux conditions sont respectées, la fonction renvoie « true » pour indiquer que l'utilisateur est bien un administrateur, sinon elle renvoie la valeur « false ». Voici à quoi pourrait ressembler le code permettant de vérifier si l'utilisateur connecté est un administrateur :

```
function isAdmin() {
    if (isset($_SESSION['admin'])) {
        if ($_SESSION['admin'] == 1) {
            $result = true;
        }
        else {
            $result = false;
        }
    }
    else {
        $result = false;
    }
    return ($result);
}
```

Après avoir déterminé le statut actuel de l'utilisateur (déconnecté, connecté, admin), on va afficher un en-tête différent pour chaque statut.

Si l'utilisateur est déconnecté, on affiche un bouton pour se connecter. Pour plus d'informations sur la connexion de l'utilisateur, se référer à « Identification utilisateur>Connexion utilisateur ».

Si l'utilisateur est connecté mais n'est pas un administrateur, on lui affiche un menu déroulant portant son nom d'utilisateur et lui permettant de se déconnecter. Pour plus d'informations sur la déconnexion de l'utilisateur, se référer à « Identification utilisateur>Déconnexion utilisateur ».

Si l'utilisateur est un administrateur, on affiche le même menu déroulant que l'utilisateur standard, mais avec un lien supplémentaire pour rediriger l'administrateur vers l'ajout de produits (cf. doc. Administration>Ajout produit). Voici quelques lignes de code représentant le fonctionnement de la fonction « getHeader() » :

```
function getHeader(){
    if (isConnected()){
        //En tête utilisateur ici (Connecté)
        if (isAdmin(){
            //En tête administrateur ici (Administrateur)
        }
    }
    else{
        //En tête visiteur ici (Déconnecté)
    }
}
```

Détail produit

Affichage du détail produit

L'affichage détaillé du produit se fait en plusieurs parties :

On commence par envoyer un paramètre en « GET » contenant l'identifiant du produit à afficher. Puis grâce à cet identifiant, on va chercher à l'aide d'une requête MySQL, les informations détaillée du produit ainsi que sa marque et ses médias. Finalement, on affiche les informations récupérées à l'aide d'une fonction permettant de structurer la page du produit.

Si l'identifiant du produit récupéré n'est pas un produit existant on renvoie l'utilisateur à la page d'accueil. Voici le code pour la redirection :

```php
if ($product == NULL) {
    header('Location: index.php');
}
```

Voici la fonction « getProductDetailledById() » qui permet de récupérer tous les champs nécessaire à l'affichage du détail produit :

```php
function getProductDetailsById($id) {
    global $tableProducts;

    $dbc = connection();
    $dbc->quote($tableProducts);
    $req = "SELECT p.id AS idProduit, p.title, p.short_desc,"
            . " p.long_desc, p.is_frontpage,"
            . " p.availability_date, p.expiration_date, p.view_count,"
            . " p.id_brand, b.id AS idBrand, b.name AS brandName, "
            . "(NOW() < availability_date OR NOW() > expiration_date) AS isExpired"
            . " FROM $tableProducts AS p"
            . " INNER JOIN brands as b ON p.id_brand = b.id"
            . " WHERE p.id = $id";


    $requPrep = $dbc->prepare($req);
    $requPrep->execute();
    $data = $requPrep->fetch(PDO::FETCH_OBJ);
    $requPrep->closeCursor();

    return $data;
}
```

Dans la requête présente ci-dessus, on récupère un champ « isExpired » qui nous indique si le produit est expiré. Pour vérifier si un produit est expiré, on regarde si la date actuelle se trouve entre la date de disponibilité et la date d'expiration du produit. On affiche cette information à l'utilisateur accordement suivit d'un icone pour faciliter la visibilité.

Gestion des médias du détail produit

Les medias des produits peuvent-être résumés en deux catégories :

1. Les images (.png, .gif, .jpg, .jpeg)
2. Les autres (.pdf, .docx, .xls etc...)

En dépendant de la catégorie du media, on les affiche dans le détail du produit à deux endroits différents. Les images sont disposées à l'intérieur d'un « carousel » à côté de la description du produit. Des flèches sont disposées à droite et à gauche du « carousel » pour permettre à l'utilisateur de naviguer entre les images et des indicateurs de positions sont disposés au centre de la partie inférieure de l'image.

Les medias classés dans la catégorie « autres » sont affichés à l'aide d'une « medialist » se trouvant en bas de la page et disposent chacun d'un lien de téléchargement.

Le détail produit possède malgré cela de petites subtilités. En effet, si le produit possède une seule image, le « carousel » ne possèdera ni de contrôles, ni d'indicateurs de position. Les différentes images disposées à l'intérieur du « carousel » doivent aussi toutes faire la même hauteur, tout en gardant les proportions initiales de l'image. On doit donc pouvoir garder une mise en page homogène. Pour cela, les images stockées à l'intérieur du carousel sont toutes stockées dans un « container » intermédiaire à hauteur fixe.

Pour différentier les medias, un champ dans la table medias nommé « isImage » sert à décrire si le produit est une image. Pour trier les medias on va donc procéder de la manière suivante :

On récupère chaque media, puis, si le media est une image, on le stocke dans un tableau d'images, sinon, on le stocke dans un tableau d'autres fichiers. Voici a quoi pourrait ressembler le code permettant de séparer les images en deux catégories :

```
foreach ($medias as $media) {
    if ($media->isImage) {
        $i++;
        $arrayImage[$i] = $media->mediaSource;
    } else {
        $j++;
        $arrayOthers[$j] = $media->mediaSource;
    }
}
```

On va par la suite parcourir les deux tableaux et créer le code HTML propre à chaque medias et contenant les informations récupérées. Voici un exemple de code permettant de remplir une variable avec le contenu des medias « autres » :

```
foreach ($arrayOthers as $other) {
    $filename = substr($other, strlen(OTHER_FOLDER));
    $otherMedia .= 'Nom media : '.$other->name.' etc...';
}
```

On procède de la même manière pour les images à l'exception que l'on retire les contrôles du « carousel » si le produit ne possède qu'une seule image. En supprimant les contrôles du carousel on évite d'induire en erreur l'utilisateur.

Notez que l'on crée un nom pour le produit qui servira de lien de téléchargement par la suite. Le nom est créé en soustrayant la longueur de l'arborescence du dossier où sont stockés les medias au chemin complet.

Par exemple : « up-content/others/test_2.pdf ». On soustrait « up-content/others/ » à notre première chaine de caractère et on récupère le nom final du fichier : « test_2.pdf ». C'est ce nom de fichier qui sera visible en tant que lien de téléchargement dans le conteneur à fichiers téléchargeables.

Ajout de vues

Pour ajouter des vues à un produit, on lance une fonction à chaque fois que la page est lancée. Cette fonction va récupérer le nombre de vues d'un produit grâce à son identifiant unique et va ajouter une vue au total grâce à une requête « UPDATE ». Après avoir récupéré le nombre de vues d'un produit, on affiche le nombre de vues en bas de la page du détail produit à l'aide d'une fonction de structure.

Voici une fonction permettant d'ajouter une vue à un produit en fonction de son identifiant :

```php
function addViewById($id) {
    global $tableProducts;

    $dbc = connection();
    $dbc->quote($tableProducts);
    $dbc->quote($id);
    $req = "UPDATE $tableProducts SET view_count = view_count+1 WHERE id = $id";
    $requPrep = $dbc->prepare($req);
    $requPrep->execute();
    $requPrep->closeCursor();
}
```

Voici la fonction permettant de structurer les vues vers un format HTML :

```php
function structViewCount($id) {
    $product = getProductById($id);
    $str = '<span>Nombre de vues : '
             . $product->view_count .
             ' <span class="glyphicon glyphicon-eye-open"></span></span>';

    return $str;
}
```

L'ajout de vue s'effectue à chaque fois qu'un utilisateur charge ou recharge la page, une amélioration envisageable est de limiter le nombre de vues ajoutées à une seule par personne. Pour plus d'informations se rendre dans la catégorie « Améliorations envisageables » de la documentation technique.

Recherches

Liste déroulante des catégories

Un bouton dépliant une liste déroulante se trouve en dessous de l'en-tête et permet à l'utilisateur à accéder rapidement à la recherche par catégories du site web. Le menu est composé d'un bouton et d'une liste HTML cachée, un script JavaScript va par la suite afficher chaque élément caché de la liste quand on clique sur le bouton. La liste est créée dynamiquement avec la liste des mots-clefs provenant la base de données et triée par ordre alphabétique.

Voici la fonction s'occupant de l'affichage des mots clefs à l'intérieur de la liste déroulante.

```php
function structKeywordsList() {
    $keywords = getAllKeywordsSorted();
    $str = '';

    //On affiche un lien pour chaque keywords de la base
    foreach ($keywords as $keyword) {
        $str.= '<a href="search.php?querryKW=' . $keyword->name . '" class="list-group-item">
                        ' . strtoupper($keyword->name) . '
                    </a>';
    }

    return $str;
}
```

On ajoute aussi un lien « href » vers la page de recherche pour permettre la recherche par mots-clefs. Pour plus d'informations sur la recherche par mot clefs, se rendre dans la catégorie « Recherche par mot-clef » de la documentation technique.

Recherche par mot-clef

Pour la recherche par mot-clef, on commence tout d'abord par récupérer la requête de recherche de l'utilisateur. Dans le cas de la recherche par mot-clef, un paramètre envoyé en « get » contient le nom de la catégorie à rechercher (keyword).

On va par la suite exécuter une fonction qui va, à l'aide d'une requête MySQL, récupérer la liste des produits possédant le même mot-clef que la requête de l'utilisateur. Par exemple : L'utilisateur clique sur « ALIMENTATIONS », on va rechercher la liste des produits liés au mot-clef « ALIMENTATIONS » et on les affiche sur la page.

Voici la fonction utilisée pour récupérer les produits recherchés à l'aide d'un mot-clef.

```php
function searchForProductWithKeywords($querry) {
    global $tableProducts;
    $dbc = connection();
    $dbc->quote($tableProducts);
    $req = 'SELECT DISTINCT p.id AS idProduct, p.title as productTitle, '
            . 'p.short_desc, p.view_count, p.is_frontpage, b.name AS brandName, '
            . 'k.name AS keywordName '
            . 'FROM ' . $tableProducts . ' AS p '
            . 'INNER JOIN brands AS b ON p.id_brand = b.id '
            . 'INNER JOIN products_has_keywords AS pk ON p.id = pk.id_products '
            . 'INNER JOIN keywords AS k ON pk.id_keywords = k.id '
            . 'WHERE k.name LIKE "%' . $querry . '%" '
            . 'GROUP BY p.title '
            . 'ORDER BY p.title DESC';

    $requPrep = $dbc->prepare($req);
    $requPrep->execute();
    $data = $requPrep->fetchAll(PDO::FETCH_OBJ);
    $requPrep->closeCursor();

    return $data;
}
```

Une des subtilités de cette procédure est le fait que l'on ne peut pas récupérer en une seule requête les medias du produit et les produits liés aux keywords. On va donc séparer cette procédure en deux parties, et créer une fonction pour structurer la liste des produits recherchés par mot-clef.

Une autre subtilité est l'affichage d'un aperçu du produit en image, en effet, un produit possède différents type de médias tous confondus. Pour résoudre ce problème, on commence par vérifier le type du media avec son champ « isImage », on récupère la totalité des images du produit dans un nouveau tableau que l'on va parcourir pour récupérer la première image du produit.

Voici comment se passe le tri des médias des produits recherchés par mot-clef :

```
foreach ($products as $product) {
    //On récupère la première image du produit
    $medias = getProductMediasById($product->idProduct);
    $imgProduct = '';
    foreach ($medias as $media) {
        if ($media->isImage) {
            $imgProduct[] = $media->mediaSource;
        }
    }
```

Après avoir trié les medias, il ne nous reste plus qu'à ajouter « $imgProduct[0] » dans une balise « img » pour que la première image du produit s'affiche.

Actuellement, la recherche de produit par mot-clef ne possède pas de pagination ou de limitation du nombre de produits par page, pour plus d'informations sur les amélioration envisageables du site web, se rendre dans la catégorie « Amélioration envisageables » de la documentation.

Recherche multicritères

Pour la recherche multicritères, on commence tout d'abord par récupérer la requête de recherche de l'utilisateur. Dans le cas de la recherche multicritères, un paramètre envoyé en « get » contient les critères entrés par l'utilisateur dans la barre de recherche.

On va par la suite exécuter une fonction qui va, à l'aide d'une requête MySQL, récupérer la liste des produits et les medias des produits ou les critères entrés par l'utilisateur sont présent dans un des différents champs des produits (titre, description courte & longue, marque). Pour réaliser cela, on va concaténer les différents champs et utiliser la commande « LIKE » pour comparer ces derniers à la requête de l'utilisateur.

Voici la fonction utilisée pour récupérer les produits recherchés à l'aide de la barre de recherche multicritères :

```
function searchForProduct($querry) {
    global $tableProducts;

    $dbc = connection();
    $dbc->quote($tableProducts);
    $req = 'SELECT DISTINCT p.id AS idProduct, p.title as productTitle,'
            . ' p.short_desc, MIN(m.src) AS mediaSource,'
            . ' m.isImage, p.view_count, p.is_frontpage, b.name AS brandName '
            . 'FROM ' . $tableProducts . ' AS p '
            . 'INNER JOIN products_has_medias AS pm ON p.id = pm.id_products '
            . 'INNER JOIN medias AS m ON pm.id_medias = m.id '
            . 'INNER JOIN brands AS b ON p.id_brand = b.id '
            . 'WHERE Concat(p.title, p.short_desc, p.long_desc, b.name) '
            . 'LIKE "%' . $querry . '%" '
            . 'AND m.isImage = 1 GROUP BY p.title ORDER BY p.title DESC';

    $requPrep = $dbc->prepare($req);
    $requPrep->execute();
    $data = $requPrep->fetchAll(PDO::FETCH_OBJ);
    $requPrep->closeCursor();

    return $data;
}
```

Après avoir récupéré la liste des produits avec leur medias et leurs marques respectives, on va les structurer et les convertir en code HTML à l'aide d'une fonction. Comparé à la recherche par mot-clef, il est possible de récupérer les produits et leurs medias en une seule requête, ce qui diminue grandement le nombre de contrôles à effectuer.

Identification utilisateur

Connexion utilisateur

Le fonctionnement de la connexion utilisateur est plutôt simple. On commence par récupérer les informations

Déconnexion utilisateur

Administration

Ajout produit

Modification produit

Gestion des medias du produit

Fonctionnalités supplémentaires

Site web mobile

Tests

Conclusion

Bilan, améliorations envisageables

Comparaison analyse et réalisation

Comparaison journal et planning

(Mes satisfactions, ce que j'ai appris)

Bibliographie