Projet "Bob-Project"

LAFON Sylvain, MAINGRET François et LEVASSEUR Thomas

25 novembre 2011

Table des matières

Première partie
Introduction

L'objectif de ce projet était de réaliser un site web pour un magasin de bricolage, Brico-Bob. Ce site devra gérer plusieurs aspects : une partie privée et une partie publique :

- La partie publique sera accessible à n'importe qui, et comprendra, entre autres, la possibilité de visionner des produits, d'en rechercher et d'en commenter.

- La partie privée sera réservée aux administrateurs du site et comprendra un panneau permettant de gérer tout le contenu du site, des produits, images, catégories...

Deuxième partie

Bob-Project

1 Objectifs

L'objectif de notre site est de proposer à la vente, comme à la location, des articles de bricolage pour le grand public. Le site devra donc être assez ergonomique et attrayant visuellement afin de ne pas repousser les visiteurs.

Voila les points importants sur lesquels nous avons fait attention :

- Un temps de chargement des pages optimisé. En effet on estime le seuil d'acceptation de chargement d'un site à deux secondes (selon plus de 47% des internautes sondés). Au delà, vous commencez à perdre des visiteurs. (source : Marketing Management par P. Kotler).

- Proposer une description des produits assez détaillée : si le client ne trouve pas assez d'informations sur un produit de notre site, il ira chercher ces informations chez un concurrent, et à moins que notre prix soit significativement plus bas, il y a de grandes chances qu'il achète chez ce concurrent.

- Il faudra utiliser le plus de termes simples à comprendre par l'utilisateur plutôt que des termes techniques barbares. Notre site permet donc d'écrire de longues descriptions, mais cette responsabilité incombe plutôt à la société Brico-Bob.

- Donner des informations sur la société et sur le service client, ainsi que sur les administrateurs du site. Le client veut savoir à qui il a affaire et sera plus enclin à acheter si il sait qu'il peut contacter quelqu'un ensuite. Le client sera mis en confiance, ce qui est vraiment important lorsque son argent est en jeu.

Ces informations seront donc accessibles depuis n'importe quelle page du site grâce au menu :

– Ne pas trop s'introduire dans la vie privée du client : nous n'avons demandé au client que des informations non personnelles lors de son inscription : son pseudo et un mot de passe. Il n'a pas besoin de donner son nom ni son adresse tant qu'il ne valide pas une commande. Cela lui permettra de ne pas se sentir "tracké" et de visiter librement le site.

– Un moteur de recherche bien réalisé. Si le client sait exactement ce qu'il veut lorsqu'il va sur le site, il ira chercher le nom du produit qu'il a en tête dans la barre de recherche. Si le moteur de recherche n'est pas bien conçu, il risque de penser que nous n'avons pas le produit qu'il cherche et il ira donc voir ailleurs. Si il n'a qu'une idée très vague, il y a alors des chances qu'il se tourne vers le moteur de recherche avancée. Cela nous permettra de cerner ses critères de prix par exemple.

– Mettre en évidence le produit en mettant une image assez grande : trop de site proposent encore des images ridiculement petites, ce qui peut décourager le client d'acheter, car il ne peut pas bien voir le produit et peut même penser qu'il y a des risques de "tromperie" sur la marchandise.

– Une navigation aisée, c'est-à-dire des catégories bien organisées. Bien que notre site permette d'afficher à la fois des produits et des catégories sur une même page, cela sera à éviter dans le cas général. Enfin, il faudra veiller à ne pas créer de catégories vides, ce qui prendrait de la place pour rien et serait une source de frustration pour le client (quoi de plus désagréable que de trouver une catégorie qui correspond parfaitement à ce que l'on recherche et se rendre compte qu'elle est vide)

– Mettre l'accent sur les produits : le but d'un site de e-commerce est de vendre, et nous avons donc fait attention de mettre l'accent sur les produits. Le design est assez sobre, et les autres éléments du site n'empiètent pas sur les produits. Il y a également sur la page d'accueil des "coups-de-cœur" permettant de stimuler un achat imprévu chez le client.

2 Logique et Organisation

2.1 Fichiers

2.1.1 header.php

Ce fichier va insérer la bibliothèque Smarty et les déclarations de classes, puis il va déclarer des constantes et crée quelques fonctions.

2.1.2 index.php

Ce fichier va tout d'abord appeller header.php puis va créer l'objet Bob et smarty.
Il va ensuite analyser la requete pour savoir quel template il va appeller pour fabriquer la page et va effectuer les actions qu'on lui demande de faire.
Il va ensuite passer à Smarty plusieurs variables puis générer la page.

2.1.3 les fichiers de /classes et de /Smarty

Ces fichiers contiennent les classes qui contiennent les infos de la base de données et qui la manipule.

2.1.4 les fichiers de /templates

Ils permettent d'afficher une page en utilisant les variables qu'on lui prete, il y a deux style de templates :

- les fichiers modèle : ils se situent dans /templates/modele
  - main.tpl : Contient ce que toute page va contenir, il va appeller les autres fichiers templates de modèle
  - menu.tpl : Contient le menu du site
  - entete.tpl : Contient les informations sur la page ($< head >$)
  - espace_membre.tpl : Contient l'espace membre

- le fichier appelé pour la page : ils se situent dans /templates.
  Il y en a un par page

2.1.5 les fichiers de /css

Ils permettent de coder le design du site, ils se situent dans /css. Ces fichiers sont séparés en plusieurs fichiers. On préférera en mettre un général puis un par page.

2.1.6 En somme

Seul index.php permet de générer une page, il va commencer par appeller toutes les ressources à l'aide de header.php (qui inclut les classes de /classes et /Smarty), il va ensuite regarder le type de la requete :

- Est-ce pour le panneau d'admin, une image, une information ?
- Est-ce une page ou une action que l'on nous demande ?

En fonction de cela nous allons faire telle ou telle action (méthodes de la classe Bob) puis choisir tel ou tel template à appeller.

Une fois l'action faite et le template choisi, Smarty (l'objet) va prendre certaines variables (contenues dans Bob, ses attributs ainsi qu'un eventuel message) puis va utiliser le template pour générer une page.

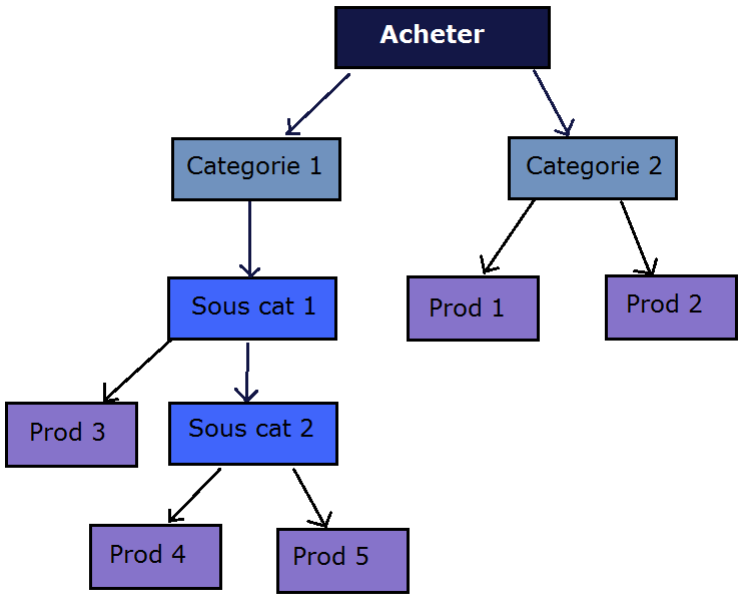
Le template appellé est une extension du template modèle templates/modele/main.tpl (divisé en plusieurs templates modèles), cela permet de garder une meme forme pour chaque page : seul le contenu, certains scripts sera modifié d'une page à l'autre !

La page html ainsi générée fera appel à certaines feuilles de design de /css et à certains scripts /js

2.2 Catégories

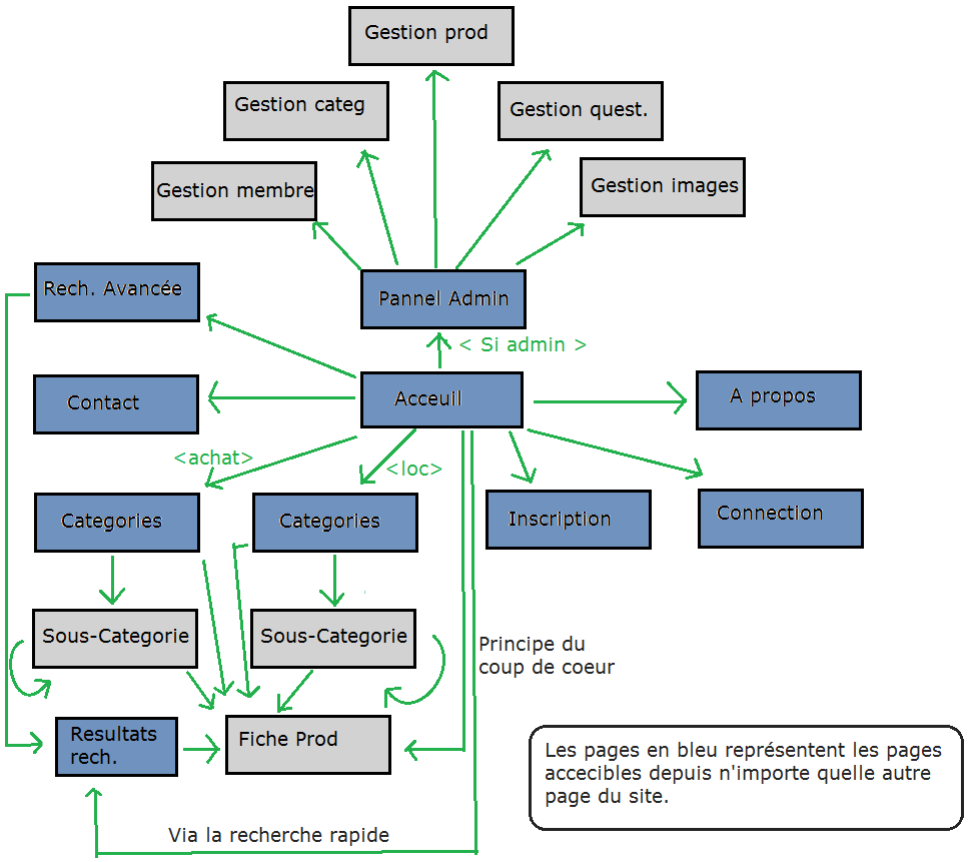
Comme nous l'avions dit dans les objectifs du Site, la navigation se fait par un système de catégories qui peuvent contenir d'autres catégories et d'autres produits.
(c'est à l'administrateur à faire attention à ne pas de creer de catégories vides, pour des raisons d'ergonomie, il est aussi conseillé d'éviter de mettre catégories et produits ensemble)

Tout ceci se résume à un simple schéma :



2.3 Pages

Plutot que de parler une à une de chauqe page 'logique' du site, voici un schèma plus explicite :



3 Architecture des pages

3.1 Un modèle

L'architecture des pages est définie par les templates de modèle. Le modèle 'principal', main.tpl contient donc les éléments principaux de la page et le design leur donne leur position et les 'décore'.

Nous avons utilisé et sur-utilisé la balise $< div >$ afin de pouvoir manipuler les elements plus aisément avec le design. Dans le modèle et dans le contenu.

3.2 Le résultat

Une page va donc contenir :
- Un header contenant :
  - Le logo
  - l'espace membre et recherche
- Un corps contenant :
  - Le Menu : "Acheter, Louer, Contact et About"
  - Le Contenu : seul element qui change (visuellement)
- Un pied de page contenant :
  - Le Sitemap (non codé)
  - Les droits d'utilisation/Les crédits
  - Les partenaires du site/de Brico-Bob

4 Technologies utilisées

4.1 Support du code

Pour le projet nous avons utilisé plusieurs supports :

Github : Il s'agit d'un système de partage de projets gratuit -si on accepte de les mettre open-source- très pratique.

Notepad++ : Il s'agit de notre éditeur de texte favori, nous l'avons tous utilisé

Wamp/Lampp : Il s'agit d'un émulateur de serveur Apache/MySql/Php pour pouvoir tester le site en local.

Apache : Il émule le serveur en lui meme, il contient notemment une interface "phpMyAdmin" et "Wamp" (pour les différents projets)

MySql : Il émule la base de données

Php : Le programme qui, suivant la requete, interpretera la bonne page php.

The GIMP : C'est un outil pour manipuler les images très puissant : beaucoup d'éléments ont été réalisées grace à lui.

4.2 Les langages de programmation

Nous avons utilisé, comme pour la plupart des sites, les langages :

HTML : Il défini les éléments de la page dans les templates

CSS : Il gère le design du site, nous avons séparé ses fichiers en plusieurs parties (Au départ il s'incluait tous, mais nous avons fait en sorte que Smarty n'appelle que ceux qui nous importe)

PHP : Il permet de creer la page HTML suivant la requête qu'on lui donne.

SQL : Il est utilisé par Php pour manipuler la base de données, laquelle contient tous les éléments du site, lesquels sont stockés au chargement de la page dans des objets (Php)

JavaScript : Il permet de faire "Bouger la page" chez le client, sans avoir besoin d'envoyer de requête Php.
Cela permet, par exemple, de faire un premier contrôle local des données que le client veut envoyer mais cela permet aussi, de rendre certains elements plus ergonomiques, par exemple pour la notation des produits.

4.3 Les bibliothèques utilisées

Nous avons utilisé deux bibliothèques :

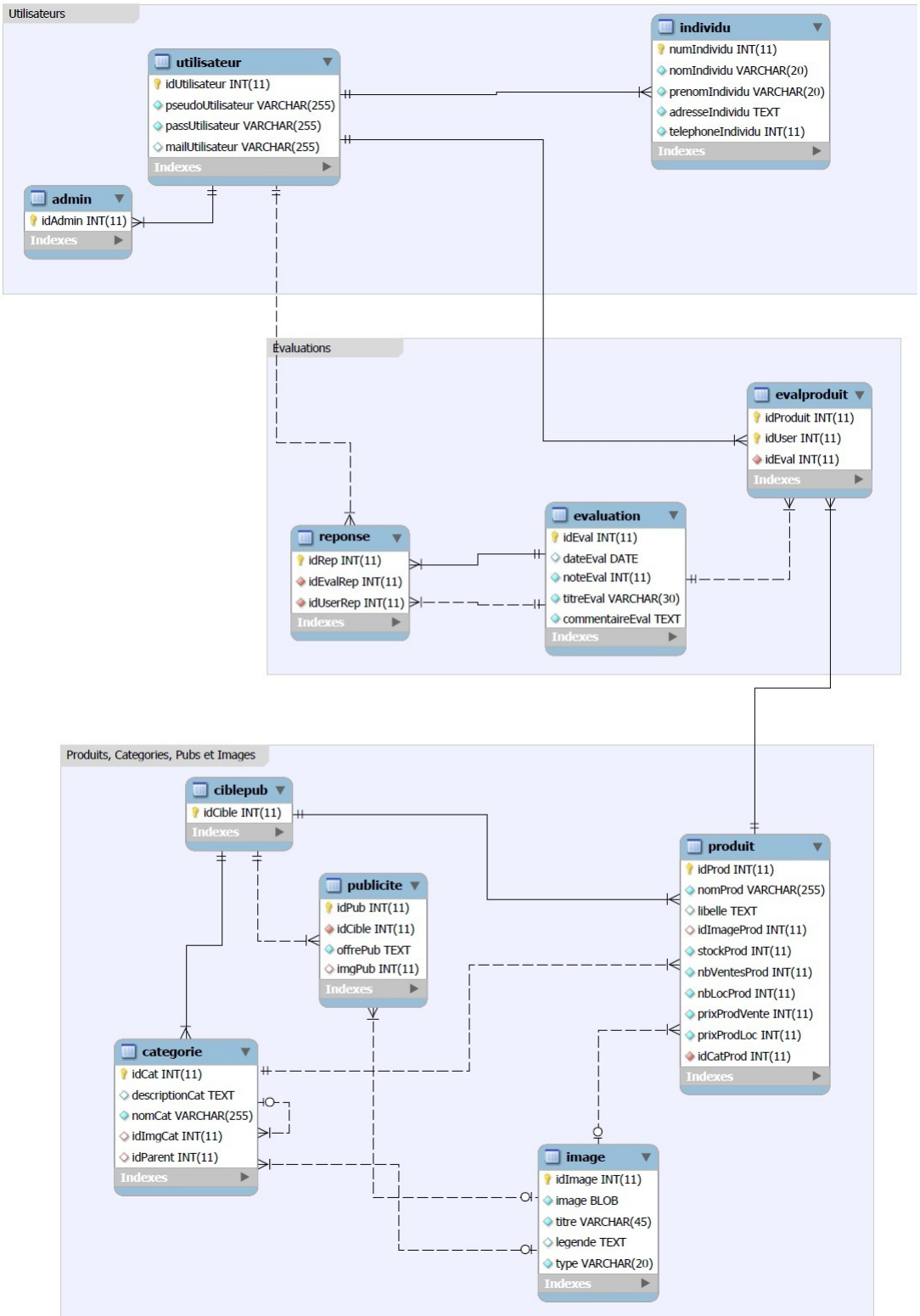
AJAX : Exploité par le JavaScript, elle permet de récuperer une autre page et de traiter ces données alors que la page ayant appelé AJAX est encore active.
Elle permet, par exemple, alors que la page d'inscription est affichée, de demander au serveur si tel ou tel pseudo est utilisé ou non.

Smarty : C'est le moteur de templates utilisé, il permet de séparer traitement des données et mise en page : Le code Php qui l'appelle va commencer par analyser la requête, et suivant son analyse va appeler tel ou tel template en lui 'passant' des variables qu'il pourra alors utiliser pour l'affichage.

5 Base de données

Elle contient toutes les infos du site, notemment :
- Les membres
- Les catégories
- Les produits
- Les commentaires
- Et meme les images !

Voici un schéma relationnel de notre base (généré à l'aide de "MySql Workbench")


Utilisateurs
utilisateur
idUtilisateur INT(11)
pseudoUtilisateur VARCHAR(255)
passUtilisateur VARCHAR(255)
mailUtilisateur VARCHAR(255)
Indexes
individu
numIndividu INT(11)
nomIndividu VARCHAR(20)
prenomIndividu VARCHAR(20)
adresseIndividu TEXT
telephoneIndividu INT(11)
Indexes
admin
idAdmin INT(11)
Indexes
Évaluations
evalproduit
idProduit INT(11)
idUser INT(11)
idEval INT(11)
Indexes
reponse
idRep INT(11)
idEvalRep INT(11)
idUserRep INT(11)
Indexes
evaluation
idEval INT(11)
dateEval DATE
noteEval INT(11)
titreEval VARCHAR(30)
commentaireEval TEXT
Indexes
Produits, Categories, Pubs et Images
ciblepub
idCible INT(11)
Indexes
publicite
idPub INT(11)
idCible INT(11)
offrePub TEXT
imgPub INT(11)
Indexes
produit
idProd INT(11)
nomProd VARCHAR(255)
libelle TEXT
idImageProd INT(11)
stockProd INT(11)
nbVentesProd INT(11)
nbLocProd INT(11)
prixProdVente INT(11)
prixProdLoc INT(11)
idCatProd INT(11)
Indexes
categorie
idCat INT(11)
descriptionCat TEXT
nomCat VARCHAR(255)
idImgCat INT(11)
idParent INT(11)
Indexes
image
idImage INT(11)
image BLOB
titre VARCHAR(45)
legende TEXT
type VARCHAR(20)
Indexes

6 Les sessions

7 Portabilité

8 Flexibilité

Troisième partie
Manuel

Quatrième partie
Conclusion