Notions abordées à travers ce TP

Mise en place d'une base de données
Mise en place de l'arborescence du site
Mise en place de l'architecture

PROJET – PHP / MYSQL

RÉALISATION D'UN SITE DE PARTAGE DE RECETTES DE CUISINE

OBJECTIFS

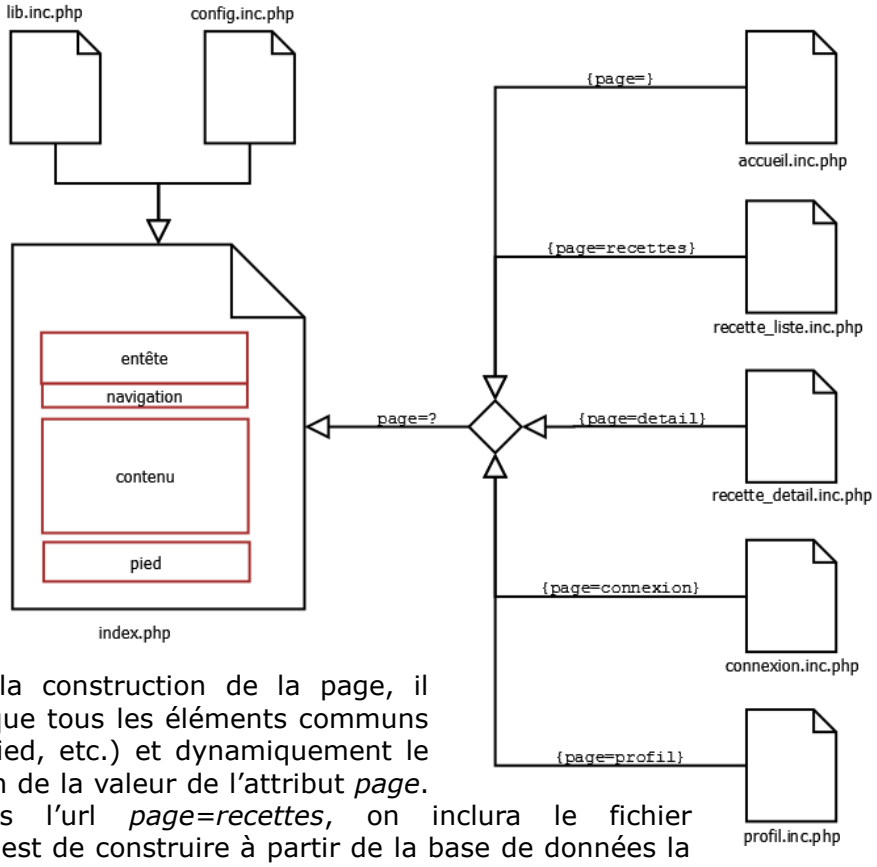
Mise en place d'une base de données
Mise en place de l'arborescence du site
Mise en place de l'architecture

PREREQUIS

La base de données qui reflète le schéma de données du projet doit être créée et les règles d'intégrité référentielle doivent être prises en compte. Pour cela, vous utiliserez le script à votre disposition.

EXERCICE 1 : MISE EN PLACE DE L'ARCHITECTURE

Avant d'aller plus loin dans la programmation des fonctionnalités du site web, vous allez mettre en place l'architecture suivante :



Dans cette architecture, la page est construite dynamiquement à partir de plusieurs éléments qui forment le squelette de la page. Cette structure repose sur la valeur d'une variable nommée *page* qui définit la page à afficher. Concrètement, lors de la construction de la page, il s'agira d'inclure de manière statique tous les éléments communs (lib, config, entête, navigation, pied, etc.) et dynamiquement le fichier "-------.inc.php" en fonction de la valeur de l'attribut *page*. Par exemple, si l'on a dans l'url *page=recettes*, on inclura le fichier *liste_recettes.inc.php* dont le rôle est de construire à partir de la base de données la liste des recettes et de l'afficher.

DESCRIPTION DE L'ARCHITECTURE

L'architecture que vous allez mettre en place va vous permettre de créer de nouvelles pages et de les intégrer facilement à votre site. Pour cela, vous allez devoir décomposer la création de la page selon les éléments suivants :

- L'**entête** : qui contiendra la bannière du site ;
- La **barre de navigation** : qui contiendra le menu de navigation ;
- Un bloc de **contenu principal** : pour le contenu qui caractérise la page ;
- Le **pied de page** : qui contiendra les informations de contact et des liens vers le sitemap et les mentions légales ;
- Et, un **layout** : qui définit comment assembler les éléments précédents.

Dans cette architecture, la page est construite dynamiquement à partir d'un fichier qui gère la structure (le layout) et d'un ensemble de fichiers qui génèrent les différents contenus (entête, barre de navigation, contenu principal, etc.).

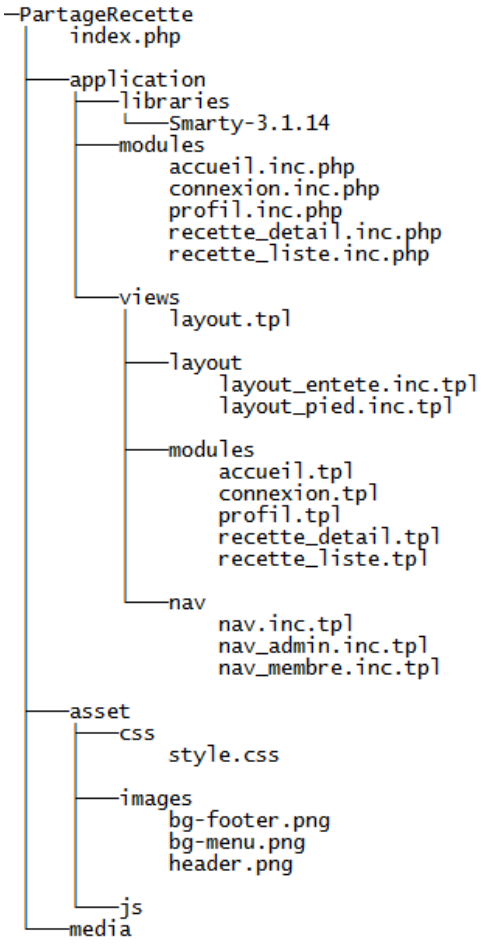
Le travail à réaliser peut être décomposé en quatre étapes :

1. Mise en place de l'architecture
2. Création des contenus
3. Création des vues
4. Assemblage de la page

TRAVAIL A REALISER

ÉTAPE 1 – 1. MISE EN PLACE DE L'ARCHITECTURE

Dans un premier temps, vous devrez avoir créé l'arborescence donnée ci-dessous sans vous préoccuper du contenu des fichiers ; créez les dossiers et ajoutez-y des fichiers vides.



application - contient l'ensemble des éléments qui sont interprétés côté serveur i.e. les scripts PHP – que ce soit pour gérer les contenus ou les afficher.

- **application/libraries -** contient les scripts tiers comme par exemple la librairie Smarty.
- **application/modules -** contient les scripts PHP qui créent les contenus du site.
- **application/views -** contient les vues utilisées pour l'affichage des données. Les vues sont classées en fonction de leur type.

asset - contient les éléments interprétés côté client.

- **asset/css -** contient les feuilles de style du site (screen, print, etc.).
- **asset/images -** contient les images utilisées dans les déclarations de styles (bandeau, fond de page, boutons, etc.).
- **asset/js -** contient les scripts JavaScript nécessaire au fonctionnement du site, y compris les scripts tiers (jquery, mootols, interpréteur LESS, etc.).

media - contient les contenus multimédia utilisés par votre site (illustrations recettes, avatar, etc.). Nous verrons l'organisation de ce répertoire plus tard.

ÉTAPE 2 – CREATION DES CONTENUS

Une fois l'arborescence en place, vous allez créer différents contenus. Cette étape est indispensable car elle va vous permettre de disposer d'un jeu de test pour contrôler que votre architecture est conforme à ce que l'on vous demande de mettre en place.

Contenus secondaires

On entend par contenus secondaires, l'ensemble des informations affichées par le site qui sont indépendantes du contexte de navigation. Par exemple, lorsque l'utilisateur demande la page *recette* : l'entête, le nom du site ou encore le pied de page constituent des informations secondaires. Ces informations sont statiques et sont donc définies directement dans les vues du répertoire *applications/views*.

Éditez les fichiers *layout_entete.inc.tpl*, *layout_pied.inc.tpl* et *nav.inc.tpl* et ajoutez-y du contenu. Tachez de définir un contenu aussi évocateur que possible. Évitez de mettre *blabla* dans tous vos fichiers, cela ne serait pas très pertinent.

Par exemple, le contenu du fichier *layout_entete.inc.tpl* pourrait être

```
<h1>ICI LE NOM DU SITE</h1>
<p>mon slogan</p>
```

Et celui de *nav.inc.tpl*

```
<nav>
    <a href="#">Accueil</a>
    <a href="#">Les Recettes</a>
</nav>
```

Contenus principaux

Le contenu principal de la page est le contenu délivré dans le cadre du contexte de navigation. Ces données sont produites de manières dynamiques (accès à la base de données, analyse de la requête, etc.). Elles sont produites à partir des scripts qui sont dans le répertoire *application/modules* et affichée par le biais des vues correspondantes qui sont dans le répertoire *applications/views/modules*. Attention, outre l'extension qui diffère, ces fichiers doivent porter strictement le même nom. Par exemple, si l'utilisateur demande la page *detail*, c'est le module *recette_detail.inc.php* qui va être appelé pour produire les données, et *recette_detail.tpl* pour la vue. Dans un premier temps, nous ne nous intéresserons qu'aux scripts PHP qui produisent les données.

Éditez le fichier *accueil.inc.php* et écrivez le script PHP qui génère un tableau $*alea* de trois entiers tirés aléatoirement entre 0 et 100. Vous pouvez utiliser la fonction *rand()* à cet effet. Votre script affectera ensuite ce tableau à un tableau nommé *$data* sous la clé *alea*.

```
$data['alea'] = $alea;
```

Éditez le fichier *profil.inc.php* et ecrivez le script PHP qui affecte à un tableau *$user* la valeur <votre nom> pour la clé *nom*, <votre prénom> pour la clé *prenom*. Comme précédemment, votre script affectera ensuite ce tableau à un tableau nommé *$data* sous la clé *user*.

Laissez vide les fichiers *connexion.inc.php*, *recette_detail.inc.php* et *recette_liste.inc.php*. Vous vous occuperez de ces fichiers plus tard.

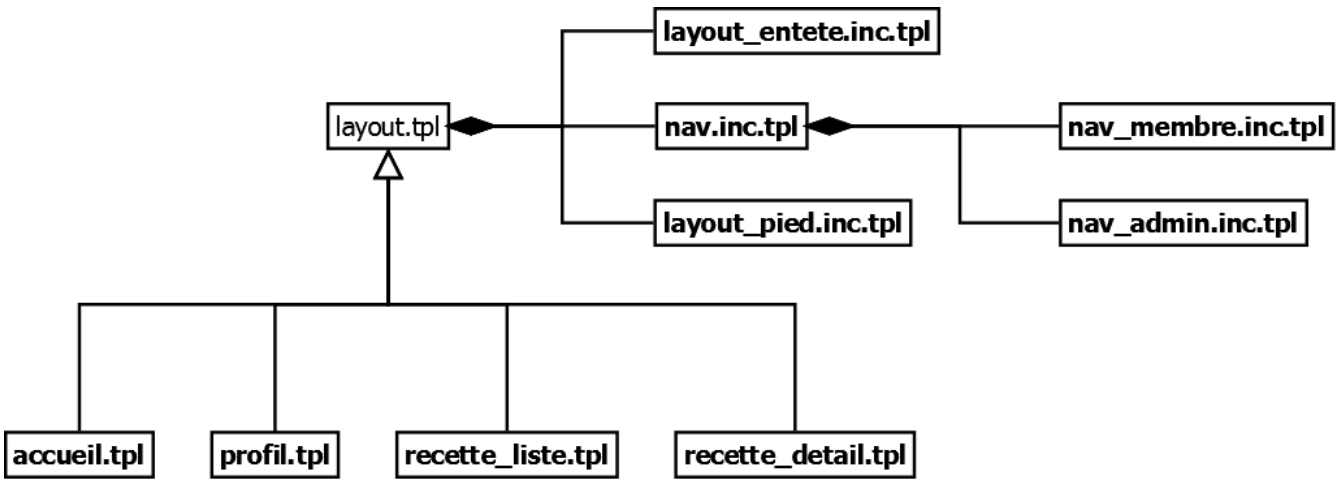
ETAPE 3 – CREATION DES VUES

Maintenant que les contenus principaux et secondaires sont en place, vous allez mettre en place l'architecture de la vue. Cette architecture va utiliser les capacités de Smarty à inclure et surcharger des gabarits. Nous avons volontairement distingué deux types de fichiers dans le répertoire *applications/views* : les fichiers dont l'extension est *.tpl* et les fichiers dont l'extension est *.inc.tpl*. Les fichiers *.tpl* sont ceux qui seront appelés pour faire le rendu final de la page et les fichiers *.inc.tpl* sont ceux qui sont inclus à différentes étapes du processus pour générer la vue finale. Par exemple, si l'utilisateur demande la page profil, c'est le fichier *profil.tpl* qui sera demandé pour le rendu i.e. utilisé comme paramètre de la fonction display de Smarty.

```
$smarty->display('profil.tpl',...);
```

Architecture de vues

Le diagramme UML suivant illustre l'architecture de vues à mettre en place.



Si ce n'est pas déjà fait, créez le fichier *layout.tpl*. Ce fichier permet de définir la structure générale du site. Il définit les différents blocs et leurs dispositions. C'est dans ce fichier qu'on retrouve la définition du DOCTYPE, l'inclusion des contenus secondaires et la définition du bloc de contenu principal.

Par exemple, pour le bloc d'entête, vous devriez avoir quelque chose comme :

```
<!doctype html>
<html lang="fr">
 ...
 <header>
   {include 'layout/layout_entete.inc.tpl'}
 </header>
...
```

Chaque fichier d'extension *.tpl* du répertoire application/views/modules hérite de *layout.tpl* et redéfini le bloc de contenu principal.

Fichier *accueil.tpl*

Éditez le fichier *accueil.tpl* et faites-le hériter du fichier *layout.tpl*. Ensuite, ajoutez-y une première section (balise article par exemple) contenant l'édito. Puis une deuxième section contenant l'affichage du tableau alea précédemment créé.

Fichier *profil.tpl*

Éditez le fichier *profil.tpl* et faites-le hériter de *layout.tpl*. Ensuite, faites-lui afficher la phrase « Bienvenu <prenom> <nom> » où <nom> et <prenom> sont issus du tableau *$user* précédent.

ÉTAPE 4 – ASSEMBLAGE DE LA PAGE

Le lien entre le gabarit et le contenu est réalisé dans le fichier de script **index.php** à la racine de votre site. Avant de s'occuper de ce fichier, vous allez tout d'abord devoir compléter le fichier **config.inc.php**.

Dans le fichier **config.inc.php** vous allez définir les éléments suivants :

1. Un tableau nommé $_PAGES qui associe à chaque module, identifié par le nom du fichier (sans l'extension), un identifiant unique. C'est cet identifiant qui, passé dans l'url, permettra de connaitre le contenu à afficher.
2. Les constantes *HOME_PAGE* et *ERROR_404* qui définissent respectivement l'identifiant de la page qui sera la page d'accueil et l'identifiant de la page 404 (cela peut-être l'accueil ou une page spécifique).

Vous avez maintenant tout ce qu'il vous faut pour créer le fichier **index.php**. Dans ce fichier, vous devez maintenant :

1. Inclure le fichier **config.inc.php** ;
2. Créer un tableau associatif *$data* vide. C'est ce tableau qui va contenir les données générées par le module et qui va être transmis à la vue.
3. Récupérer, via la super globale *$_GET,* la valeur de l'attribut *page* passé en paramètre dans l'url et l'affecter à une variable *$current_page*.
4. Utiliser le tableau *$_PAGES* et la variable $current_page pour inclure le module correspondant et générer les données nécessaire dans $data.
5. Transmettre les données à Smarty (regardez du côté de la fonction *assign(…)*)
6. Appeler la fonction display de Smarty, avec les bons paramètres, pour déclencher le rendu de la vue.

Remarque :

N'oubliez pas de traiter le cas où il n'y a aucun identifiant de page fourni dans l'URL ainsi que le cas où l'identifiant n'existe pas.