

# Présentation

L'application Docu-Mission permet pour une entreprise de créer un gestionnaire de documentation, qui permettra aux salariés de gérer/créer puis consulter/rechercher dans une documentation partagée.

## Contraintes techniques

- PHP/Mysql
- Doctrine pour le mappage relationnel/Objet
- CodeIgniter pour la mise en place MVC
- Bootstrap pour le fonctionnement côté client

## Documents

### Script de la base de données

[docu.sql](#)

### Diagramme de cas d'utilisation



# Listages thèmes / documents par domaine

## Modèle

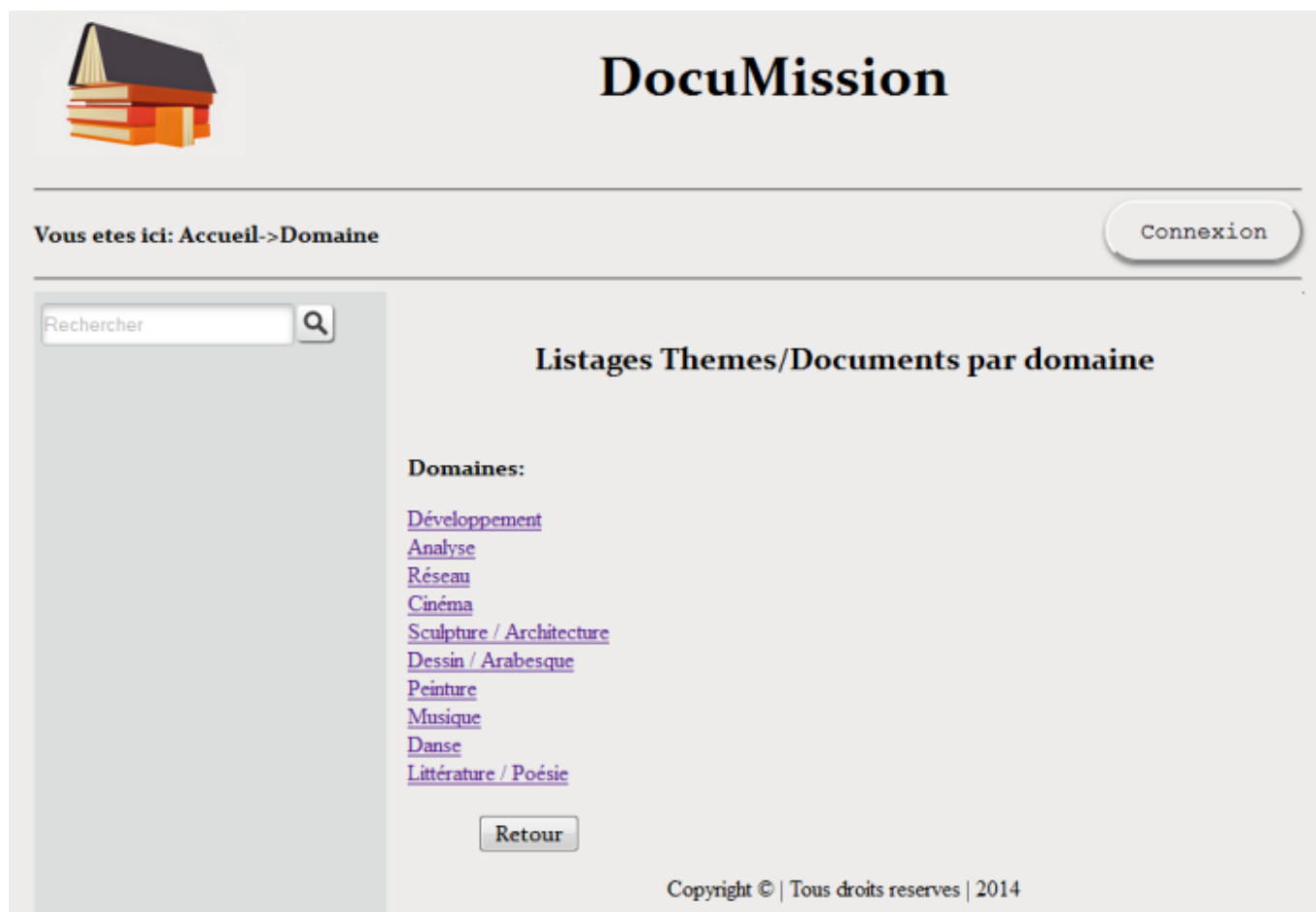
Travail sur 3 classes métiers:

1. Domaine
  2. Thème
  3. Document
- Un thème appartient à un domaine
  - Un thème peut contenir zéro, un, ou plusieurs documents
  - Un document appartient à un thème

## Vue

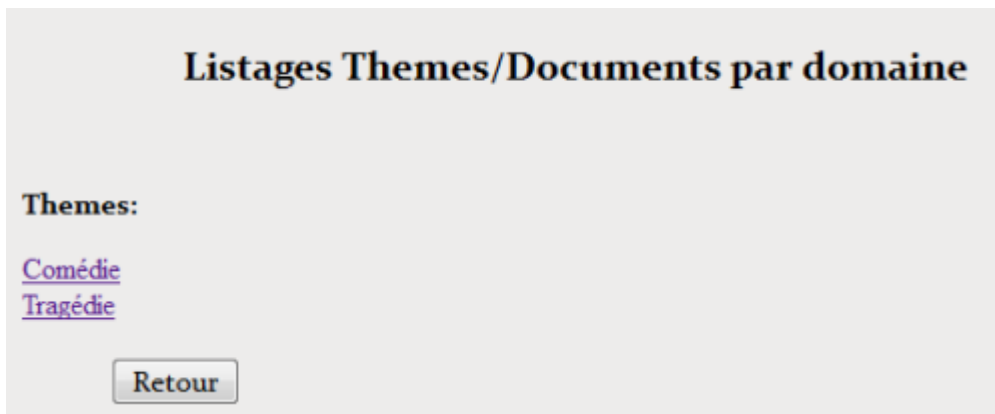
Pour le listage des thèmes / documents par domaine, 1 seule vue est utilisée:

- v\_listages (Vue d'ensemble)

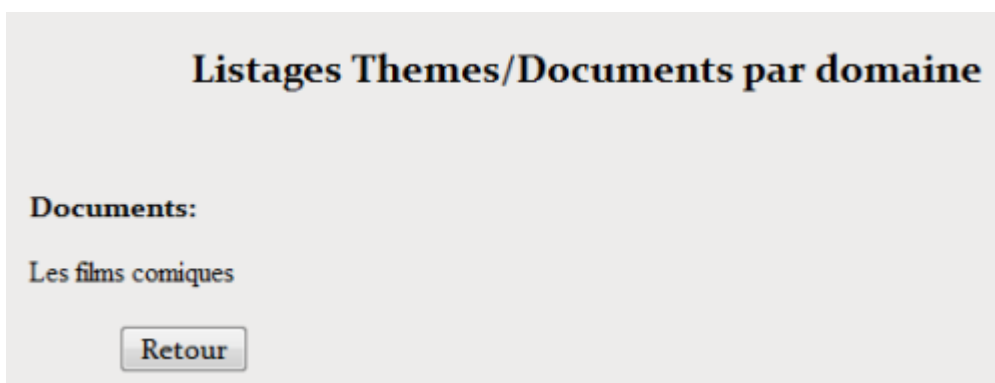


The screenshot shows the DocuMission website interface. At the top left is a logo of a stack of books. The title "DocuMission" is centered at the top. Below the title, there is a navigation bar with "Vous etes ici: Accueil->Domaine" on the left and a "Connexion" button on the right. A search bar with the placeholder "Rechercher" and a magnifying glass icon is located on the left side. The main content area is titled "Listages Themes/Documents par domaine". Underneath, there is a section labeled "Domaines:" followed by a list of domain names: Développement, Analyse, Réseau, Cinéma, Sculpture / Architecture, Dessin / Arabesque, Peinture, Musique, Danse, and Littérature / Poésie. A "Retour" button is positioned below the list. At the bottom right, there is a copyright notice: "Copyright © | Tous droits reserves | 2014".

- Exemple (Cinéma)



- Exemple (Comédie)



## Contrôleur

Le listage des thèmes et des documents par domaine est effectué dans "Listages.php". Le contrôleur possède plusieurs méthodes pour optimiser la mise en place des fonctionnalités. (Voir ci-dessous)

- **index** : Méthode appelée par défaut, appel les vues supplémentaire (header, left, footer...)
- **refresh** : Méthode appelée par l'index et utilisé pour l'affichage initial et le rafraîchissement après une modification (Paramètre de la BDD/fonction JS...)
- **selectDomaine** : Méthode qui permet d'afficher les thèmes correspondant au domaine qui lui est associé.
- **selectTheme** : Méthode qui permet d'afficher les documents correspondant au thème qui lui est associé. (Si pas de documents dans le thème, affichage vide).

## jsUtils

- **jsUtils** : Les fonctions jsUtils sont appelées pour rendre le contenu dynamique.

```
public function __construct()
{
    parent::__construct();
    $this->load->library('jsUtils');
}
```

1. **\$library\_src** doit-être ajouté la vue pour charger à la librairie jQuery
2. **\$script\_foot** script compilé pour effectuer les actions.

```
<?php echo $library_src;?>  
<?php echo $script_foot;?>
```

### Liste des méthodes utilisées :

1. **getAndBindTo** : Effectue une action suite à un événement (les paramètres de l'id Html sont envoyés automatiquement)
2. **compile** : Ajoute la portion de code écrite vers la vue
3. **doSomethingOn** : Peut servir à ajouter ou supprimer des éléments dans un contenu existant
  - Les méthode jsUtils sont appelées de la façon suivante : `$this->jsUtils->`

## Requête doctrine

Les méthode doctrine sont appelées de la manière suivante : `$this->doctrine->em->`

1. **createQuery** : Création de la requête
2. **getResult** : Récupération multiple

- Exemple 1:

```
$query = $this->doctrine->em->createQuery("SELECT t FROM Theme t JOIN t.domaine d WHERE d.id=".$param);  
$domaine = $query->getResult();
```

- Exemple 2:

```
$query = $this->doctrine->em->createQuery("SELECT doc FROM Document doc JOIN doc.theme t WHERE t.id=".$param);  
$theme = $query->getResult();
```

From:  
<http://slamwiki2.kobject.net/> - SlamWiki 2.1

Permanent link:  
<http://slamwiki2.kobject.net/etudiants/2014/croullier/docuwiki?rev=1418895276>

Last update: 2019/08/31 14:32

