

# Rapport sur le projet de Technologie - Web/Serveur

Wassim Saidane, Aurélien Authier

## Table des matières

<b>1</b>	<b>Technologies utilisés</b>	<b>1</b>
<b>2</b>	<b>Comment lancer notre code ?</b>	<b>1</b>
<b>3</b>	<b>La base de données</b>	<b>1</b>
3.1	categories . . . . .	1
3.2	membres . . . . .	2
3.3	sujet . . . . .	2
3.4	postSujet . . . . .	2
<b>4</b>	<b>Le code</b>	<b>3</b>
4.1	L'inscription . . . . .	3
4.2	La connexion et la deconnexion . . . . .	5
4.3	Ajouter un sujet et une discussion . . . . .	6
4.4	Modifier et supprimer . . . . .	6
4.5	Administrateur et bannis . . . . .	6

# Introduction

Le but de ce projet était de concevoir et coder un mini-blog, avec toutes les fonctionnalités associées (gestion des articles, système de commentaires, gestion des utilisateurs, administration...). Le rapport qui suit va expliquer nos choix de technologies utilisées et comment démarrer le programme.

## 1 Technologies utilisés

Pour ce projet, nous avons utilisé le langage de programmation PHP, nous l'avons choisi car c'était le langage le plus répandu, il est également facile à utiliser car on peut y inclure du HTML. On peut donc coder sans se soucier d'aspect technique côté serveur et un de nous deux connaissait déjà ce langage. Nous avons utilisé une base de données MySQL, le langage nous avait été enseigné en L2. Une autre raison est l'utilisation d'un serveur apache (plus simple pour nous qui codons sur Windows), pour le PHP, nous avons donc utilisé php-MyAdmin via WampServeur. Tous ces outils fonctionnent bien ensemble, et nous ont permis de gagner du temps sur des aspects techniques (serveur) que nous ne maîtrisons pas encore bien.

## 2 Comment lancer notre code ?

Le code devrait fonctionner sous n'importe quel serveur apache et sur php-MyAdmin. Si cela ne fonctionne pas nous avons fait une fiche readme (Markdown, html et pdf) avec quelques ressources. Si les problèmes persistent vous pouvez nous contacter.

## 3 La base de données

Avant de coder quoi que soit, nous nous sommes d'abord intéressés au contenu de la base de données. Nous avons ainsi créé quatre tables sur php-MyAdmin.

### 3.1 categories

Il s'agit d'un ajout (par rapport au sujet initial) de notre part. L'utilisateur choisit la catégorie en rapport avec son article.

La table est composée dans son état actuel de deux champs :

- id
- name

On a deux catégories <sup>1</sup>

---

1. On peut imaginer qu'un administrateur ou même un utilisateur crée des catégories selon son envie

### 3.2 membres

Ici sera stocké les données des utilisateurs qui ce seront inscrit. On a six champs :

- id
- pseudo
- email
- mdp
- isAdmin<sup>2</sup>
- isBanned<sup>2</sup>

### 3.3 sujet

Nous avons décidé de marquer une petite différence avec l'énoncé (par facilité), au lieu d'avoir des articles et des commentaires classiques, nous avons des sujets et une discussion sous chaque sujets. D'un point de vu utilisateur cela ne change rien.

On a trois champs :

- id
- name
- categorie

### 3.4 postSujet

Il s'agit de la discussion dont nous avons parler précédement. On a cinq champs :

- id
- propri<sup>3</sup>
- contenu
- date
- sujet

---

2. binaire même si dans les faits il s'agit d'un entier

3. L'identifiant de celui qui à créer le commentaire

## 4 Le code

Nous avons divisé le code en trois, un répertoire css pour le style, un répertoire pour les fonctions et le reste.

### 4.1 L'inscription

Le fichier inscription.php va utiliser un formulaire HTML classique qui va appeler inscription.class.php dans le répertoire fonction.

Extrait de inscription.php :

```
1      <form method="post" action="inscription.php">
2          <p>
3              <input name="pseudo" type="text" placeholder="
Pseudo..." required /><br>
4              <input name="email" type="text" placeholder="
Adresse email..." required /><br>
5              <input name="mdp" type="password" placeholder="Mot
de passe..." required /><br>
6              <input name="mdp2" type="password" placeholder="
Confirmation..." required /><br>
7              <input type="submit" value="S'inscrire!" />
8          <?php
9              if(isset($erreur)){
10                  echo $erreur;
11              }
12          ?>
13      </p>
14  </form>
15
```

Ansı toutes les données entrées vont être récupérées dans le constructeur de la classe.

Extrait de inscription.class.php :

```
1      class inscription{
2
3          private $pseudo;
4          private $email;
5          private $mdp;
6          private $mdp2;
7          private $bdd;
8
9          public function __construct($pseudo,$email,$mdp,$mdp2){
10
11              $pseudo = htmlspecialchars($pseudo);
12              $email = htmlspecialchars($email);
13
14              $this->pseudo = $pseudo;
15              $this->email = $email;
16              $this->mdp = $mdp;
17              $this->mdp2 = $mdp2;
18              $this->bdd = bdd();
19
20          }
21
```

Ensuite nous allons faire des requetes SQL pour entrer les données dans la BDD et faire une session active.

Extrait de inscription.class.php :

```
1      public function enregistrement(){
2
3          $requete = $this->bdd->prepare('INSERT INTO membres(pseudo,
4          email,mdp) VALUES(:pseudo,:email,:mdp)');
5          $requete->execute(array(
6              'pseudo'=> $this->pseudo,
7              'email' => $this->email,
8              'mdp' => $this->mdp
9          ));
10
11          return 1;
12      }
13
14      public function session(){
15          $requete = $this->bdd->prepare('SELECT id FROM membres
16          WHERE pseudo = :pseudo ');
17          $requete->execute(array('pseudo'=> $this->pseudo));
18          $requete = $requete->fetch();
19          $_SESSION['id'] = $requete['id'];
20          $_SESSION['pseudo'] = $this->pseudo;
21
22          return 1;
23      }
```

## 4.2 La connexion et la deconnexion

Pour la connexion le procéder est similaire, la seule différence réside dans le fait que nous n'entrons pas de valeur dans la base de données mais nous ne faisons qu'une vérification.

Extrait de connexion.class.php :

```
1      public function verif(){
2
3          $requete = $this->bdd->prepare('SELECT * FROM membres WHERE
pseudo = :pseudo');
4          $requete->execute(array('pseudo'=> $this->pseudo));
5          $reponse = $requete->fetch();
6          if($reponse){
7
8              if($this->mdp == $reponse['mdp']){
9                  return 'ok';
10             }
11             else {
12                 $erreur = 'Le mot de passe est incorrect';
13                 return $erreur;
14             }
15
16         }
17         else {
18             $erreur = 'Le pseudo est inexistant';
19             return $erreur;
20         }
21     }
22
23
24 }
25
```

Pour la déconnexion on détruit juste la session active.

### 4.3 Ajouter un sujet et une discussion

De la même façon un formulaire est utilisé et les données sont transportées.  
Extrait de addSujet.class.php :

```
1      public function insert(){
2          echo $this->categorie;
3          $requete = $this->bdd->prepare('INSERT INTO sujet(name,
4          categorie) VALUES(:name,:categorie)');
5          $requete->execute(array(
6              'name'=> $this->name,
7              'categorie'=> $this->categorie
8          ));
9
10         $requete2 = $this->bdd->prepare('INSERT INTO postSujet(
11         propri,contenu,date,sujet) VALUES(:propri,:contenu,NOW(),:sujet
12         )');
13         $requete2->execute(array('propri'=>$_SESSION['id'],'contenu
14         '=> $this->sujet,'sujet'=> $this->name));
15
16         return 1;
17     }
18 }
```

Même procéder pour la discussion.

### 4.4 Modifier et supprimer

Pour supprimer on va utiliser un bouton et la requête tiens en quelques lignes.

delete.php :<sup>4</sup>

```
1      $requete = $bdd->prepare('DELETE FROM postSujet WHERE id =
2      '.$id);
```

Pour modifier le message, le procéder est le même, on utilise une textarea pour transporter la modification.

### 4.5 Administrateur et bannis

Ce point de code n'est malheureusement pas optimal, nous avons juste mis une visibilité totale pour l'administrateur en jouant avec les if.<sup>5</sup>

---

4. On a pas utilisé de fonction/class par manque de temps  
5. Il aurait fallu utiliser un modèle vue-controlleur

## Conclusion

Ce projet nous demandais de réaliser un mini-blog pour cela nous avons utilisé PHP, MySQL, Apache et WampServer. Cependant, par manque de temps certains points n'ont pas pu être finis, comme la gestion de l'administration et l'UI/UX de notre site, ce qui donne un résultat, un peu austère. Nous avons cependant réalisé le plus important, la base technique, un délai un tout petit peu plus long aurait suffi pour finir le projet.