



BTS SIO – Option SLAM

Documentation d'épreuve

Rédacteur	Version	Date	Nb pages
Johann DIETRICH	1.3	21/01/2023	19

Mise en place d'un système de création d'articles à base de « What you see is what you get » et de commentaires sur ceux-ci

SOMMAIRE

1	CAHIER DES CHARGES	3
1.1	Introduction.....	3
1.2	Expression fonctionnelle du besoin	4
1.3	Contraintes.....	5
1.4	Gestion des droits d'accès.....	6
2	DESCRIPTION DES ENVIRONNEMENTS	7
3	METHODOLOGIE.....	8
3.1	Méthodologie et versioning.....	8
3.2	Gestion des tests de la solution	8
3.3	Gestion de projet	9
4	MISE EN OEUVRE	10
4.1	Création d'articles.....	10
4.2	Affichage d'articles	13
4.3	Création et affichage de commentaires	15
5	GESTION DE LA MAINTENANCE (CORRECTIVE / EVOLUTIVE).....	18
5.1	Mise à jour de la documentation du SI.....	18
5.2	Evaluation de la qualité de la solution.....	18
5.3	Procédure de correction d'un dysfonctionnement	18
6	BILAN DU PROJET.....	19
6.1	Validation des exigences point par point.....	19
6.2	Axes d'amélioration	19
6.3	Compétences acquises	19

1 Cahier des charges

1.1 Introduction

Type de mission
Mission effectuée durant du temps libre personnel
Contexte
Besoin de créer une système de création d'articles performant et utilisable par n'importe qui sans compétences de programmation.
Budget disponible
Aucun budget disponible/ budget personnel mais inutile dans ce contexte-ci
Outils disponibles
VSCode, TinyMCE

1.2 Expression fonctionnelle du besoin

Liste des fonctionnalités attendues :

Front office

Page de création d'articles :

- Permettant de créer des articles en mode « what you see is what you get »

Page de sélection d'articles:

- Permettant de voir les articles et d'ajouter des commentaires

1.3 Contraintes

Générales
Fait en une semaine avec aucune condition
Juridiques
Aucune information personnelle n'est entrée donc aucun besoin
Ergonomique
Doit être compréhensible visuellement

1.4 Gestion des droits d'accès

Sachant qu'il s'agit plus d'une test qu'un cas réel, il n'y a qu'un seul utilisateur mais dans un cas concret on aurait :

Administrateur
Peut créer des articles les lire et mettre des commentaires

Utilisateurs
Peux lire des articles et mettre des commentaires

Visiteurs
Peux lire des articles

2 Description des environnements

Environnement de développement

Environnement local à base de serveur localhost grâce à VSCode(liveserver) et de base de données WAMP (donc phpmyadmin)

3 Méthodologie

3.1 Méthodologie et versioning

Utilisation de GitHub, simplement en mettant en place une nouvelle version à chaque ajout important au projet.

3.2 Gestion des tests de la solution

unitaires

Utilisation simple des messages d'erreurs automatiques de PHP + vérification de la base de données à chaque action ayant un lien avec celle-ci.

fonctionnel

Création d'un article avec titre et contenu

Essai de lecture/affichage de celui-ci

Essai d'ajout de commentaire

3.3 Gestion de projet

3.3.1 Budget

Aucun budget nécessaire, potentiel besoin d'acheter TinyMCE premium en cas d'utilisation professionnelle mais dans notre cas, ce n'est pas nécessaire du tout.

Et vu que le reste est en local, aucun prix à payer (mis à part l'électricité)

4 Mise en oeuvre

Le principe de notre projet est assez simple, il nous faut créer l'équivalent de trois pages. Une page permettant l'insertion des articles, une page permettant l'affichage des articles et une dernière « page » permettant l'affichage et la création de commentaires.

Ses pages seront liées à une base de données appelée à chaque fois mais tout sera précisé et montré lors de la présentation de chaque page.

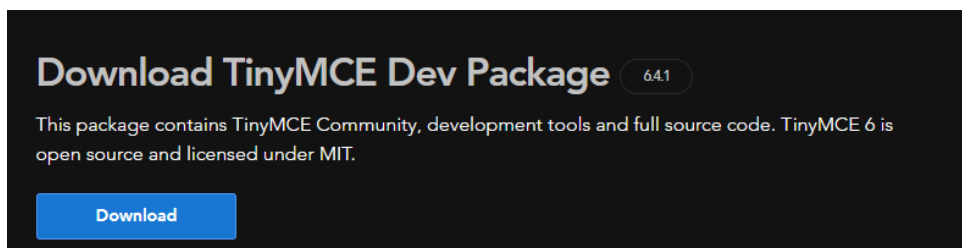
Nous allons donc commencer avec la partie la plus importante, la page de création d'article avec tinyMCE.

4.1 Création d'articles

Avant de commencer, il y a deux choses à mentionner pour cette page.

TinyMCE est un outil d'écriture qui existe en tant que plugin ou en tant qu'API, c'est-à-dire on peut soit avoir une version téléchargée et intégrée à notre site, soit une version qui fait appel au programme via internet.

Par préférence, j'ai opté pour le téléchargement et l'intégration directe au site afin d'éviter de devoir créer un compte et demander une clé afin d'utiliser l'API mais c'est tout à fait possible de faire le même projet avec l'API.



Une fois que celui-ci a été téléchargé, il suffit de le mettre dans le même fichier que notre projet et d'indiquer le chemin dans notre code afin de pouvoir s'en servir. Dans notre cas, il ressemble à ceci mais il est important à noter qu'il faudra potentiellement le changer selon le chemin d'accès aux fichiers.

```
<link rel="stylesheet" href="tinymce\js\tinymce\skins\ui\oxide\content.min.css">
<link rel="stylesheet" href="tinymce\js\tinymce\skins\ui\oxide\skin.min.css">
<script src="tinymce\js\tinymce\tinymce.min.js"></script>
```

La deuxième chose à mentionner est l'utilisation de la base de données qui elle aussi demande une connexion grâce à un snippet de code qui sera réutilisé sur toutes les pages comme ci-dessous :

```
$DB_host = "localhost";
$DB_user = "root";
$DB_pass = "aaaa";
$DB_name = "wysiwyg";
try{
    $conn = new PDO("mysql:host={$DB_host};dbname={$DB_name}", $DB_user, $DB_pass);
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
} catch(PDOException $e){
    echo $e->getMessage();
}
```

Ce code permettant tout simplement de se connecter à la base de données voulue avec le mot de passe et l'utilisateur donné.

Pour commencer donc, nous aurons donc besoin d'une table créée afin de stocker nos articles.

<input type="checkbox"/>	1	id	int		Non	Aucun(e)	AUTO_INCREMENT	Modifier	Supprimer	Plus
<input type="checkbox"/>	2	titre	varchar(255)	utf8mb3_general_ci	Non	Aucun(e)		Modifier	Supprimer	Plus
<input type="checkbox"/>	3	contenu	text	utf8mb3_general_ci	Non	Aucun(e)		Modifier	Supprimer	Plus
<input type="checkbox"/>	4	date_creation	datetime		Non	Aucun(e)		Modifier	Supprimer	Plus

Celle-ci retenant que les informations les plus importantes tel que le titre le contenu et la date de création (l'ID sert à les identifier à nous servira plus tard).

Maintenant qu'on a enfin toute la structure nécessaire, passons au code. Celui-ci étant assez simple malgré tout, ce sont les fonctionnalités qui seront plus impressionnantes. En effet, pour notre page de création d'article il nous suffit de faire un petit formulaire et de laisser TinyMCE s'occuper de le customiser.

```
<form action="creer_article.php" method="post">
  <label for="titre">Titre :</label>
  <input type="text" id="titre" name="titre"><br><br>
  <label for="contenu">Contenu :</label>
  <textarea id="contenu" name="contenu"></textarea><br><br>
  <input type="submit" value="Créer l'article">
</form>
```

Avec comme petit ajout, le script ci-dessous pour autoriser TinyMCE de modifier toutes les balises avec l'ID « contenu ».

```
<script>
  tinymce.init({
    selector: '#contenu'
  });
</script>
```

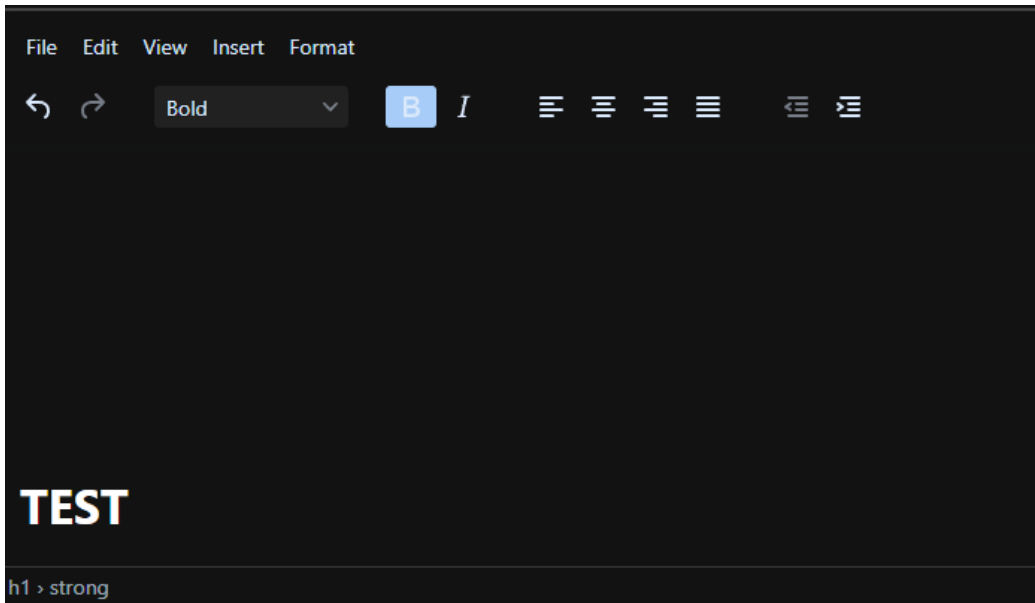
Ce qui nous donne le résultat final suivant :

On peut donc voir les deux parties distinctes, avec le titre (très basique)

Et la partie de rédaction de texte, celle-ci ressemblant énormément à beaucoup des outils de rédactions utilisés (word) et qui offre également les spécifications basiques tel que le décalage de texte (mettre au centre, à droite), le changement de style (en gras, en italique), les titres et également la barre permettant d'enregistrer même ce qui a été écrit.



Il est intéressant à noter également que tous les styles utilisés pour un mot sont montrés en bas à gauche de la boîte de texte, dans ce cas, h1 et strong. C'est donc de cette manière que celui-ci enregistre la forme du texte écrit, avec des balises html.



Maintenant qu'on s'est occupé de la partie graphique, il nous faut plus que la partie en arrière-plan qui enregistre donc les articles dans notre base de données.

Celle-ci sera effectuée lors de l'appel de la page « créer_article.php » qui est appelé après la confirmation du formulaire.

```
$sql = "INSERT INTO articles (titre, contenu, date_creation) VALUES (:titre, :contenu, NOW())";
$stmt = $conn->prepare($sql);
$stmt->bindParam(':titre', $titre);
$stmt->bindParam(':contenu', $contenu);

// Exécuter la requête SQL
$stmt->execute();

header("location:index.php");
exit;

// Fermer la connexion à la base de données
$conn = null;
```

Le code faisant simplement une insertion dans la base de données avec toutes les valeurs du formulaire et la valeur « now » qui permet d'indiquer le temps et la date actuelle afin de l'insérer également.

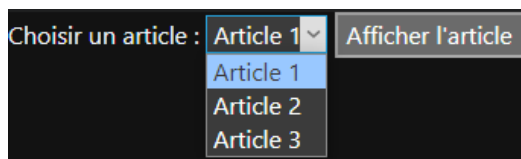
4.2 Affichage d'articles

Pour notre page d'affichage d'article, il y a deux choses à faire. Faire un sélecteur d'articles qui permet à l'utilisateur de choisir celui qu'il veut lire, et justement l'affichage de l'article choisi. Les deux sont faits avec grâce à des requêtes SQL mélangées à du code PHP.

Le code de notre sélecteur ressemble à cela :

```
<form method="get" action="">
  <label for="article-select">Choisir un article :</label>
  <select id="article-select" name="article_id">
    <?php
      // Récupérer les articles de la base de données
      $sql = "SELECT id, titre FROM articles";
      $result = $conn->query($sql);
      while ($row = $result->fetch()) {
        // Afficher les options du selecteur avec les titres des articles
        echo '<option value="' . $row['id'] . '">' . $row['titre'] . '</option>';
      }
    ?>
  </select>
  <button type="submit">Afficher l'article</button>
</form>
```

Il s'agit donc d'un mélange entre un formulaire de choix à plusieurs options (select), une boucle while et une requête SQL qui récupère les articles. Ce qui permet de faire en sorte d'ajouter tout les articles en tant qu'options sélectionnables. Le résultat donne donc ceci :



La prochaine partie est la récupération des informations selon l'article choisi. En effet, il nous faut rajouter du code afin que le bouton afficher l'article fonctionne. Celui-ci doit récupérer l'ID de l'article sélectionné et le réutiliser pour en sortir les autres informations tel que le nom, le contenu et la date.

```
if (isset($_GET['article_id'])) {
  // Récupérer l'id de l'article sélectionné
  $article_id = $_GET['article_id'];

  // Récupérer l'article de la base de données
  $sql = "SELECT * FROM articles WHERE id = :article_id";
  $stmt = $conn->prepare($sql);
  $stmt->bindParam(':article_id', $article_id);
  $stmt->execute();
  $article = $stmt->fetch();
}
```

Une fois ces informations récupérées, il suffit de les afficher avec des simples lignes :

```
// Afficher l'article
echo '<h1>' . $article['titre'] . '</h1>';
echo '<p>' . $article['contenu'] . '</p>';
echo '<p>Date de création : ' . $article['date_creation'] . '</p>';
```

Ce qui au final nous affiche cela

Choisir un article : Article 1 Afficher l'article

Article 2

Test de grand format d'article

Test italique

Test de texte centré



















test d'écriture de code

Date de création : 2023-04-24 10:45:56

Ce qui est notre article avec tout ce qui avait été défini dans notre création d'articles.

4.3 Création et affichage de commentaires

Pour notre dernière partie nous allons donc devoir nous occuper d'une nouvelle fonctionnalité, la création de commentaires. Pour celle-ci, comme pour les articles, nous allons devoir créer une table dans notre base de données afin de les stocker de manière appropriés. Notre table ressemble à ceci :

<input type="checkbox"/>	1	id	int	UNSIGNED	Non	Aucun(e)	AUTO_INCREMENT			
<input type="checkbox"/>	2	article_id	int	UNSIGNED	Non	Aucun(e)				
<input type="checkbox"/>	3	nom	varchar(50)	utf8mb4_0900_ai_ci	Non	Aucun(e)				
<input type="checkbox"/>	4	email	varchar(50)	utf8mb4_0900_ai_ci	Non	Aucun(e)				
<input type="checkbox"/>	5	contenu	text	utf8mb4_0900_ai_ci	Non	Aucun(e)				
<input type="checkbox"/>	6	date_creation	timestamp		Oui	CURRENT_TIMESTAMP	DEFAULT_GENERATED			

Il est important à noter que la ligne « article_id » est très importante pour le fonctionnement des commentaires. Celle-ci permet de savoir à quel article appartient le commentaire, ce qui est essentiel si l'on ne veut pas avoir les commentaires d'un autre article.

Afin de créer un commentaire il nous faut donc créer un formulaire, et insérer les informations de ce formulaire dans notre base de données. On fait cela comme ça :

```
<div class="col-md-6">
  <h3>Ajouter un commentaire</h3>
  <form action="ajouter_commentaire.php" method="POST">
    <input type="hidden" name="article_id" value="<?php echo $article_id; ?>">
    <div class="form-group">
      <label for="nom">Nom :</label>
      <input type="text" class="form-control" id="nom" name="nom" required>
    </div>
    <label for="commentaire">Commentaire :</label>
    <textarea class="form-control" id="commentaire" name="commentaire" rows="3"
required></textarea>
  </div>
  <button type="submit" class="btn btn-primary">Ajouter</button>
</form>
```

Ce qui n'est qu'un simple formulaire qui appelle une fonction que l'on verra un peu plus tard mais nous donne le visuel ci-dessous (grâce à Bootstrap).

Ajouter un commentaire

Nom :

Adresse e-mail :

Commentaire :

Ajouter

Il nous reste encore à récupérer et insérer les données dans notre base de données. Ceci est fait avec ce code ci-dessous :

```
$sql = "INSERT INTO commentaires (nom, email, contenu, date_creation, article_id) VALUES (:nom, :email, :contenu, NOW(), :article_id)";
$stmt = $conn->prepare($sql);
$stmt->bindParam(':nom', $nom);
$stmt->bindParam(':email', $email);
$stmt->bindParam(':contenu', $commentaire);
$stmt->bindParam(':article_id', $article_id);

// Exécuter la requête SQL
$stmt->execute();
header("Location: article.php?id={$article_id}");
exit;
```

Celui-ci s'occupe de récupérer toutes les informations du formulaire + l'ID de l'article auquel il est lié + la date actuelle. Puis finalement redirige vers la page des articles.

	id	article_id	nom	email	contenu	date_creation
<input type="checkbox"/>	Éditer		Copier		Supprimer	1 3
	johann	jodiet@hotmail.fr	Contenu de test	2023-04-24 11:26:31		

Maintenant qu'on a pu créer des commentaires, la dernière étape de notre projet sera donc de les afficher et ceci de manière à ce que chaque article n'ait que ses propres commentaires et pas ceux d'autres articles.

L'affichage est très simple au final, le fonctionnement est similaire au sélecteur d'articles que l'on avait avant. Une requête SQL mélangée à du code afin d'avoir un mélange des deux.

```
$article_id = $_GET['article_id'];
$sql = "SELECT * FROM commentaires WHERE article_id = :article_id ORDER BY date_creation DESC";
$stmt = $conn->prepare($sql);
$stmt->bindParam(':article_id', $article_id);
$stmt->execute();
$resultats = $stmt->fetchAll(PDO::FETCH_ASSOC);
```

La première partie de notre code récoltant simplement tout les commentaires ou l'ID d'article correspond à l'article actuellement sur la page et les stocke dans l'ordre croissant de création.

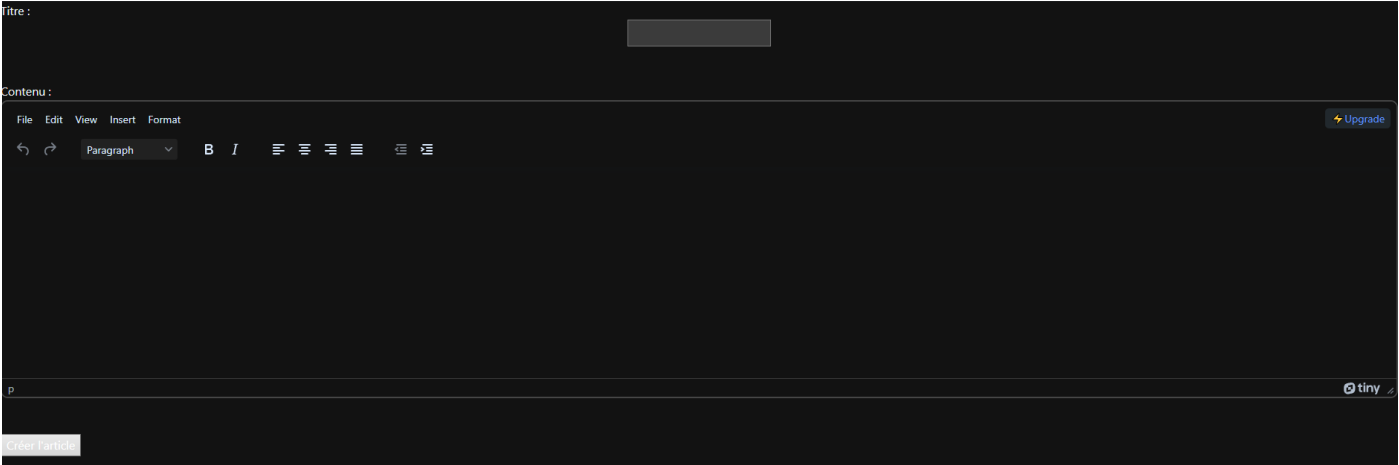
```
// Afficher les commentaires
foreach ($resultats as $commentaire) {
    echo "<div class='card my-3'>";
    echo "<div class='card-body'>";
    echo "<h5 class='card-title'>" . $commentaire['nom'] . "</h5>";
    echo "<p class='card-text'>" . $commentaire['contenu'] . "</p>";
    echo "</div>";
    echo "</div>";
}
```

Et notre deuxième partie s'occupe simplement d'afficher les informations qui ont été récupérées avec une boucle. Tant qu'il y a un commentaire, celui-ci va être affiché. Ce qui nous donne le résultat ci-dessous :

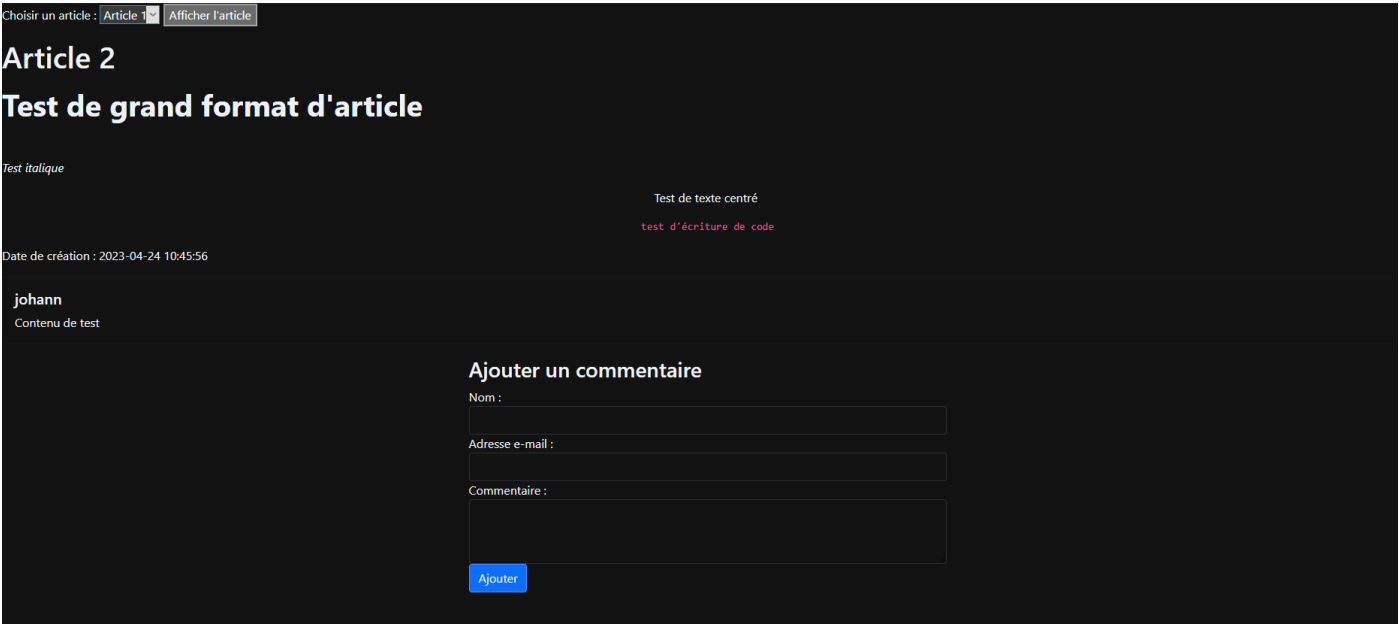
johann
Contenu de test

Le résultat final de notre projet donnant donc :

Page de création :



Page d’affichage :



5 Gestion de la maintenance (corrective / évolutive)

5.1 Mise à jour de la documentation du SI

Recueillir, analyser et mettre à jour les informations sur une version d'une solution

- Normalement, TinyMCE ne devrait pas être changé/changer de fonctionnement de manière drastique. Mais si jamais cela arrive à être le cas, le meilleur choix serait d'aller voir la documentation officielle sur [leur site](#).

Mettre à jour des documentations technique et d'utilisation d'une solution

- Vu que ce projet n'est qu'un projet de test, normalement aucune mise à jour de ce document devrait être nécessaire, mais dans le cas où une mise à jour est d'œuvre, il suffirait de me mettre à jour en adaptant le numéro de version noté en haut du document.

5.2 Evaluation de la qualité de la solution

La solution est adaptée par rapport à ce qui a été demandé et voulue, elle a l'air qualitative.

5.3 Procédure de correction d'un dysfonctionnement

Aucune réelle procédure de correction, une simple règle à suivre serait la lecture des messages d'erreur et la vérification de l'état de la base de données à chaque erreur.

6 Bilan du projet

6.1 Validation des exigences point par point

Besoin d'un système de création d'article avec TinyMCE

Besoin d'un moyen de voir les articles créés

Besoin de pouvoir mettre des commentaires sur dit articles

6.2 Axes d'amélioration

Potentiellement l'ajout d'un système de comptes utilisateurs, permettant oui ou non à un utilisateur d'écrire des articles/commentaires.

6.3 Compétences acquises

Utilisation d'un plugin dans un code PHP

Utilisation d'une base de données pour enregistrer des informations