Cahier des Charges- Système de Gestion de Blog

1 Introduction et Objectif

Ce document spécifie les exigences fonctionnelles et techniques pour la réalisation d'un **système complet de gestion de blog** moderne et robuste. L'objectif est de fournir une plateforme performante pour la publication, l'administration et l'interaction autour de contenus, en mettant l'accent sur la sécurité, l'expérience utilisateur (UX) et l'architecture logicielle.

2 Architecture Technique et Technologies

2.1 Technologies Imposées

Composant	Technologie	Version	Notes
Backend	**PHP**	8.X (POO)	Utilisation de la **Programmation Orientée Objet (POO)**.
Base de Données	**MariaDB**	Dernière stable	Base de données relationnelle robuste.
Templating	**Twig**	3.X	Moteur de template pour la séparation des préoccupations.
Frontend/UI	**Bootstrap**	5.X	Framework CSS pour la réactivité et le style.
Interactivité	**Alpine.js**	Dernière stable	Micro-framework JavaScript réactif pour les composants frontend.

2.2 Patrons de Conception (Design Patterns)

- **MVC (Modèle-Vue-Contrôleur) :** Architecture obligatoire pour structurer le code et séparer la logique métier, la présentation et le contrôle.
- **Singleton :** Utilisé spécifiquement pour la classe de gestion de la connexion à la base de données via **PDO**, garantissant une instance unique.

2.2.1 Composants Fondamentaux

- **Gestionnaire de Session :** Module dédié pour gérer l'état de l'utilisateur (authentification, informations de session, flash messages).
- **Logger :** Implémentation d'un système de journalisation (logs) pour enregistrer les événements critiques (erreurs, tentatives de connexion, actions admin importantes).
- **Accès DB :** Utilisation de **PDO** pour des requêtes sécurisées et préparées.

3 Exigences Fonctionnelles (EF)

3.1 Tableau de Bord Administrateur (EF-ADMIN)

Le tableau de bord est le point central de l'administration, accessible uniquement aux utilisateurs autorisés.

- **EF-ADMIN-01 :** Vue synthétique des statistiques clés (nombre d'articles, commentaires en attente, utilisateurs actifs, etc.).
- **EF-ADMIN-02 :** Navigation claire et intuitive vers toutes les sections de gestion.
- **EF-ADMIN-03 :** Affichage d'un fil d'activité récent (derniers articles publiés, derniers commentaires, modifications d'utilisateurs).

3.2 Contrôle d'Accès, Rôles et Permissions (EF-ACL)

Système de sécurité basé sur les rôles (Rôle Based Access Control).

Le contrôle d'accès basé sur les rôles (RBAC), ou sécurité basée sur les rôles, **est une méthode utilisée pour attribuer des autorisations et accorder l'accès en fonction du rôle d'un utilisateur au sein d'une organisation**

- **EF-ACL-01 :** Gestion des **Utilisateurs** (création, modification, désactivation, suppression).
- **EF-ACL-02 :** Gestion des **Rôles** (exemples : Administrateur, Éditeur, Contributeur, Modérateur).
- **EF-ACL-03 :** Gestion des **Permissions** (exemples : Créer Article, Éditer Article, Supprimer Commentaire, Gérer Utilisateurs).
- **EF-ACL-04 :** Un utilisateur peut se voir attribuer **plusieurs Rôles**.
- **EF-ACL-05 :** Un rôle agrège un ensemble de **Permissions**. L'accès aux fonctionnalités du système doit être strictement conditionné par la possession des permissions requises.
- **EF-ACL-06 :** Système d'authentification sécurisé (hachage du mot de passe).

3.3 Gestion des Articles (EF-ARTICLE)

- **EF-ARTICLE-01 :** Création, lecture, mise à jour et suppression (CRUD) des articles.
- **EF-ARTICLE-02 :** Éditeur WYSIWYG (ou Markdown) pour le corps de l'article.
- **EF-ARTICLE-03 :** Gestion des métadonnées (titre, URL slug, image à la une, statut : Brouillon, Publié, Archivé).
- **EF-ARTICLE-04 :** Association d'un article à un ou plusieurs **Tags**.

3.4 Gestion des Tags (EF-TAG)

- **EF-TAG-01 :** CRUD complet des Tags (nom, URL slug).
- **EF-TAG-02 :** Affichage de la liste des Tags dans le tableau de bord avec le nombre d'articles associés.

3.5 Gestion des Commentaires (EF-COMMENT)

- **EF-COMMENT-01 :** Affichage des commentaires associés à un article sur la partie publique du blog.
- **EF-COMMENT-02 :** Possibilité pour les utilisateurs non connectés de poster un commentaire (avec nom/email).
- **EF-COMMENT-03 :** Section d'administration pour la **Modération** des commentaires (Approuver/Désapprouver/Supprimer).
- **EF-COMMENT-04 :** Notification (dans le tableau de bord ou par email) des nouveaux commentaires en attente de modération.

4 Exigences d'Expérience Utilisateur (UX)

4.1 Interface Mobile First (UX-RESPONSIVE)

- **UX-RESPONSIVE-01 :** Conception et développement axés sur l'approche **Mobile First**, garantissant une ergonomie parfaite sur les appareils mobiles.
- **UX-RESPONSIVE-02 :** Utilisation complète des classes de grille et des composants de **Bootstrap 5.X** pour un design entièrement adaptatif (*responsive*).

4.2 Gestion des Thèmes (UX-THEME)

- **UX-THEME-01 :** Implémentation d'un sélecteur de **Thème Clair** et **Thème Foncé** (Dark Mode).
- **UX-THEME-02 :** Le choix du thème doit être persistant (via un cookie ou la session).

4.3 Accessibilité : Police Dyslexique (UX-ACCESSIBILITY)

- **UX-ACCESSIBILITY-01 :** Intégration d'une option ou d'un paramètre pour utiliser une **police spécifiquement conçue pour les personnes dyslexiques** (exemples : OpenDyslexic, Dyslexie Font) dans l'interface publique du blog et, idéalement, dans le tableau de bord.

5 Modèle de Données

La base de données **MariaDB** doit implémenter un schéma supportant les relations suivantes :

- **Utilisateurs** : ID, nom, email, mot_de_passe (hash), etc.
- **Roles** : ID, nom_role.
- **Permissions** : ID, nom_permission.
- **Role_User** : Pivot (ID_Role, ID_User) — **Supporte plusieurs rôles par utilisateur.**
- **Role_Permission** : Pivot (ID_Role, ID_Permission).
- **Articles** : ID, titre, slug, contenu, id_utilisateur, statut, date_creation, date_maj.
- **Tags** : ID, nom_tag, slug.
- **Article_Tag** : Pivot (ID_Article, ID_Tag).
- **Commentaires** : ID, id_article, id_utilisateur (ou nom/email), contenu, statut, date_creation.

5.1 Script de génération

Ce script SQL permet la création de la base de données **MariaDB** ou **MySQL** ainsi que l'insertion de données de test centrées sur le thème du **VTT (Vélo Tout-Terrain)**, en respectant la structure de rôles et permissions demandée.

```
--

-- Base de données : `blog_db`

--

CREATE DATABASE IF NOT EXISTS `blog_db` DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;

USE `blog_db`;


-- 1. DROP (SUPPRESSION) DES TABLES EXISTANTES (Pour un environnement de test propre)

--

SET FOREIGN_KEY_CHECKS = 0;

DROP TABLE IF EXISTS `Commentaires`;

DROP TABLE IF EXISTS `Article_Tag`;

DROP TABLE IF EXISTS `Tags`;

DROP TABLE IF EXISTS `Articles`;

DROP TABLE IF EXISTS `Role_Permission`;

DROP TABLE IF EXISTS `Role_User`;

DROP TABLE IF EXISTS `Permissions`;
```

```
DROP TABLE IF EXISTS `Roles`;
DROP TABLE IF EXISTS `Utilisateurs`;
SET FOREIGN_KEY_CHECKS = 1;

--
-- 2. CRÉATION DES TABLES
--

-- Table Utilisateurs
CREATE TABLE `Utilisateurs` (
    `id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
    `nom_utilisateur` VARCHAR(50) NOT NULL UNIQUE,
    `email` VARCHAR(100) NOT NULL UNIQUE,
    `mot_de_passe` CHAR(60) NOT NULL, -- Stocke le hash bcrypt/argon2 (60 chars pour bcrypt)
    `est_actif` BOOLEAN DEFAULT TRUE,
    `date_inscription` DATETIME DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- Table Roles
CREATE TABLE `Roles` (
    `id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
    `nom_role` VARCHAR(50) NOT NULL UNIQUE,
    `description` VARCHAR(255) NULL,
    PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
-- Table Permissions
CREATE TABLE `Permissions` (
    `id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
    `nom_permission` VARCHAR(100) NOT NULL UNIQUE,
    PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- Table de Pivot Role_User (Un Utilisateur peut avoir plusieurs Rôles)
CREATE TABLE `Role_User` (
    `role_id` INT UNSIGNED NOT NULL,
    `user_id` INT UNSIGNED NOT NULL,
    PRIMARY KEY (`role_id`, `user_id`),
    FOREIGN KEY (`role_id`) REFERENCES `Roles`(`id`) ON DELETE CASCADE,
    FOREIGN KEY (`user_id`) REFERENCES `Utilisateurs`(`id`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- Table de Pivot Role_Permission (Un Rôle a plusieurs Permissions)
CREATE TABLE `Role_Permission` (
    `role_id` INT UNSIGNED NOT NULL,
    `permission_id` INT UNSIGNED NOT NULL,
    PRIMARY KEY (`role_id`, `permission_id`),
    FOREIGN KEY (`role_id`) REFERENCES `Roles`(`id`) ON DELETE CASCADE,
    FOREIGN KEY (`permission_id`) REFERENCES `Permissions`(`id`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
-- Table Articles
CREATE TABLE `Articles` (
    `id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
    `utilisateur_id` INT UNSIGNED NOT NULL,
    `titre` VARCHAR(255) NOT NULL,
    `slug` VARCHAR(255) NOT NULL UNIQUE,
    `contenu` TEXT NOT NULL,
    `image_une` VARCHAR(255) NULL,
    `statut` ENUM('Brouillon', 'Publié', 'Archivé') DEFAULT 'Brouillon',
    `date_creation` DATETIME DEFAULT CURRENT_TIMESTAMP,
    `date_mise_a_jour` DATETIME DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
    PRIMARY KEY (`id`),
    FOREIGN KEY (`utilisateur_id`) REFERENCES `Utilisateurs`(`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- Table Tags
CREATE TABLE `Tags` (
    `id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
    `nom_tag` VARCHAR(50) NOT NULL UNIQUE,
    `slug` VARCHAR(50) NOT NULL UNIQUE,
    PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- Table de Pivot Article_Tag (Un Article a plusieurs Tags)
CREATE TABLE `Article_Tag` (
    `article_id` INT UNSIGNED NOT NULL,
    `tag_id` INT UNSIGNED NOT NULL,
    PRIMARY KEY (`article_id`, `tag_id`),
    FOREIGN KEY (`article_id`) REFERENCES `Articles`(`id`) ON DELETE CASCADE,
    FOREIGN KEY (`tag_id`) REFERENCES `Tags`(`id`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
-- Table Commentaires
CREATE TABLE `Commentaires` (
    `id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
    `article_id` INT UNSIGNED NOT NULL,
    `nom_auteur` VARCHAR(100) NOT NULL,
    `email_auteur` VARCHAR(100) NULL,
    `contenu` TEXT NOT NULL,
    `statut` ENUM('En attente', 'Approuvé', 'Rejeté') DEFAULT 'En attente',
    `date_commentaire` DATETIME DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (`id`),
    FOREIGN KEY (`article_id`) REFERENCES `Articles`(`id`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- 3. INSERTION DES DONNÉES DE TEST (THEME VTT)
--

-- Utilisateurs (Mot de passe 'password' hashé avec bcrypt pour l'exemple)
-- Le hash réel dépendra de votre implémentation PHP.
-- Ceci est un hash de démonstration pour 'test'
SET @HASHED_PASSWORD =
'$2y$10$Q7iR7/h7Gq6yRzW2gP0pT.0.1oQ5t4T8W0y5fG8E7C8zM7/V2C9a'; -- Hash pour le mot de
passe 'vttadmin'
INSERT INTO `Utilisateurs` (`id`, `nom_utilisateur`, `email`, `mot_de_passe`) VALUES
(1, 'AdminVTT', 'admin@vtt.com', @HASHED_PASSWORD),
(2, 'EditeurTrail', 'editeur@vtt.com', @HASHED_PASSWORD),
(3, 'ContributeurRando', 'contributeur@vtt.com', @HASHED_PASSWORD);
```

```
-- Rôles
INSERT INTO `Roles` (`id`, `nom_role`, `description`) VALUES
(1, 'Administrateur', 'Accès complet au tableau de bord et à la gestion des utilisateurs.'),
(2, 'Éditeur', 'Peut créer, modifier et publier ses propres articles et ceux des contributeurs.'),
(3, 'Contributeur', 'Peut créer et modifier ses propres articles (statut Brouillon uniquement).');

-- Permissions
INSERT INTO `Permissions` (`id`, `nom_permission`) VALUES
(1, 'admin_access'),
(2, 'article_creer'),
(3, 'article_editer_tous'),
(4, 'article_publier'),
(5, 'article_supprimer'),
(6, 'commentaire_gerer'),
(7, 'utilisateur_gerer'),
(8, 'tag_gerer');

-- Association Rôle - Permission
-- Administrateur (Role 1) : Toutes les permissions
INSERT INTO `Role_Permission` (`role_id`, `permission_id`) VALUES (1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6), (1, 7), (1, 8);
-- Éditeur (Role 2) : Créer, Éditer Tous, Publier, Gérer Commentaires, Gérer Tags
INSERT INTO `Role_Permission` (`role_id`, `permission_id`) VALUES (2, 1), (2, 2), (2, 3), (2, 4), (2, 6), (2, 8);
-- Contributeur (Role 3) : Créer Article
INSERT INTO `Role_Permission` (`role_id`, `permission_id`) VALUES (3, 2);
```

-- Association Utilisateur - Rôle

-- AdminVTT est Administrateur

INSERT INTO `Role_User` (`user_id`, `role_id`) VALUES (1, 1);

-- EditeurTrail est Éditeur ET Contributeur (Un utilisateur peut avoir plusieurs rôles)

INSERT INTO `Role_User` (`user_id`, `role_id`) VALUES (2, 2), (2, 3);

-- ContributeurRando est Contributeur

INSERT INTO `Role_User` (`user_id`, `role_id`) VALUES (3, 3);

-- Tags (VTT)

INSERT INTO `Tags` (`id`, `nom_tag`, `slug`) VALUES

(1, 'Traces GPS', 'traces-gps'),

(2, 'Enduro', 'enduro'),

(3, 'XC', 'xc'),

(4, 'Suspension', 'suspension'),

(5, 'Nutrition', 'nutrition'),

(6, 'Entraînement', 'entrainement'),

(7, 'Hydratation', 'hydratation');

-- Articles (VTT)

INSERT INTO `Articles` (`id`, `utilisateur_id`, `titre`, `slug`, `contenu`, `statut`) VALUES

(1, 1, 'Top 5 des Traces VTT Enduro en Rhône-Alpes', 'top-5-traces-vtt-enduro-rhone-alpes', '# Introduction

Découvrez les descentes les plus mythiques pour les amateurs d\'**Enduro** en France. Freins puissants et protections obligatoires !

La piste de l\'Écureuil

Une trace rapide avec de gros dénivelés négatifs. Idéale pour tester votre **suspension**.', 'Publié'),

(2, 2, 'Réglage de la suspension : le SAG parfait pour le XC', 'reglage-suspension-sag-xc', 'Le **SAG** (Sinking At Ground) est crucial en **XC** pour optimiser le rendement et le confort. Nous détaillons ici le processus étape par étape. Un mauvais réglage impacte directement la performance.', 'Publié'),

(3, 3, 'Gérer l\'Hydratation sur une longue sortie VTT', 'gerer-hydratation-longue-sortie', 'Au-delà de 3h, l\'eau seule ne suffit plus. Il faut intégrer des électrolytes et des glucides. Notre guide complet sur la **Nutrition** et l\'**Hydratation**.', 'Brouillon');

```
-- Association Article - Tag

INSERT INTO `Article_Tag` (`article_id`, `tag_id`) VALUES

(1, 1), (1, 2), (1, 4),

(2, 3), (2, 4),

(3, 5), (3, 7);


-- Commentaires

INSERT INTO `Commentaires` (`article_id`, `nom_auteur`, `email_auteur`, `contenu`, `statut`) VALUES

(1, 'Nicolas Rider', 'nic@trail.fr', 'Super article, je connaissais pas la piste de l\'Écureuil ! J\'y vais ce weekend.', 'Approuvé'),

(1, 'Anonyme', NULL, 'Trop de monde sur ces pistes, dommage...', 'En attente'),

(2, 'ProXC', 'pro@xc.com', 'J\'utilise le même SAG, c\'est le meilleur compromis !', 'Approuvé');
```

5.2 Notes et Instructions d'Utilisation

1. **Exécution :** Exécuter ce script directement dans votre interface **MariaDB** (via phpMyAdmin ou la console).
2. **Sécurité :** Le champ mot_de_passe est de type CHAR(60) pour stocker le hash généré par **PHP** (via password_hash() avec l'algorithme par défaut, souvent bcrypt). N'utilisez **JAMAIS** de mot de passe en clair.
3. **Données de Connexion Test :**
    - **Utilisateur Admin :** admin@vtt.com / Mot de passe : vttadmin (à changer en fonction du hash que vous générerez côté PHP).
    - **Article 3 :** Est en statut **Brouillon**, il ne devrait pas apparaître sur la partie publique du blog tant qu'il n'est pas publié par un **Administrateur** ou un **Éditeur**.
4. **Rôles et Permissions :**
    - L'utilisateur EditeurTrail possède les rôles **Éditeur** et **Contributeur**, démontrant le contrôle d'accès multi-rôles.
    - Le commentaire Trop de monde... est en statut **En attente** et nécessite une action de modération (permission commentaire_gerer).

6 Critères d'Acceptation (4 points par item)

- Le système est pleinement fonctionnel et conforme à toutes les exigences fonctionnelles et techniques.
- Le code source respecte les principes **POO** de PHP 8.X et le pattern **MVC**.
- La sécurité est assurée (validation des données, prévention des injections SQL/XSS, hachage des mots de passe).
- L'interface est entièrement **responsive** et le basculement Thème Clair/Foncé est opérationnel.
- Le contrôle d'accès basé sur les Rôles et Permissions fonctionne sans faille.

7 Livrables

1. Code source complet (Back-end PHP/Twig, Front-end Bootstrap/Alpine.js).
   Accessible à partir d'une forge logicielle.
2. Documentation technique (structure du MVC, utilisation des Design Patterns).
3. Document décrivant la répartition du travail au sein du binôme, les difficultés rencontrées, ce que vous avez retiré de ce projet.

Le travail du binôme est à me rendre au plus tard le **11 janvier 2026 à 23H59** dans ma boite mail philippe.audier@univ-lyon1.fr. Dans le message, il y aura les noms composant le binôme, le lien vers la forge contenant l'ensemble des livrables.