Cahier des charges simplifié

**1. Présentation du projet**

1. **Nom du projet :** ***MeetUpSpace***
2. **Description :** Développement d'une plateforme minimaliste permettant aux utilisateurs de réserver des salles de conférence et aux administrateurs de gérer les disponibilités.
3. **Objectif principal :** Simplifier la planification et la gestion des réservations de salles tout en évitant les conflits d'utilisation.

**2. Problèmes à résoudre / besoins exprimés**

1. **Besoins des utilisateurs :**
   * Réserver une salle rapidement pour un créneau précis.
   * Consulter et annuler leurs réservations.
2. **Besoins des administrateurs :**
   * Gérer l'ajout, la modification et la suppression des salles.
   * Valider ou annuler les réservations pour éviter les conflits.

**3. Utilisateurs cibles**

1. **Administrateurs :**
   * Responsables de la gestion des salles et des réservations.
2. **Utilisateurs :**
   * Employés ou clients souhaitant réserver une salle pour des réunions ou événements.

**4. Fonctionnalités attendues**

1. **Pour les administrateurs :**

* Gérer les salles (ajout, modification, suppression).
* Valider ou annuler les réservations.
* Consulter la liste des réservations.

1. **Pour les utilisateurs :**

* Réserver une salle pour un créneau spécifique.
* Consulter leurs réservations passées et futures.
* Annuler une réservation si nécessaire.

**5. Contraintes techniques**

* **Technologies :** Laravel (architecture MVC), MYSQL.
* **Base de données :** MYSQL pour le stockage des données.
* **Hébergement :** Solution compatible avec Laravel (ex : shared hosting, VPS).
* **Accessibilité :** Application responsive (PC et mobile).

**Objectifs techniques:**

* Respect des bonnes pratiques de développement.
* Optimisation des requêtes SQL.
* Interface intuitive et ergonomique.

**Tableau des utilisateurs et rôles**

**1. Liste des utilisateurs cibles**

|  |  |
| --- | --- |
| **Type d’utilisateur** | **Description** |
| **Administrateur** | Responsable de la gestion des salles et des réservations (ex : gestionnaire d’infrastructure). |
| **Utilisateur** | Employé ou client souhaitant réserver une salle pour un événement ou une réunion. |

**2. Roles et permissions**

| **Type d’utilisateur** | **Permissions / Actions autorisées** | **Lien avec les technologies** |
| --- | --- | --- |
| **Administrateur** | - Ajouter, modifier, supprimer des salles.  - Valider/annuler les réservations.  - Consulter toutes les réservations. | **Laravel :** Middleware admin ou Policies pour restreindre l’accès.  **React :** Routes /admin/\* protégées. |
| **Utilisateur** | - Réserver une salle. - Consulter/modifier/supprimer SES réservations.  - Voir les salles disponibles. | **Laravel :** Middleware auth pour les réservations. **React :** Routes /reservations conditionnelles. |

### ****3. Implémentation technique****

1. **Laravel :**
   * Utilisation de **Policies** pour gérer les droits (ex : SallePolicy pour vérifier si un utilisateur peut modifier une salle).
   * Middleware role:admin pour les routes critiques (/admin/salles, /admin/reservations).
2. **React :**
   * Affichage conditionnel des composants (ex : bouton "Gérer les salles" uniquement pour les admins).
   * Redirection vers /login si l’utilisateur n’est pas authentifié.

**Planification du Projet**

## **1. Liste des Tâches à Réaliser**

### ****Phase 1 : Analyse et Conception****

* **Analyse des besoins**
* Finaliser le cahier des charges et les user stories.
* Identifier les cas d’utilisation principaux.
* **Conception UML**
* **Diagramme de Cas d’Utilisation** : Interactions entre administrateurs et utilisateurs.
* **Diagramme de Classe** : Modélisation des entités (Salle, Réservation, Utilisateur).
* **Diagramme de Séquence** : Processus de réservation et validation.
* **Prototypage (Figma)**
* Maquettes interactives des interfaces :
  + Page de connexion / inscription.
  + Tableau de bord admin (gestion des salles et réservations).
  + Interface utilisateur (réservation, consultation).

### ****Phase 2 : Développement Back-End (Laravel)****

* **Setup initial**
* Installation de Laravel, configuration de MYSQL.
* Mise en place de l’authentification (Laravel Breeze/Jet).
* **Fonctionnalités clés**
* CRUD pour les salles (admin).
* Système de réservation (utilisateurs).
* Middleware de rôles (admin, user).

### ****Phase 3 : Développement Front-End (React/Blade)****

* **Interface Admin**
* Tableau de gestion des salles (ajout/modification/suppression).
* Validation des réservations (liste avec actions).
* **Interface Utilisateur**
* Calendrier interactif pour choisir un créneau.
* Liste des réservations personnelles (annulation possible).

### ****Phase 4 : Tests et Déploiement****

* **Tests**
* Unitaires (PHPUnit) : vérification des règles métier.

## **2. Estimation des Durées et Coûts**

### ****Durées****

| **Phase** | **Durée Estimée** |
| --- | --- |
| Analyse & Conception | 7 jours |
| Développement Back-End | 12 jours |
| Développement Front-End | 10 jours |
| Tests & Déploiement | 6 jours |
| **Total** | **35 jours** |

## **3. Outils et Technologies**

### ****Développement****

* **Back-End** : Laravel 12, MYSQL.
* **Front-End** : React + TailwindCSS.
* **Design** : Figma (prototypes), Miro (diagrammes UML).
* **Collaboration** : GitHub.

## **4. Livrables Attendus**

1. **Documentation**

* Diagrammes UML (cas d’usage, classes, séquences).
* Lien Figma des maquettes.

1. **Code et Déploiement**

* Dépôt GitHub avec code source.
* Application en production (URL d’accès).