Cahier des charges simplifié

1. Présentation du projet

**But :** Dans un contexte où la transformation digitale devient essentielle pour améliorer l'efficacité des structures éducatives, les établissements scolaires privés cherchent à moderniser leur mode de gestion. De nombreuses tâches administratives telles que l’inscription des élèves, le suivi des notes,

**Nom du projet : ÉcoleSmart**

**Description :**

EcolSmart : est une application web de gestion scolaire conçue pour les écoles privées, offrant une plateforme centralisée pour gérer les élèves, les professeurs, les cours, les emplois du temps, les notes et les absences.

Ce projet vise à digitaliser les processus administratifs et pédagogiques afin de renforcer l’organisation et la communication au sein de l’établissement**.**

**Objectif principal :**

Permettre à une école privée de gérer facilement ses élèves, professeurs, cours, emplois du temps, inscriptions, notes, etc .

**2. Problèmes à résoudre :**

**But :** Le but de l'application EcolSmart est d'offrir aux établissements scolaires privés une solution numérique centralisée leur permettant de gérer efficacement les élèves, les cours, les emplois du temps, les absences et les notes

**Liste des types d’utilisateurs :** Administrateur, Professeur et , Élève .

**Description brève de chacun :**

**Administrateur (Admin) :** L’administrateur gère l’application, crée et modifie des utilisateurs, et configure les paramètres scolaires. Il a un accès complet à toutes les fonctionnalités.

**Professeur:** Le professeur gère les notes et absences des élèves, consulte son emploi du temps et peut communiquer avec les élèves et les parents.

**Élève :** L’élève consulte son emploi du temps, ses notes et son historique scolaire, et peut communiquer avec ses enseignants et parents.

**4. Fonctionnalités attendues :**

**:** L'Administrateur gère l'ensemble de l'application, ajoute et modifie les utilisateurs, gère les classes et emplois du temps, et accède aux statistiques globales. Le Professeur peut consulter et modifier son emploi du temps, enregistrer les notes et absences des élèves, et communiquer avec eux via la messagerie interne. L'Élève consulte son emploi du temps, ses notes et son historique scolaire, signale ses absences et communique avec les enseignants via la messagerie.

**4. Contraintes techniques :**

**technologies imposées :** Laravel 12, MySQL / SQL, React, , CSS, Bootstrap

**Hébergement envisagé :**

**Mobilité / Responsiveness (PC,)**

Tableau des utilisateurs et rôles

|  |  |  |  |
| --- | --- | --- | --- |
| Nom de l'utilisateur | Description | Rôles et Permissions | Technologies et Liens avec l'implémentation |
| Administrateur | Gère l'ensemble de l'application et ses utilisateurs. | Accès complet à toutes les fonctionnalités.  - Gérer les utilisateurs, les classes, les matières et les emplois du temps. | **Laravel :** Middleware Admin  **React :** Routes protégées pour admin |
| Professeur | Enseignant responsable des notes et absences des élèves. | - Consulter/modifier son emploi du temps.  - Enregistrer et consulter les notes et absences des élèves | **Laravel**  Policy pour accès aux ressources du professeur.  **React :**  Routes conditionnelles pour les professeurs |
| Élève | Élève inscrit à l'école, consulte ses notes et son emploi du temps | Consulter ses notes et son emploi du temps.  - Signaler ses absences. | **Laravel :** Middleware pour protection des données personnelles  **React :** Routes accessibles pour les élèves |

**Liste des fonctionnalités attendues**

1. Présentation générale

Dans un contexte où la transformation digitale devient essentielle pour améliorer l'efficacité des structures éducatives, les établissements scolaires privés cherchent à moderniser leur mode de gestion. De nombreuses tâches administratives telles que l’inscription des élèves, le suivi des notes,

**Utilisateurs cibles**

* + - **Administrateur**
    - **Professeur**
    - **Élève**

.

2. Liste des fonctionnalités par type d’utilisateur

**Administrateur :** Responsable de la gestion globale de l’établissement (utilisateurs, classes, matières, emploi du temps…).

**Professeur :** Enseignant qui gère ses cours, ses élèves, les notes et les absences.

**Élève :** Apprenant inscrit dans l’établissement, qui consulte ses notes, son emploi du temps, etc …

3. User Stories

*Élève*: En tant qu’élève, je veux consulter mes notes, afin de suivre ma progression scolaire

*Professeur*: En tant que **professeur**, je veux ajouter les notes de mes élèves, afin d’évaluer leur performance*.*

* En tant que **professeur**, je veux voir mon emploi du temps, afin de planifier mes cours.

Administrateur : -> En tant qu’**administrateur**, je veux gérer les utilisateurs (élèves, professeurs, parents), afin de maintenir une base de données à jour.

* En tant qu’administrateur, je veux créer et modifier les classes et les matières, afin d’organiser l’année scolaire.

4. Priorisation des fonctionnalités

|  |  |  |  |
| --- | --- | --- | --- |
| Fonctionnalité | |  | | --- | |  |  |  | | --- | | **Utilisateur concerné** | |
| Authentification et gestion des rôles | Tous |
| Consultation des notes | Élève |
| Ajout des notes | |  | | --- | |  |  |  | | --- | | Professeur | |
| Gestion des utilisateurs (CRUD) | Administrateur |
| Gestion des classes et matières | Administrateur |
| Consultation de l’emploi du temps | Élève, Professeur |

*planification du projet*

*1. Liste des tâches à réaliser*

***=> Phase 1 – Analyse du projet***

* Définir les rôles : Administrateur, Professeur, Élève
* Identifier les fonctionnalités principales (ex : gestion des notes, absences, emploi du temps)
* Écrire des user stories simples pour chaque utilisateur
* Créer les premiers schémas UML (cas d’utilisation, classes)

***=> Phase 2 – Conception technique***

* Concevoir la base de données (tables + relations)
* Définir les modèles Laravel à créer (User, Classe, Note, etc.)
* Préparer la structure du projet React (pages, composants, navigation)
* Planifier les routes API (backend) et les routes React (frontend)

***=> Phase 3– Développement Back-end (Laravel 12)***

* *Mettre en place l’authentification (Sanctum, JWT ou Laravel Breeze)*
* *Créer les modèles + migrations (avec relations entre les tables)*
* *Développer les contrôleurs API pour gérer :*
* *les utilisateurs*
* *les classes*
* *les matières*
* *les notes*
* *Ajouter les permissions avec des policies ou middleware*
* *Testez routes(API) avec Postman*

***=> Phase 4– Développement front-end (REACT.Js)***

* Créer la page de connexion
* Afficher un tableau de bord différent selon le rôle (élève, prof, admin)
* Faire des appels API (avec axios ou fetch) pour afficher les données
* Mettre en place les routes sécurisées selon le rôle
* Ajouter du style avec Bootstrap ou css, pour une interface claire

***=> Phase 5– Tests***

* Tester la création, la lecture, la modification et la suppression de données
* Vérifier que chaque utilisateur ne voit que ce qu’il a droit de voir
* Corriger les bugs s’il y en a

***=> Phase 6– Déploiement***

* Configurer les variables d’environnement (.env)
* Tester la version en ligne avec des comptes test

*2. Estimation de la durée*

|  |  |
| --- | --- |
| *Taches* | *Durée estimée* |
| *- Définir rôles, user stories, fonctionnalités - Schémas UML* | *2 jours* |
| *- Base de données - Modèles Laravel - Structure React + routes* | *3 jours* |
| *Authentification - Modèles, Migrations - API (CRUD : notes, absences...) - Policies & Middleware* | *10 jours* |
| *Pages : Login, Dashboard, Notes, etc. - Intégration API - Routage + Rôles* | *8 jours* |
| *Stylisation avec Bootstrap - Amélioration responsive & UX* | *2 jours* |
| *Vérification des rôles - Tests fonctionnels API & UI* | *2 jours* |
| *Déploiement Laravel & React - Tests finaux en ligne*   * *- Préparation de la soutenance - Documentation et livrables* | *2 jours*  *1 jours* |

*4. Outils nécessaires au développement*

***Langages principaux****: PHP (Laravel 12) ,JavaScript (React), HTML5 , CSS3, SQL*

***Frameworks / Libs****: LARAVEL 12 (bak-end), React (front-end), Bootsrap (UI / Responsiveness)*

|  |
| --- |
| ***Database*** *:* |

*MySQL (via XAMPP / phpMyAdmin)  
- Modélisation avec MySQL Workbench*

|  |
| --- |
| ***IDE*** *:* |

*- Visual Studio Code (principal)  
- MySQL Workbench (modèle BDD)*

|  |
| --- |
| ***Versioning*** *:* |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
| *- Git (en local) - GitHub (pour héberger le projet et suivre l’évolution du code)* |

|  |
| --- |
| ***Serveur local*** *:* |

*-XAMPP (Apache + Mysql pour executer laravel en local )*

|  |
| --- |
| ***Outils de test*** *:*  *- Postman (pour tester les api Laravel )* |

*-Navigateur web (test des interfaces React )*