Cahier des charges pour l'Application de Gestion Scolaire en Python

1. Introduction

1.1 Objectif :

L'objectif général de cette application est de fournir un système complet de gestion scolaire, englobant la gestion des cours, l'établissement des emplois du temps, et le suivi de la présence des élèves. Cette application vise à faciliter la tâche des établissements scolaires en automatisant et en centralisant la gestion des informations cruciales liées aux activités académiques, offrant ainsi une solution efficace et organisée pour les enseignants, les administrateurs et les élèves.

1.2 Technologies utilisées :

L'application sera développée en utilisant le langage de programmation Python avec une base de données SQLite pour le stockage des données.

2. Fonctionnalités :

2.1 Gestion des élèves :

Ajouter un nouvel élève avec des détails tels que nom, prénom, age, classe_id etc.

Afficher la liste des élèves.

Rechercher un élève par son nom ou son ID.

Mettre à jour les détails d'un élève.

Supprimer un élève de la base de données.

2.2 Gestion des professeurs :

Ajouter un nouvel enseignant avec des détails tels que nom, prénom, age, matière_id, etc.

Afficher la liste des enseignants.

Rechercher un enseignant par son nom ou son ID.

Mettre à jour les détails d'un enseignant.

Supprimer un enseignant de la base de données.

2.3 Gestion des matières :

Ajouter une nouvelle matière avec un identifiant, un nom, etc.

Afficher la liste des matières.

Rechercher une matière par son identifiant ou son nom.

Mettre à jour les détails d'une matière.

Supprimer une matière de la base de données.

2.4 Gestion des classes :

Ajouter une nouvelle classe avec un identifiant, un nom, etc.

Afficher la liste des classes.

Rechercher une classe par son identifiant ou son niveau.

Mettre à jour les détails d'une classe.

Supprimer une classe de la base de données.

2.5 Gestion des cours :

Ajouter un nouveau cours avec des détails tels que id, matière, professeur, classe, jour, heure de début et heure de fin.

Afficher la liste des cours.

Rechercher un cours par matière, professeur, classe ou jour.

Mettre à jour les détails d'un cours.

Supprimer un cours de la base de données.

2.6 Gestion de l'emploi du temps :

Afficher l'emploi du temps pour une classe spécifique.

Afficher l'emploi du temps pour un professeur spécifique.

Afficher l'emploi du temps pour un jour spécifique.

2.7 Gestion de la présence :

Enregistrer la présence des élèves à un cours spécifique.

Afficher la liste des élèves présents et absents pour un cours donné.

Générer un rapport de présence pour une classe sur une période donnée.

3. Structure de la Base de Données Étendue

3.1 Tables :

Table "eleves" avec des colonnes telles que id, nom, prenom, date_naissance, classe_id, etc.

Table "prof" avec des colonnes telles que id, nom, prenom, matiere_id, etc.

Table "matiere" avec des colonnes telles que id, nom, etc.

Table "classes" avec des colonnes telles que id, niveau, etc.

Table "cours" avec des colonnes telles que id, matiere_id, prof_id, classe_id, jour, heure_debut, heure_fin.

Table "presence" avec des colonnes telles que id, eleve_id, cours_id, date, present.

Les colonnes "classe_id", "matiere_id", "prof_id" dans les tables "eleves", "cours", et "prof" respectivement sont des clés étrangères faisant référence aux tables correspondantes.

3.2 Relations :

Une relation entre la table "eleves" et "classes" via la colonne "classe_id".

Une relation entre la table "prof" et "matiere" via la colonne "matiere_id".

Des relations entre les tables "cours" et "matiere", "prof", "classes" via les colonnes "matiere_id", "prof_id", "classe_id" respectivement.

Une relation entre la table "presence" et "eleves", "cours" via les colonnes "eleve_id" et "cours_id" respectivement.

4. Interface Utilisateur

L'interface utilisateur sera mise à jour pour inclure des fonctionnalités liées aux cours, à l'emploi du temps, et à la présence.

5. Sécurité

Les requêtes SQL seront correctement paramétrées pour éviter les attaques par injection SQL. Un mécanisme d'authentification simple peut être mis en place si nécessaire.

6. Maintenance

La base de données peut être mise à jour ou modifiée sans perdre les données existantes. Des sauvegardes régulières peuvent être programmées.

7. Contraintes Techniques

L'application sera développée en utilisant Python 3.x avec SQLite comme système de gestion de base de données. Les bibliothèques Python telles que SQLite3 peuvent être utilisées pour interagir avec la base de données.

8. Livrables

Code source de l'application.

Documentation utilisateur mise à jour.

Documentation technique mise à jour.

9. Calendrier

Le développement de l'extension de l'application devrait être achevé d'ici [insérer la date ici].

10. Références

Toutes les références, bibliothèques tierces utilisées, et les tutoriels suivis pour le développement seront documentés.