

	<b>Reference</b>	<b>Esilv\Projets\Python</b>
	<b>Etudiants</b>	<b>Yobe GNADAME Mileina Malou Jean-Eudes Wandji</b>
	<b>Update Date</b>	<b>31/01/2026</b>
	<b>Project</b>	<b>MACSIN4A0425</b>

## Projet

### Application de Gestion des Étudiants

## Sommaire

- I. Introduction
- II. Présentation du projet et objectifs
- III. Analyse des besoins fonctionnels
- IV. Choix techniques et architecture de l'application
- V. Conception de la base de données
- VI. Fonctionnalités principales de l'application
- VII. Limites et perspectives d'amélioration
- VIII. Conclusion
- IX. Annexes

**ASSOCIATION LEONARD DE VINCI**

Etablissement d'Enseignement Supérieur Technique Privé

12, avenue Léonard de Vinci - 92916 PARIS LA DEFENSE CEDEX - N° Siren : 402 850 226 - N° Siret : 402 850 226 00018 - APE : 8542Z +33 1 41 16 70 00

## 1. Introduction

La gestion des données académiques constitue un enjeu central pour les établissements d'enseignement supérieur. Le suivi des étudiants, des inscriptions, des résultats, des absences et des structures pédagogiques nécessite des outils fiables, cohérents et faciles à utiliser. Lorsque ces informations sont gérées de manière manuelle ou à l'aide de solutions fragmentées, le risque d'erreurs augmente et le travail administratif devient rapidement lourd et inefficace.

Dans ce contexte, ce projet a pour objectif de concevoir une application desktop permettant de centraliser et structurer l'ensemble des données liées au parcours académique des étudiants. L'application vise à accompagner les acteurs pédagogiques et administratifs dans leurs tâches quotidiennes, tout en offrant une vision claire et globale de la situation académique.

Développée en Python, l'application propose des fonctionnalités essentielles telles que la gestion des étudiants et des inscriptions, le suivi des notes et des absences, la génération de documents officiels, ainsi que l'analyse des données à travers des indicateurs et tableaux de bord. Une attention particulière a également été portée à la sécurité et à l'intégrité des données, notamment par l'utilisation d'une base de données centralisée et d'un système d'authentification.

Ce projet s'inscrit ainsi dans une démarche pragmatique visant à répondre à des besoins concrets, tout en mettant en pratique les compétences techniques acquises au cours de la formation.

## 2. Présentation du projet et objectifs

Le projet consiste à développer une application desktop dédiée à la gestion académique des étudiants au sein d'un établissement d'enseignement supérieur. L'objectif principal est de regrouper, dans un outil unique, l'ensemble des informations nécessaires au suivi du parcours universitaire des étudiants, depuis leur inscription jusqu'à l'analyse de leurs résultats et de leur assiduité.

L'application a été pensée pour répondre à des besoins concrets rencontrés dans la gestion quotidienne : enregistrer et consulter les informations des étudiants, organiser les structures académiques (filières, niveaux, modules, enseignants), saisir et exploiter les notes, suivre les absences et produire des documents officiels tels que les relevés de notes et attestations de scolarité.

Au-delà des fonctionnalités de gestion, le projet vise également à faciliter l'analyse des données académiques. Des calculs automatiques (moyennes, taux d'absences) et des tableaux de bord permettent d'avoir une vision synthétique de la situation globale et d'identifier rapidement les cas nécessitant une attention particulière.

Enfin, un objectif important du projet est la fiabilité et la sécurité des données. Pour cela, l'application s'appuie sur une base de données SQLite centralisée, des contraintes d'intégrité et un système d'authentification garantissant un accès contrôlé aux fonctionnalités.

### 3. Analyse des besoins fonctionnels

L'application doit répondre à des besoins fonctionnels précis liés à la gestion académique des étudiants. Ces besoins ont été identifiés à partir des tâches couramment réalisées par les services administratifs et pédagogiques.

Tout d'abord, l'application doit permettre la **gestion complète des étudiants**. Cela inclut l'enregistrement des informations personnelles, la génération d'un matricule unique, la consultation rapide des dossiers et l'accès à une fiche détaillée regroupant l'ensemble du parcours académique.

Ensuite, un besoin essentiel concerne la **structuration académique**. L'application doit permettre de gérer les filières, les niveaux et les modules, ainsi que l'affectation des enseignants aux enseignements. Cette organisation est nécessaire pour garantir la cohérence des inscriptions et des évaluations.

La **gestion des inscriptions** constitue une autre fonctionnalité clé. Le système doit enregistrer les inscriptions des étudiants par année académique, filière et niveau, tout en conservant un historique fiable du parcours universitaire.

La **gestion des notes** doit permettre la saisie, la consultation et la modification des résultats, avec un contrôle des valeurs et un calcul automatique des moyennes. Toute modification doit être tracée afin d'assurer la transparence et la fiabilité des données.

Le suivi de l'**assiduité et des absences** est également un besoin important. L'application doit permettre d'enregistrer les absences, de gérer les justificatifs et de déclencher des alertes lorsque certains seuils sont dépassés.

Enfin, l'application doit offrir des fonctionnalités de **génération de documents** (relevés de notes, attestations) ainsi que des **outils d'analyse et de visualisation** sous forme de statistiques et de tableaux de bord, afin d'aider à la prise de décision.

### 4. Choix techniques et architecture de l'application

Le développement de l'application a été réalisé en Python, un langage adapté à la création d'outils de gestion grâce à sa simplicité, sa lisibilité et la richesse de son écosystème. Ce choix permet de maintenir un code clair, évolutif et facilement compréhensible.

L'interface graphique repose sur la bibliothèque **Tkinter**, enrichie par **ttkbootstrap**, afin de proposer une interface structurée, cohérente et agréable à utiliser. Ce choix garantit une

application desktop légère, sans dépendance à un navigateur web, et directement exploitable sur un poste local.

La persistance des données est assurée par une base de données **SQLite**. Cette solution a été retenue pour sa simplicité de mise en œuvre, sa fiabilité et son adéquation avec une application locale. L'ensemble des données académiques est centralisé dans une base unique, garantissant la cohérence et l'intégrité des informations.

L'architecture de l'application est organisée autour de trois éléments principaux :

- l'interface utilisateur, responsable de l'affichage et des interactions,
- la logique applicative, qui gère les traitements et les règles métier,
- la base de données, qui assure le stockage et la persistance des données.

Cette organisation permet une séparation claire des responsabilités et facilite la maintenance ainsi que l'évolution future de l'application.

## 5. Conception de la base de données

La base de données constitue le cœur de l'application, car elle assure la centralisation, la cohérence et la persistance de l'ensemble des informations académiques. Une base de données relationnelle **SQLite** a été mise en place afin de stocker de manière structurée les données relatives aux étudiants, aux structures académiques et au suivi pédagogique.

Le modèle de données repose sur plusieurs tables principales interconnectées. La table **etudiants** contient les informations personnelles et administratives des étudiants. Les tables **filières**, **niveaux** et **modules** permettent de structurer l'offre de formation et d'organiser les enseignements. Les inscriptions des étudiants sont gérées via la table **inscriptions**, assurant le suivi du parcours académique par année universitaire.

La gestion des résultats repose sur la table **notes**, qui enregistre les notes obtenues par les étudiants pour chaque module. Afin de garantir la transparence et la fiabilité des données, une table **notes\_audit** a été mise en place pour tracer toutes les opérations de création, de modification et de suppression des notes.

Le suivi de l'assiduité est assuré par la table **absences**, permettant d'enregistrer les absences, leur justification et d'effectuer des analyses statistiques. D'autres tables complètent le modèle, notamment celles liées aux enseignants, aux affectations pédagogiques et au calendrier académique.

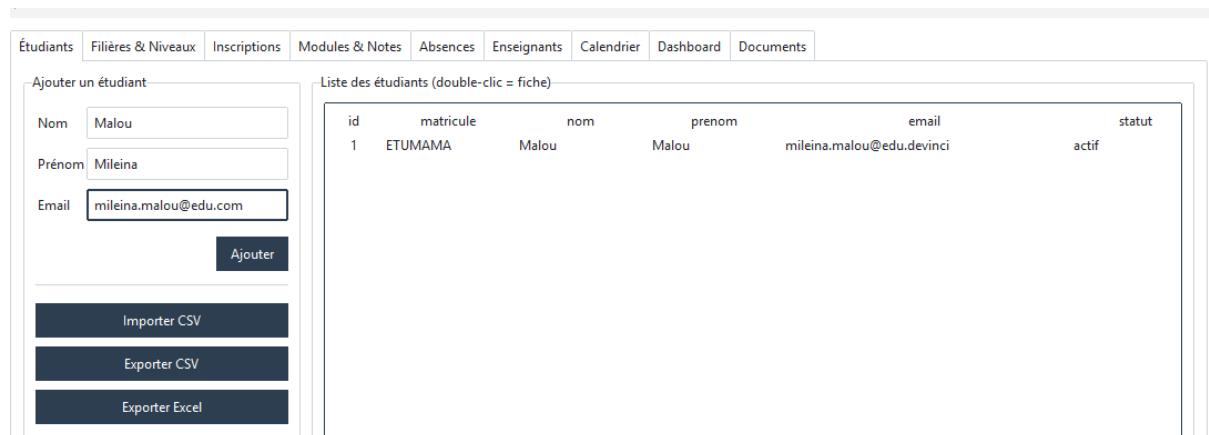
L'intégrité des données est garantie par l'utilisation de clés primaires, de clés étrangères et de contraintes relationnelles. Ces mécanismes permettent d'éviter les incohérences et assurent un lien fiable entre les différentes entités du système.

## 6. Fonctionnalités principales de l'application

L'application est structurée en onglets. Chaque onglet regroupe des actions ciblées afin de gérer les données académiques stockées dans la base SQLite.

## 1) Étudiants

- Ajout d'un étudiant avec : **nom, prénom, email**.
- **Génération automatique** d'un matricule unique (format ETU + lettres + compteur).
- Affichage de la liste des étudiants dans un tableau (Treeview).
- Ouverture d'une **fiche étudiant** au double-clic, contenant :
  - Identité (matricule, nom, prénom, email, statut)
  - Historique des inscriptions
  - Notes
  - Absences
- Import CSV d'étudiants (lecture de colonnes nom/prenom/email selon l'en-tête).
- Export de la liste des étudiants en **CSV** et en **Excel**.



id	matricule	nom	prenom	email	statut
1	ETUMAMA	Malou	Malou	mileina.malou@edu.devincl	actif

## 2) Filières & Niveaux

- Ajout de filières (code, nom) avec contrôle d'unicité du code.
- Ajout de niveaux (code, nom, ordre) avec contrôle du champ ordre (entier).
- Affichage des listes filières/niveaux.
- Alimentation des listes déroulantes utilisées dans les autres onglets.

Étudiants Filières & Niveaux Inscriptions Modules & Notes Absences Enseignants Calendrier Dashboard Documents

Filières

Code

Nom

**Ajouter filière**

1 - MACSIN4A1125 - Advanced Networks

Niveaux

Code

Nom

Ordre

**Ajouter niveau**

1 - 4 - MSC - ordre:1

### 3) Inscriptions

- Création d'une inscription via sélection :
  - étudiant, filière, niveau, année académique
- Affichage de l'historique des inscriptions dans un tableau.

Gestion des étudiants

Étudiants Filières & Niveaux Inscriptions Modules & Notes Absences Enseignants Calendrier Dashboard Documents

Nouvelle inscription

Étudiant

Filière

Niveau

Année académique

**Enregistrer inscription**

Historique des inscriptions

id	matricole	étudiant	filière	niveau	annee	statut
1	ETUMAMA	Malou Malou	MACSIN4A1125 - Advanced Networks	4 - MSC	2025-2026	inscrit

### 4) Modules & Notes

#### Modules

- Ajout d'un module avec : code, nom, coefficient, crédits (optionnel), filière (optionnel), niveau (optionnel).
- Affichage de la liste des modules.
- Alimentation des listes déroulantes utilisées pour les notes et absences.

## Notes

- Ajout d'une note pour un étudiant et un module :
  - contrôle de la valeur (0 à 20)
  - type d'évaluation (optionnel)
  - année académique (optionnel)
- Consultation des notes dans un tableau.
- Modification et suppression des notes via sélection.
- Calcul de la moyenne générale pondérée par les coefficients.
- Traçabilité des actions sur les notes (INSERT / UPDATE / DELETE) via la table notes\_audit, avec date et utilisateur connecté.

	etudiant	module	note	coef	annee
1	ETUMAMA - Malou Malou	123 - Python	19.0	2.0	2025-2026

## 5) Absences

- Enregistrement d'une absence pour un étudiant et un module :

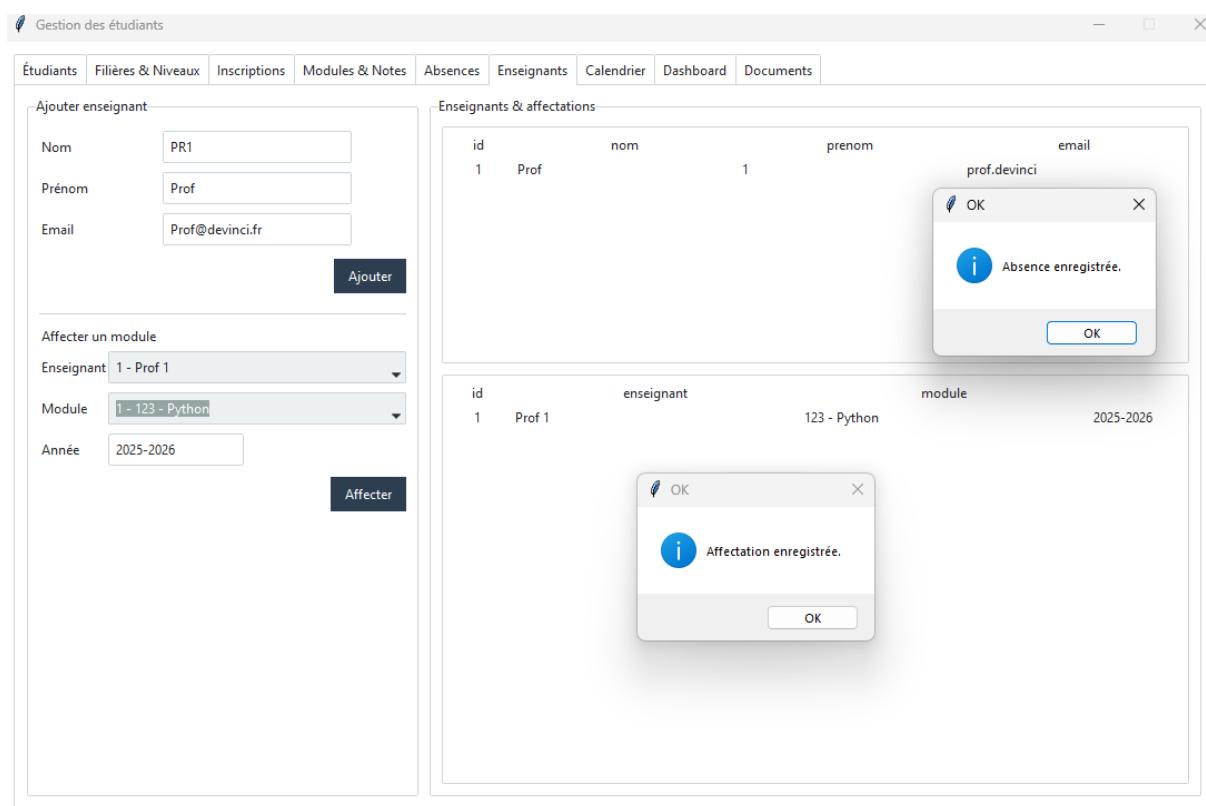
- date, justificatif (oui/non), motif (optionnel)
- Affichage des absences dans un tableau.
- Statistiques simples :
  - nombre total d'absences
  - taux "absences par étudiant" (total\_abs / nb\_etu)
- Système d'alerte : liste des étudiants ayant un nombre d'absences supérieur ou égal à un seuil défini.

Gestion des étudiants

Étudiants	Filières & Niveaux	Inscriptions	Modules & Notes	Absences	Enseignants	Calendrier	Dashboard	Documents												
Enregistrer une absence Étudiant: 2 - ETUMAMI - Malou Mileina Module: 1 - 123 - Python Date (YYYY-MM-DD): 2026-01-31 Justifiée: <input checked="" type="checkbox"/> Motif: Malade <input type="button" value="Enregistrer absence"/> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;">           OK              Absence enregistrée.             <input type="button" value="OK"/> </div>																				
Historique des absences <table border="1"> <thead> <tr> <th>id</th> <th>étudiant</th> <th>module</th> <th>date</th> <th>justifiée</th> <th>motif</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Malou Mileina</td> <td>123 - Python</td> <td>2026-01-31</td> <td>Oui</td> <td>Malade</td> </tr> </tbody> </table>									id	étudiant	module	date	justifiée	motif	1	Malou Mileina	123 - Python	2026-01-31	Oui	Malade
id	étudiant	module	date	justifiée	motif															
1	Malou Mileina	123 - Python	2026-01-31	Oui	Malade															
Analyse absences Seuil alerte (nb absences): <input type="text" value="3"/> <input type="button" value="Afficher alertes"/> Taux: 0.50 abs/étudiant   Alertes: -																				

## 6) Enseignants

- Ajout d'un enseignant : nom, prénom, email (optionnel).
- Affectation d'un enseignant à un module (année académique optionnelle).
- Affichage :
  - liste des enseignants
  - liste des affectations (enseignements).



## 7) Calendrier

- Ajout de semestres (code, libellé optionnel, date début, date fin).
- Ajout de périodes rattachées à un semestre (type, libellé optionnel, date début, date fin).
- Affichage des semestres et périodes dans deux tableaux.

Gestion des étudiants

Étudiants	Filières & Niveaux	Inscriptions	Modules & Notes	Absences	Enseignants	Calendrier	Dashboard	Documents										
Semestres																		
Code (ex: S1)	<input type="text"/>																	
Libellé	<input type="text"/>																	
Début (YYYY-MM-DD)	<input type="text"/>																	
Fin (YYYY-MM-DD)	<input type="text"/>																	
<b>Ajouter semestre</b>																		
<table border="1"> <thead> <tr> <th>id</th> <th>code</th> <th>lib</th> <th>deb</th> <th>fin</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>S1</td> <td>Semestre1</td> <td>2025-10-10</td> <td>2026-08-20</td> </tr> </tbody> </table>									id	code	lib	deb	fin	1	S1	Semestre1	2025-10-10	2026-08-20
id	code	lib	deb	fin														
1	S1	Semestre1	2025-10-10	2026-08-20														
Périodes (cours/examens/vacances)																		
Semestre	<input type="text"/> 1 - S1 (2025-10-10--2026-0																	
Type	<input type="text"/> Vacances																	
Libellé	<input type="text"/> Vacances de Noël																	
Début	<input type="text"/> 2025-12-22																	
Fin	<input type="text"/> 2026-01-04																	
<table border="1"> <thead> <tr> <th>id</th> <th>sem</th> <th>type</th> <th>I</th> </tr> </thead> <tbody> <tr> <td></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>									id	sem	type	I						
id	sem	type	I															

## 8) Dashboard

- Affichage d'indicateurs (KPI) :
  - nombre d'étudiants, modules, inscriptions, absences
- Top 5 des étudiants avec le plus d'absences.
- Bouton de rafraîchissement.

Gestion des étudiants

Étudiants: 2 | Modules: 1 | Inscriptions: 1 | Absences: 1

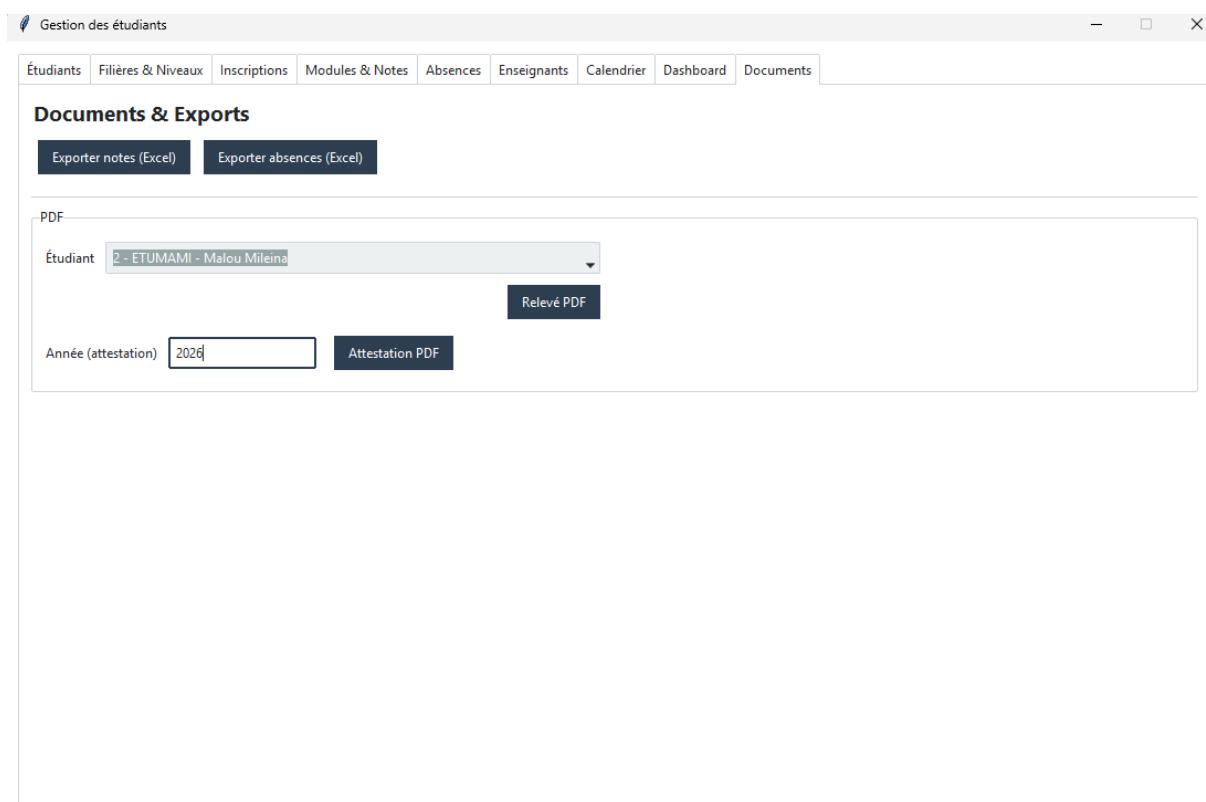
Top absences

matricole	etudiant	absences
ETUMAMI	Malou Mileina	1

Rafraîchir

## 9) Documents

- Export Excel :
  - notes (liste complète)
  - absences (liste complète)
- Génération PDF :
  - relevé de notes d'un étudiant avec calcul de moyenne générale pondérée
  - attestation de scolarité d'un étudiant pour une année académique donnée



## 7. Synthèse

L’application développée permet de couvrir l’ensemble des fonctionnalités essentielles décrites dans la consigne du projet Application de gestion des étudiants. Les différents modules sont opérationnels et interconnectés autour d’une base de données SQLite centralisée.

La gestion des étudiants est fonctionnelle, depuis l’ajout manuel ou l’import CSV jusqu’à la consultation détaillée de la fiche étudiant. Les inscriptions, notes et absences sont correctement reliées à chaque étudiant, assurant une vision cohérente de son parcours académique.

La gestion des structures académiques (filières, niveaux, modules, enseignants) permet de configurer l’organisation pédagogique de manière simple et fiable. Les inscriptions et les évaluations reposent sur ces structures, ce qui garantit la cohérence des données enregistrées.

Le module de gestion des notes permet la saisie, la modification et la suppression des résultats avec un contrôle des valeurs. Le calcul automatique des moyennes est effectué à partir des coefficients des modules. Les modifications sont tracées grâce au système de journalisation des notes, assurant la transparence des actions réalisées.

Le suivi des absences est opérationnel et permet d'identifier les étudiants présentant un nombre élevé d'absences grâce à un mécanisme d'alerte basé sur un seuil défini. Des indicateurs globaux sont également affichés afin de donner une vision synthétique de l'assiduité.

Le tableau de bord fournit des indicateurs clés tels que le nombre total d'étudiants, de modules, d'inscriptions et d'absences, ainsi qu'un classement des étudiants les plus absents. Ces éléments facilitent l'analyse rapide de la situation académique.

Enfin, l'application permet la génération de documents officiels. Les relevés de notes et attestations de scolarité peuvent être exportés au format PDF, et les données académiques (notes et absences) peuvent être exportées au format Excel pour un usage externe.

## 8. Limites et perspectives d'amélioration

Bien que l'application réponde aux objectifs principaux du projet, certaines limites peuvent être identifiées. Tout d'abord, l'application repose sur une architecture monolithique dans laquelle l'interface, la logique métier et l'accès à la base de données sont regroupés dans un même projet. Cette approche est adaptée dans un cadre académique, mais peut devenir plus difficile à maintenir pour une application de grande taille.

L'interface graphique, développée avec Tkinter, est fonctionnelle et stable, mais reste limitée en termes de personnalisation et d'ergonomie avancée par rapport à des solutions web ou à des frameworks graphiques plus modernes.

Par ailleurs, l'application ne dispose pas de tests automatisés, ce qui limite la détection précoce d'éventuelles régressions lors des évolutions du code. De plus, l'application est conçue pour un usage local et mono-utilisateur, sans gestion de concurrence ni accès distant.

Plusieurs pistes d'amélioration peuvent être envisagées. L'application pourrait évoluer vers une architecture plus modulaire, facilitant la maintenance et l'ajout de nouvelles fonctionnalités. Une version web, reposant sur une API, permettrait un accès multi-utilisateurs et distant. L'ajout de graphiques statistiques plus avancés et d'un système de gestion des rôles plus détaillé pourrait également enrichir l'outil.

## 9. Conclusion

Ce projet a permis de concevoir et de développer une application desktop complète de gestion des étudiants, couvrant l'ensemble du cycle académique, de l'inscription au suivi des résultats et de l'assiduité. L'application centralise les données académiques dans une base de données

SQLite cohérente et sécurisée, et propose des fonctionnalités adaptées aux besoins administratifs et pédagogiques.

Au-delà de l'aspect fonctionnel, ce projet a permis de mettre en pratique des compétences essentielles en programmation Python, en conception de bases de données relationnelles, en développement d'interfaces graphiques et en structuration d'une application logicielle. Il constitue une synthèse concrète des connaissances acquises au cours de la formation et une base solide pour des évolutions futures.