**Documentation du Projet de Gestion des Notes**

**1. Introduction**

Ce projet est une application web permettant la gestion des notes. Les utilisateurs peuvent créer, modifier et supprimer des notes. L'application fournit également une fonctionnalité d'affichage des notes avec pagination et permet l'exportation des notes en format PDF ou Excel.

**2. Architecture de l'application**

L'architecture de l'application suit une structure MVC (Modèle-Vue-Contrôleur), où :

**Modèle (Model)** : Gère la base de données et la logique de stockage des données. Le modèle principal est la table Note, qui contient les informations sur chaque note (titre et valeur de la note).

**Vue (View)** : Représente l'interface utilisateur de l'application. La vue est réalisée à l'aide de templates HTML, et le CSS est utilisé pour personnaliser l'apparence de l'application.

**Contrôleur (Controller)** : Gère les interactions entre le modèle et la vue. Le contrôleur contient la logique de l'application pour ajouter, modifier, supprimer et afficher les notes, ainsi que pour exporter les notes vers PDF et Excel.

**2.1 Structure des Dossiers**

Voici la structure des dossiers du projet :

note\_manager /

app.py # Point d'entrée de l'application

utlis.py # Configuration de l'application

models.py # Modèles de base de données

templates/ # Dossier contenant les fichiers HTML

* + - index.html # Page d'accueil
    - modifier\_note.html #Interface pour les modification des note
    - gestion\_notes.html # Interface de gestion des notes
    - afficher\_notes.html # Interface d'affichage des notes
    - imprimer\_notes.pdf #Interface pour l’exportation pdf

static/

css /

* + - * afficher\_notes.css
      * gestion\_notes.css
      * imprimer\_pdf.css
      * index.css
      * modifier\_note.css

migrations/ # Dossier contenant les fichiers de migration de la base de données

**3. Technologies utilisées**

**Flask** : Framework web léger pour le développement d'applications Python.

**SQLAlchemy** : ORM (Object-Relational Mapping) pour la gestion de la base de données.

**SQLite** : Base de données relationnelle légère, utilisée pour stocker les notes.

**Flask-Migrate** : Extension pour gérer les migrations de base de données.

**WTForms** : Bibliothèque pour la gestion des formulaires HTML.

**Jinja2** : Moteur de template utilisé par Flask pour générer des fichiers HTML.

**Pandas** : Bibliothèque Python utilisée pour exporter les données en format Excel.

**ReportLab** : Bibliothèque Python pour générer des fichiers PDF.

**4. Installation**

Crée un environnement virtuel et active-le :

venv\Scripts\activate # Sur Windows

Installe les dépendances :

Assure-toi que la base de données est prête (en utilisant Flask-Migrate ou en supprimant et recréant la base de données si nécessaire).

Pour les migrations :

flask db init

flask db migrate -m "Initial migration"

flask db upgrade

Lance l'application Flask :

python app.py

L'application sera accessible à l'adresse http://127.0.0.1:5000/.

**5. Guide d'utilisation**

**5.1 Page d'accueil**

Lorsque l'utilisateur accède à l'application, il arrive sur la **page d'accueil**, qui affiche les différentes options de gestion des notes.

**5.2 Gestion des Notes**

Sur la **page de gestion des notes**, l'utilisateur peut :

**Ajouter une note** : Un formulaire permet de saisir un titre et une valeur pour la note.

**Modifier une note** : En cliquant sur une note existante, l'utilisateur peut la modifier en remplissant à nouveau le formulaire.

**Supprimer une note** : Chaque note affichée possède un bouton pour la supprimer.

Le formulaire de gestion des notes est créé à l'aide de **WTForms**, ce qui permet de valider facilement les données.

**5.3 Affichage des Notes**

Sur la **page d'affichage des notes**, l'utilisateur peut :

Voir une liste paginée des notes disponibles dans la base de données. Les notes sont affichées sous forme de tableau avec leur titre et leur valeur.

Naviguer à travers les pages de résultats grâce à la pagination (par exemple, 10 notes par page).

**5.4 Exporter les Notes**

L'utilisateur peut exporter les notes sous deux formats différents :

**PDF** : Un bouton permet d'exporter les notes en format PDF à l'aide de **ReportLab**.

**Excel** : Un autre bouton permet d'exporter les notes vers un fichier Excel en utilisant **Pandas**.

**5.5 Pagination**

Pour les listes de notes longues, la pagination est implémentée pour diviser l'affichage des notes en plusieurs pages, facilitant ainsi la navigation.

python

# Exemple d'utilisation de pagination dans Flask

page = request.args.get('page', 1, type=int)

notes = Note.query.paginate(page, 10, False)

**6. Architecture de la Base de Données**

La base de données utilise un modèle simple pour stocker les informations sur les notes. Le modèle principal est la classe Note, définie comme suit dans models.py :

from flask\_sqlalchemy import SQLAlchemy  
  
db = SQLAlchemy()  
  
class Note(db.Model):  
 id = db.Column(db.Integer, primary\_key=True)  
 titre = db.Column(db.String(100), nullable=False)  
 note = db.Column(db.Float, nullable=False)

**Table Note**

| **Colonne** | **Type** | **Description** |
| --- | --- | --- |
| id | INTEGER | Identifiant unique de la note |
| titre | VARCHAR(100) | Titre de la note |
| note | FLOAT | Valeur de la note |

**7. Déploiement**

**7.1 Déploiement local**

Pour déployer l'application localement, il suffit de suivre les étapes d'installation et de lancer le serveur Flask avec la commande python app.py.

**8. Conclusion**

Ce projet fournit une application complète de gestion des notes avec une interface utilisateur simple et intuitive. Les fonctionnalités incluent la gestion des notes, l'affichage avec pagination, et l'exportation vers des formats PDF et Excel. L'application peut être étendue pour inclure des fonctionnalités supplémentaires selon les besoins.