

Mise en place de chaque app

NOTES

1. Description générale de l'app **notes**

L'app **notes** constitue le **cœur fonctionnel** du projet **KeepNotes**.

Elle gère la **création, la modification, la suppression et la consultation des notes**, ainsi que des **fonctionnalités annexes** comme les **tags (étiquettes)** et les **rappels**.

Objectif :

Permettre à un utilisateur authentifié de gérer ses notes personnelles de manière intuitive, organisée et visuelle.






2. Fonctionnalités principales à développer

N°	Fonctionnalité	Description	Niveau	Priorité
1	Créer une note	L'utilisateur peut créer une nouvelle note avec un titre et un contenu.	Basique	● Élevée
2	Lister les notes	Afficher toutes les notes de l'utilisateur connecté (triées par date ou tag).	Basique	● Élevée
3	Modifier une note	Permettre à l'utilisateur de mettre à jour le contenu ou le titre d'une note.	Basique	● Élevée
4	Supprimer une note	Supprimer définitivement ou archiver une note.	Basique	● Élevée

N°	Fonctionnalité	Description	Niveau	Priorité
5	Afficher les détails d'une note	Consulter une note spécifique dans une vue détaillée.	Basique	🟢 Moyenne
6	Gérer les tags	Ajouter des tags (étiquettes) pour regrouper les notes par thème.	Intermédiaire	🟡 Moyenne
7	Archiver / Restaurer une note	Marquer une note comme archivée sans la supprimer.	Intermédiaire	🟡 Moyenne
8	Rappels / Notifications	Définir une date/heure de rappel pour une note.	Avancée	🔴 Faible
9	Recherche / Filtrage	Rechercher une note par mot-clé ou tag.	Avancée	🔴 Faible

3. Ordre logique de développement

L'ordre de développement est essentiel pour ne pas se perdre 📌

Étape	Objectif	Explication
	Modéliser les entités (<code>Note</code> , <code>Tag</code> , <code>Reminder</code>)	C'est le cœur du M dans MVT. Tu définis la structure de tes données.
	Faire les migrations et tester dans l'admin	Tu confirmes que ta base de données est bien alignée.
	Créer les vues (Views)	Commence par des vues simples : <code>list</code> , <code>detail</code> , <code>create</code> , <code>update</code> , <code>delete</code> .
	Créer les templates correspondants	Pour chaque vue, un template HTML : <code>note_list.html</code> , <code>note_detail.html</code> , etc.
	Configurer les routes (urls.py)	Relier tes vues à des chemins d'accès clairs.

Étape	Objectif	Explication
6	Ajouter les tags, recherche et rappels	Une fois la base solide, tu enrichis les fonctionnalités.

4. Modèles (Models) – conception conceptuelle

a) Note

- **Rôle** : Représente une note individuelle.
- **Champs essentiels** :
 - `title` : Titre de la note
 - `content` : Contenu textuel
 - `created_at` , `updated_at` : Dates automatiques
 - `is_archived` : Booléen pour l'archivage
 - `owner` : Lien vers l'utilisateur (clé étrangère vers `User`)

b) Tag

- **Rôle** : Sert à catégoriser les notes.
- **Champs** :
 - `name` : Nom du tag
 - `color` : Couleur d'affichage
 - `notes` : Relation ManyToMany avec `Note`

c) Reminder

- **Rôle** : Gère la planification de rappels pour une note.
- **Champs** :
 - `reminder_date` : Date/heure du rappel
 - `is_active` : Activation du rappel
 - `note` : Relation OneToOne avec `Note`

Relations entre modèles :

- `User` 1—* `Note`

- **Note** — **Tag**
- **Note** 1—1 **Reminder**

🔧 5. Vues (Views) – logique de traitement

Vue	Type	Description	Fichier Template
NoteListView	Lecture	Affiche toutes les notes actives de l'utilisateur	note_list.html
NoteDetailView	Lecture	Affiche le contenu d'une note spécifique	note_detail.html
NoteCreateView	Création	Formulaire de création d'une note	note_form.html
NoteUpdateView	Modification	Formulaire d'édition d'une note	note_form.html
NoteDeleteView	Suppression	Confirme la suppression d'une note	note_confirm_delete.html
TagListView	Lecture	Liste tous les tags existants	tag_list.html
ReminderListView	Lecture	Liste les rappels actifs	reminder_list.html

🧠 Logique MVT :

Les vues sont la passerelle entre le modèle (les données) et le template (l'affichage).

Elles reçoivent la requête, interagissent avec les modèles, et renvoient une page HTML.

🎨 6. Templates (affichage HTML)

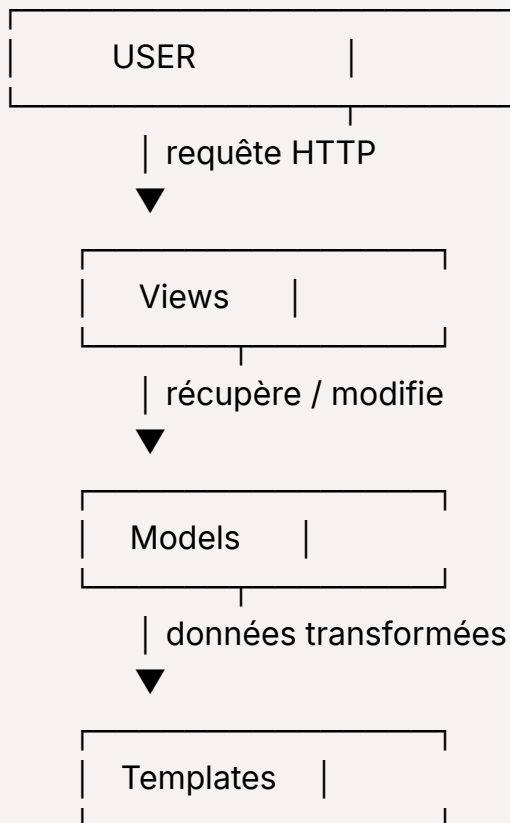
Fichier	Rôle
note_list.html	Affiche la liste des notes avec bouton "+ Nouvelle note"
note_detail.html	Affiche une note spécifique avec ses tags et rappel
note_form.html	Formulaire réutilisé pour créer / modifier une note
note_confirm_delete.html	Page de confirmation avant suppression
tag_list.html	Liste de tous les tags et leurs couleurs

Fichier	Rôle
reminder_list.html	Affiche les rappels planifiés

💡 Les templates hériteront tous de `core/templates/base.html` via :

```
{% extends 'base.html' %}
{% block content %}
...
{% endblock %}
```

7. Résultat attendu du MVT complet



C'est cette logique que tu vas **apprendre à maîtriser concrètement** avec l'app `notes`.