Prototype Django transactionnel – Allons plus loin !

Objectif

Ce travail constitue la suite directe du premier projet **Prototype Django transactionnel**. Vous devez poursuivre le développement de votre application Django en intégrant de nouvelles fonctionnalités avancées.

Les sous-objectifs :

- Approfondir vos compétences avec le module admin de Django.
- Mettre en place la gestion de fichiers médias (images).
- Intégrer des mécanismes professionnels : signaux, pagination, traduction, gestion d'utilisateurs et groupes, permissions et fixtures.
- Construire un site web plus complet, proche d'une application en production.

Consignes générales

- Travail individuel.
- Vous devez continuer à travailler sur l'application que vous avez réalisée pour le travail précédent.
- Si vous aviez déjà exploré certaines fonctionnalités en avance, assurez-vous que celles-ci sont correctement mises en place et fonctionnelles.
- Comme toujours : respect de la structure Django, code clair et bien organisé, interface utilisateur soignée.

Exigences obligatoires

1. Module « admin » personnalisé

- Personnalisez l'affichage de tous vos modèles pour rendre le module « admin » plus complet : intuitif et ergonomique.
    - Utilisez entre autres (admin.py) : *list_display, search_fields, list_filter, ordering, readonly_fields, exclude, form, etc*.
    - Lorsque pertinent, affichez des images, des colonnes personnalisées, …

2. Gestion d'images avec *ImageField*

- Ajoutez au moins un champ *ImageField* dans un de vos modèles.
- Adaptez vos formulaires et vos *templates,* ainsi que le module « admin » :
    - Affichage d'un *thumbnail* dans la liste admin.
    - Affichage d'un aperçu (*preview*) dans le formulaire d'édition admin.

3. Signaux (*signals.py*)

- Implémentez des signaux pour gérer automatiquement les fichiers images dans le dossier media/ :
    - Suppression physique du fichier lors de la suppression d'un objet.
    - Remplacement correct du fichier lors d'une mise à jour.

4. Pagination

- Ajoutez de la pagination (via *Paginator*) sur au moins une page listant des objets de votre base de données.

5. Gestion des utilisateurs

- Intégrez un système complet basé sur le modèle d'utilisateurs de Django :
    - Création de compte.
    - Modification du profil et du mot de passe.
    - Connexion et déconnexion.

6. Groupes et permissions

- Créez au moins 2 groupes d'utilisateurs ayant des permissions différentes sur votre site web (*view, add, change, delete*).
- Adaptez vos *templates* pour afficher ou masquer certains éléments (liens, boutons, etc.) selon les permissions.
- Assurez-vous que seuls les utilisateurs autorisés ont accès à vos différentes vues (*views.py)*.

7. Traduction du site avec django-modeltranslation

- Installez et configurez *django-modeltranslation*.
- Assurez-vous de bien comprendre comment mettre le tout en place.
- Traduisez :
  - Les champs pertinents de vos modèles.
  - Les éléments statiques de vos *templates*.
- Votre site doit être consultable en français (par défaut) et en anglais.

8. Fixtures pour données initiales

- Générez un ou plusieurs fichiers fixtures (format JSON).
- Ces fichiers doivent permettre d'importer des données de départ afin que votre application soit fonctionnelle dès le premier déploiement.

Exigences optionnelles (pour aller plus loin)

- Définir un ou plusieurs *context_processors* pour partager des données dans l'ensemble de vos *templates*.

- Explorer les vues génériques spécialisées : *ListView, DetailView, CreateView, UpdateView, DeleteView, …*

- Rédiger des tests unitaires pour valider vos modèles, vos vues et vos formulaires.

Vous devez remettre sur Léa :

- **Votre code source** (sans votre dossier « .git » et « .venv »).
  - N'oubliez pas de générer votre fichier **requirements.txt**!

Travail terminé