Scripts de peuplement des données - Application Inventory

Ce dossier contient les scripts pour peupler la base de données de l'application `inventory` avec des données de test basées sur les données simulées du frontend.

Scripts disponibles

1. `populate_inventory.py`

Script autonome qui peut être exécuté via le shell Django.

Usage :

```
cd backend
python manage.py shell
>>> exec(open('inventory/scripts/populate_inventory.py').read())
```

2. Commande Django Management

Une commande Django management est également disponible pour une utilisation plus simple.

Usage :

```
cd backend

# Peupler avec les données existantes (sans suppression)
python manage.py populate_data

# Peupler en supprimant d'abord toutes les données existantes
python manage.py populate_data --clear
```

Données créées

Le script peuple les tables suivantes :

Catégories (Category)

- **Sandales Homme** (type: sandales)
- **Sandales Femme** (type: sandales)
- **Sabots Homme** (type: sabots)
- **Sabots Femme** (type: sabots)
- **Mules Homme** (type: mules)
- **Mules Femme** (type: mules)

- **Chaussures Homme** (type: sandales)
- **Espadrilles Homme** (type: espadrilles)

Produits (Product)

Exemples de produits créés :

- **Sandales Cuir Élégantes** (SAN-H-001) - 189,90 DH
- **Sandales Bohème** (SAN-F-001) - 127,92 DH (en promotion)
- **Sabots Traditionnels** (SAB-H-001) - 149,90 DH
- **Mules Casual** (MUL-H-001) - 139,90 DH
- **Espadrilles Méditerranéennes** (ESP-H-001) - 119,90 DH

Stock (Stock)

Chaque produit est créé avec un stock associé contenant :

- Code article (référence du produit)
- Nom de l'article
- Couleur
- Quantité disponible (variable selon le produit)

Images de produits (ProductImage)

Chaque produit est créé avec une image principale (provenant d'URLs Pexels).

Structure des données

Les données respectent la structure des modèles Django :

```
Category:
- name: str (nom de la catégorie)
- type: str (choix parmi: sandales, mules, sabots, etc.)
- description: str
- slug: str (généré automatiquement)

Product:
- name: str
- reference: str (unique, ex: SAN-H-001)
- category: ForeignKey(Category)
- gender: str (homme/femme/unisexe)
- price: Decimal
- old_price: Decimal (optionnel, pour les promotions)
- description: str
- available_sizes: str (tailles séparées par virgules)
- colors: str
- material: str
- is_featured: bool
- stock: OneToOne(Stock)
```

```
Stock:
- article_code: str (référence du produit)
- article_name: str
- color: str
- quantity_available: int

ProductImage:
- product: ForeignKey(Product)
- image: str (URL de l'image)
- alt_text: str
- is_main: bool
- order: int
```

Prérequis

- Django configuré et applications installées
- Base de données configurée
- Modèles `inventory` migrés

Sécurité

⚠ **Attention** : L'option `--clear` supprime toutes les données existantes des tables :

- ProductImage
- Product
- Stock
- Category

Utilisez cette option uniquement en développement ou après avoir fait une sauvegarde.

Source des données

Les données sont basées sur le fichier `Frontend/src/data/mockData.ts` et adaptées pour respecter la structure des modèles Django backend.

Logs et debugging

Le script affiche des logs détaillés :

- ☑ Succès des créations
- 🔁 Éléments déjà existants
- ❌ Erreurs avec détails

Extensibilité

Pour ajouter de nouveaux produits, modifiez la liste `products_data` dans la fonction `create_products_and_stock()` en respectant la structure existante.