Mini-PIM Excel Manager v1.4

Présentation

Mini-PIM Excel Manager est un outil web autonome (Single Page Application hybride) conçu pour faciliter la création, l'enrichissement et la modification en masse de produits destinés à être importés dans un système e-commerce (notamment Magento).

Il permet de s'affranchir de la lourdeur d'un tableur classique en offrant une interface de type formulaire dynamique, qui s'adapte en temps réel aux **Jeux d'attributs (Attribute Sets)** et aux **Catégories** de votre boutique.

L'outil fonctionne en "hors-ligne" (côté navigateur) pour une fluidité maximale, et communique avec le serveur uniquement pour la sauvegarde automatique et le chargement initial des dictionnaires d'attributs.

Architecture des Fichiers

L'application repose sur un écosystème de 3 fichiers principaux (et des dossiers de stockage).

1. `index.php` (L'Interface Utilisateur)

C'est le cœur de l'application. Bien qu'il ait une extension `.php`, il contient principalement du **HTML, du CSS et du Javascript**.

- **Rôle :** Affiche le formulaire, gère l'import/export Excel (`SheetJS`), gère le Drag & Drop (`SortableJS`), la saisie semi-automatique (`TomSelect`), le filtrage du tableau, l'édition en masse, et la duplication en cascade.
- **Particularité :** Il lit les fichiers JSON du cache pour afficher instantanément les bons champs selon le Set d'attributs choisi.

2. `sync.php` (Le Moteur de Synchronisation Magento)

Ce script est appelé manuellement (ex : tâche planifiée CRON ou clic bouton) pour mettre à jour la base de données locale.

- **Rôle :** Se connecte à l'API SOAP de Magento pour télécharger l'arborescence des catégories, la liste des jeux d'attributs et toutes les options possibles (menus déroulants) pour chaque attribut.
- **Sortie :** Il génère et met à jour les petits fichiers `.json` statiques dans le dossier `/cache/`. Cela évite de solliciter l'API Magento à chaque affichage de l'interface.

3. `save_backup.php` (Le Gestionnaire de Sauvegarde)

Script appelé silencieusement en arrière-plan (AJAX) par `index.php` à chaque modification apportée par l'utilisateur.

- **Rôle :** Reçoit l'état complet du tableau (les produits, les colonnes utilisées, l'ordre des attributs) en morceaux (*chunks*) pour contourner les limites de taille d'envoi du serveur.
- **Sortie :** Reconstruit le fichier `current_session.json` dans le dossier `/backups/`. Si l'utilisateur ferme l'onglet par erreur, sa session est restaurée à la prochaine ouverture.

Fonctionnement Détaillé

A. Chargement de l'application (`init`)

1. Le navigateur lit les fichiers `/cache/categories_level1.json` et `/cache/sets_list.json` pour remplir les premiers menus déroulants.
2. Il interroge `/backups/current_session.json`. Si une sauvegarde récente existe, il propose à l'utilisateur de restaurer son travail.

B. Création / Édition de Produit

- Lors du choix d'un "Jeu d'attributs", `index.php` charge le fichier `/cache/set_XX.json` correspondant et construit dynamiquement les champs du formulaire.
- **Le Détective d'Attributs :** Si une valeur tapée manuellement (ou importée) n'existe pas dans le dictionnaire Magento, le script la surligne en jaune pâle/orange pour indiquer qu'elle devra être créée côté Magento. L'utilisateur peut l'utiliser sans être bloqué.
- Le SKU s'auto-incrémente (`new_product_1`, `new_product_2`...) pour garantir l'unicité lors de la création de nouveaux produits de A à Z.

C. Import / Export Excel

- **Import :** Lit un fichier `.xlsx` ou `.csv`. Fusionne les nouvelles lignes avec le tableau existant (en ignorant les doublons basés sur le SKU). Les colonnes inconnues de Magento sont gérées dans un bloc "Données Hors-Set".
- **Export :** Exporte uniquement les colonnes réellement utilisées dans la session en cours. Conserve le surlignage jaune pour les attributs personnalisés/orphelins.

D. Édition en Masse (Bulk Edit)

Permet de sélectionner plusieurs lignes du tableau, puis via une modale, d'écraser simultanément :

- Les statuts système (Activé/Désactivé, Visibilité).
- Le jeu d'attributs ou la catégorie.

- N'importe quel attribut spécifique (en ajoutant dynamiquement une ligne d'attribut dans la modale).

E. Attributes Filtering

L'utilisateur peut réorganiser visuellement les champs techniques en fonction de filtres de sélection, et les déplacer en glissant les poignées (). Cet ordre est sauvegardé dans le navigateur (`localStorage`) par Jeu d'attribut, assurant que l'interface s'affiche toujours selon les préférences de l'utilisateur.

Structure des Dossiers Requise

Pour que l'application fonctionne, l'arborescence suivante doit exister (les dossiers doivent avoir les droits d'écriture pour PHP) :

```
/mon-projet
 |-- index.php
 |-- sync.php
 |-- save_backup.php
 |-- /cache/           <-- Fichiers .json générés par sync.php
 |    |-- sets_list.json
 |    |-- categories_level1.json
 |    `-- set_4.json, set_9.json...
 `-- /backups/         <-- Sauvegardes de session
      `-- current_session.json
```

Installation & Utilisation

> **Note :**
> Pour le moment cet algo est opérationnel que sur des IDE Python comme VS Code. On tentera plus tard l'implémentation FileZilla.

1. **Configuration API :** Éditez le fichier `sync.php` et renseignez vos identifiants d'API SOAP Magento (`$apiUser`, `$apiKey`).
2. **Synchronisation Initiale :** Ouvrez la page `sync.php` dans votre navigateur et patientez. Ce processus peut prendre quelques minutes. Vérifiez que le dossier `/cache/` s'est bien rempli.
3. **Démarrage :** Ouvrez `index.php`. L'interface doit s'afficher et le badge de synchronisation (en bas à gauche) doit indiquer la date de la dernière mise à jour réussie.
4. **Sauvegarde :** L'outil sauvegarde automatiquement. Pensez toutefois à utiliser le bouton **Exporter Excel** en fin de session pour récupérer votre fichier de travail final.