Extracteur d'E-mails Parcoursup - Documentation Professionnelle

Objectif

Ce script permet d'extraire automatiquement des adresses e-mail de contact depuis une liste d'URLs Parcoursup présentes dans un fichier Excel.

Stratégie d'extraction :

1. **PRIORITÉ** : Recherche d'abord les e-mails sur la fiche Parcoursup (rubrique "Contacter et échanger avec l'établissement")
2. **FALLBACK** : Si aucun e-mail n'est trouvé sur Parcoursup, va chercher sur le site officiel de l'établissement
3. **RÉSULTAT** : Sauvegarde les e-mails trouvés dans un nouveau fichier Excel avec statut détaillé

Processus d'extraction détaillé

Pour chaque URL Parcoursup dans votre fichier Excel :

1. **Accès à la fiche Parcoursup** : Le script ouvre l'URL de la formation
2. **Recherche sur Parcoursup** : Cherche la section "Contacter et échanger avec l'établissement"
3. **Extraction Parcoursup** : Tente d'extraire les e-mails (général, pédagogique, administratif)
4. **Si e-mails trouvés** : Sauvegarde et passe à l'URL suivante
5. **Si aucun e-mail sur Parcoursup** :
   - Cherche le lien du site officiel de l'établissement
   - Va sur le site officiel
   - Extrait les e-mails depuis le site officiel
6. **Sauvegarde** : Enregistre tous les résultats avec statut détaillé

Fonctionnalités principales

- **Lecture automatique** d'un fichier Excel listant les URLs des fiches formations Parcoursup
- **Double stratégie d'extraction** :
  - **Priorité** : E-mails depuis les pages Parcoursup
  - **Fallback** : E-mails depuis le site officiel de l'établissement
- **Catégorisation intelligente** : Distingue les contacts général, pédagogique et administratif
- **Sauvegarde détaillée** dans un nouveau fichier Excel :
  - Chaque ligne contient : ligne d'origine, URL, mails trouvés, statut du traitement, horodatage
- **Système de pauses automatiques** pour limiter la sollicitation du serveur

- **Gestion robuste des erreurs** (poursuite même en cas d'échec partiel)

Prérequis

Environnement

- **Python 3.7+**
- **Système d'exploitation** : Windows, macOS, Linux

Bibliothèques requises

```
pip install requests beautifulsoup4 openpyxl
```

Structure des fichiers

```
projet/
├── scraper_bulk.py          # Script principal
├── 20250117_cartographie_for.xlsx  # Fichier Excel source
├── README.md                # Cette documentation
└── logs/                    # Dossier des logs (créé automatiquement)
```

Installation et utilisation

1. Installation des dépendances

```
pip install requests beautifulsoup4 openpyxl
```

2. Configuration du fichier Excel source

Votre fichier Excel doit contenir :

- **Colonne O** : URLs des fiches Parcoursup
- **Ligne 1** : En-têtes
- **À partir de la ligne 2** : Données

3. Lancement du script

```
python scraper_bulk.py
```

4. Configuration personnalisée

Modifiez les variables dans le script :

```
input_file = '20250117_cartographie_for.xlsx'
url_column = 'O'
start_row = 2
```

Architecture du script

1. `extract_emails_from_text(text)`

- **But** : Extraire toutes les adresses mails présentes dans un texte brut
- **Entrée** : Chaîne de caractères
- **Sortie** : Liste des e-mails trouvés
- **Regex utilisée** : `r'\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,}\b'`

2. `extract_contacts_from_url(url)` - FONCTION PRINCIPALE

- **But** : Extraire les e-mails "général", "pédagogique" et "administratif" depuis une URL Parcoursup
- **Stratégie complète** :
  1. **ÉTAPE 1** : Recherche dans la section "Contacter et échanger avec l'établissement" sur Parcoursup
  2. **ÉTAPE 2** : Catégorisation intelligente selon le contexte (pédagogique/administratif)
  3. **ÉTAPE 3** : **SI AUCUN E-MAIL trouvé sur Parcoursup** → Appelle `get_official_website_from_parcoursup()` puis `scrape_official_website()`
- **Timeout** : 15 secondes par requête Parcoursup, 10 secondes pour le site officiel
- **Sortie** : Tuple `(contact_général, contact_pédagogique, contact_admin)`

3. `get_official_website_from_parcoursup(soup)` - FONCTION AUXILIAIRE

- **But** : Récupérer le site officiel de l'établissement à partir de la fiche Parcoursup
- **Entrée** : Objet BeautifulSoup de la page Parcoursup
- **Sortie** : URL du site officiel ou `None`
- **Filtres** : Exclut les domaines `parcoursup.fr` et `gouv.fr`
- **Usage** : Appelée par `extract_contacts_from_url()` uniquement si aucun e-mail trouvé sur Parcoursup

4. `scrape_official_website(url)` - FONCTION AUXILIAIRE

- **But** : Scraper le site officiel de l'établissement pour y trouver des e-mails
- **Utilisation** : Appelée par `extract_contacts_from_url()` en fallback uniquement
- **Limitation** : Maximum 3 e-mails pour éviter le spam
- **Timeout** : 10 secondes par requête
- **Gestion d'erreur** : Retourne une liste vide si l'URL est invalide ou inaccessible
- **Sortie** : Liste d'adresses mails (0 à 3)

5. `process_excel_bulk(input_file, url_column='O', start_row=2)`

- **But** : Traitement en masse du fichier Excel
- **Fonctionnalités** :
  - Sauvegarde automatique toutes les 100 entrées
  - Pauses programmées (1s entre requêtes, 10s tous les 50 appels)
  - Gestion complète des erreurs
  - Statistiques en temps réel

Format du fichier de sortie

Le fichier généré `contacts_extraits_YYYYMMDD_HHMMSS.xlsx` contient :

Colonne	Description	Exemple
Ligne	Numéro de ligne source	17
URL	URL Parcoursup traitée	https://dossierappel.parcoursup.fr/…
Contact Général	E-mail général trouvé	contact@etablissement.fr
Mail Pédagogique	E-mail pédagogique	pedago@etablissement.fr
Mail Administratif	E-mail administratif	admin@etablissement.fr
Statut	Résultat du traitement	Traité avec succès
Timestamp	Horodatage du traitement	2025-01-17 14:30:25

Statuts possibles

- **Traité avec succès** : E-mails trouvés
- **Traité - Aucun e-mail trouvé** : Page accessible mais pas d'e-mail
- **Erreur** : Problème technique (timeout, page inaccessible)
- **Skipped** : URL invalide ou vide

Performances et optimisations

Temps de traitement estimé

- **104 679 lignes** avec pauses de sécurité
- **Durée estimée** : ~29 heures
- **Sauvegarde automatique** : Toutes les 100 lignes

Optimisations possibles

```
# Réduire les pauses (à vos risques et périls)
time.sleep(0.5)  # Au lieu de 1 seconde

# Traitement par lots
if processed_count % 25 == 0:  # Au lieu de 50
    time.sleep(5)  # Au lieu de 10 secondes
```

Bonnes pratiques et sécurité

Respect des serveurs

- **Pauses automatiques** entre les requêtes
- **Limitation des timeouts** (10-15 secondes)
- **Monitoring des erreurs** pour détecter les blocages

Gestion des erreurs

- **Poursuite automatique** en cas d'erreur ponctuelle
- **Sauvegarde régulière** pour éviter les pertes
- **Logs détaillés** pour le débogage

Considérations légales

- **Usage responsable** : Respecter les CGU des sites
- **Finalité légitime** : Études, recherche, information
- **Pas de spam** : Limitation à 3 e-mails par site

Dépannage

Problèmes courants

1. Erreur "Permission denied" sur Excel

```
# Solution : Fermer le fichier Excel avant de lancer le script
# Le script créera automatiquement un nouveau fichier si nécessaire
```

2. Timeouts fréquents

```
# Augmenter les timeouts dans le script
response = requests.get(url, timeout=30)
```

3. Colonne URL introuvable

```
# Vérifier la colonne avec le script d'analyse
python check_excel.py
```

Logs et débogage

```
# Consulter les logs détaillés
tail -f scraper_bulk.log

# Vérifier les résultats intermédiaires
python check_results.py
```

Statistiques et reporting

Métriques collectées

- **Total de lignes traitées**
- **Taux de succès** (e-mails trouvés)
- **Nombre d'erreurs**
- **Lignes ignorées** (URLs invalides)
- **Temps de traitement**

Exemple de rapport final

```
============================================================
TRAITEMENT TERMINÉ
============================================================
Total de lignes traitées: 10000
Succès (avec e-mails): 7850
Erreurs: 1200
Lignes ignorées: 950
Taux de succès: 78.5%
Fichier de sortie: contacts_extraits_20250117_143025.xlsx
============================================================
```

Modularité et évolution

Adaptation aux changements

- **Structure modulaire** : Chaque fonction est indépendante
- **Points de modification** :
  - `extract_contacts_from_url()` : Si Parcoursup change sa structure
  - `get_official_website_from_parcoursup()` : Pour de nouveaux filtres
  - `extract_emails_from_text()` : Pour des formats d'e-mails spécifiques

Extensions possibles

- **Support multi-sites** (autres plateformes que Parcoursup)
- **Interface graphique** pour les utilisateurs non-techniques
- **Mode incrémental** (reprendre un traitement interrompu)
- **Reporting avancé** (graphiques, statistiques détaillées)

Support et contribution

En cas de problème

1. **Consulter les logs** : `scraper_bulk.log`
2. **Vérifier les prérequis** : Python, bibliothèques
3. **Tester sur un échantillon** : Modifier `start_row` et limiter le nombre de lignes

Améliorations suggérées

- **Parallélisation** : Traitement multi-thread (avec précaution)
- **Filtres avancés** : Exclusion de certains types d'établissements
- **Base de données** : Stockage dans PostgreSQL/MySQL

Résumé

Ce script propose une méthode **automatisée**, **rapide** et **documentée** pour consolider les mails de contact d'établissements ou de formations à partir du portail Parcoursup et de leur site officiel.