

DOCUMENTATION TECHNIQUE

EXPORT CONTACTS HEKTOR → APIMO

Migration automatisée de 14 445 contacts via scraping
Python et API REST Apimo

Client	Réseau Bonaparte
Auteur	Baptiste Fort
Date	12 janvier 2026
Version	1.0

14 445

CONTACTS

12 225

ACQUÉREURS

2 220

PROPRIÉTAIRES

4j

DURÉE

SOMMAIRE

01. Contexte et Objectifs

02. Architecture Technique

03. Stack Technologique

04. Implémentation

05. Intégration API Apimo

06. Résultats & Métriques

01 CONTEXTE & OBJECTIFS

Problématique

Le Réseau Bonaparte utilise Hektor pour la gestion de ses contacts. La migration vers Apimo nécessite le transfert de 14 445 contacts (acquéreurs et propriétaires). Hektor ne propose pas d'API d'export, rendant l'extraction manuelle impossible à grande échelle.

Objectifs de la mission

1 Extraction automatisée

Développer un script Python capable d'extraire tous les contacts depuis l'interface web Hektor via scraping.

2 Migration vers Apimo

Injecter les contacts nettoyés et mappés via l'API REST Apimo v3 avec gestion des erreurs.

Contraintes techniques identifiées

- Pas d'API disponible côté Hektor
- Authentification session requise pour le scraping
- Rate limiting API Apimo (150 req/min)
- Formats de données différents entre les deux plateformes
- Gestion des doublons potentiels

Solution retenue

Développement d'un pipeline Python en 4 étapes : extraction Selenium → nettoyage Pandas → mapping JSON → injection API REST avec retry automatique.

02 ARCHITECTURE TECHNIQUE

Vue d'ensemble

Le système est composé de 4 modules indépendants qui s'exécutent séquentiellement pour assurer la traçabilité et permettre la reprise en cas d'erreur.



Composants du système

Extracteur Module Selenium qui navigue dans Hektor, parcourt les pages et extrait les données HTML.	Nettoyeur Module Pandas qui valide les emails, normalise les téléphones et détecte les doublons.	Injecteur Module requests qui envoie les contacts vers l'API Apimo avec gestion du rate limiting.
---	--	---

03 STACK TECHNOLOGIQUE

COMPOSANT	TECHNOLOGIE	VERSION	RÔLE
Runtime	Python	3.12	Langage principal
Scraping	Selenium	4.15	Automatisation navigateur
Driver	ChromeDriver	120+	Interface Chrome
Data	Pandas	2.1	Manipulation données
HTTP	Requests	2.31	Appels API

Dépendances

```
selenium=4.15.2
pandas=2.1.3
requests=2.31.0
python-dotenv=1.0.0
email-validator=2.1.0
tqdm=4.66.1
```

04 IMPLÉMENTATION

Structure du projet

```
hektor-to-apimo/
├── src/
│   ├── extractor.py # Module extraction Hektor
│   ├── cleaner.py # Nettoyage et validation
│   ├── mapper.py # Mapping champs Hektor → Apimo
│   └── injector.py # Injection API Apimo
├── data/
│   ├── raw/ # Données brutes extraites
│   ├── clean/ # Données nettoyées
│   └── logs/ # Logs d'exécution
├── config.py # Configuration
├── main.py # Point d'entrée
└── requirements.txt
```

Code principal - Extracteur

```
class HektorExtractor:
    def __init__(self, driver):
        self.driver = driver
        self.contacts = []

    def extract_all(self,
                    contact_type: str):
        """Extrait tous les contacts d'un type"""
        self.navigate_to_contacts(contact_type)

        while True:
            contacts = self.extract_page()
            self.contacts.extend(contacts)

            if not self.has_next_page():
                break
            self.next_page()

        return self.contacts

    def extract_page(self):
        """Parse la page courante"""
        rows = self.driver.find_elements(By.CSS_SELECTOR, ".contact-row")
        return [self.parse_row(r) for
                r in rows]
```

Points clés de l'implémentation

- Gestion automatique des sessions expirées
- Sauvegarde incrémentale toutes les 500 entrées
- Logs détaillés pour chaque opération
- Mode reprise en cas d'interruption

05 INTÉGRATION API APIMO

Endpoints utilisés

MÉTHODE	ENDPOINT	DESCRIPTION
GET	/contacts	Vérification existence contact
POST	/contacts	Création nouveau contact
PATCH	/contacts/{id}	Mise à jour contact existant

Authentification

```
headers = {
    "Authorization": f"Basic
        {base64(provider:token)}",
    "Content-Type": "application/json"
}
```

Mapping des champs

```
FIELD_MAPPING = {
    "nom": "lastname",
    "prenom": "firstname",
    "email": "email",
    "telephone": "phone",
    "type": "category", # owner/buyer
    "budget": "budget_max"
}
```

06 RÉSULTATS & MÉTRIQUES

14 445

TOTAL TRAITÉS

12 225

ACQUÉREURS

2 220

PROPRIÉTAIRES

~1h

TEMPS EXEC.

Logs d'exécution (extrait)

```
[2026-01-16 09:00:01] INFO - Demarrage extraction Hektor
[2026-01-16 09:00:05] INFO - Connexion reussie
[2026-01-16 09:15:32] INFO - Extraction terminee: 14445 contacts
[2026-01-16 09:15:33] INFO - Debut nettoyage...
[2026-01-16 09:16:01] INFO - Doublons detectes: 127
[2026-01-16 09:16:02] INFO - Emails invalides: 43
[2026-01-16 09:16:02] INFO - Contacts valides: 14275
[2026-01-16 09:16:03] INFO - Debut injection API Apimo
[2026-01-16 10:12:45] INFO - Injection terminee: 14275/14275
[2026-01-16 10:12:45] INFO - MIGRATION COMPLETE
```

✓ Mission terminée avec succès

Tous les contacts ont été migrés vers Apimo. Les propriétaires et acquéreurs sont correctement catégorisés. Les doublons ont été fusionnés.

Gestion des erreurs

Codes de retour

- 200 : Succès
- 201 : Contact créé
- 400 : Données invalides
- 429 : Rate limit atteint
- 500 : Erreur serveur

Stratégie de retry

Retry exponentiel : 1s → 2s → 4s. Maximum 3 tentatives. Les erreurs 400 ne sont pas retentées.

Recommandations

- Vérifier la qualité des données migrées sur un échantillon
- Planifier une synchronisation régulière si Hektor reste utilisé
- Former les équipes à l'utilisation d'Apimo

Documentation rédigée par **Baptiste Fort**
SIRET 98939247400013 – contact@baptistefort.fr

