



# Job App Assistant

Automatisation & Optimisation de Candidature par IA

Maxim Quénel | Pierre Louis Brun | Mame Alé Seye | Denis Bereziuc

# Le Contexte

## La barrière des ATS

Les candidats font face à un "trou noir". Les systèmes de suivi des candidatures (ATS) filtrent automatiquement les CVs ne correspondant pas exactement aux mots-clés, rejetant souvent des profils qualifiés.

## Le Défi

Adapter manuellement son CV pour chaque offre est :

- 🕒 Chronophage
- 🔄 Répétitif
- 🔧 Sujet à l'erreur humaine





# Architecture Technique

## Structure Monolithique Modulaire

Une application web locale conçue pour la confidentialité ("Local-First") et la performance.

### Backend

Flask (Python). Gestion des threads pour les tâches IA longues.

### Stockag

Système de fichiers (CSV/TXT).  
Pas de SGBD lourd.

### Frontend

HTML/JS Templates. Polling temps réel des logs.

### Inférence

Hugging Face Transformers en local.

```
job_opp_assistant/  
├── app.py                # Point d'entrée principal (Serveur Web / Orchestrateur)  
├── requirements.txt      # Dépendances Python  
├── msedgedriver.exe      # Driver pour Selenium (Scraping Edge)  
  
├── data/                # Stockage des données (Entrées/Sorties)  
│   ├── jobs_raw.csv     # Données brutes scrapées  
│   ├── jobs_rewritten.csv # Offres réécrites par IA  
│   ├── cv_converted.txt  # CV converti en texte brut  
│   ├── final_matches.csv # Résultats du matching simple  
│   └── final_matches_cross.csv # Résultats du matching avancé (Cross-Encoder)  
  
├── services/            # Logique métier (Backend)  
│   ├── scraper.py       # Script de scraping (Selenium)  
│   ├── cv_converter.py  # Extraction de texte depuis PDF  
│   ├── cv_rewriter.py   # Optimisation du CV par LLM  
│   ├── job_rewriter.py  # Nettoyage des offres par LLM  
│   ├── matcher.py       # Algorithme de matching vectoriel  
│   ├── cross_encoder_matcher.py # Algorithme de matching précis (Re-ranking)  
│   └── raw_job_parser.py # Parsing initial des offres  
  
├── static/              # Assets Frontend  
│   ├── css/             # Styles (Design des résultats et previews)  
│   └── js/              # Scripts côté client (main.js)  
  
├── templates/           # Vues HTML  
│   └── index.html       # Page principale de l'interface  
  
└── utils/               # Utilitaires transverses  
    └── logger.py        # Gestion des logs
```

# Modèles & Intelligence Artificielle

## Génération & Réécriture Qwen2.5-1.5B-Instruct



**Type :** SLM (Small Language Model).

**Usage :** Nettoyage des offres, anonymisation et structuration du CV.

**Pourquoi ?** Léger (1.5B paramètres), tourne sur GPU grand public, excellent suivi d'instructions.

## Matching (Niveau 1) BAAI/bge-m3



**Type :** Bi-Encoder (Sentence Transformer).

**Usage :** Filtrage rapide.

**Méthode :** Calcul de similarité cosinus entre les vecteurs (embeddings) du CV et des offres.

## Reranking (Niveau 2) BAAI/bge-reranker-v2-m3



**Type :** Cross-Encoder.

**Usage :** Scoring de précision (0-100%).

**Méthode :** Analyse conjointe (CV + Offre) pour capturer les nuances sémantiques fines.



# Pipeline de Données Détaillé (Flux)



# Stack Technique Détaillée



## Core & Data



**Python 3.12.8** : Langage principal



**Pandas** : Manipulation des DataFrames (CSV)



**Selenium** : Automatisation Web (Edge Driver)



**pdfplumber** : Extraction texte depuis PDF



## IA & NLP



**Transformers** : Hugging Face Hub



**Torch** : Backend Deep Learning



**Scikit-learn** : Utilitaires mathématiques



**Sentence-Transformers** : Embeddings



# Interface Utilisateur

## Simplicité & Contrôle

L'interface propose un parcours en 6 étapes distinctes.



**Feedback Temps Réel** : Polling API pour afficher les logs d'exécution.



**Prévisualisation** : Vérification des fichiers intermédiaires (JSON/TXT) avant de passer à l'étape suivante.



**Graphiques** : Visualisation immédiate des scores de matching.

### Job AI Assistant

Automatisation intelligente de votre recherche d'emploi

#### 01 Recherche d'Offres

Scrapes les dernières offres depuis HelloWork.

Par Mot-clé

Par Lieu

Par Type de Stut

Modèle

Data Analyst

Nombre d'offres

5

Lancer Analyse

En attente

02 Réécriture Offres (IA)

Analysez et résumez les offres avec Owen 1.5B.

Lancer Réécriture

En attente

03 Conversion CV

Convertissez votre CV PDF en texte brut nettoyé.

Choisir un PDF

Aucun fichier

Convertir

En attente

04 Synthèse CV (IA)

Génère une synthèse structurée de votre profil.

Générer Synthèse

En attente

05 Calcul Matching

Comparez votre profil aux offres (Semantic Search).

Calculer Scores

En attente

06 Cross Matching (Ultra-Précis)

Celui-ci est plus précis avec Cross Encoder (plus lent mais meilleur).

Lancer Cross Match

En attente

# Test et Comparaison



**Matching avec CV  
Correspondant**



**Matching avec CV  
Non Correspondant**

## 05 Calcul Matching

Comparez votre profil aux offres  
(Semantic Search).

Calculer Scores

Terminé

**Stage Intelligence  
Artificielle H/F** **75.2%**

Proxiad

IA Résumé: - Type de profil  
recherché: Etudiant(s) en Sciences  
des Données (ou équivalent) | -...

[Voir l'offre](#)

06

## Cross Matching (Ultra-Précis)

Calcul précis avec Cross Encoder (plus  
lent mais meilleur).

Lancer Cross Match

Terminé

**Stage Intelligence  
Artificielle H/F** **67.5%**

Proxiad

IA Résumé: - Type de profil  
recherché: Etudiant(s) en Sciences  
des Données (ou équivalent) | -...

[Voir l'offre](#)

05

## Calcul Matching

Comparez votre profil aux offres  
(Semantic Search).

Calculer Scores

Terminé

**Stage Intelligence  
Artificielle H/F** **52.4%**

Proxiad

IA Résumé: - Type de profil  
recherché: Etudiant(s) en Sciences  
des Données (ou équivalent) | -...

[Voir l'offre](#)

06

## Cross Matching (Ultra-Précis)

Calcul précis avec Cross Encoder (plus  
lent mais meilleur).

Lancer Cross Match

Terminé

**Responsable  
Intelligence  
Artificielle H/F** **50.0%**

UNIKALO

IA Résumé: - Type de profil  
recherché: Responsable Intelligence  
Artificielle H/F | - compétences clés...



# Améliorations & Roadmap



## Robustesse

Remplacer threading par une file de tâches (Celery/Redis) pour gérer les échecs et la montée en charge.



## Données

Migration des fichiers CSV vers **SQLite** pour permettre des requêtes complexes et un meilleur filtrage.



## Performance

Quantification des modèles (4-bit via bitsandbytes) pour réduire la consommation VRAM.

# Questions ?

Merci de votre attention.



[github.com/Maxim-Quenel/job\\_app\\_assistant\\_matching\\_OV](https://github.com/Maxim-Quenel/job_app_assistant_matching_OV)



Démo sur youtube :

[https://youtu.be/R0ofgB8\\_0bw?si=ktxXZxt8kp71zwy1](https://youtu.be/R0ofgB8_0bw?si=ktxXZxt8kp71zwy1)