

# CAHIER DES CHARGES : APPLICATION D'ÉVALUATION DE CV POUR CONSULTANTS PYTHON/DJANGO (MYSQL)

-----/// -----

## I. Contexte et Objectifs

### 1. Contexte

L'entreprise souhaite automatiser l'évaluation des CV des consultants spécialisés pour optimiser le processus de recrutement.

### 2. Objectifs\*\*

- Automatiser l'analyse des CV basée sur des critères techniques (Python, Django, etc.).
- Noter et classer les candidats selon leurs compétences.
- Faciliter la prise de décision\*\* pour les recruteurs.
- Centraliser les données dans une base **MySQL** pour une meilleure gestion.

## II. Fonctionnalités Principales

### 1. Gestion des Utilisateurs

- Inscription/Connexion (Consultants & Recruteurs).
- Rôles distincts
- Consultant : Peut uploader son CV.
- Recruteur/Admin : Peut évaluer et filtrer les CV.

## **2.Upload et Analyse Automatique des CV**

- Formats supportés : PDF, DOCX.
- Extraction de texte (NLP, OCR si nécessaire).
- Analyse des compétences :
- Python (Pandas, NumPy, Flask, etc.).
- Django (ORM, REST, Modèles, etc.).
- Autres frameworks (FastAPI, Celery, etc.).
- Calcul d'un score global (0-100%).

## **3 Tableau de Bord d'Évaluation**

- Filtrage par compétences, score, expérience.
- Visualisation des profils (graphiques, classement).
- Commentaires et notes manuelles\*\* (optionnelles).

## **4 Base de Données MySQL**

- Stockage structuré des CV, évaluations, utilisateurs.
- Requêtes optimisées pour un chargement rapide.
- Backup automatisé.

## **III. Spécifications Techniques**

### **1 Backend (Django)**

- Framework : Django 4.x
- Base de données : MySQL 8.x

- Analyse de CV :
- Bibliothèques NLP: `nltk`, `spaCy`, `scikit-learn`.
- Extraction PDF/DOCX\*\* : `PyPDF2`, `python-docx`.
- API REST (optionnel) : Django REST Framework.

## 2 Frontend


- Templates HTML/CSS : Bootstrap 5.
- Visualisation : Chart.js (graphiques).
- Expérience utilisateur (UX) :
- Upload simplifié.
- Affichage clair des scores.

## 3 Sécurité



- Authentification : Django Auth + JWT (optionnel).
- Protection des fichiers:
- Stockage sécurisé (`/media/cvs/`).
- Vérification anti-malware.
- MySQL
- Chiffrement des données sensibles.
- Gestion des permissions.

## 4. Architecture Logicielle

 app\_cv\_evaluator/

```
└─  core/      # Modèles Django (MySQL)
|   └─ models.py    # Consultant, CV, Evaluation
|   └─ utils.py     # Analyse NLP, extraction
```

```

| └─ views.py      # Logique métier
|   └─  templates/      # Frontend
|     └─ dashboard.html  # Tableau de bord
|   └─ upload.html      # Upload de CV
|     └─  media/cvs/      # Stockage des CV
|       └─ manage.py
|     └─ settings.py      # Config MySQL
|   ...

```

## 5. Contraintes et Hypothèses

### ▪ 1 Contraintes

- Performances : L'analyse d'un CV doit prendre < 10 secondes.
- Compatibilité : Support des CV en français/anglais.
- Sécurité\*\* : Respect du RGPD (données personnelles).

### ▪ 2 Hypothèses

- Les CV contiennent des sections standard (Expérience, Compétences, etc.).
- Pas de reconnaissance avancée de formats exotiques (images scannées).

## 7. Planning Prévisionnel

Phase	Durée
Analyse & Conception	2 sem.
Développement Django	3 sem.
Tests & Optimisation	1 sem.
Déploiement	1 sem.