INETUM TUNISIE

Challenge Night 2026

SmartTender AI

Documentation Technique

Automated Tendering Intelligence Platform

1. Executive Summary

SmartTender AI est une plateforme d'intelligence artificielle conçue pour automatiser le cycle complet de réponse aux appels d'offres. Développée dans le cadre du Challenge Night Inetum, la solution adresse trois problèmes opérationnels critiques : la détection manuelle des opportunités, la sélection fastidieuse des experts, et la rédaction répétitive des livrables.

Périmètre du MVP

Le prototype fonctionnel couvre deux modules pleinement opérationnels (CV Matching & Scoring Engine, Document Generation) et un module de monitoring (Dashboard). Le module de Tender Detection est implémenté en mode simulé (mock local), avec une architecture prévue pour l'intégration de sources réelles.

Module	Statut	Technologie principale
CV Matching & Scoring	Opérationnel	Chroma + Embeddings + SQLite
Document Generation	Opérationnel	python-docx + LLM
Smart Dashboard	Opérationnel	Interface locale
Tender Detection	Mock / Simulé	RSS / API (prévu)

2. Architecture du système

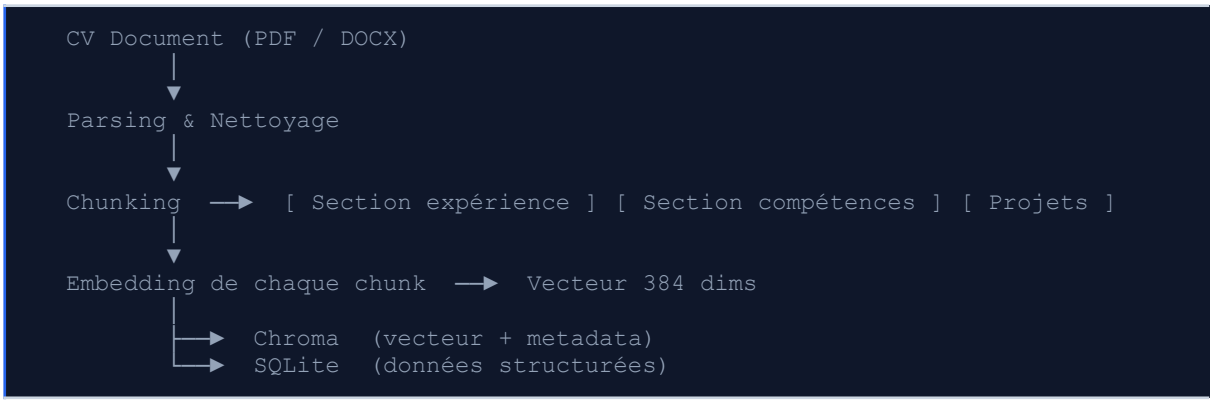
L'architecture suit un modèle modulaire local, organisé en quatre couches fonctionnelles indépendantes. Cette séparation garantit la maintenabilité, la testabilité et l'évolutivité de chaque composant.

Couche	Composant	Rôle
Input Layer	Tender PDF + CV documents	Ingestion des données brutes
Processing Layer	LLM Parser	Extraction structurée des exigences
Processing Layer	Embedding Engine	Vectorisation sémantique des textes
Processing Layer	Hybrid Matching Engine	Scoring multicritère des profils
Storage Layer	SQLite	Données structurées CV (métadonnées)
Storage Layer	Chroma Vector Index	Index sémantique persistant
Output Layer	Shortlist & Ranking	Résultats classés avec justifications
Output Layer	DOCX Report Generator	Génération automatique de livrables
Output Layer	Dashboard	Monitoring et traçabilité

3. Flux de données

Le pipeline de traitement suit un flux séquentiel et déterministe, depuis l'ingestion du document tender jusqu'à la production du rapport final.

3.1 Flux d'indexation des CV (phase offline)



4. Techniques IA utilisées

4.1 Embeddings sémantiques

Les embeddings constituent le mécanisme fondamental du moteur de matching. Chaque texte — CV ou tender — est transformé en un vecteur dense de 384 dimensions via un modèle de sentence embedding. Cette représentation capture la sémantique du contenu, permettant de rapprocher des formulations différentes décrivant la même réalité.

```
Input  : "Senior Data Engineer with AWS and Spark experience"
Output : [ 0.012, -0.441, 0.823, 0.091, -0.334, ... ]  (384 valeurs)
```

```
Input  : "Ingénieur données confirmé, maîtrise cloud et traitement distribué"
Output : [ 0.019, -0.428, 0.811, 0.104, -0.318, ... ]  (384 valeurs)

→  Cosine similarity ≈ 0.94  (très proches sémantiquement)
```

4.2 Stratégie de chunking

Un CV complet ne peut être représenté par un unique vecteur global sans perte d'information significative. Le système segmente chaque document en chunks avant vectorisation, selon deux stratégies complémentaires.

Stratégie	Description	Avantage
Segmentation sémantique	Découpage par sections logiques (expérience, compétences, projets, formation)	Préserve la cohérence contextuelle
Segmentation par taille	Blocs fixes de 256–512 tokens avec overlap de 50 tokens	Garantit une couverture uniforme

4.3 Chroma – Base vectorielle persistante

Chroma est utilisé comme index sémantique persistant. Il stocke, pour chaque chunk de CV, le vecteur d'embedding associé et ses métadonnées (identifiant, nom, expérience, compétences déclarées). À l'interrogation, le vecteur du tender est soumis à une recherche par plus proches voisins (ANN), retournant les profils les plus pertinents sans recalcul des embeddings.

Propriété	Valeur
Méthode de similarité	Cosine similarity
Persistance	Index réutilisé entre sessions (pas de recalcul)
Scalabilité	Efficace jusqu'à des centaines de milliers de documents
Métadonnées	Filtrables (expérience, langue, disponibilité)

4.4 Score hybride

Le système de scoring ne délègue pas la décision aux seuls embeddings. Il combine trois couches d'analyse complémentaires pour produire un score final explicable et auditable.

```
Score_final = 0.50 × Score_sémantique
            + 0.30 × Score_lexical
            + 0.10 × Score_expérience
            + 0.10 × Score_langues

Score_sémantique  =  cosine_similarity(tender_vector, cv_vector) × 100
Score_lexical     =  |skills_matched| / |skills_required| × 100
Score_expérience  =  min(years_cv / years_required, 1.0) × 100
Score_langues     =  |languages_matched| / |languages_required| × 100
```

Couche	Méthode	Force	Limite
Sémantique (50%)	Cosine similarity (Chroma)	Comprend le contexte, tolère les synonymes	Peut être trop permissif

Couche	Méthode	Force	Limite
Lexicale (30%)	Exact skill matching	Précis sur les compétences critiques	Sensible aux variations de formulation
Contraintes (20%)	Règles explicites	Garanti, transparent, auditable	Rigide, pas d'interpolation

4.5 Rôle du LLM

Le LLM intervient de manière ciblée et délibérément limitée à trois tâches : extraction structurée des exigences depuis le tender (parsing JSON), normalisation des CV si nécessaire, et reformulation dans la génération du rapport final. Il n'intervient à aucun moment dans le calcul du score.

Choix architectural — Pourquoi ne pas scorer via le LLM ?

Confier le scoring à un LLM produirait des décisions non reproductibles, difficilement auditables, et potentiellement biaisées selon le prompt. L'approche hybride retenue (embedding + règles) garantit stabilité, explicabilité et traçabilité — qualités essentielles dans un contexte de décision RH à enjeux.

5. Stack technique

Composant	Technologie	Version	Rôle
Embedding model	Sentence-Transformers	Latest	Génération des vecteurs sémantiques
Vector store	ChromaDB	0.4+	Index persistant + recherche ANN
Structured DB	SQLite	3.x	Stockage métadonnées CV
LLM	Gemini (primary) / OpenRouter (optional)	Configurable	Extraction + reformulation
Document gen.	python-docx	1.x	Génération rapports DOCX
Backend	Python / FastAPI	3.11+	API et orchestration
Secrets	.env (dotenv)	—	Gestion sécurisée des clés API

6. Limitations actuelles

Le MVP présente des limitations connues, documentées ici de manière transparente :

Limitation	Impact	Criticité
Extraction de keywords bruitée	Mots administratifs inclus dans les requirements	Moyen
Pondération de scoring statique	Non adaptée aux spécificités de chaque tender	Moyen
Tender detection non automatisée	Veille manuelle nécessaire (mock local)	Élevé
Absence de feedback loop	Pas d'apprentissage sur l'historique des succès	Faible (MVP)

Limitation	Impact	Criticité
Interface en cours de finalisation	UX non optimisée pour usage production	Faible (MVP)
Pas de gestion multi-tenant	Architecture mono-organisation	Faible (MVP)

7. Feuille de route — Améliorations futures

Court terme (0–3 mois)

- Amélioration du pipeline NLP pour le filtrage des keywords administratifs
- Pondération dynamique du scoring configurable par tender
- Fine-tuning du chunking strategy selon les formats de CV courants

Moyen terme (3–9 mois)

- Intégration d'un module de Tender Detection via flux RSS et APIs publiques (ANPR, TUNEPS)
- Adoption d'une approche RAG pour une analyse contextuelle plus fine des exigences complexes
- Feedback loop : ajustement dynamique des poids selon l'historique des appels d'offres gagnés
- Authentification multi-utilisateur et gestion des rôles

Long terme (9+ mois)

- Déploiement SaaS cloud-ready avec architecture microservices
- Support multilingue (FR / EN / AR) pour les marchés MENA
- API exposée pour intégration aux systèmes d'information existants (ERP, SIRH)
- Module de Tender Detection proactive avec alertes en temps réel

8. Conclusion

Le MVP SmartTender AI démontre la faisabilité technique d'une automatisation intelligente du processus de sélection d'experts pour appels d'offres. L'architecture hybride retenue — combinant embeddings sémantiques, scoring lexical et règles métier — illustre une maîtrise concrète des systèmes d'IA appliqués à un cas d'usage enterprise réel.

Les choix architecturaux (LLM limité à l'extraction, scoring déterministe, index persistant) reflètent une compréhension des enjeux de fiabilité, d'explicabilité et de scalabilité propres aux environnements professionnels. Le système est modulaire, évolutif, et constitue une base solide vers une solution enterprise à part entière.

Compétences démontrées

Architecture vectorielle (RAG/embeddings) · Hybrid AI (neural + symbolic) · LLM appliqué à l'extraction structurée · Design modulaire et scalable · Pipeline de données end-to-end · Génération automatisée de documents