Contribution au développement du chatbot IA

Date 08-08-2025

1. Contexte initial

Dans la version précédente du chatbot :

- La détection d'entité (library pour documents, product pour produits) dépendait beaucoup de mots-clés explicites (ex. « document », « produit »).
- Les champs comme type étaient ambigus, car ils existent dans les deux entités :
  - Document → doc_type
  - Produit → form
- Si l'utilisateur n'indiquait pas clairement l'entité, la requête échouait souvent ou renvoyait une réponse incomplète.

Exemple problématique avant :

Utilisateur : donner le type de "Guide ANSM"
→ Bot : Je n'ai pas bien compris la demande.

2. Améliorations apportées

2.1. Fuzzy matching des champs

- Ajout de la fonction _guess_fields basée sur **RapidFuzz** (fuzz.token_set_ratio).
- Permet de détecter un champ même si :
  - Il y a une faute de frappe.
  - Il est dans une formulation différente (« forme » vs form, « statut » vs status).

- Fonctionne pour **DOC_FIELD_MAP** et **PROD_FIELD_MAP.**

2.2. Extraction avancée des valeurs

- Fonction _extract_quoted : capture les valeurs entre guillemets (", ', « ») → utile pour extraire le nom/titre cible.
- Patrons regex (DOC_FILTER_PATTERNS et PROD_FILTER_PATTERNS) pour repérer des expressions comme :
  - de type X, type: X
  - statut Commercialisé
  - principe actif: Y ou PA Y
- Les valeurs sont ensuite mappées vers les bons champs via **DOC_TRIGGER_MAP** ou **PROD_TRIGGER_MAP.**

2.3. Détection automatique de l'entité

Nouvelle logique dans parse_request :

1. On tente de détecter les champs dans **DOC_FIELD_MAP** et **PROD_FIELD_MAP.**
2. Si les deux matchent (ex. type existe pour les deux entités) :
   a. On extrait le nom ou titre (name_or_title_hint) depuis la phrase.
   b. On utilise _best_entity_for_name pour comparer la similarité avec les noms de produits et les titres de documents (basé sur find_best_product et find_best_document).
   c. L'entité avec le meilleur score est choisie.
3. Si aucun champ clair → fallback vers l'intent détecté (question_type).

2.4. Application des filtres

- _extract_filters applique les regex selon l'entité choisie.
- Les valeurs sont normalisées avec normalize_filter_value (gestion des accents, synonymes, codes internes).

- Filtrage souple (iexact ou icontains) dans list_documents et list_products.

2.5. Comportement attendu désormais

- **Cas ambigu** :

```
Utilisateur : donner le type de "Guide ANSM"
→ Détection : entity = library, field = doc_type
→ Réponse : Type du document = guideline
```

- **Cas produit** :

```
Utilisateur : donner le type de Para
→ Détection : entity = product, field = form
→ Réponse : Type du produit = comprimé
```

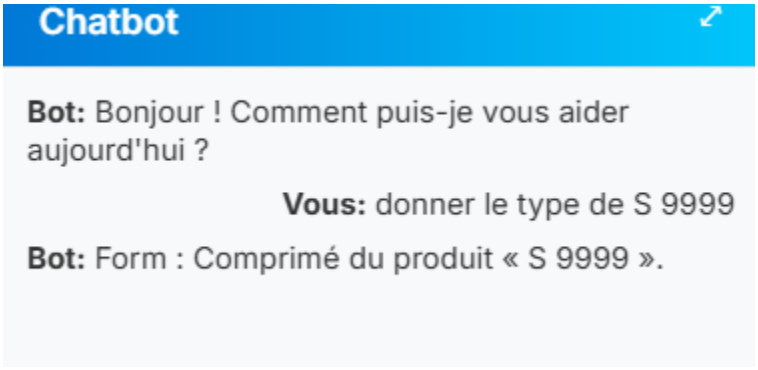
- **Cas filtre sans entité précisée** :

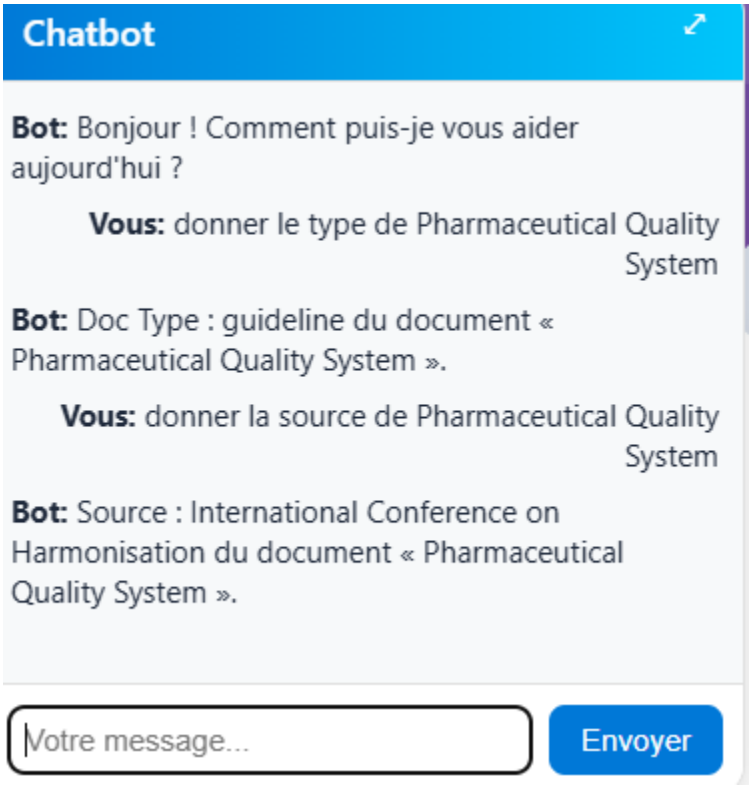
```
Utilisateur : donner tous les produits avec statut Commercialisé
→ Détection : entity = product, filtre = status=Commercialisé
→ Réponse : tableau des produits concernés
```

3. Points techniques clés

- **Robustesse** : plus de dépendance aux mots « document » ou « produit ».
- **Souplesse** : même une phrase incomplète ou mal formulée peut être comprise.
- **Réutilisabilité** : patrons regex, mapping d’alias, et fonctions de normalisation facilement extensibles.
- **Lisibilité** : séparation claire entre extraction (_extract_quoted, _extract_filters), détection (_guess_fields, _best_entity_for_name), et réponse (list_*, detail_*).

RESULTAT :


Chatbot
Bot: Bonjour ! Comment puis-je vous aider aujourd'hui ?
Vous: donner le type de S 9999
Bot: Form : Comprimé du produit « S 9999 ».


Chatbot
Bot: Bonjour ! Comment puis-je vous aider aujourd'hui ?
Vous: donner le type de Pharmaceutical Quality System
Bot: Doc Type : guideline du document « Pharmaceutical Quality System ».
Vous: donner la source de Pharmaceutical Quality System
Bot: Source : International Conference on Harmonisation du document « Pharmaceutical Quality System ».
Votre message...
Envoyer