Assignment: Smart Librarian – AI cu RAG + Tool Completion

Scop:

Construiește un chatbot AI care recomandă cărți în funcție de interesele utilizatorului, folosind OpenAI GPT + RAG (ChromaDB), iar apoi completează recomandarea cu un rezumat detaliat obținut printr-un tool separat.

Cerințe:

1. Creează o bază de date de rezumate (book_summaries). Include minim 10 cărți.

Format recomandat:

Title: 1984
A dystopian story about a totalitarian society controlled by surveillance, propaganda, and thought of police. Winston Smith, the protagonist, secretly rebels against the system in search of truth and freedom.

Title: The Hobbit
Rezumat scurt (3–5 rânduri). Include temele principale (ex: prietenie, aventură).

2. Încarcă fișierul în ChromaDB sau alta DB la alegere (**NU** OpenAi vectore store)
Folosește embeddings de la OpenAI (text-embedding-3-small sau similar). Creează un retriever care permite căutarea semantică după temă sau context.

3. Construiește un chatbot AI (CLI sau Streamlit) si Integrează OpenAI GPT

Chatbotul trebuie să:

- Primească întrebări de tip:
  „Vreau o carte despre prietenie și magie”
  „Ce recomanzi pentru cineva care iubește povești de război?”
- Găsească o potrivire în vector store (RAG)
- Răspundă conversațional cu o recomandare de carte

4. Adaugă un tool: **get_summary_by_title**(title: str)
Creează o funcție Python care returnează rezumatul complet pentru un titlu exact.

Se poate folosi un dicționar, JSON sau altă sursă locală.

Modelul de dicționar:

```
book_summaries_dict = {
   "The Hobbit": (
      "Bilbo Baggins, un hobbit confortabil și fără aventuri, este luat prin surprindere "
      "atunci când este invitat într-o misiune de a recupera comoara piticilor păzită de dragonul Smaug. "
      "Pe parcursul călătoriei, el descoperă curajul și resursele interioare pe care nu știa că le are. "
      "Povestea este plină de creaturi fantastice, prietenii neașteptate și momente tensionate."
   ),
   "1984": (
      "Romanul lui George Orwell descrie o societate distopică aflată sub controlul total al statului. "
      "Oamenii sunt supravegheați constant de „Big Brother”, iar gândirea liberă este considerată crimă. "
      "Winston Smith, personajul principal, încearcă să reziste acestui regim opresiv. "
      "Este o poveste despre libertate, adevăr și manipulare ideologică."
   )}


def get_summary_by_title(title: str) -> str:
   # caută titlul și returnează rezumatul complet
```

Înregistrează funcția ca tool (function calling) în OpenAI Chat API.

După ce LLM face o recomandare, apelează automat tool-ul cu titlul respectiv și afișează rezultatul complet după recomandare.

5. (Opțional) Adaugă un filtru de limbaj nepotrivit
Dacă utilizatorul trimite mesaje cu cuvinte ofensatoare, chatbotul trebuie să răspundă politicos și să nu trimită promptul la LLM.

6. (Text to Speech – Optional) Oferă utilizatorilor posibilitatea de a asculta recomandările și rezumatele cărților. Adaugă o opțiune în interfață (CLI sau Streamlit) prin care utilizatorul poate apasa.

După ce chatbotul oferă o recomandare + rezumat detaliat, opțional, convertește textul în audio și salvează fișierul sau îl redă direct.

7. (Speech to text – optional) Permite utilizatorului să interacționeze cu chatbotul vocal, prin comenzi vorbite. Adaugă un buton / mod „voice mode" în interfață. Dacă este activ, aplicația: Ascultă microfonul utilizatorului, transcrie în text, trimite întrebarea transcrisă către chatbot.

8. (Image Generation – optional) Generează o imagine reprezentativă pentru cartea recomandată, poate fi o copertă sugestivă, o scenă sau o tema din carte, afișează imaginea generate.

9. (Backend and Frontend UI – optional) Streamlit poate sa fie destul de greoi ca si interfata, folositi un framework de frontend pentru a construi interfata grafica care sa fie accesibila din browser (folosind localhost) si care sa fie integrata cu API-urile expuse de backend-ul scris in Python, (Ex: React, Angular, Vue, etc), si va puteti folosi de tool-uri precum ChatGPT sau Github Copilot.

Ce trebuie livrat:
Fișier book_summaries cu 10+ cărți

Codul sursa Python si ce alte limbaje si dependinte s-au folosit, avand incluse:

Inițializare vector store

Tool get_summary_by_title()

Chat complet cu interacțiune LLM + tool

UI simplu: CLI sau Streamlit (opțional)

README.md cu pașii de build si de rulare

Exemple de întrebări pentru testare:
„Vreau o carte despre libertate și control social."

„Ce-mi recomanzi dacă iubesc poveștile fantastice?"

„Ce este 1984?"

Notes

1. Punctele de mai sus sunt doar orientative, scopul este sa vedem un proiect care foloseste RAG cu un vector store, care nu a fost prezentat in trainings si ca se face un tools calling pentru extra details.
2. Daca nu reusiti cu alt vector store, e acceptabil sa folositi si openai vector store dar vrem sa mentionati care au fost problemele si ce ati incercat pentru a le rezolva.
3. Nu va blocati in formate, tipuri de fisiere, dictionare, dupa cum spuneam , indicatiile de sus sunt doar orientative
4. Puteti folosi chatgpt sa generati documentele necesare , copilot pentru coding
5. **MOST IMPORTANT**: Vrem sa intelegeti codul si flowul pentru acest proiect