Task C: Evidence-Grounded RAG Chatbot

1. Workflow

The chatbot uses a hybrid retrieval-augmented generation (RAG) approach to answer questions about district-level climate resilience risks, grounded in field notes and portfolio data. All claims are cited with source file and quoted snippet.

Architecture:

- Data loading: Read files for 90 field notes, field notes index, risk label snippets, gold evaluation questions, district profiles, interventions catalog, top-10 funded district data
- Chunking: text is split into typed chunks - full_note, observation, index_meta, risk_snippet, header_partner, top10_intervention, top10_summary
- Fabricated snippet filtering: risk snippets whose text does not appear in the cited source file are discarded (~67 removed)
- Embedding engine: Use SBERT (all-MiniLM-L6-v2) for embedding if available, else TF-IDF + TruncatedSVD(128) dense embeddings as fallback
- Hybrid retriever:
  Step 1 - strict entity filter (district, file, partner, risk category).
  Step 2 - dense embedding similarity ranking within filtered set
- Answer generator: produces natural-language answers with separate citations. Conditional display of partner metadata, budget notes, and top-10 data based on query intent
- Gold evaluation: 15 gold questions tested for exact match and file retrieval
- Flask web UI with chatbot tab, example Q&As tab, and gold evaluation tab

2. Output

Web interface at http://localhost:5050

Three tabs:

- Chatbot: free-form question input with quick-access buttons for example questions
- Example Q&As: three worked examples with answers and citations
- Gold Evaluation: automated evaluation against 15 gold-standard questions, showing exact match rate, file retrieval rate, and failure mode analysis

API endpoints:

- POST /api/chat - send a query, receive answer + citations JSON
- GET /api/examples - returns the three example Q&As with responses
- GET /api/eval - returns gold evaluation results

Answer format:

Answers are natural-language sentences (e.g., 'For District 112 in North Azuria, critical components are back-ordered...'). Citations appear below in monospace font with source file, quoted snippet, and risk category.

3. Instructions

- Ensure the dalberg_case_study/ data contains field_notes/, field_notes_index.csv, risk_labels_seed.csv, gold_questions.json, district_profile.csv, interventions_catalog.csv
- Ensure dalberg_case_study/ output contains top10_codes.json and top10_districts.csv
- Ensure your task_c_chatbot.py file is in the same folder as the data and output folder
- Optionally, run Tasks A and B first to generate top10_districts.csv and top10_codes.json for funding data integration

- Run: python task_c_chatbot_v5.py
- Open http://localhost:5050 in a browser

4. Dependencies

- Python >= 3.8
- flask
- numpy
- sentence-transformers (all-MiniLM-L6-v2 model)
- scikit-learn (TfidfVectorizer, TruncatedSVD - used as embedding fallback)