Multimodal RAG — SIH Summary (List Format)

Project title

Multimodal Retrieval-Augmented Generation (RAG) for Offline Document, Image, and Audio Search

One-line problem

Centre handles PDFs, DOCX, images, screenshots, recorded calls, and notes but lacks a single search that understands and links across formats.

Goal

Build an offline-capable system that ingests documents, images, and audio; indexes them in a shared semantic space; supports natural language queries; and returns grounded answers with numbered citations and source navigation.

Key objectives

1. Ingest DOCX/PDF, images, screenshots, and audio recordings.
2. Extract usable text from each modality (OCR for images, speech-to-text for audio, parse DOCX/PDF).
3. Convert all content into embeddings in a shared vector space.
4. Provide natural-language search and chat that retrieves text snippets, images, and audio segments.
5. Return grounded answers with numbered citations and links to open source files or play audio.
6. Create cross-format links (example: transcript segment linked to a paragraph and a screenshot).

Core features (user-facing)

- Unified search/chat box for plain-language queries.
- Upload or drag-and-drop DOCX, PDF, images, or audio for ingestion.
- Optional: speak query to search via on-device speech recognition.
- Results show ranked items across modalities with numbered citations.
- Expand citation to open the source file, view full transcript, or inspect image metadata.
- Image-to-text and text-to-image semantic search.
- Optional: play audio segment inline and highlight its transcript.

Example queries

- "Show the report that mentions international development in 2024."
- "Find the screenshot taken at 14:32 that matches the account statement in doc_2024.pdf."
- "Upload this screenshot and find any related email or meeting transcript."

High-level architecture

1. Frontend
2. Single-page app with search/chat box, upload area, and results pane.
3. Ingestion layer
4. DOCX/PDF parser for text extraction and basic layout metadata.
5. OCR pipeline for images to extract text and bounding-box metadata.
6. Speech-to-text pipeline for audio files, with timestamps for segments.
7. Preprocessing
8. Clean and split text into chunks with provenance metadata.
9. Extract image visual embeddings plus OCR text and bounding boxes.
10. Extract audio segment transcripts and attach timestamps.
11. Embedding & indexing
12. Convert each chunk (text, image vector, audio-transcript chunk) into embeddings.
13. Store embeddings in a local vector index (FAISS) with metadata linking to original file, offset, and citation id.
14. Retrieval & Rerank
15. Semantic search over the vector index to retrieve top-k candidates across modalities.
16. Lightweight relevance reranker using a local language model or BM25 hybrid signal.
17. Response generation
18. Local LLM consumes retrieved context and user query to produce grounded answers with numbered citations.
19. UI for source navigation
20. Clickable citation list that opens the document page, image viewer with bounding box, or audio player at timestamp.

Offline considerations

- All components must run without internet access.
- Use local models and libraries: local speech-to-text, local image/text embedding models, and an on-prem LLM or open-source llama-family model.
- Keep models that fit target hardware or provide fallbacks (reduced-size models) for low-spec machines.

Suggested open-source components

- Document parsing: python-docx, pdfminer.six or PyMuPDF (fitz).
- OCR: Tesseract (local) or easy-ocr for better language support.
- Image embeddings: CLIP-family local model (open-source).
- Speech-to-text: Whisper local (tiny/base for speed, medium for accuracy).
- Text embeddings: SentenceTransformers (local) or on-device transformer models.
- Vector index: FAISS (flat/IVF) or hnswlib for memory-limited setups.
- Local LLM for grounding: Llama-like model via transformer inference (quantized weights where possible).
- Backend frameworks: FastAPI or Flask; Frontend: React or simple HTML+JS SPA.

Data model & metadata

- File-level metadata: filename, type, upload time, uploader, checksum.
- Chunk-level metadata: chunk id, parent file id, byte/character offsets, page number, image bounding box coords, audio start/end timestamps.
- Citation id: unique numeric tag used in user-facing answers.

Ingestion pipeline (step-by-step)

1. User uploads file. System assigns file id and stores original.
2. If PDF/DOCX: extract plain text with page numbers and simple layout metadata.
3. If Image: run OCR to extract text and keep bounding boxes; compute image embedding.
4. If Audio: run speech-to-text and split by silence or fixed-length windows; store timestamps.
5. Normalize text (lowercase, basic punctuation clean), split into chunks using semantic chunking and overlap.
6. Compute embeddings for each chunk and store in vector index with metadata.

Query flow

1. User submits natural-language query or voice query.
2. Convert query to embedding and run approximate nearest neighbor search against the index.
3. Retrieve top-N chunks across modalities and rerank by relevance and provenance quality.
4. Pass selected chunks and query to local LLM to generate a concise answer with numbered citations.
5. Present answer with clickable citations to open sources or play audio at timestamps.

Citation format (UI)

- Answer example: "There is a paragraph describing international development in 2024. [1]"
- Citation pane:
- doc_2024.pdf — page 7, para 2. Open | View raw text
- call_20240901.wav — 00:12-00:24. Play | Show transcript
- screenshot_1432.png — OCR text match. View image

Evaluation metrics

- Retrieval Recall@K for multimodal targets.
- Precision of citations (does the cited source actually support the answer).
- OCR word error rate and speech-to-text WER.
- Latency for typical offline hardware.
- Human evaluation: correctness, helpfulness, and transparency.

Demo & deliverables (SIH-ready)

1. Working demo: local server with UI to upload and query a bundled sample dataset.
2. Ingested sample set: mix of 10 PDFs, 10 DOCX, 20 images, 10 short audio clips with ground-truth transcripts.

3. Readme with setup and model size options for low and high resource machines.
4. Short demo video or scripted walkthrough.
5. Design document and architecture diagram.
6. Test cases and evaluation report.

Project milestones (example 8 week plan)

1. Week 1: Project setup, dataset collection, baseline parsers wired.
2. Week 2: OCR and speech-to-text pipelines integrated.
3. Week 3: Chunking, metadata, and embedding pipeline implemented.
4. Week 4: Vector index and basic semantic search working.
5. Week 5: Local LLM integration and basic answer generation.
6. Week 6: Cross-modal linking and UI for citations.
7. Week 7: Evaluation, optimization and model size tradeoffs.
8. Week 8: Polishing, demo, documentation, and final submission.

Team roles (suggested)

1. Project lead / integrator — overall architecture and demo.
2. Backend developer — ingestion pipelines and indexing.
3. ML engineer — embeddings, speech, OCR, and model tuning.
4. Frontend developer — SPA, search UI, and source viewers.
5. Tester / QA — evaluation metrics and test cases.
6. Documentation & presentation — readme, demo script, slides.

Risks and mitigations

- Risk: Large models do not fit target offline hardware. Mitigation: provide quantized smaller models or offload heavy preprocessing to a higher-spec build machine to create indexes that run lighter queries.
- Risk: Poor speech or OCR accuracy. Mitigation: use higher quality local models and allow manual correction workflows for critical data.
- Risk: Confusing citations from noisy chunks. Mitigation: enforce chunk quality thresholds and add provenance scoring.

Extensions (optional stretch goals)

- Real-time audio-to-document linking during live calls.
- Visual region search: click on an image region to find text/audio related to that region.
- Annotation UI for users to correct transcripts, OCR, or add tags that improve future retrieval.

Short-term test checklist for SIH demo

- Can upload 3 files of each modality and finish ingestion in under 10 minutes on target hardware.
- Can run 5 sample queries showing cross-modal results and playable audio snippets.
- Citations are numbered with working "Open" and "Play" actions.

End of summary