



Project: From Unstructured Clinical Notes to Actionable Care Plan

1) Scenario & Goal

You're given extra-large, unstructured clinical notes (think multi-page dictations). Build an end-to-end solution that:

1. parses and organizes the text into a navigable Table of Contents (sections/headers inferred),
2. produces a clinically-useful summary (problem list, history, meds, allergies, labs/imaging, assessment),
3. outputs treatment recommendations (Plan) with grounded references back to the source text (character spans/line ranges or paragraph IDs), and
4. exposes a simple CLI or notebook demo.

2) Dataset (free & public)

Use one of the following (pick one and document why):

1. **MTSamples** (medical dictations/transcriptions) — open, de-identified example notes across many specialties. Kaggle mirror is convenient.
https://metsamples.com/?utm_source=chatgpt.com
2. **HuggingFace** – Synthetic Clinical Notes (Asclepius or similar) — large synthetic discharge summaries; safe for experimentation.
https://huggingface.co/datasets/starmppcc/Asclepius-Synthetic-Clinical-Notes?utm_source=chatgpt.com

3) Requirements

A. Ingestion & Structuring

1. Accept raw .txt (very long). Perform de-duplication, section inference (e.g., detect headings like “HPI/ROS/Assessment/Plan,” or infer via topic segmentation).



2. Produce a machine-readable ToC with byte/char offsets for each section.

B. Summarization

1. Output a hierarchical summary:
 - Patient snapshot (age/sex if present), key problems, pertinent history, meds/allergies, objective findings, labs/imaging, and a concise Assessment.
2. For each bullet/claim, attach citations to source spans (e.g., [src: start_char–end_char] or [para 12–13]).

C. Recommendations (Plan)

1. Produce a prioritized treatment plan (diagnostics, therapeutics, follow-ups, risks/benefits).
2. Each recommendation must:
 - include rationale grounded in the source (with explicit citations to spans/paragraph IDs), and
 - include a confidence score (0–1) and hallucination guard note if the source evidence is weak/ambiguous.

D. Prompting / Modeling

1. Provide the solution prompt(s) you designed (system + user + any RAG instructions).
2. You may combine:
 - LLM (your choice) with chunking + retrieval,
 - lightweight section classifier or topic segmentation,
 - optional regex/ML heuristics for medical entities.
3. Must run locally on CPU or a single modest GPU ($\leq 12\text{--}24\text{GB}$). Document any paid APIs and provide an offline alternative (smaller open model) if you use one.

E. Output Formats

1. toc.json (section titles, offsets)



2. summary.json (structured fields + citations)
3. plan.json (recommendations with rationale, citations, confidence)

F. Quality & Safety

1. No PHI addition; stick to the de-identified text.
2. Show prompt hardening (instructions against fabrications, insist on citations, handle uncertainty).
3. Add unit tests for: section detection, citation alignment, and “no-evidence → no-claim.”

4) Deliverables

1. Repo with README (setup, data download instructions, how to run).
2. Prompt pack: final prompts + short “prompt reasoning” notes.
3. Metrics: simple intrinsic checks (e.g., % of summary bullets with valid citations; citation overlap Jaccard; hallucination rate measured by “orphan claims w/o source”).
4. Short slide deck (≤ 10 slides) summarizing: challenge, approach, models, data, results, failure modes, next steps. (Outline below.)
5. One-pager “Model Card” (intended use, limits, safety, bias).

5) Slide Deck Outline (≤ 10 slides)

1. Title & Problem: “Doctor’s notes → Actionable care plan” (why it’s hard).
2. Data: which dataset, note length, specialties, prep steps, de-ID mention. (Cite source.)
3. System Design: ingestion → segmentation/ToC → retrieval → LLM reasoning → outputs.
4. Prompt & Retrieval Strategy: chunking, window sizes, prompt schema, citation rules.
5. Models & Tools: LLM(s), embeddings, vector store, libraries (and why).
6. Evaluation: citation coverage, factuality checks, examples of caught hallucinations.



ethermed

7. Results Demo: before/after snippets, ToC screenshot, summary & plan JSON.
8. Safety & Limits: uncertainty handling, edge cases (conflicting notes).
9. Cost & Performance: latency, token/\$\$ estimates, tradeoffs.

6) Time Expectation

1. This should only take 6-12 hours. Don't over-engineer.
2. We're evaluating problem decomposition, practical ML/LLM skills, code quality, and honest assessment of limitations.