Full-Stack Engineer Technical Challenge

Goal

Build a small learning web app that ingests arbitrary learning material, uses AI to extract main themes, generates a concise lesson (text + voice), and presents & stores the results.

Timeline

- **Phase 1 (3 days):** Choose strategically what to work on to present for feedback.
- **Phase 2 (4 days):** Iterate from feedback; present final demo, code walkthrough, and slides.

Core Requirements

1. **Ingestion**
    - Upload a file (PDF/TXT/Docx).
    - Handle at least two types (e.g., PDF + plain text)
2. **AI Processing**
    - Extract **3–7 main themes** from the material.
    - Generate a **short lesson** (≈250–400 words) aimed at an employee upskilling for work.
    - Generate **voice narration** for the lesson (in-page player).
    - Cite the **top 2–3 source snippets** (line/para refs).
3. **Frontend (Designed)**
    - Clean, responsive UI showing:
        - Material metadata (title/source/length)
        - Themes (tags or cards)
        - Lesson (text + embedded audio player)
    - A “for work” tone and layout (e.g., “What you’ll learn,” “Apply at work,” “Key takeaways”).
4. **Persistence**
    - Store the **raw material**, **themes**, **and lessons** in a database with appropriate schema.
    - Associate artifacts by an **ingestion_id**.
5. **Docs & Presentation**
    - **Architecture Note (2–3 pages):** key decisions, trade-offs, data flow diagram, sequence diagram.
    - **BRS (1 page):** user goals, outcomes, constraints, success criteria.
    - **FRS (2–3 pages):** endpoints, data model, error states, non-functionals.

    - **Slides (≤5):** problem → approach → design → stack → risks → roadmap.
6. **Dev Hygiene**
    - Public **Git repo** with clear README, setup/run scripts, .env.example.
    - Basic tests (unit or integration) for 1–2 critical paths.
    - Linting/formatting.

Acceptance Criteria (What we'll verify)

- Upload/ingest works for at least two source types.
- Themes are coherent and grounded in the source (spot-check by reading cited snippets).
- Lesson is concise, workplace-relevant, and matches themes.
- Audio plays reliably and matches lesson text.
- Data persists and is retrievable by ingestion_id.
- Docs are complete and consistent with the implementation.

Deliverables

- URL (if hosted) or local run steps.
- Repo with README and .env.example.
- **/docs** folder: Architecture Note, FRS, BRS (PDF or markdown).
- Slide deck (PDF or PPTX).
- Phase-2 presentation (demo + code walk-through).

Evaluation Rubric (weights)

- Product thinking & UX for learners: **20%**
- Correctness & reliability of ingestion/analysis: **25%**
- Code quality & tests: **20%**
- Architecture & trade-offs: **20%**
- Docs & presentation clarity: **15%**