User Journey and FAQ’s

User Journey – Document Insight & Engagement System

1. Context & Problem

Users (e.g., researchers, students, professionals) deal with large volumes of documents daily — research papers, business reports, study material, etc. Over time, it becomes impossible to remember all details or connect insights across these documents.

2. Goal of the System

Enable users to:

- Quickly surface related, overlapping, or contradicting information from their personal document library.
- Use AI/LLM-powered capabilities to enhance understanding and engagement — without introducing unrelated or ungrounded external knowledge.

3. Journey Flow

Step 1 – Reading & Selection

- **Trigger:** User is reading a document within the system.
- **Action:** User selects a portion of text (e.g., a scientific method, a business strategy, a key metric).
- **System Response:** Instantly surfaces relevant sections from past documents in the user’s library. Includes overlaps, contradictions, and related examples. Uses semantic search and optionally an LLM to ensure context-aware matching.

Step 2 – Insight Generation

- **Goal:** Go beyond finding similar text.
- **System adds value by:** Highlighting contradictory viewpoints or alternate uses of the selected concept. Offering contextual insights that enrich understanding. Grounding all results in documents the user has actually read or uploaded — not generic web sources.

Step 3 – Rich Media Experience

- **Optional Action:** User requests an audio overview / podcast for the selected topic.
- **System Capabilities:** Generates a natural-sounding, engaging audio summary. Pulls content only from user’s documents to maintain trust and accuracy. Structures audio for easy listening — highlights key points, contrasts perspectives, and connects concepts.

4. Key UX Considerations

- **Speed:** Minimal delay between text selection and insight surfacing — keeps user engaged.
- **Relevance:** High-quality matches ensure trust in the system.
- **Engagement:** Audio should be natural and dynamic, not robotic.
- **Extensibility:** Users can explore beyond core tasks (bonus features) while staying aligned with the main flow.

5. Example Use Case

A researcher reading a paper on “neural network training techniques” selects a paragraph on “transfer learning.” The system instantly shows:

- Similar methods in 3 previous papers.

- Contradictory findings from another study.
- An audio podcast summarizing these points for quick listening on the go.

Feature Clarifications

1.What exactly does the “Snippets” feature require in terms of functionality and output?

- User journey:
    1. User makes text selection on sentences of interest.
    2. System figures out which sections (PDF + Heading) from past + current PDFs are most relevant to the selection. And which sentences (“snippets”) from those sections are most relevant.
    3. If the user clicks one of these snippets/sections, the corresponding PDF (which may be different from the current PDF) should be shown with that section in view.
    4. The system also generates insights such as examples, contradictions, trends, etc. Based on the documents, possibly using LLMs.
    5. And give an option to generate an engaging podcast based on these snippets.

2. In cases where relevant sections across PDFs have different meanings, should the engine prioritize:
    a. Keyword similarity, or
    b. Semantic meaning/context?
    c. Up to you – we will measure relevance.
3. For “podcast mode,” is it just text-to-audio playback, or is additional formatting/content design expected?
    a. We will judge based on the quality of the podcast – including content relevance, flow, naturalness, ...
4. What is the accuracy benchmark for offline relevant section extraction (e.g., 80%+ as mentioned before)?
    a. We have our internal rubric that weights relevance, insightfulness, etc.

File Handling & Input Flow

1. Will the user upload all PDFs together, or can they upload them one by one?
    - A bunch of files initially followed by an additional PDF. This additional PDF should be shown in the UI, and the insights generated.
2. What exactly are “past PDFs,” and how are they different from PDFs uploaded for the first time?
    - User Journey is that there are a few files which were originally uploaded by the user initially, and then the user comes back and adds additional file, and generated insights on it.
3. For this round, are we expected to run both Round 1A and 1B models together and produce a combined result?
    - For the finale not all of 1a and 1b is required to run, you need to run only part which can capture the section in the finale challenge.
4. Are the “sections” and “snippets” in the requirement referring to:
    - The sections generated in Round 1A
    - The summaries generated in Round 1B
    - Or something else entirely?

    - Sections are same as Round 1a from the original PDFs, and snippets are relevant sentences from those sections.
5. Is the interface supposed to work with a single PDF at a time or across multiple PDFs at once?
    - The document which is accessed should be visible and the insights and podcasts will need to be generated across all documents.
6. Should the application connect information between multiple documents automatically, or only when prompted by the user?
    - This will be based on text selection

Prompt Handling & User Interaction

1. Should the UI allow the user to input a prompt (like in Round 1B), or should it derive connections automatically?
    1. Should be based on selected text. Input prompt is not required.
2. Should the application retain previous user interactions, like an LLM conversation history with tabs/sessions?
    1. Files uploaded/indexed should be retained. There is no prompt bar/conversational context that needs to be retained.

API Integration & LLM Usage

1. Can we modify the Round 1B execution logic now that we can use APIs and LLMs, or must the original logic remain untouched?
    1. Yes, sure.
2. For the "snippets" feature, can we use API calls, or must it be entirely local?
    1. You can use LLM calls.
3. Should we continue with the same base model from Round 1B or replace it with an API-based approach?
    1. You can use LLM calls.
4. Are we allowed to integrate Azure TTS (as mentioned) for podcast mode, and can we use Google TTS for local development?
    1. Sure!

Online vs. Offline Requirements

1. Which features must work **offline** and which can be **online**?
    1. Your external calls should be limited to LLM and TTS and Embed API
2. For "persona-based section extraction" (and other base features), must it still work offline?
    1. Persona based extraction is not part of Finale.
3. If the Adobe PDF Embed API is used for rendering, it requires an internet connection — would that still count as "offline" usage?
    1. PDF Embed API is allowed to make network usage
4. Are we expected to add an offline fallback for PDF rendering in case internet is unavailable?
    1. As mentioned Embed API can use network. PDFs will be uploaded.
5. Is the "relevant section" feature explicitly required to run offline?
    1. Your external calls should be limited to LLM and TTS and Embed API

Backend & Hosting

1. Are we required to submit a fully functional backend, or is a demo video sufficient?
    1. Code should be fully functional and runnable via docker. But demo video is also required.
2. Can the backend be hosted locally for judging, or must it be deployed online?
    1. We will run the docker in our environment for evaluation.
3. If the PDF Embed API requires hosting on a domain, can we use `localhost` for local testing?
    1. Yes
4. Do we need to host frontend and backend separately, or can they be combined?
    1. It should be combined so it can be run in a single docker image. You do not have to host the server for evaluation, we will run the docker in our environment for evaluation.

Execution Time & Performance Constraints –

1. In Round 1B, persona-based queries had a 60-second execution limit. For the finale, the limit is 10 seconds — does this also apply to persona-based queries?
    1. There is no set limit, but the scoring will factor in speed. For ingesting past documents, the same limit as previous round applies.
2. Is the 10-second execution limit per document or for the whole batch, regardless of PDF size (e.g., 20 pages vs. 150 pages)?
    1. Refer above questions.
3. Is there still a model size constraint of under 1 GB? Can we use a larger model if it processes within 10 seconds?
    1. For finale, there is no limit, but for limiting the docker image size, we would prefer the model size to be less than 20 GB.

Embed API useful links

Events: https://developer.adobe.com/document-services/docs/overview/pdf-embed-api/howtodata/#basic-events

- Events like SELECTION_END, PAGES_IN_VIEW_CHANGE, etc might be useful.

getSelectedContent: https://developer.adobe.com/document-services/docs/overview/pdf-embed-api/howtos_ui/#getselectedcontent

goToLocation: https://developer.adobe.com/document-services/docs/overview/pdf-embed-api/howtos_ui/#gotolocation

Zoom APIs: https://developer.adobe.com/document-services/docs/overview/pdf-embed-api/howtos_ui/#zoom-apis - Might be needed if you want to build your own UI for zoom but otherwise zoom buttons are already present in the default viewer that comes with PDF Embed API.

LLM and TTS generation implementation links

Chat_with_LLM: https://github.com/rbabbar-adobe/sample-repo/blob/main/chat_with_llm.py

TTS/generate_audio: https://github.com/rbabbar-adobe/sample-repo/blob/main/generate_audio.py