You’re a **Technical APM** on the AI Browser team. The browser already supports basic AI Q&A over a single page. Your team now wants to add an **“AI Research Workspace”**: a space where users can save links/notes and ask AI to summarize or compare everything in that workspace.

This mini-assignment is designed to be **doable in one day** (≈3–5 focused hours), and you are **encouraged to use AI tools** (ChatGPT, Copilot, etc.) for assistance. We care most about your **judgment, structure, and trade-offs**.

What You Need to Deliver

You’ll submit **two things**:

1. **A short product + technical doc (max 3 pages)**
2. **A 1-page Technical Trade-offs & Tracking note**

Deliverable 1 – 3-Page Product + Tech Doc

Format: Google Doc / Notion / PDF is fine.
Hard limit: **3 pages** (excluding title page, if any).

Section A – Persona & Problem (≈½ page)

1. Pick **one primary persona** (e.g., student, analyst, founder, researcher).
2. In a short paragraph, explain:
    - What they’re trying to do when doing research in the browser.
    - What’s painful or inefficient about their current workflow.
    - Why an **AI Research Workspace** would help them.

Keep it concrete. Imagine a real person doing a real task (e.g., “comparing 5 tools for a college project”).

Section B – Core Features & Single User Journey (≈1 page)

1. List **exactly 3 core features** you would include in **v1** of the AI Research Workspace.
    - 1–2 sentences each: what it does + why it matters for your persona.

2. Describe **one end-to-end user journey** in a few bullet points, for example:
   “As <persona>, I want to <do X> so that <Y outcome>.”

   Then 6–10 bullets showing the flow, e.g.:
   - User creates a new workspace.
   - User adds 5 links and 2 PDFs.
   - User asks AI: “Summarize the key pros/cons of each option.”
   - User reviews and tweaks the summary.
   - …

Focus on clarity and practicality, not covering every edge case.

Section C – Light Architecture & One API Sketch (≈1–1.5 pages)

You don’t need a full system design—just show you can think **like a TAPM**.

1. **Key components (bullet list, no diagram required)**
   Name **3–5 components** and 1 line about each, e.g.:
   - *Workspace Service* – stores workspaces and their items.
   - *Content Ingestion Service* – fetches and cleans page/PDF content.
   - *LLM Orchestrator* – builds prompts and calls the AI model.
   - *Analytics/Events Pipeline* – records feature usage events.

2. **Single flow, step-by-step (5–7 bullets)**
   Describe the flow when the user clicks:
   “Ask AI: Summarize this workspace”, for example:

   - Browser sends request to backend with workspace ID.
   - Backend fetches items for that workspace.
   - Backend selects or chunks content to stay within model limits.
   - Backend calls LLM with a prompt and the selected content.
   - LLM returns a summary; backend returns it to the browser.
   - Browser displays the summary to the user.

3. **One example API (high level)**

   Define **one** key endpoint, for example:
   - Name: `POST /workspaces/{id}/summarize`
   - Brief purpose: *“Generate an AI summary for the given workspace.”*
   - Example **request JSON** (just top-level fields & types).

    - Example **response JSON** (summary text + maybe metadata like `tokens_used`, `latency_ms`).

It's okay if the JSON is rough; we're looking for structure and intent, not perfection.

Section D – Simple Metrics (≈½ page)

List **1 North Star metric** and **3 supporting metrics** for this feature.

Examples (you can choose your own):

- North Star:
    - `# of weekly active workspaces where AI was used at least once`
- Supporting:
    - `avg AI queries per active workspace`
    - `% of users who return to the same workspace within 7 days`
    - `avg latency of AI summaries`

Add **1–2 sentences** on how you'd know if the feature is "working" after launch (e.g., "I'd expect AI-active workspaces to have higher 7-day return rate than non-AI workspaces.").

Deliverable 2 – Technical Trade-offs & Tracking (≈1 page)

Part A – Two Implementation Options & Trade-off (≈½ page)

Pick **one part** of your design (for example, "how AI summaries are generated" or "how content is stored/indexed") and:

1. Describe **two plausible implementation approaches** in a couple of sentences each.
2. For each approach, list **2–3 pros and 2–3 cons** (e.g., latency, complexity, cost, UX quality).
3. Choose **one approach** you would ship first, and explain **why** in 3–4 sentences.

> We don't need deep engineering details; we're looking for structured trade-off thinking.

Part B – Tracking Plan Snippet & Sample Query (≈½ page)

1. Define **3 analytics events** related to this feature. For each, specify:
    - **Event name** (e.g., `workspace_created`, `ai_summary_requested`, `ai_summary_viewed`)
    - **When it fires**
    - **Key properties** (e.g., `workspace_id`, `items_count`, `model_version`, `latency_ms`)

2. Write **one product question** you'd want to answer with data, e.g.:
   "Do users who trigger an AI summary come back to their workspace more often than those who don't?"

3. Sketch a **pseudo-SQL query** that could help answer that question. It doesn't have to be perfect SQL; just show what tables/events you'd use and how you'd join/aggregate them.