For 'Cline' AI: Scoped Implementation Plan (v1.0) & Master Prompt (v1.3)

PRIORITY 1 DIRECTIVE: Forget all previous project documents, conversations, and instructions related to the 'Second Brain' project. Disregard any document titled 'PRD v7' or 'PRD v8'. Your memory of this project's requirements is now wiped clean.

Master Prompt for 'Cline' Implementation Assistant:

"You are to act as a senior full-stack software engineer. Your new, sole, and exclusive source of truth for all subsequent tasks is the 'Scoped Implementation Plan' document provided below. You must not deviate from this plan or introduce any functionality not explicitly defined within it.

Your instructions are as follows:

1. You will execute the tasks in the exact sequence they are listed in the plan. Your goal is to complete the entire plan in a single, uninterrupted run.

2. Before creating any new file or code structure, you must first check if a file with that name or purpose already exists. If it does, you must read, analyze, and refactor the *existing* code to meet the new requirements, preserving sound logic. Do not blindly overwrite existing work.

3. Before beginning any task, you must read its corresponding detailed sub-section and its 'Verification & Acceptance Criteria' in their entirety.

4. You will maintain an internal, rolling summary of your progress as you complete each task. You will not stop or ask for confirmation after each step. You will only present this full summary in a final report at the very end of the entire process.

5. If you encounter an issue or an error that prevents you from completing a task as specified, you will not stop. Instead, you will execute the following error handling protocol: a) Log the failed task number and the specific error message in detail. b) Add the failed task to a 'Parking Lot' section for later review. c) Immediately proceed to the next task in the sequence. You will continue this process until you have attempted every task in the plan.

6. Upon attempting all tasks, you will generate a single, final 'Execution Report.' This report must contain three sections: 1) A 'Completed Tasks' summary log. 2) The 'Parking Lot' detailing all failed or skipped tasks and their error messages. 3) A 'Dependency Analysis' explaining if any failures were caused by a previous failed task."

AI: Scoped Implementation Plan (v1.0)

Context-Setting Preamble:

You are building a **'Truth-Seeking Engine,'** an expert system designed to be an objective, evidence-based thought partner for its user. Its primary goal is to **challenge and guide** by synthesizing internal knowledge with external, time-sensitive information, providing a multi-axis analysis of any topic. The initial user is a **'Professional Strategist'** who must analyze complex information to build and defend high-stakes arguments. Your immediate task is to implement the **'Debate Prep' workflow** by following the sequential plan below.

Part 4: The Scoped Implementation Plan (with Acceptance Criteria)

Phase 1: Foundation & Setup (Target: 2 Days)

- **Task 1.1: Project Initialization & Configuration**
  - **Details:** Ensure the Next.js project uses the .env file for all credentials. Create a central configuration file at lib/config.ts to export configured singleton instances of the Prisma client, Redis client, and Google AI client.
  - **Verification & Acceptance Criteria:**
    - **1.1.1:** The lib/config.ts file must exist and successfully export the required clients without hardcoding any secret keys.
    - **1.1.2:** The application must successfully connect to all three services (Supabase, Redis, Google AI) using the configured clients upon startup.
- **Task 1.2: Database Setup & pgvector Enablement**
  - **Details:** Prepare the Supabase database by enabling the vector extension and synchronizing it with the Prisma schema.
  - **Action 1:** Execute SQL: CREATE EXTENSION IF NOT EXISTS vector;
  - **Action 2:** Run CLI: npx prisma migrate dev --name "alpha-setup-and-pgvector"
  - **Action 3:** Run CLI: npx prisma generate
  - **Verification & Acceptance Criteria:**
    - **1.2.1:** The pgvector extension must be listed as "enabled" in the Supabase dashboard.
    - **1.2.2:** The Prisma migration must complete without errors, and all tables defined in prisma.schema must exist in the Supabase database.
- **Task 1.3: Queue & Worker Scaffolding**
  - **Details:** Create the entry point for the background worker and verify its connection to BullMQ.
  - **Action:** Create app/worker/index.ts and instantiate a new Worker connecting to the "jobQueue".
  - **Verification & Acceptance Criteria:**
    - **1.3.1:** When a job is manually added to the Redis queue, the running worker process must log the job's data to the console.

Phase 2: Core Engine Implementation - The Agent Chain (Target: 5 Days)

- **Task 2.1: Implement 'Ingestion Agent'**
  - **Details:** Create the first agent in the worker's chain to fetch and sanitize raw text content from a URL.
  - **Verification & Acceptance Criteria:**
    - **2.1.1:** Given a YouTube URL, the agent must return the full, sanitized text transcript.
    - **2.1.2:** Given an article URL, the agent must return the full, sanitized text content of the article body.
    - **2.1.3:** If a URL is unreachable or returns a 404 error, the job must fail gracefully and report the error.
- **Task 2.2: Implement 'Defensive Analysis Agent'**
  - **Details:** This agent deconstructs an opponent's document to find weaknesses.
  - **Verification & Acceptance Criteria:**
    - **2.2.1:** Given a test text containing a known logical fallacy (e.g., a "strawman argument"), the agent's JSON output must correctly identify and quote the fallacy.
    - **2.2.2:** The agent must successfully pass its structured JSON output to the next agent in the chain.
- **Task 2.3: Implement 'Internal Coherence Agent'**
  - **Details:** This agent compares the opponent's arguments against the user's own knowledge library using pgvector.
  - **Verification & Acceptance Criteria:**
    - **2.3.1:** Given an argument that is known to be contradicted by a document in the user's test library, the agent's output must correctly identify the relationship as "CONTRADICTS" and cite the correct internal document.
    - **2.3.2:** The pgvector search must complete successfully and return the most relevant results.
- **Task 2.4: Implement 'Synthesis & Opinion Agent'**
  - **Details:** This final agent synthesizes all prior findings into the comprehensive "opinion" document.
  - **Verification & Acceptance Criteria:**
    - **2.4.1:** The agent's output must be a single, valid JSON object that conforms to the schema required by the Interactive Report frontend components.
    - **2.4.2:** The generated text must adhere to the "disclaimer-ready" persona, using analytical language rather than prescriptive commands.
- **Task 2.5: Implement 'Persistence Agent'**
  - **Details:** This agent saves the complete analysis to the database within a single transaction.

- **Verification & Acceptance Criteria:**
  - **2.5.1:** Upon successful completion of a job, a new ContentSummary record and its associated VectorChunk records must exist in the database.
  - **2.5.2:** The parent BatchJob status must be updated to "completed."
  - **2.5.3:** If any database write fails, the entire transaction must be rolled back, and no partial data should be saved.

Phase 3: Frontend Integration & UI Build (Target: 4 Days)

(Each task has an implicit acceptance criterion of being fully responsive and visually polished)

- **Task 3.1: Build the "Debate Prep" UI** (DebatePrepView.tsx)
  - **Verification & Acceptance Criteria:**
    - **3.1.1:** A user must be able to upload a PDF file, which successfully triggers a call to the /api/v1/batch endpoint.
- **Task 3.2: Build the Interactive Report Viewer** (InteractiveReport.tsx)
  - **Verification & Acceptance Criteria:**
    - **3.2.1:** The component must correctly fetch and render all parts of a completed analysis JSON, including all interactive sub-components.
- **Task 3.3: Build the "Ingestion Sandbox" UI** (IngestionSandbox.tsx)
  - **Verification & Acceptance Criteria:**
    - **3.3.1:** A user must be able to see a list of external sources discovered during an analysis.
    - **3.3.2:** Clicking the "Add to Library" button for a sandboxed item must successfully trigger a new batch processing job for that item.
- **Task 3.4: Final Wiring & Polish**
  - **Verification & Acceptance Criteria:**
    - **3.4.1:** The entire application must be migrated to .tsx with no type errors.
    - **3.4.2:** The end-to-end "Debate Prep" workflow must be fully functional, from file upload to viewing the final interactive report.