Product Requirements Document (PRD)

Project Name: SpecGap (Council Edition)

Version: MVP-1.0 (Hackathon Build)

Architecture: 3-Agent Recursive Loop (Draft → Critique → Final)

1. Problem Statement

Solo founders and freelancers often sign contracts that conflict with their technical capabilities or financial reality. They lack a team to cross-reference documents.

- **The Gap:** A contract promises "Unlimited Storage" (Legal), but the Tech Spec builds "Local Storage" (Engineering). No single human reads both docs simultaneously to catch this.
- **The Solution:** An AI Council of 3 Agents (Legal, Business, Finance) that reads all docs, debates the risks among themselves, and presents a unified verdict.

2. User Flow (The Demo Experience)

1. **Input:** User uploads Contract.pdf and Tech_Spec.docx (Total ~5 pages).
2. **Domain Selection:** User selects "Software Engineering" (Configures Agent domain prompt Knowledge).
3. **The Council Session (The "Waiting" State):**
    - *Visual:* 3 Avatars (Legal, Biz, Finance).
    - *Round 1:* Avatars light up (Reading).
    - *Round 2:* Lines connect the avatars (Debating/Comparing).
    - *Round 3:* Avatars align (Finalizing).
4. **Action:** User sees a stack of "Flashcards."
    - Swipe Right = "Fix" (Add to Patch Pack).
    - Swipe Left = "Ignore".
5. **Output:** User downloads the "Patch Pack" (Files generated from Swiped Right items).

3. Technical Architecture: The "3-Loop" Council

Input Data: 5 Pages of Text = approx. **5,000 Tokens**.

Round 1: The Blind Draft (Parallel Execution)

- **Goal:** Independent, unbiased analysis.
- **Input:** Raw Documents Only.
- **Logic:**
    - Agent A (Legal) reads Docs → Outputs Draft 1.

  - Agent B (Business) reads Docs → Outputs Draft 1.
  - Agent C (Finance) reads Docs → Outputs Draft 1.
- **API Calls:** 3.

Round 2: The Cross-Check & Update (Parallel Execution)

- **Goal:** Conflict Detection.
- **Input:** Raw Documents + **The Other 2 Drafts**.
- **Logic:**
  - Agent A reads (Docs + Draft B + Draft C) → Compares with own Draft A → Outputs **Draft 2**.
  - Agent B reads (Docs + Draft A + Draft C) → Compares with own Draft B → Outputs **Draft 2**.
  - Agent C reads (Docs + Draft A + Draft B) → Compares with own Draft C → Outputs **Draft 2**.
- **API Calls:** 3.

Round 3: The Final Verdict (Parallel Execution)

- **Goal:** Final Consensus.
- **Input:** Raw Documents + **The Other 2 Updated Drafts**.
- **Logic:**
  - Agent A reads (Docs + Draft 2B + Draft 2C) → Finalizes own Flashcards.
  - Agent B reads (Docs + Draft 2A + Draft 2C) → Finalizes own Flashcards.
  - Agent C reads (Docs + Draft 2A + Draft 2B) → Finalizes own Flashcards.
- **API Calls:** 3.

4. Engineering Budget: Tokens & Limits

Constraint: Gemini 3 Flash (Free Tier).

- **Rate Limit:** 15 Requests Per Minute (RPM).
- **Token Limit:** 250,000 Tokens Per Minute (TPM).

The Math (Per User/Session)

Round	**Input Context (Approx)**	**Output Generated**	**API Calls**	**Total Token Cost**
Round 1	5,000 tokens (Docs)	~500 tokens (x3)	3	**16,500**

Round 2	5,000 (Docs) + 1,000 (Peers)	~500 tokens (x3)	3	**19,500**
Round 3	5,000 (Docs) + 1,000 (Peers)	~500 tokens (x3)	3	**19,500**
TOTAL			**9 Calls**	**~55,500 Tokens**

Safety Check

- **Token Usage:** ~55,500 / 250,000 Limit. (**22% Utilization** - ✅ SAFE).
- **Request Usage:** 9 / 15 Limit. (**60% Utilization** - ✅ SAFE).
- **Note:** The entire process must take at least **45 seconds** to clear the Request Limit per minute safely for the next demo run.

5. Deliverables (The Patch Pack)

When the user swipes "Right" (Accept Fix) on the final flashcards, the system generates:

1. **Contract_Addendum.txt:** Specific clauses to insert (from Legal Agent).
2. **Spec_Update.md:** Missing requirements added (from Business Agent).
3. **Negotiation_Email.txt:** A drafted email to the client pushing back on terms (Synthesized).