Slide: The "Algorithm": A Hybrid AI Architecture

Headline: The "Algorithm": A Hybrid AI Architecture

Key Points:

- **The Problem: Loan analysis has two distinct challenges:**
  1. **Known Unknowns: The user has specific questions they need answered (e.g., "What is my interest rate?", "What's the late fee?").**
  2. **Unknown Unknowns: The user has *no idea* what predatory clauses to even look for (e.g., "Balloon Payments," "Mandatory Arbitration").**
- **Our Solution: A simple Q&A bot isn't enough. We architected a Hybrid AI Platform to solve *both* problems simultaneously. Our system has two core processes:**
  1. **The Comprehension Engine (RAG): For flexible, user-driven Q&A.**
  2. **The Interrogation Engine (Interceptor): For proactive, system-driven risk detection.**

Slide: Process 1: The Comprehension Engine (How it Works)

Headline: Process 1: The Comprehension Engine (For User Q&A)

- **Why We Chose This: To give the user *flexibility*. This engine allows the user to ask any specific, open-ended question about their document in natural language.**
- **How It Works (RAG - Retrieval-Augmented Generation):**
  1. **Ingest: When a document is uploaded, we parse it and break it into small, semantic "chunks" of text.**
  2. **Vectorize: Each chunk is converted into a vector (a numerical representation of its meaning) and stored in a high-speed FAISS vector database.**
  3. **Retrieve: When a user asks a question ("What is the late fee?"), we vectorize the *question* and use the database to instantly find the 5-10 *most relevant* chunks of text from the document.**
  4. **Generate: We feed *only* those relevant chunks (as "context") and the user's question to the LLM (Gemini 2.0 Flash) with a prompt: "Using *only* this context, answer this question."**
  5. **Result: The user gets a precise answer that is 100% grounded in the facts of their document, eliminating LLM hallucination.**

(This slide proves you built a sophisticated RAG system, but frames it as "Process 1" of your platform.)

Slide: Process 2: The Interrogation Engine (How it Works)

Headline: Process 2: The Interrogation Engine (For Proactive Risk-Finding)

- **Why We Chose This: The Comprehension Engine is *passive*—it only answers what it's asked. It cannot find dangers the user doesn't know to ask about. This engine *actively hunts* for them.**
- **How It Works (The "Interceptor" Loop):**
    1. **Load Knowledge: We first load a human-curated Risk Knowledge Base (risks.md). This file is the "brain," defining "villain" clauses like "Prepayment Penalty" and "Mandatory Arbitration."**
    2. **Input Document: The user provides the raw loan text.**
    3. **Interceptor Loop: The backend *loops* through every single risk in its Knowledge Base.**
    4. **Craft Prompt: For each risk, it generates a *surgical prompt* for the LLM:**

"You are an expert. Find *only* this one risk: [Risk Name]. Here is its definition: [...]. Scan the entire document and return *only* the JSON I've requested."

5. **Aggregate Report: The system collects all the individual JSON responses and aggregates them into a single, simple "Risk Report" (e.g., "3 Risks Found / 7 Checks Passed") for the user.**

Slide: Why This Hybrid Approach is the Right Solution

Headline: Why This Architecture is the Complete Solution

- **Comprehension Engine (RAG):**
    - **Pro: Provides 100% *flexibility*. The user can ask anything.**
    - **Con: It's *passive*. It can't find risks you don't know exist.**
- **Interrogation Engine (Interceptor):**
    - **Pro: It's *proactive* and *auditable*. It guarantees the document is checked for the 10 most critical dangers. Its findings are based on our *expert knowledge*, not a "black box" guess.**

    - **Con: It's *inflexible*. It can *only* find what's in its Knowledge Base.**
- **The Synergy: By combining these two processes into one platform, we built a complete solution. The Interceptor protects the user from *unknown* dangers, while the RAG Chat empowers them to investigate their *known* questions.**