Google Cloud H2S

Gen AI Exchange Hackathon

Team Name : AC

Team Leader Name : Srijan Verma

Problem Statement : [Student]Generative AI for Demystifying Legal Documents

Brief about the prototype:

RAG System for Legal Document Analysis

An AI-powered system that transforms complex insurance policies into understandable, actionable insights. The prototype combines vector databases (ChromaDB) and graph databases (Neo4j) with advanced language models to automatically analyze legal documents and provide structured decision-making support.

***Core Functionality*:**

- Processes PDF insurance documents using intelligent text chunking
- Generates embeddings using Qwen3-Embedding-0.6B model
- Performs hybrid retrieval combining semantic similarity and relationship mapping
- Uses DeepSeek-R1-Distill-Qwen-1.5/7B (Reasoning) model to generate structured JSON responses.

Key Innovation: The dual-database architecture captures both semantic meaning and structural relationships within legal documents, enabling more accurate and comprehensive analysis than traditional single-approach systems.

Output: Provides automated eligibility decisions, payout calculations, and exclusion identification with specific clause references - making complex legal language accessible to both professionals and consumers.

Opportunity should be able to explain the following:

- **How different is it from any of the other existing solutions?**

Current Solutions:

Traditional document search relies on keyword matching, missing semantic context

Existing **RAG** systems use only vector databases, losing document structure and relationships

Manual policy analysis requires legal expertise and takes hours/days

Rule-based systems are rigid and can't handle policy variations

Our Differentiation:

Dual-Database Architecture: Combines ChromaDB (semantic search) + Neo4j (relationship mapping) - no existing solution uses this hybrid approach for legal documents

Structured Decision Output: Generates JSON responses with clause-level justifications, unlike generic chatbot responses

Graph-Enhanced Retrieval: Maps relationships between policy clauses, capturing dependencies that vector-only systems miss.

Domain-Specific Prompt Engineering: Tailored specifically for insurance policy analysis with structured field extraction

- **How will it be able to solve the problem?**

End-to-End Automation:

Document Ingestion: Automatically processes PDF policies, extracting and chunking text intelligently

Relationship Mapping: Neo4j captures how different policy sections interact and depend on each other

Semantic Understanding: Uses Qwen3-Embedding model to understand context beyond keywords

Intelligent Retrieval: Hybrid search finds relevant clauses using both similarity and structural relationships

- **USP of the proposed solution**

"The only AI system that understands both the meaning AND structure of legal documents"

Key Differentiators:

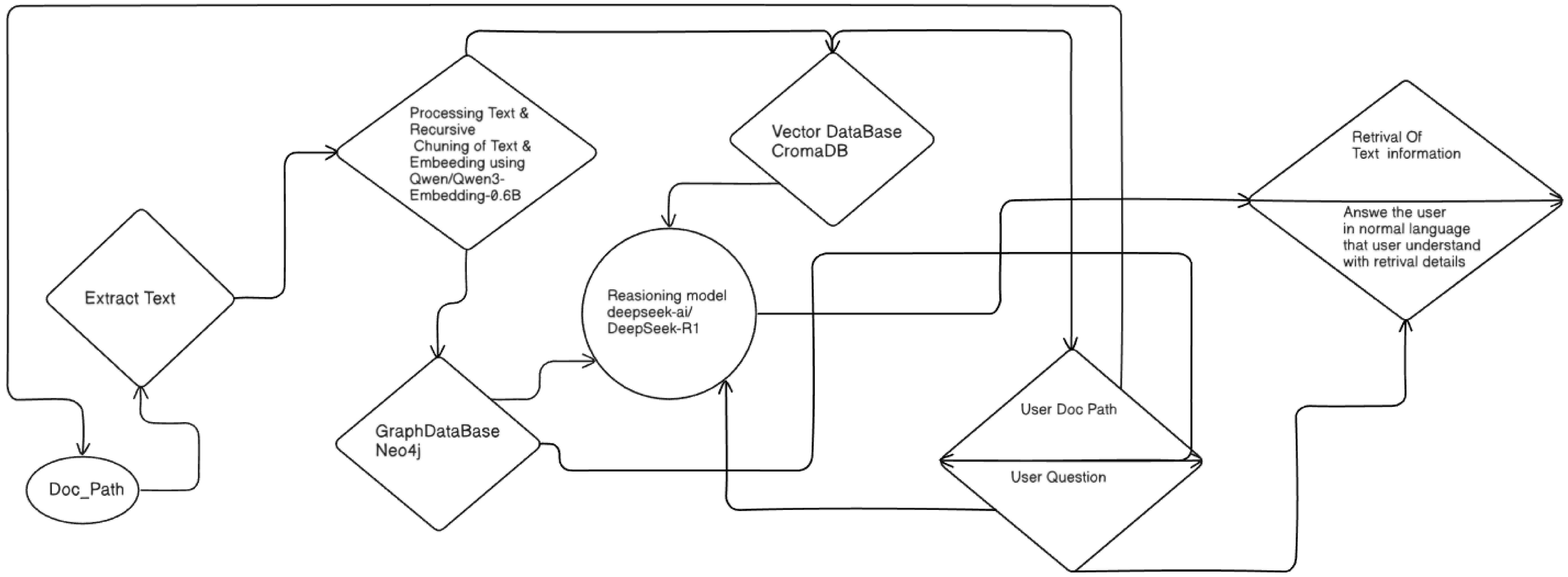
Hybrid Intelligence ArchitectureVector + Graph database combination is unique in legal document analysis
Captures semantic meaning while preserving document relationships
Justifiable AI DecisionsEvery decision backed by specific clause references
JSON output format for seamless system integration
Domain-Specialized ProcessingBuilt specifically for insurance policies, not generic documents
Extracts structured data (age, gender, procedures, coverage options)
Handles insurance-specific logic and calculations
Scalable Graph KnowledgeAs more policies are processed, the graph becomes smarter
Learns policy patterns and clause relationships over time
Network effect: better performance with more data
Enterprise-Ready OutputStandardized decision format for easy integration
Compliance-ready audit trails
Consistent decision-making across all policy types

List of features offered by the solution

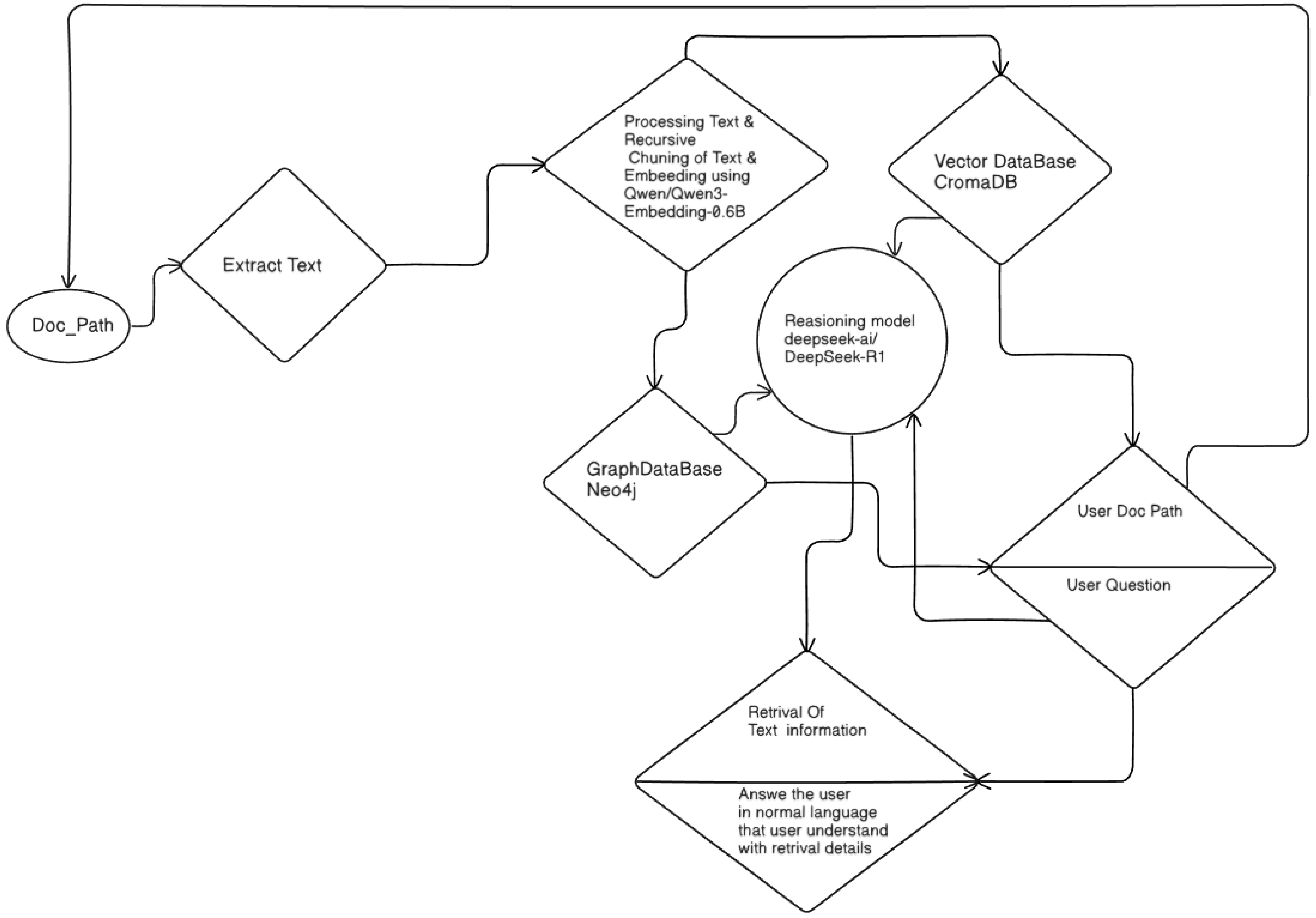
- Automated PDF processing: text extraction and intelligent chunking
- Natural language queries: understands plain English policy questions
- Hybrid search: combines semantic and graph-based retrieval
- Clear explanations: translates legal jargon into simple language
- Structured decisions: JSON output with justification and clause references
- Scalable and fast: handles multiple documents with real-time responses
- Transparent results: no AI hallucination; sources traced to exact clauses
- Easy integration: API-driven with optional web interface

Process flow diagram or Use-case diagram

Add a flow diagram or a use case diagram or an architecture diagram.



Architecture diagram of the proposed solution


Doc_Path
Extract Text
Processing Text & Recursive Chuning of Text & Embeeding using Qwen/Qwen3-Embedding-0.6B
Vector DataBase CromaDB
Reasioning model deepseek-ai/ DeepSeek-R1
GraphDataBase Neo4j
User Doc Path
User Question
Retrival Of Text information
Answe the user in normal language that user understand with retrival details

Technologies to be used in the solution:

- This tech stack supports AI and NLP app development, with **Qwen Embeddings** and **DeepSeek/Phi-3** via **Hugging Face**, using **PyTorch**. **ChromaDB** handles vectors, and **Neo4j Aura** manages graph databases.
- Text processing uses **PyPDF** for PDFs and **LangChain** for chunking. The backend employs **Python 3.10+ with FastAPI** . **Streamlit** is used for frontend interfaces.
- Deployment is streamlined with **Hugging Face Hub, Neo4j Aura**, and **AWS** or **Azure** for cloud infrastructure, enabling sophisticated AI applications.

Implementation Plan & Next Steps

Phase 1: Prototype Refinement

- Enhance legal terminology processing
- Improve JSON output consistency

Phase 2: MVP

- Build a lightweight backend with Flask/FastAPI
- Develop a simple Streamlit frontend
- Enable multi-document ingestion and query tracking

Phase 3: Deployment Ready

- Use Docker for containerization
- Deploy on AWS/Azure with ChromaDB and Neo4j Aura
- Add user authentication and session management

Phase 4: Scaling & Enhancements

- Support more document types (e.g., claims forms, medical reports)
- Add voice input and multi-language support
- Develop a mobile-responsive interface

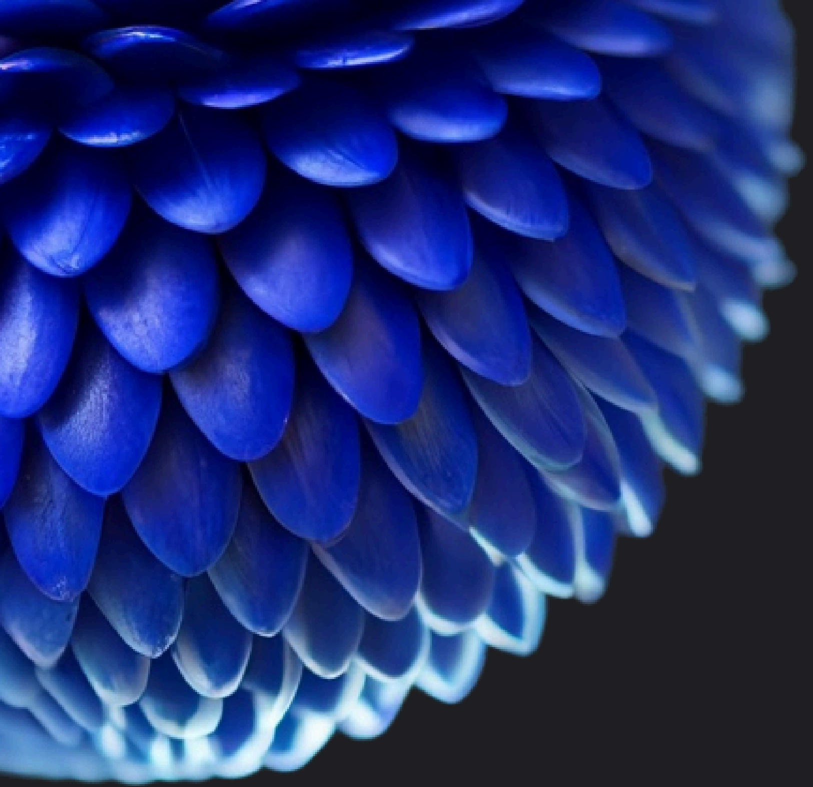
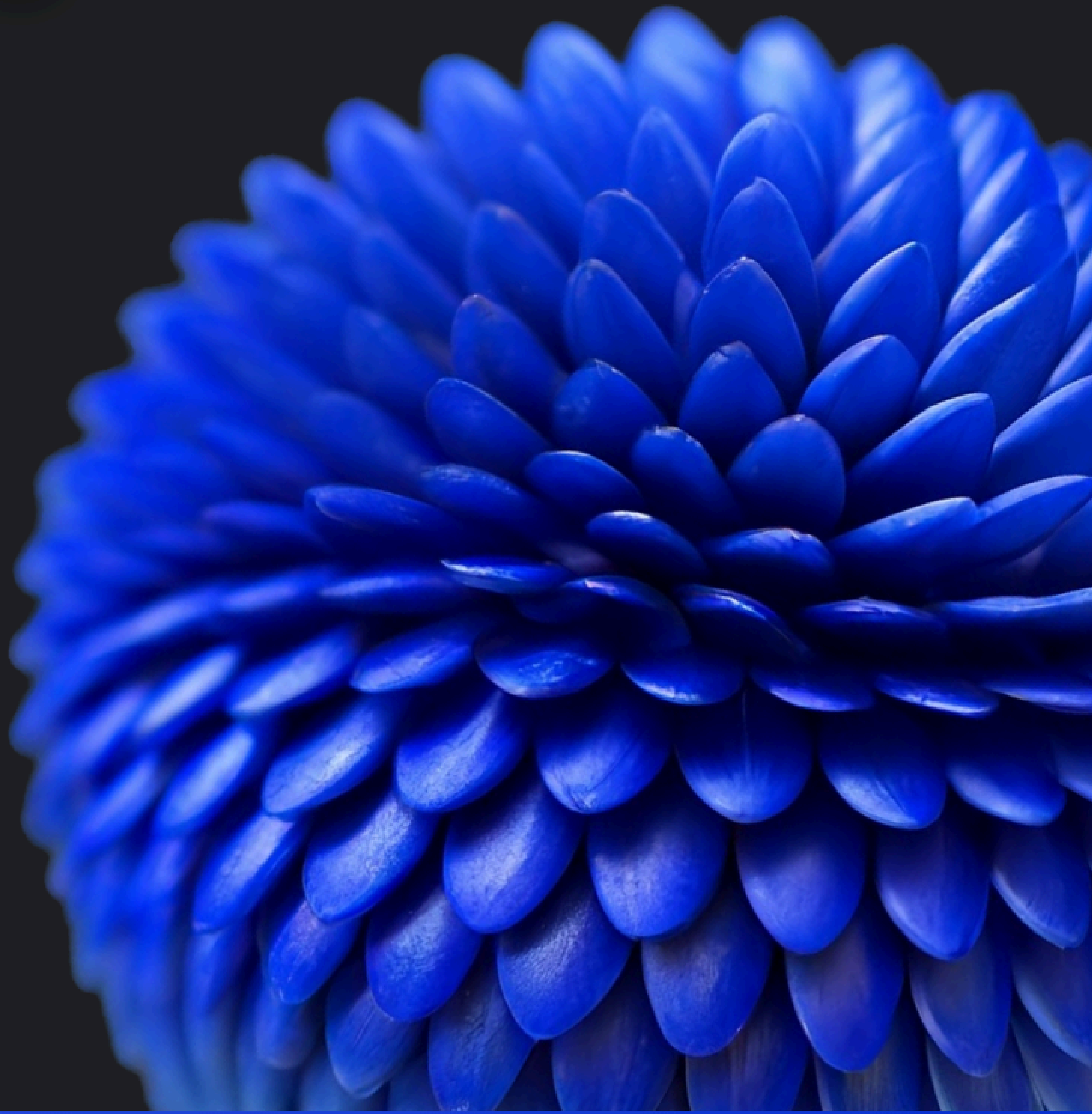
Stretch Goals

- Refine the language model with specific datasets
- Add real-time collaborative annotation

Offer downloadable compliance reports

Google Cloud H2S

Gen AI Exchange Hackathon

Thank you