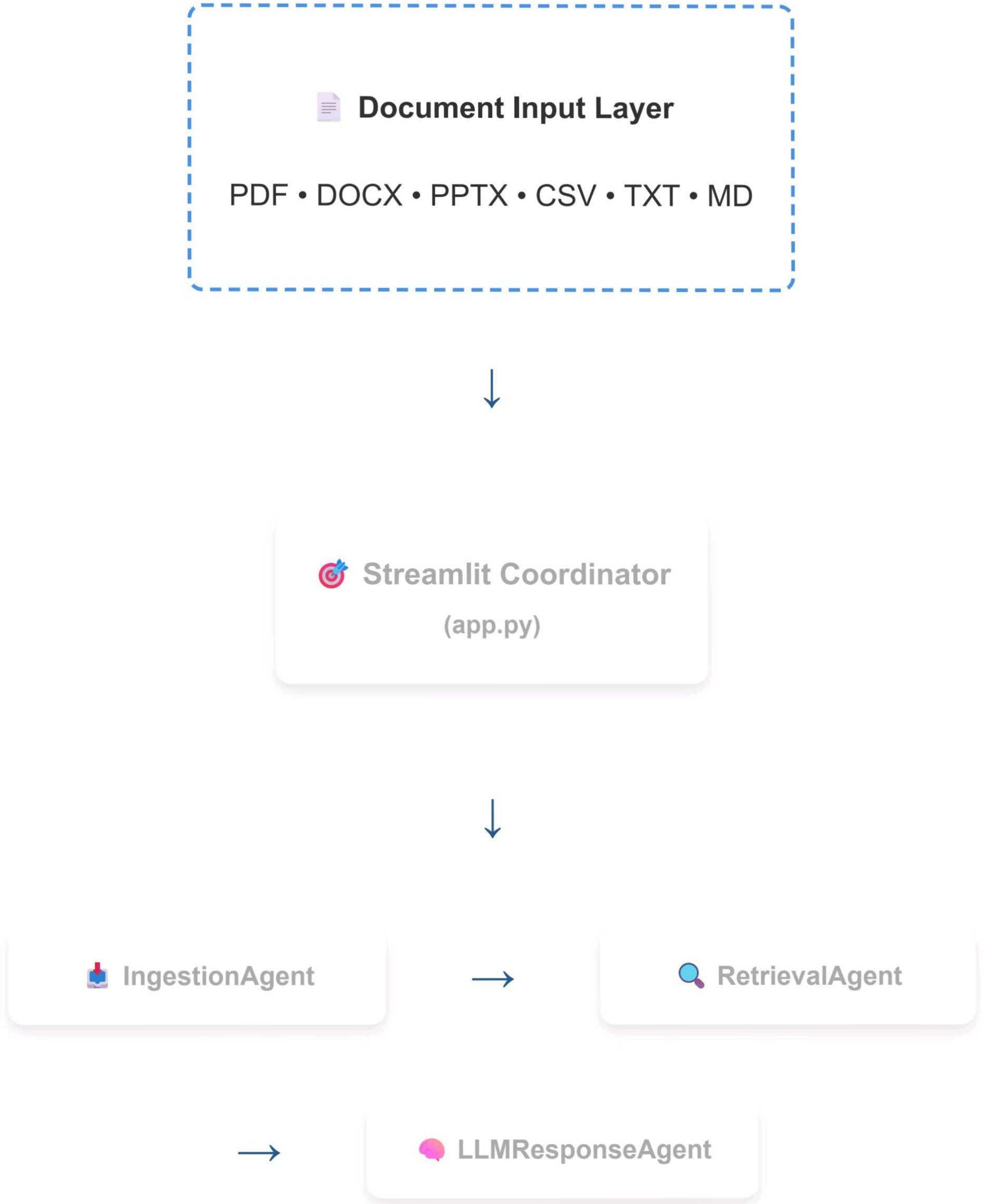
# Multi-Format Document QA using Model Context Protocol (MCP)

## High-Level System Overview



**Document Input Layer**

PDF • DOCX • PPTX • CSV • TXT • MD

↓

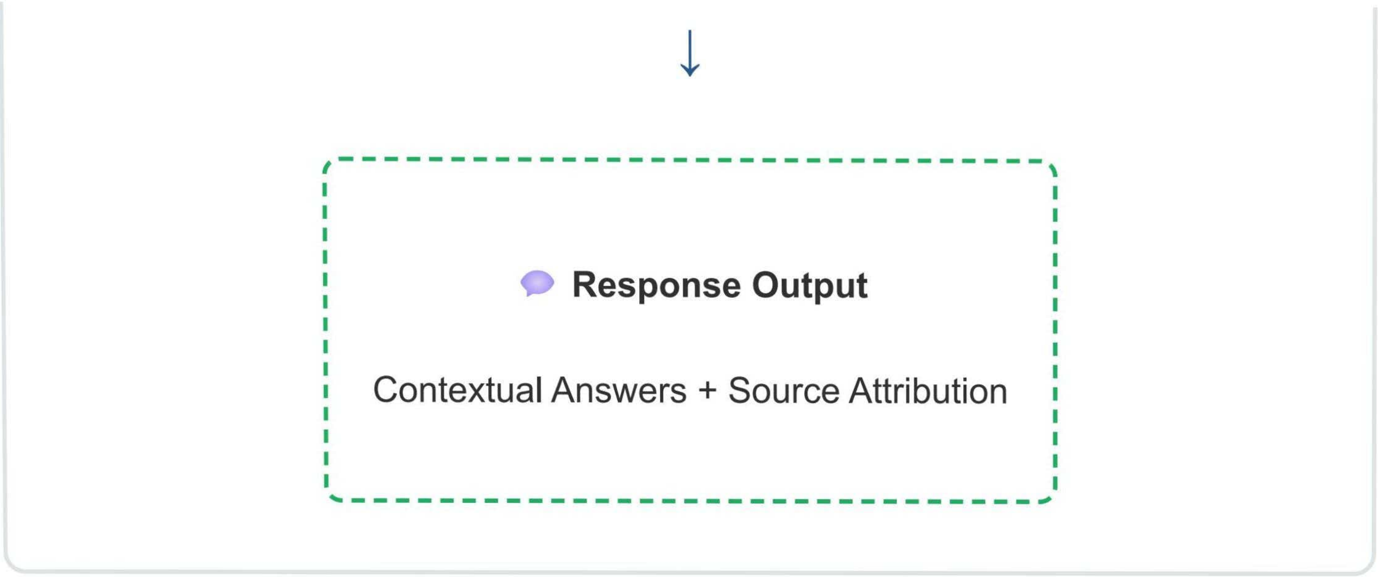
**Streamlit Coordinator**

(app.py)

↓

**IngestionAgent** → **RetrievalAgent**

→ **LLMResponseAgent**



💬 **Response Output**

Contextual Answers + Source Attribution

### 🖥️ Frontend Framework

Streamlit UI

### 🤖 Large Language Model

Google Gemini 1.5 Flash

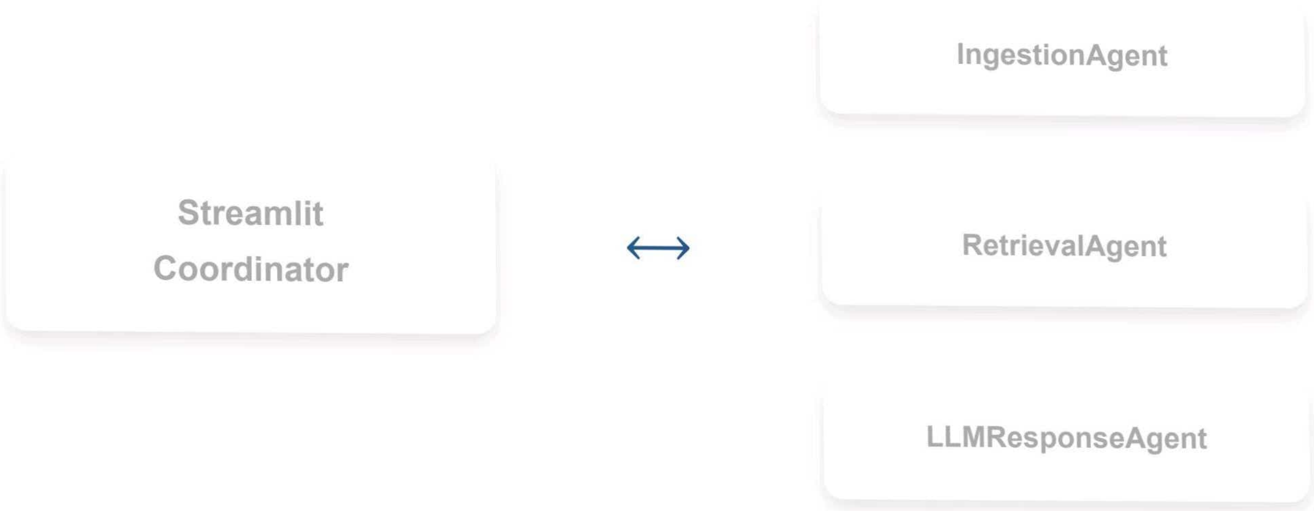
### 🔗 Embedding Model

all-MiniLM-L6-v2

### 📊 Vector Database

FAISS (CPU Version)

## Model Context Protocol (MCP) Communication Flow



Streamlit Coordinator

⟷

IngestionAgent

RetrievalAgent

LLMResponseAgent

*Standardized JSON message passing enables loose coupling and scalable architecture*

# Data Flow & Processing Pipeline

## Document Processing Workflow

### 1. Document Upload

User uploads files via Streamlit sidebar

Supported: PDF, DOCX, PPTX, CSV, TXT, MD

### 2. Format Detection & Parsing

IngestionAgent identifies file types and applies appropriate parsers

Extract text content while preserving semantic structure

### 3. Text Chunking

Split documents into overlapping chunks

Optimize for context retention and retrieval precision

### 4. Vector Embedding

RetrievalAgent generates embeddings using Sentence-Transformers

384-dimensional vectors for semantic representation

### 5. Index Creation

FAISS index construction for efficient similarity search

CPU-optimized flat index for accurate retrieval

↓

## Query Processing Flow

**User Query**

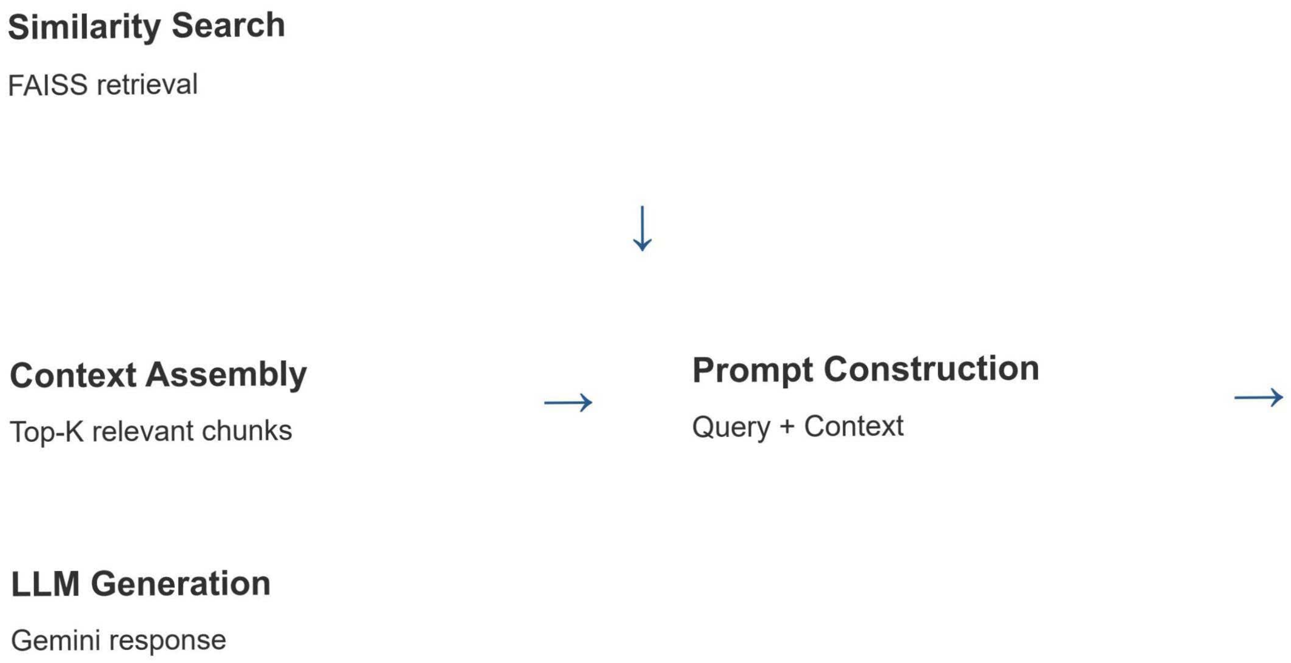
Natural language question

→

**Query Embedding**

Vector representation

→



**Similarity Search**

FAISS retrieval

↓

**Context Assembly**

Top-K relevant chunks

→

**Prompt Construction**

Query + Context

→

**LLM Generation**

Gemini response