

Product Design Document (PDD) – Agentic AI Demo

1. Product Overview

The Agentic AI Demo is a multi-agent orchestration system built using LangGraph, LangChain, FastAPI, PostgreSQL, Gmail MCP, and Ollama (qwen3-vl:8b). It demonstrates agent collaboration, RAG capabilities, database querying, email retrieval, and tool-augmented reasoning.

2. Objectives

- Provide a functional demo of LLM-powered multi-agent workflows. - Enable interaction with structured, semi-structured, and unstructured data sources. - Demonstrate RAG over PDFs using pgvector. - Showcase integration with Gmail MCP for email retrieval. - Provide clean architecture, logging, and API structure.

3. Key Features

1. Five AI Agents: - Router Agent - Weather Agent - PDF RAG Agent - PostgreSQL Agent - Gmail MCP Agent 2. Supports: - Weather queries - PDF question answering via RAG - SQL database queries - Gmail email extraction - General LLM chat 3. Uses Ollama (qwen3-vl:8b) for inference.

4. System Architecture

The system is centered around a LangGraph multi-agent pipeline orchestrated by a Router Agent. FastAPI exposes the main endpoint, forwarding messages into the agent graph. High-level Flow: User → FastAPI → Router Agent → Specialized Agent → Tools / LLM → Response

5. Functional Requirements

FR1: The system must classify user input and route it to the correct agent. FR2: Users must be able to request weather information. FR3: Users must be able to upload PDFs and query them via RAG. FR4: Users must retrieve data from PostgreSQL securely. FR5: Gmail MCP must allow accessing inbox email data. FR6: All responses must be generated via Ollama. FR7: System should provide extensive logs for debugging and analysis.

6. Non Functional Requirements

NFR1: System must be modular and maintainable. NFR2: API latency should remain under acceptable limits for LLM tasks. NFR3: Logs must be structured and centralized. NFR4: System must support scalability for additional agents. NFR5: Embedding and vector search must remain performant.

7. Technical Stack

Backend: FastAPI Orchestration: LangGraph LLM: Ollama (qwen3-vl:8b) Database: PostgreSQL + pgvector RAG: LangChain vectorstore + retriever Email Integration: Gmail MCP UI (optional): Chainlit

8. Folder Structure

app/ graph/ services/ utils/ data/ readme.md requirements.txt

9. API Endpoints

POST /ask POST /pdf/upload GET /weather

10. Logging Strategy

- Router decisions logged - Agent execution times logged - SQL queries logged safely - RAG retrieval chunks logged - Gmail MCP calls logged

11. Success Criteria

- Users can successfully interact with all five agents. - System correctly routes requests. - RAG answers match PDF content. - SQL responses return accurate data. - Gmail retrieval works consistently. - System documentation is complete.

12. Conclusion

This PDD defines the required behavior, architecture, and design standards for building the Agentic AI Demo. It serves as the foundation for implementation and future expansion.