Phases & Deliverables (Sequential, Actionable)

Phase 0 — Discovery & Scope Lock (Deliverable: Project Spec)

- Define core vs. nice-to-have features (must have: multi-file upload, RAG Q&A;, conversation threads, embeddings; nice: streaming answers, analytics, OCR).

- List supported file types (PDF, DOCX, TXT, possibly images).

- Identify constraints: hosting (cloud/local), budget for vector DB, privacy requirements.

Phase 1 — Architecture & Tech Decisions (Deliverable: Architecture Diagram + Tech Stack Doc)

- System components: Next.js frontend, FastAPI backend, LangGraph + LangChain for orchestration, MCP for worker management, MySQL, Vector DB.

- Choose vector DB: Milvus, Weaviate, Pinecone, Qdrant Cloud.

- Establish authentication: JWT + refresh tokens.

- Data flow: upload → worker → preprocess → embeddings → vector DB → MySQL metadata.

Phase 2 — Data Model & API Contract (Deliverable: DB Schema + OpenAPI)

- Schema: users, teams, threads, messages, documents, chunks, embedding metadata, jobs.

- API endpoints: auth, thread creation, send message, streaming responses, doc upload, listing, re-indexing.

- Websocket/SSE for streaming responses and progress.

Phase 3 — Ingestion & RAG Pipeline (Deliverable: Ingestion Service + Worker)

- Worker handles ingestion outside main API.

- Steps: validation → text extraction → cleaning → chunking → embeddings → store vectors → update MySQL → notify frontend.

- Idempotent ingestion using hash checks.

Phase 4 — Conversational RAG Flow (Deliverable: RAG QA Endpoint)

- Pipeline: query → retrieve → rerank → generate → stream tokens.

- Store assistant message + citations.

- Include source attribution metadata.

Phase 5 — Frontend UX & Polish (Deliverable: Next.js App)

- Chat UI: messages, streaming, attachments, provenance toggle.

- Upload UI: drag-and-drop, multi-select, progress, validation.

- Admin/debug: job logs, vector stats, doc list.

- Responsive design + subtle animations.

Phase 6 — Security, Tests & Reliability (Deliverable: Tests + Security Checklist)

- Unit, integration, end-to-end tests.

- Hardening: input limits, file caps, rate limiting, RBAC.

- Logs + metrics (latency, QPS, failures).

Phase 7 — CI/CD + Deployment (Deliverable: Deployable Stack + Runbook)

- Containerized backend & worker.

- CI pipeline: lint, tests, build, push.

- Deployment: secrets, migrations, backup strategy, vector index rebuild.

Concrete Task Breakdown (Developer Checklist)

- Monorepo structure setup.

- Conventions: lint, formatting, hooks.

- MySQL schema + Alembic migrations.

- Auth service (JWT).

- File upload + storage.

- Worker with queue system.

- Embeddings + vector DB integration.

- RAG endpoint with LangGraph.

- Next.js chat UI skeleton.

- Websocket streaming.

- Tests + CI.

- Logging + monitoring + docs.

Tech Decisions & Recommendations

- Use separate ingestion worker.

- Store canonical data in MySQL; vectors in vector DB only.

- Use robust chunking (500–1000 tokens).

- Idempotent ingestion using file checksum.

- Implement streaming early for strong UX.

- Track provenance for explainability.

- Ensure local dev parity with optional in-memory vector DB.

Minimal Viable Feature Set (MVP)

- User auth + thread creation.

- Multi-upload + ingestion worker.

- Basic RAG search + single-turn Q&A.;

- Chat UI with streaming + document list.

- Source snippet display.

Nice-to-Have Features

- Streaming ingestion progress.

- OCR for images.

- Multi-language.

- User quotas.

- Advanced retriever (reranking).

- Analytics dashboard.