RAG Demo - Main Components Overview

Backend

Vector store (Qdrant)

- **QdrantVectorStoreService**: Integrates with Qdrant for vector storage and similarity search. Handles collection initialization, upserts, searches, and deletes.

Embeddings (ONNX)

- **LocalEmbeddingService**: Generates embeddings locally using ONNX Runtime with an all-MiniLM-L6-v2 compatible model (384 dimensions). If the model or vocab is missing, it falls back to a lightweight hash/TF-IDF style embedding.

Document ingestion

- **DocumentService**: Extracts text from PDFs, chunks content, generates embeddings, and stores them in Qdrant. Also scrapes websites (static HTML or headless browser for SPAs), chunks content, embeds, and stores results.

Chat/RAG pipeline

- **ChatService**: Orchestrates the RAG workflow. Detects conversational messages, retrieves relevant chunks, and generates answers. Uses GitHub Models (gpt-4o-mini) when enabled; otherwise returns a mock response based on retrieved context.

API layer

- **ChatController**: REST endpoints for asking questions, uploading PDFs, ingesting URLs, deleting documents, and stats/health checks.

Core models

- **DocumentChunk**: Represents a chunk of text with metadata (source, index, type, timestamps, language).
- **ChatRequest / ChatResponse**: Request/response contracts for chat.
- **SourceType**: Enum for source classification (PDF, Website, Text, Unknown).

Startup & configuration

- **Program.cs**: Registers services, middleware (CORS, rate limiting, logging, compression), and initializes Qdrant on startup.
- **appsettings.json**: Configures Qdrant, embedding/search settings, GitHub Models integration, and rate limits.

Frontend (Angular)

Chat UI

- **Chat component**: User-facing chat interface. Sends questions, renders responses with basic markdown formatting, and shows loading states.

Admin UI

- **Admin component**: Admin panel for document ingestion (PDF upload and URL scraping), including options for SPA route handling.

Document upload

- **DocumentUpload component**: Handles PDF selection, validation, upload progress, and success/error notifications.

API integration

- **ChatService (Angular)**: HTTP client for backend endpoints: ask, upload, ingest URL, stats, and health.

Routing

- **Routes**: `/chat` and `/admin` with a default redirect.