**Application Architecture (High-Level)**

**1. Frontend Layer**

* **HTML/Jinja2 templates** rendered directly by FastAPI.
* Displays search results, knowledge graph queries, and document information in a user-friendly interface.
* User submits queries via the web interface.

**2. Backend Layer (FastAPI Application)**

* **API Endpoints** (Python) handle requests from the frontend.
* Query logic connects to the databases:
  + **Neo4j** → for graph-based queries (stakeholders, actions, categories, relationships).
  + **MongoDB** → for document storage, metadata, and preprocessed text.
* Integrates any preprocessing, entity normalization, or summarization if required on-the-fly.

**3. Databases**

* **Neo4j Graph Database**
  + Stores entities (stakeholders, policies, topics, actions) and their relationships.
  + Supports graph queries for connections and visualizations.
* **MongoDB Document Store**
  + Stores structured and semi-structured data (e.g., transcripts, action plans, extracted entities).
  + Provides quick access to raw and processed text.

**5. User Interaction Workflow**

1. User opens the web app in browser.
2. Sends query → FastAPI processes request.
3. FastAPI queries **Neo4j** (relationships) and **MongoDB** (documents).
4. Results are rendered in **HTML templates** (tables, graphs, text).
5. User sees interactive outputs (search results, knowledge graph relationships, summaries).