Policy Navigator Agent

A Multi-Agent RAG System for Government Regulation Search

Objective

Build an Agentic RAG system that allows users to query and extract insights from complex government regulations, compliance policies, or public health guidelines.

Agent's Skills:

Some examples of what the agent should be capable to do:

1. Check the latest policy status via external government APIs

    User:

    "Is Executive Order 14067 still in effect or has it been repealed?"

    Agent:

    "I checked the Federal Register API—Executive Order 14067 is still active as of May 2025. No amendments or repeals have been filed."

    API used: Federal Register API

2. Retrieve case law summaries linked to specific regulations

    User:

    "Has Section 230 ever been challenged in court? What was the outcome?"

    Agent:

    "Yes. I found multiple court rulings referencing Section 230. For example, Fair Housing Council v. Roommates.com clarified limits on platform immunity. Would you like a summary or the full case history?"

    API used: CourtListener API (via the Free Law Project)

How should the agent work?

Users should be able to upload a set of policy documents or specify a public URL (e.g., a government or regulatory site) from which the agent will extract and index information.

Users can ask questions in plain language, such as "When does this policy take effect?" or "What are the compliance requirements for small businesses?"

The agent should process the question, retrieve the most relevant information from the indexed content, and return a clear, structured answer—ideally with source references or section highlights.

The agent should connect with at least one external tool (e.g., Slack, Vercel, Calendar API, Notion) so that users can receive updates, schedule reminders, or take next steps based on the agent's output.

Technical Scope

Your solution should include the following components:

1. **RAG Pipeline (Agentic Version)**: For the design of your agent you can choose to use a single agent architecture or a team agent.
2. **Data Ingestion**: You must provide your agents with knowledge from at least two data sources: **a dataset and a website**. Examples:
    a. Upload a dataset of policy/regulation documents. You can use:
        i. Kaggle Datasets (search for "privacy law", "compliance", "climate policy", etc.)
        ii. UCI Machine Learning Repository
        iii. Data.gov
        iv. Scrape from a public source like the EPA website or WHO health guidelines

    Demonstrate how to create a vector index (unstructured data)

    Ground agent responses using a vector index
3. **Tool Integration**: You must utilise three types of tools (see list below) to help your agent retrieve information, process information and/or perform an action via a third-party API. Examples:
    a. Add a marketplace tool to the agent
    b. Add a custom Python tool
    c. Add a pre-promoted LLM as a tool
    d. Add a SQL or CSV tool (structured data)
    e. Add a pipeline as a tool
    f. Add code interpreter as a tool
    g. Configure tool parameters and description
4. UI or CLI (optional, but encouraged): **You must demonstrate integrating your final agent into an external application**. Examples:
    a. Create a minimal interface (could be CLI, web-based, or chatbot UI) to interact with the agent

Submission Requirements

1. GitHub Repository:
    a. Make your project public.
    b. Include a well-documented README.md that explains:
    c. What your agent does
    d. How to set it up

    e. Dataset/source links
    f. Tool integration steps
    g. Example inputs/outputs

   NOTE: Here is a guide to create a GitHub repository

2. Demo Video (2–3 minutes)
    a. Walk us through what your agent does, the workflow, and a short live demo.
3. Future Improvements Section in README
    a. Suggest enhancements like:
        i. Adding more agents (e.g., summarization or analytics)
        ii. UI improvements
        iii. Additional data integrations
        iv. Caching or memory features
4. Timeline
    a. You have **1 week** to complete the project.
5. Other Enhancements (if time permits)
    a. Must integrate vector storage
    b. Add error handling and logs
    c. Support multilingual policy documents
6. **Submit your completed GitHub repo with all files to devrel@aixplain.com**

What You'll Learn

1. How to design and structure a multi-agent RAG workflow
2. Working with unstructured policy data
3. Integrating custom tools using the aiXplain SDK
4. Deploying practical, real-world AI agents with explainable components