

# Backend Specification: DGS Research & Crawling Engine

## Overview

Implement a dual-phase search and synthesis system using FastAPI. The system must support intelligent routing, source discovery, and deep content extraction via web crawling.

## 1. Environment & Dependencies

- **Search:** Tavily (Primary) or DuckDuckGo-Search (Fallback/Free).
- **Crawling:** Crawl4AI (Python library) for HTML-to-Markdown conversion.
- **LLM:** Gemini-2.5-Flash (Intent Routing) and Gemini-1.5-Pro (Long-context Synthesis).
- **Storage:** Use Firestore to log search\_id and selected sources for audit trails.

## 2. API Endpoints

### POST /v1/research/discover

**Description:** Performs the initial web search and returns candidate URLs.

- **Input:** query: str, context: str (optional)
- **Logic:**
  1. **Router:** LLM identifies if the query is General, Academic, Security, or News.
  2. **Search Engine:** Call Tavily API with search\_depth="basic".
  3. **Domain Filtering:**
    - Security: Add site:github.com OR site:nvd.nist.gov.
    - Academic: Add site:\*.edu OR site:arxiv.org.
- **Output:** Returns a list of CandidateSource objects including title, url, and snippet.

### POST /v1/research/synthesize

**Description:** Deep crawls provided URLs and generates a cited report.

- **Input:** query: str, urls: List[str], user\_id: str
- **Logic:**
  1. **Crawl Phase:** Execute Crawl4AI concurrently for all URLs.
  2. **Context Assembly:** Concatenate Markdown outputs. Prefix each section with [Source #X: URL].
  3. **Prompting:**
    - **System Instruction:** "Synthesize an answer using ONLY the provided context. Use numeric citations [1]. Format the output in clean Markdown."
  4. **Safety Check:** Ensure no malicious scripts are extracted from crawled Markdown.
- **Output:** answer: str, sources: List[Dict]

### **3. Storage Paths (Rule 1 Compliance)**

- **Public Metadata:** /artifacts/{appId}/public/data/research\_logs
- **User Sessions:** /artifacts/{appId}/users/{userId}/search\_history

### **4. Error Handling**

- Implement exponential backoff (1s, 2s, 4s...) for Tavily and LLM API calls.
- If a URL fails to crawl, skip it and notify the user in the metadata rather than failing the whole request.