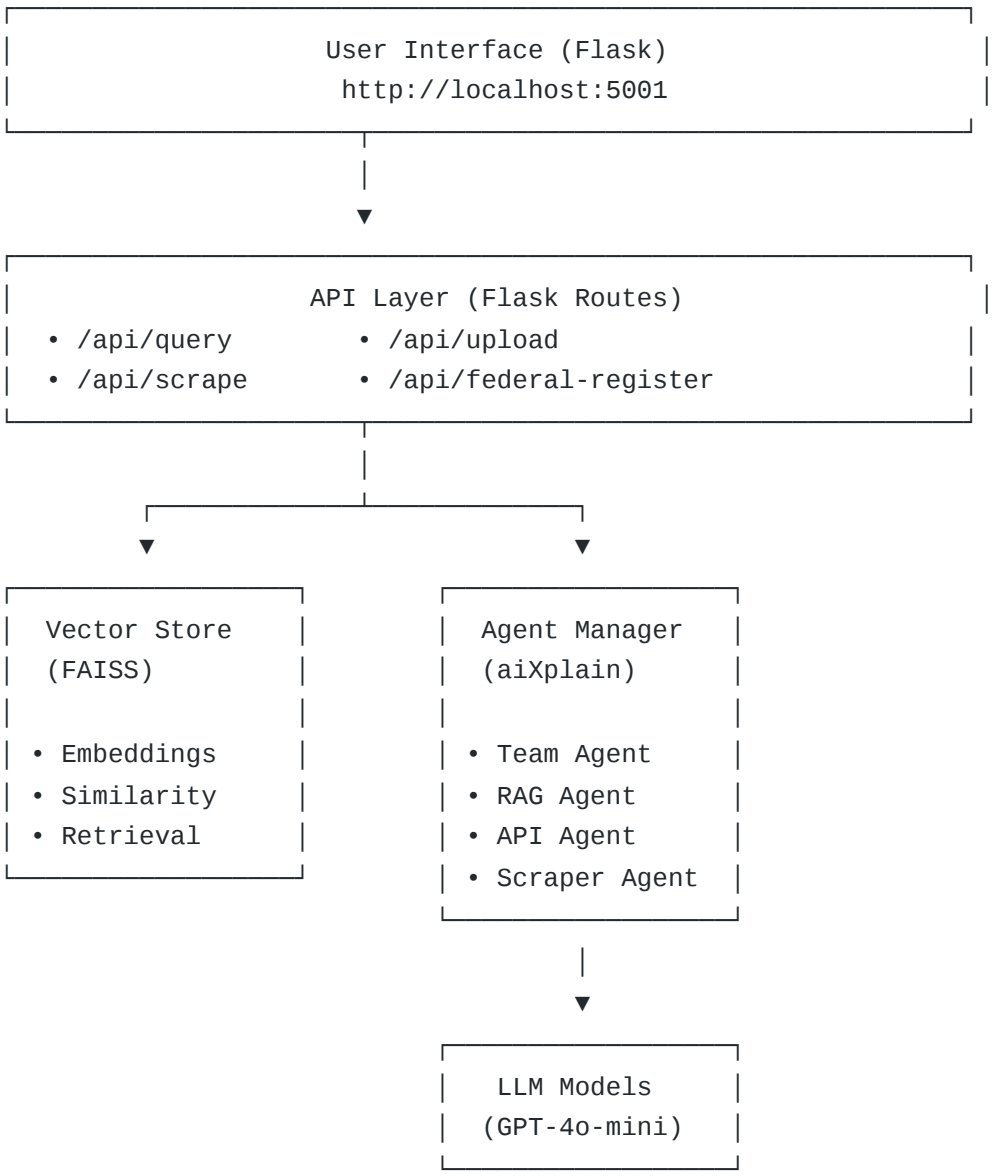
آلية عمل نظام Policy Navigator - دليل تقني شامل

نظرة عامة على النظام

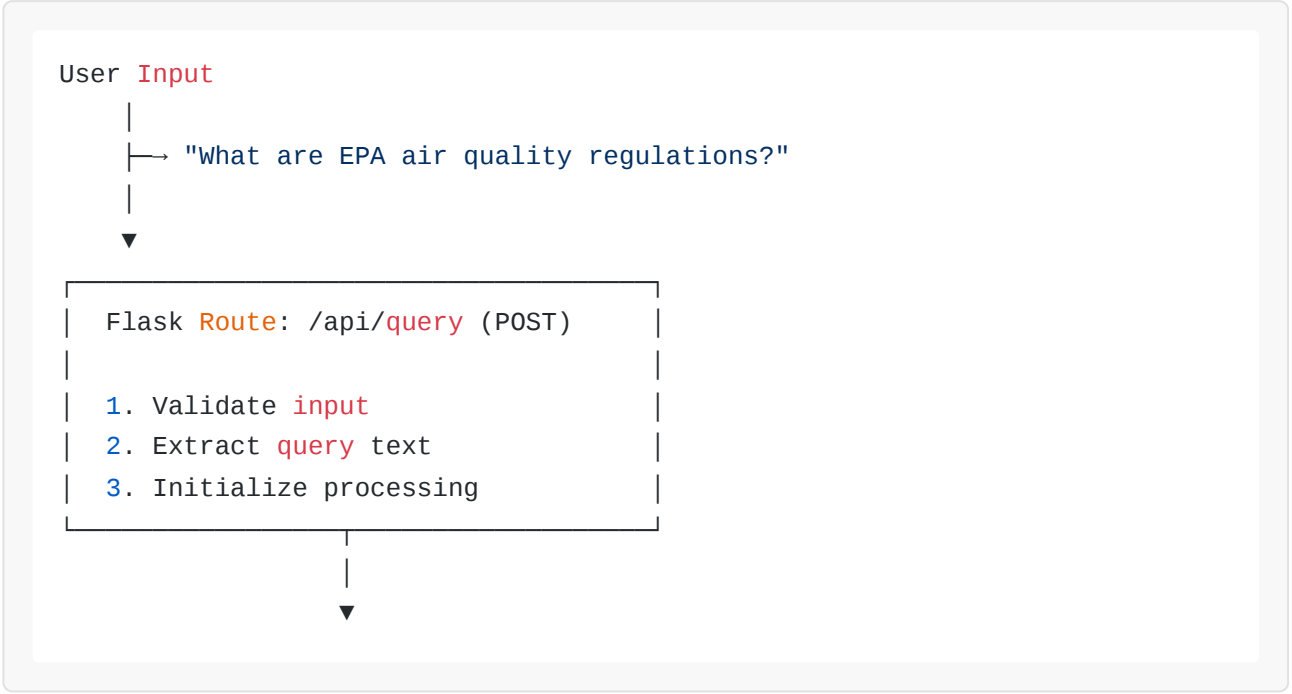
Policy Navigator هو نظام RAG متعدد الوكلاء (Multi-Agent RAG System) يستخدم منصة aiXplain لمعالجة الاستفسارات حول السياسات والتنظيمات الحكومية الأمريكية.

1. معمارية النظام (System Architecture)

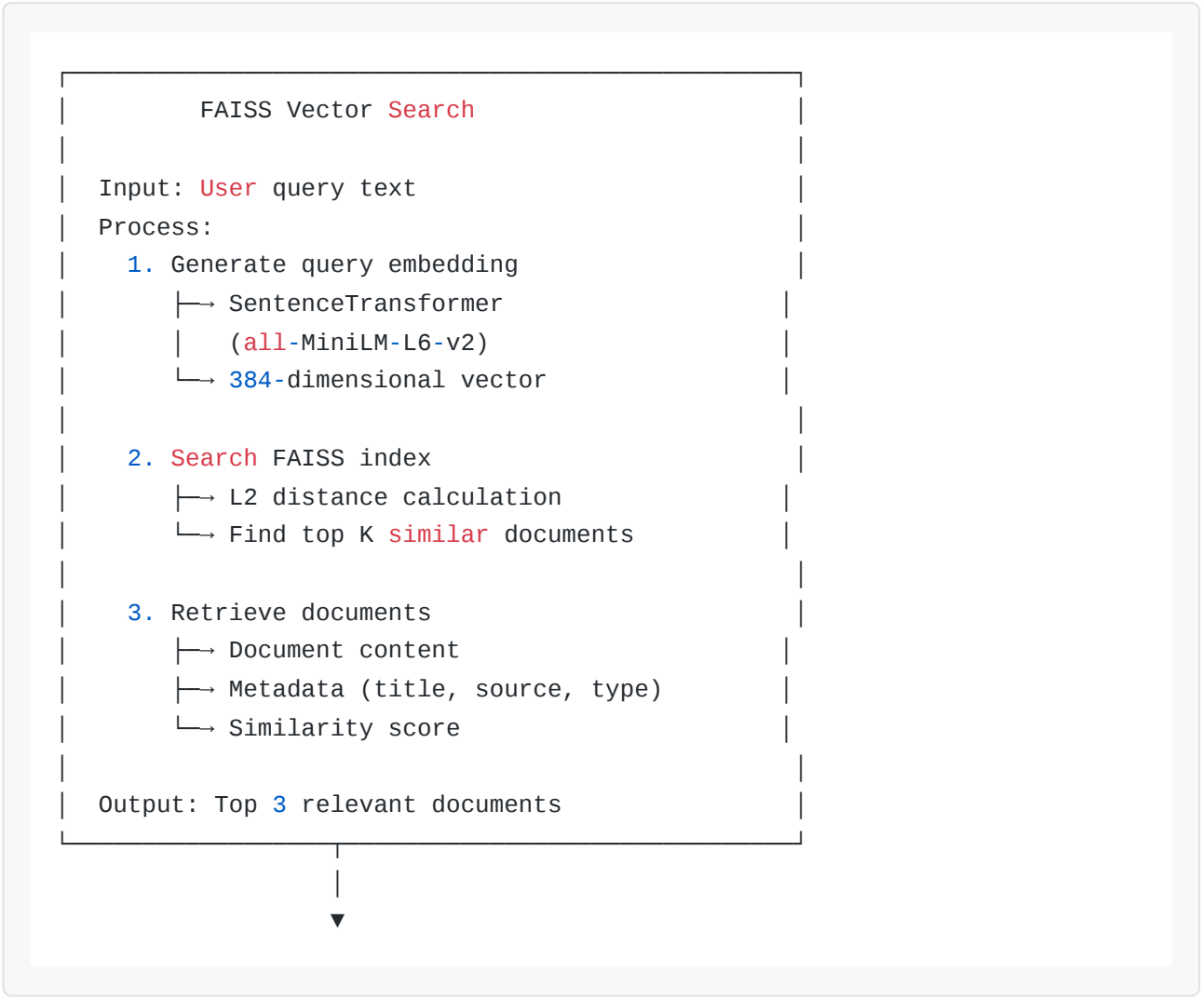


2. تدفق معالجة الاستفسار (Query Processing Flow)

المرحلة 1: استقبال الاستفسار



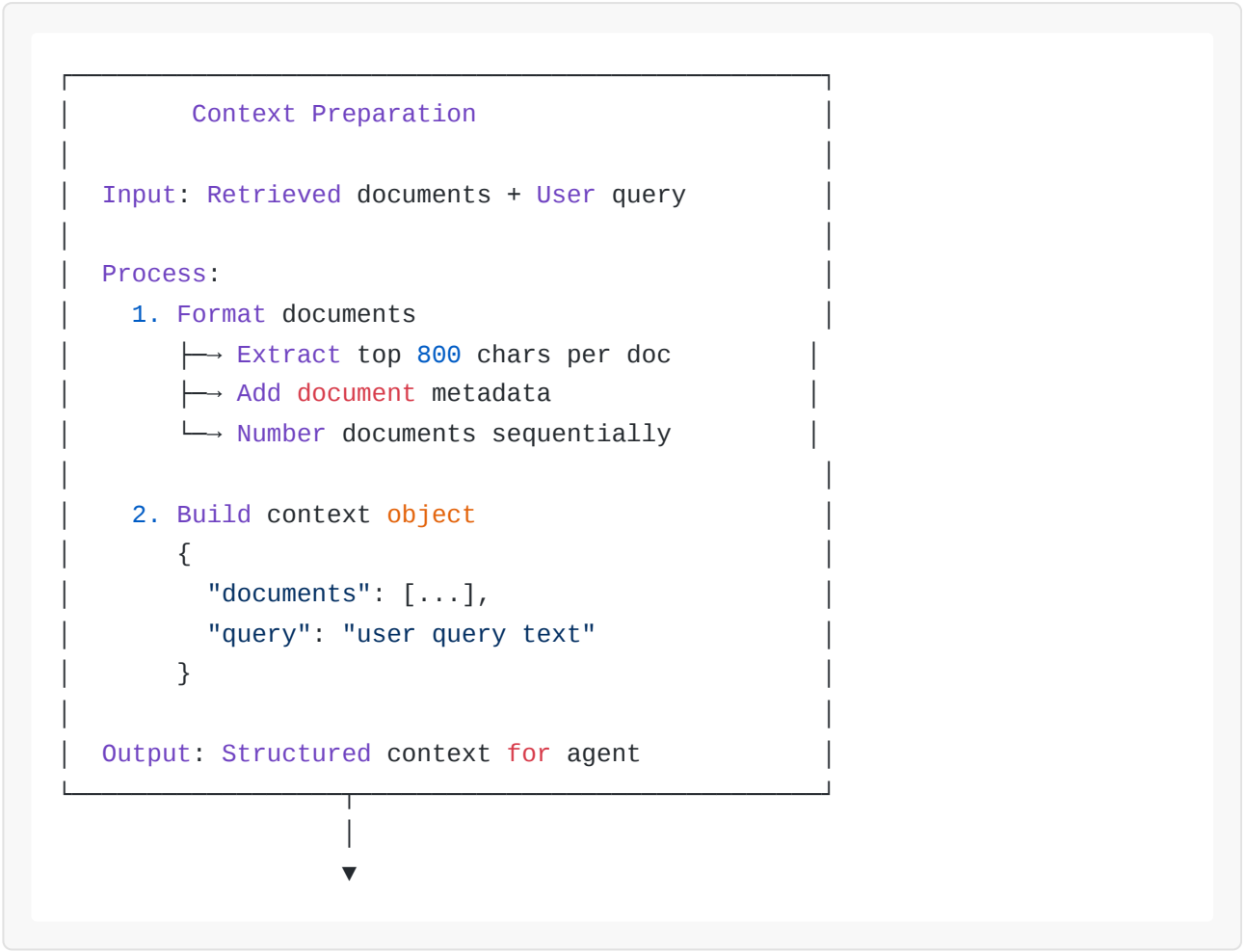
المرحلة 2: البحث في قاعدة البيانات الشعاعية



مثال على النتائج:

```
[
  {
    "content": "The EPA regulates air quality under Clean Air Act...",
    "metadata": {
      "title": "40 CFR § 50.4",
      "source": "CFR Title 40",
      "type": "regulation"
    },
    "score": 0.87
  },
  {
    "content": "National Ambient Air Quality Standards (NAAQS)...",
    "metadata": {
      "title": "40 CFR § 50.6",
      "source": "CFR Title 40",
      "type": "regulation"
    },
    "score": 0.82
  }
]
```

المرحلة 3: تحضير السياق للوكيل



مثال على السياق المُعد:

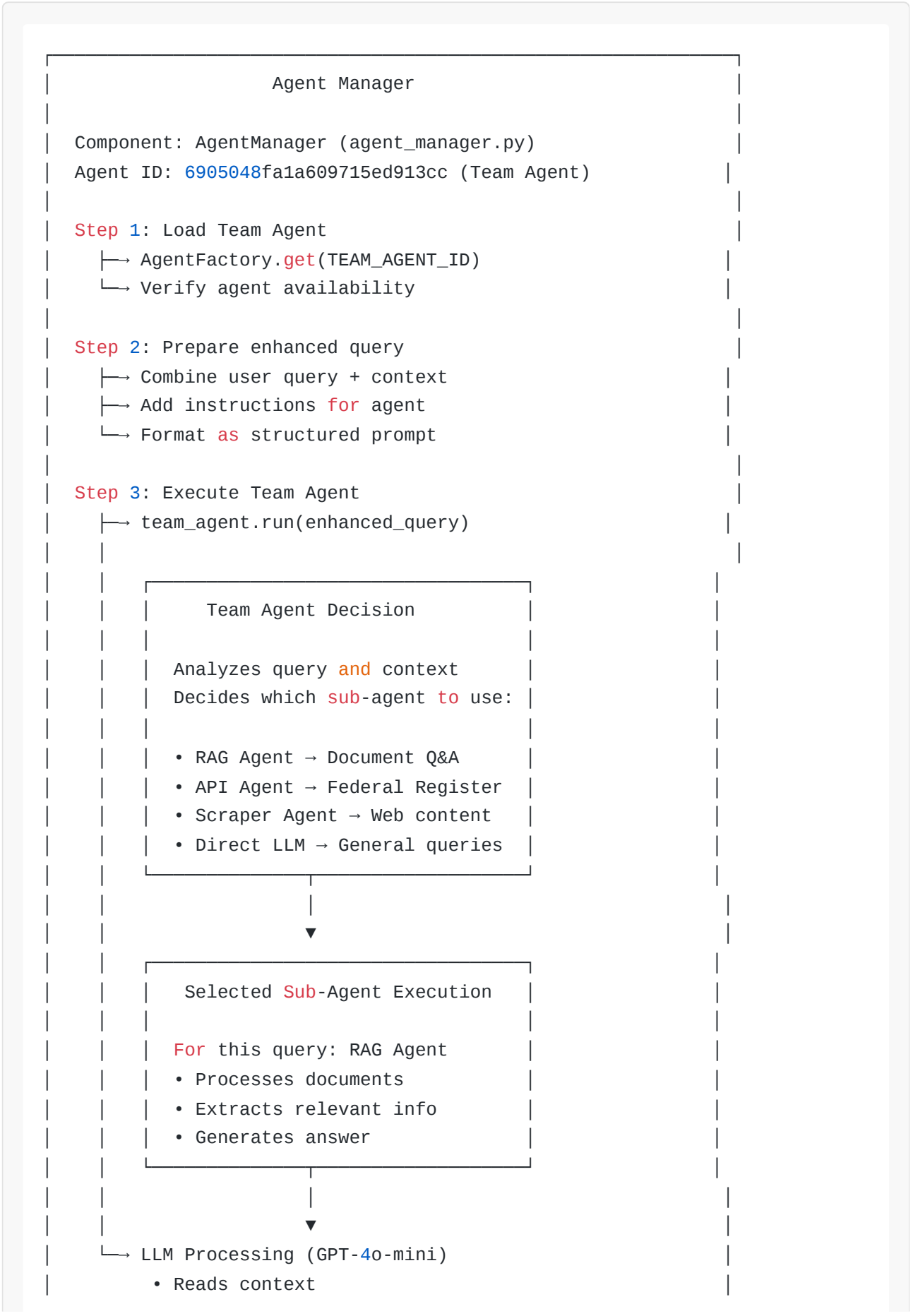
```
User Question: What are EPA air quality regulations?

Retrieved Policy Documents:

Document 1 (from 40 CFR § 50.4):
The EPA regulates air quality under the Clean Air Act. National
Ambient Air Quality Standards (NAAQS) are established for six
principal pollutants: carbon monoxide, lead, nitrogen dioxide,
ozone, particulate matter, and sulfur dioxide...

Document 2 (from 40 CFR § 50.6):
The national primary and secondary ambient air quality standards
for particulate matter are set forth in this section. The primary
standards are designed to protect public health...
```

المرحلة 4: معالجة الوكيل (Agent Processing)

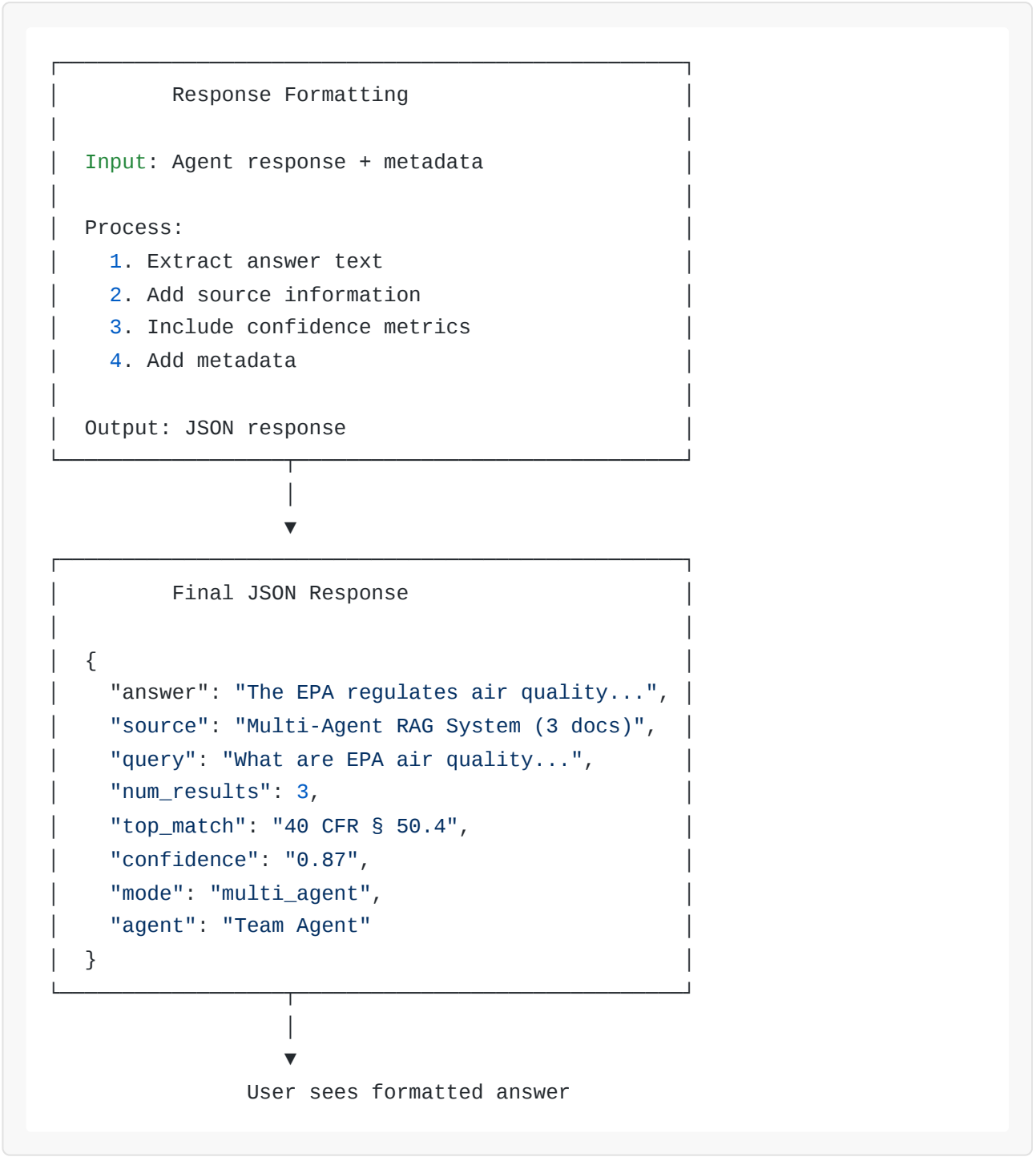


```
│          • Understands query                                    │
│          • Synthesizes answer                                   │
│          • Formats response                                     │
│                                                                 │
│  Output: Agent response object                                  │
└─────────────────────┬───────────────────────────────────────────┘
                      │
                      ▼
```

مثال على استجابة الوكيل:


```
{
  "success": true,
  "answer": "The EPA regulates air quality through the Clean Air Act, which establishes National Ambient Air Quality Standards (NAAQS) for six principal pollutants: carbon monoxide, lead, nitrogen dioxide, ozone, particulate matter, and sulfur dioxide. These standards are designed to protect public health and the environment. The regulations are codified in 40 CFR Part 50.",
  "agent": "Team Agent",
  "agent_id": "6905048fa1a609715ed913cc"
}
```

المرحلة 5: تنسيق الاستجابة النهائية



---

3. تدفق رفع الملفات (File Upload Flow)

```
    │   │
    │   └──→ Return single document
    │
    ▼
```

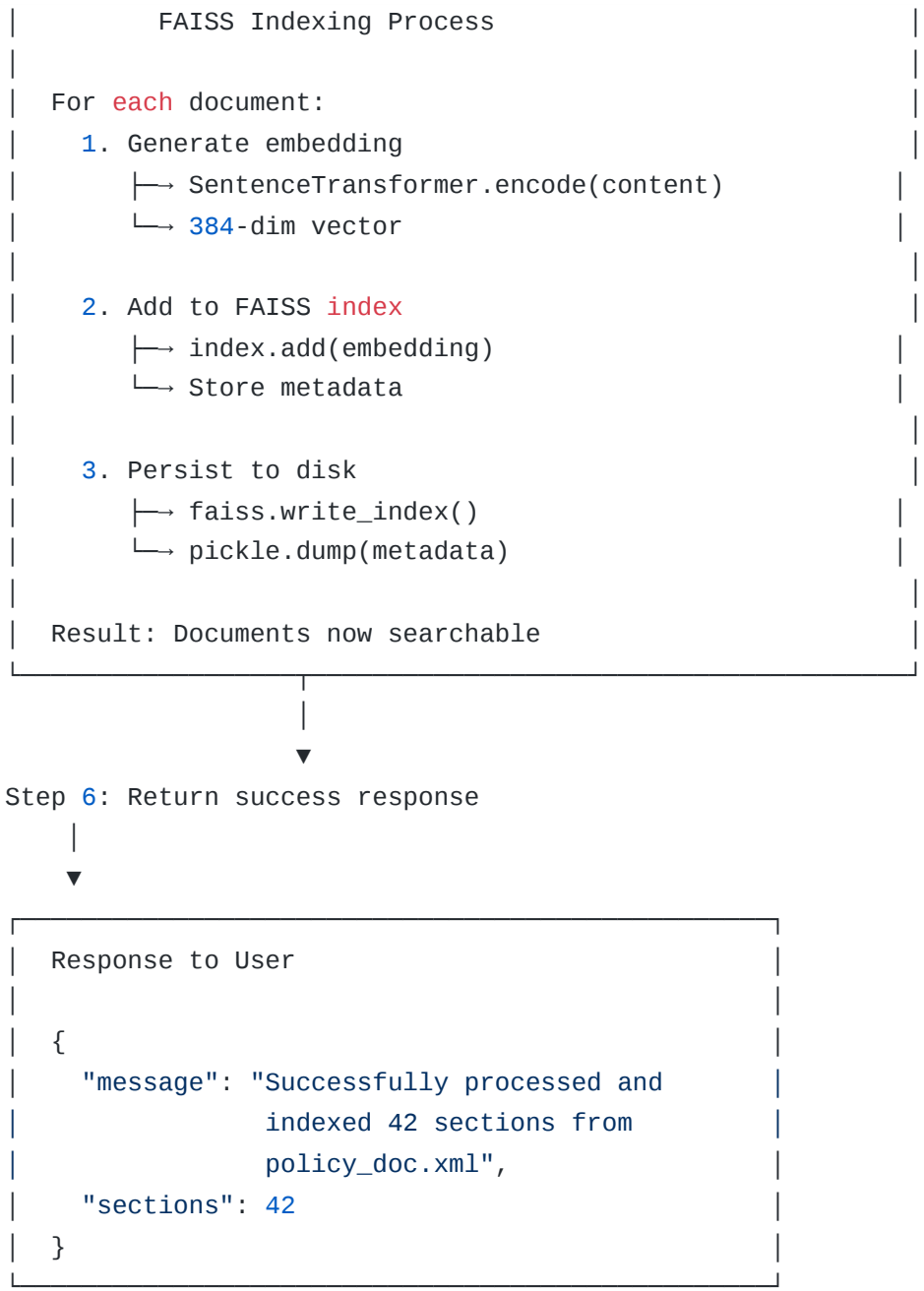
```
┌─────────────────────────────────────────────────────┐
│  Document Processing Output                         │
│                                                     │
│  [                                                  │
│    {                                                │
│      "content": "Section text...",                  │
│      "title": "Section 1.1",                        │
│      "section_number": "1.1"                        │
│    },                                               │
│    ...                                              │
│  ]                                                  │
└─────────────────────┬───────────────────────────────┘
                      │
                      ▼
```

```
Step 4: Prepare for indexing
    │
    ├──→ Create document objects
    │
    ▼
```

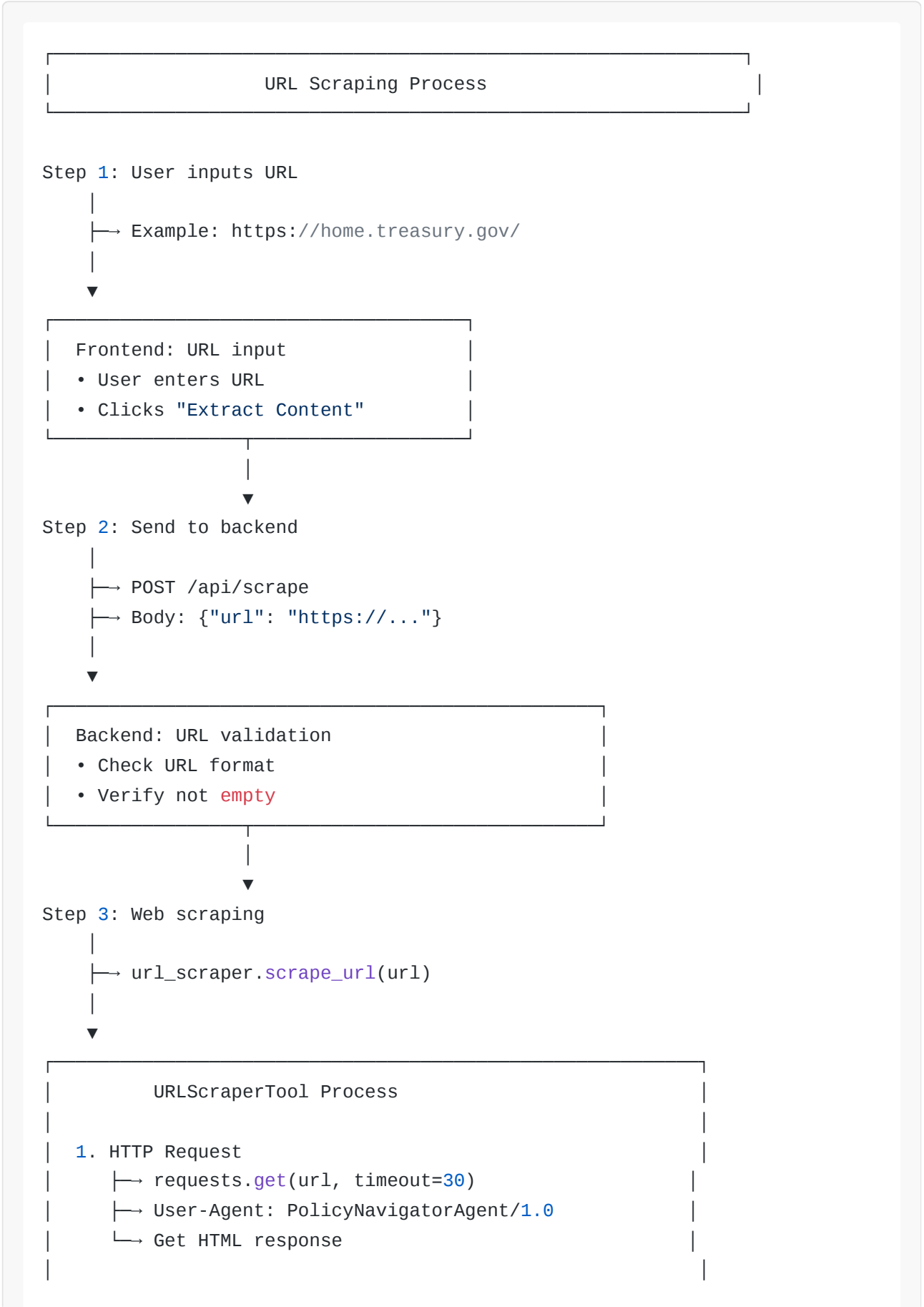
```
┌─────────────────────────────────────────────────────┐
│  Document Object Creation                           │
│                                                     │
│  documents = [                                      │
│    {                                                │
│      'content': section['content'],                 │
│      'metadata': {                                  │
│        'title': section['title'],                   │
│        'section_number': section['section_num'],    │
│        'source': filename,                          │
│        'type': 'uploaded'                           │
│      }                                              │
│    }                                                │
│  ]                                                  │
└─────────────────────┬───────────────────────────────┘
                      │
                      ▼
```

```
Step 5: Generate embeddings & index
    │
    ├──→ vector_store.add_documents(documents)
    │
    ▼
```

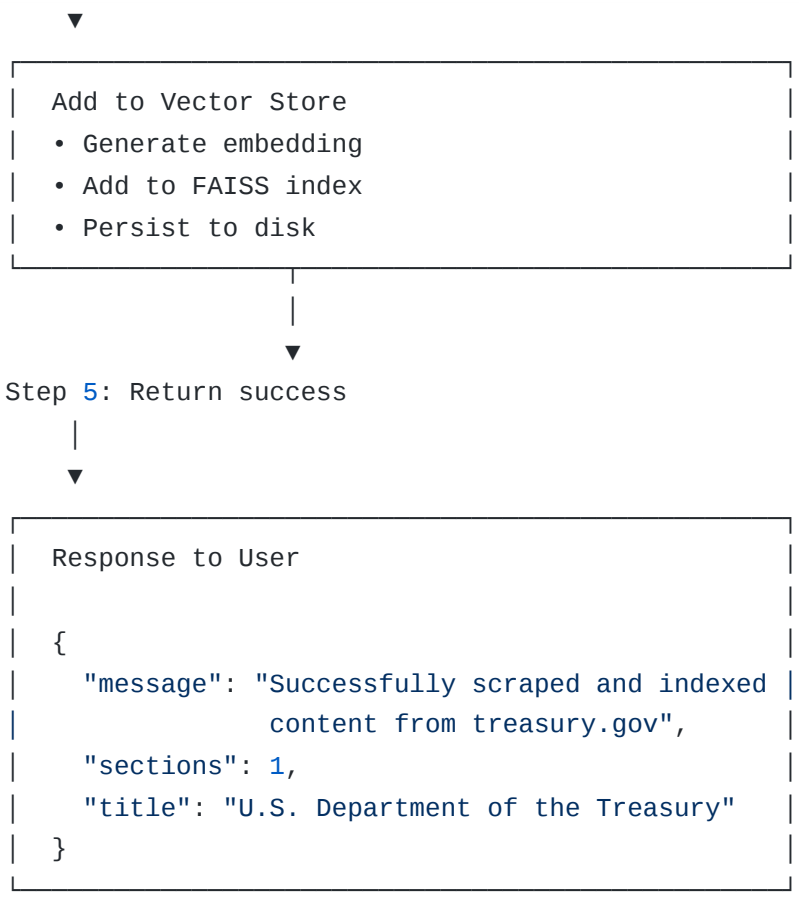
```
┌─────────────────────────────────────────────────────────────┐
```


FAISS Indexing Process
For each document:
1. Generate embedding
├→ SentenceTransformer.encode(content)
└→ 384-dim vector
2. Add to FAISS index
├→ index.add(embedding)
└→ Store metadata
3. Persist to disk
├→ faiss.write_index()
└→ pickle.dump(metadata)
Result: Documents now searchable
Step 6: Return success response
Response to User
{
"message": "Successfully processed and indexed 42 sections from policy_doc.xml",
"sections": 42
}

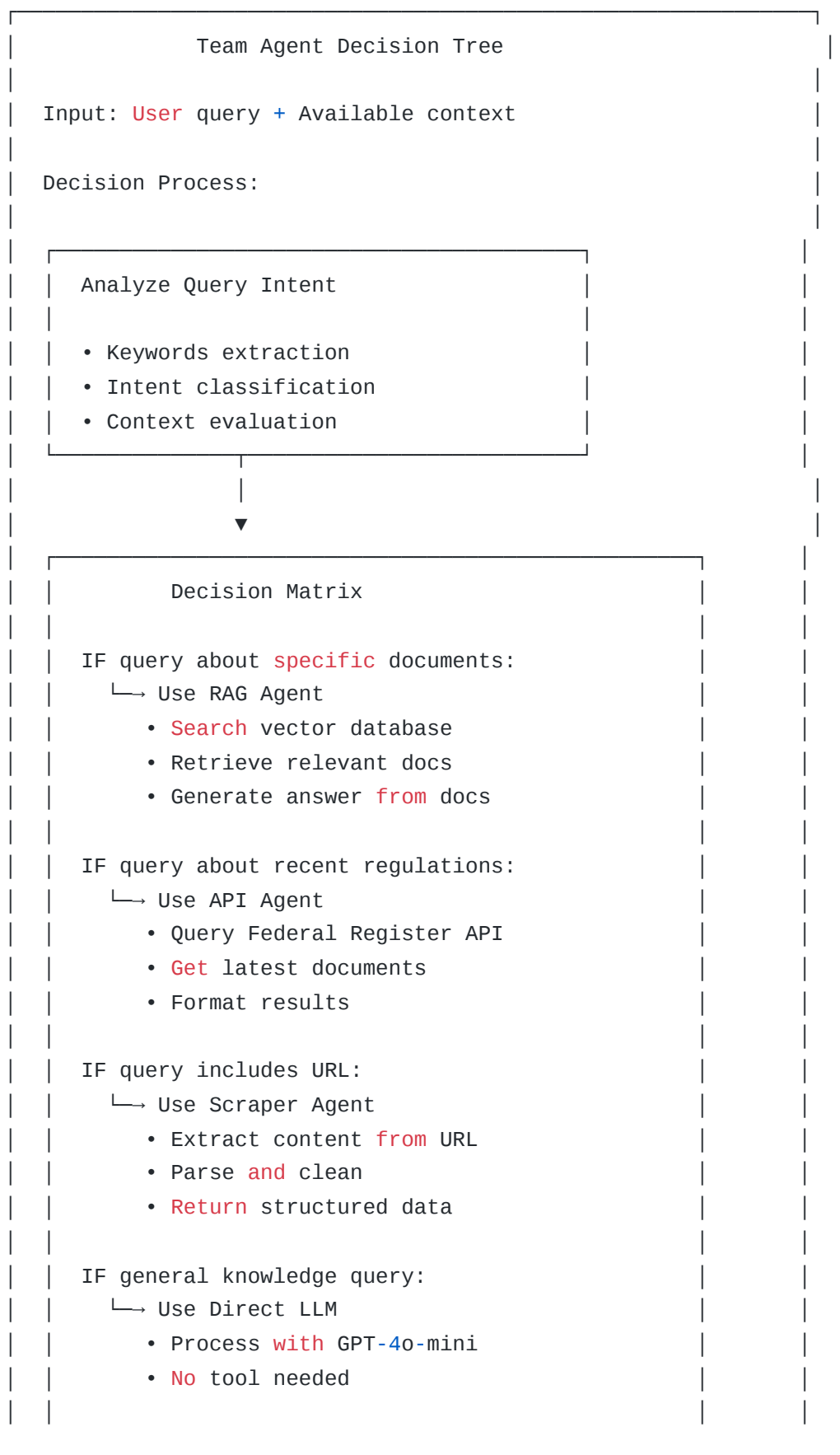
4. تدفق استخراج المحتوى من URL (URL Scraping Flow)



```
│  2. HTML Parsing                                          │
│     ├─→ BeautifulSoup(html, 'html.parser')                │
│     └─→ Parse DOM structure                               │
│                                                           │
│  3. Content Extraction                                    │
│     ├─→ Extract title                                     │
│     │   ├─→ Try <title> tag                               │
│     │   └─→ Fallback to <h1>                              │
│     │                                                     │
│     ├─→ Extract main content                              │
│     │   ├─→ Find <main>, <article>, or <div>              │
│     │   ├─→ Remove <script>, <style>, <nav>               │
│     │   └─→ Extract clean text                            │
│     │                                                     │
│     └─→ Extract links                                     │
│         ├─→ Find all <a> tags                             │
│         └─→ Convert to absolute URLs                      │
│                                                           │
│  4. Metadata                                              │
│     ├─→ Check if government site (.gov domain)            │
│     ├─→ Count words                                       │
│     └─→ Set status                                        │
│                                                           │
│  Output: Scraped content object                           │
└──────────────────────┬────────────────────────────────────┘
                       │
                       ▼
┌──────────────────────────────────────────────────┐
│  Scraping Result                                 │
│                                                  │
│  {                                               │
│    "url": "https://home.treasury.gov/",          │
│    "title": "U.S. Department of the Treasury",   │
│    "content": "The Treasury Department is...",   │
│    "links": [...],                               │
│    "is_government": true,                        │
│    "status": "success",                          │
│    "word_count": 1247                            │
│  }                                               │
└──────────────────────┬───────────────────────────┘
                       │
                       ▼
Step 4: Index scraped content
    │
    ├─→ Same as file upload indexing process
    │
```


Add to Vector Store
• Generate embedding
• Add to FAISS index
• Persist to disk
Step 5: Return success
Response to User
{
"message": "Successfully scraped and indexed content from treasury.gov",
"sections": 1,
"title": "U.S. Department of the Treasury"
}

5. اختيار الأداة المناسبة (Tool Selection Logic)



```
|  |  IF complex multi-step query:                          |       |
|  |     └──→ Use Multiple Agents                           |       |
|  |        • Coordinate sub-agents                         |       |
|  |        • Combine results                               |       |
|  |        • Synthesize final answer                       |       |
|  └────────────────────────────────────────────────────────┘       |
|                                                                   |
└───────────────────────────────────────────────────────────────────┘
```

أمثلة على اختيار الأداة:

Query Example	Selected Tool	Reason
"What are EPA air quality standards?"	RAG Agent	Document-based question
"Latest EPA rules published this week"	API Agent	Recent/time-sensitive
"Extract content from epa.gov"	Scraper Agent	URL provided
"What is the Clean Air Act?"	Direct LLM	General knowledge
"Find recent rules about air quality and summarize"	Multiple Agents	Multi-step task

---

6. التقنيات المستخدمة (Technologies Used)

Backend Stack:

```
┌─────────────────────────────────────────┐
│  Python 3.9+                            │
│  ├──→ Flask 3.0.0 (Web framework)       │
│  ├──→ Flask-CORS (Cross-origin)         │
│  └──→ python-dotenv (Environment vars)  │
│                                         │
│  Vector Database:                       │
│  ├──→ FAISS (Facebook AI Similarity)    │
│  └──→ sentence-transformers             │
│      └──→ all-MiniLM-L6-v2 model        │
│                                         │
│  AI/ML:                                 │
│  ├──→ aiXplain SDK 0.2.36               │
│  │   ├──→ AgentFactory                  │
│  │   └──→ ModelFactory                  │
│  └──→ GPT-4o-mini (via aiXplain)        │
│                                         │
│  Web Scraping:                          │
│  ├──→ BeautifulSoup4                    │
│  ├──→ requests                          │
│  └──→ lxml                              │
│                                         │
│  Data Processing:                       │
│  ├──→ numpy                             │
│  └──→ pickle                            │
└─────────────────────────────────────────┘
```

:Frontend Stack

```
┌─────────────────────────────────────────┐
│  HTML5 + CSS3 + JavaScript              │
│  ├──→ Responsive design                 │
│  ├──→ Dark/Light mode toggle            │
│  └──→ Modern UI components              │
│                                         │
│  Features:                              │
│  ├──→ Drag & drop file upload           │
│  ├──→ Real-time query processing        │
│  ├──→ Animated gradients                │
│  └──→ Interactive elements              │
└─────────────────────────────────────────┘
```

---

7. تدفق البيانات الكامل (Complete Data Flow)

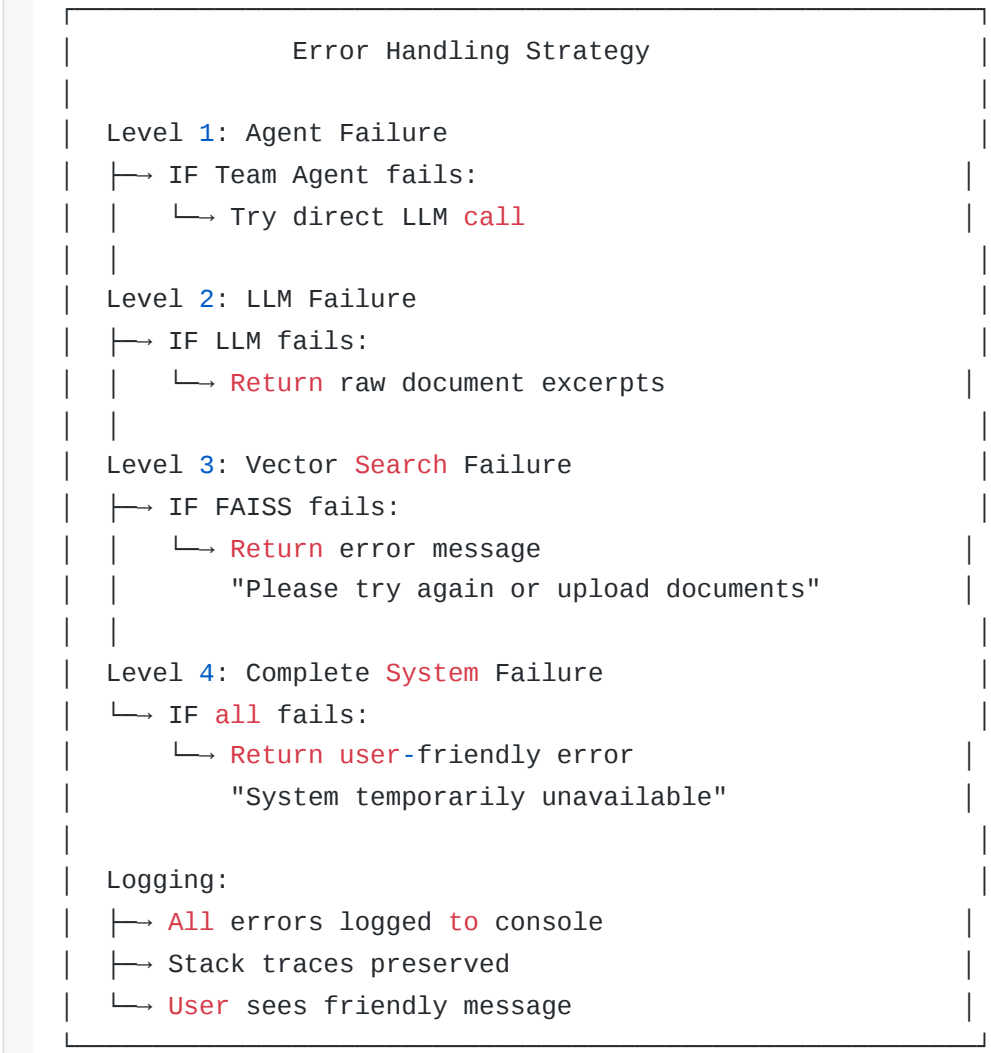
```
┌─────────────────────────────────────────────────────────────────┐
│                    Complete System Flow                         │
└─────────────────────────────────────────────────────────────────┘

1. User Interaction
   │
   ├──→ Query: "What are EPA regulations?"
   │
   ▼
2. Frontend Processing
   │
   ├──→ Validate input
   ├──→ Show loading state
   ├──→ Send AJAX request
   │
   ▼
3. Backend Reception
   │
   ├──→ Flask route: /api/query
   ├──→ Extract query text
   │
   ▼
4. Vector Search
   │
   ├──→ Generate query embedding (384-dim)
   ├──→ Search FAISS index (L2 distance)
   ├──→ Retrieve top 3 documents
   │    └──→ [Doc1: score=0.87, Doc2: score=0.82, Doc3: score=0.78]
   │
   ▼
5. Context Preparation
   │
   ├──→ Format documents
   ├──→ Build context object
   ├──→ Prepare agent prompt
   │
   ▼
6. Agent Processing
   │
   ├──→ Load Team Agent (ID: 6905048fa1a609715ed913cc)
   ├──→ Send query + context
   │
```

```
    ▼
7. Team Agent Decision
    │
    ├──→ Analyze query intent
    ├──→ Evaluate available context
    ├──→ Select appropriate sub-agent
    │    └──→ Decision: Use RAG Agent
    │
    ▼
8. RAG Agent Execution
    │
    ├──→ Read retrieved documents
    ├──→ Process with LLM (GPT-4o-mini)
    │    ├──→ Understand query
    │    ├──→ Extract relevant info
    │    └──→ Generate coherent answer
    │
    ▼
9. Response Synthesis
    │
    ├──→ Format answer
    ├──→ Add metadata
    ├──→ Include source info
    │
    ▼
10. Return to Frontend
     │
     ├──→ JSON response
     ├──→ Display formatted answer
     ├──→ Show source documents
     └──→ Display confidence score
```

---

8. معالجة الأخطاء والاستراتيجيات الاحتياطية (Error Handling & Fallback)



9. الأداء والتحسينات (Performance & Optimization)

Vector Search Performance:

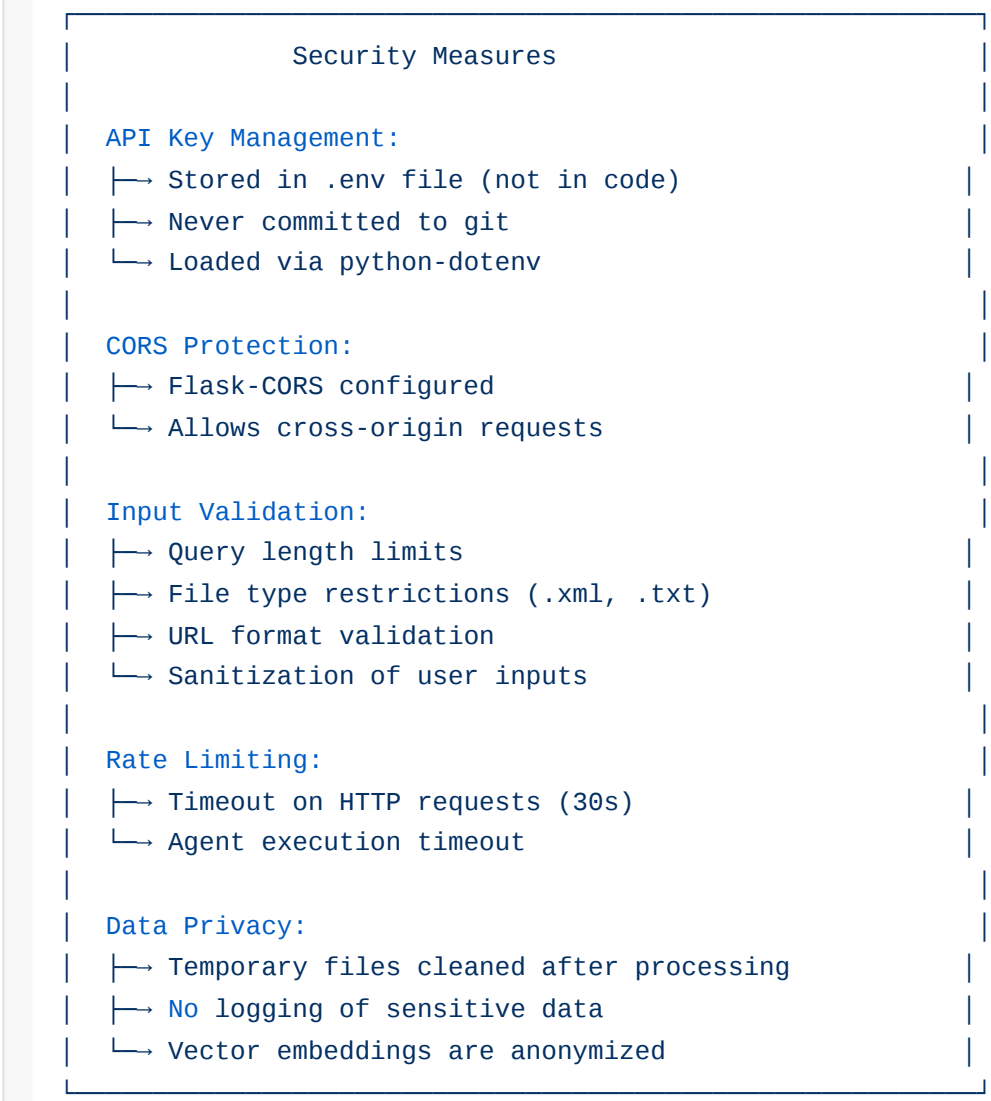
```
┌─────────────────────────────────────────┐
│  FAISS Index Statistics                 │
│                                         │
│  • Index type: IndexFlatL2              │
│  • Embedding dimension: 384             │
│  • Total documents: 3,136               │
│  • Average search time: ~50ms           │
│  • Memory usage: ~5MB                   │
│                                         │
│  Optimization:                          │
│  ├─→ Batch embedding generation         │
│  ├─→ Index persistence to disk          │
│  └─→ Lazy loading of embeddings         │
└─────────────────────────────────────────┘
```

Agent Response Time:

```
┌─────────────────────────────────────────┐
│  Response Time Breakdown                │
│                                         │
│  1. Vector search: ~50ms                │
│  2. Context preparation: ~10ms          │
│  3. Agent processing: ~2-5s             │
│     ├─→ Agent decision: ~500ms          │
│     └─→ LLM generation: ~1.5-4.5s       │
│  4. Response formatting: ~5ms           │
│                                         │
│  Total: ~2-5 seconds                    │
└─────────────────────────────────────────┘
```

---

10. الأمان وأفضل الممارسات (Security & Best Practices)



الخلاصة (Summary)

نظام **Policy Navigator** هو نظام RAG متقدم يجمع بين:

1. **البحث الشعاعي** (FAISS) - للعثور على المستندات ذات الصلة
2. **الوكلاء المتعددين** (Multi-Agent) - لاتخاذ القرارات الذكية
3. **نماذج اللغة الكبيرة** (LLM) - لتوليد إجابات طبيعية
4. **أدوات مخصصة** (Custom Tools) - لمعالجة البيانات المتخصصة

التدفق الكامل:

```
User Query → Vector Search → Context Prep → Team Agent →
Sub-Agent Selection → LLM Processing → Response Generation →
User Interface
```

النظام يوفر:

- ✅ إجابات دقيقة مبنية على المستندات
- ✅ اختيار ذكي للأدوات المناسبة
- ✅ معالجة أخطاء قوية
- ✅ أداء محسّن
- ✅ واجهة مستخدم حديثة

تم إنشاء هذا المستند بواسطة: Policy Navigator Technical Team
التاريخ: November 2025
الإصدار: 2.0 (Multi-Agent System)