Usage Examples

This guide provides practical examples of using the RAG System.

Table of Contents

Basic Usage

Example 1: Upload and Query a Research Paper

1. **Upload a PDF**:
   - Navigate to “Upload PDF” tab
   - Select your research paper (e.g., `machine_learning_paper.pdf`)
   - Wait for processing

2. **Ask Questions**:
   `Question: What is the main contribution of this paper?`

`Question: What datasets were used in the experiments?`

`Question: What were the key results?`

Example 2: Scrape and Query Documentation

1. **Scrape Documentation**:
   - Navigate to “Scrape URL” tab
   - Enter: `https://docs.python.org/3/tutorial/index.html`
   - Click “Scrape & Index”

2. **Ask Questions**:
   `Question: How do I create a virtual environment in Python?`

`Question: What are the basic data types in Python?`

Example 3: Multiple Sources

1. **Upload multiple PDFs** about climate change
2. **Scrape URLs** from climate science websites
3. **Ask comparative questions**:
   `Question: What are the common themes across all sources about climate change?`

API Examples

Using cURL

Upload PDF

```
curl -X POST "http://localhost:8000/api/upload-pdf" \
  -H "Content-Type: multipart/form-data" \
  -F "file=@/path/to/document.pdf"
```

Scrape URL

```
curl -X POST "http://localhost:8000/api/scrape-url" \
  -H "Content-Type: application/json" \
  -d '{"url": "https://example.com/article"}'
```

Query

```
curl -X POST "http://localhost:8000/api/query" \
  -H "Content-Type: application/json" \
  -d '{
    "question": "What is the main topic?",
    "n_results": 5,
    "return_sources": true
  }'
```

Get Statistics

```
curl -X GET "http://localhost:8000/api/stats"
```

List Documents

```
curl -X GET "http://localhost:8000/api/documents"
```

Delete Document

```
curl -X DELETE "http://localhost:8000/api/documents/{document_id}"
```

Using Python

```python
import requests

BASE_URL = "http://localhost:8000/api"

# Upload PDF
def upload_pdf(file_path):
    with open(file_path, 'rb') as f:
        files = {'file': f}
        response = requests.post(f"{BASE_URL}/upload-pdf", files=files)
    return response.json()

# Scrape URL
def scrape_url(url):
    data = {"url": url}
    response = requests.post(f"{BASE_URL}/scrape-url", json=data)
    return response.json()

# Query
def query(question, n_results=5):
    data = {
        "question": question,
        "n_results": n_results,
        "return_sources": True
    }
    response = requests.post(f"{BASE_URL}/query", json=data)
    return response.json()

# Example usage
result = upload_pdf("document.pdf")
print(f"Uploaded: {result}")

result = scrape_url("https://example.com")
print(f"Scraped: {result}")

result = query("What is the main topic?")
print(f"Answer: {result['answer']}")
print(f"Sources: {len(result['sources'])}")
```

Using JavaScript/Node.js

```javascript
const axios = require('axios');
const FormData = require('form-data');
const fs = require('fs');

const BASE_URL = 'http://localhost:8000/api';

// Upload PDF
async function uploadPDF(filePath) {
  const form = new FormData();
  form.append('file', fs.createReadStream(filePath));

  const response = await axios.post(`${BASE_URL}/upload-pdf`, form, {
    headers: form.getHeaders()
  });
  return response.data;
}

// Scrape URL
async function scrapeURL(url) {
  const response = await axios.post(`${BASE_URL}/scrape-url`, { url });
  return response.data;
}

// Query
async function query(question, nResults = 5) {
  const response = await axios.post(`${BASE_URL}/query`, {
    question,
    n_results: nResults,
    return_sources: true
  });
  return response.data;
}

// Example usage
(async () => {
  try {
    const uploadResult = await uploadPDF('document.pdf');
    console.log('Uploaded:', uploadResult);

    const scrapeResult = await scrapeURL('https://example.com');
    console.log('Scraped:', scrapeResult);

    const queryResult = await query('What is the main topic?');
    console.log('Answer:', queryResult.answer);
    console.log('Sources:', queryResult.sources.length);
  } catch (error) {
    console.error('Error:', error.message);
  }
})();
```

Advanced Queries

Technical Documentation

Scenario: You have Python documentation indexed

```
Q: How do I handle exceptions in Python?
Q: What's the difference between lists and tuples?
Q: Show me examples of list comprehensions
```

Research Papers

Scenario: Multiple ML research papers indexed

```
Q: What are the common evaluation metrics used across these papers?
Q: Which papers discuss transformer architectures?
Q: Compare the datasets used in different papers
Q: What are the limitations mentioned in the papers?
```

Business Documents

Scenario: Company policies and reports indexed

```
Q: What is our remote work policy?
Q: What were the key achievements in Q2?
Q: What are the upcoming deadlines mentioned?
Q: Who are the stakeholders in project X?
```

News Articles

Scenario: News articles about AI indexed

```
Q: What are the latest developments in AI regulation?
Q: Which companies are mentioned most frequently?
Q: What are the concerns raised about AI?
Q: Summarize the main trends discussed
```

Best Practices

1. Document Preparation

- **PDFs**: Ensure text is selectable (not scanned images)
- **Web Pages**: Scrape article pages, not just landing pages
- **Naming**: Use descriptive filenames for PDFs

2. Asking Questions

Good Questions:
- "What are the three main points discussed in the introduction?"
- "List the key findings from the experiment"
- "Who are the authors mentioned in the acknowledgments?"

Less Effective Questions:
- "Tell me everything" (too broad)
- "Yes or no?" (need more context)
- Questions about content not in documents

3. Query Optimization

- **Be Specific**: Instead of "What is this about?", ask "What is the main hypothesis tested?"

- **Use Keywords**: Include specific terms from your documents
- **Adjust Sources**: If answer is incomplete, increase `n_results` to 7-10

4. Source Management

- **Regular Cleanup**: Delete outdated documents
- **Organize by Topic**: Use descriptive URLs/filenames
- **Update Regularly**: Re-scrape URLs periodically for fresh content

5. Performance Tips

- **Batch Uploads**: Upload multiple PDFs at once
- **Monitor Stats**: Check document count and sources regularly
- **Clear Unused**: Remove documents you no longer need

Use Cases

1. Academic Research

```python
# Upload all papers in a directory
import os
import requests

papers_dir = "/path/to/papers"
for filename in os.listdir(papers_dir):
    if filename.endswith(".pdf"):
        upload_pdf(os.path.join(papers_dir, filename))

# Ask comparative questions
query("What methodologies are commonly used?")
query("Which papers achieve the best results?")
```

2. Technical Documentation Hub

```python
# Scrape documentation sites
docs_urls = [
    "https://docs.python.org/3/library/index.html",
    "https://fastapi.tiangolo.com/",
    "https://docs.docker.com/get-started/"
]

for url in docs_urls:
    scrape_url(url)

# Ask technical questions
query("How do I containerize a FastAPI application?")
```

3. Content Analysis

```python
# Upload articles or reports
for article in article_list:
    upload_pdf(article)

# Analyze content
query("What are the key themes discussed?")
query("What sentiment is expressed?")
query("List all statistics mentioned")
```

4. Knowledge Base

```python
# Build company knowledge base
scrape_url("https://company.com/policies")
scrape_url("https://company.com/procedures")
upload_pdf("employee_handbook.pdf")

# Query for information
query("What is the vacation policy?")
query("How do I submit an expense report?")
```

Example Session

Here’s a complete example session:

```
# 1. Start the server
./run.sh

# 2. Upload a document (using web interface)
# - Upload "AI_Ethics_Report_2024.pdf"

# 3. Scrape related content
# - Scrape "https://ai-ethics.org/principles"

# 4. Ask questions
Q: What are the main ethical concerns discussed?
A: The document discusses several key ethical concerns including bias in AI systems,
   privacy implications, transparency in decision-making, and accountability...

Q: What recommendations are made?
A: The report recommends establishing clear governance frameworks, implementing
   regular audits, ensuring diverse development teams...

Q: Are there any case studies mentioned?
A: Yes, the report includes case studies on facial recognition bias, healthcare AI
   decision systems, and automated hiring tools...

# 5. Check sources
# - View the sources tab to see which documents the answers came from

# 6. Clean up if needed
# - Delete outdated sources from the Manage Documents tab
```

Happy Querying! 🎯