```
import requests
import json

def fetch_data(api_url, page=1):
    """Fetch data from the paginated API."""
    response = requests.get(f"{api_url}?page={page}")
    if response.status_code == 200:
        return response.json()
    else:
        return None

def identify_citations(response_text, sources):
    """Identify which sources contributed to the response text."""
    citations = []
    for source in sources:
        if source['context'] in response_text:
            citations.append(source)
    return citations

def process_responses(api_url):
    """Process all pages of the API and return the citations for each response."""
    page = 1
    all_citations = []
    while True:
        data = fetch_data(api_url, page)
        if not data:
            break
        for item in data:
            response_text = item['response']
            sources = item['sources']
            citations = identify_citations(response_text, sources)
            all_citations.append({'response': response_text, 'citations': citations})
        page += 1
    return all_citations

def main():
    api_url = "https://devapi.beyondchats.com/api/get_message_with_sources"
    citations = process_responses(api_url)
    print(json.dumps(citations, indent=2))

if __name__ == "__main__":
    main()
```

```
[]
```

```
pip install Flask
```

```
Requirement already satisfied: Flask in /usr/local/lib/python3.10/dist-packages (2.2.5)
Requirement already satisfied: Werkzeug>=2.2.2 in /usr/local/lib/python3.10/dist-packages (from Flask) (3.0.3)
Requirement already satisfied: Jinja2>=3.0 in /usr/local/lib/python3.10/dist-packages (from Flask) (3.1.4)
Requirement already satisfied: itsdangerous>=2.0 in /usr/local/lib/python3.10/dist-packages (from Flask) (2.2.0)
Requirement already satisfied: click>=8.0 in /usr/local/lib/python3.10/dist-packages (from Flask) (8.1.7)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from Jinja2>=3.0->Flask) (2.1.5)
```

```
from flask import Flask, jsonify
import requests

app = Flask(__name__)

def fetch_data(api_url, page=1):
    response = requests.get(f"{api_url}?page={page}")
    if response.status_code == 200:
        return response.json()
    else:
        return None

def identify_citations(response_text, sources):
    citations = []
    for source in sources:
        if source['context'] in response_text:
            citations.append(source)
    return citations

def process_responses(api_url):
    page = 1
    all_citations = []
    while True:
        data = fetch_data(api_url, page)
        if not data:
            break
        for item in data:
            response_text = item['response']
            sources = item['sources']
            citations = identify_citations(response_text, sources)
            all_citations.append({'response': response_text, 'citations': citations})
        page += 1
    return all_citations

@app.route('/citations', methods=['GET'])
def get_citations():
    api_url = "https://devapi.beyondchats.com/api/get_message_with_sources"
    citations = process_responses(api_url)
    return jsonify(citations)

if __name__ == "__main__":
    app.run(debug=True)
```

```
 * Serving Flask app '__main__'
 * Debug mode: on
INFO:werkzeug:WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead
 * Running on http://127.0.0.1:5000
INFO:werkzeug:Press CTRL+C to quit
INFO:werkzeug: * Restarting with stat
```