https://youtu.be/vKvNmotlq88?si=DILvowbg04f4aeMi

SearXng: How to Set Up Your Private Search Engine! (AI APP Integration)

Introduction

SearXng is a free, privacy-focused meta-search engine that aggregates results from multiple search services without tracking or profiling users. In this tutorial, a video creator demonstrates how to install SearXng locally using Docker, configure it for API integration, and connect it to AI applications such as Perplexa. By following these steps, developers can embed a private search backend into their AI workflows while preserving user privacy. Let's dive into the setup process and see how SearXng can power your next AI project.

Report Body

Overview of SearXng

SearXng is a free, privacy-focused meta-search engine that aggregates results from multiple search services without tracking or profiling users. It runs entirely on the user's machine, exposing a JSON-formatted API that can be consumed by other applications. The project is open source and hosted on GitHub, allowing community contributions and customizations.

Key Features

- **Local Operation** – Eliminates external tracking.
- **Meta-Search Capability** – Pulls results from a variety of search engines and databases.
- **API-Ready** – Provides a machine-readable JSON interface.
- **Open Source** – Available at `https://github.com/searxng/searxng`.

Installation Process

Prerequisites

- Docker installed and running on the host machine.
- Basic command-line knowledge (Git, terminal commands).

Cloning the Repository

```
bash git clone https://github.com/cxng/CRX cd CRX
```

(The repository name in the video is `CRX`*; adjust if the actual repo differs.)*

Configuring Settings

1. Open the project in a code editor (e.g., VS Code).
2. Navigate to `settings/` → `settings.yml` (or `settings.json`).
3. Locate the `search` section and set the output format to **JSON**: `yaml search: format: json`
4. Save the file. This ensures the API returns data in a machine-readable format suitable for AI integration.

Docker Workflow

Building the Docker Image

```
bash make docker-build
```

The build process pulls necessary dependencies and creates a Docker image named `searxng`. Verify the image with:

```
bash docker image ls
```

You should see an entry for `searxng` with the latest tag.

Running the Docker Container

```
bash docker run -d -p 32768:8080 searxng
```

- `-d` runs the container in detached mode.
- `-p 32768:8080` maps host port `32768` to container port `8080`.

Confirm the container is running:

```
bash docker ps
```

Verifying the Container

Access the local SearXng interface by navigating to `http://localhost:32768` (or `http://127.0.0.1:32768`). Perform a test search (e.g., “OpenAI”) to verify functionality.

Accessing the Local Interface

Once the container is running, the web UI is available at the mapped port. A quick search confirms that results are aggregated from the configured search engines and returned in JSON format when the API endpoint is queried.

Integration with AI Applications

Example: Perplexa

Perplexa is an AI search engine that can consume external search APIs. When running Perplexa via Docker, it automatically detects the SearXng instance on port `32768`. If running Perplexa manually, configure its settings to point to `http://localhost:32768` as the search backend.

General Integration Steps

1. **Expose the API Endpoint** – Ensure the SearXng container is reachable at a stable URL/port.
2. **Configure the AI App** – In the AI application's settings, specify the SearXng API URL and any required authentication (none needed for local setups).
3. **Test the Connection** – Perform a query from the AI app and confirm that results are returned from SearXng.

These steps provide a straightforward path to embedding a private, local search engine into AI workflows, preserving user privacy while offering a flexible search backend.

Conclusion

By following the step-by-step Docker workflow, you can spin up a fully private SearXng instance that delivers JSON-formatted search results ready for AI consumption. Integrating it with tools like Perplexa is as simple as pointing the AI app to the local endpoint, ensuring seamless, privacy-preserving search capabilities. This setup empowers developers to embed a customizable, open-source search backend into their AI pipelines without relying on external services.