Recipe Search API Documentation

Setup Instructions

1. Install dependencies:

- Make sure you have Python 3.7+ installed.
- Run the following command to install necessary libraries:

  `pip install fastapi[all] opensearch-py python-dotenv`

2. Environment setup:

- Create a `.env` file in the project root directory with the following

  content: OPENSEARCH_HOST=<Your_OpenSearch_Host>

  OPENSEARCH_PORT=<Your_OpenSearch_Port (e.g., 9200)>

  OPENSEARCH_USERNAME=<Your_OpenSearch_Username>

  OPENSEARCH_PASSWORD=<Your_OpenSearch_Password>

3. OpenSearch setup:

- Ensure that OpenSearch is properly set up and running.
- You can use Docker to set up OpenSearch:

```
docker run -d --name opensearch -p 9200:9200 -p 9600:9600 -e "discovery.type=single-node" opensearchproject/opensearch:latest
```

4. Running the FastAPI Application:

- Start the FastAPI server by running the following command in the terminal:

  `uvicorn main:app --reload`

- The server will be accessible at `http://127.0.0.1:8000`.

5. OpenAPI documentation:

- You can access the automatically generated API documentation at:

  `http://127.0.0.1:8000/docs`

- You can also see the OpenAPI schema in JSON format at:

  `http://127.0.0.1:8000/openapi.json`

API Endpoints

1. GET /search

- Description: Search for recipes based on various filters. This is the main search endpoint.

- Query Parameters:

  - Query (Optional): Keywords to search in the recipe's title, ingredients, or instructions.

  - Category (Optional): Filter recipes by a specific category (e.g., 'dessert', 'breakfast').

  - Ingredients (Optional): Filter recipes that contain specific ingredients. Provide this as a list (e.g.,`ingredients=egg&ingredients=sugar`).

  - Min_rating, max_rating (Optional): Filter recipes within a rating range (e.g., min_rating=4).

  - Min_protein, max_protein (Optional): Filter recipes based on protein content.

  - Page (Optional): Page number for pagination. Default is 1.

  - Size (Optional): Number of results per page. Default is 10, with a maximum limit of 100.

- Response:

  - Total: Total number of recipes matching the criteria.

  - Recipes: List of recipes with details such as title, ingredients, categories, calories, protein, fat, sodium, rating, and directions.

2. GET /filter/categories

- Description: Fetch all unique recipe categories available in the database.
- Response: A list of unique category names used in the recipes.

3. GET /filter/ingredients

- Description: Fetch all unique ingredients available in the recipe index for filtering purposes.

- Response: A list of unique ingredient names found across all recipes.

4. Error Handling:

  - 404 Not Found: Returned if the OpenSearch index is not found.
  - 500 Internal Server Error: Returned if there's an issue retrieving categories or ingredients.

OpenSearch Connection

- The OpenSearch client is initialized using the `opensearch-py` library.
- The connection details (host, port, and authentication) are pulled from environment variables stored in a `.env` file.
- Ensure that the OpenSearch server is accessible using the specified credentials. SSL is used but certificate verification is disabled for local development.

Data Model

- Recipe: Represents a single recipe with the following attributes:
  - Id: Unique identifier for the recipe.
  - Title: Title of the recipe.
  - Ingredients: List of ingredients required for the recipe.
  - Categories: List of categories the recipe belongs to (e.g., 'dessert', 'breakfast').
  - Calories: Total calories of the recipe.
  - Protein: Amount of protein per serving (in grams).
  - Fat: Total fat content (in grams).
  - Sodium: Total sodium content (in milligrams).
  - Rating: Recipe rating on a scale of 0 to 5.
  - Date: The date the recipe was published.
  - Desc: A brief description of the recipe.
  - Directions: Step-by-step directions to prepare the recipe.