[RFC] Content Search Engine For Cloud Storage

Summary: Service to search across contents in cloud storage.

Created: May 26, 2021
Current Version: 0.0.1
Target Version: 1.0.0

Status: WIP | In-Review | Approved | Obsolete

Owner: susmith krishnan

With the increasing usage of cloud services, an efficient method to search file contents is very essential. A huge number of files are uploaded in various clouds on a daily basis. To find something in that heap is a great deal. Here is the proposed service to search across contents in cloud storage. This method helps to search your content across all cloud services irrespective of the providers. The primary database used here is Elastic search and NodeJS is the backend server application.

Background

A single user can have accounts in multiple cloud storage providers such as Google Drive, One Drive, Dropbox, etc. To find a file among those is a tedious task. And existing cloud storage providers only allow users to search the file by its name. Also downloading a large document just for searching a keyword within it is not an efficient solution. This proves the relevance of this application.

Documents scattered in multiple cloud storage are difficult to find, after a period of time. To overcome this, a common search engine which fetches document from all types of cloud is indispensable.

Proposal

The goal is to index content in the files across multiple cloud storage providers and implement a keyword-based text search engine that returns the source file URL. Design and develop a Google-like search interface as a frontend that displays the source files.

Proposed stack:

- Nodejs
- ElasticSearch
- Redis
- Dropbox

Implementation [Backend]

1. **Authenticate to cloud storage providers**:
   (Choosing dropbox for PoC)
   Create a new app in the dropbox developers console. Grant the app following permissions.
   1. *files.metadata.write*
   2. *files.metadata.read*
   3. *files.content.write*
   4. *files.content.read*

   Issue a token and connect to bropbox api v2.

2. **Get the list of text files:**
   Get the list of all files from dropbox recursive mode. The server returns a list of files, folders, and a cursor value. Filter only supported documents from the list. Move the cursor value to Redis to check for updates in the future.

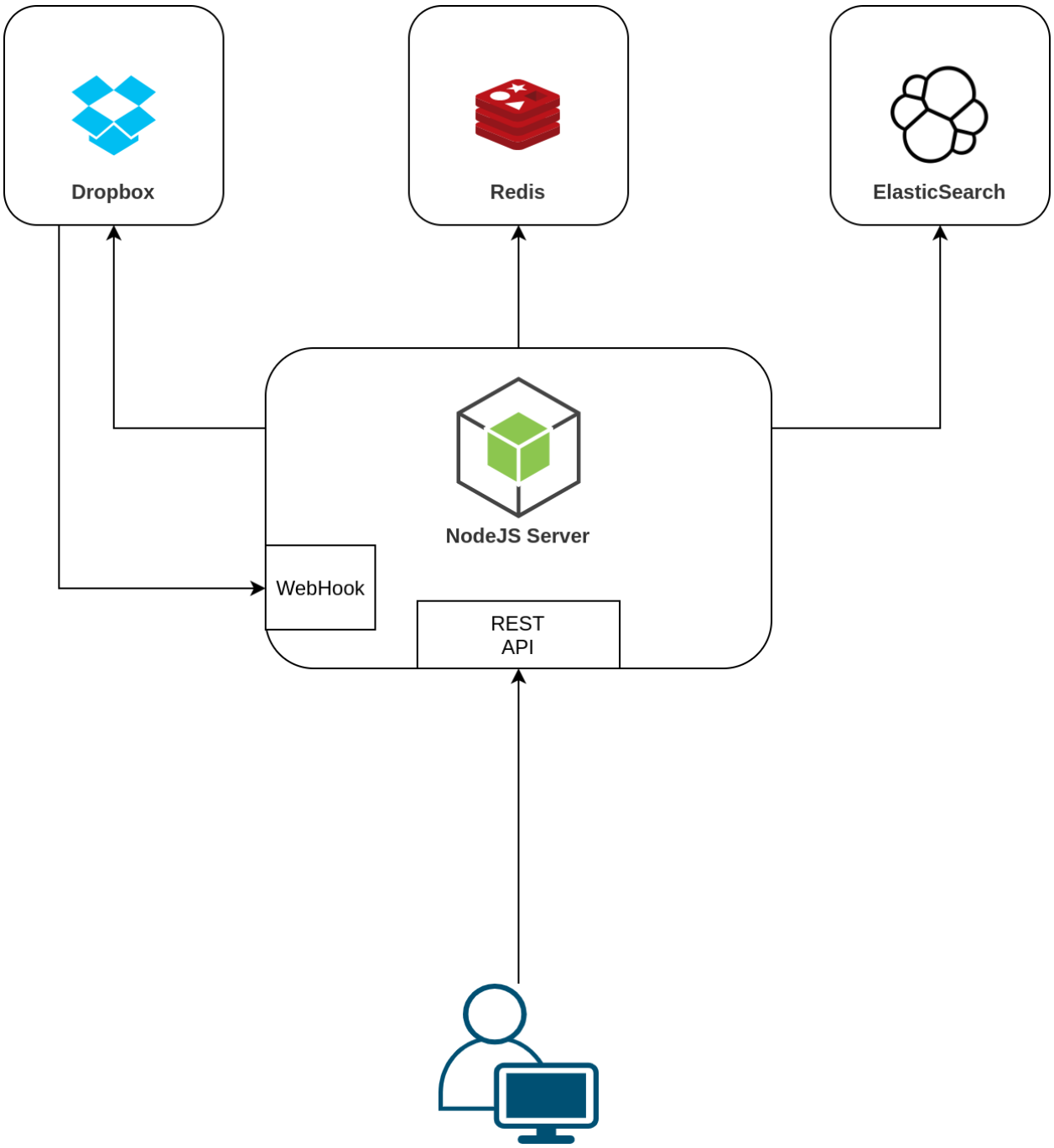
3. **Parse text contents:**
   Extract text content from documents using the textract npm module. Including OCR reader to extract text from images.

4. **ElasticSearch indexing:**
   Connect to ElasticSearch database and add extracted text and metadata as new documents to an index named *files.* Refresh indexes after adding every new batch of documents. The document contains the following fields:
   1. *url: <The dropbox URL of file>*
   2. *fid: <File id assigned by dropbox>*
   3. *hash: <File hash>*
   4. *filename: <Filename>*
   5. *content: <Parsed text content>*

5. **API Endpoint:**
   A REST API that takes a keyword as an input parameter and connects to elasticsearch search API. Returns matching files URL as JSON.
   GET: /api/search?q=<keyword>

6. **Webhooks and cronjobs:**
   A webhook endpoint is configured to actively listen for notifications. Dropbox webhook is verified and connected to this URL. Whenever a file change occurs dropbox will notify the server. Fetch for updates using the previously saved cursor.

Architecture Overview



Implementation [Frontend]

1. Google search like interface with a text input field
2. Upon submitting the form, make a request to the REST API.
3. Display response from the API request as a list.
4. The resulting list should contain the filename and URL.

Frontend design