https://neptune.ai/blog/building-search-engine-with-pre-trained-transformers-guide

The keywords discovered within the page's content – what topics does the page cover? Keywords in data can be used to help.

The type of content that is being crawled (using microdata called Schema) – what is included on the page?

The freshness of the page – how recently was it updated?

The previous user engagement of the page and/or domain – how do people interact with the page?

Retrieve the data we will be searching through. Create a Term-Document Matrix with TF-IDF weighting. Calculate the similarities between documents and search output. Retrieve the articles that have the highest similarity on it.

Establish a Search query and Inverted Index.

Semantic Search:

Use https://huggingface.co/BAAI/bge-large-en-v1.5 or other similar feature extraction models on course descriptions.

Extract vectors, and use cosine similarity or similar distance metric to look at distances between inputted query and course descriptions.

Store vectors, descriptions, other info in a table on supabase.

https://supabase.com/vector

https://supabase.com/docs/guides/database/extensions/pgvector – for embedding search query

- https://www.npmjs.com/package/@xenova/transformers – JS package to use huggingface models

https://supabase.com/docs/guides/database/extensions/pg_cron – may be useful

- Set up a cron job for every _ months to run a python script, which scrapes the courses website.
    - Saves this data to supabase (via https://github.com/supabase-community/supabase-py)