Hypothesis:

- Spring boot project
- Provide a documentation for running it locally:
    - Building instructions
    - Database related properties
- Backend API only
- If you provide a running version on a container or a server, that's a plus

Title: Search engine and simplified annotations

Description

This search engine is developed using web annotation. When users enter specific words or phrases in a search engine, it automatically fetches the most relevant *resources* that contain those keywords. Web annotation makes it possible. Web annotation helps to make an application user-friendly. Thanks to web annotation, users can add, modify, and remove information from Web resources without altering the resource itself.

This project uses web annotation on pages and images. When the user enters words, names, or phrases in the system, it will fetch the information and pictures having the same annotation. Then the system displays a list of results that contain the image or content matching to the user input. For this search engine, you need to use an effective algorithm to generate a query result page/search result records based on users' queries.

Resources:

- HTML pages
- Pictures

Acceptance criteria

- The content of the page can be just an html file with an H1 tag containing a title no need for formatted pages, it's irrelevant
- The API must allow creating two types of objects:
    - HTML pages
    - Images
- For each resource the user must provide the title, the type and the web annotations (let's call them tags)
- The search engine retrieves the list of resources according to a keyword query
    - The list shall contain a path or an id that allows retrieving the resource it self
    - The list must contain resources that match the keyword. partial matching is possible
    - ex: if I have a page with the following web annotations: [books, chapters, pages], the "ook" query shall return this page
- The API shall provide a method for retrieving a given resource (in order to visualize the resource, it self -ex: a picture)
- Provide Javadoc and unit tests for the API (no need for database unit tests)
- The application needs to run with a PostgreSQL database