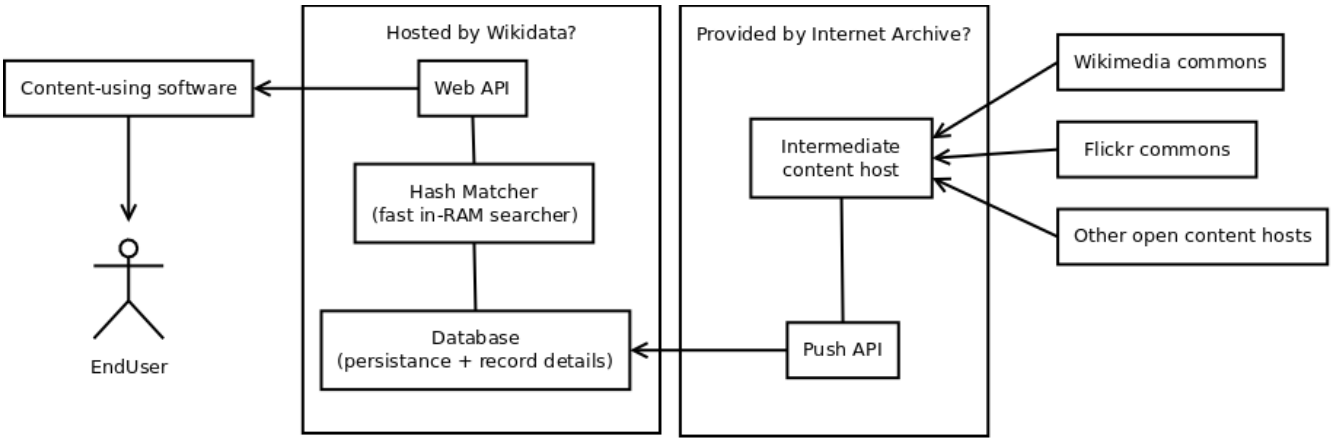
Overview

The Creative Commons registry (working project name) being developed at Seneca College will allow users who have an image to learn what licence that image is available under based on a perceptual hash of the file. Later this can be extended to work with other file types.



The components being built are:

Image downloader

In order to add the image metadata to the database we need the image, or someone to obtain that metadata for us. We also need to have a permanent URL to a page hosting the full-sized image.

This would be quite challenging for us at Seneca given our limits in bandwidth. Projects like the Internet Archive probably have a much cheaper and more efficient way to obtain copies of CC-licenced images.

New image push API

New images are constantly added to the internet, including many CC-licenced images. We don't know the exact number but likely tens of thousands per day (https://en.wikipedia.org/wiki/Wikimedia_Commons#/media/File:Commons_Growth.svg). Ideally a content host would call the registry's API automatically when new images are added. This can be a primary content host (like Flickr) but having a single intermediate host like the Internet Archive would make it much easier to build and maintain.

Some custom code needs to be run on the content host in order to generate the parameters for the requests. For example: generating one or more perceptual hashes, creating a thumbnail. Hashing is a more CPU-heavy than IO-heavy operation: generating a pHash takes about 0.3 seconds for an average image.

Database

All the generated metadata for each image needs to be stored somewhere persistantly. This database is not expected to do complex matching queries, only simple select of an entire row based on id. Preliminary performance test results using PHP/MySQL:

- INSERT times were constant at approximately 16 inserts per second;
- SELECT * was timed at 0.01 seconds (system) for 105,000 records
- SELECT WHERE (search on hash value) averaged at 0.04 seconds in general.

We are hoping to be able to host this database with Wikidata, but currently we don't know much about how that project works and whether a custom database would be ok, or our data would need to fit a predefined schema.

Hash Matcher

The hamming distance search algorithm needed to query against the database is easily parametrizable but otherwise cannot be optimised much. On a dual-core 2.90GHz i7-3520M CPU one search takes 0.85 seconds, not including the IPC that would be needed to receive the request and send a response.

Various options exist for speeding this up including an optimized build (0.5 seconds), caching, a faster server (we're getting a quad-core Xeon for testing), using many more cores (e.g. a 48 or 96 core Arm server), using Cuda on GPUs, or Xeon Phi coprocessors.

The hash matcher and database should be on the same local network if for no other reason than startup time.

Web API

Only one query is expected to get a lot of use, some form of “getMetadata()” which can be configured to return all or a subset of stored metadata for the image. Other web API calls will be added for internal use (e.g. to deal with error reporting and revocation).

The requirements for this component aren't clear at this point. To start with a simple web server should be more than enough. Then obviously the more demand there is for the service - the greater the resources needed to support it.