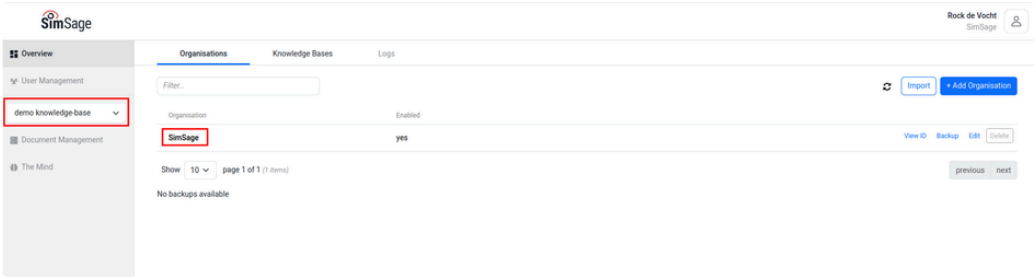
Web Crawler Documentation

Introduction

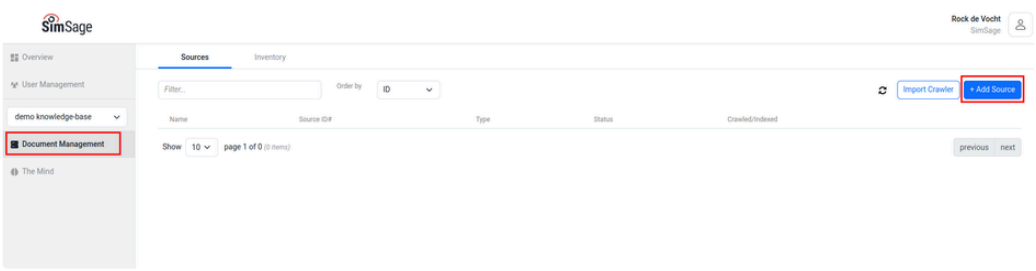
This Document sets out the steps need to step up an internal web crawler on the SimSage platform.

SimSage Source configuration

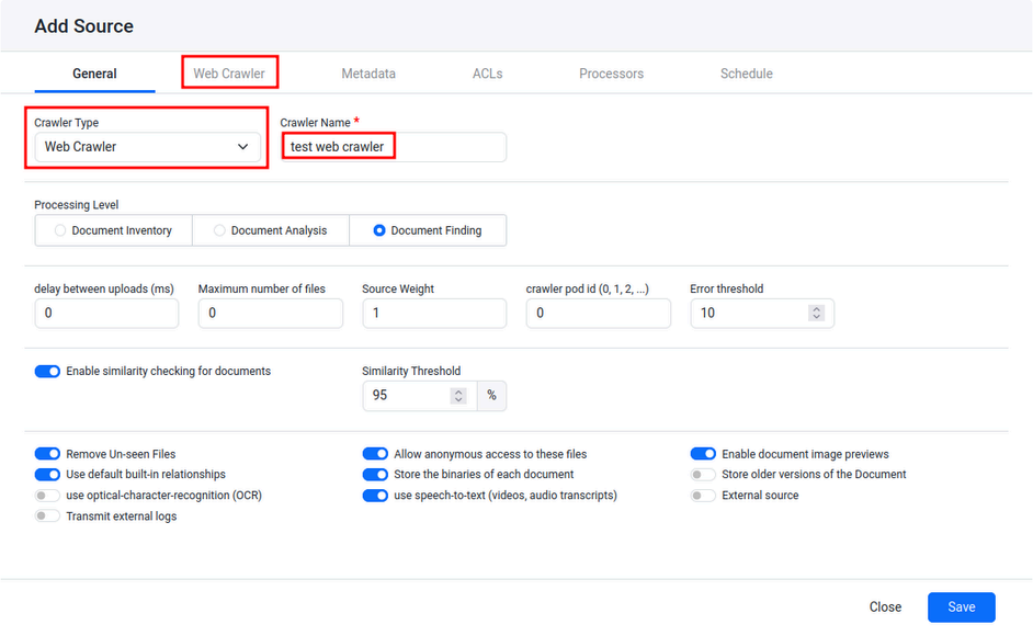
1. Select your organisation (SimSage shown in the image below)
2. Select your knowledge-base from the drop-down box (demo knowledge-base shown below)



3. Click on Document Management link in the left hand side of the menu
4. Click on the + Add Source button



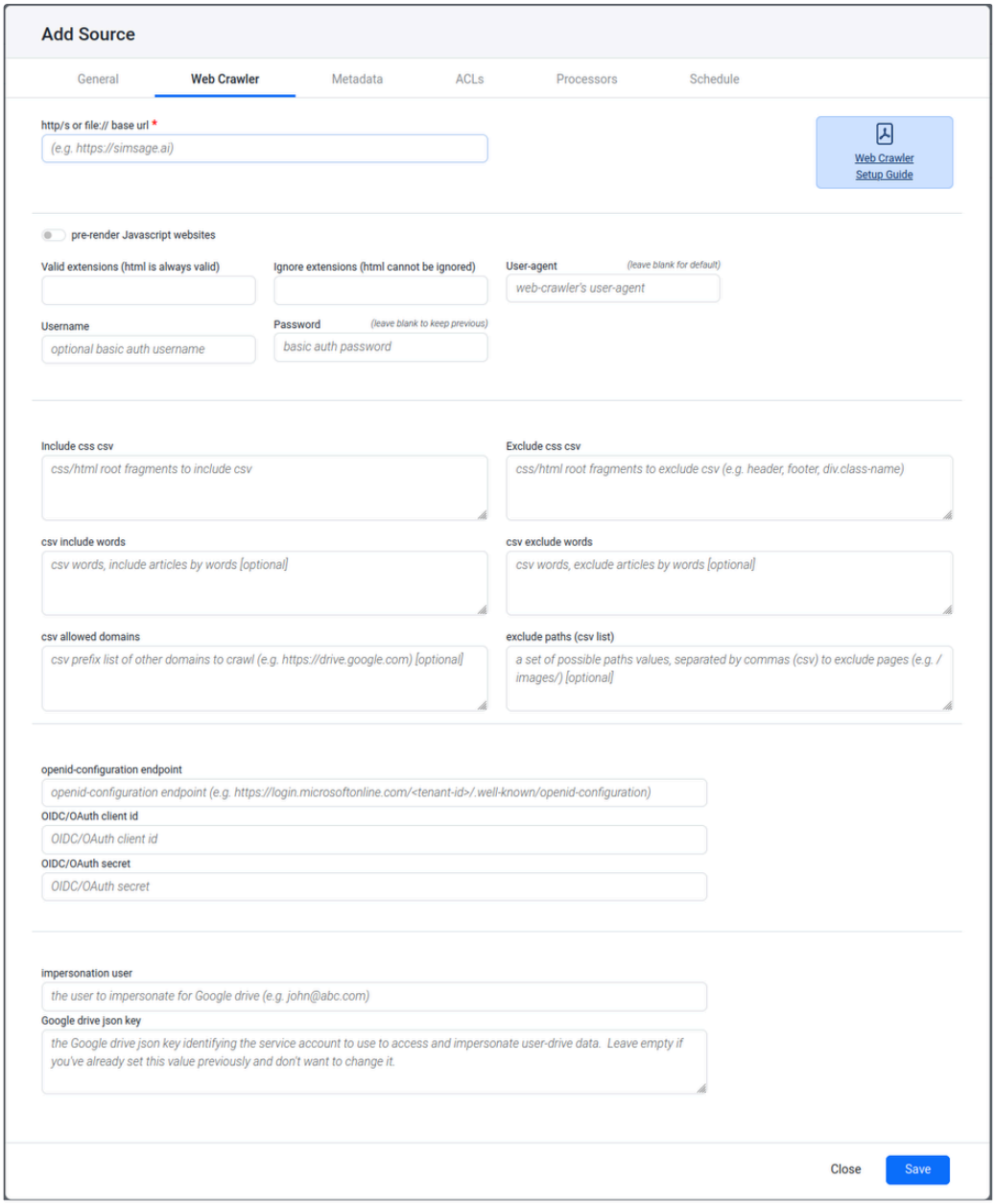
5. In the General tab
    a. set the Crawler Type to “Web Crawler”
    b. give the Crawler a Name (test web crawler in the image below)
    c. Select the “Web Crawler” tab to set up your web crawler



The Web Crawler Tab

1. Navigate to the “Web Crawler” tab.
2. Populate the base url field.
3. Populate the exclude css csv field. As a general rule we recommend using “header, footer” to exclude any repeating headers or footers from your HTML.

> Not all webpages use semantic HTML tags such as <header> or <footer>. Sometimes websites use regular div tags with class names. For example, . In this case you should enter div.footer instead of “footer”.



field	meaning and content
http/s or file:// base url	a compulsory field with a comma separated list of http:// https:// or file:// URLs / URIs to visit. This set forms the seed set for the initial crawler.
pre-render Javascript websites	a special flag that uses a browser to fetch and render remote pages that use active javascript
Valid extensions (html is always valid)	A comma separated list with valid file extensions (e.g. doc, docx, pdf). All file extensions are fetched if this field is left empty. Leave this field empty if you are planning to use “Ignore extensions”.
Ignore extensions (html cannot be ignored)	A comma separated list of extensions that must not be included. Leave this field empty if you are planning to use “Valid extensions”.

User-agent	a string for the user-agent to pass to remote sites in your request headers. Leave empty for default. Default is “Apache-HttpClient/UNAVAILABLE”
Username	the BasicAuth username to send if set along with password in the request headers.
Password	the BasicAuth password to send in the request headers if set.
Include css csv	a comma separated list of CSS selectors for elements to keep in an HTML document. All elements are kept if this field is left blank. Keep this field blank if you’re using the “Exclude css csv” (see: J Use CSS selectors to find elements)
Exclude css csv	a comma separated list of CSS selectors for elements to exclude from an HTML document. Keep this field blank if you’re using the “Include css csv”. (see: J Use CSS selectors to find elements)
csv include words	A comma separated list of words used to filter HTML pages. Any literal “word” found inside the text of an HTML page qualifies this page to be crawled / used by SimSage (e.g. accounting, invoice). All pages are included if left blank. Keep this field blank if you intend to use the “csv exclude words” filter.
csv exclude words	A comma separated list of words used to filter HTML pages. Any literal “word” found inside the text of an HTML page eliminates this page from being crawled / used by SimSage. No pages are excluded if left blank. Leave this field blank if you intent to use the “csv include words” filter.
csv allowed domains	A comma separated list of URL prefixes (e.g. https://another.server.com/folder1/) of other domains apart from the seeds above to allow crawling across.
exclude paths	A comma separated list of path fragments used to excluded pages from being crawled (e.g. /invoices/)

Extra security headers for Bearer Authentication

field	**meaning and content**
openid configuration endpoint	Used for OpenID authentication. The crawler will use this endpoint to get the OpenID configuration required for fetching JWT tokens for Bearer authentication. (e.g. https://server.com/.well-known/openid-configuration)
OIDC/OAuth client id	Used for OpenID authentication. This is the client ID passed to the identification endpoint for acquiring a JWT token.

OIDC/OAuth secret	Used for OpenID autentication. This is the client secret passed to the identification endpoint for acquiring a JWT token.

Google drive authentication

field	meaning and content
impersonation user	A users (an email address) to impersonate to access Google drive links on web pages.
Google drive json key	A Google drive json key (see Google drive setup) for authenticating against Google drive for reading, converting and downloading Google drive links seen on web pages.

NB. The Google drive impersonation user might not have access to links of personal drives of other users if this impersonation user is not in the ACLs for that link. This is a Google drive sharing feature. All shared drive links can be read as long as the drive has been shared with the impersonation user.

the Metadata Tab

The web crawler has the option to replace certain fields of a SimSage record with metadata from HTML pages.

NB. You must save this crawler before altering the mappings.

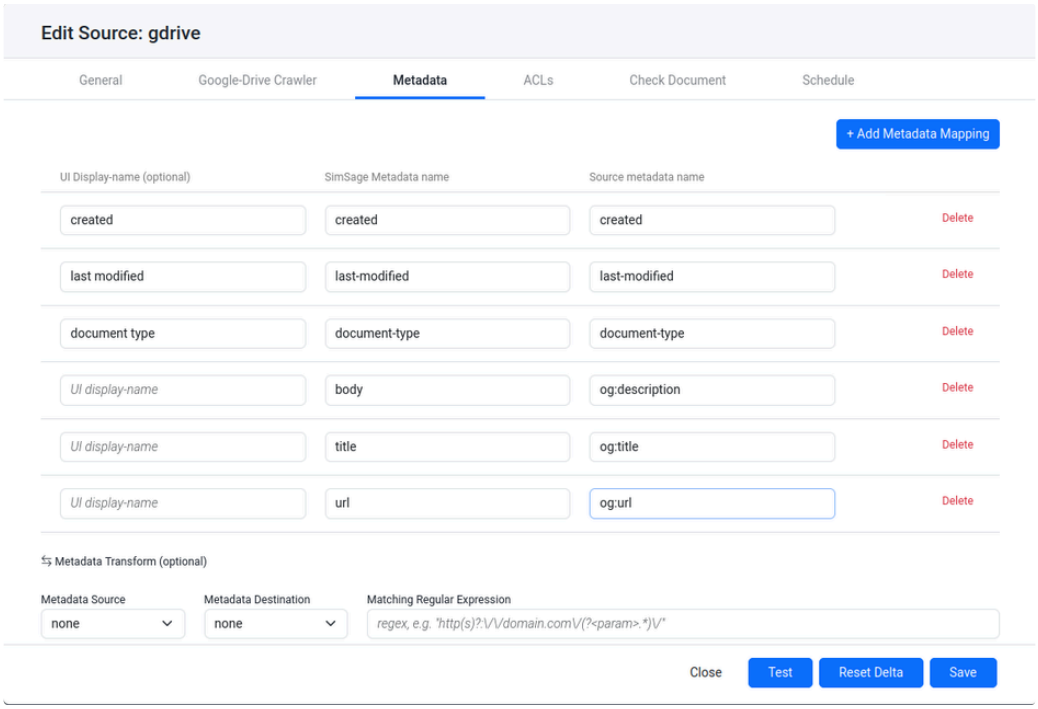
SimSage has a series of special names that can be used to map the main items used in SimSage. You can map any item, even if SimSage doesn't directly map it to one of its main items. Items that don't map are just mapped as metadata and can be found through metadata searches.

The main items understood by SimSage are

SimSage item	Description
url	the primary key of a SimSage item. If the primary key is an http or https reference the item can be directly access from the remote system.
title	The title of a SimSage record, important in search as it gives each record a score boost
body	The main content of a SimSage record, the default text that is searchable
author	The author of a SimSage record
created	The created Date-time (must be a date as a minimum, various formats supported) of a SimSage record
last-modified	The last-modified Date-time (must be a date as a minimum, various formats supported) of a SimSage record

Below is an example of SimSage mapping HTML meta items called "og:dscription", "og:title", and "og:url" into its body, title, and url fields respectively. Note that the type of the asset will be changed from HTML to TEXT if the "body" text is no longer HTML. This applies to both

“<meta name=…>” and “<meta property=…>” data found in your document.



Any other metadata fields (whether included or not) are mapped into the metadata of each SimSage record.

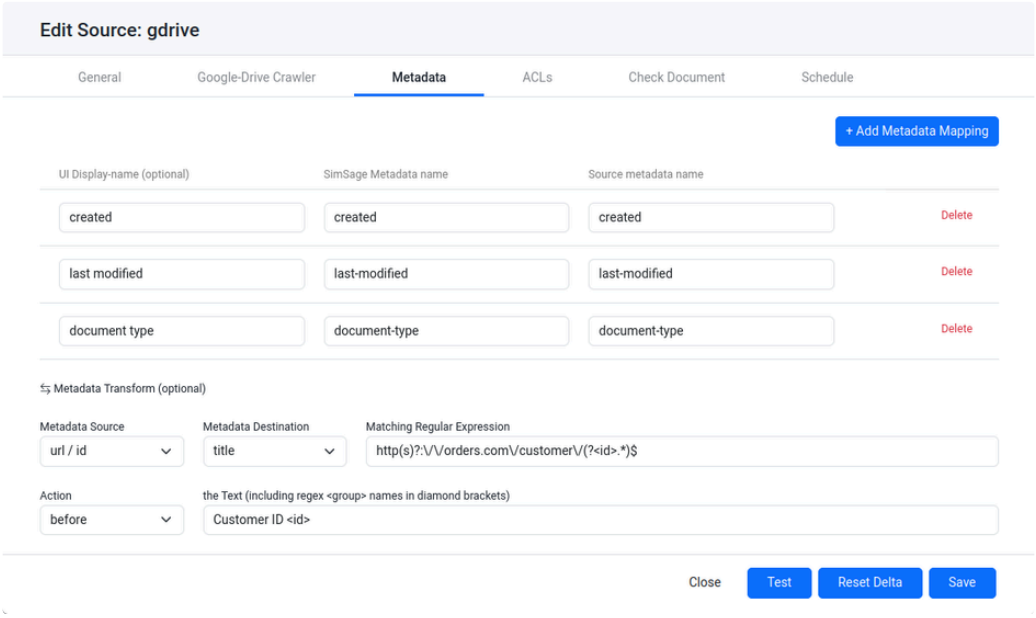
Metadata Transforms

SimSage has the ability to transform / enhance metadata using regular expression matching. Imagine there was an important ID number part of your URL that you want to insert into every title of your document.

This is what metadata transforms are for.

Metadata transform field	Description
Metadata Source	what metadata field to apply a regular expression to for processing / extracting information from.
Metadata Destination	if successful, what metadata field to apply the new data to
Action	How to apply the new data, one of {after, before, replace}. After places the result after the existing field’s metadata. Before places the result before the existing field’s metadata. Replace ‘replaces’ any data in the destination metadata field.
the Text	What text to create. This can be any text, including <name> (diamond bracket name) fields as used in the regular expression.
Matching Regular Expression	the regular expression to apply to each metadata source field.

Here is an example. We have an ID number that doesn't occur in the title of our documents but the URL of our documents. We would like to add this ID with a "Customer ID" prefix to each title of matching documents. The ID number only occurs in certain URLs of our "orders.com" website as part of the second path entry of an order URL (e.g. https://orders.com/customer/123). We set up the values as shown below.



Any data ingested by this source will now change the title of each record that matches the regular expression applied to it's URL field. Records that do not match the regular expression are not modified. Take special care with expressions like this to allow for ending "/" extra slashes where applicable.

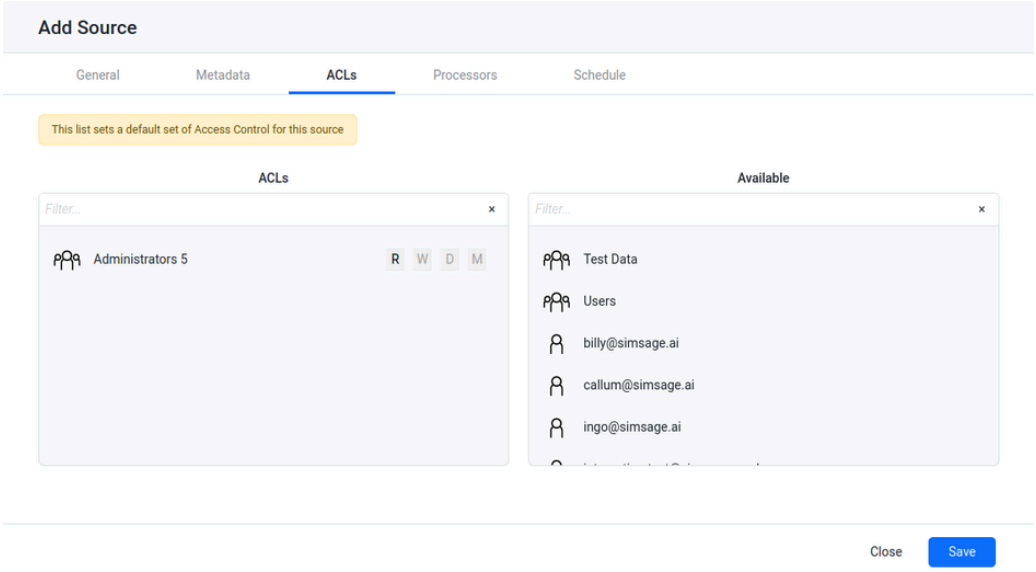
Changing the Metadata Transform does require any existing data (already ingested data) to be re-processed.

The ACLs Tab

In this tab we can customise the security based on existing SimSage Users and Groups.

NB: Users and Groups are automatically imported from external systems as they are crawled, where available.

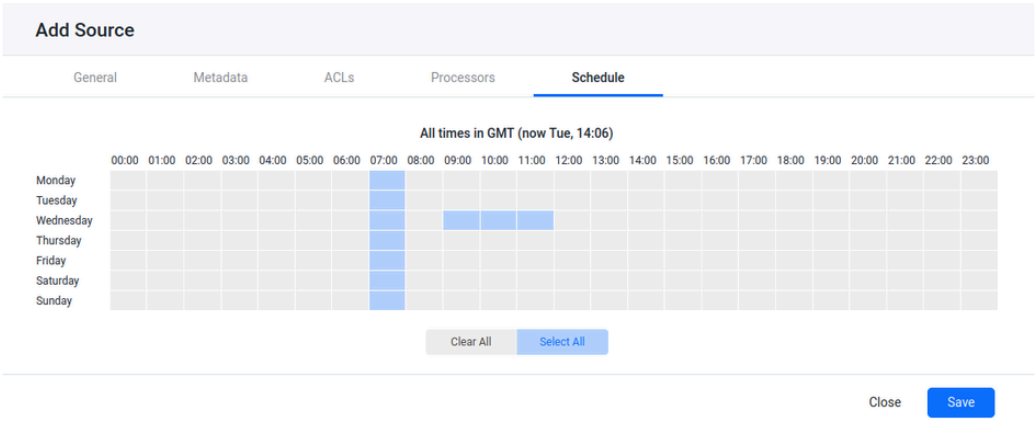
In the below screenshot example we have selected the SimSage default User group as having access. Note, if you have opted to use the "allow anonymous access to these files" in the General source configuration, these ACLs settings will be overridden. *SimSage always recommends setting up a group initially in case you change your mind at any point.*



The Schedule Tab

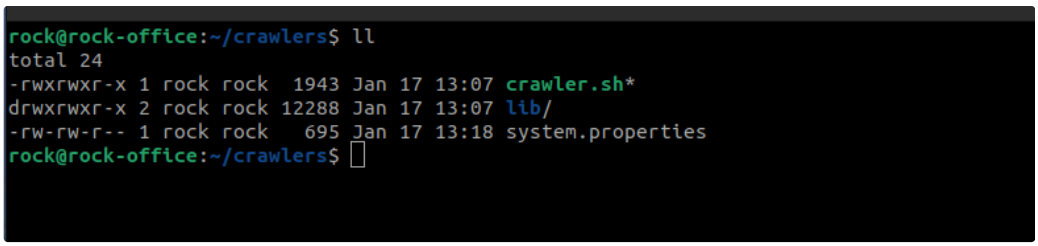
Here you can specify when you would like the crawler to be allowed access to the SimSage platform.

This is done by setting desired time frames to active. In the example provided we used the “select all” button to allow SimSage to always be available to your crawler.



External Crawler Configuration

The external crawler is a small Java program, provided by SimSage for the version of your platform.



The software provided consists of:

- A *lib* folder with all the java libraries required to run the crawler.

- A *crawler.sh* executable shell file for running the crawler,
- and a *system.properties* file that needs to be edited to match your platform.

The current version of the external crawler file is 7.6.3 and can be downloaded here.

Once you have the external docker file, you will need to populate the values in the docker compose file as below. A docker-compose file might look like this:

```
version: "3.9"
services:
  web:
    build: .
    deploy:
      resources:
        limits:
          cpus: "1"
          memory: 512M
        reservations:
          cpus: "1"
          memory: 512M
    environment:
    - source_id=1
    - crawler_type=web
    - organisation_id=c276f883-e0c8-43ae-9119-df8b7df9c574
    - kb_id=46ff0c75-7938-492c-ab50-442496f5de51
    - sid=48f9a7f5-6d6b-9766-a232-6ef59eae7cae
    - simsage_endpoint=http://192.168.1.10:8080/api
```

The environment settings in bold must have the correct values supplied by SimSage and can be found via the SimSage Admin platform.

The source_id should be set to the id of the internal crawler source you created in the above steps.