Design Document

Archivist

Members:	Scott Selinger, Rob Edwards, Tyler Boswell, Charles Gilliam, Dung Mai
Advisor:	Rosina Weber
Stakeholder:	Lori Richards

Revision

Name	Date	Reason	Version
Scott Selinger, Rob Edwards, Tyler Boswell, Charles Gilliam, Dung Mai	November 8, 2016	Initial Draft	1.0
Scott Selinger, Charles Gilliam, Rob Edwards	March 24, 2017	Add chrome extension diagram and update advanced search diagram	1.1
Scott Selinger, Charles Gilliam, Rob Edwards, Dung Mai, Tyler Boswell	May 18, 2017	Reflect changes during the development process	1.2

Table of Contents

1. Introduction

1.1. Purpose

This document provides the design specifications for the Archivist web application interface and chrome extension. It will describe components of the backend architecture and frontend architecture and how they should behave and function.

1.2. Target Platform

The application will guarantee support on the Google Chrome browser version 53 or later. The chrome extension will only run on the Chrome browser.

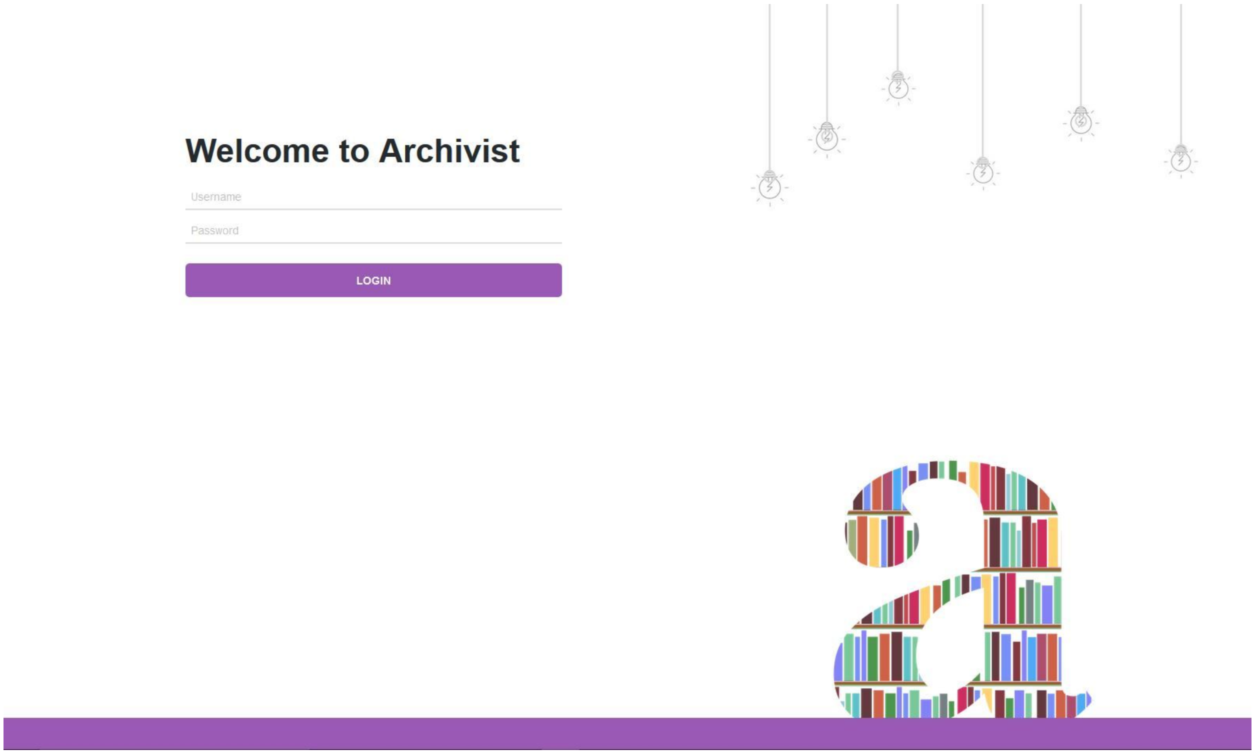
2. Interface

2.1. Login Screen

Purpose: This screen will allow a user to gain access into Archivist.

Details: The login screen will always be the first screen a user sees when they visit Archivist if that user has not logged in before or has been logged out of Archivist.

Navigation & Interaction: A user will need to enter their username and password in the two text fields under “Welcome to Archivist” and press the “Login” button. If authentication fails, the user will remain on this screen, otherwise the main screen will be displayed to the user (see section 2.2 for main screen.)



Welcome to Archivist

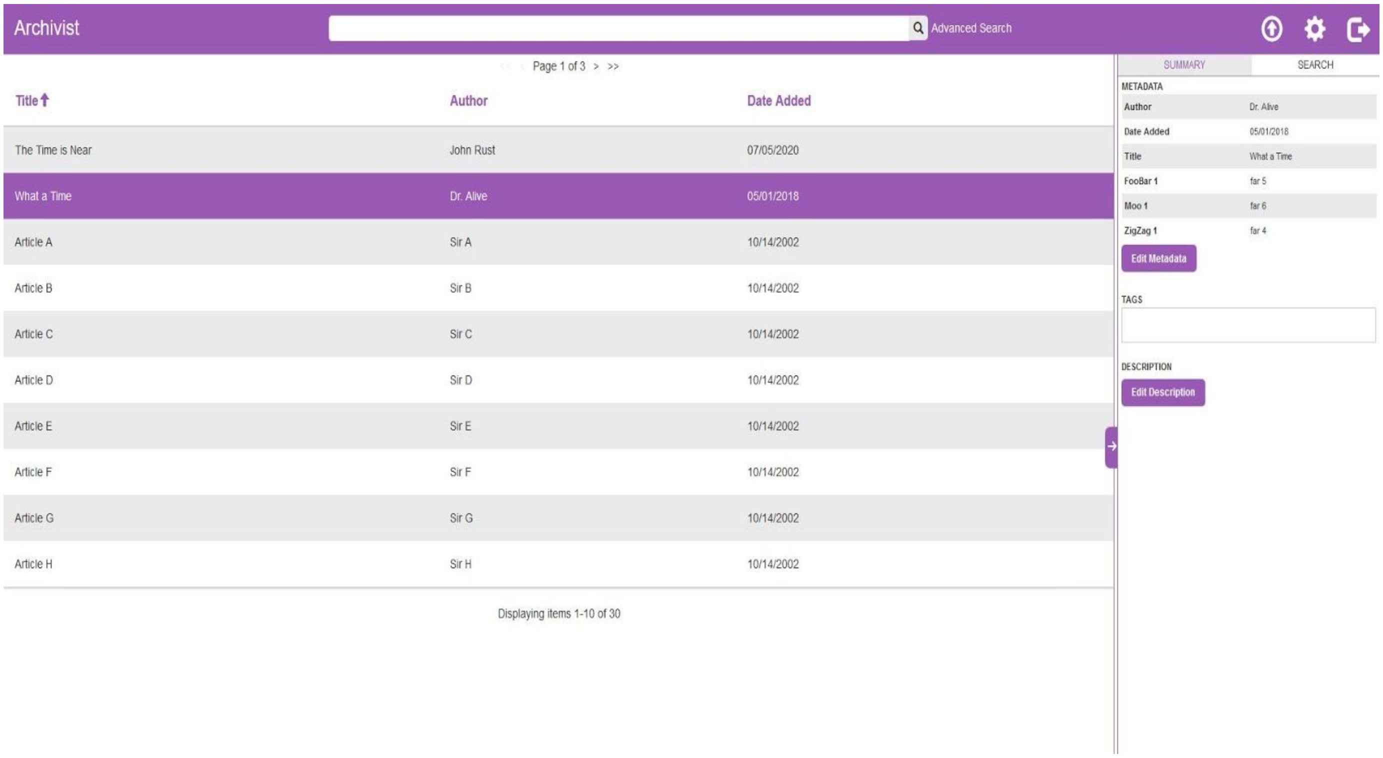
Figure 1. Login Screen

2.2. Main Screen

Purpose: This screen will allow users to view a list of all the files that were uploaded using the upload screen (section 2.5) or through the chrome extension (section 2.8).

Details: This screen will show if the user is logged in, otherwise they will be redirected to the login screen. The default number of files that will be displayed on the screen is 10 (this number can be changed in the settings screen, section 2.7). There is no limit to how many files can be displayed on the list. The default sort for the list is by the order they were added. The last file added would be at the end of the list.

Navigation & Interaction: A user can single click on a file to view additional details about that file. Single clicking on a file will extend the side panel which is where the additional details about that file will be displayed. Double clicking on a file will redirect the user to the view file screen (section 2.6) for that particular file. The user will be allowed to sort the list of files by clicking on the title of each column such as “Title”, “Date Added”, and “Author.” The user can click the arrow “>” to view the next 10 files (by default or the number of files to display that were set in the settings screen, section 2.7). The user can click the double arrow “>>” to skip to the first or last page. On this screen the user can access other screens using the top action bar, to see what icon leads to which screen and what each feature on the action bar does, see section 2.3 for full details.



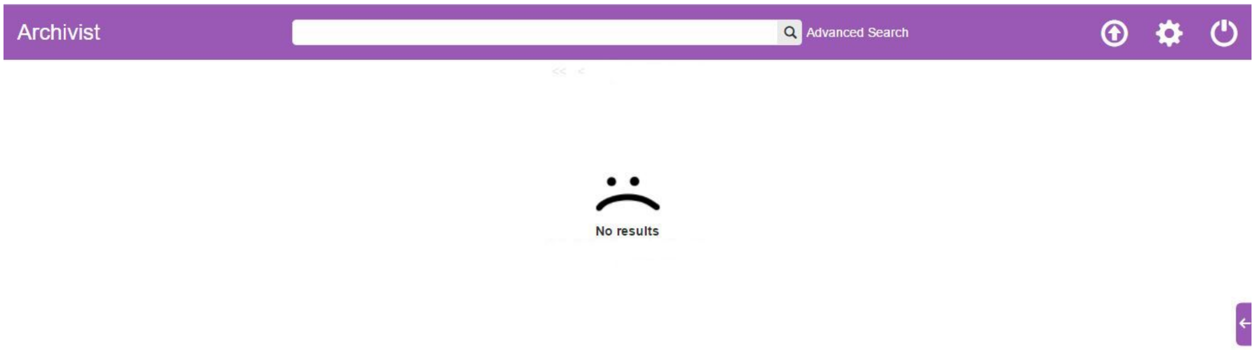
Title	Author	Date Added
The Time is Near	John Rust	07/05/2020
What a Time	Dr. Alive	05/01/2018
Article A	Sir A	10/14/2002
Article B	Sir B	10/14/2002
Article C	Sir C	10/14/2002
Article D	Sir D	10/14/2002
Article E	Sir E	10/14/2002
Article F	Sir F	10/14/2002
Article G	Sir G	10/14/2002
Article H	Sir H	10/14/2002

Displaying items 1-10 of 30

METADATA

Author	Dr. Alive
Date Added	05/01/2018
Title	What a Time
FooBar 1	far 5
Moo 1	far 6
ZigZag 1	far 4

Figure 2. Main screen with single click on a file and side panel showing



No results

Figure 3. Main screen without any files in the system

2.3. Action Bar

Purpose: The action bar will allow user to navigate to other screens while logged in.

Details: This bar will always be available to the user at the top of their screen except on the login screen (section 2.1). The icons are displayed based on the context of the current page, see figure 4, 5, and 6 below for details of what icon will be displayed on what screen.

Navigation & Interaction: From left to right on the action bar, clicking the word “Archivist” (figure 4.1) will always take the user to the main screen (section 2.2). The search box (figure 4.2) will allow a user to perform searches by typing in the search box. Clicking the “Advanced Search” (figure 4.2) will open up the side panel search tab (section 2.4). Clicking on the upload icon (figure 4.3) will take the user to the upload screen. Clicking the settings icon (figure 4.4) will take the user to the settings screen. Clicking the logout icon (figure 4.5) will logout a user and take the user to the login screen (section 2.1). The back button (figure 5.1) , in figure 5 and 6 below, takes the user back to the previous screen that they were on.

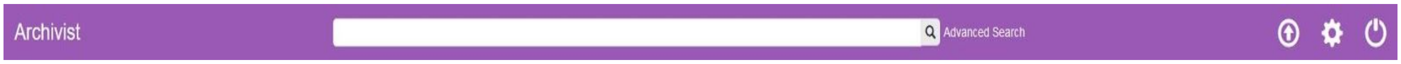


Figure 4. Action bar on the main screen and view file screen

Figure 4.1 Actionbar home button

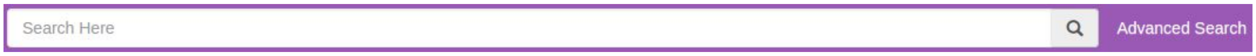


Figure 4.2 Search bar and advanced search button

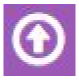


Figure 4.3 Upload file button

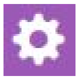


Figure 4.4 Settings button

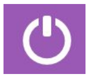


Figure 4.5 Logout button

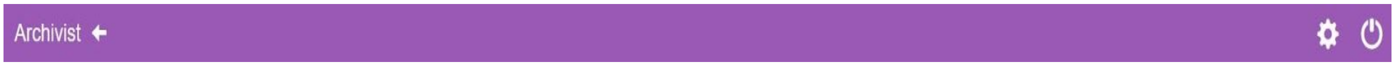


Figure 5. Action bar upload screen

Figure 5.1 Back button

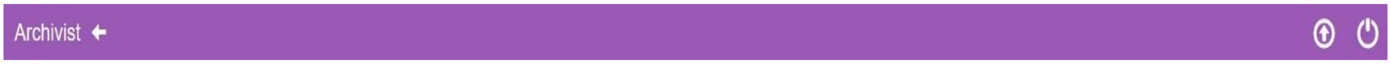


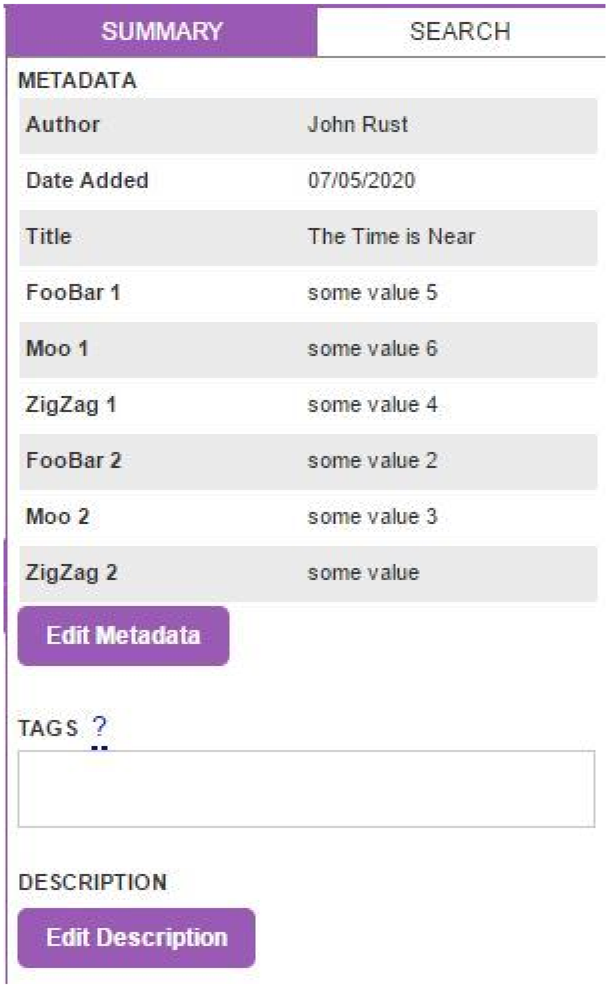
Figure 6. Action bar on settings screen

2.4. Side Panel

Purpose: The side panel will allow a user to view additional information about a specific file when a user single clicks on a file on the main screen (section 2.2). It will allow a user to perform advanced searches. The side panel will also allow a user to edit metadata, description, and tags on specific file.

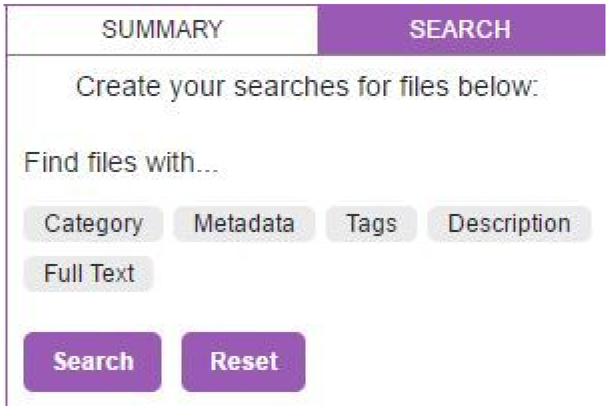
Details: The side panel is only available on the main screen (section 2.2) and the view item screen (section 2.6). This panel will always be on the right side of the screen under the action bar (section 2.3). The initial width of the panel is ⅕ of the whole screen. The width of the panel can be resized. The maximum width of the side panel is ⅘ of the whole screen. The side panel width will expand into the main screen (section 2.2), so other elements on the screen will take up less space so that the components never overlap. The side panel can be collapsed by clicking on the narrow bar containing the arrow icon to the left of the sidebar. The side panel can be expanded by clicking the arrow on the right side of the screen. When expanded, the width of the panel will go back to how it was before it was collapsed. For example, if the width of the side panel was resized to be ⅗ of the screen, after collapsing, when expanded, the width should still be ⅗.

The side panel will contain two tabs. One is summary (see figure 7 below) and one is a search tab (see figure 8 below). The summary tab will display the metadata, tags, and description of a particular file that was selected on the main screen (section 2.2). If no file was selected, the summary tab will display nothing but “Please select an item on the table to the left.” The search tab will display the groups of searches a user can perform along with the search and reset button. The groups are “categories”, “metadata”, “tags”, “description”, and “full text.”



Author	John Rust
Date Added	07/05/2020
Title	The Time is Near
FooBar 1	some value 5
Moo 1	some value 6
ZigZag 1	some value 4
FooBar 2	some value 2
Moo 2	some value 3
ZigZag 2	some value

Figure 7. Summary tab on side panel with no description or tags



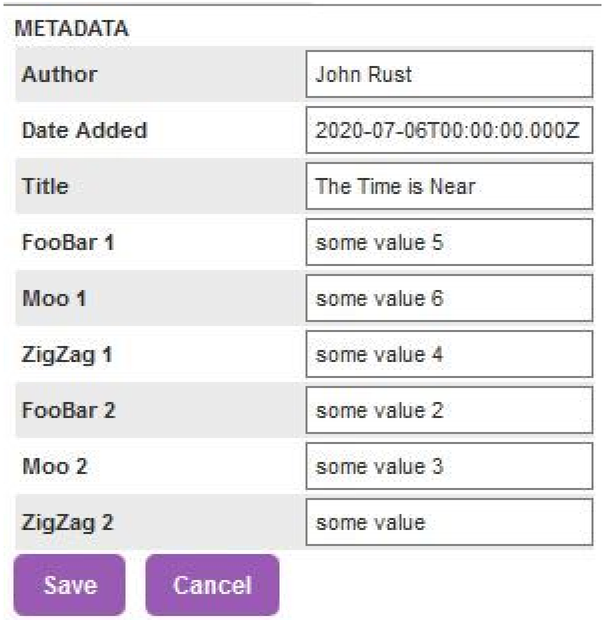
Create your searches for files below:

Find files with...

Figure 8. Default search tab in the side panel

Navigation & Interaction: Clicking the arrow will expand the side panel. Clicking it again will collapse the side panel. While on the summary tab and an item on the main screen is selected, a user can click the "Edit Metadata." This will turn all metadata values into different editable

HTML5 input types depending on the metadata field type. For example, date types will use a datepicker, and numeric types will only allow numeric values. Clicking save will save all the current values in those input fields. See figure 9 below.

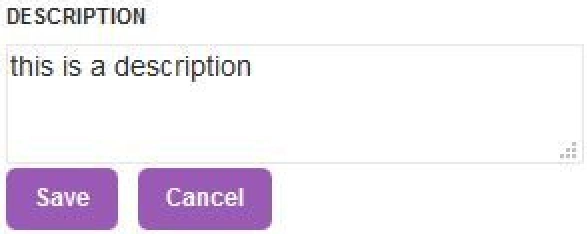


METADATA

Author	John Rust
Date Added	2020-07-06T00:00:00.000Z
Title	The Time is Near
FooBar 1	some value 5
Moo 1	some value 6
ZigZag 1	some value 4
FooBar 2	some value 2
Moo 2	some value 3
ZigZag 2	some value

Figure 9. Metadata section after clicking the “Edit Metadata” button

Clicking the “Edit Description” button will open up a textbox that allow a user to type and save the current value in the textbox by clicking the save button. This description accepts full Markdown support. See figure 10 below.



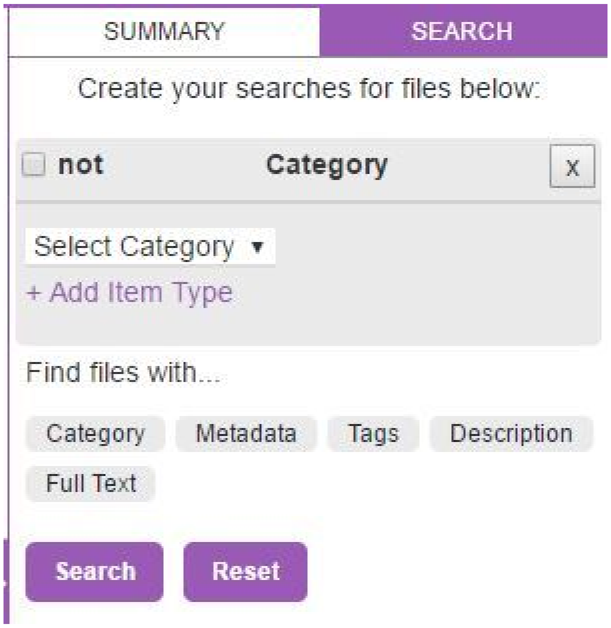
DESCRIPTION

this is a description

Figure 10. Description textbox after clicking the “Edit Description” button

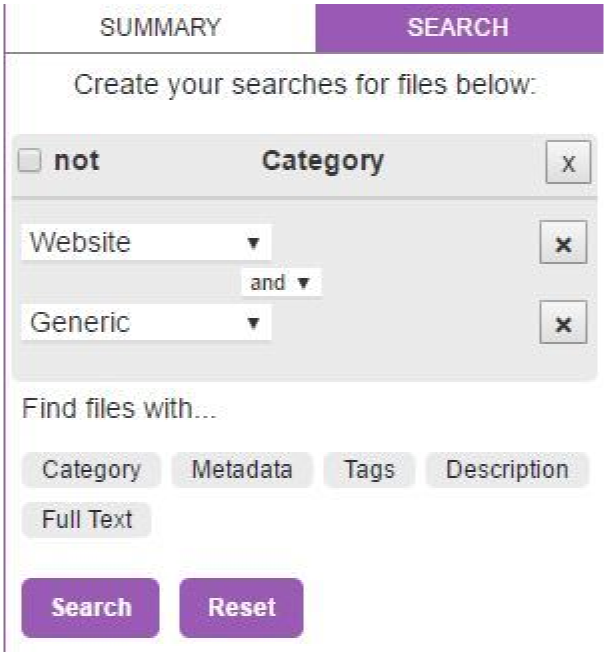
While on the search tab, clicking on any of the groups such as “categories”, “metadata”, “tags”, “description”, and “full text” will show a new section to search within the group that was clicked. Clicking the “Reset” button will reset the list on the main screen (section 2.2) to its default state.

Clicking on the “category” word will show a new section in the tab that allow users to pick a category from the drop down. The number of categories the user can choose from depends on the user's custom created categories using the settings screen (section 2.7), otherwise, “generic” and “website” will always be the default options. Users can also pick more than one category by clicking the link “+Add Item Type.” Clicking the search button will perform a search using the category or categories picked by the user. Users can also choose to select “and” or “or” between categories. “And” will search for files that are in all categories picked by the user. “Or” will search for files that contain at least 1 of the category picked by the user. Clicking the “not” checkbox on top left will allow the user to perform search for files that are not in the category picked by the user when clicking the search button. See figure 11 and 12 below.



Create your searches for files below:

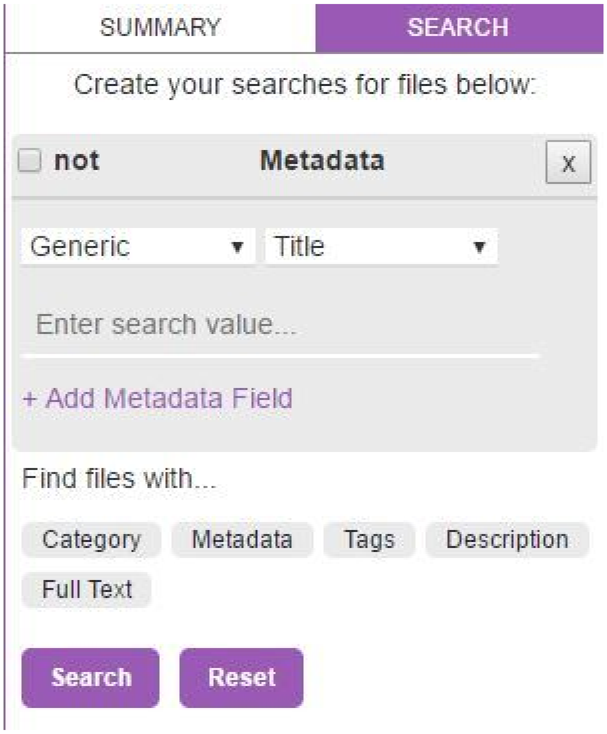
Figure 11. Default category section after clicking the group “Category”



Create your searches for files below:

Figure 12. Category section after clicking “+Add Item Type” with two maximum categories

Clicking the “Metadata” group will show a new section. Users can select the category name then select the field name that relate to the category in the drop downs that they would like to search for. Users then will need to enter the value that they are looking for in the text box. For example, if a user selects “website” and “url” in the drop downs, and they would like to search for a file that was saved into the system from www.amazon.com, the user would enter “amazon” in the text box. This would search for all files that have a URL containing the word “amazon.” Users can also search for files with multiple metadata fields by clicking the “+Add Metadata Field.” Users can choose “and”/”or” option after clicking the “+Add Metadata Field”. Selecting “and” will search for files that containing every field, and selecting “or” will search for files containing at least one of the fields.



Create your searches for files below:

Figure 13. Metadata section with “Generic” and “Title” selected

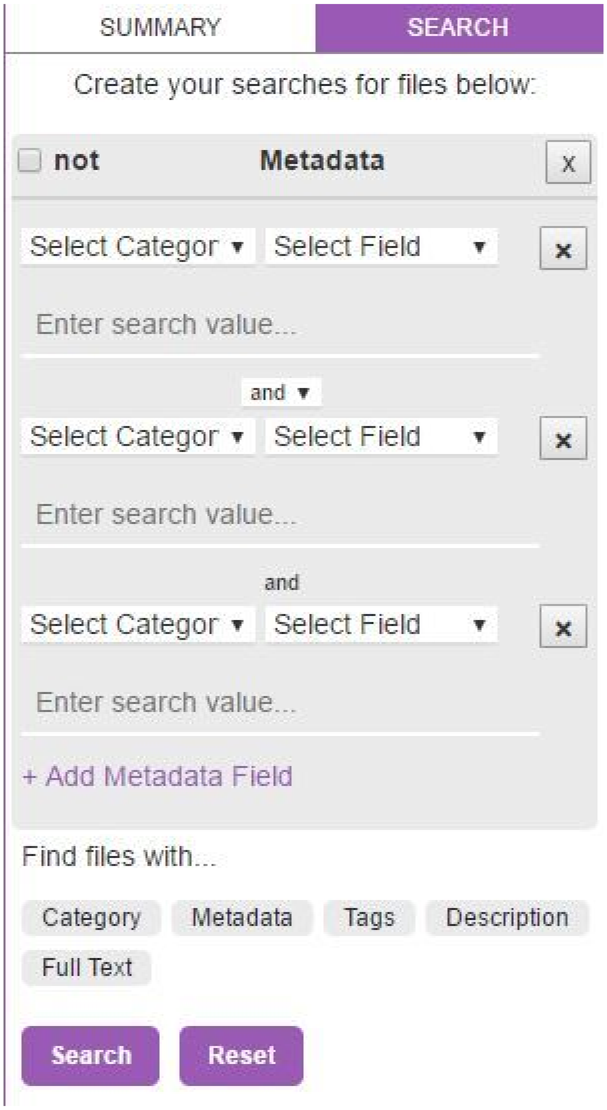
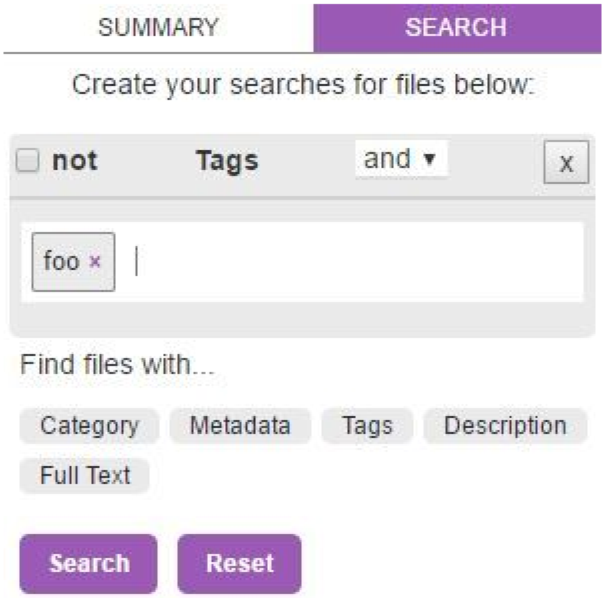


Figure 14. Metadata section after clicking “+Add Metadata Field” three times

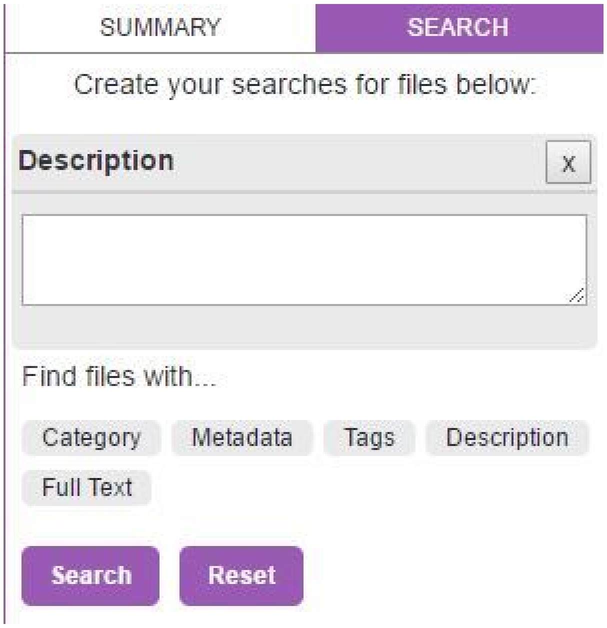
Clicking the “Tags” group will show a new section with a text box for users to enter tags that they would like to search. Typing in the words in the text box and pressing the “Enter” key will register a tag. After entering in all the tags users want to search, clicking the “Search” button will search for all files in the system that have been tagged with the tags user entered in the text box.



Create your searches for files below:

Figure 15. Tags section with the word “foo” after pressing “Enter” key

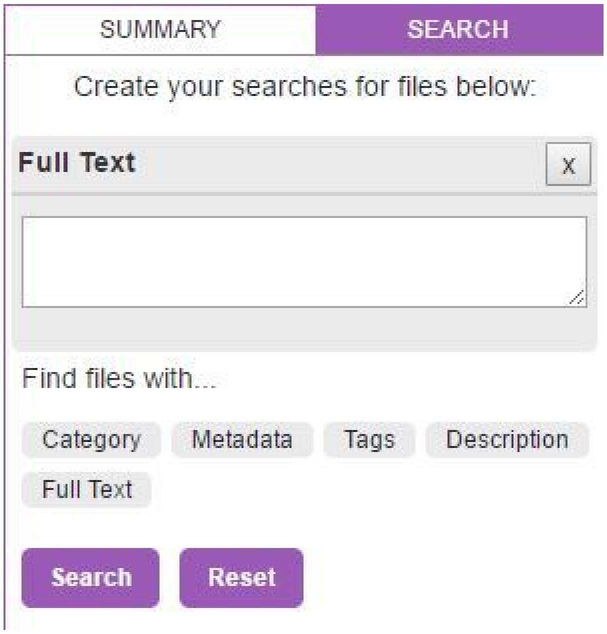
Clicking the “Description” group will show a new section with a text box. Users can enter any words in the text box that they would like to search for files that contain those words in their description. A description of a file is text that were entered by the user or through the Chrome extension that describes what the file is about.



Create your searches for files below:

Figure 16. Description section

Clicking the “Full Text” group will show a new section with a text box. Users can enter any words in the text box. Clicking the “Search” button after entering words in the text box will search for any files that contain those words in the file's content. For example, if a user saved a website that contains the word “job” on the website that was saved and the user entered “job” in the full text search box, that file will show up on the list on the main screen.

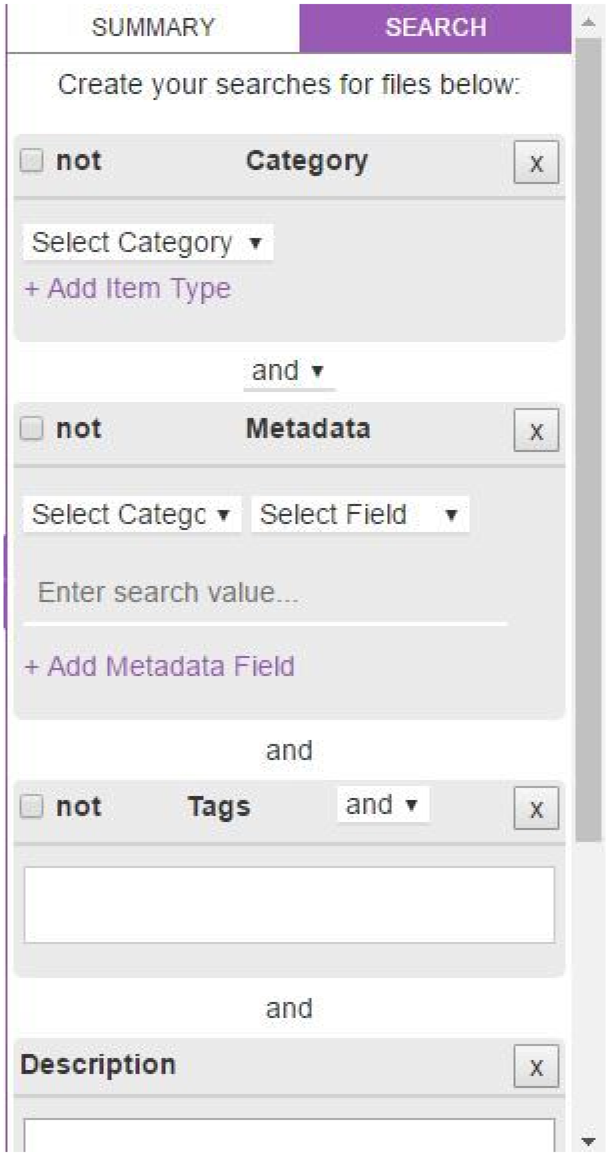


Create your searches for files below:

Find files with...

Figure 17. Full Text section

Users can also click on multiple groups to show multiple sections. This way users can search files using multiple groups. Any files that contain content that are in all of the groups will show up on the list in the main screen. Users can also check the “not” checkbox on each of the group. This will search for files and show these files on the list on the main screen that don’t belong in these groups that user checked.



Create your searches for files below:

Figure 18. Search tab in side panel with multiple groups clicked

2.5. Upload File Screen

Purpose: This screen allow a user to upload a file from their computer to the Archivist system.

Details: A user can only upload either a PDF or HTML file. The user is required to enter a title in the generic group and the user is required to select a file before uploading. The file that is uploaded through this screen will appear on the list on the main screen (section 2.2). There will be a description text box and a tag box. There will be two default metadata groups called "generic" and "website" (see figure 19 below for more details). All metadata groups will be checkboxes with the exception of the "generic" group. By checking the checkbox, textboxes that represent fields within that metadata group will appear. See figure 20 below. Extra metadata groups can be created in the settings screen (see section 2.7). Any custom metadata groups that are created in the settings screen will also appear on this screen.

Navigation & Interaction: A user can choose a file from their computer by clicking the upload icon on the left of the screen, this will bring up the standard file selection for the user's operating system window. The user will be required to enter in a title before being able to submit the file. The user can enter a description and tags. The user can check a checkbox to view the metadata fields for that particular metadata group. Then the user can enter in any values for each field. Clicking the "upload" button will upload the selected file, along with all the values the user has entered on this screen to the database. After clicking upload, the user will still be on this screen.

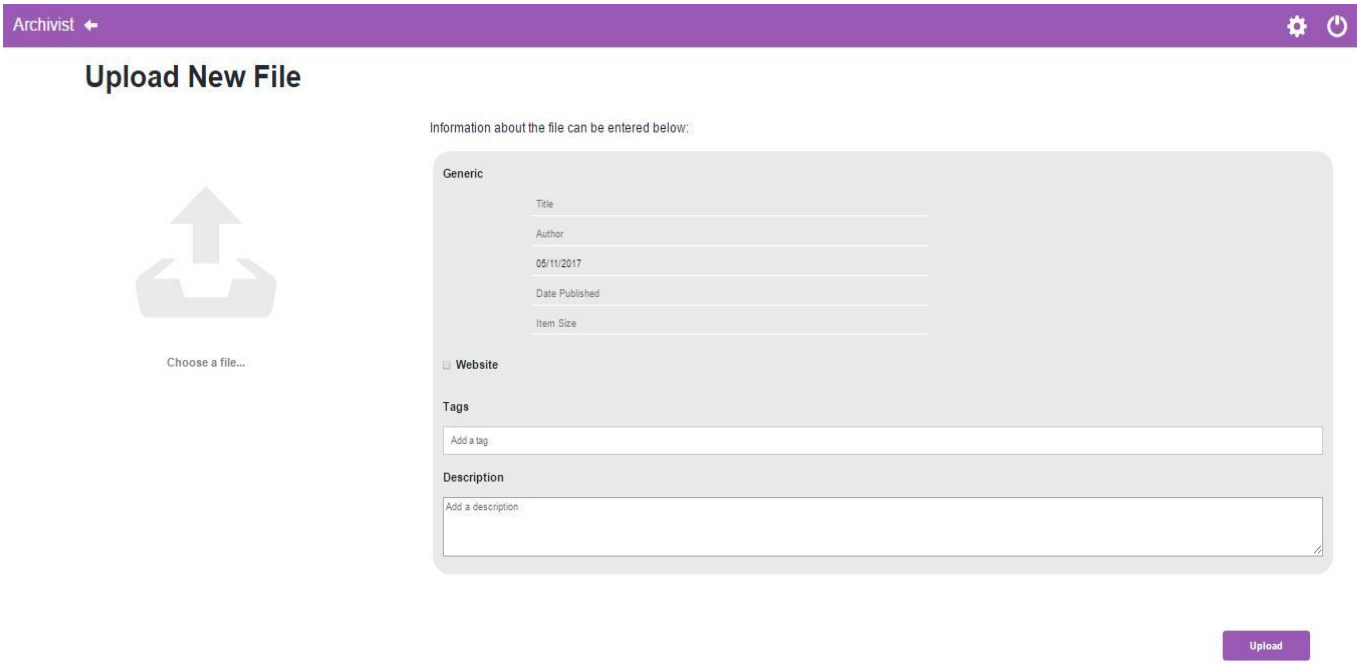
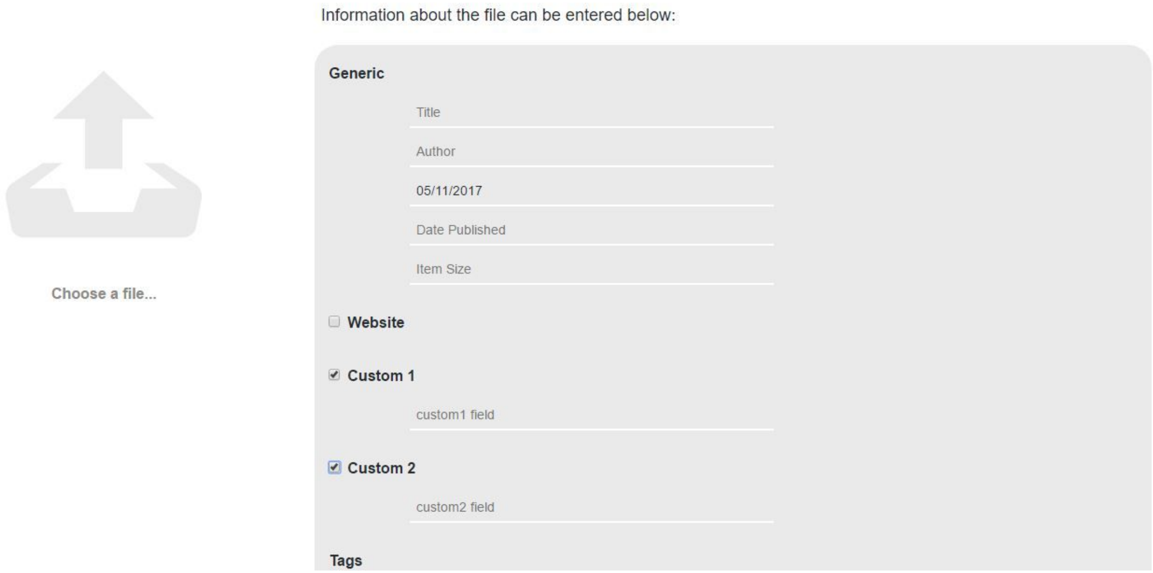


Figure 19. Default Upload Screen


Information about the file can be entered below:

Choose a file...

Generic

Title

Author

05/11/2017

Date Published

Item Size

☐ **Website**

☑ **Custom 1**

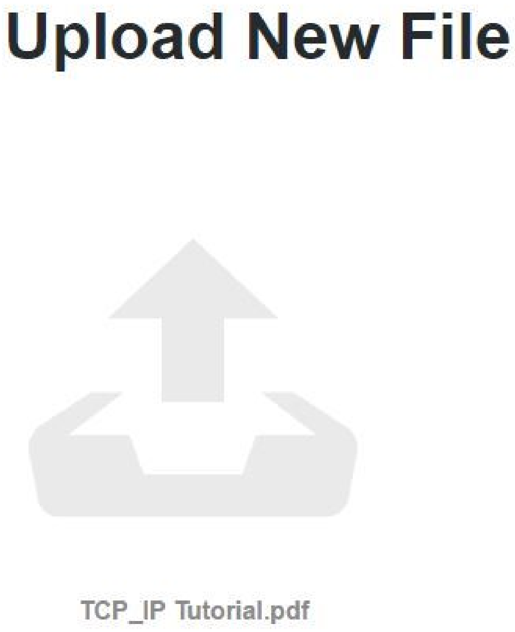
custom1 field

☑ **Custom 2**

custom2 field

Tags

Figure 20. Upload Screen with Custom Metadata Selected


Upload New File

TCP_IP Tutorial.pdf

Figure 21. Upload Icon Displaying The File Selected

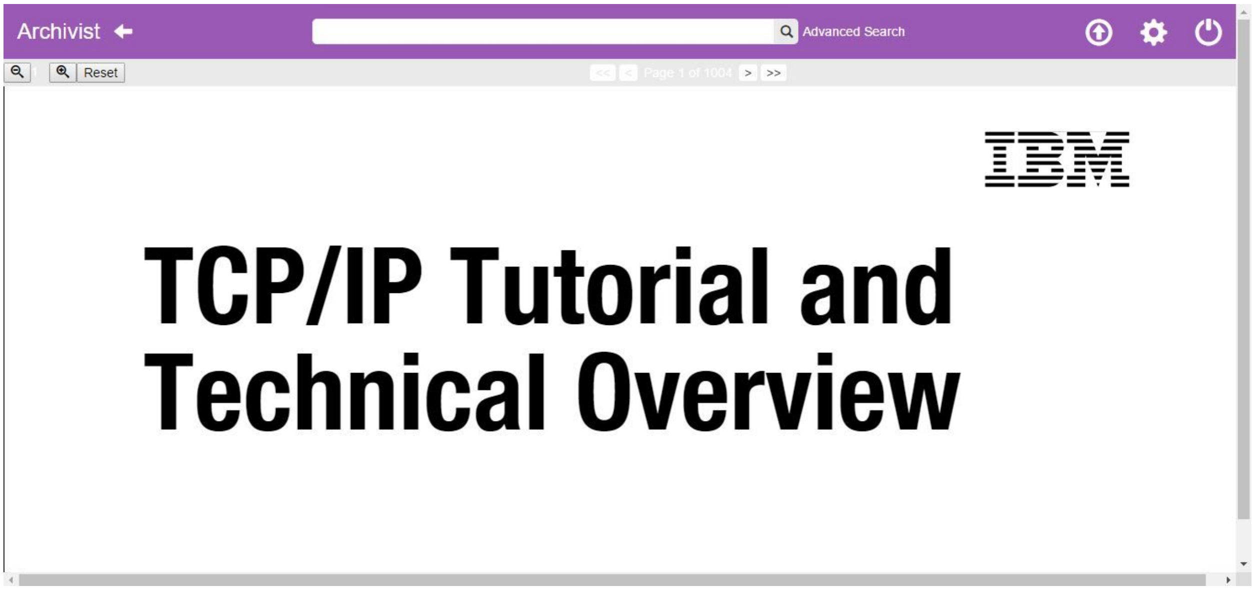
2.6. View File Screen

Purpose: This screen will allow users to view the contents of the file they have uploaded.

Details: The user will be redirected to this screen upon double clicking a file on the main screen. The file will be represented as close to the original viewing experience as possible. If the file was a PDF and was originally spread across multiple pages, then it will be spread across multiple pages in the view file screen as well, with arrows located at the top of the screen to navigate through those pages. The current page and total page count will also be displayed at the top of the screen with the option to change the current page which will allow the user to more easily switch from page to page.

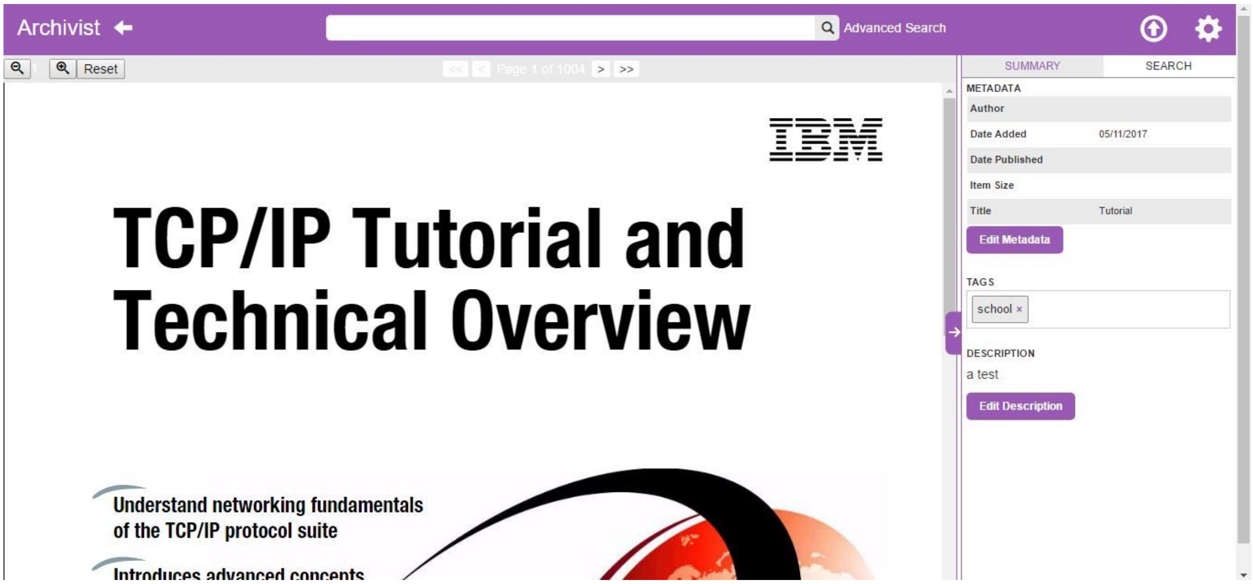
The side panel (section 2.4) will be on this screen. The size of the file content will adjust to fill the screen base on the width of the side panel. If the original viewing experience was on one single page, then the same will be true for the view file screen. If the file is too long to be displayed on a single page, then there will be a scroll bar along the right side of the screen in between the document and the side panel. There will be a zoom in, zoom out, and reset buttons for PDFs to allow zooming in the file content, zooming out the file content, or reset back to default zoom, which is 100%.

Navigation & Interaction: A user can expand or collapse the side panel (section 2.4) by clicking the arrow on the right side of the screen. The user can click the arrow “>” or “>>” to navigate through the content of the file. The user can click the plus magnifying glass to zoom in or the minus magnifying glass to zoom out or the reset button to reset the view of the file content. The user can navigate to all other screens using the action bar (see section 2.3).



TCP/IP Tutorial and Technical Overview

Figure 22. Viewing a PDF file at default zoom without side panel


Archivist
Advanced Search
Reset
<< < Page 1 of 1004 > >>
SUMMARY
SEARCH
IBM
TCP/IP Tutorial and Technical Overview
Understand networking fundamentals of the TCP/IP protocol suite
Introduces advanced concepts
Edit Metadata
Edit Description

METADATA

Author	
Date Added	05/11/2017
Date Published	
Item Size	
Title	Tutorial

TAGS

school ×

DESCRIPTION

a test

Figure 23. Viewing a PDF file at default zoom with side panel opened

2.7. Settings Screen

Purpose: This screen will allow users to create and edit metadata groups, view how many files they have uploaded, the amount of disk space these files take up, and change the number of files to display on the main screen (section 2.2.)

Details: “Usage” will contain the number files in the database and the amount of disk space these files take up.

“File Category” will allow users to edit and add more metadata groups to the upload screen. The two default ones will always be display are “generic” and “website.” These two metadata groups cannot be deleted or edited.

“File List” will allow a user to specify how many files will be display on the main screen (section 2.2).

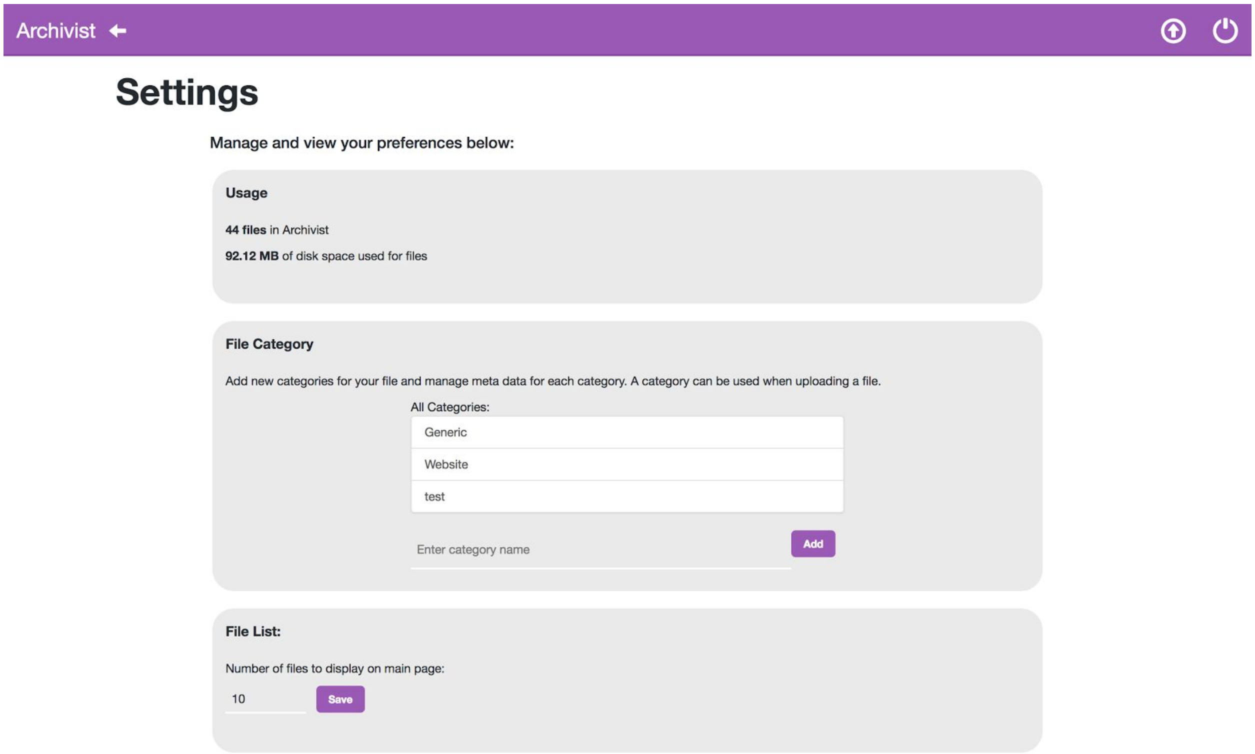
Navigation & Interaction: Users can add new categories and metadata fields in the “File Category” section. These new categories and metadata fields will show up in the upload screen (section 2.5) and the search tab in the side panel (section 2.4) for the category group and metadata group. Two default groups, "Generic" and "Website", are always present and cannot be removed.

To add a new group, users can type in the name in the "Enter category name" text input and click "Add" button. The new category name will show up under the "All Categories" list.

To view the metadata fields or edit them, click on a category on the list to see more details (see figure 25). After clicking on a category on the list, users can see a list of fields (if there are any) for that particular category.

To add a new metadata field to the category, enter the name of the metadata field in the "enter field name" text box and select the field type using the drop down and click "Add" button next to it to add a new metadata field. The new metadata field will show up on “All Fields” list (see figure 28).

To delete a metadata field, click the “X” button next to the metadata (figure 28) a user would want to delete. “Generic” and “Website” metadata fields will not have the “X” since these two categories cannot be edited by the users.



Settings

Manage and view your preferences below:

Usage

44 files in Archivist

92.12 MB of disk space used for files

File Category

Add new categories for your file and manage meta data for each category. A category can be used when uploading a file.

All Categories:

Generic

Website

test

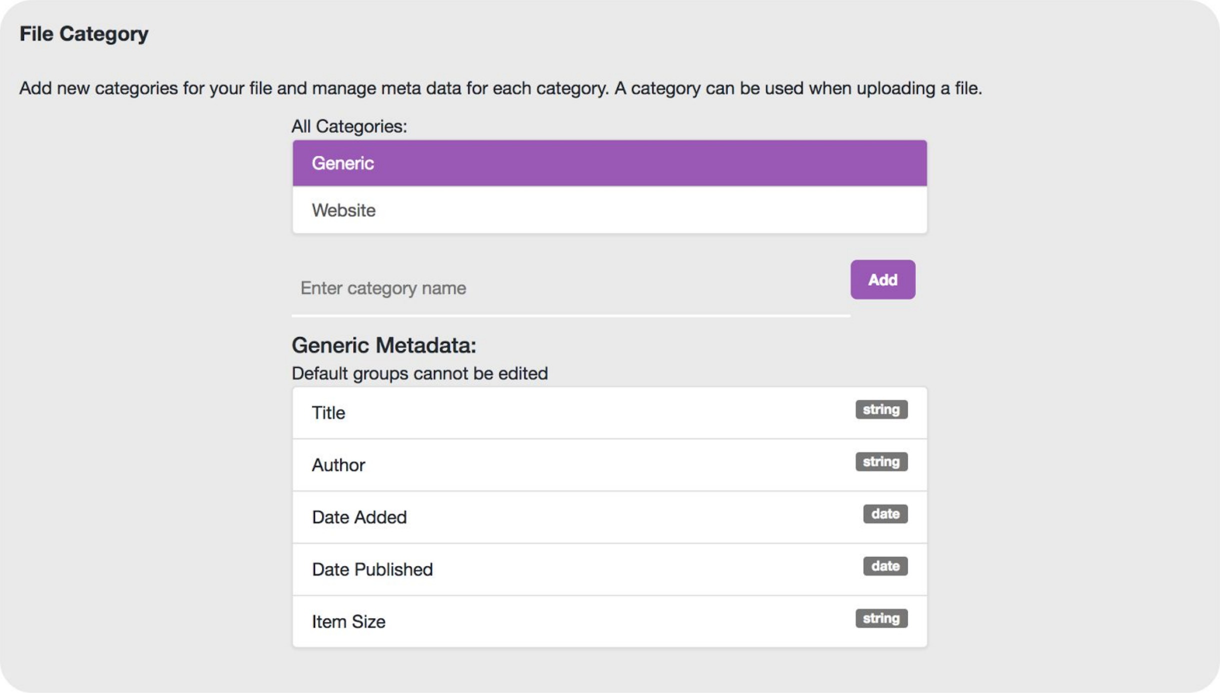
Enter category name

File List:

Number of files to display on main page:

10

Figure 24. Default Settings Screen



File Category

Add new categories for your file and manage meta data for each category. A category can be used when uploading a file.

All Categories:

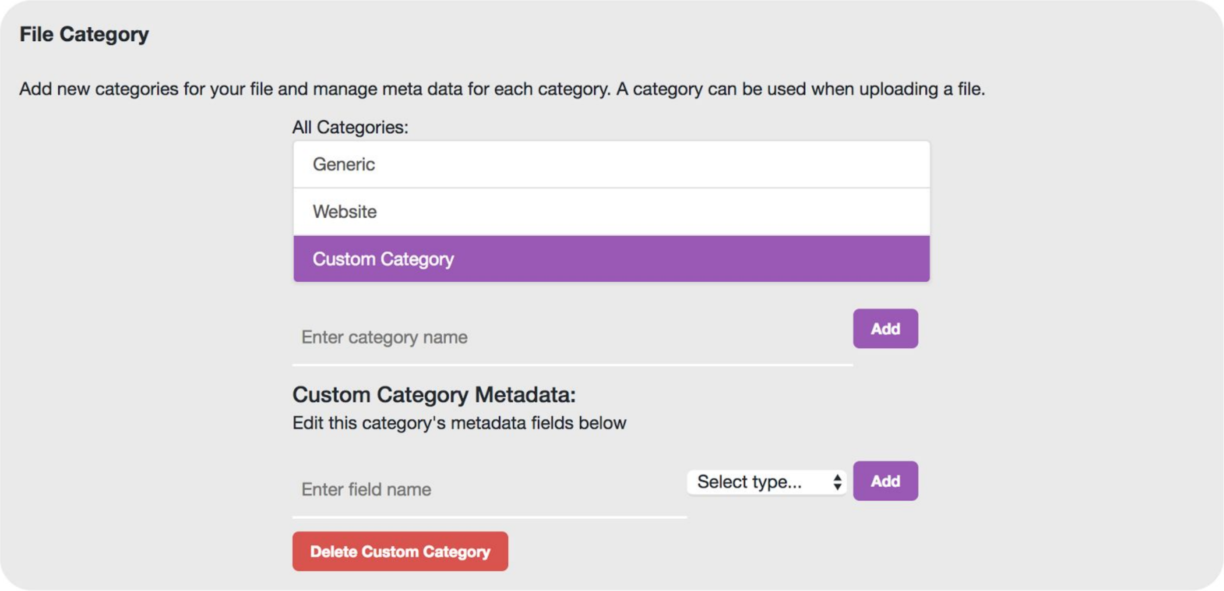
- Generic
- Website

Generic Metadata:

Default groups cannot be edited

Title	string
Author	string
Date Added	date
Date Published	date
Item Size	string

Figure 25. File Category section when “Generic” is selected. All fields within the selected category are displayed below



File Category

Add new categories for your file and manage meta data for each category. A category can be used when uploading a file.

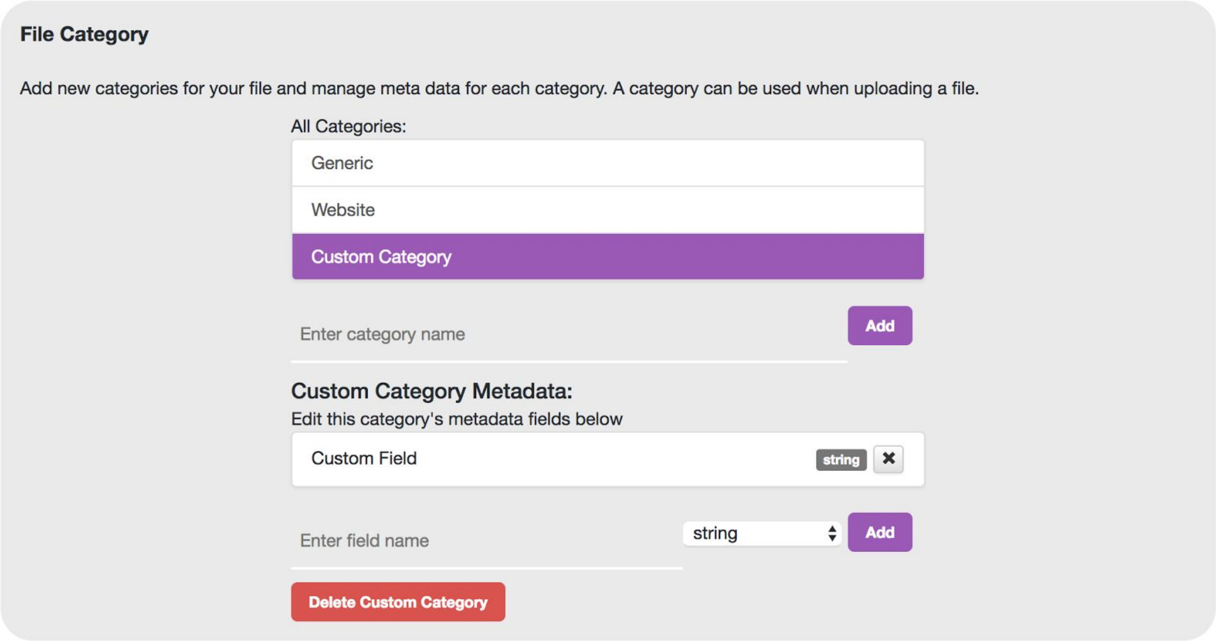
All Categories:

- Generic
- Website
- Custom Category

Custom Category Metadata:

Edit this category's metadata fields below

Figure 26. File Category when a user-defined category is selected. In this case, “Custom Category” is selected



File Category

Add new categories for your file and manage meta data for each category. A category can be used when uploading a file.

All Categories:

- Generic
- Website
- Custom Category

Custom Category Metadata:

Edit this category's metadata fields below

Custom Field — string

Figure 27. User defined category with one new metadata field added

To change number of files that will display on the list of the main screen (section 2.2), under the “File List” section, hover the mouse over number in that section. An arrow up and down will appear. Clicking up will increase the number of files to display on a single page on the list on the main screen, and click down will decrease the number. Click “Save” button next to it to save the change.

2.8 Chrome Extension

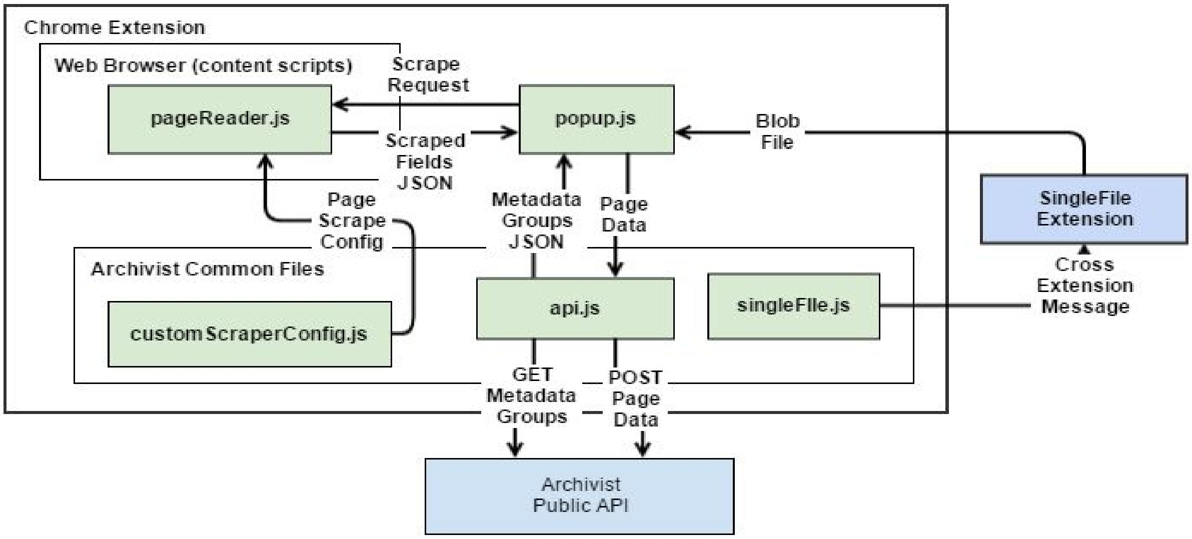


Figure 28. Chrome Extension high level data flow diagram

If the user wants to upload a file using the Web Extension, the process is handled differently than uploading a file via the Upload File Screen for the user. Before being able to do so, the user must configure the extension with an API location corresponding to the server where Archivist is installed. Once that is complete and the user requests to upload a document via the Web Extension, the extension then communicates with the Ruby on Rails Public API, which is exposed to the network, which then communicates with the MongoDB database.

2.9 UI Component Design

For simplicity, we broke this section into multiple different diagrams to enhance readability.

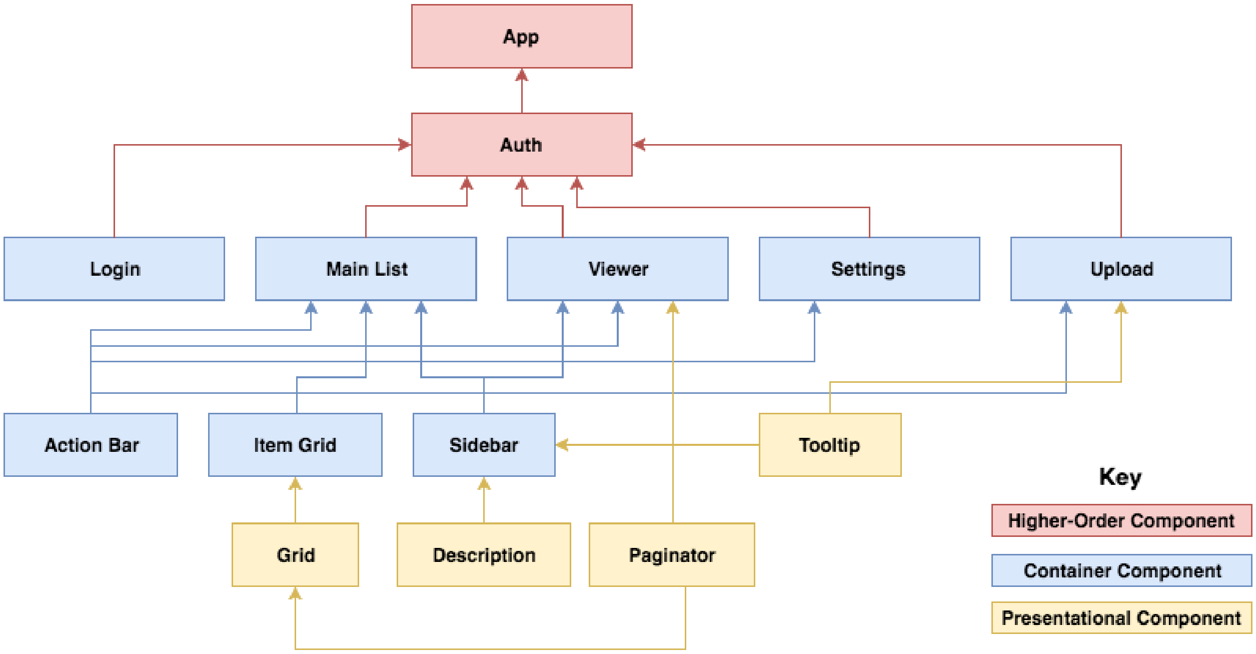


Figure 29. Front-end Component Design (React)

The root of the application contains a single, higher-order component called App. It has a child HOC called Auth, which handles all authentication and makes sure the user is logged in before showing any content. Container components are “smart” components, who are connected to the state of the application, and contain several child components. Container components are also able to make changes to the state of the application. Presentational components are “dumb” components, and are reused throughout the application for various needs. They do not control the state of the application, and only manage their own state.

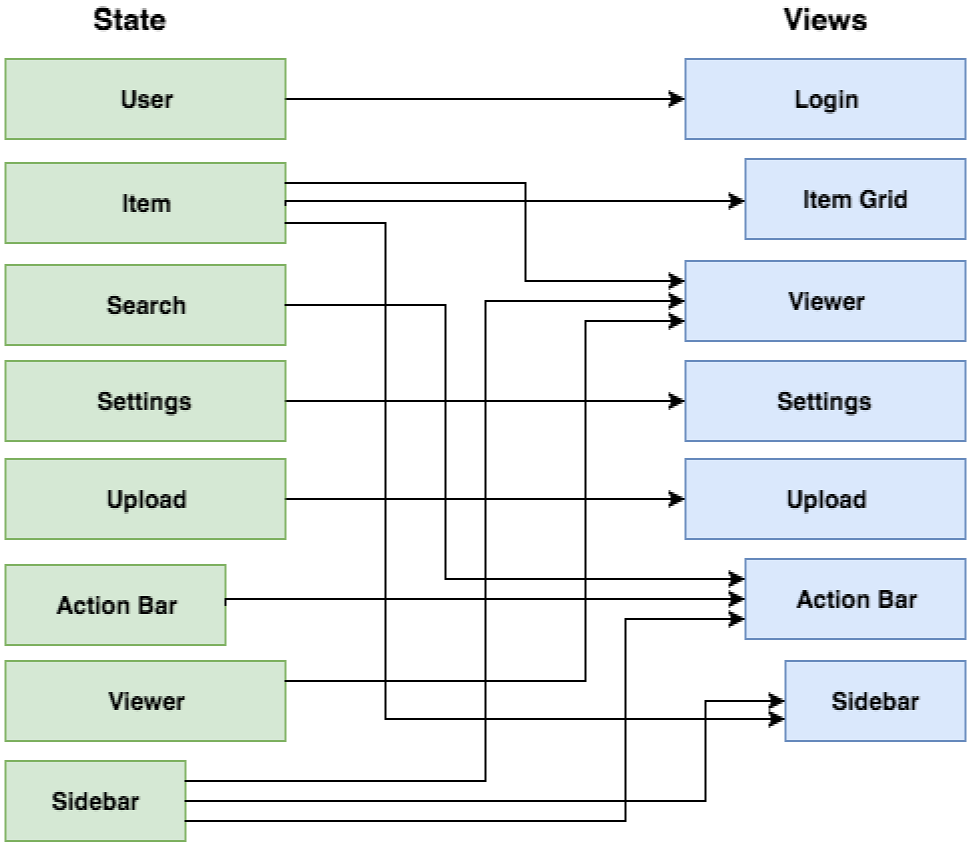


Figure 30. Front-end State Design (Redux/React)

Container components are linked to the state of the application. They receive state properties from various slices of the state. For example, the Viewer component portions of the item state, sidebar state, and viewer state. The viewer needs to get the current item's data, resize based on the sidebar's width, and gets information like current page number for pdfs from the viewer state. Each container has the ability to call actions, which can trigger various reducers (or listeners) in each state slice, and the state is updated accordingly. For example, if the current selected item is changed, the item state slice is updated. Then, the viewer, sidebar, and item grid will receive the new information about the item and update based on the new item.

3. Architecture

This section will detail the backend architecture for Archivist through the use of various diagrams as well as a brief description of each of the different components.

3.1 Context Diagram

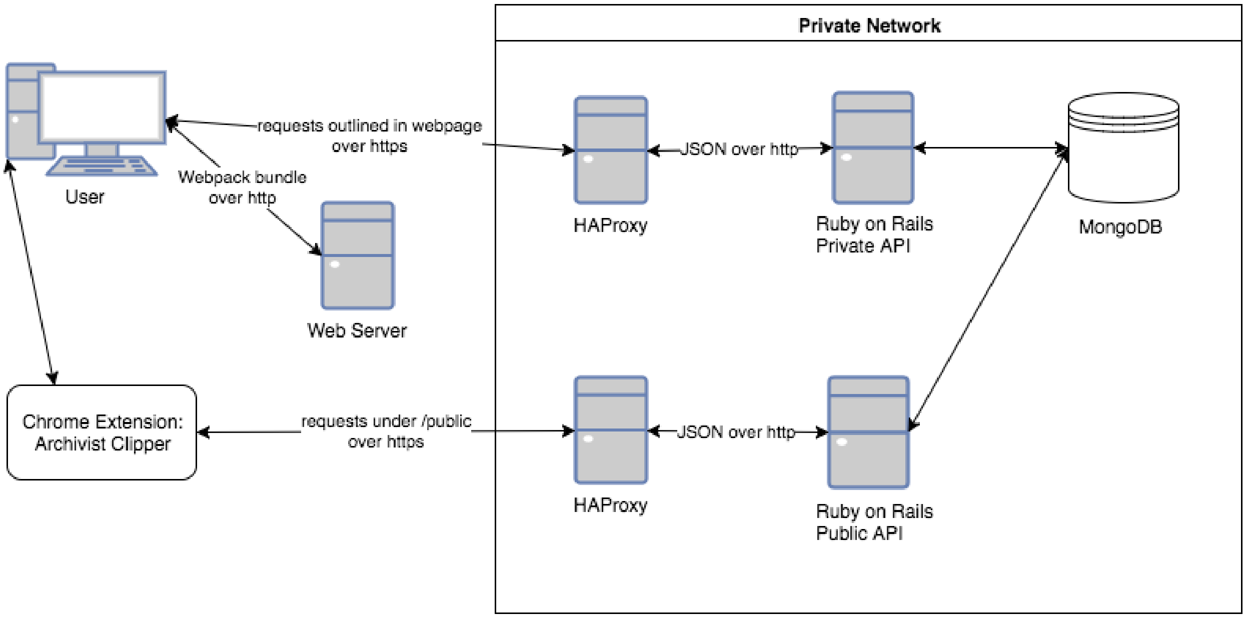

Private Network
User
requests outlined in webpage over https
Webpack bundle over http
Web Server
HAProxy
JSON over http
Ruby on Rails Private API
MongoDB
Chrome Extension: Archivist Clipper
requests under /public over https
HAProxy
JSON over http
Ruby on Rails Public API

Figure 31. Context Diagram explains the flow of data between pieces of our system.

Figure 31 shows the overall architecture for the Archivist project. The architecture consists of five major components: the Web Server front end, the Ruby on Rails Private API, the Ruby on Rails Public API, the Chrome Web Extension, and MongoDB.

3.2 Components

All the following components, sans Web Extension, will be run using Docker in their own container. A docker-compose file will be used to network the containers together and ensure that connections can only be made from approved sources.

MongoDB: The product will use the version 3.2. Mongo will be configured for small file support to store the uploaded files in addition to the user data.

Ruby on Rails APIs: Both will use Rails 5 with Ruby version 2.2.5, and will handle data retrieval as well as business logic. All access to Mongo will be through the Rails APIs. The private API will only be accessible to/from the MongoDB container, and Web Server container. The public API will be accessible publicly — but will require a token for authentication — and to MongoDB.

Web Server: This component will be hosted with the most recent version of Node.js, and the user interface will be designed with React version 15.3.2 and Redux version 3.6. It will communicate with the users' web browsers, store the authentication token for the user across sessions, and retrieve information from the Rails private API.

Web Extension: The extension is built specifically for the Google Chrome browser and will only be available on that platform. The extension uses much of the same technical platform as the main front end of the application. It is built using ECMAScript 6 style javascript without any extra frameworks. The current implementation of the extension is in two parts. One part is the already existing SingleFile core extension which handles the page processing. The other part is our extension which receives html file output from the core extension and passes that to the public API. Our extension is also responsible for allowing the user to enter metadata, tags and description. In some cases those fields can be automatically filled in from specific custom web scraper configurations or through more general OpenGraph standards if they are used.

4. Data

4.1 Model Design

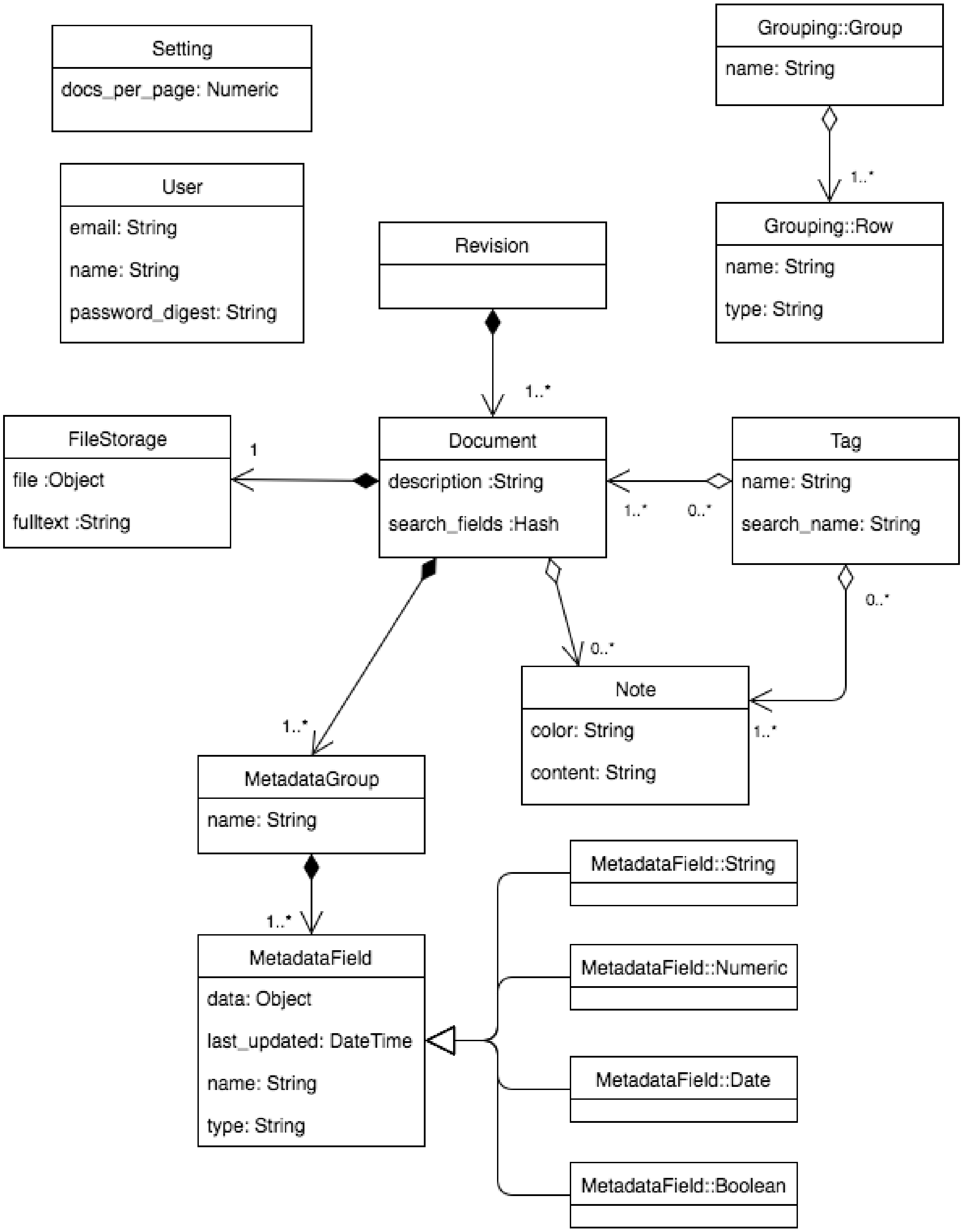


Figure 32. UML Class Diagram for Backend Code

4.2 API Endpoints

The body of

4.2.1 Authentication

URL	/authentication/login
Method	POST
Input Body	{ email: String, password: String }

Output	If did login: { auth_token: String }
	Otherwise: { error: String }

Description	The authorization token is a JsonWebToken.

URL	/authentication/status
Method	GET
Input Header	{ Authorization: String }
Output	If token is valid: { valid: true } Otherwise: { valid: false }
Description	

4.2.2 Documents

URL	/documents
Method	GET
Input Query Params	{ page: Integer }

Output	{
	  documents: [{
	    id: String,
	    metadata_fields: [{
	      id: String,
	      name: String,
	      type: String,

	data: String,
	group: String
	}, ...]
	}, ...],
	meta: {
	current_page: Integer,
	next_page: Integer,
	prev_page: Integer,
	total_pages: Integer,
	total_count: Integer
	}
	}

Description	

URL	/documents
Method	POST

Input Body	{ document: [{
	file: String,
	description: String,
	tags: [String, …],
	metadata_fields: [{
	name: String,
	type: String,
	data: String,
	group: String,
	}, ...]
	}

Output	{ success: true }
Description	

URL	/documents/search
Method	GET
Input Query Params	{ page: Integer }

Input Body	{ search: {
	andOr,
	groups: [{
	groupType: String,

	andOr: Boolean, not: Boolean, QUERY }, …]} }

Output	{
	documents: [{
	id: String,
	metadata_fields: [{
	id: String,
	name: String,
	type: String,
	data: String,
	group: String
	}, ...]
	}, ...],
	meta: {
	current_page: Integer,
	next_page: Integer,
	prev_page: Integer,
	total_pages: Integer,
	total_count: Integer
	}
	}

Description	QUERY can be: (a) description: String, (b) terms: String, (c) tags: [String, …], (d) item_types: [String, …], (e) fields: [{name: String, type: String, group: String, data: String}, …]

URL	/documents/:id
Method	GET

Output	{ document: {
	id: String,
	content_type: String,
	description: String,
	tags: [String, …],
	metadata_fields: [{
	name: String,
	type: String,
	data: String,
	group: String,
	}

	}
Description	

URL	/documents/:id
Method	PUT

Input Body	{ document: {
	description: String,
	tags: [String, …],
	count: Integer,
	}
	}

Output	{ success: true }
Description	Count is the number of tags in the array.

URL	/documents/:id/content
Method	GET
Output	The stored file as a blob.
Description	

4.2.3 Metadata

URL	/metadata_fields/types
Method	GET
Output	{ types: [String, …] }
Description	All the supported metadata types.

URL	/metadata_fields/:id
Method	PUT
Input Body	{ metadata_field: { data: String } }

Output	{ success: true }
Description	

4.2.4 Groups and Fields

URL	/system/groups
Method	GET

Output	{ groups: [{
	id: String,
	name: String,
	can_edit: Boolean,
	fields: [{
	id: String,
	name: String,
	type: String
	}, ...]
	}

Description	

URL	/system/groups
Method	POST

Input Body	{ group: {
	name: String,
	fields: [{
	name: String,
	type: String
	}
	}

Output	{ group: {
	id: String,
	name: String,
	can_edit: Boolean,
	fields: [{
	id: String,
	name: String,
	type: String
	}

	}
Description	

URL	/system/groups/:id
Method	GET

Output	{ group: {
	id: String,
	name: String,
	can_edit: Boolean,
	fields: [{
	id: String,
	name: String,
	type: String
	}
	}

Description	

URL	/system/groups/:id
Method	PUT
Input Body	{ name: String }
Output	If group is editable: { success: true } Otherwise: { success: false }
Description	

URL	/system/groups/:id
Method	DELETE
Output	If group is editable: { success: true } Otherwise: { success: false }
Description	

URL	/system/groups/:group_id/field
Method	POST

Input Body	{ field: {
	    name: String,
	    type: String
	  }
	}

Output	If group is editable: { success: true } Otherwise: { success: false }
Description	

URL	/system/groups/:group_id/field/:id
Method	PUT

Input Body	{ field: {
	    name: String,
	    type: String
	  }
	}

Output	If group is editable: { success: true } Otherwise: { success: false }
Description	

URL	/system/groups/:group_id/field/:id
Method	DELETE
Output	If group is editable: { success: true } Otherwise: { success: false }
Description	

4.2.5 Public

URL	/public/documents
Method	POST
Description	This is equivalent to POST /documents from 4.2.2

URL	/public/groups

Method	GET
Description	This is equivalent to GET /public/groups from 4.2.4

4.3 Algorithms

The following sections describe, at a high level, the most crucial algorithms to the Archivist project structure and functionality.

4.3.1 Login

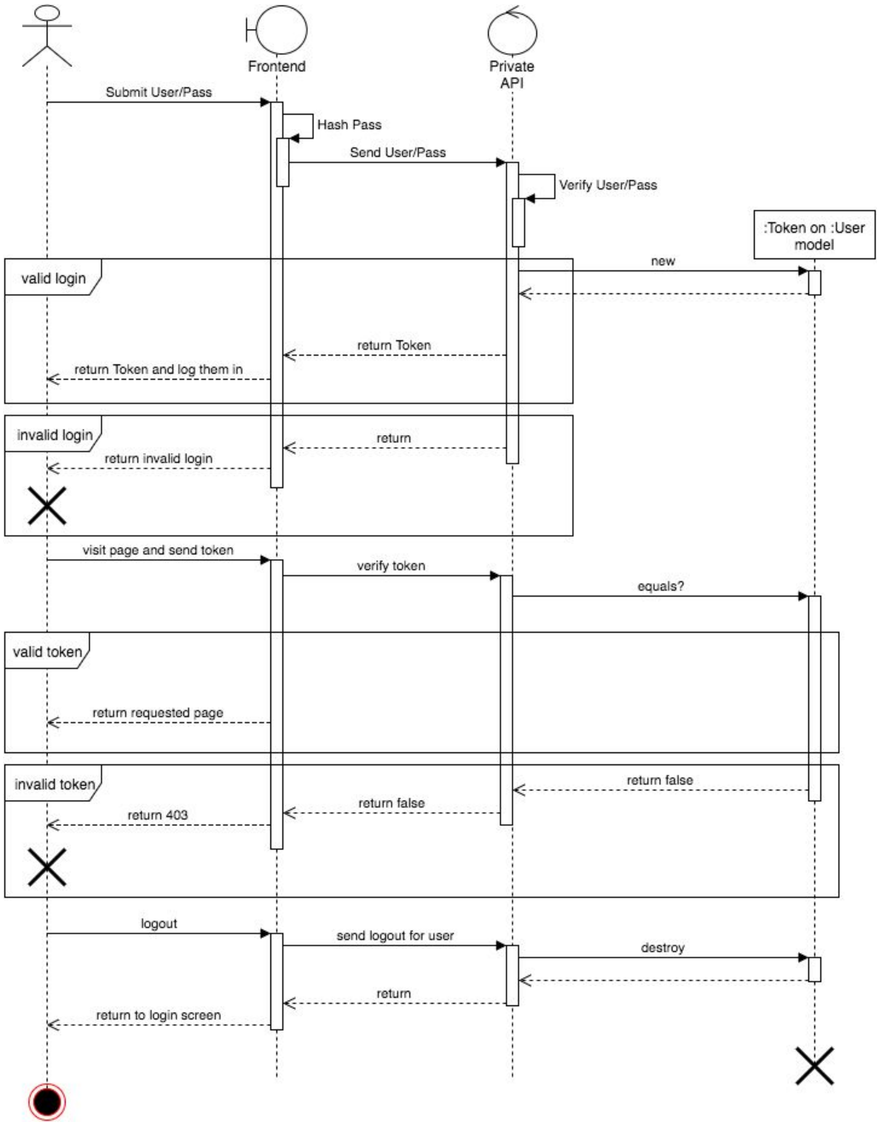


Figure 33. Token Authentication Model

4.3.2 Automatic metadata

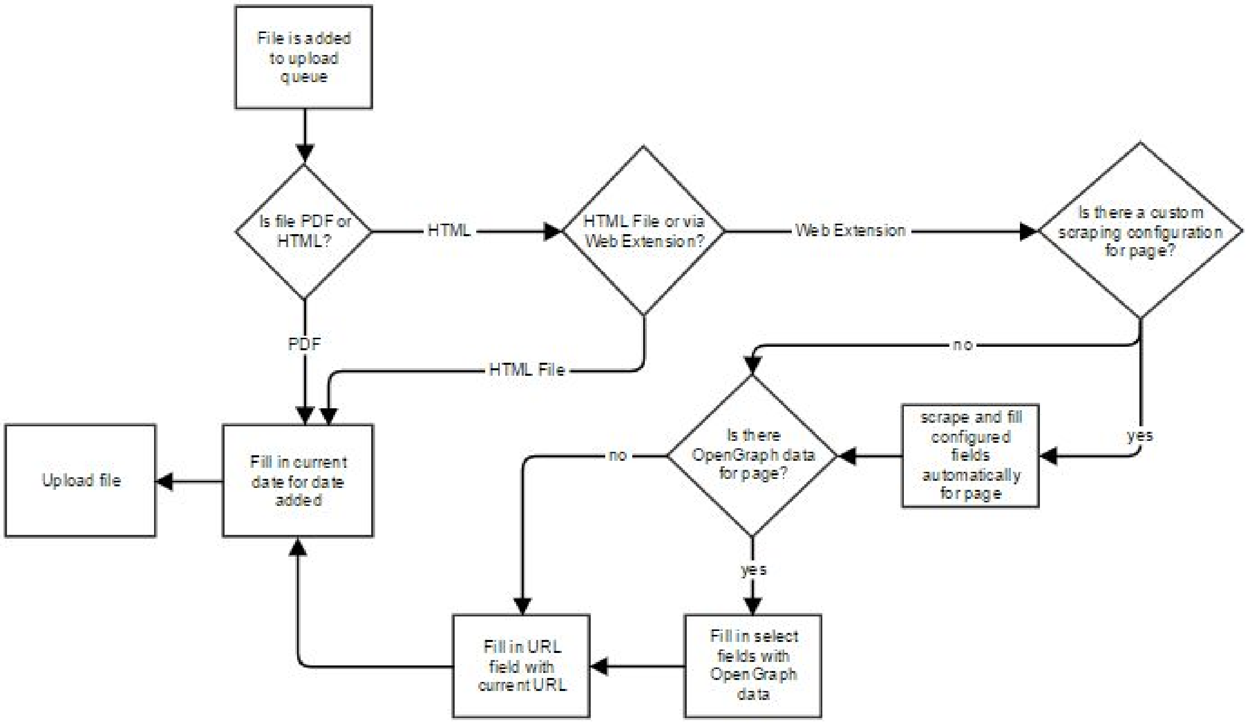

File is added to upload queue
Is file PDF or HTML?
HTML
HTML File or via Web Extension?
Web Extension
Is there a custom scraping configuration for page?
PDF
HTML File
no
yes
Upload file
Fill in current date for date added
no
Is there OpenGraph data for page?
scrape and fill configured fields automatically for page
yes
Fill in URL field with current URL
Fill in select fields with OpenGraph data

Figure 34. File upload automatic metadata logic

4.3.3 Advanced Search Algorithm

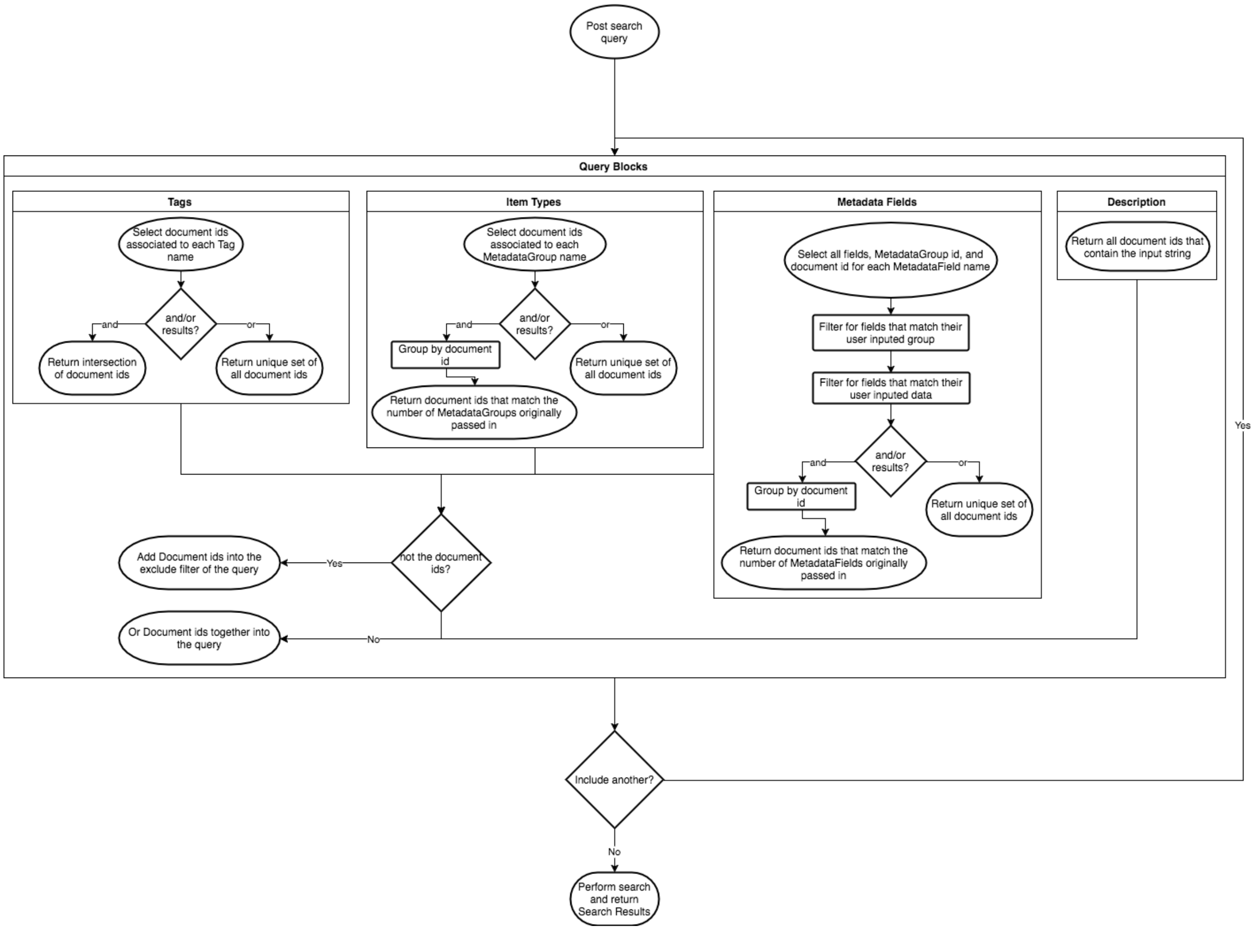

Post search query
Query Blocks
Tags
Select document ids associated to each Tag name
and/or results?
and
or
Return intersection of document ids
Return unique set of all document ids
Item Types
Select document ids associated to each MetadataGroup name
and/or results?
and
or
Group by document id
Return unique set of all document ids
Return document ids that match the number of MetadataGroups originally passed in
Metadata Fields
Select all fields, MetadataGroup id, and document id for each MetadataField name
Filter for fields that match their user inputed group
Filter for fields that match their user inputed data
and/or results?
and
or
Group by document id
Return unique set of all document ids
Return document ids that match the number of MetadataFields originally passed in
Description
Return all document ids that contain the input string
not the document ids?
Yes
Add Document ids into the exclude filter of the query
No
Or Document ids together into the query
Include another?
Yes
No
Perform search and return Search Results

Figure 35. Advanced Search Algorithm State Diagram