IBM Cloud Pak for Business Automation Demos and Labs 2026

Configuration and Usage of IBM Content Assistant with Cloud Pak for Business Automation

V 0.8s (for CP4BA 25.0.1 with IBM Content Assistant 1.0.x)

Matthias Jung, Ph.D.
matthias.jung@de.ibm.com

NOTICES

This information was developed for products and services offered in the USA.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
United States of America

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

TRADEMARKS

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

IT Infrastructure Library is a Registered Trade Mark of AXELOS Limited.

ITIL is a Registered Trade Mark of AXELOS Limited.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Table of Contents

1 Introduction

1.1 IBM Content Assistant

IBM Content Assistant (ICA) provides natural language insights about business content in IBM FileNet Content Manager repositories. The Content Assistant helps to quickly find information across large volumes of content and saves time by avoiding manual, repetitive review of document content.

IBM Content Assistant includes generative AI capabilities that can summarize document content, compare documents and versions, and provide accurate answers to questions within the context of the selected business content in an IBM FileNet Content Manager repository.

These capabilities are based on the IBM watsonx.ai provided Large Language Model (LLM) that is coupled with context-specific Retrieval-Augmented Generation (RAG). The RAG augments the LLM with context-specific knowledge base and information that is hosted in the IBM FileNet Content Manager repository.

IBM Content Assistant (ICA) provides these capabilities in a secure way, by ensuring that any user running queries will get information only from those documents in repository, to which this user has access.

1.2 Architectural Overview

The IBM Content Assistant architecture is composed of three primary components:

1. The **ICA plugin** for IBM Content Navigator (ICN) aka IBM Automation Navigator exposes the capabilities of ICA to the end-user by embedding a chat interface into ICN.
2. The **FNCM Gen AI Connector Service** also referred to as **ICA Server** is responsible for transforming documents into vector embeddings, executing semantic search queries against a vector index, and perform the interactions with the underlying Large Language Model (LLM) hosted by IBM watsonx.ai.

   Document ingestion involves converting unstructured content into high-dimensional vector representations, enabling efficient similarity search and contextual retrieval. These embeddings are stored in a Vector database, which can run on premises, allowing customers to maintain full control over data locally, or operate as a SaaS-managed service.
3. Two specialized **FileNet Object Store Add-ons** that connect an Object Store in a Cloud Pak for Business Automation environment to the ICA Server.

   The **5.6.0 Persistent Text Extract Extensions** add-on handles the following aspects:

   - Generate text extracts for document content in your repository
   - Store the extracted text as Annotations.

   The **5.7.0 Gen AI Extensions** add-on handles the following aspects:

   - **Expose ICA's capabilities via FileNet Content Engine extensions**
     ICA is seamlessly integrated by utilizing standard CPE capabilities like document classes, subscriptions, event handlers and so on.

     All capabilities are available through both the Java API and a GraphQL interface, that are also used by the ICA plug-in for Navigator but also support flexible integration into custom applications.
   - **Trigger vectorization and AI summary generation and storage**
     When a document is to be indexed, the extracted document text is sent for vectorization and AI summary generation to the ICA Server.

     For each document the following is stored:
     - **Indexing metadata**: Status and timestamp indicating when the document was ingested into the vector index and an AI summary was generated.

- **AI-generated summary**: A concise abstract generated by the configured LLM hosted by watsonx.ai.

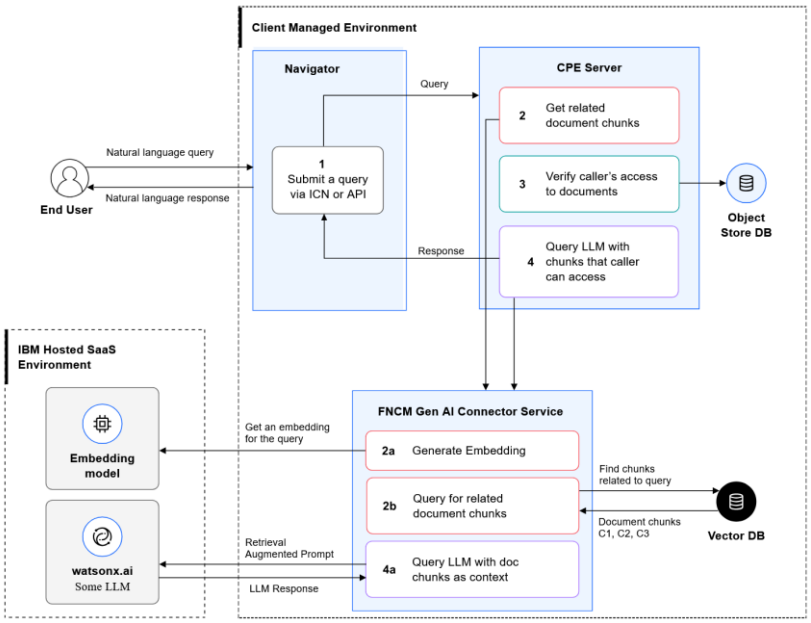
- **Facilitate the execution of ICA queries**
  When a user submits a query to the IBM Content Assistant and a query object is created in the Content Platform Engine (CPE) object store.

  The add-on reacts on query objects being created in the Content Platform Engine (CPE) object store and performs a series of calls to the Gen AI connector service to produce a response:

  - Get document chunks that are related to the query from ICA Server.
  - Submit the user question and document chunks to the LLM hosted by IBM watsonx.ai to generate a response.

- **Ensure that only chunks for document are fed into the LLM which the user can access**
  As part either determining the documents chunks related to the query or when the whole document is sent to the LLM, it verifies that the user executing the request has at least VIEW_CONTENT access for the specified or identified documents.

Below picture contains the above-mentioned components and a respective call flow when a user performs an ICA query:



For more information, refer to the Overview section in the documentation for IBM Content Assistant https://www.ibm.com/docs/en/content-assistant?topic=assistant-overview-content.

1.3 IBM Content Assistant Form-factors

IBM Content Assistant is available as a SaaS (documentation) offering or can be deployed within a Cloud Pak for Business Automation deployment, called IBM Content Assistant Client Managed Software (documentation).

The release cycles for both form-factors are not completely aligned. Depending on how ICA is made available in the environment you are using that can result in:

- the chat UI looking slightly differently
- the LLMs and the various prompts for these LLMs being used may differ
- bugs potentially existing in one but not the other form-factor

In addition, some capabilities may be available in one but not the other form-factor altogether.

1.4 Remarks regarding Screenshots and Results

IBM Content Assistant is an innovative approach to make the knowledge that resides in enterprise repositories accessible by using Large Language Models. As such it exists in the very dynamic and rapidly evolving space of generative AI. This means that the ICA itself and the components it relies on (e.g. LLMs) evolve and change quickly.

In relation to this lab that means that:

- Screenshots may look slightly different.
- The sample queries provided may not result in the exact output as seen in the screenshots.
- Executing the same query may produce different output depending on the form-factor.

A general word of caution
In general, a Large Language Model (LLM) based capability does not guarantee that the output is always identical when asking the same question multiple times. So be prepared to see output differing from the screenshots. Feel free to play around with the suggested queries to see how the output changes.

The LLM used for this lab is **llama-4-maverick-17b-128e-instruct-fp8**. Using a different LLM or a different version of the same LLM would result in different responses.

1.5 Lab Overview

This lab demonstrates the key use-cases of IBM Content Assistant. The exercises serve as examples. The API exercises are moreover inspired by the needs of automation projects such as the Client Onboarding application showcased in other labs.

The Content Assistant lab can be performed independently of the other Content labs, i.e. independently of "Setting up FileNet Content Platform Engine for Automation Projects on Cloud Pak for Business Automation", “Interfacing FileNet Content Platform Engine with GraphQL on Cloud Pak for Business Automation” and “Introduction to Business Automation Navigator in Cloud Pak for Business Automation”.

Exercise “End-user interface” demonstrates the user interface of IBM Content Assistant in IBM Content Navigator aka IBM Business Automation Navigator. It shows how IBM Content Assistant presents itself to the end-users and how it can be used to determine facts from documents in the repository via a chat interface, without the need to open the documents and search for the information. Questions on single documents can be raised directly from the Browse tool of IBM Content Navigator or from the open DaeJa Viewer. Further, questions can also be asked on the whole repository, and it is important that ICA will consider only those documents, to which a given user has access.

Exercise “Administration” provides a deep dive into the administrative aspects of IBM Content Assistant, primarily managed through ACCE (Administrative Console for Content Engine). It focuses on the configuration and behavior of the GenAI Extensions add-on, which enables vector indexing and AI-based summarization for both newly ingested and existing documents via sweeps and event subscriptions.

The exercise demonstrates how to configure:

- **Automatic Text Extraction and Persistent Storage** using a dedicated Object Store add-on to cache extracted text for reuse.
- **Automated Vector Indexing** to register documents in the vector database.
- **Automated Summarization** using an LLM hosted by watsonx.ai to generate document abstracts.

It also shows where these artifacts—such as indexing metadata, summaries, and extracted text—are stored within the FileNet repository, offering insight into the internal workings of IBM Content Assistant.

Exercise “Application Programming Interface” introduces capabilities to create IBM Content Assistant queries through the APIs of FileNet Content Manager. The exercise uses the GraphQL interface for some hands-on examples, which can be adapted to the needs of any automation project. It is out of scope for this lab to create and run Java programs. Refer to the documentation for additional information.

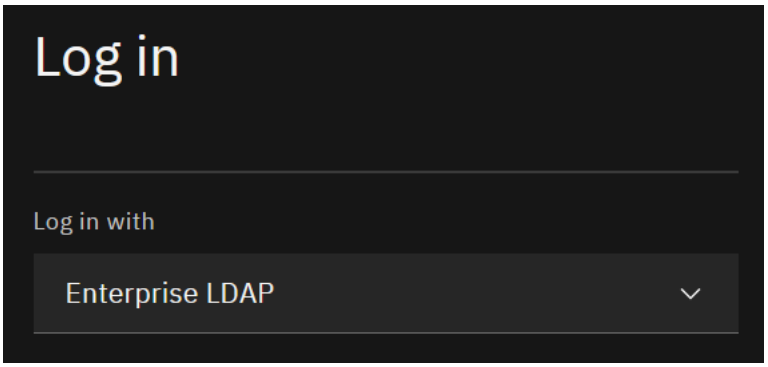
1.6 Lab Setup Instructions

_1. If you are performing this lab as a part of an **IBM event**, access the document that lists the available systems and URLs along with login instructions. For this lab, you will need to access **IBM Content Services GraphQL**, the **IBM Content Services ACCE,** and the **IBM Content Navigator - ICN desktop**. Further, you will need to have access to a second account named **icademo.** Request the password from the lab instructors.

If you are performing this lab **self-paced** in an environment, which has been enabled for the Client Onboarding labs including the IBM Content Assistant component, you can find the available URLs and information about the icademo user in the **ConfigMap "000-client-onboarding-information"** of the IBM Cloud Pack for Automation project in the OpenShift Web Console.

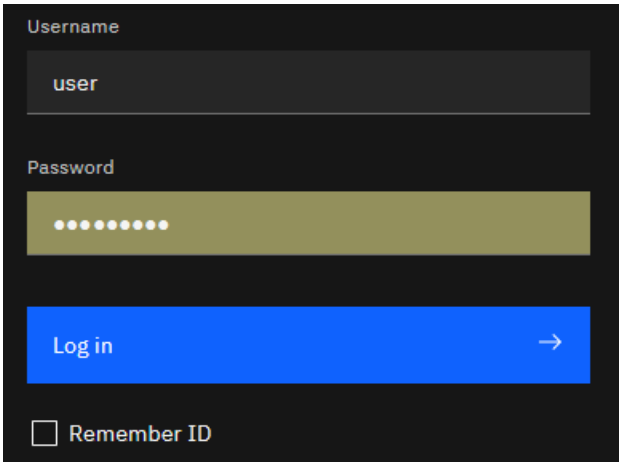
_2. The different exercises require different components of the IBM Cloud Pak for Business Automation. Paste the respective URL used into your web browser. The first exercise will use the **IBM Content Navigator - ICA desktop**.

_3. If you haven't logged on before, the browser will be redirected to a Log in page. Select **Enterprise LDAP log in** option.



_4. **Enter** your ***Username*** (usrXXX) and ***Password*** as described in step 1 and **click Log in**.

Only when stated explicitly in the lab instructions, use the **icademo** account.



_5. After successful authentication you will be redirected to the **IBM Content Navigator ICN desktop**. IBM Content Navigator will show the Home page with a tile for each feature configured for the desktop.

2 Exercise: End-user interface

2.1 Introduction

This exercise demonstrates the user interface of IBM Content Assistant (ICA) in IBM Content Navigator (ICN) aka IBM Business Automation Navigator. It shows how IBM Content Assistant presents itself to the end-users and how the chat interface can be used to determine facts from documents in the repository, without the need to open the documents and search for the information. This is achieved through the IBM Content Assistant plugin for IBM Content Navigator which needs to be enabled for a desktop.

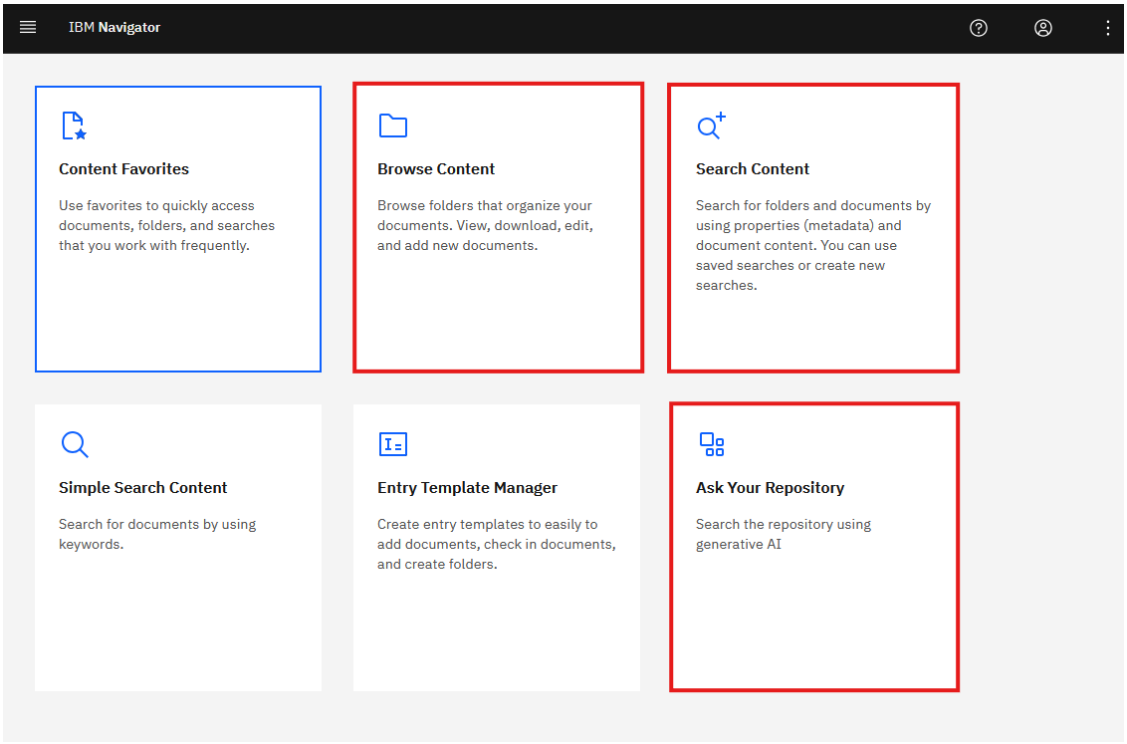
Questions on single documents can be raised directly from the Browse or Search tools of IBM Content Navigator or from the open IBM DaeJa Viewer. IBM Content Assistant is available as a Search Assistant Chatbot in the lower right corner of the browser window. Through the “Ask Your Repository” tool, questions can also be asked on the whole repository. It is an important feature that ICA will consider only the documents, to which a given user has access.

Section 2.2.1 introduces the ICA chat window and its components. Section 2.2.2 focuses on executing queries on single documents and section 2.2.3 on executing queries on multiple documents. Section 2.2.4 demonstrates how to run queries directly from the DaeJa Viewer. IBM Content Navigator is frequently embedded into other applications via bookmark.jsp URLs, which directly opens the DaeJa Viewer for a specific document. In such cases, the IBM Content Assistant chat window remains accessible from within the DaeJa Viewer. Finally, section 2.2.5 shows how to ask questions against the complete repository showcasing that ICA considers only those documents to which a given user has access.

2.2 Exercise Instructions

2.2.1 IBM Content Assistant Chat Window

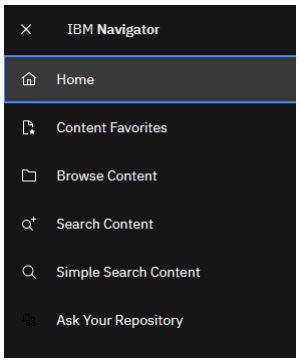
_1. After logging in, the IBM Content Navigator Desktop comes up, showing tiles for the Features, which are available on this desktop. For the ICN desktop the following tiles are available.



For IBM Content Assistant foremost the features “Search Content”, “Browse Content” and “Ask Your Repository” are relevant.

The features are also available through the so-called “Hamburger Menu”, this is the icon in the top left corner of IBM Content Navigator. For faster switching, use the Hamburger menu.

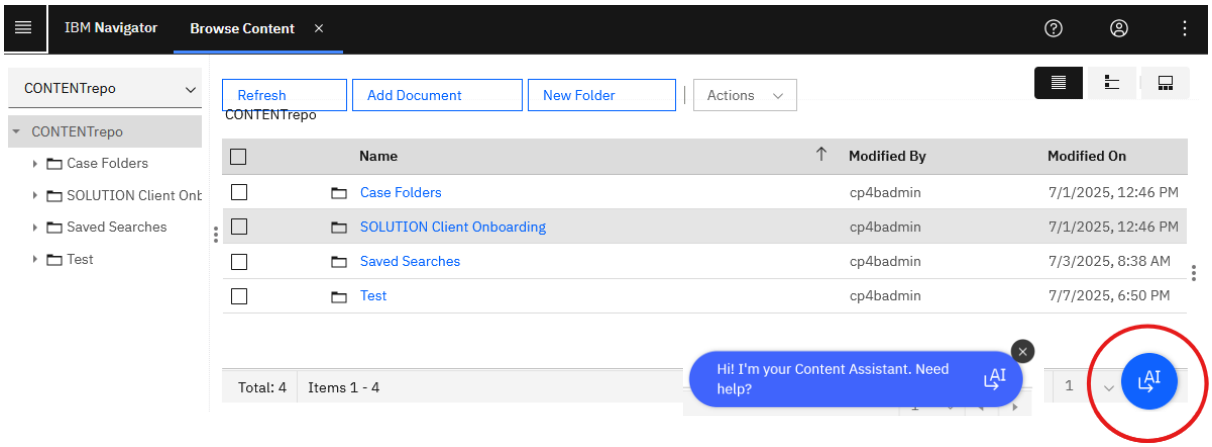
_2. **Switch** to the **Browse Content** feature.



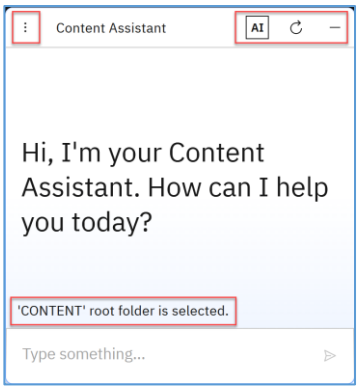
When selected, the “Browse Content” feature shows the root folder of the default FileNet Object Store configured for the desktop, in this case for the CONTENT Object Store. In addition, a dropdown allows you to switch to any other Object Store configured for the desktop.

Depending on the environment you are using the object store name may be shown as CONTENT or CONTENTrepo for the default object store.

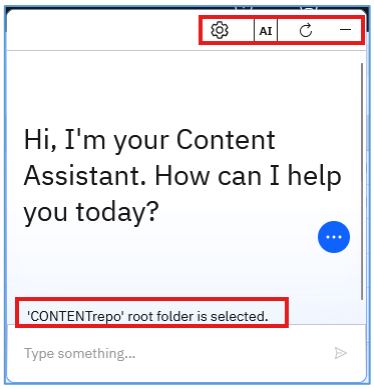
When the ICA plugin is enabled, the Browse Content feature shows a round blue button in the lower right corner to invoke IBM Content Assistant. In later versions of the ICA plugin an additional text may be shown in addition, to make that more obvious. Other features of Content Navigator might expose other AI assistants, like the Workplace Assistant when using the Workplace feature.



_3. **Click** on the **AI button** to open the IBM Content Assistant chat window.



or

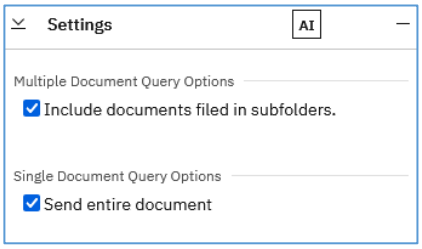


Observe the information “’CONTENT’ root folder is selected”. It indicates that as part of a query all documents in the root folder will be passed as context.

Explore the chat window starting with the **toolbar buttons** at the top of the chat window.

_4. **Click** the **AI button**, to open a pop-up window displaying information that IBM watsonx is being the underlying AI technology.

_5. **Open** the **Settings pane**, that is either accessible via the menu shown when clicking the three vertical dots in the top left corner of the chat window or a wheel icon.



The settings allow you to define whether to **include documents filed in subfolders** in the search prompt or not. This option is enabled by default.

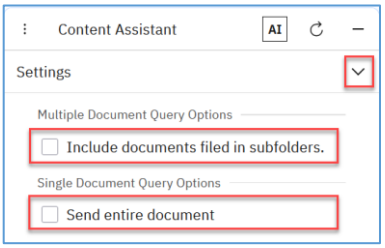
The second option allows you to specify to **Send entire document** when performing single document queries or if only those document chunks are sent to the LLM that are most relevant to the query. This option is enabled by default.

Disabling the option might be useful when the extracted text exceeds 400,000 characters and the entire document cannot be passed as context. In this case, a vector query then automatically retrieves the most relevant document chunks, which are then sent to the LLM along with the prompt to generate a response.

As will be seen later in the lab, the availability of excerpts from the document that was used to generate the response is only available, when a vector query is executed to retrieve the most relevant document chunks, before calling the LLM.

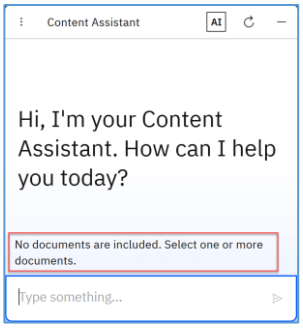
> More details are available in the section “Asking questions about a single document” in the documentation https://www.ibm.com/docs/en/content-assistant?topic=assistant-asking-questions-about-single-document.

_6. For now, **disable** both features **Include documents filed in subfolders** and **Send entire document**.



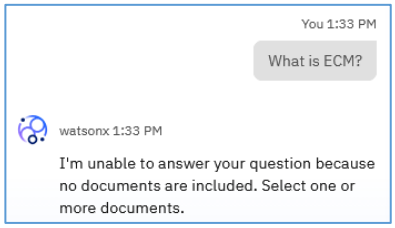
Click on the **symbol** in the left or right of the Settings label to return to the chat window.

Notice that now no documents are included for a query.



_7. The **last two buttons restart** the conversation and **minimize** the chat window.

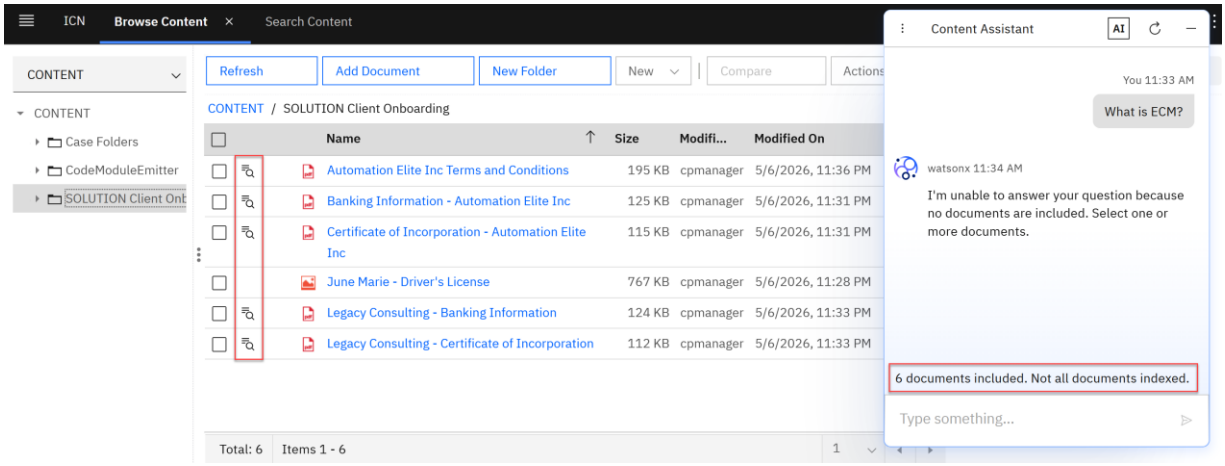
_8. **Enter** in **a prompt**. It doesn't matter what you write, you might for example ask **What is ECM?** IBM Content Assistant will refuse to answer, as no documents are included in the query.



Leave the chat window open for the next subsection.

2.2.2 Single Document Queries in the Browse View

_1. **Select** the **SOLUTION Client Onboarding** folder in the navigation area on the left side of IBM Content Navigator.



Notice in the Browse view that some documents have an icon on the left side. This icon indicates that the document is vector indexed and can be used in the context of IBM Content Assistant.

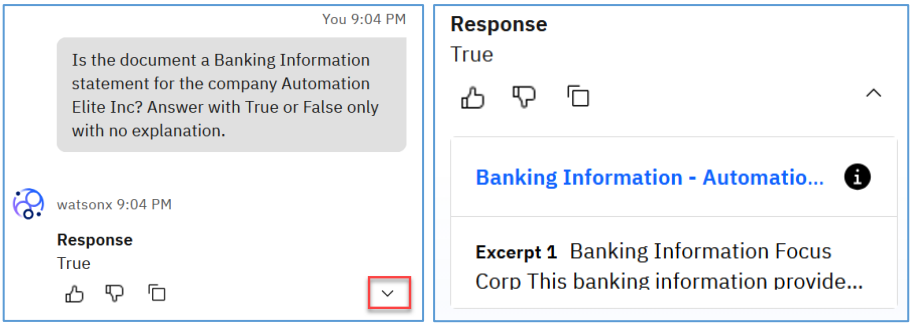
Notice further that in the IBM Content Assistant chat window, now the 6 documents contained in that folder are selected for being included in the search query. If no documents are selected in the Browse view, automatically all documents will be included for any search prompt.

_2. **Select** the document with the title **Banking Information - Automation Elite Inc**. by checking the checkbox at the beginning of the row. Notice the change in the IBM Content Assistant window.

> In the Desktop settings, it can be configured whether document selection checkboxes are shown or hidden. If the checkboxes are hidden, documents can be selected by clicking on them directly. In this case, care should be taken not to click the blue link, as this will open the DaeJa Viewer.

_3. **Type** or **copy & paste** the question **Is the document a Banking Information statement for the company Automation Elite Inc? Answer with True or False only with no explanation.**

> When post-processing answers from ICA in a custom application, it's a good idea to specify what result would be expected.
> You can play around leaving out the **with no explanation** part to see the difference.

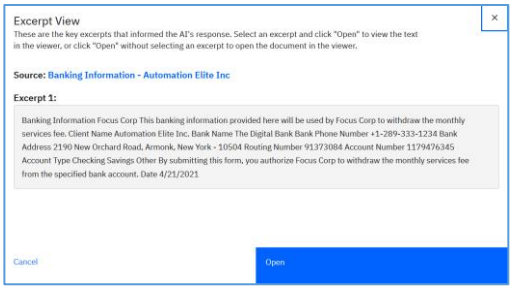


_4. **Expand** the **twisty** to show the parts of the document (excerpts) from which the answer was derived.

> As we have unchecked the option to **Send entire document** in the **Settings**, ICA first performed a vector search to find the most relevant document chunks for our query before sending the query and the document chunks to the LLM.
> That is the reason why the twisty is shown.

_5. **Click** on the **excerpt** to open it in a pop-up window followed by **Cancel**.

You could also open the full document in Daeja Viewer by clicking Open.

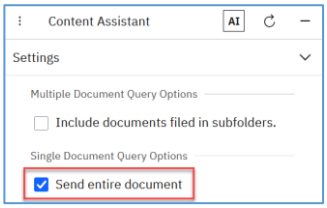


> The banking information document is rather short, but in general making use of excerpts helps to validate the response if required, without going through the full document.

_6. If present, **click** the **thumbs up** icon as the answer is correct.

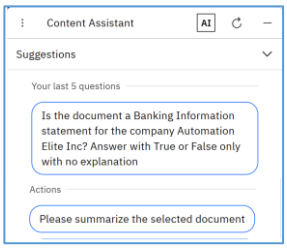
> The Thumbs-up / Thumbs-down icons are only available when using ICA SaaS. In ICA Customer Managed Software there is no way to feed back the information to the Large Language Model.

_7. **Open** the **Settings** again and **enable** the **Send entire document** checkbox before **closing** them again.



_8. In the ICA chat window, **click** either the **three vertical dots** in the upper left corner and **select Suggestions** or the **blue circle** with the **three dots** to bring up the suggestions pane.

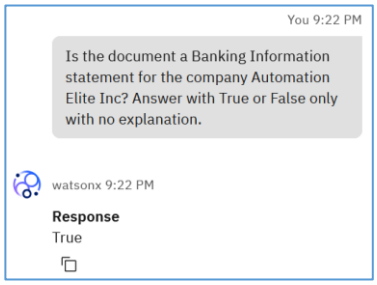
It contains possibly useful further queries or requests, along with a history of your past queries.



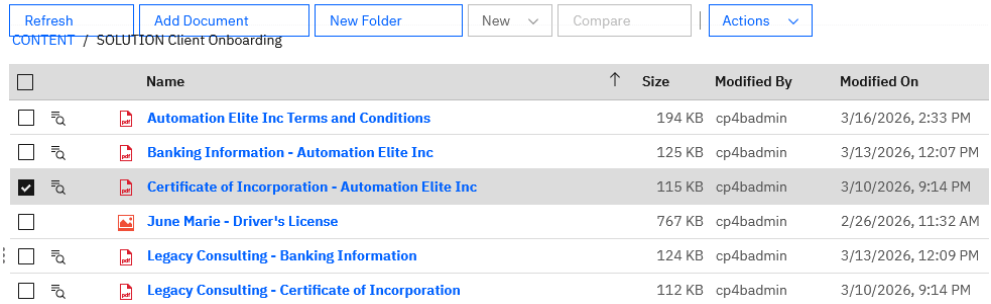
_9. **Click** on the **previously executed query** to run it again.

Observe that the response is identical, but the twisty to show the excerpts is not shown.

> The reason is that with enabling the **Send entire document** checkbox, ICA no longer first performs a vector search to identify the most relevant document chunks but immediately sends the full document content to the LLM.

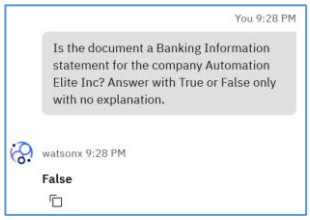


_10. **Select** the **Certificate of Incorporation – Automation Elite Inc** document and **deselect** the **Banking Information – Automation Elite Inc** document.



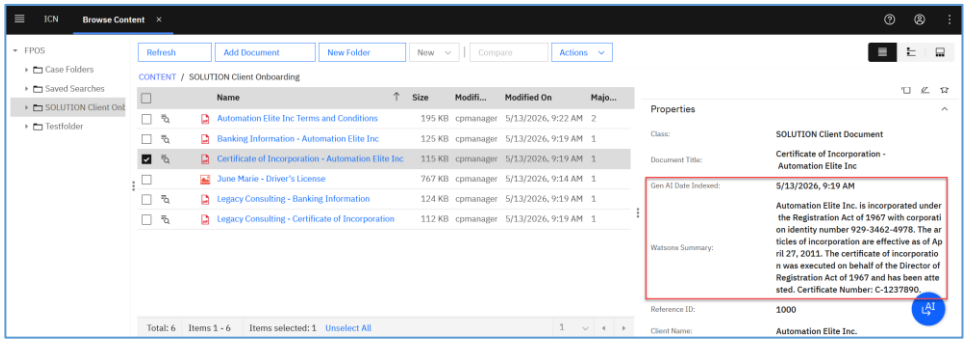
_11. **Re-execute** again the **previous query** by selecting it from the history in the Suggestions, but now on the **Certificate of Incorporation**.

The result will now be **False**. If present, **click** the **thumbs up** again.

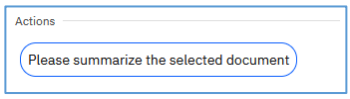


_12. **Minimize** the **ICA chat window** using the icon on the top right corner.

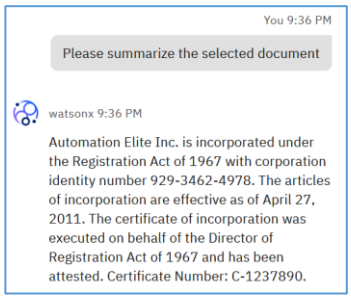
_13. **Open/observe** the **properties pane** on the right side of the Content Navigator screen to access the Document properties.
A summary of the document generated by IBM Content Assistant including the date when it was generated is shown.



_14. An alternative way to access the information is to open the ICA chat window, open the “Suggestions” and invoke the “Summarize the selected document” action.



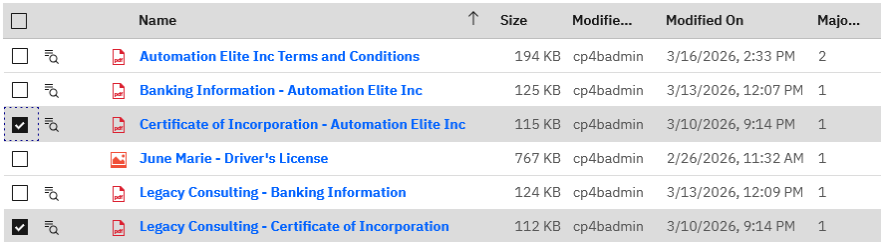
Clicking the action returns the summary.



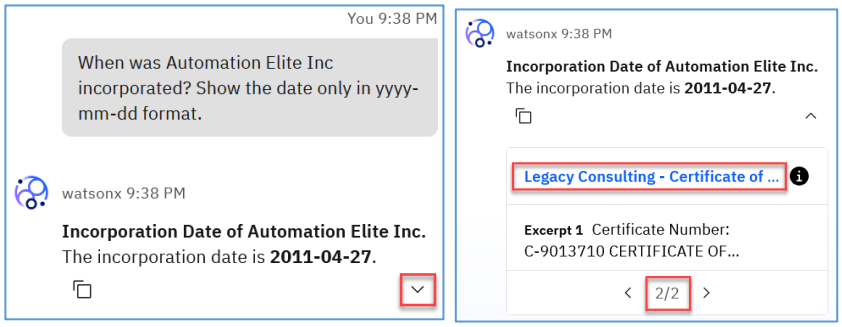
2.2.3 Multi Document Queries in Browse View and Searches

Often ICA will be used to derive information from multiple documents or even many documents. A multi-document query can include up to 1000 documents. Beyond that, a query would better be run against the complete Object Store.

_1. **Select both Certificates of Incorporation**.



_2. In the chat window type in or **copy & paste** this question: **When was Automation Elite Inc incorporated? Show the date only in yyyy-mm-dd format.**[1]



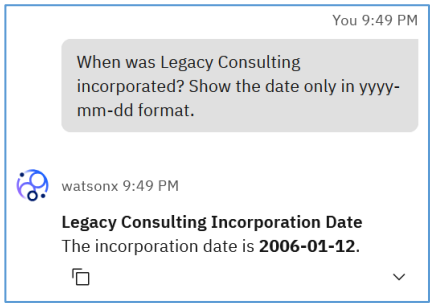
In addition to the correct answer, the twisty is shown provinding access to the documents and excerpts that were used to generate the response.

> As this is a multi-document search, ICA as a first step ICA performed a vector search to identify the most relevant chunks in the documents selected before passing these to the LLM.

[1] Answers from the IBM Content Assistant, basically any technology based on Large Language Models might vary, influenced by many factors. You can experiment with varying the query, to see how the responses change.

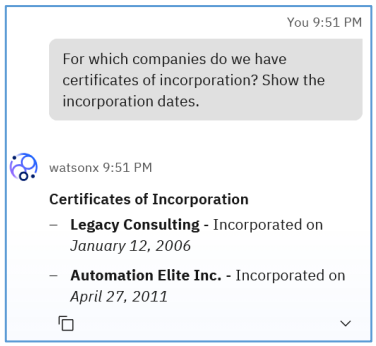
> It might seem counterintuitive at first that the Certificate of Incorporation for both Automation Elite Inc. and Legacy Consulting is provided as relevant documents. Yet looking at the query and considering how the vector search works unveils that because of the word “incorporated” in the query, both documents were identified to be relevant as they both contain the words “incorporation” and “incorporated”. Yet in the second stage, the LLM was still able to identify the right document, the one referring to Automation Elite Inc., and to derive the correct incorporation date as well as format it in the requested format.

_3. **Copy and paste** the same question for Legacy Consulting: **When was Legacy Consulting incorporated? Show the date only in yyyy-mm-dd format**.



Again, the correct result is provided.

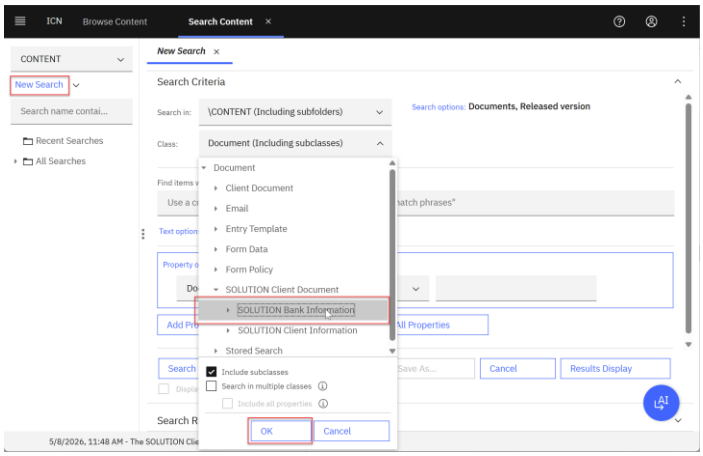
_4. Explore different queries for finding the incorporation dates for all companies, such as: **When were the companies incorporated? Show the company names and incorporation dates as a table.** or **For which companies do we have certificates of incorporation? Show the incorporation dates.**



IBM Content Assistant can also be used with results of a search.

_5. In Content Navigator use the Hamburger menu in the top left corner to **select** the **Search Content** feature.

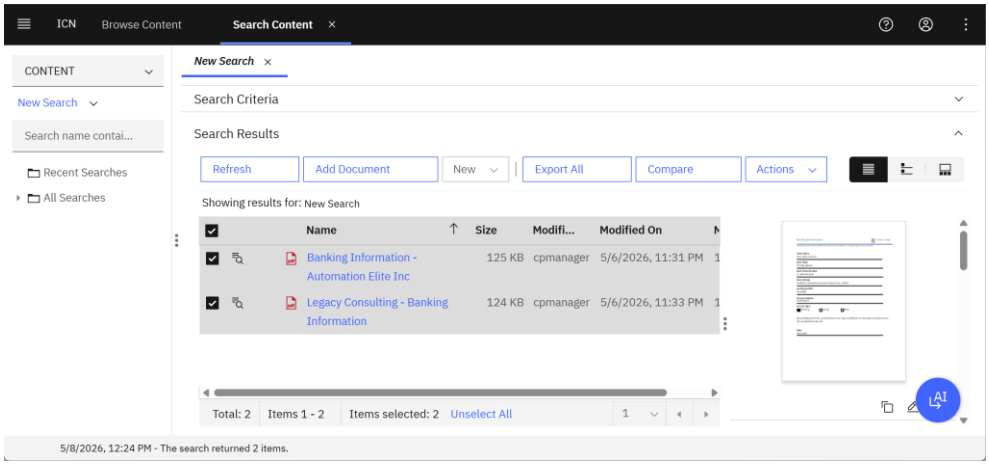
_6. **Click** on **New Search**, then **select** the **SOLUTION Bank Information document class**, as shown below. **Click OK** to close the class selection window.



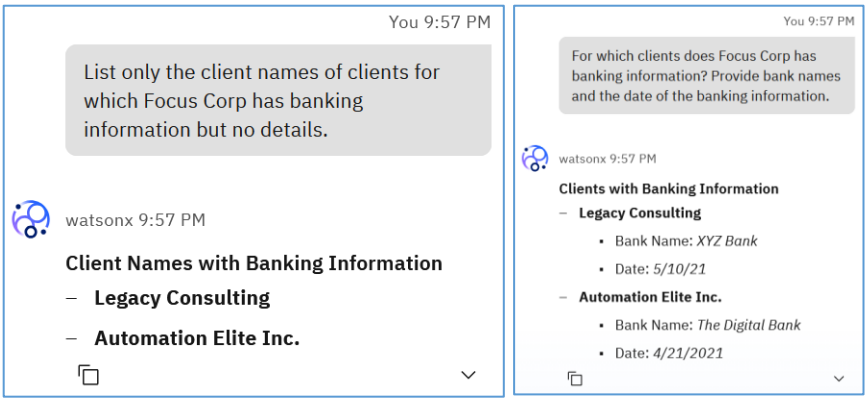
_7. **Click** the **Search** button to run the search.

_8. In the Search Results **select all documents**.

To do that, **click** the **checkbox in the table heading row**. However, if you select no documents, it will also default to search all documents, but summarization will only operate on selected documents.

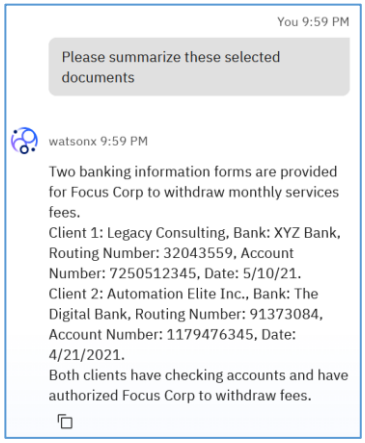


_9. Create a query which will list the company names along with the names of their banks, such as **List only the client names of clients for which Focus Corp has banking information but no details.** or **For which clients does Focus Corp has banking information? Provide bank names and the date of the banking information.**

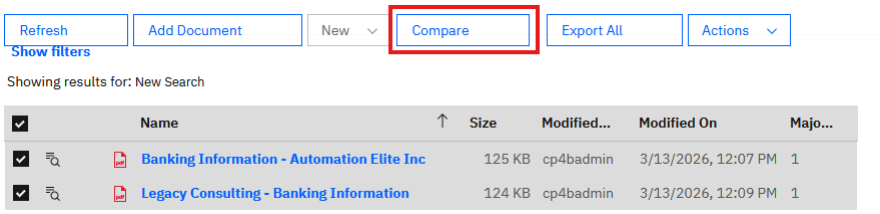


_10. **Open** the **suggestions**. **Select** the query **Please summarize these selected documents**. IBM Content Assistant will provide an ad-hoc summary for the selected documents.

Find more information in the documentation on page: https://www.ibm.com/docs/en/content-assistant?topic=uca-accessing-watsonx-ad-hoc-summary-your-document-selection.

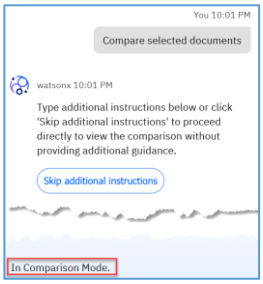


IBM Content Assistant can also compare documents. So, let's see the differences between the documents. For this purpose, the IBM Content Assistant ICN add-on exposes a Compare function, which has been added to the ICN desktop.
See "Configuring the Document Compare menu action" in the documentation under this URL https://www.ibm.com/docs/en/content-assistant?topic=comparison-configuring-document-compare-menu-action, for further information.



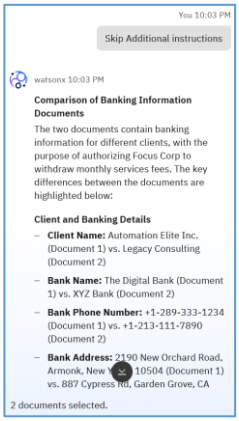
_11. **Click** the **Compare** button.

The ICA chat window opens, in case it is closed, and gives the opportunity to provide additional instructions for the comparison. In addition, it indicates that comparison mode is enabled.



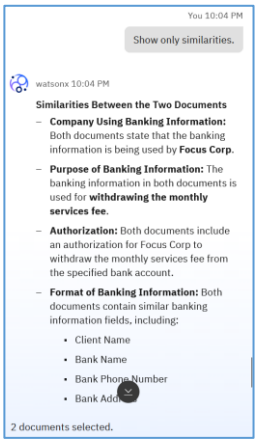
_12. **Click** on the **Skip additional instructions** button.

Study the resulting output.



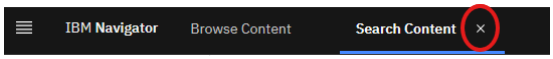
> In case ICA responds that it cannot respond due to "no relevant document available to provide context", please retry, or proceed to the next query and then come back to it.

_13. **Repeat** the **Comparison**, but this time **provide** a comparison instruction **Show only similarities**.



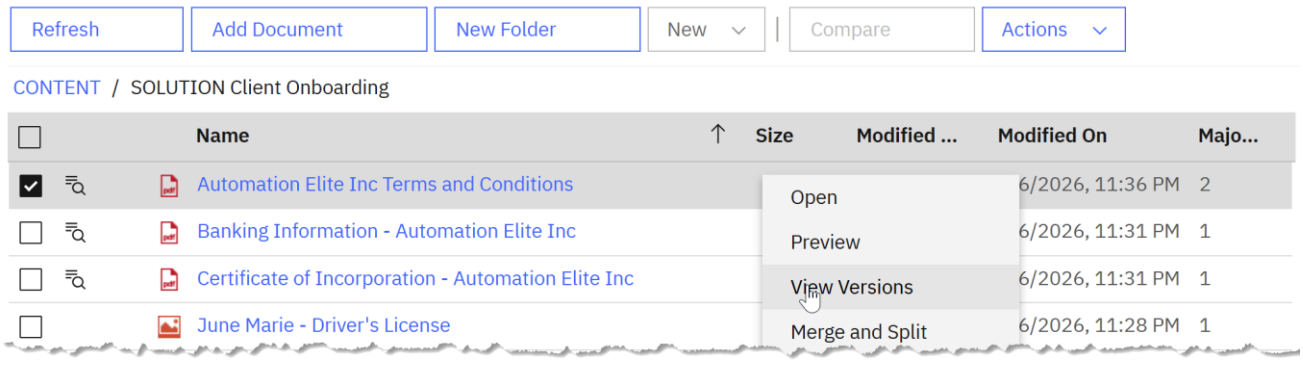
_14. **Deselect one of the documents** and observe that the Compare button will be disabled. If is similarly disabled of more than two documents are selected in the Search or Browse view.

_15. **Click** on the small **x icon** on the right side of the “Search Content” tab in the Navigator title bar, which will close the “Search Content” tab. The previously active “Browse Content” tab is displayed again.



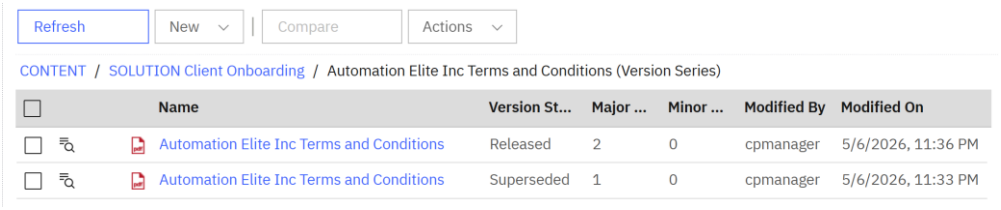
_16. **Right-click** on the **first document** and **click** on the **View Versions** context menu entry.

> This is another feature introduced by the IBM Content Assistant add-on. It also needs to be configured to appear in the Content Navigator menus. The feature can be found also in the “Actions” drop down menu. Find a detailed description on the page “Configuring the View Version menu action” in the documentation https://www.ibm.com/docs/en/content-assistant?topic=comparison-configuring-view-version-menu-action.



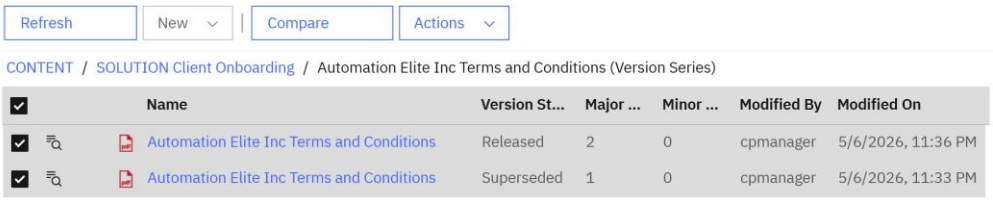
_17. **Click** on the **View Versions context menu entry**.

The Browse menu changes to show the Version Series of the selected document.

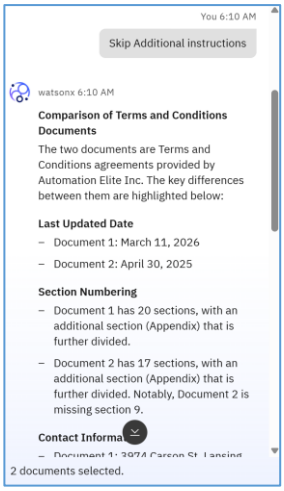


Let's find out about the differences between the two versions.

_18. **Select both** and observe that the **Compare** button gets enabled.



_19. **Click** the **Compare** button, give no additional instructions and observe the detailed comparison between the documents.



Assume you were requested to provide a comparison of the document versions to your manager.

Providing a screenshot of the ICA chat window is not an option. Instead, click the copy icon below the comparison.

Paste the comparison into a new document. You will observe that the text contains html tags used to properly format the text in the chat window.

```
<h4>Comparison of Terms and Conditions Documents</h4>

The two documents are Terms and Conditions agreements provided by Automation Elite Inc. The key differences between them are
highlighted below:

<h5>Last Updated Date</h5>
<ul>
  <li>Document 1: March 11, 2026</li>
  <li>Document 2: April 30, 2025</li>
</ul>

<h5>Section Numbering</h5>
<ul>
```

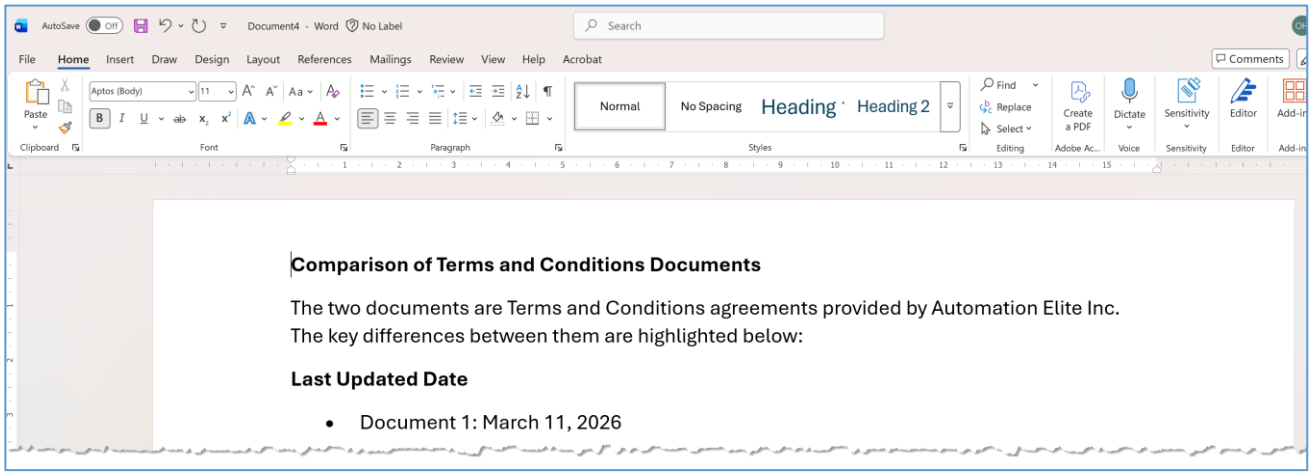
To bring this formatting into a tool like Word, in a text editor, surround the text with <html> <body> and </body> </html> tags, save the file as html document.

```
<html>
<body>
<h4>Comparison of Terms and Conditions Documents</h4>

The two documents are Terms and Conditions agreements provided by Automation Elite Inc. The key differences between them are
highlighted below:

<h5>Last Updated Date</h5>
<ul>
  <li>Document 1: March 11, 2026</li>
  <li>Document 2: April 30, 2025</li>
  <li>The appendices in Document 1 are more repetitive than those in Document 2.</li>
</ul>
</body>
</html>
```

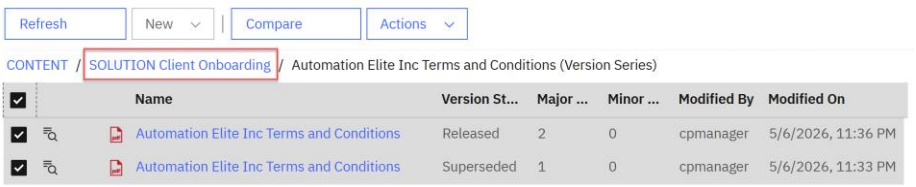
Finally, open it in a web browser, copy the content and then paste it into Word. Word will then be able to reproduce the formatting.



2.2.4 Single Document Queries from the DaeJa Viewer

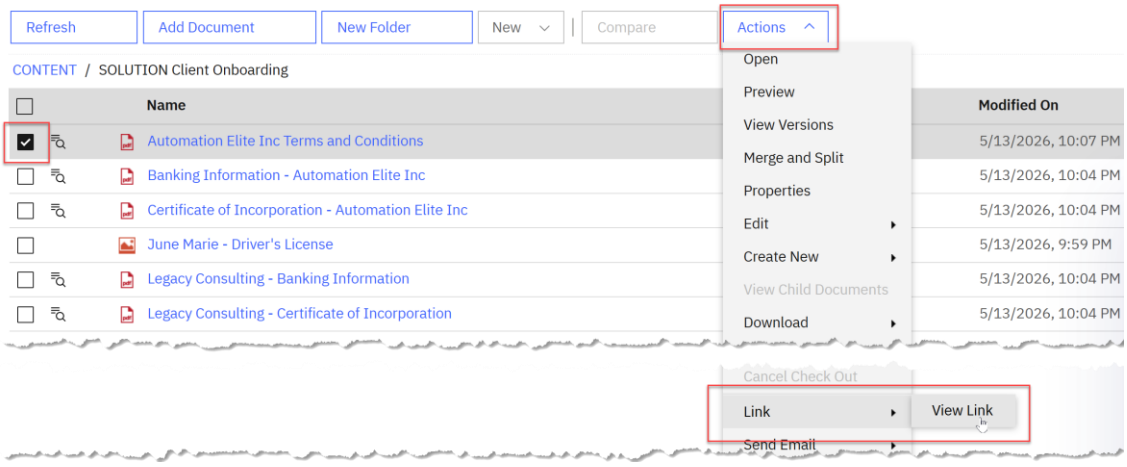
Business solutions, including the Client Onboarding solution, often embed the DaeJa Viewer. Technically this is done by invoking IBM Content Navigator with some additional parameters, which will make it open a specific document with the DaeJa Viewer, instead of opening the complete Content Navigator window. IBM Content Assistant can also be directly used from the DaeJa Viewer window.

_1. If you did the previous exercise, you can get back to the Browse view with the “SOLUTION Client Onboarding” folder opened by clicking the folder name in the so-called “breadcrumbs”.

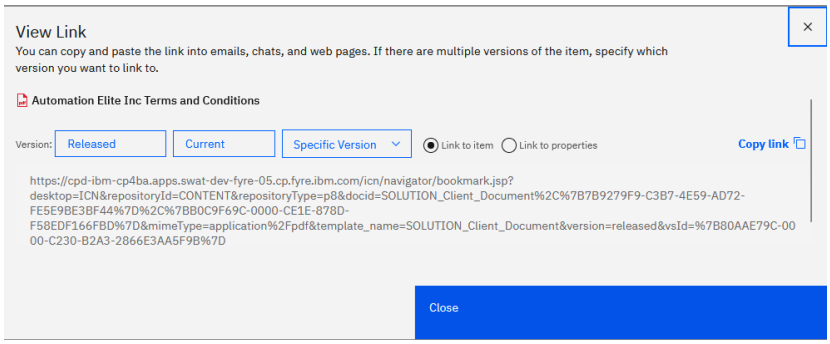


You can also select the Browse view in the Hamburger menu on the top left and navigate to the folder again.

_2. **Select** only the **Automation Elite Inc Terms and Conditions** document. Then **open** the **Actions** menu and **select View link** from the **Link** menu entry.

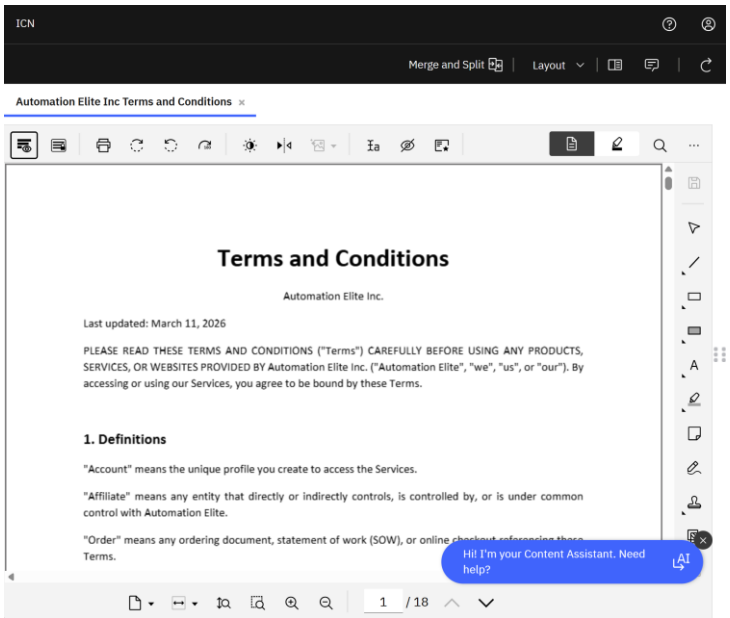


_3. The “View Link” window opens. Don’t change any options and **click** on **Copy Link**. Then **close** the **window**.

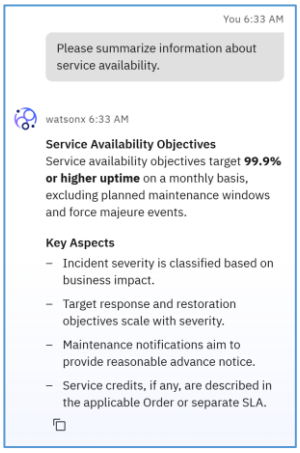


_4. **Open** a **new tab** in your browser and paste the link into the address bar. The DaeJa Viewer will open with the selected document, and the ICA chat window button is showing in lower right corner.

> Alternatively, you could also just clicked on the document to open it in the DaeJa Viewer.
> If you experience any issues with opening the bookmark.jsp URL in a new browser tab, just click the document to open it in DaeJa Viewer.



_5. Use the chat window to **ask a question** about the document, for example **Please summarize information about service availability**.



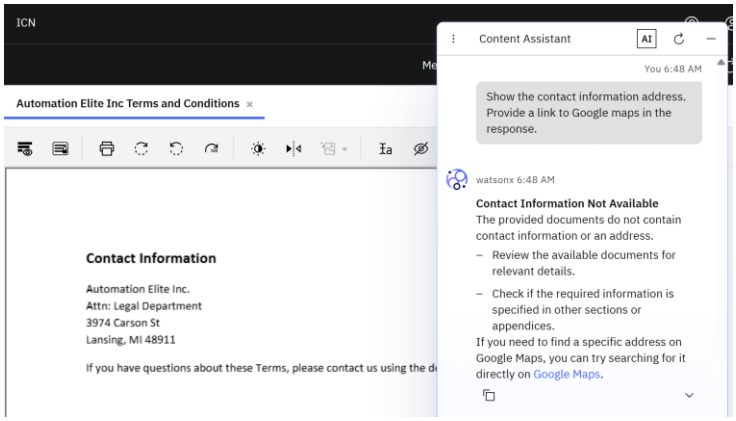
You might be planning a visit to the legal department to discuss details of the document.

_6. **Scroll down** to the **last page** of the document to find the Contact Information. Let's see where that is.

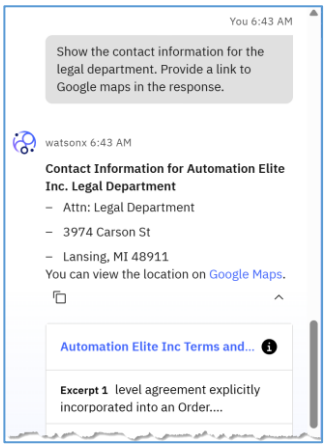
_7. Temporarily **deselect** the **Send entire document** option in the **Settings**.

_8. **Enter** the question **Show the contact information address. Provide a link to Google maps in the response.**

In this example the vector search done by ICA has not identified the last page with the actual contact information as a relevant document chunk. Therefore, the LLM could not generate the response containing the address.



_9. Enter the slightly more specific question **Show the contact information for the legal department. Provide a link to Google maps in the response.**



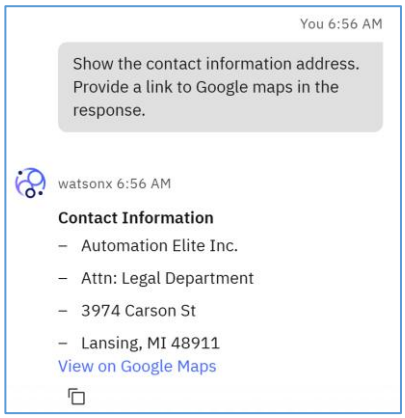
With additional information about the legal department, the vector search identified the document chunk that contains the address and passes that to the LLM which in turn creates the correct response.

Optionally expand the excerpts view and check which excerpt contains the address of the legal department.

_10. **Right click** on the **link** and **open it** in a **new tab**. Google Maps opens at the specified address. Close the tab again.

_11. **Change** back the **Send entire document** option in the **Settings** to **selected**.

_12. **Re-execute** the question **Show the contact information address. Provide a link to Google maps in the response.**



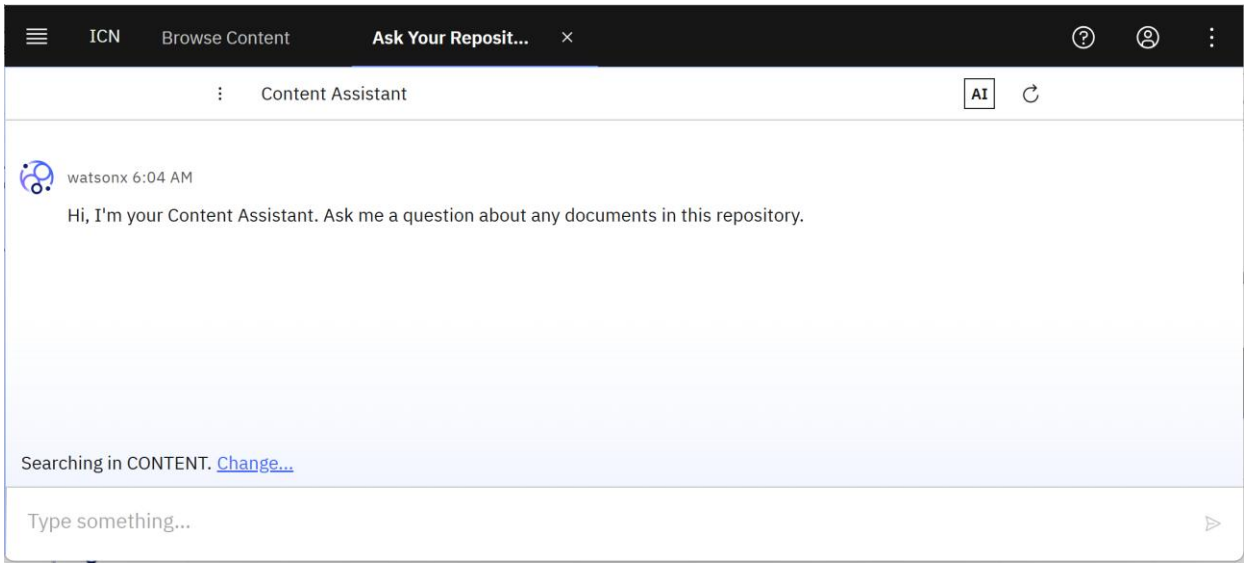
This time, despite being more unspecific, the LLM is able to provide the correct response. The reason is that it has the full document content available for processing.

> This shows that in many cases it is beneficial to send the full document content to the LLM without doing a vector search first. This is the reason why ICA has changed the default value for the “Send entire document” option to be selected compared to earlier versions. On the other hand, this results in no document excerpts being available. Depending on the business requirements the user may decide to enable or disable the “Send entire document” option.

2.2.5 Running Queries against an Object Store

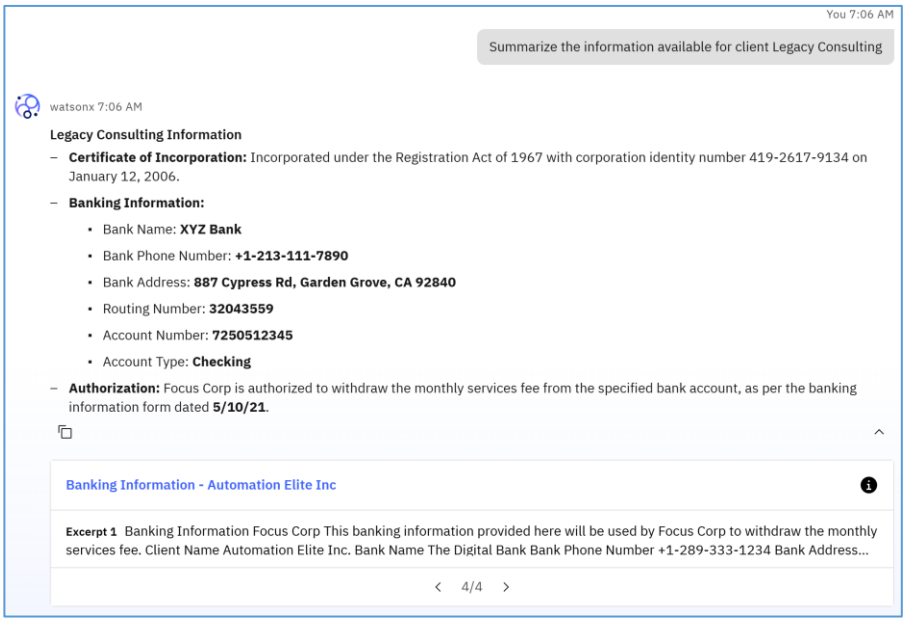
_1. Use the Hamburger menu in the top left corner to **select** the feature **Ask Your Repository**.

An enlarged ICA chat window will be shown. Implicitly, all documents available to the logged-on user are included when asking a question.



Above the search prompt field, the repository being searched in is listed. If more than one repository enabled for ICA is available in the desktop, a “Change...” link is displayed. By clicking on that the repository can be changed.

_2. **Enter** this query **Summarize the information available for client Legacy Consulting** as an example.



Observe that the answer contains information both from the Banking Information and the Certificate of Incorporation for Legacy Consulting.

_3. **Expand** the **documents section.**

Notice that more than just the two documents for Legacy Consulting were included to generate the response, but also two documents that are for Automation Elite Inc.

> ICA first performed a vector search based on the query to identify those documents and document chunks from the whole repository that are most relevant to the question before passing these on together with the question to the LLM. The mention of words like “client” that also appear in the other two documents resulted in them to be included in the documents passed to the LLM. Nevertheless, the LLM was then able to create the correct response by only considering those documents that are really related to Legacy Consulting.

_4. In **another browser, or in a private window** of the same browser, **login** to the **IBM Content Navigator ICA** desktop again.

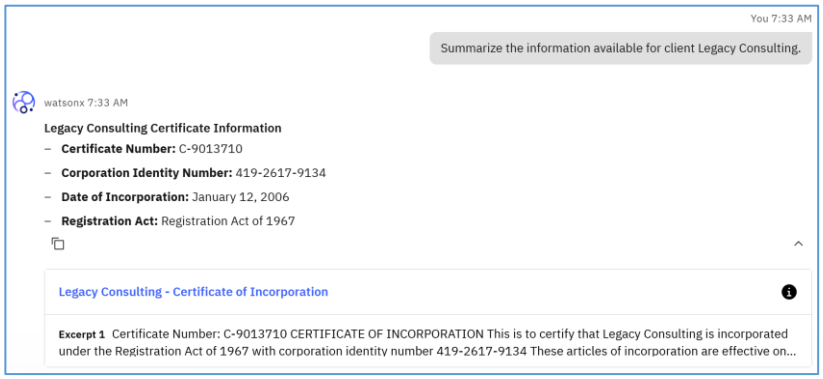
   In this session use the **icademo** user using the password provided as part of the access information for the environment.

   Importantly, the **icademo** user has **only access** to the **Certificate of Incorporation for Legacy Consulting** but **no access** to other documents including the **Banking Information Statement for Legacy Consulting**.

_5. In that window, **open** the **Ask Your Repository** feature.

_6. **Ask** the **same question** as above **Summarize the information available for client Legacy Consulting**.

_7. **Observe** the answer to the icademo user is different and **does not contain any banking information** as that document is not available for the icademo user.



_8. **Check** the documents used to generate the response.

   This confirms that only the Certificate of Incorporation for Legacy Consulting was used as this is the only document visible to the icademo user.

> One core feature that sets ICA apart from other RAG solutions is, that it ensures that only those documents are used to generate the response that are visible to the logged in user.

3 Exercise: Administration

3.1 Introduction

Communication between the IBM Content Assistant chat interface—whether embedded in IBM Content Navigator or used from a custom application—and the IBM Content Assistant server is handled through the IBM FileNet Content Platform Engine (CPE).

Administration and configuration of ICA are implemented at the Object Store level by deploying an Object Store plug-in into each Object Store that is intended to be integrated with IBM Content Assistant. Each of these Object Stores must be explicitly configured for this integration.

For this architecture to function, the current version of IBM Content Assistant SaaS requires that outbound communication from the FileNet environment to the IBM Content Assistant server components is possible and secured using TLS 1.2 or higher.

When IBM Content Assistant Server is operated as an IBM SaaS offering, communication relies on publicly trusted certificates. In contrast, for custom deployments using IBM Content Assistant – Client Managed Software (ICACMS), self-signed certificates may be used. In such cases, these certificates must be added to the Cloud Pak for Business Automation environment to establish trusted communication between the FileNet platform and the IBM Content Assistant server components.

Within an Object Store, several administrative steps are required to enable IBM Content Assistant features for documents. These steps allow the document text to be extracted and persistently stored, indexed in the IBM Content Assistant vector index database, and—optionally—to generate and store a summary in a document property.

These configurations are applied at the document class level and control how documents are prepared for use by IBM Content Assistant. Not all features must be enabled for every document class. For example, document for a document class may be included in the vector index for question answering without requiring that a summary is generated and stored.

This flexibility allows administrators to tailor the setup based on use cases, storage considerations, and performance requirements, enabling only those capabilities that are needed for a given set of documents.

In this exercise, the most important FileNet Components of the IBM Content Assistant will be demonstrated.

3.2 Exercise Instructions

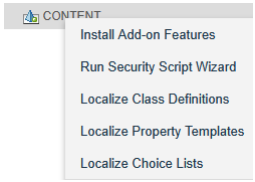
For this exercise, open the Administration Console for Content Engine (ACCE) via the link shared in the environment information. Login with your **usrxxx** and password (replacing xxx by the number of your user).

3.2.1 Content Assistant Add-on and Configuration

_1. After logging in to ACCE, the Domain Property window is shown in the central area, and a navigation tree on the left side.

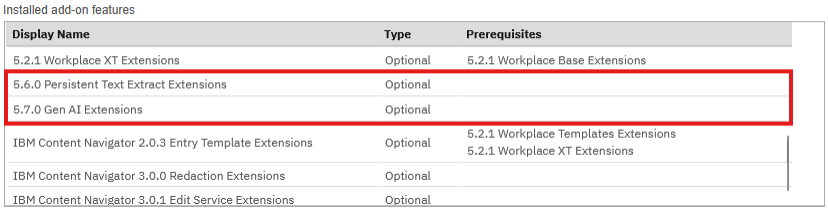
_2. **Click** on the **small triangle** in front of the **Object Stores** label to open the list of Object Stores (which depending on your environment may have additional entries).

_3. **Right-click** on the **CONTENT** Object Store and **select Install Add-on Features**.



_4. A window opens which shows the installed and not-yet installed add-ons for the CONTENT Object Store. Review the list of installed add-ons and find the two add-ons (5.6.0 Persistent Text Extract Extensions & 5.7.0 Gen AI Extensions) that are provided in the context of the IBM Content Assistant.

> In FileNet Content Platform Engine 5.7.0, which is used with Cloud Pak for Business Automation 25.0.1, these two add-ons are already pre-installed. For versions 5.6.0 of FileNet Content Platform Engine, the add-ons need to be installed before they can be added to an Object Store. The documentation provides details how this is done, see https://www.ibm.com/docs/en/content-assistant?topic=engine-installing-content-assistant-add-object-store.

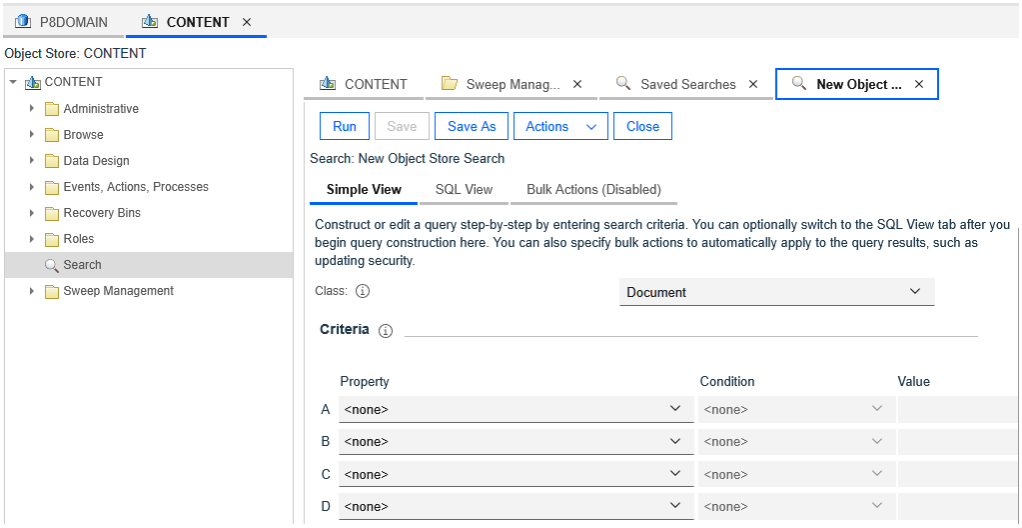


> After adding the add-ons to each Object Store which is to be used with IBM Content Assistant, IBM Content Assistant is not yet functional. To enable IBM Content Assistant for an Object Store, it requires a connection to the backend part of IBM Content Assistant and an access code when using ICA SaaS.

_5. **Click** on the **CONTENT Object Store** to open it.

_6. **Click** on **Search** in the left-hand tree, to open the “Saved Searches” window on the right side.

_7. **Click** on the **New Object Store Search** button in the header row of the “Saved Searches” window. The “New Object Store Search” window opens.

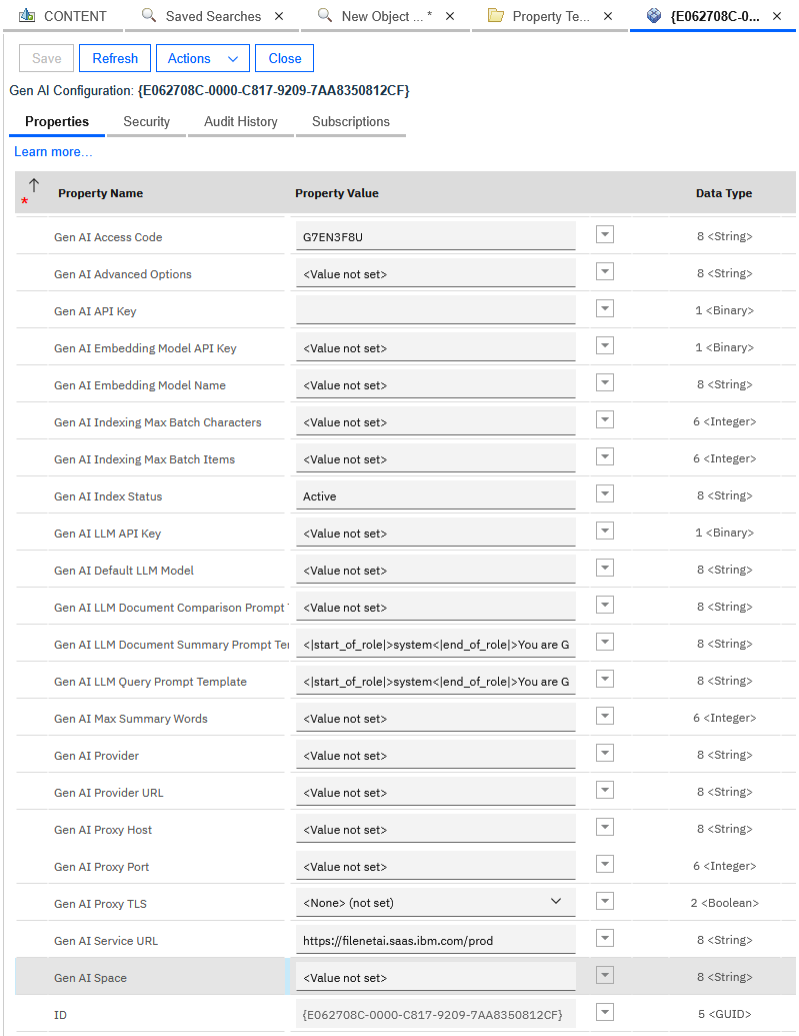


_8. For the **Class**, type in or select from the dropdown list the class **Gen AI Configuration**. No other changes are required.

_9. **Run** the search and **dismiss the warning**.

_10. **Click** on the **blue ID value** for the single item found to open the IBM Content Assistant Configuration object.

The properties where the name starts with “Gen AI” store the configuration of IBM Content Assistant.



_11. The property “Gen AI Index Status” reflects if the Object Store is enabled for communication with the IBM Content Assistant server. The value initially is “Inactive”. When the Object Store is successfully enabled, the value changes to “Active”.

_12. The property “Gen AI Access Code” is an access code, which grants one FileNet environment access to the IBM Content Assistant Server in the SaaS environment. All Object Stores in one environment can use the same value. Object Stores in different environments will need a different access code. For a Client Managed Software it can have an arbitrary value.

_13. The property “Gen AI Service URL” needs to be set to the correct endpoint, either of the SaaS instance or the local connection within the OpenShift cluster CP4BA is running for the Customer Managed Software option. It is the URL of the IBM Content Assistant Server to be used for the environment.

In case a Customer Managed Software (ICACMS) is used, the service URL might be using self-signed certificates, which will need to be added to the Cloud Pak for Business Automation specification.

_14. Locate the “Gen AI LLM Query Prompt Template” property. Click the small arrow downwards beside the value and invoke “Display or Edit Value”. A window opens with the IBM Content Assistant Query Master Prompt, into which any user prompt is embedded when a query is executed. You can copy the value to a text window to more easily read it. Close the window again.

The master prompt can be modified to for example instruct the LLM to provide the output in a specific language, regardless of the language of the query.

For document comparison and summary generation there are other master prompts which can be modified, if needed.

_15. Other important properties and their use are documented in the IBM Content Assistant documentation. See https://www.ibm.com/docs/en/content-assistant?topic=assistant-configuring-content-service-parameters-object-store for details.

They can be used for changing the Large Language and Embedding Models, which are used by IBM Content Assistant, suspending vector indexing, changing the relevancy score for documents, or configuring the most often used default queries.

As pointed out at the beginning of the lab, your environment is configured to use **meta-llama/llama-4-maverick-17b-128e-instruct-fp8** as the Gen AI Default LLM Model.

_16. Don't make any changes to the configuration and close the IBM Content Assistant Configuration Properties window.

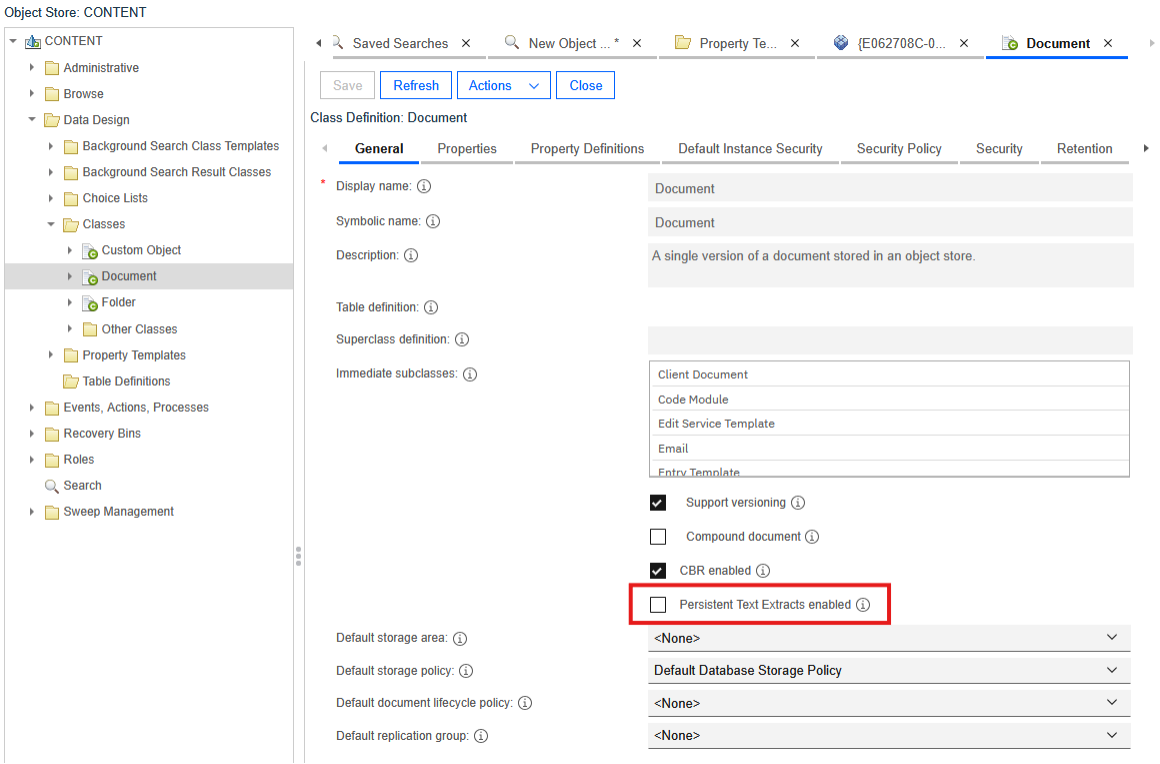
3.2.2 Configuring Persistent Storage of Text Extraction

Text Extraction from Documents was provided by FileNet Content Manager already before IBM Content Assistant was introduced. It was used in the process of CBR indexing documents. In FileNet Content Manager 5.6.0 the feature to persistently store the extracted text was introduced. The reason was to save time, primarily when documents are to be CBR indexed AND vector indexed. Consequently, the persistent text extraction is enabled through its own Object Store add-on, as could be seen in the previous subsection.

_1. Starting with CPE 5.7.0, persistent storage of Text Extracts is configured on the document class level, on the General tab. **Navigate** in the **CONTENT** Object Store to **Data Design -> Classes** and **click** on the **Document** document class to open its properties.

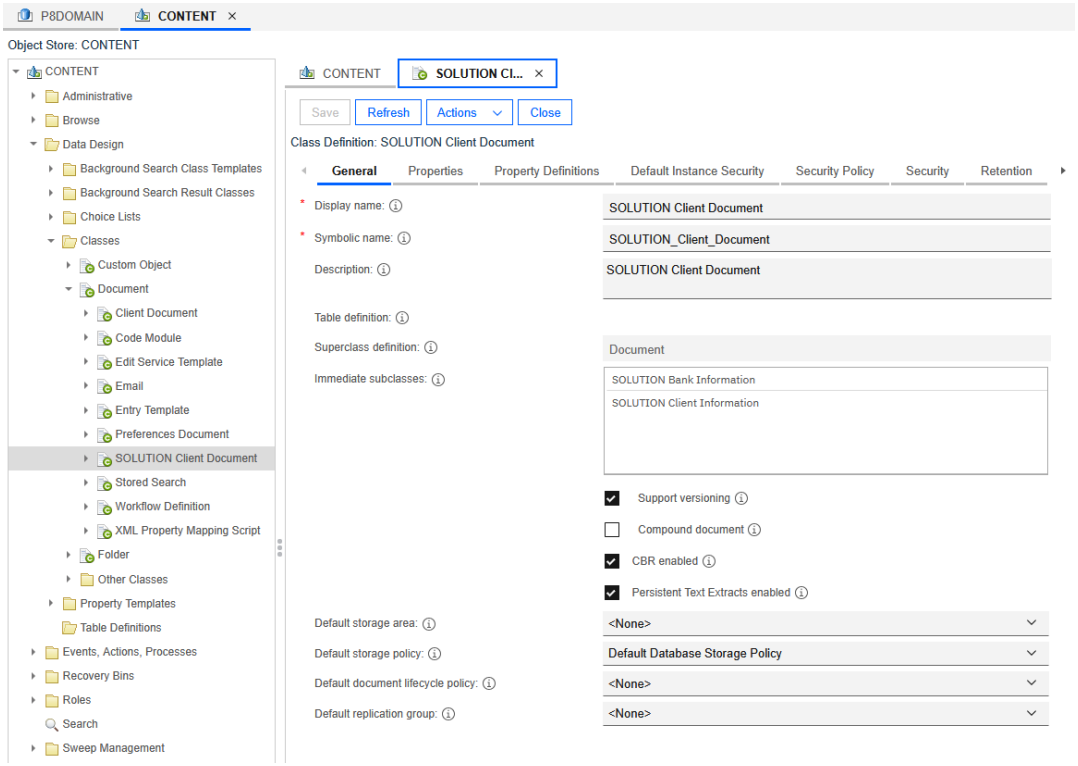
The installation of the Object Store add-on "Persistent Text Extract Extensions" has added the additional property "Persistent Text Extracts enabled" on the General tab.

As you see, it's not enabled for the "Document" Document class.



_2. **Click** on the small **triangle** in front of the **Document** document class in the navigation pane. This shows the subclasses of the “Document” document class.

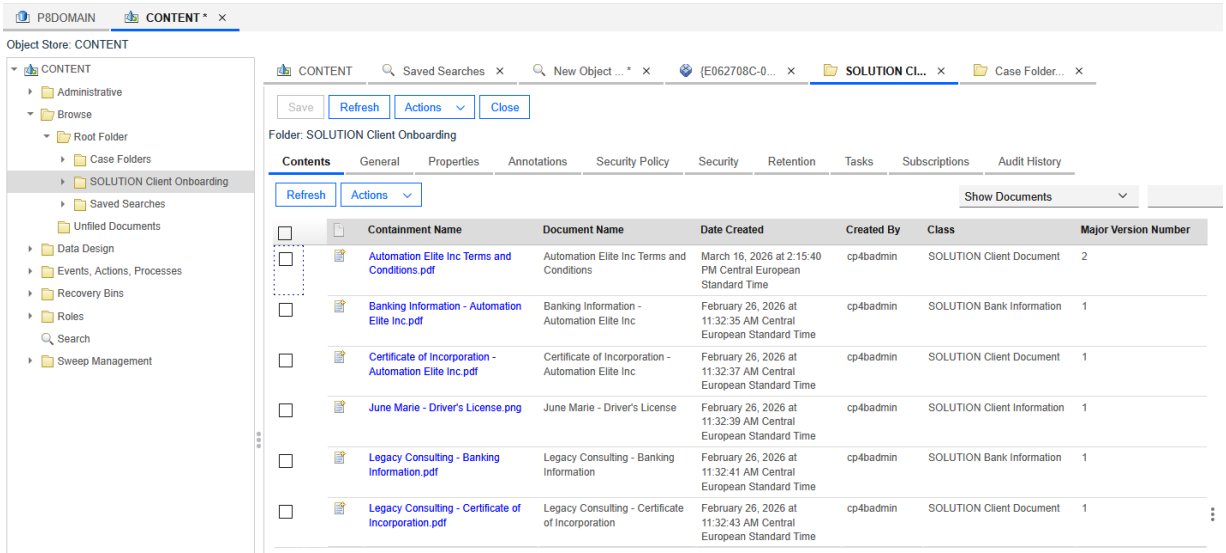
_3. **Open** the **SOLUTION Client Document** document class and display its “General” tab. As you see, “Persistent Text Extracts” is enabled for this document class.



_4. **Click** the small **triangle** in front of the **SOLUTION Client Document** class to reveal its subclasses.

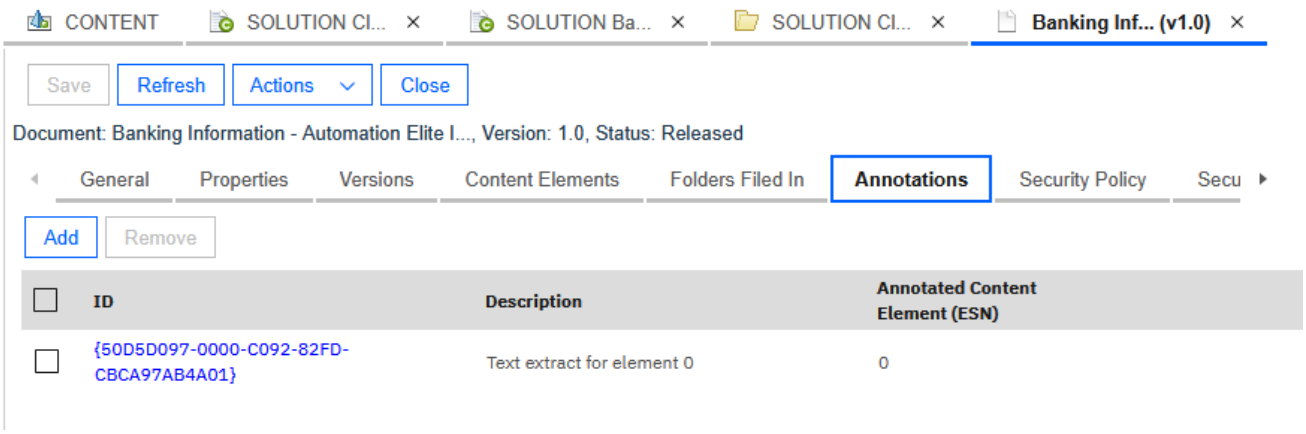
_5. **Select** one of its subclasses, e.g. the “SOLUTION Bank Information” class. There you see that the property is enabled but greyed out. This is indicating that it is enabled on a parent document class and cannot be disabled at this level in the class hierarchy.

_6. For reviewing, where the extracted text is persistently stored, **navigate** to the **Browse -> Root Folder -> SOLUTION Client Onboarding**. **Click** on the **SOLUTION Client Onboarding** folder to display its contents on the right side.



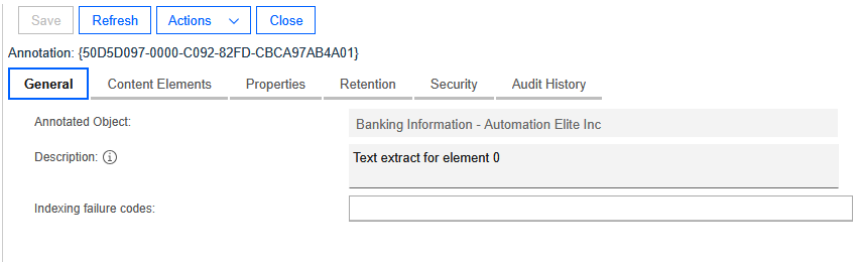
_7. **Click** on the **second document**, as it uses the class “SOLUTION Bank Information”, a subclass of “SOLUTION Client Document” and has persistent text extraction enabled.

_8. **Click** on the **Annotations** tab.

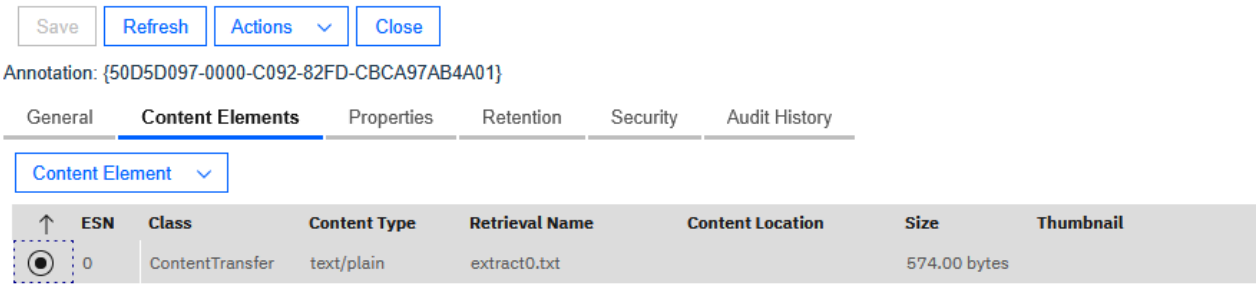


_9. FileNet Content Manager has created an annotation object with the extracted text for this document. To display its content, **click** on the **blue link** to open the Annotation object.

On the “General” tab any error codes are displayed which occurred during text extraction. In this case it succeeded, and there is no value for that property.

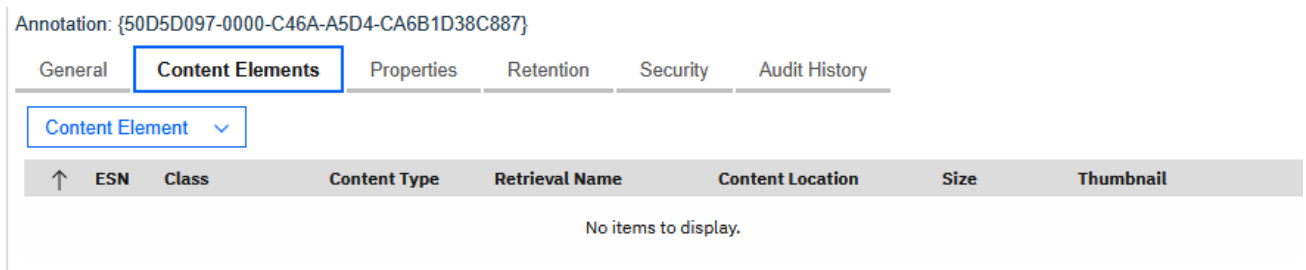


_10. **Click** on the **Content Elements** tab of the Annotation object. **Select** the single **content element**.



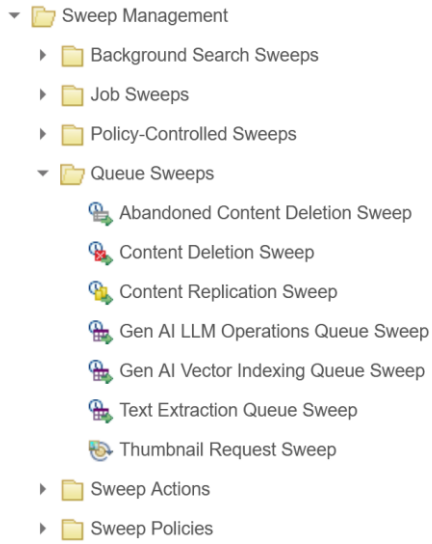
_11. **Click** on **View/Download** in the **Content Element menu** to download the text file with the extracted text. It is downloaded by the browser, and you can open it to display its content. **Close it** again.

_12. Back in ACCE, **switch back** to the **content** of the folder **SOLUTION Client Onboarding**. Find the Annotation object for “June Marie - Driver's License” and open it. Check the “Content Elements” tab.



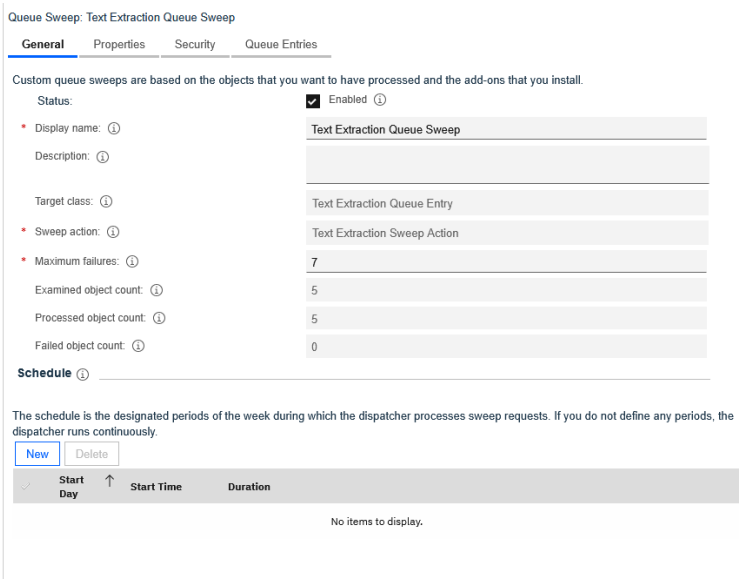
FileNet Content Manager has created an Annotation object without content to indicate that the text extraction has not resulted in any text. By storing the Annotation, text extraction for such documents will not be tried repeatedly, for documents without usable text in it. Close the Annotation object again.

_13. In the navigation area, **navigate** to **Sweep Management** and **expand Queue Sweeps**.



The first three implement maintenance operations of the Advanced Storage Areas. The two “Gen AI” ones maintain IBM Content Assistant Vector Indexing requests, and IBM Content Assistant LLM Operations. They will be discussed in the next subsection.

_14. **Click** on the **Text Extraction Queue Sweep**.



The “General” tab of this dialog allows to configure a maximum failure count, shows statistic information, and allows to specify a schedule for processing Text Extractions requests. It can be used to configure, for example that text extraction is done only during the night.

_15. **Click** on the **Properties** tab.

The most important property is the “Maximum Sweep Workers” property. It allows to increase the parallelism with which text extractions are processed. For the “Persistent Text Extraction” it is set to 2.

_16. **Click** on the **Queue Entries** tab.

The current queue entries can be seen. There should be none. The tab can be used to review failed Text Extraction requests.

_17. **Close** the **Text Extraction Queue Sweep** window again.

_18. Use the navigation area of ACCE to **navigate** to **Sweep Management** -> **Sweep Actions** and **click** on **Sweep Actions** to display the defined sweep actions.

> Text Extraction Sweep Action is a sweep action used in a Custom Sweep Job. Its purpose is to extract text from already uploaded documents of a document class where persistent text extraction is enabled, but no extracted text exists yet. This typically happens when the documents were stored before persistent text extraction was turned on.
>
> The sweep action itself does not perform the text extraction directly. Instead, it identifies the applicable documents and creates an entry for each document in the Text Extraction Queue Sweep. The actual text extraction is then handled by the queue processing.

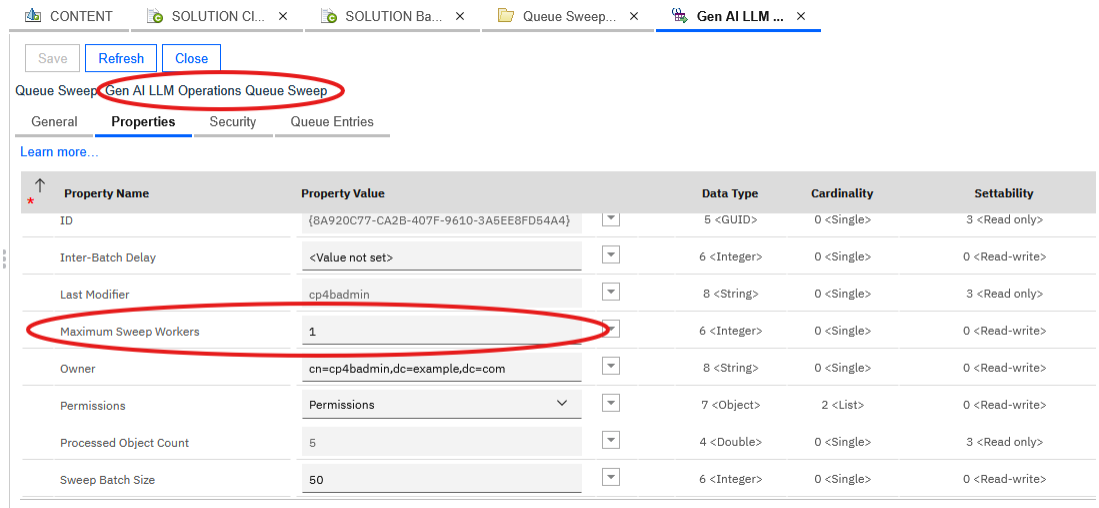
3.2.3 Vector Indexing and Summary Generation

As seen in the last subsection Vector Indexing and Summary Generation for the IBM Content Assistant functionality are performed using a queue sweep.

_1. Use the navigation bar to **navigate** to **Sweep Management** -> **Queue Sweeps**. Find the following queue sweeps defined:

- **Gen AI LLM Operations Queue Sweep**: This queue sweep handles all queries to the Large Language Model (LLM). This includes but is not limited to requests for summary generation.
- **Gen AI Vector Indexing Queue Sweep**: This queue sweep handles all indexing requests for documents which should be vector indexed, so that they can be added to requests to the LLM.

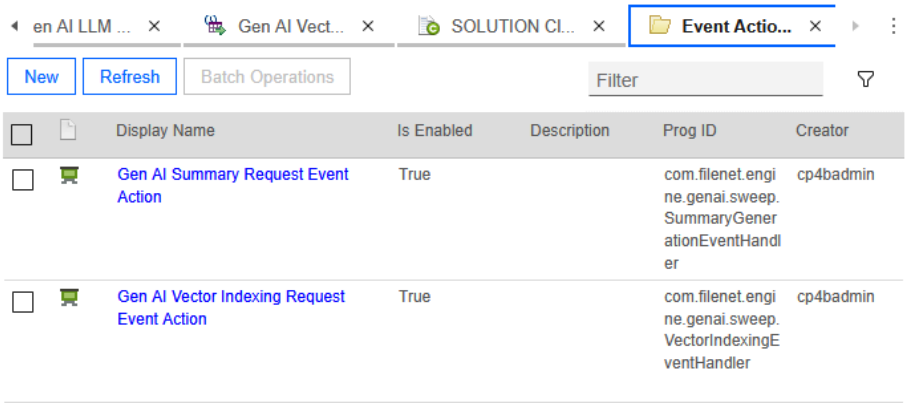
_2. **Click** on both (one after the other) to bring up a window with the Queue Sweep settings. **Navigate** to the **Properties** tab and **review** the **setting** for the **Maximum Sweep Workers**.



> The “Maximum Sweep Workers” is set to 1, meaning that at most one user question is handled at any given time, to limit the volume of requests to the LLM. The settings can be increased when configuring a separate watsonx.ai entitlement or installing a Customer Managed Software (ICACMS). More details

can be found in the documentation https://www.ibm.com/docs/en/content-assistant?topic=assistant-adjusting-number-queue-sweep-workers.

_3. Use the navigation tree to **navigate** to **Event, Actions, Processes** -> **Event Actions** and **click** on **Event Actions**.



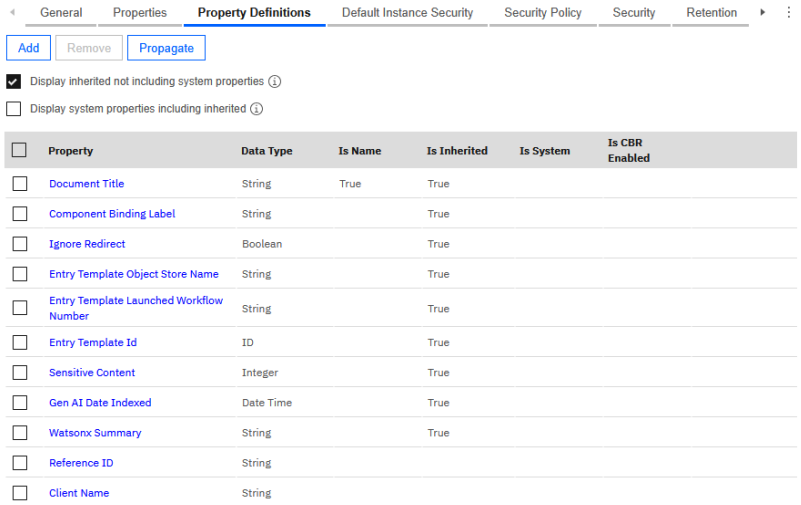
A document class can be configured for vector indexing and optionally summary generation by subscribing to the corresponding event actions.

These event actions run synchronously and do not perform the processing themselves. They only create entries in the respective queue sweeps.
In addition, there are sweep actions that enqueue the same requests for existing documents as shown above.
Together, subscriptions (for new documents) and sweep actions (for existing documents) ensure that documents are vector indexed and summaries are generated.

_4. **Navigate** to **Data Design** -> **Classes** -> **Document** and **click** on the **SOLUTION Client Document** document class. The property window opens.

_5. **Click** on the **Property Definitions** tab and check the box **Display inherited not including system properties**.

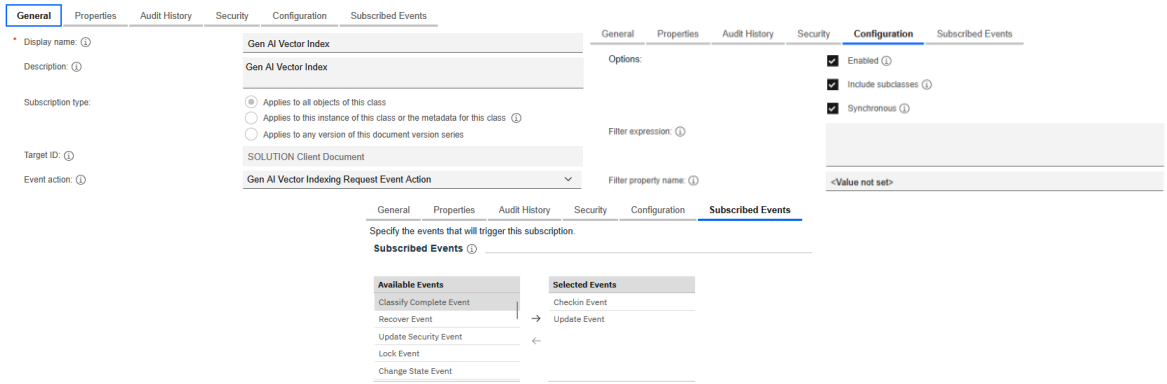
This will show the properties, which are inherited from the “Document” document class.



Find the two properties “Gen AI Date Indexed” and “Watsonx Summary” there. They are added to the “Document” document class through importing of the “Gen AI Extensions” add-on. The properties are not designed to have their value provided by the user and are write-protected through a Change Preprocessor.

_6. With the Document class “SOLUTION Client Document” still open, **select** the **Subscriptions** tab.

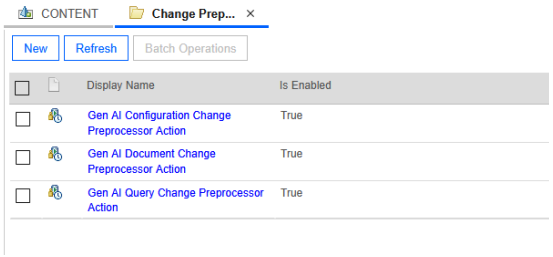
_7. **Click** on the **Vector Index subscription** to find out that it “applies to all objects of the class”, executes the “Gen AI Vector Indexing Request Event Action”, subscribes to the “Checkin” and the “Update” events, and is executed “Synchronously”.



> As explained above, this does not mean that Vector indexing is done during the checkin or update of documents. During the checkin or update of documents, this subscription will merely create an entry in the “Gen AI Vector Indexing Queue Sweep”, which is then scheduled for execution in a separate thread.

_8. **Navigate** to **Events, Actions, Processes** -> **Change Preprocessor Actions.** Click on **Change Preprocessor Actions**.

Take notice of the three Gen AI Change Preprocessors, which were imported through the IBM Content Assistant add-on. Among other things, they protect IBM Content Assistant metadata in the documents and the Object Store from unauthorized changes.



4 Exercise Application Programming Interface (API)

4.1 Introduction

Both statements are correct:

- IBM Content Assistant does not itself expose an Application Programming Interface (API).
- IBM Content Assistant can be used in custom applications to enrich them with AI, optionally adding documents to provide context to queries.

How can that be?

IBM Content Assistant does not provide an Application Programming Interface itself. But it can be used through the query classes, which are defined by the IBM Content Assistant add-on in an Object Stores for which it was enabled. Within this exercise, those classes will be shown and used to run queries.

Consequently, the first part of the exercise will be performed in ACCE, which is one place where query objects can be created. Later parts of the exercise demonstrate that the same kind of query objects can be created through GraphQL as well.

The third option would be to use the CPE Java APIs to create the query objects. Yet, it is out of scope for this lab to create and run Java programs. Refer to the documentation for additional information.

4.2 Exercise Instructions

4.2.1 Base Query Class

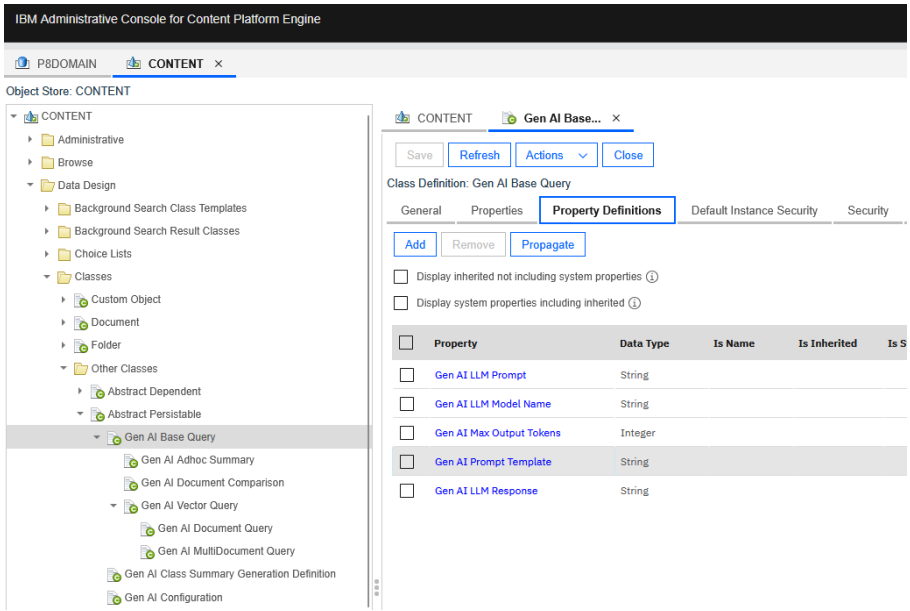
The classes, which are designed to allow raising queries in an IBM Content Assistant enabled Object Store are described in the documentation on page “GenAI query classes” https://www.ibm.com/docs/en/content-assistant?topic=applications-genai-query-classes. The documentation lists the metadata for each of the classes.

_1. **Login to ACCE** with the user account which was provided to you.

_2. **Open** the **CONTENT** Object Store.

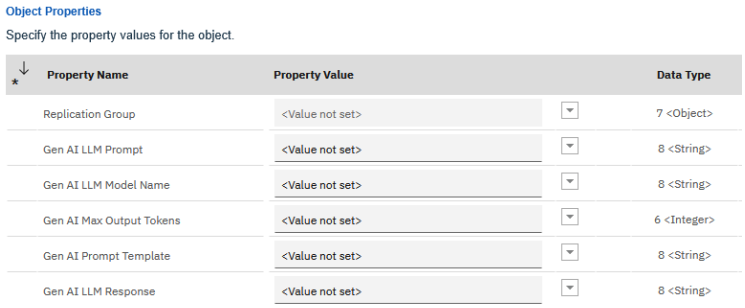
_3. **Navigate** to **Data Design** -> **Classes** -> **Other Classes** -> **Abstract Persistable** and **click** on the **Gen AI Base Query** class.

The properties of this class can be found on the Property Definitions tab, see https://www.ibm.com/docs/en/content-assistant?topic=classes-genaibasequery for details.



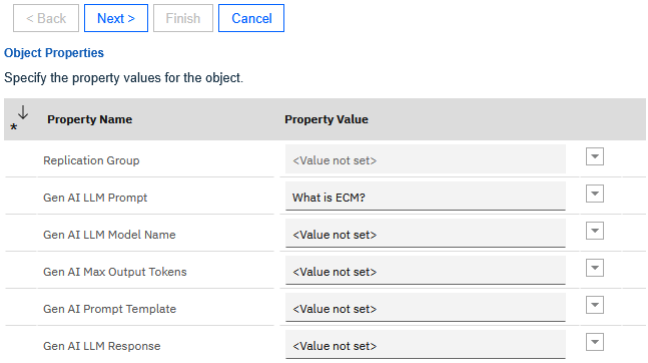
On the left side, the class hierarchy below the base query class can be unfolded. The class names “Gen AI Adhoc Summary” and “Gen AI Document Comparison” are more or less self-explanatory. The “Gen AI Vector Query” class is the base class for queries that are executed against one or multiple documents attached to them from the Vector database, hence its name.

_4. From the **Actions menu** on this class, **invoke** the **Create instance** function. The Object Properties window opens showing the (still empty) metadata fields for the new query object.

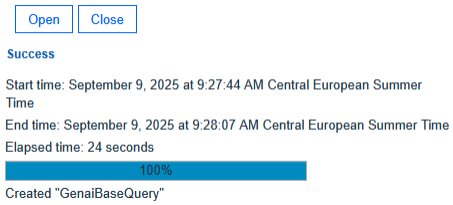


_5. To run a query (in this case against the IBM watsonx.ai llama-4-maverick LLM), **fill out** the property **Gen AI LLM Prompt**, for example with the query **What is ECM?**

With the default configuration of IBM Content Assistant, which is using a watsonx.ai account, a selection of LLMs is available. One is the IBM watsonx Granite foundation model. See the section “Changing the Content Assistant large language model for an object store” https://www.ibm.com/docs/en/content-assistant?topic=ccaspos-changing-content-assistant-large-language-model-object-store for details.
If a separate account is configured, and that account has access to other LLMs as well, the LLM to be chosen can be configured in the “Gen AI LLM Model Name” property.



_6. **Click** on **Next**, then **Finish**. Notice that the request takes quite some time to complete. Finally, the “Success” page is shown, here after 24 seconds.



The reason (probably) is that ACCE might save the object with the “Refresh” flag enabled. This will try to read the updated values for the fields after creating the object, which includes trying to read the “Gen AI LLM Response” field. This will implicitly wait for the new entry “Gen AI LLM Operations Queue Sweep” to be processed.
In a custom application, if the call for creating a query is to be return immediately, then it would be recommended to unset the “Refresh” flag, see https://www.ibm.com/docs/en/content-assistant?topic=applications-handling-genai-query-objects for more details.

_7. **Click** on **Open**, which opens the new object with the “Properties” tab. **Find** the updated field **Gen AI LLM Response** with the answer to the query. Click on the icon next to the field and

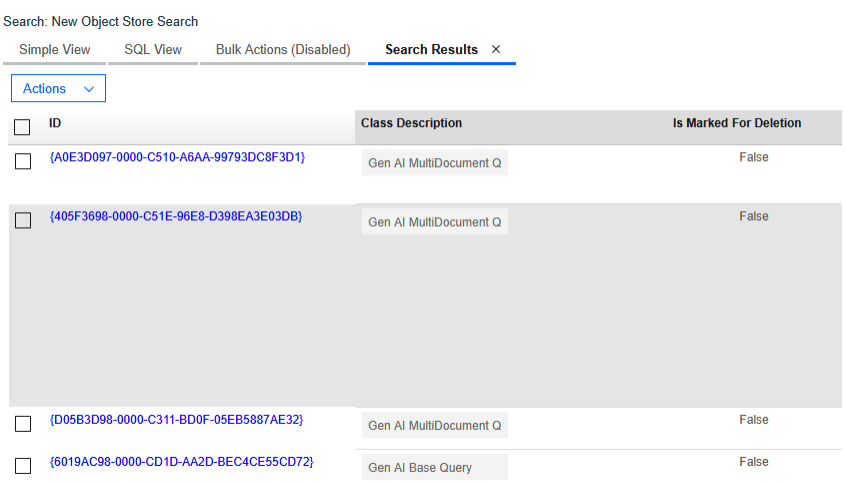
> Do you recall why the response is using HTML tags? Find the answer in section 3.2.1 Content Assistant Add-on and Configuration in step _14.

_8. **Click** on **Search** in the navigation. The “Saved Searches” window opens on the right side.

_9. **Click** on **New Object Store Search**.

_10. In the “New Object Store Search” window, **select** the **Gen AI Base Query** class, and **click** on **Run** to run the search. **Dismiss** the **warning dialog**.

A list of prior query objects to which the logged-on user has access appears in the Search Results window as shown in the example below:



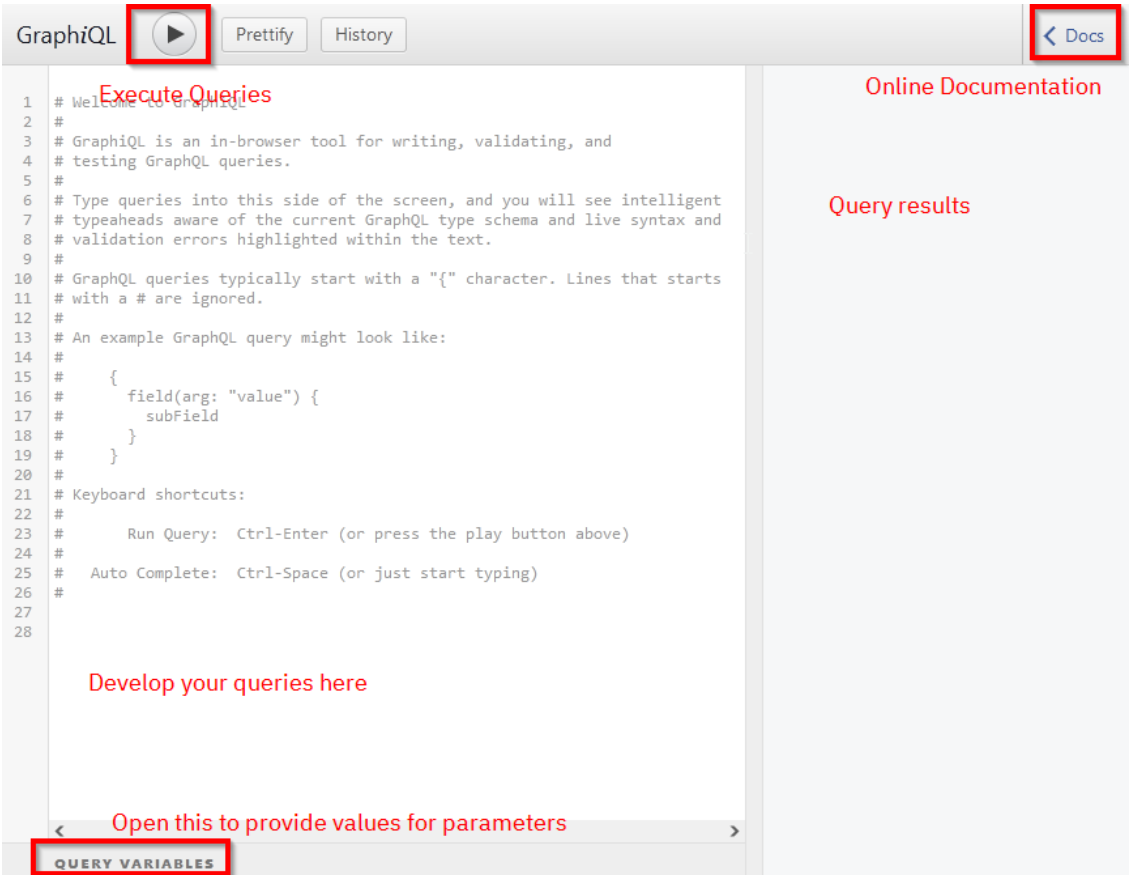
> By default, FileNet retains a history of the searches, which were performed by the logged-on users. This history can be seen in the IBM Content Assistant chat window in IBM Content Navigator, too. There are a couple possibilities to fine-tune this behavior according to the needs of customers:
> - The “Default Instance Permission” of the “Gen AI Base Query” class can be modified to only include permissions for the “#CREATOR-OWNER”. Then the history of queries would only be visible to the user who raised them.
> - A Disposal Policy can be defined for “Gen AI Base Query” objects, to automatically delete such objects after a defined time span. See “Disposal Policies” in the FileNet Content Manager documentation for details https://www.ibm.com/docs/en/filenet-p8-platform/5.7.0?topic=development-disposal-policies.
> - The history could be summarized through a Background search to derive information for invoicing users based on the Content Assistant queries they have raised.

4.2.2 Using Base Query Class from GraphQL

Instead of using ACCE, the queries from this and further subsections can also be executed from the Content Services GraphQL UI. This approach allows for easier embedding of ICA capabilities into custom applications.

_1. **Login** to the **Content Services GraphQL UI** with the URL and user account which was provided to you.

For a compact introduction, write the queries or mutations into the pane in the upper left corner. He existing sample can be removed before. The field in the lower left corner contains values for variables appearing in the query. By clicking the round play button (Execute Query Button), the query can be executed. Query results are displayed on the right side.



For running a query, new objects for running LLM queries are created in FileNet Content Engine using one of the IBM Content Assistant Query classes.

The first example will run the same query as the one from the last subsection.

_2. **Remove** any **query** shown when opening the GraphQL window.

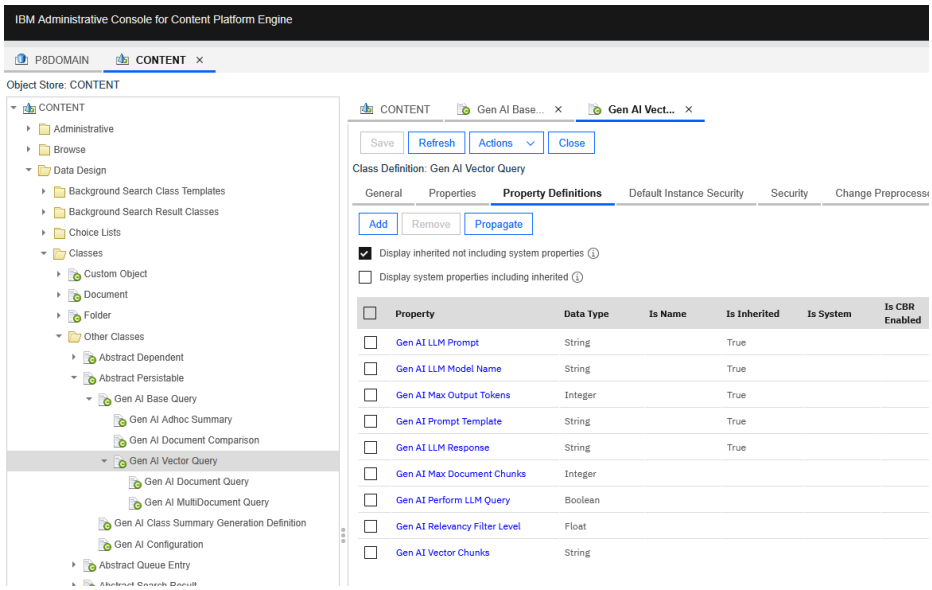
_3. **Copy & Paste** below **query** into the field in the upper left corner and **click** on the **Execute Query Button**.

```
mutation basequery {
  whatisecm: createCmAbstractPersistable(
    repositoryIdentifier: "CONTENT",
    classIdentifier: "GenAIBaseQuery",
    cmAbstractPersistableProperties: {
      properties:
       [ { GenaiLLMPrompt: "What is ECM?" }]
    }
  )
  {
    properties(includes: ["GenaiLLMPrompt", "GenaiLLMResponse" ])
      { alias value }
  }
}
```

See Content Lab 2 “Interfacing FileNet Content Platform Engine with GraphQL on Cloud Pak for Business Automation” for an introduction into FileNet Content Services GraphQL.

In this mutation, the request “createCmAbstractPersistable” can be used to create any subclass of the base class “Abstract Persistable” in FileNet Content Engine. The mutation request is provided with the Object Store, the class name, and the properties of the new object. In this case this is only the GenaiLLMPrompt property. The lower part describes which components from the newly created object shall be contained in the result. In this case its only two properties, and from them only name and value. Finally, the label “whatisecm” is configured to be used as the name of the response in the returned JSON information. This way, several such mutations can be sent to GraphQL within one request, if needed, for improved efficiency over wide area network connections.

_4. Queries against the complete vector database can be made in much the same way. Look up the class “Gen AI Vector Query” in ACCE on the Property Definitions tab to get an overview of its properties. Enable “Display inherited not including system properties” to have the display include the inherited properties from the base class.



To use the class, the property “Gen AI LLM Prompt” needs to be provided with the prompt for the LLM to answer, as before. Additionally, the property “Gen AI Perform LLM Query” needs to be set to true. The query can be configured further by passing other configuration properties, see the description of this query class in the IBM Content Assistant documentation https://www.ibm.com/docs/en/content-assistant?topic=classes-genaivectorquery for details.

However, on the execution of such queries, only documents available to the logged-on user will be used.

_5. Use below sample to perform such a query. Compare the query with the previous query using the GenaiBaseQuery class. There are only few differences, which are typeset here in bold.

```
mutation vectorquery {
  bautomationelite: createCmAbstractPersistable(
    repositoryIdentifier: "CONTENT",
    classIdentifier: "GenaiVectorQuery",
    cmAbstractPersistableProperties: {
      properties:
       [ { GenaiLLMPrompt: "Summarize the information available for Automation Elite
Inc. Since when are we making business with them?" },
         { GenaiPerformLLMQuery: true }]
    }
  )
  {
    properties(includes: ["GenaiLLMPrompt", "GenaiLLMResponse" ])
      { alias value }
  }
}
```

4.2.3 Queries with One or More Documents

For most other kind of queries, one or more documents need to be passed as references, to be stored in properties of type OBJECT in the IBM Content Assistant query classes. As a blueprint for this kind of queries, the first example creates an LLM query with one document passed to it.

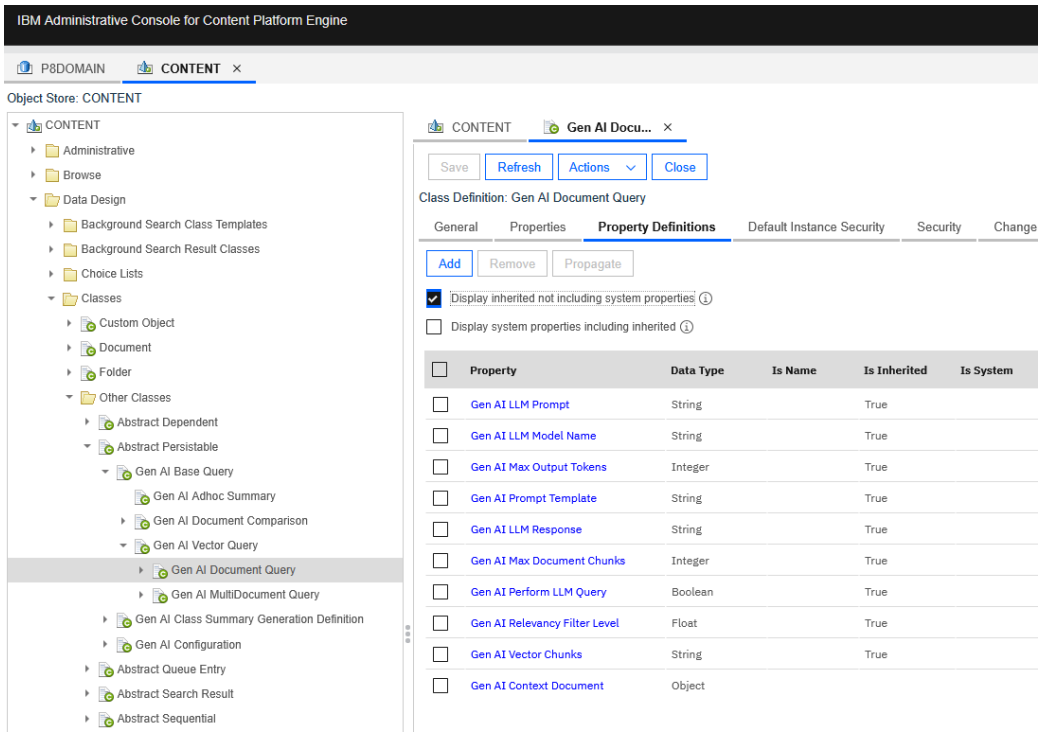
_1. For running this kind of query, information to the referenced document needs to be provided, i.e. the Object ID of the document, and the Object ID or the name of the Document class. The following GraphQL query lists the documents, which are stored in the Folder “SOLUTION Client Onboarding”. **Copy & Paste the query to GraphQL**.

```
query solutionDocuments {
  solutiondocuments: folder(
    repositoryIdentifier: "CONTENT",
    identifier: "/SOLUTION Client Onboarding") {
    name
    containedDocuments {
      documents {
        id name className
      }
    }
  }
}
```

_2. **Copy the result into an editor window.** On a Windows System you might invoke Notepad and copy the result into the Notepad window on a new tab. Use similar tools on computers with different operation systems.

Another solution would be to leave the browser tab with the output from GraphQL open in the browser for future reference and use a new tab for the following queries.

_3. Before using the query, review the class definition of the GenaiDocumentQuery again in ACCE. Go to the Property definitions tab and include display of inherited properties, to get an overview over which parameters can / must be provided. As the class inherits from the GenaiVectorQuery document class, find its properties among the inherited ones. The only new one which needs to be provided is the GenaiContextDocument property.



_4. **Copy & Paste** the below the **parameterized GraphQL query** into the **query pane** of the GraphQL browser window for running a single document query. For providing the value of the GenaiLLMPrompt property, a String parameter is used, for the GenAIContextDocument property, as an Object is expected (see above), the parameter type is ObjectReferenceInput in Graph QL.

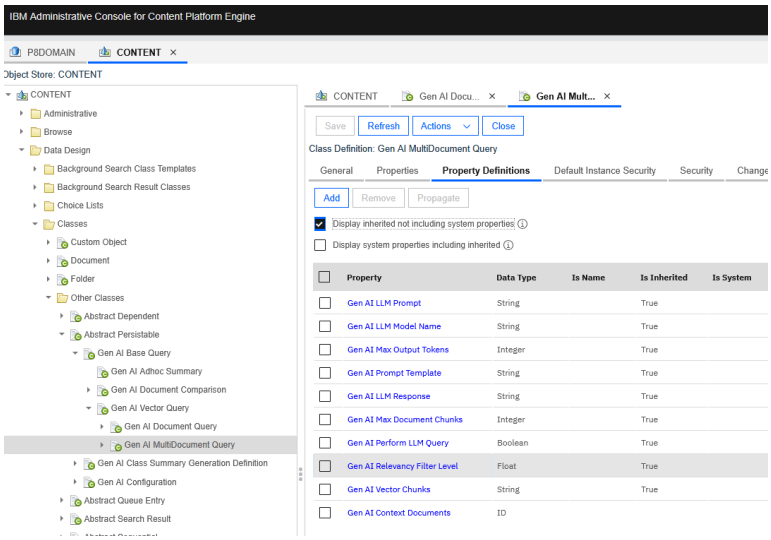
Similar as for the Vector Query, for the Document query the property “GenaiPerformLLMQuery” needs to be set to true. By passing “false” here, developers of custom applications can test their development of an IBM Content Assistant integration in custom applications, without actually executing the queries.

```
mutation documentquery($query: String!, $doc: ObjectReferenceInput!) {
  singleDocQuery: createCmAbstractPersistable(
    repositoryIdentifier: "CONTENT",
    classIdentifier: "GenaiDocumentQuery",
    cmAbstractPersistableProperties: {
      properties:
        [ { GenaiLLMPrompt: $query },
          { GenaiPerformLLMQuery: true },
          { GenaiContextDocument: $doc }
        ]
      }
    )
    {
      properties(includes: ["GenaiLLMPrompt", "GenaiLLMResponse"])
       { alias value }
    }
  }
```

_5. For running the query, in the lower left corner of the GraphQl UI, the parameter values need to be passed using JSON syntax. In the example below **replace DOCID** first with the **ID of the Banking Information Document for Automation Elite Inc.** then with the **ID of the Banking Information Document for Legacy Consulting**.

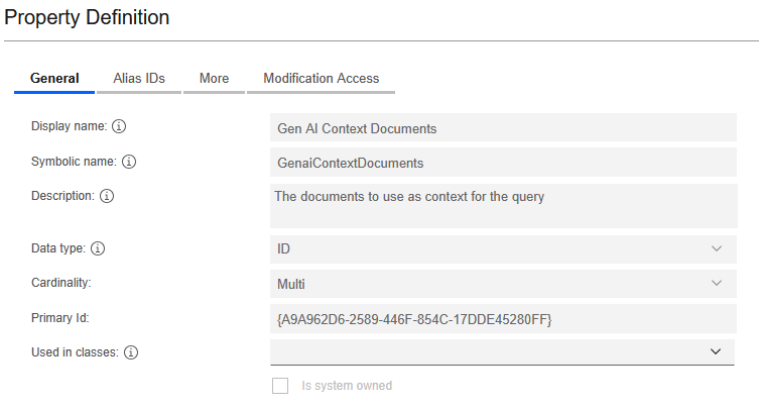
```
{
  "query": "Is the document a Banking Information statement for the company
Automation Elite Inc? Answer with True or False only.",
  "doc": {
    "repositoryIdentifier": "CONTENT",
    "classIdentifier": "SOLUTION_Bank_Information",
    "identifier": "DOCID"
  }
}
```

_6. As the next example in this section, a GenaiMultiDocumentQuery will be executed. Look up the class definition in ACCE first, as before. It looks very similar to the GenaiDocumentQuery class.



_7. **Click** on the **Gen AI Context Documents** property. A window with details about the “Property Definition” opens. **Verify** that the **Cardinality** of the property is given as **Multi** which means that an Array of Document IDs need to be passed for the property value. **Close** the Property Definition **window**.

> Note that a description of the property is also available there. Note further that the symbolic name of that property is “GenaiContextDocuments”. Within applications, always the symbolic names are used. The display names are not only subject to change, but they can also appear translated, when switching the browser to a different language.



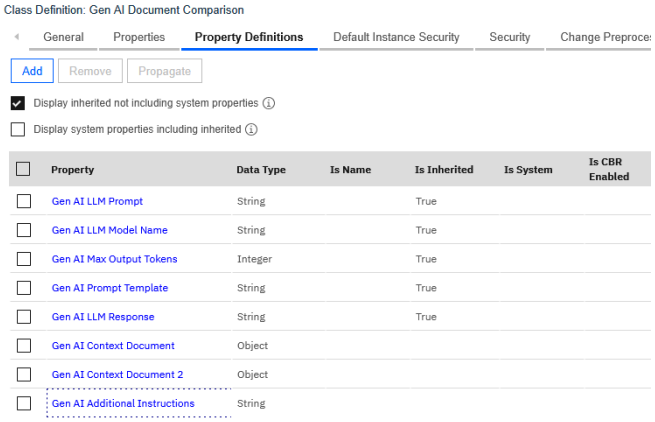
_8. Below GraphQL mutation will be creating a GenaiMultiDocumentQuery object. It is again parameterized, as the GenaiDocumentQuery. The type of the second parameter was adapted to provide an array of Document IDs. **Copy & Paste** the **query** into the GraphQL query window.

```
mutation multidocumentquery($query: String!, $docs: [ ID ]) {
  multiDocQuery: createCmAbstractPersistable(
    repositoryIdentifier: "CONTENT",
    classIdentifier: "GenaiMultiDocumentQuery",
    cmAbstractPersistableProperties: {
      properties:
            [ { GenaiLLMPrompt: $query },
        { GenaiPerformLLMQuery: true },
        { GenaiContextDocuments: $docs }
       ]
      }
    )
    {
      properties(includes: ["GenaiLLMPrompt", "GenaiLLMResponse"])
      { alias value }
    }
  }
```

_9. The parameter values are again provided using JSON. **Replace** the **two Object IDs** from the Banking Information Documents of the two companies.

```
{
  "query": "What companies do we have banking information for? Provide bank
names and the date of the banking information.",
  "docs": [
    "DOCID_AUTOMATION_ELITE",
    "DOCID_LEGACY_CONSULTING"]
}
```

_10. For the comparison of two documents, the Gen AI add-on defines the “Gen AI Document Comparison” subclass of the Gen AI Base query. It has following properties



_11. Below GraphQL mutation will be creating a GenAIDocumentComparison object. It is again parameterized, as the GenaiDocumentQuery. The type of the parameter is the same as for the GenaiDocumentQuery, just two of them are needed for the two documents. And there is an additional field for the additional instructions. **Copy & Paste** the **query** into the GraphQL query window.

```
mutation multidocumentquery($instructions: String!,
                            $doc1: ObjectReferenceInput!,
                            $doc2: ObjectReferenceInput!) {
  compare: createCmAbstractPersistable(
    repositoryIdentifier: "CONTENT",
    classIdentifier: "GenaiDocumentComparison",
    cmAbstractPersistableProperties: {
      properties:
         [ { GenaiLLMPrompt: "Compare the documents" },
           { GenaiContextDocument: $doc1 },
           { GenaiContextDocument2: $doc2 },
           { GenaiAdditionalInstructions: $instructions }
        ]
      }
    )
    {
      properties(includes: ["GenaiLLMPrompt", "GenaiLLMResponse",
                            "GenaiAdditionalInstructions"])
      { alias value }
    }
  }
```

_12. The parameter values require substitutions of the Document IDs again. Provide the two banking information documents.

```
{
  "instructions": "List only the similarities",
  "doc1": {
    "repositoryIdentifier": "CONTENT",
    "classIdentifier": "SOLUTION_Bank_Information",
    "identifier": "DOCID_AUTOMATION_ELITE"
  },
  "doc2": {
    "repositoryIdentifier": "CONTENT",
    "classIdentifier": "SOLUTION_Bank_Information",
    "identifier": "DOCID_LEGACY_CONSULTING"
  }
}
```

_13. Finally, try to create a parameterized GraphQL query for invoking the GenaiAdhocSummary query class, by looking up the class definition in ACCE, and adapting the example for the GenaiMultiDocumentQuery. For that class only the documents need to be supplied, in the same way as for the GenaiMultiDocumentQuery, the values for the GenaiPerformLLMQuery and the GenaiLLMPrompt can be removed.

Find the solution at the end of this document, following the Troubleshooting section.

Congratulations you have successfully completed the lab
“Configuration and Usage of IBM Content Assistant on Cloud Pak for Business Automation”!

5 Troubleshooting

5.1 GENAI_QUERY_FAILED

When the connection between the FileNet environment and the IBM Content Assistant Server in the SaaS environment is unavailable or overloaded, the following error might be observed in the GraphQL GUI:

```
"errorCode": "FNRJG1005",
"message": "Error when calling the Content Platform Engine API.",
"explanation": "An error occurred when calling the Content Platform Engine API. An
exception occurred in the handler. Message was: GENAI_QUERY_FAILED: Vector query failed.
The response is: {\"message\": \"Endpoint request timed out\"}",
"userResponse": "Contact your system administrator for assistance.",
"statusCode": "500",
"serverErrorMessage": "An exception occurred in the handler. Message was:
GENAI_QUERY_FAILED: Vector query failed. The response is: {\"message\": \"Endpoint request
timed out\"}",
"classification": "DataFetchingException"
```

Retrying the request has proved to be a good strategy to handle this error situation.

5.2 Body has already been consumed.

```
Response.text: Body has already been consumed.
```

This error can happen when the authentication token of the CP4BA environment has expired. To resolve the situation, place the mouse cursor on the address bar of the browser, and hit Return, to reload the page. Pressing F5 key on a Windows machine will have the same result.

If the login page of Cloud Pak for Business Automation appears as a result, then the problem was the Authentication token. Use the provided username and password to login again. Your query and parameter values should still be there after logging in again.

5.3 Subscription contract limits

```
Subscription contract limit for the number of users has exceeded, additional packages can
be purchased.
```

If you get this message, the problem is a limitation configured on the IBM Content Assistant server. Many kinds of limitations are imposed there. Configured usage limits can be increased by the IBM Content Assistant administrator.

If you get this error, reach out to the instructors of the event for assistance.

6 Sample GraphQL for the GenaiAdhocSummary query class

Below GraphQL query implements the requirement. Compared to the GenaiMultiDocumentQuery, the parameter for the query was removed, as well as the two property definitions, which are not needed.

```
mutation multidocumentquery($docs: [ ID ]) {
  adhocsummary: createCmAbstractPersistable(
    repositoryIdentifier: "CONTENT",
    classIdentifier: "GenaiAdhocSummary",
    cmAbstractPersistableProperties: {
      properties:
            [{ GenaiContextDocuments: $docs }]
      }
    )
    {
          properties(includes: ["GenaiLLMPrompt", "GenaiLLMResponse"])
     { alias value }
    }
  }
```

For the parameters, only remove the prompt, for example:

```
{
  "docs": [
    "DOCID_AUTOMATION_ELITE",
    "DOCID_LEGACY_CONSULTING"]
}
```