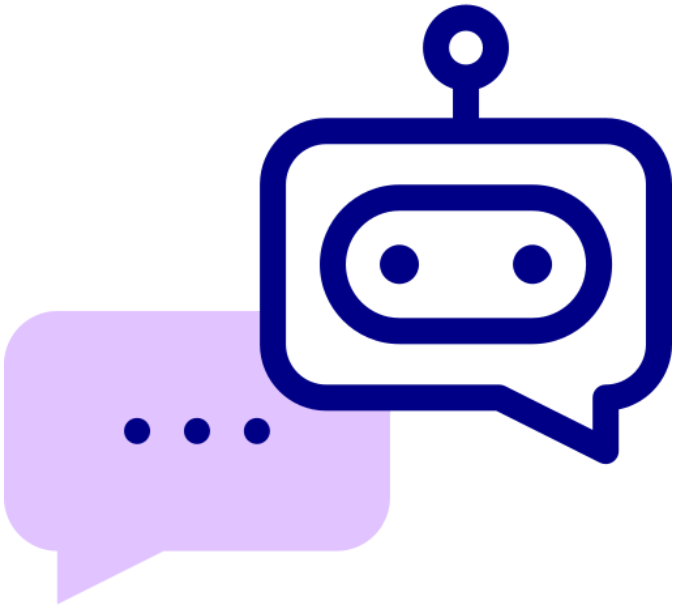


AI Knowledge Assistant Component for Appian

V3.1.0

Appian Corporation

Version 3.1.0

Table of Contents

Overview

The AI Knowledge Assistant is an innovative UI component tailored to revolutionize user interactions within the Appian platform. By harnessing the power of large language models, this tool offers an immersive chatbot experience, allowing users not just to converse with state-of-the-art Generative AI models, but also source answers from their Appian Knowledge Base documents. This means, beyond traditional data retrieval, users can now semantically search and engage with their private data. Designed with adaptability in mind, the AI Knowledge Assistant seamlessly integrates with various organizational themes, ensuring a cohesive brand experience.

Features

1. **Intuitive Chat Experience:** With the integration of large language models, users can have natural, flowing conversations with the system, making data retrieval feel less like querying and more like conversing.
2. **Semantic Document Search:** Go beyond keyword searches. With the AI Knowledge Assistant, users can chat with documents, extracting nuanced insights from the Appian Knowledge Base. This semantic search capability ensures that the information retrieved is contextually relevant and precise.
3. **Privacy-Centric Interactions:** Recognizing the importance of data privacy, the AI Knowledge Assistant is designed to allow users to interact with their private data in a secure environment. This ensures that sensitive information remains protected while still being accessible.
4. **Customizable Themes:** Every organization is unique, and the AI Knowledge Assistant celebrates this individuality. It comes with a range of customizable themes that can be tailored to resonate with your company's brand aesthetics.
5. **Vector Database Integration:** At the core of its functionality, the AI Knowledge Assistant utilizes a vector database. This advanced technology allows for efficient and accurate document interactions, transforming the way users engage with written content.

6. **Document Security:** The AI Knowledge Assistant manages the security of Appian Documents to prevent unauthorized access to documents that users do not have the necessary permissions to view or edit.
7. **Response Streaming:** The AI Knowledge Assistant delivers responses seamlessly by streaming information, providing users with a chatbot-like experience. This intuitive interface incorporates a convenient stop button, allowing users to halt the response stream at any point during the interaction. This feature enhances user control, enabling them to manage the flow of information according to their preferences and needs.
8. **Embedded PDF Viewer:** Introducing the Embedded PDF Viewer feature within the AI Knowledge Assistant! This empowers users to seamlessly view referenced sources directly within the component. With the Embedded PDF Viewer, users can effortlessly navigate through PDF documents, conveniently scrolling to the exact page referenced for enhanced comprehension and efficiency.
9. **Markdown Response:** The AI Knowledge Assistant introduces styled responses rendered in markdown format. Now, users can enjoy content presented with clear structure and visual appeal, featuring titles, bulleted lists, code blocks, and more. This update ensures that information is not only comprehensive but also visually engaging and facilitates easier understanding.

Prerequisites

- Download and configure the Document Vector Database Connected System found on the Appian App Market.
- Third Party Credentials set up for using Document Querying functionality. Refer here for the details on how to set up Third Party Credentials in the Admin Console.

Production Usage

- If the Document Vector Database and AI Knowledge Assistant plugins are being deployed in a production environment, open a support case to increase Heap Max for app server by 1GB.
- If the AI Knowledge Assistant is displaying warning messages of low memory, also consider opening a support case to increase Heap Max for app server by 1GB.

Migration Guide

If you are already using an older version of the AI Knowledge Assistant Component, complete the following steps to migrate to the newer version. This will be a one-time setup.

- Install the latest version of the Document Vector Database Connected System.
- Once the latest version of the connected system is installed, create a new Vector Database Connected System object. The previous connected system object and integrations based on it will continue to work.
    - **Important Note:** While creating the connected system object, do not use the same Database Name, provide a different value.
- Use the List Documents Integration (based on the older version of the connected system) to get the list of all documents uploaded to the previous vector database.
- Create an integration based on the new connected system object using the Upload Document Integration template and upload all the documents that were in the previous database.
- Create new integration objects for the new connected system. Older integration objects will continue to function off the previous connected system version.
- After the completion of the migration, set up the Third Party Credentials in the Admin Console as shown here and configure the "securityKey" parameter.
- There are also other parameter configuration changes to the component, please refer to the following Component Parameter Configuration for more details.

IMPORTANT NOTE: When migrating/deploying the application from one environment to another, all documents that have been previously uploaded to the vector database in the older environment will not be present in the new vector database. Since documents do not retain their document ID across environments, you will need to upload each document again to the new vector database.

Component Parameter Configuration

Rule Name : AIKnowledgeAssistantField()

Name	Type	Description	Required
systemMessage	Text	(Optional - Defaults to "You are a helpful assistant.") The system message helps set the behavior of the assistant.	No
initialMessage	Text	(Optional) Initial message shown to the user from the chatbot. This will be the beginning of the chat conversation.	No
style	Dictionary	(Optional) Customized styling for the component's title, icons, chatHeight, collapsible state, and theme. Copy and paste the following parameters style: { titleText: "AI Knowledge Assistant", theme: "DEFAULT"/* valid values for themes are: ACCENT, DARK, BLUE,GREEN, LIGHT,DEFAULT. Default theme: "DEFAULT" */,isCollapsible: "Optional-Boolean"/*Provide false to disable the collapsible functionality of the component. Default: true*/,collapseChatOnLoad: false/* Provide true to collapse the chat on component load*/, chatHeight: "700px"/* set the height of the chat component. Use this property to control height rather than the OOTB height field.(Defaults to 700px, minimum is 400px and maximum is 2000px.)*/, userIcon: ""/* use document(documentId: cons!G_IMAGE, property: "url") to set a custom icon */, GPTIcon: "",showShadow: "Optional-Boolean" /* Provide true to show shadow for the component. Default: false*/, showBorder:"Optional-Boolean" /*Provide true to show border for the component. Default: true*/,shape:"Optional-Text"	No

		/*Controls the shape of the component. Valid values: "SQUARED", "SEMI_ROUNDED","ROUNDED". Default:"SQUARED"*/ }	
conversationValue	List of Dictionary	The history of all messages in the conversation, starting with systemMessage and initialMessage. Each dictionary in the list conforms to the following pattern: { role: system/assistant/user, content: "The message content",metadata: (List of Dictionary) Each dictionary conforms to the following structure: { documentId: integer, documentName: text, pageNumber: integer, text: text,similarityScore: decimal}}. The first item starts with the system role and is followed by alternative assistant and user roles. As a rule input, this parameter is of type List of Map.	No
conversationSaveInto	List of Save	One or more variables that are updated with the conversation value when the user changes it.	No
vectorDatabaseConnectedSystem	Connected System	(Required) Provide the Document Vector Database Connected System constant reference.	Yes
securityKey	Text	To ensure the Assistant accesses only authorized documents and get streaming responses, set up a key in the 'Third-Party Credentials' section of the Admin Console. Choose an 'Identity name', enter 'password' as the field name, input the Vector Database password as the field value, and add Document Vector Database to the 'Plug-ins List'. Use the 'Identity key', not the 'Identity name', for the securityKey parameter in this component. This is required when using the queryEmbeddedDocuments parameter. Refer here for the steps to create Third Party Credentials.	Yes

queryEmbeddedDocuments	Dictionary	If not configured, the component will work as a normal ChatGPT chatbot without access to Knowledge Center documents). Copy and paste the following setup: queryEmbeddedDocuments: { documentList: /* Determines the list of documents the Assistant will use to answer questions. Documents not previously uploaded through the upload integration will be uploaded when the component loads*/ { { displayName: "Optional - Text", documentId: "Required - Integer" } },topK: "Optional - Integer"/*topK allows you to customize the number of text chunks to pull from the document to help answer the user's query.*/, chatWithoutDocuments: "Optional-Boolean"/*Provide true to chat without documents in the component. Default: false. The documentList is required when this value is provided as false.*/,enableDocumentSelection: "Optional - Boolean"/* Allows the user to select which documents from the documentList to query (false - query all documents in the documentList; true - query user selected documents) */, inaccessibleDocumentsVisibility: "Optional - 'SHOW'/'HIDE'/'BLUR'"/* (Defaults to "SHOW") Allows the user to either SHOW/HIDE/BLUR the Inaccessible Documents in the Select Documents dropdown. This will be applicable when enableDocumentSelection is true. Valid values: "SHOW","HIDE","BLUR"*/, irrelevantContentMessage: "Optional - Text" /*This message will be used as a response when no relevant content is found for the user's query. Defaults to "Unable to find any relevant information for the given query. Please rephrase the query and try again."*/ }	No
onSourceDocumentSelect	List of Save	(Optional - Integer) - When OpenAI responds to a query using documents as sources and a user clicks on the source document name, the corresponding	No

		Document ID is saved into this value. Use a!save(ri!documentRuleInput, save!value) where ri!documentRuleInput can be used to display the source pdf embedded next to the chat component.	
enableEmbeddedPDFViewer	Boolean	(Optional - Boolean) - Provide true to enable the embedded PDF viewer in the component. Default: true.	No

Component Usage Guide

To use the Chat Completions functionality, check the following types of parameter configurations:

- Chatbot with no ability to use Appian Documents to answer user questions:
  - Provide 'true' for 'chatWithoutDocuments' property in the queryEmbeddedDocuments parameter. This is 'false' by default and documentList should not be empty.

    Example: AIKnowledgeAssistantField(

    vectorDatabaseConnectedSystem: cons!CS_CONSTANT,

    securityKey: "*scsKey*",

    queryEmbeddedDocuments: {

    chatWithoutDocuments:true

    }

)

- Chatbot that can source answers from Appian Documents:
  - Configure 'chatWithoutDocuments' as 'false' (Default) and documentList with the documents you want the chatbot to reference to answer a user's question. **Pre-Embedding Documents**: Prior to the user querying the component, use the "Upload Document" integration from the Document Vector Database Connected System to configure which documents the component can reference. Since the "Upload Document" step takes time to embed the document in the vector database, it is highly recommended that this step be performed prior to the user interacting with the chatbot.

Example: AIKnowledgeAssistantField(

vectorDatabaseConnectedSystem: cons!CS_CONSTANT,

securityKey: "*scsKey"*,

queryEmbeddedDocuments: {

documentList: {

{

documentId: document(cons!DOC_CONSTANT, "id"),

displayName: "Document Display Name"

}

},

chatWithoutDocuments: false()

}

)

- **Embedding Documents On-the-Fly:** While pre-embedding the documents with the Upload integration is preferred, the component can handle new documents that have not been previously uploaded to the vector database. Simply reference these new documents in the documentList and when the user prompts the chatbot, the component will embed these documents on-the-fly. This is the same as running the "Upload Document" integration, but the user will have to wait for the document to be embedded and added to the vector database, which, depending on the documents' size, can add extra time to the query.
    - If all the documents given in the documentList are not pre-embedded or are currently being processed (i.e. embedded), the user will not be able to select and query from the documents. Instruction text explaining this will be shown above the paperclip icon. Larger texts will take a longer time to process.
    - If some of the documents are pre-embedded and the remaining are not embedded, we can use the component to query from the pre-embedded documents and once the embedding of other documents is completed we can query from the given list of documents.

Example: AIKnowledgeAssistantField(

vectorDatabaseConnectedSystem: cons!CS_CONSTANT,

securityKey: "*scsKey"*,

queryEmbeddedDocuments: {

documentList: {

{

documentId: document(cons!DOC_CONSTANT, "id"),

displayName: "Document Display Name"

}

},

chatWithoutDocuments:false()

}

)

- Allow the user to select which documents to query.
  - Set 'enableDocumentSelection' to 'true' to allow the user to specifically set which documents they would like to source answers from. The user can use the file icon to select which documents they want the chatbot to reference. If the user does not select any documents, the message input field and send button will be disabled until the user selects any document to query from.
  - The document icon for selecting the documents will be disabled if the documentList is empty.

    Example: AIKnowledgeAssistantField(

    vectorDatabaseConnectedSystem: cons!CS_CONSTANT,

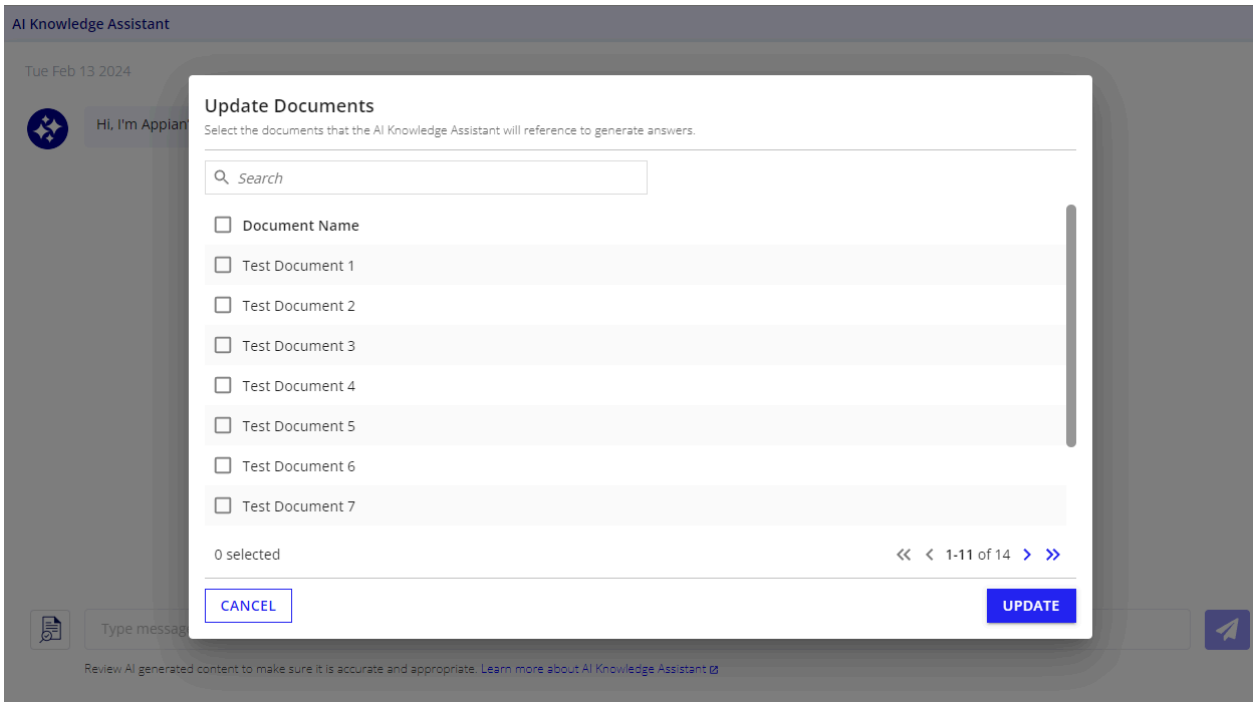
    securityKey: "*scsKey"*,

    queryEmbeddedDocuments: {

    documentList: {

    {

```
                documentId:document(cons!DOC_CONSTANT, "id"),
                displayName: "Document Display Name"
                }
        },
        enableDocumentSelection: true(),
        chatWithoutDocuments: false()
        }
)
```



- Query all documents in documentList by default.
    - Set enableDocumentSelection to false. No file icon will be shown to the user (chatWithoutDocuments should be false or not set which defaults to false).
    - If both enableDocumentSelection and chatWithoutDocuments are false and the documentList is empty or null, validation will be shown in the component.
- Provide topK input denoting the number of top relevant results to be used to generate the response.

Example: AIKnowledgeAssistantField(

vectorDatabaseConnectedSystem: cons!CS_CONSTANT,

securityKey: "*scsKey"*,

queryEmbeddedDocuments: {

documentList: {

{

documentId: document(cons!DOC_CONSTANT, "id"),

displayName: "Document Display Name"

}

},

chatWithoutDocuments:false(),

topK: *your_preferred_topK_value,*

}

)

Refer the following table for the topK value configuration depending on the model you are using:

Model	Max Tokens for the model	Default Value	Maximum Allowed Value
gpt-3.5-turbo gpt-3.5-turbo-0613 gpt-3.5-turbo-0301	4096	8	8
gpt-3.5-turbo-16k gpt-3.5-turbo-16k-0613 gpt-3.5-turbo-1106	16385	24	41
gpt-4 gpt-4-0613 gpt-4-0314	8192	16	20

gpt-4-32k gpt-4-32k-0613 gpt-4-32k-0314	32768	32	95
gpt-4-1106-preview gpt-4-vision-preview	128000	40	410

- Configure UI with different themes, icons, chatHeight, and, and titleText using the style parameter.

  Example:

```
AIKnowledgeAssistantField(
  vectorDatabaseConnectedSystem: cons!DCS_CSP,
        securityKey: "scsKey",
        queryEmbeddedDocuments: {
          documentList: {
                {
                  documentId: document(cons!Document,"id"),
                  displayName: "Document Display Name"
                }
        },
        chatWithoutDocuments: false(),
        topK: 8
  },
height: "AUTO",
style: {
 titleText: "Appian Assistant",
```

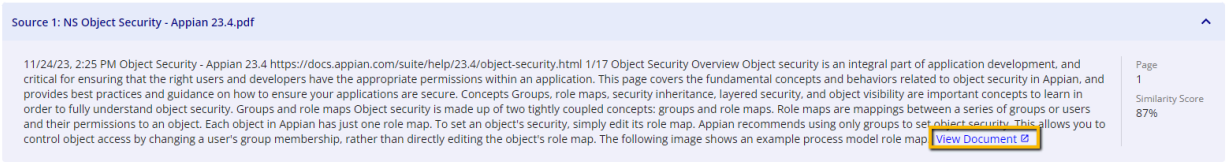
```
chatHeight: “700px”,

collapseChatOnLoad: false,

theme: "DEFAULT", /*valid values for themes are: DEFAULT,ACCENT, DARK, BLUE,GREEN,   LIGHT*/,

showBorder: true,

showShadow: false,

shape: “SQUARED”,

GPTIcon: document(documentId: cons!G_IMAGE, property: "url",

userIcon: document(documentId: cons!G_IMAGE, property: "url"

 }

)
```

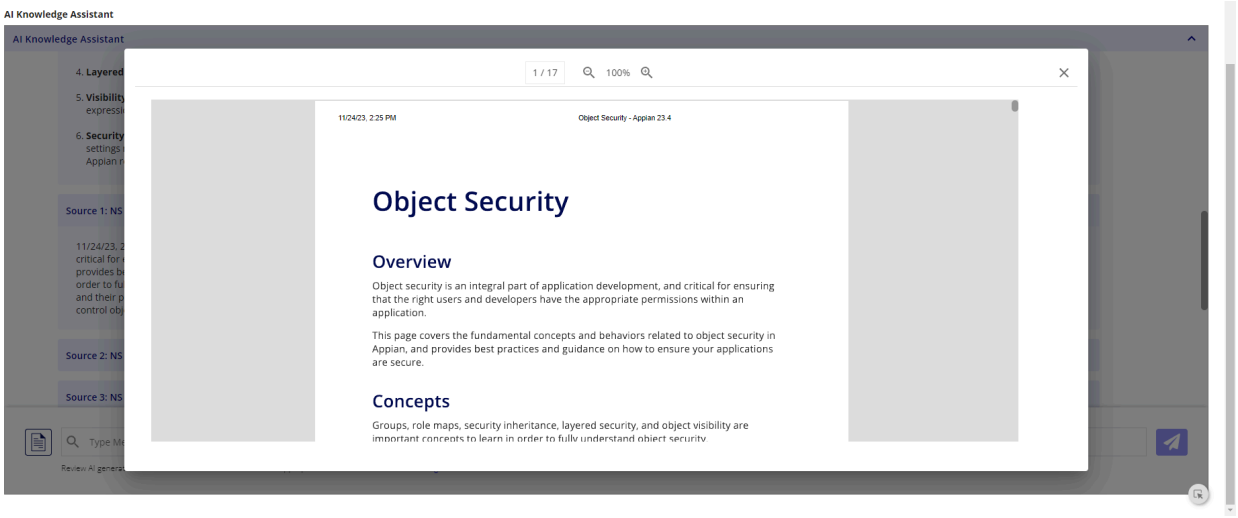
- If the “enableEmbeddedPDFViewer” is set to true (Defaults to true), the users can view the source document within the component and the PDF will open at the same page where that particular source is located.

Example: AIKnowledgeAssistantField(

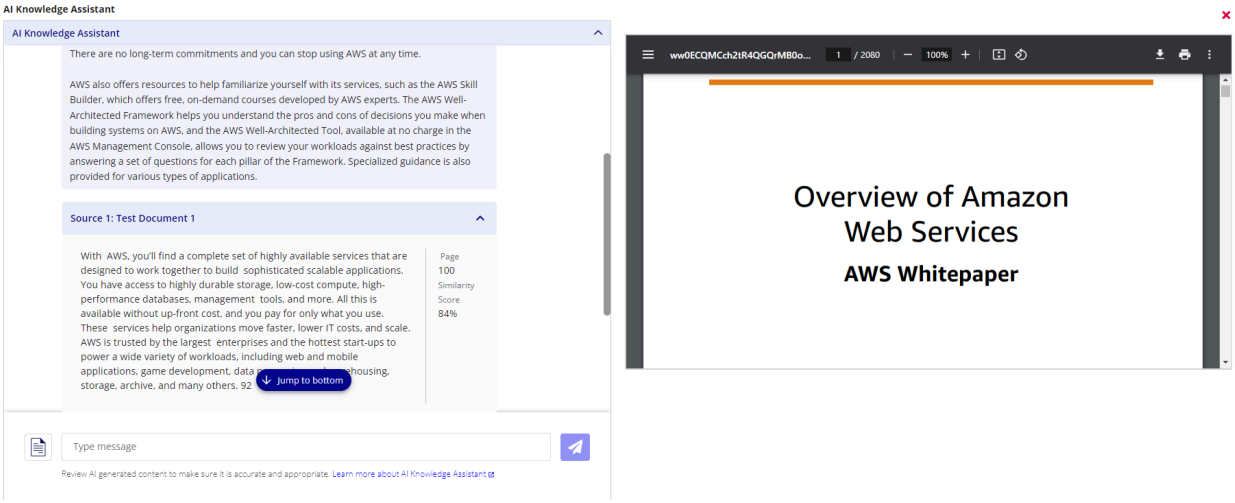
```
vectorDatabaseConnectedSystem: cons!CS_CONSTANT,

securityKey: “scsKey”,

enableEmbeddedPDFViewer: true,

queryEmbeddedDocuments: {

        documentList: {

                {

                documentId: document(cons!DOC_CONSTANT, "id"),

                displayName: "Document Display Name"

                }

         },

        chatWithoutDocuments:false(),

        }
```

)

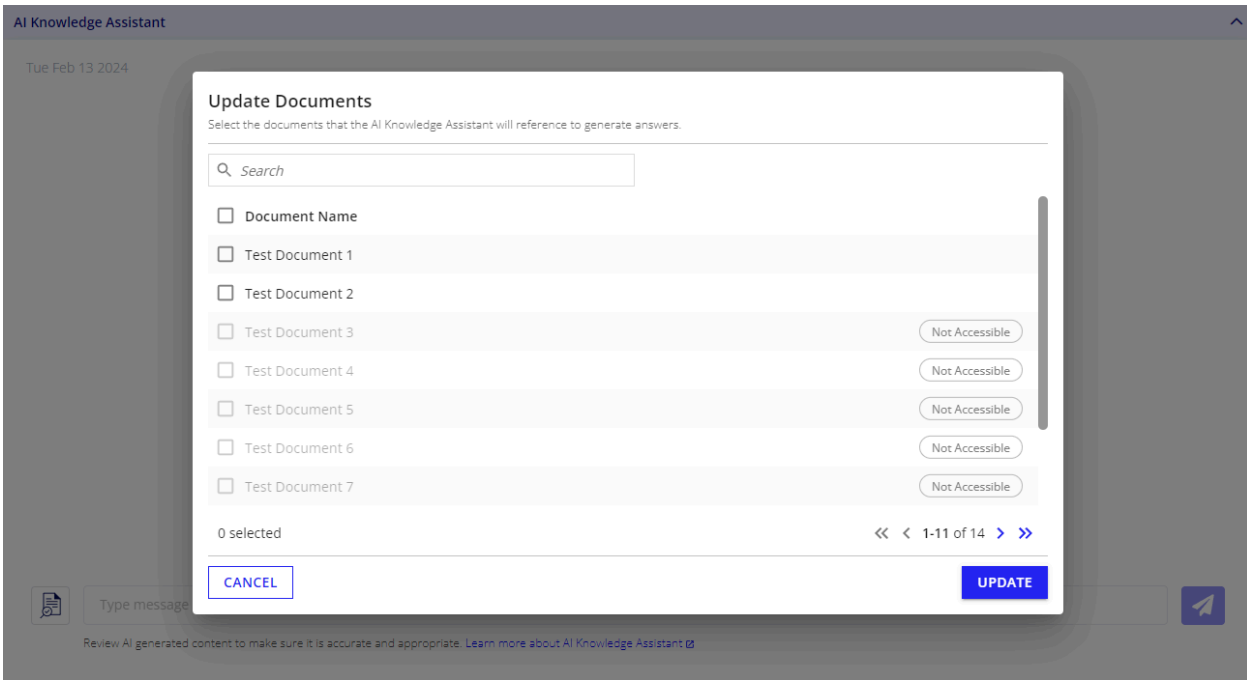




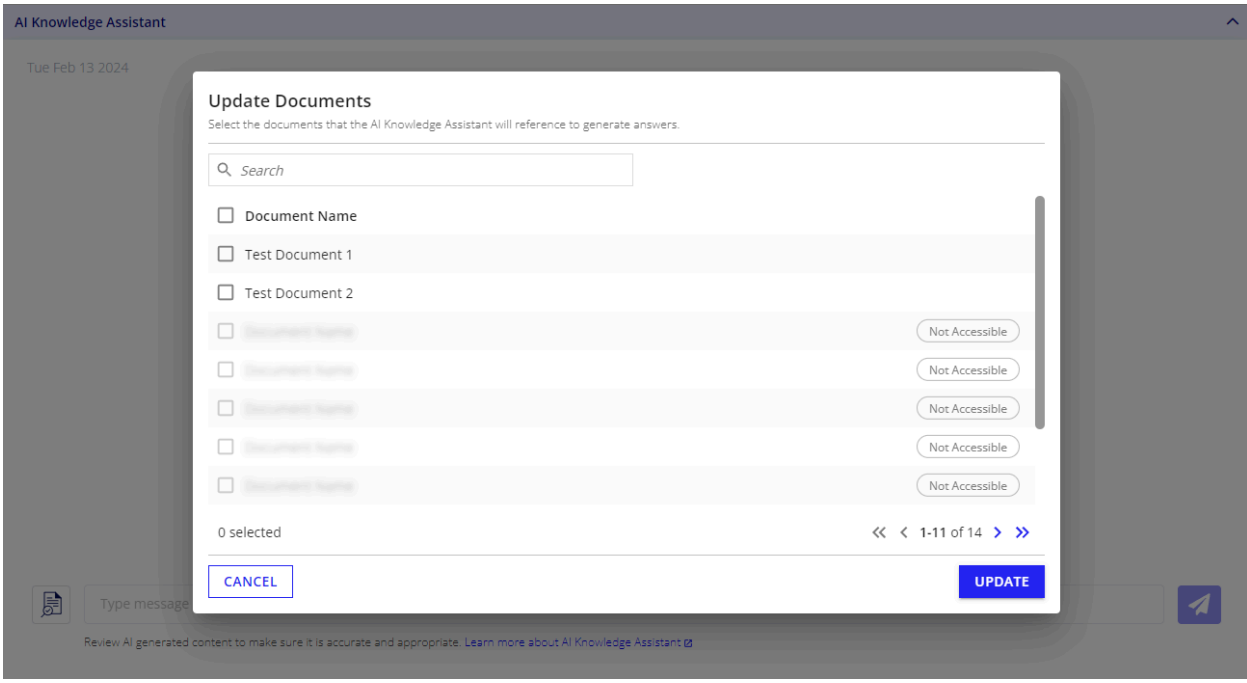
- Configure "onSourceDocumentSelect" to allow a user to click on the document name of a source and view the document(outside component) that was used to answer their question.



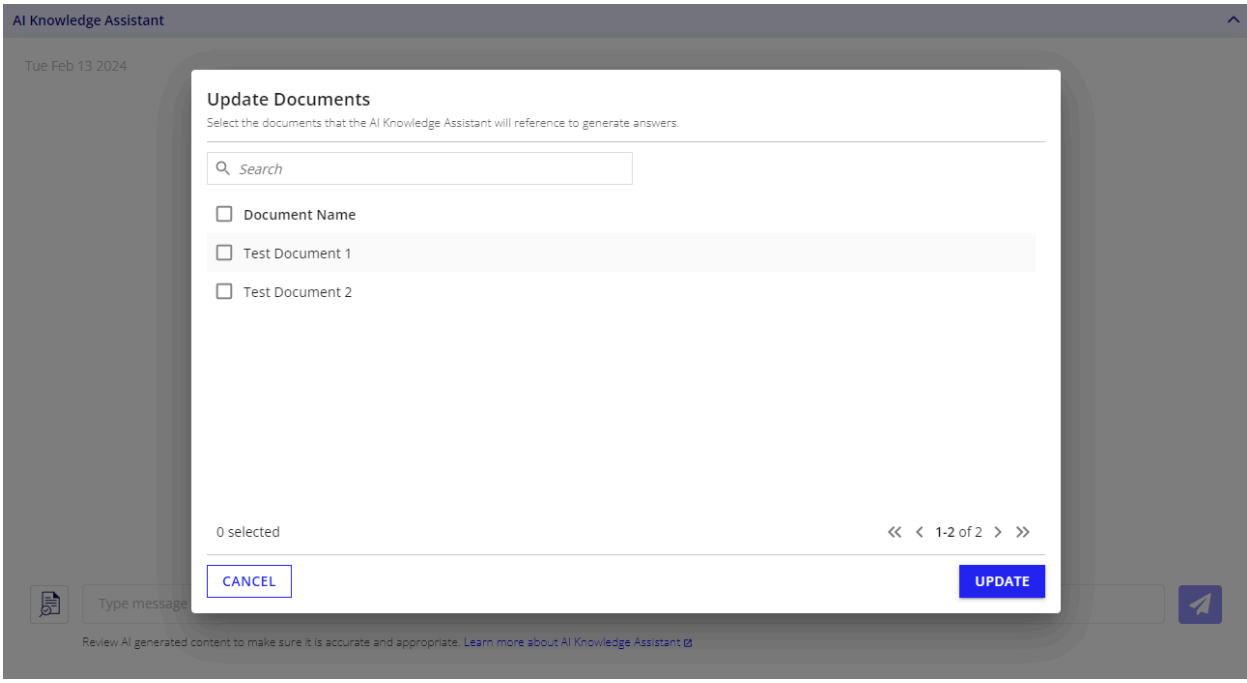
- **Inaccessible Documents Visibility:**
  - Use the **inaccessibleDocumentsVisibility** property in the queryEmbeddedDocuments parameter to show/hide/blur the documents without access for the logged in user.
    - Show - Will display the documents in the dropdown as disabled.



- Blur - Will display the documents in the dropdown with their names being blurred.



- Hide - Will not display in the dropdown.



Note:

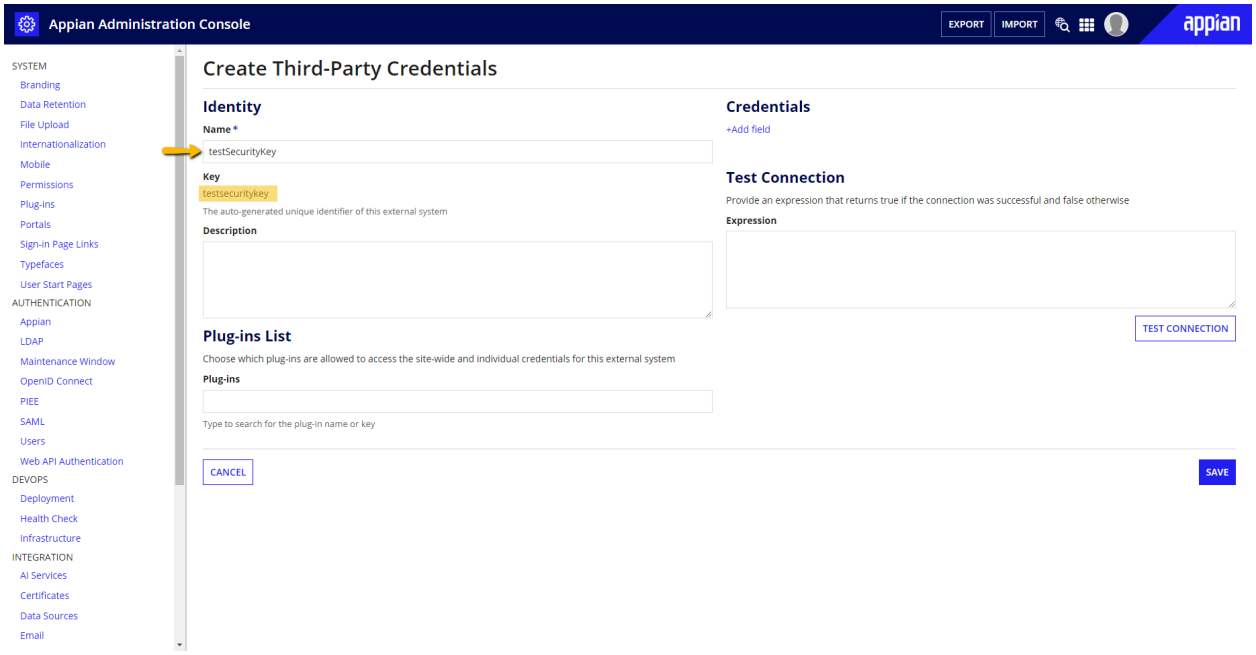
- This component will not work in an Incognito as it relies on browser cookies to decode which Appian user is interacting with the component.
- The component will show a retry button in the following scenarios:
    - If the querying takes more than a minute.
    - If there is not sufficient heap memory available for querying the given set of documents.
    - If there are any rate limit errors from OpenAI/Azure.
- It is recommended not to query any single document or documents that have more than 5000 pages cumulatively for better performance.

Third Party Credential Setup

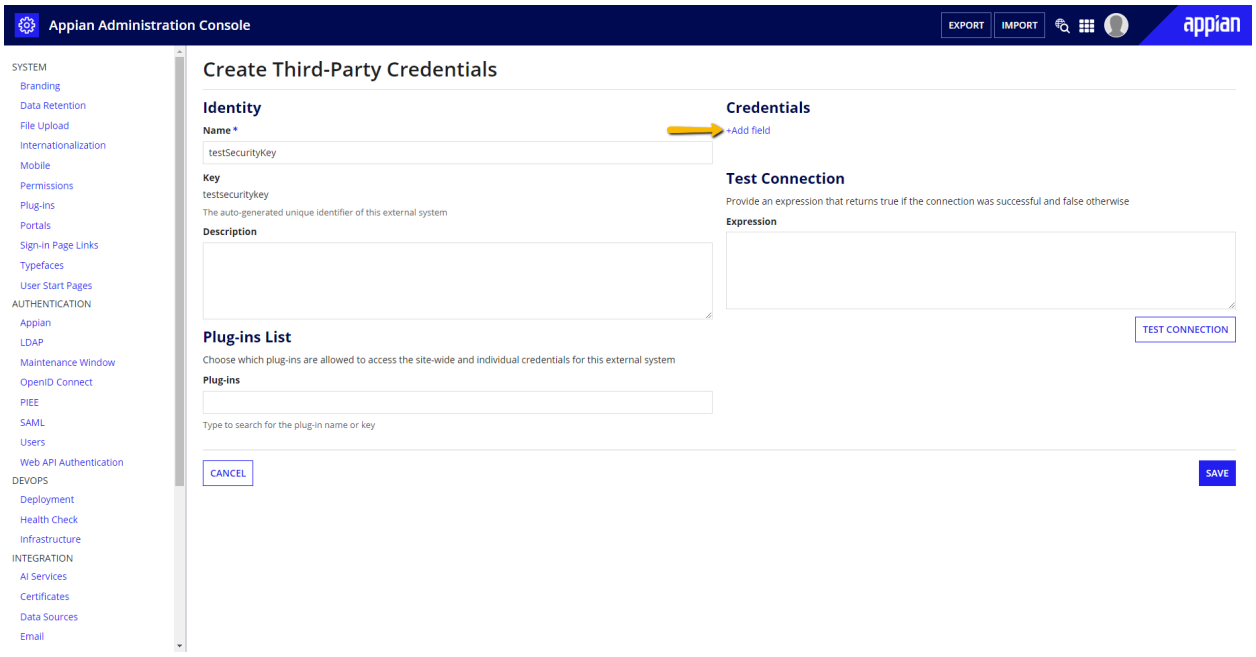
- Navigate to the Admin console -> Third Party Credentials in your Appian environment.

- Click on the CREATE button to create a new credential.

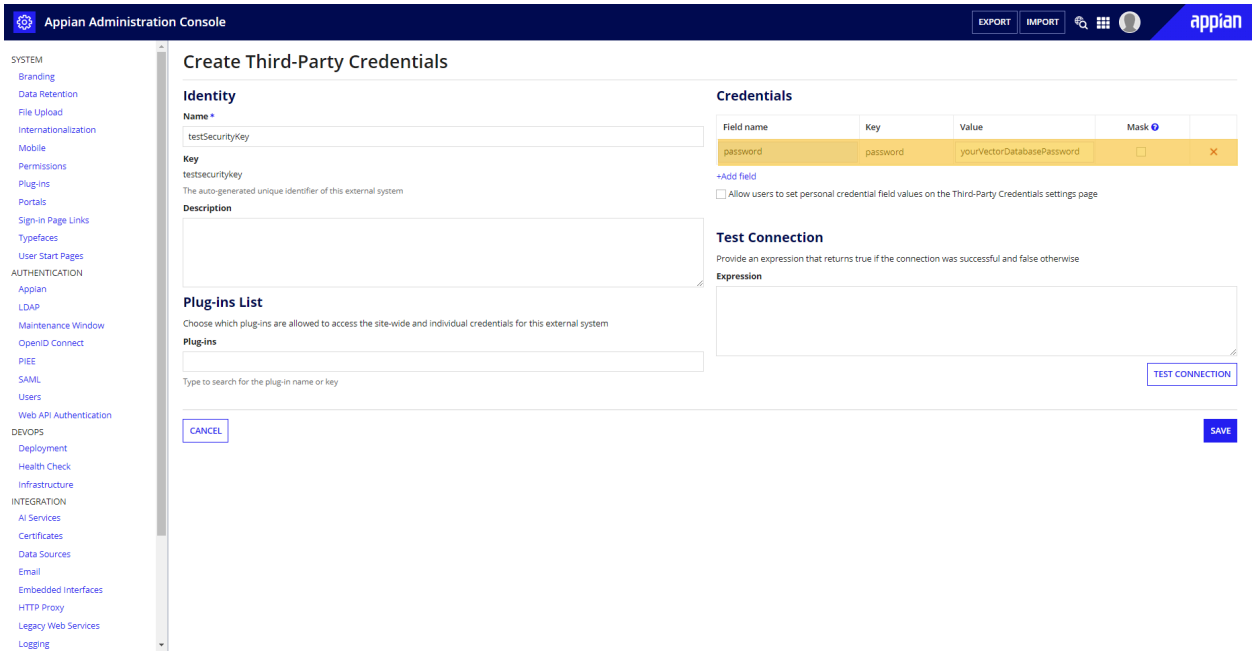
- Provide a name for the identity and a unique key will be auto-generated below the name field. Use this auto-generated key as the value for the 'securityKey' parameter in the component configuration to setup document security.



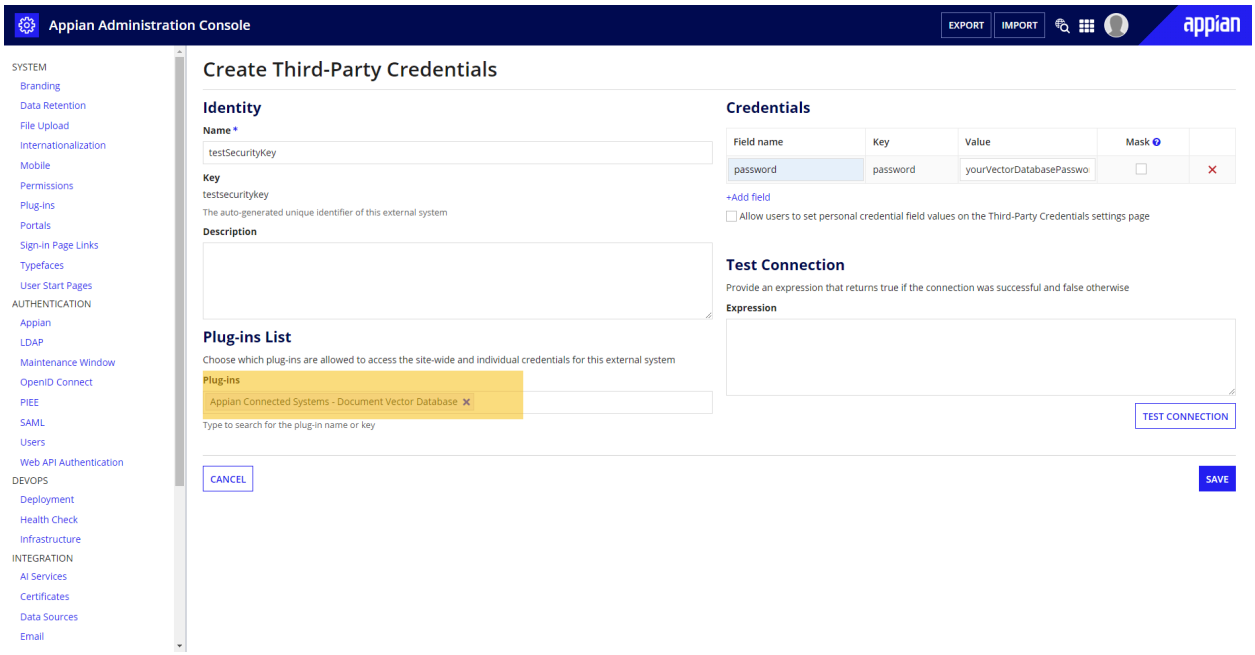
- Click on **+Add Field** to add a new credential.



- Provide the field name as **password** and the value as the same as your Database Password input in the Document Vector Database Connected System configuration.



- Add the Document Vector Database Connected System plugin to the Plugins Allow List as shown below and click on Save to save this configuration.



Performance Metrics Table for Document Querying

The below metrics are for reference on the performance of the Document Querying functionality.

Note: By using a caching mechanism to securely store and more quickly retrieve document information, we have greatly reduced the time required for each document query. This improvement can be seen in consecutive queries to the same document. The initial query to a document will be a non-cached query; any subsequent queries to the same document will experience the faster times of a cached query, as described in the chart below.

Total Number of Pages Being Queried (approx.)	**Time taken to complete Document Querying and Chat Completions (in seconds)**			
	Non Cached		**Cached**	
	GPT-3.5-Turbo	**GPT-4**	**GPT-3.5-Turbo**	**GPT-4**
100	4.30	25.87	3.45	23.00
200	5.78	37.55	4.61	29.31
500	7.21	29.44	5.80	25.52
1000	8.55	18.96	4.29	11.87

Document Vector Database Connected System Version Compatibility

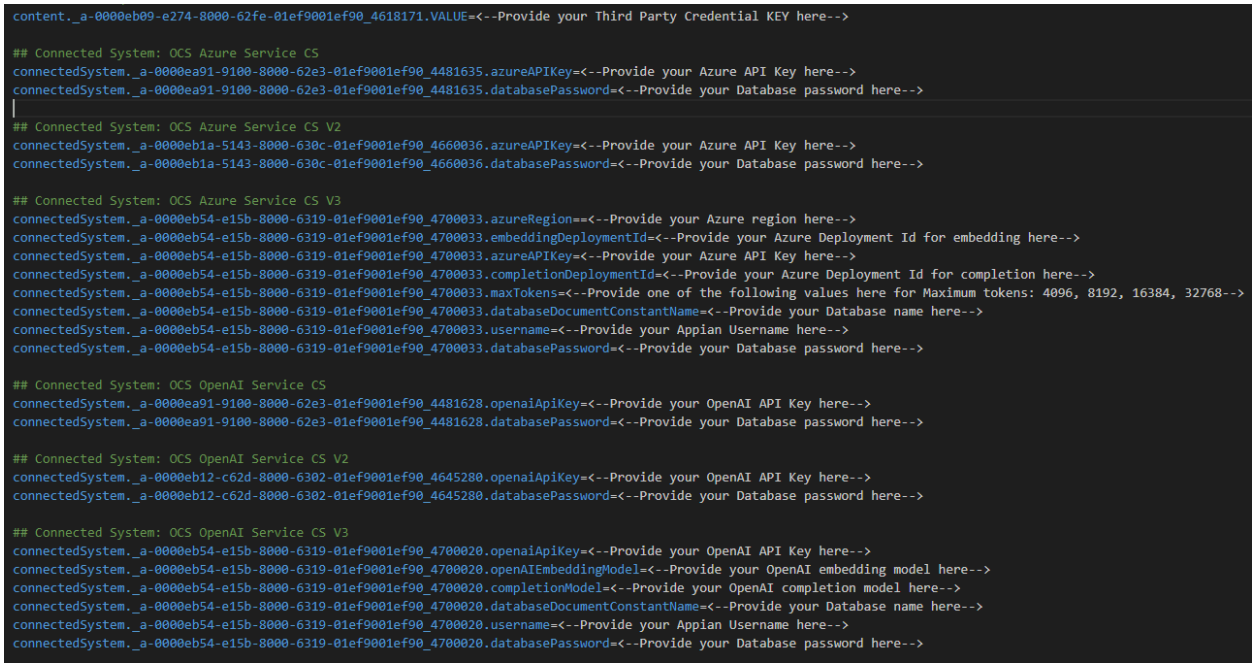
The following table shows the compatible versions of Document Vector Database Connected System for each AI Knowledge Assistant Component version.

Sl. No	Component Version	Compatible Connected System Versions
1	2.0.0	1.0.0 1.0.1 1.0.2 1.0.3 1.0.4
2	2.1.0	2.0.0 2.0.1
3	2.1.1	2.0.0 2.0.1
4	3.0.0	3.0.0 3.0.1
5	3.0.1	3.0.0 3.0.1
6	3.0.2	3.1.0 3.1.1 3.1.2 3.1.3
7	3.0.3	3.1.0 3.1.1 3.1.2 3.1.3
8	3.0.4	3.1.0 3.1.1 3.1.2 3.1.3
9	3.1.0	3.2.0

Sample Application Setup

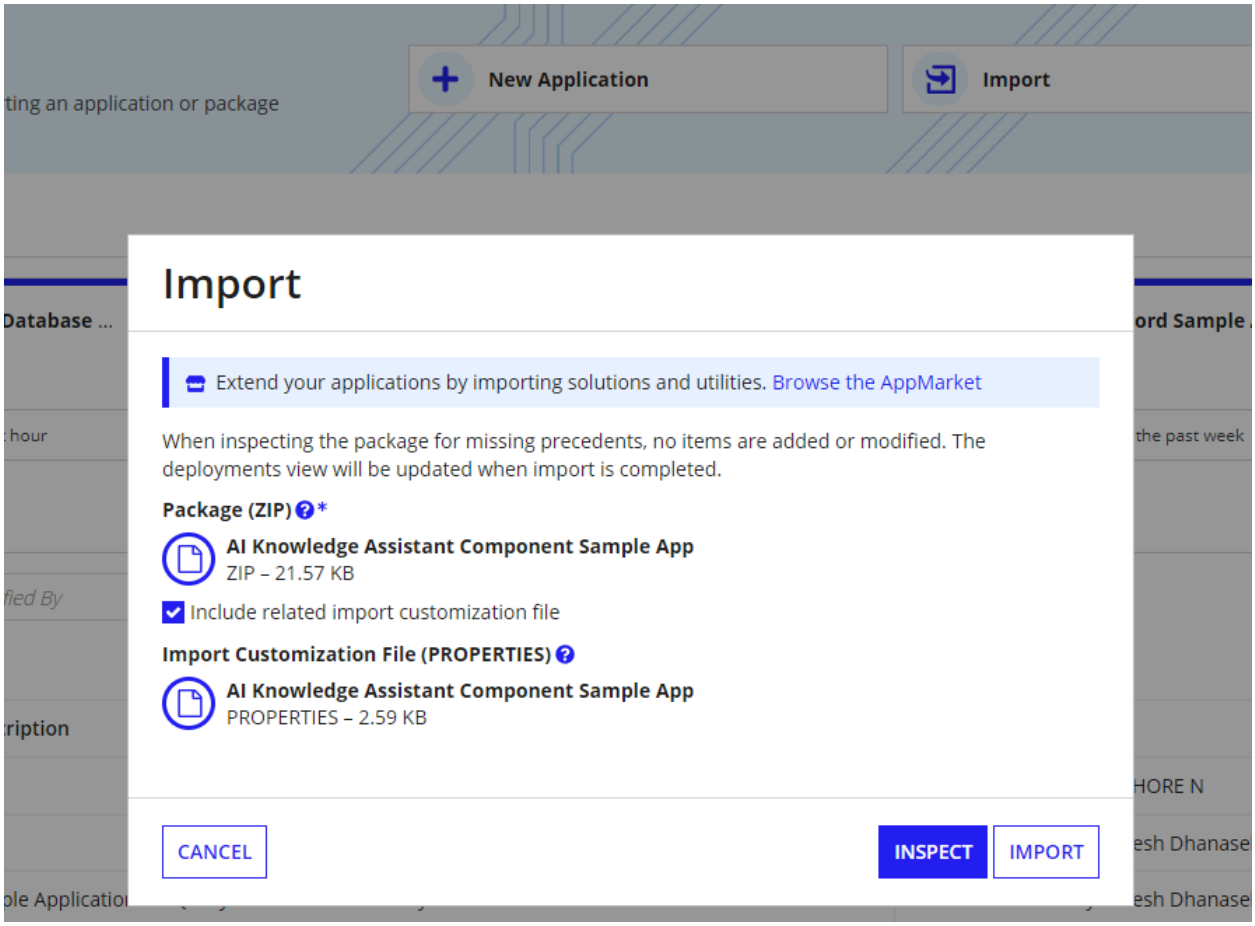
This will walk you through importing the sample application

1. Open the "AI Knowledge Assistant Component Sample App.properties" file in a text editor.



```
content._a-0000eb09-e274-8000-62fe-01ef9001ef90_4618171.VALUE=<--Provide your Third Party Credential KEY here-->

## Connected System: OCS Azure Service CS
connectedSystem._a-0000ea91-9100-8000-62e3-01ef9001ef90_4481635.azureAPIKey=<--Provide your Azure API Key here-->
connectedSystem._a-0000ea91-9100-8000-62e3-01ef9001ef90_4481635.databasePassword=<--Provide your Database password here-->

## Connected System: OCS Azure Service CS V2
connectedSystem._a-0000eb1a-5143-8000-630c-01ef9001ef90_4660036.azureAPIKey=<--Provide your Azure API Key here-->
connectedSystem._a-0000eb1a-5143-8000-630c-01ef9001ef90_4660036.databasePassword=<--Provide your Database password here-->

## Connected System: OCS Azure Service CS V3
connectedSystem._a-0000eb54-e15b-8000-6319-01ef9001ef90_4700033.azureRegion=<--Provide your Azure region here-->
connectedSystem._a-0000eb54-e15b-8000-6319-01ef9001ef90_4700033.embeddingDeploymentId=<--Provide your Azure Deployment Id for embedding here-->
connectedSystem._a-0000eb54-e15b-8000-6319-01ef9001ef90_4700033.azureAPIKey=<--Provide your Azure API Key here-->
connectedSystem._a-0000eb54-e15b-8000-6319-01ef9001ef90_4700033.completionDeploymentId=<--Provide your Azure Deployment Id for completion here-->
connectedSystem._a-0000eb54-e15b-8000-6319-01ef9001ef90_4700033.maxTokens=<--Provide one of the following values here for Maximum tokens: 4096, 8192, 16384, 32768-->
connectedSystem._a-0000eb54-e15b-8000-6319-01ef9001ef90_4700033.databaseDocumentConstantName=<--Provide your Database name here-->
connectedSystem._a-0000eb54-e15b-8000-6319-01ef9001ef90_4700033.username=<--Provide your Appian Username here-->
connectedSystem._a-0000eb54-e15b-8000-6319-01ef9001ef90_4700033.databasePassword=<--Provide your Database password here-->

## Connected System: OCS OpenAI Service CS
connectedSystem._a-0000ea91-9100-8000-62e3-01ef9001ef90_4481628.openaiApiKey=<--Provide your OpenAI API Key here-->
connectedSystem._a-0000ea91-9100-8000-62e3-01ef9001ef90_4481628.databasePassword=<--Provide your Database password here-->

## Connected System: OCS OpenAI Service CS V2
connectedSystem._a-0000eb12-c62d-8000-6302-01ef9001ef90_4645280.openaiApiKey=<--Provide your OpenAI API Key here-->
connectedSystem._a-0000eb12-c62d-8000-6302-01ef9001ef90_4645280.databasePassword=<--Provide your Database password here-->

## Connected System: OCS OpenAI Service CS V3
connectedSystem._a-0000eb54-e15b-8000-6319-01ef9001ef90_4700020.openaiApiKey=<--Provide your OpenAI API Key here-->
connectedSystem._a-0000eb54-e15b-8000-6319-01ef9001ef90_4700020.openAIEmbeddingModel=<--Provide your OpenAI embedding model here-->
connectedSystem._a-0000eb54-e15b-8000-6319-01ef9001ef90_4700020.completionModel=<--Provide your OpenAI completion model here-->
connectedSystem._a-0000eb54-e15b-8000-6319-01ef9001ef90_4700020.databaseDocumentConstantName=<--Provide your Database name here-->
connectedSystem._a-0000eb54-e15b-8000-6319-01ef9001ef90_4700020.username=<--Provide your Appian Username here-->
connectedSystem._a-0000eb54-e15b-8000-6319-01ef9001ef90_4700020.databasePassword=<--Provide your Database password here-->
```

2. Replace the Third Party Credential KEY placeholder with the auto-generated key obtained while setting up Third Party Credentials.
3. Replace the placeholder text with your Azure and OpenAI API keys. For Database password and Database name(from V3 only), provide a password and a name that will be used as the password and database name for H2 Database. Provide suitable values for other fields. Save the file.
4. Import the "AI Knowledge Assistant Component Sample App.zip" file into your Appian environment. Check on the Include related customization file check box and upload the properties file saved in the previous step.



5. Navigate to the "AI Knowledge Assistant Demo" site, click the site link, and test out the component.