**ChatDSI – Technical Design Document**

**Revision History**

| **Version No** | **Version Date** | **Change Reference** | **Changes** | **Initiator** |
| --- | --- | --- | --- | --- |
| 1.0 | 22nd May’24 |  | Initial draft |  |
| 2.0 | 18th June’24 |  | UI revamp, addition of Regenerate and edit options, Incorporation of GPT-4o |  |
| 3.0 | 7th Aug’24 |  | Addition of Translator functionality (For response and document) and Image Analysis |  |
| 4.0 | 5th Sept’24 |  | Addition of Voice Interaction and Citations for File Upload |  |
| 5.0 | 14th Sept’24 |  | Bulk Upload(file limit extended up to 200 MB) |  |
| 6.0 | 13th Jan’25 |  | SharePoint Connection |  |
| 7.0 | 13th Jan’25 |  | Teams Integration |  |

**Table of Contents**

[**1** **OVERVIEW** 4](#_Toc169619432)

[**1.1 INTRODUCTION** 4](#_Toc169619433)

[**1.2 PURPOSE OF THE DOCUMENT** 4](#_Toc169619434)

[**1.3 IN SCOPE** 4](#_Toc169619435)

[**1.4 OUT OF SCOPE** 4](#_Toc169619436)

[**1.5 ASSUMPTIONS, CONSIDERATIONS AND RISKS** 5](#_Toc169619437)

[***1.5.1 Assumptions*** 5](#_Toc169619438)

[***1.5.2 Risks*** 5](#_Toc169619439)

[**1.6 REFERENCES** 5](#_Toc169619440)

[**1.7 ACRONYMS AND ABBREVIATIONS** 6](#_Toc169619441)

[**1.8 EXCLUSIONS** 6](#_Toc169619442)

[**2** **ARCHITECTURE AND DESIGN** 6](#_Toc169619443)

[**2.1** **High Level Architecture** 6](#_Toc169619444)

[**2.1.1** **Architecture Diagram for General QnA** 6](#_Toc169619445)

[**2.1.2** **Architecture Diagram for File-Upload (PDF/DOCX/PPTX/TXT)** 7](#_Toc169619446)

[**2.1.3** **Architecture Diagram for File-Upload (CSV/Excel)** 7](#_Toc169619447)

[**2.2** **DESIGN DETAILS** 8](#_Toc169619448)

[**2.3** **TECH STACK** 8](#_Toc169619449)

[**2.4** **PROCESS FLOW REPRESENTATION** 9](#_Toc169619450)

[**2.5** **UI DESIGN CHANGES** 10](#_Toc169619451)

[**3** **ENVIRONMENT DETAILS** 12](#_Toc169619452)

[**4** **DATABASE ARTIFACTS** 12](#_Toc169619453)

[**4.1** **DATABASE DESIGN** 12](#_Toc169619454)

[**5** **ERROR HANDLING MECHANISM** 12](#_Toc169619455)

**6 API DETAILS**………………………………………………………………………………………………………..………………………….13

[**7** **SECURITY CONSIDERATIONS IN DATA PROCESSING** 14](#_Toc169619456)

# **OVERVIEW**

## **1.1 INTRODUCTION**

“ChatDSI” is a browser-based Generative Artificial Intelligence (AI) assistant developed by DSI. This chatbot leverages advanced natural language processing to understand and respond to text-based user queries, facilitating seamless conversational interactions. The addition of the File Upload feature further streamlines user interactions, making ChatDSI an indispensable tool for modern information needs.

## **1.2 PURPOSE OF THE DOCUMENT**

The purpose of this document is to provide overview of ChatDSI's functionalities, technical specifications, and high-level architecture diagrams to guide its development and implementation.

## **1.3 IN SCOPE**

* Single Sign-On (SSO): The application will integrate with Azure Active Directory (Now known as Microsoft Entra ID) for seamless user authentication and access management.
* Azure OpenAI Models: Utilizes the GPT-4o model to deliver precise responses, enhancing user productivity and information retrieval.
* Copy/Export Response: Users can copy and export responses in PDF, TXT, or DOCX formats, improving documentation and sharing capabilities.
* File Analysis: The system can analyze uploaded files, summarize text, extract pertinent information, and answer related questions using the GPT-4o and Azure Cognitive service. GPT 4 Turbo model to be used for excel/ csv file upload. For all other file formats GPT 4o will be used.
* Feedback Mechanism: A 'Feedback' button directs users to a Microsoft Form for submitting suggestions and concerns, promoting continuous improvement.
* Override Prompt: Allows users to customize their chat experience, such as setting the assistant to function as a 'French translator.'
* Reset button on the side of the input bar: Enables users to clear chat history from the current session, ensuring privacy and a fresh start.
* Help Window: Provides immediate assistance and guidance on application functionalities.
* File Upload: Users can upload multiple files (PDF, DOCX, PPTX, TXT) simultaneously, or single CSV/Excel files, streamlining workflow efficiency.
* Regenerate Chat: Enables users to regenerate another response for the same prompt.
* Edit prompt: Enables users to edit the original prompt to generate the appropriate response without typing out the prompt again.
* Prompt Suggestions: Provides users with prompt suggestions at the start of the chat.
* Response Translation: Allows users to translate the chat responses into different languages, enhancing accessibility and comprehension for non-native speakers.
* File Translation: Provides users with the capability to translate uploaded documents (PDF, DOCX, PPTX, TXT, CSV, EXCEL) into different languages, facilitating multilingual document handling.
* Image Analysis: The system can analyze uploaded images (JPG, PNG, JPEG, BMP, WEBP, TIF, GIF), summarize them, extract information, and answer related questions using the GPT-4o.
* Voice Interaction: Enables users to perform searches using the mic, making interactions more intuitive and accessible by recognizing spoken queries.
* Read Aloud: Converts the chat responses to speech, allowing users to listen to the responses, enhancing accessibility for those who prefer auditory information.
* Citations include links to the specific page/ slide in the document, stored in the Azure container.
* Citations will be provided when multiple images are uploaded
* Bulk Upload: Users can now upload files(.pdf,.docx,.pptx,.txt) up to 200MB in size limit. Increased size limit from 10mb to 200mb will allow users to query or summarize large amount of data effortlessly and efficiently.
* Microsoft Teams integration: Users will be able to access the ChatDSI application within the Microsoft Teams Application.
* Microsoft SharePoint Integration: Users will be able to upload files directly from their SharePoint Repository (which they have access to) using the link of their SharePoint folder. Users can select the required file/folder from the SharePoint folder (from the UI) and query the files.
* Context reset: Whenever user switches from Generic QnA to Document QnA or vice-versa, context will get reset ie user will not get response from the previous conversation.
* Keywords added for Summarizing uploaded documents or getting the count of uploaded documents.

## **1.4 OUT OF SCOPE**

* Database Access: The web application will not retrieve or access information from company-related databases or documents, operating independently from these data sources.
* Human Oversight: While providing intelligent responses, the application will not replace human oversight, especially for critical accuracy and appropriateness. Human intervention will be necessary to ensure quality and compliance.
* Custom Data Integration: Integration with custom or proprietary data sources specific to the company's internal systems is not within the scope.
* Model Training: Extensive model training or continuous improvement for specific company-related knowledge domains is not included. The application will rely on the existing capabilities of the GPT model without custom domain training.
* Real-Time Editing: The file upload feature does not support real-time editing or collaboration on documents. Users cannot modify uploaded files directly within ChatDSI.
* File Size Limitations: There will be restrictions on the size of uploaded files. Large files exceeding the set size limit will not be supported to ensure optimal performance and responsiveness.
* CSV and Excel Uploads: Users are restricted to uploading a single file at a time for CSV and Excel formats, unlike the multiple file uploads allowed for other formats.
* The model can generate and understand text from document content but has limited capabilities in processing and interpreting visual information, such as embedded images or graphics within documents.
* It does not support translation of images. It focuses on translating text-based documents and will not extract or translate text embedded in images or graphics.
* Simultaneous processing of multiple files for translation is not supported. Users can upload and translate one file at a time into multiple languages.
* Only supports JPG, PNG, JPEG, BMP, WEBP, TIF, and GIF file formats for image upload.
* Citation is only supported for PDF, DOCX, PPTX and TXT file formats and above-mentioned image formats.
* Images within documents will not be consider for querying by default. User will have to manually select/ignore processing the images within the documents.
* Processing time more than 320 seconds will lead to timeout.
* Sharepoint Upload: for CSV and Excel Uploads, Users are restricted to uploading a single file at a time for CSV and Excel formats, unlike the multiple file uploads allowed for other formats. If users try to upload CSV/Excel file along with other format files, CSV/Excel files won’t be uploaded and user will get a message on screen for the same.
* Teams Integration: Citations will not open inside the application, although it will open in another window in browser.

## **1.5 ASSUMPTIONS, CONSIDERATIONS AND RISKS**

### ***1.5.1 Assumptions***

|  |  |
| --- | --- |
| ***ID*** | ***DESCRIPTION*** |
| *1.5.1.1* | *The performance of the GPT model, including its accuracy and response quality, is assumed to be consistent with its documented capabilities.* |

### ***1.5.2 Risks***

|  |  |
| --- | --- |
| ***ID*** | ***DESCRIPTION*** |
| *1.5.2.1* | *ChatDSI might generate responses that are factually incorrect or misleading. It may have biases that could result in incorrect or inappropriate responses, posing reputational and compliance risks.* |
| *1.5.2.2* | *An influx of large files or a high volume of uploads could potentially overload the system, leading to slower response times or disruptions in service.* |
| *1.5.2.3* | *Technical issues, such as bugs in the file upload process or compatibility problems with certain file types, could hinder the functionality and user experience, affecting the overall utility of the feature.* |

## **1.6 REFERENCES**

<<Place document references in alphabetical/numerical order>>

| Document Number | Document Name |
| --- | --- |
| SOP-XXX | Standard operations manual of X application |

## **1.7 ACRONYMS AND ABBREVIATIONS**

<<Provide additional terms, acronyms, and abbreviations required to properly interpret this document>>

| Acronym | Definition |
| --- | --- |
| DSI | Daiichi Sankyo Inc. |

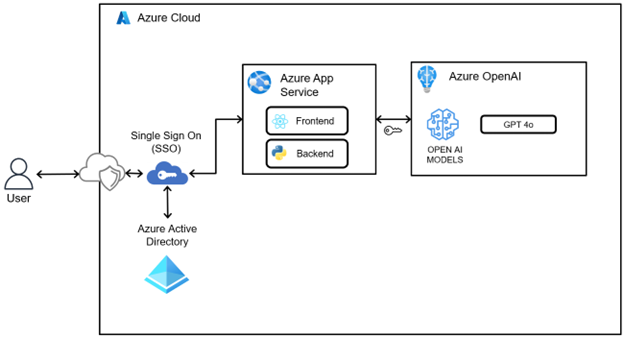
## **1.8 EXCLUSIONS**

Mention the sections which are not applicable for the Application with the appropriate reason.

# **ARCHITECTURE AND DESIGN**

## **High Level Architecture**

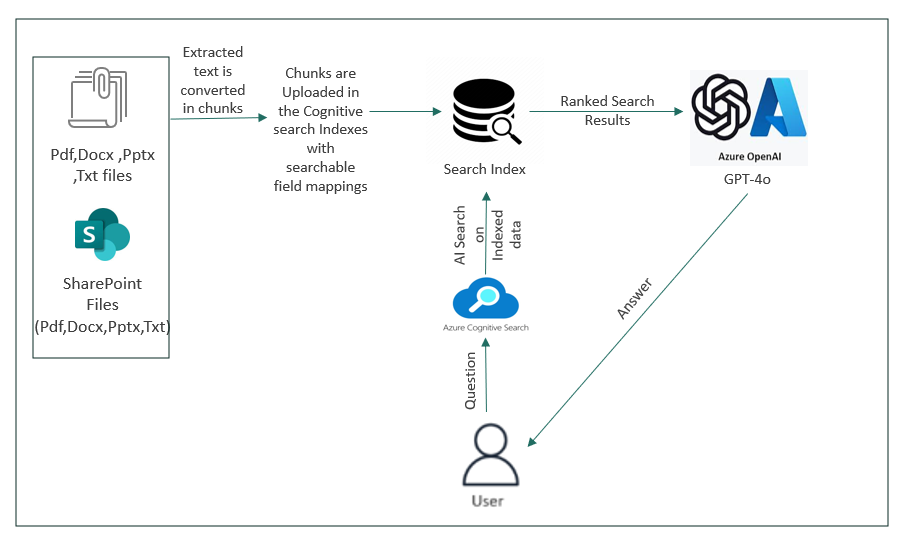
### **Architecture Diagram for General QnA**



**Process Flow**

1. User submits a user query from the chat UI.
2. Backend captures the current timestamp and latest user message.
3. Metadata (user, timestamp, token count) is logged for analytics.
4. An OpenAI **ChatCompletion** request is created with streaming enabled.

### **Architecture Diagram for File-Upload (PDF/DOCX/PPTX/TXT)**



1. User clicks Upload and selects one or more documents.

2. User chooses processing mode:

* Continue without images → skip image extraction/analysis
* Continue → enable image extraction via xref/fitz

3. docx or txt files converted to pdf using LibreOffice.

4. Index creation begins using Azure Cognitive Search service.

5. Convert remaining pptx files (if any) to pdf using LibreOffice.

6. Upload all PDFs to Azure Blob Storage.

### **Architecture Diagram for File-Upload (CSV/Excel)**

A diagram of a chat dsi

Description automatically generated

**The model being used for CSV/ Excel file upload is GPT 4 Turbo**

**Process Flow :**

1. Users upload a single Excel or CSV file through the interface (only one file is allowed at a time).
2. The backend reads the file and creates a corresponding table in a SQLite database.
3. Metadata (table name, column names, and data types) is extracted from the created SQLite table.
4. The metadata and user's natural language question are passed to the GPT-4 Turbo model to generate a SQL query.
5. The generated SQL query is executed, and the result is returned to the user as an appropriate answer.

### **Architecture Diagram for Image Upload**

A screenshot of a computer

Description automatically generated**Process Flow :**

1. Users upload multiple image files through the interface.
2. Each uploaded image is converted to a Base64-encoded string.
3. The Base64-encoded images are passed to the GPT-4o model along with the user's natural language question.
4. GPT-4o processes the visual content in the images and interprets the question in context.
5. The model generates an appropriate response based on the image content and the question, which is then displayed to the user.

## **2.1.5 Architecture Diagram for Document Translation**

A screenshot of a computer

Description automatically generated

**Process Flow**

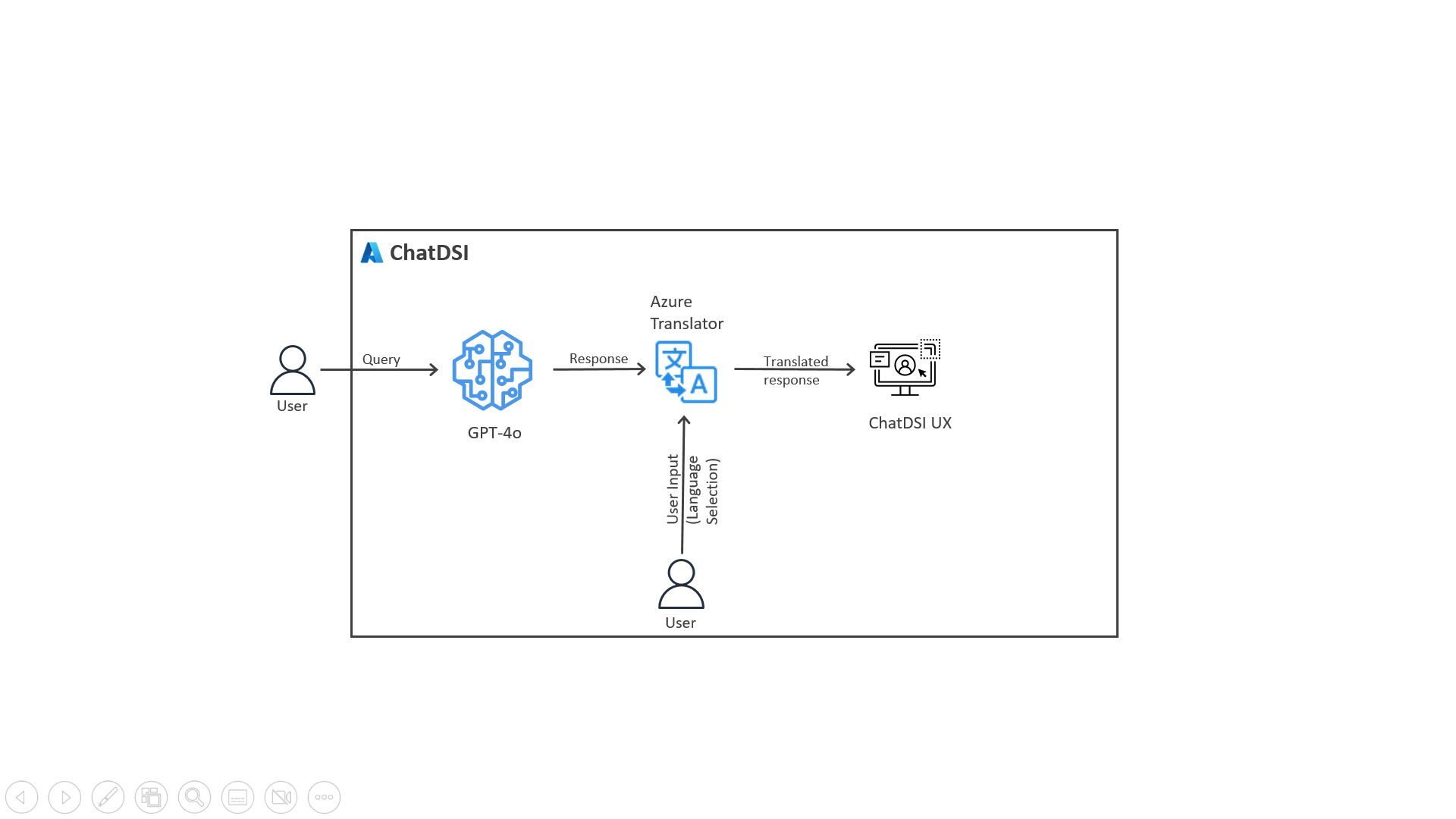
1. User uploads a document and selects 1–3 target languages.
2. File is temporarily saved and uploaded to Azure Blob Storage (Source container).
3. Previously translated files (if any) under the user's folder in the Target container are deleted.
4. For each target language: Call the Azure Translator API’s begin\_translation method for file conversion,
5. Translated files are saved in the Target container under the user's folder.
6. Blob SAS URLs are generated for users to download translated files.
7. Response is sent back to the frontend with download links.

## **2.1.6** **Architecture Diagram for Response Translation**

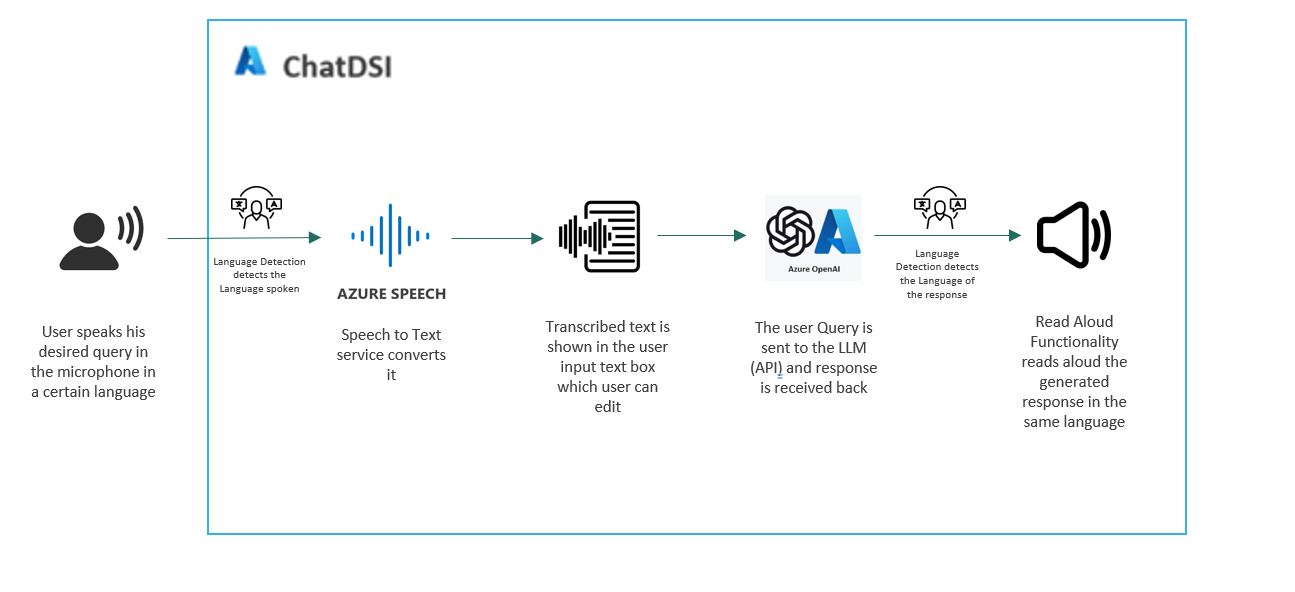
A screenshot of a computer

Description automatically generated

**Process Flow**

1. 
2. On clicking the translate button, a window opens prompting the user to select the desired language to be translated to.
3. On clicking translate, azure translation service API is called, and the latest response is sent. The Translated response is fetched and displayed in the front-end as a new response.
4. Language selection: 13 supported languages.
5. Translation Service <https://api.cognitive.microsofttranslator.com> is used which is from microsoft azure cognitive translator.

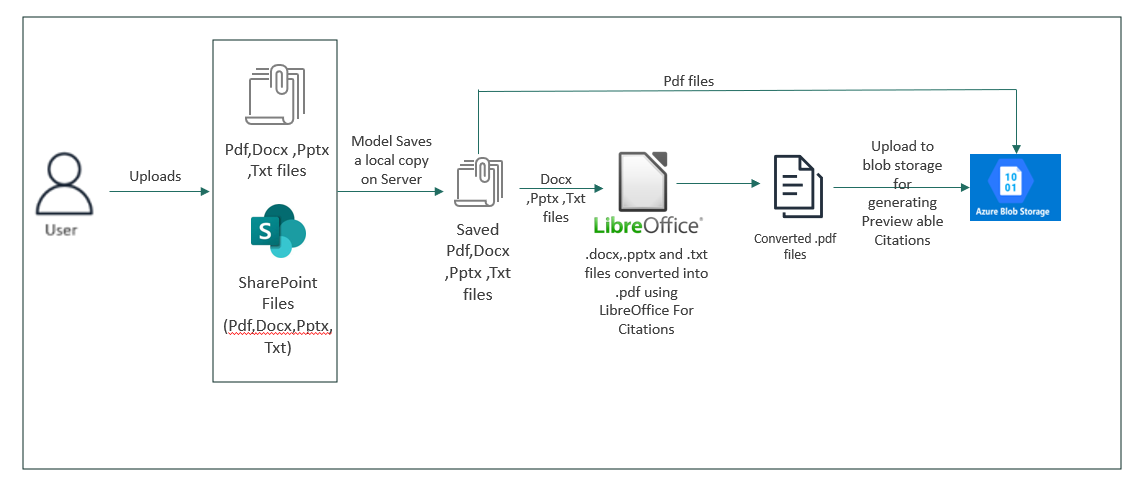
## **2.1.7 ARCHITECTURE DIAGRAM FOR VOICE RECOGNITION**



**Process Flow :**

1. User starts the voice recognition feature and speaks a query.
2. Azure Speech Service detects the spoken language (supports Hindi, English, Japanese) and converts the speech into text.
3. The transcribed text is auto-filled into the user input text box, allowing the user to edit it if needed.
4. The user sends the final query to the backend for processing.
5. Once the response is received, the user has the option to use the "Read Aloud" feature.
6. When the user clicks the "Read Aloud" button, the system detects the language of the response using a React inbuilt library and reads the answer aloud accordingly.

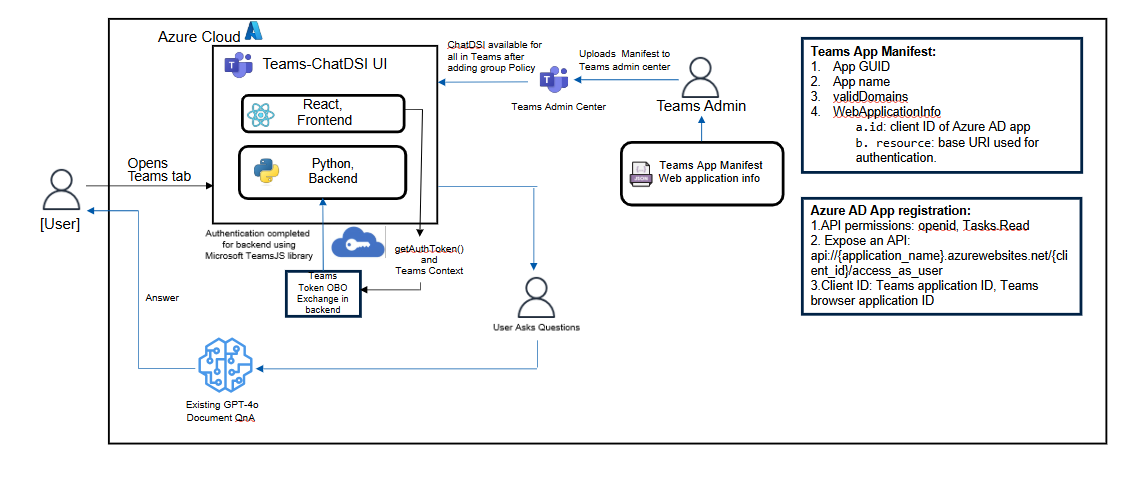
## **2.1.8 ARCHITECTURE DIAGRAM FOR CITATION (DOCX, PPTX ,TXT or PDF)**



**Process Flow**

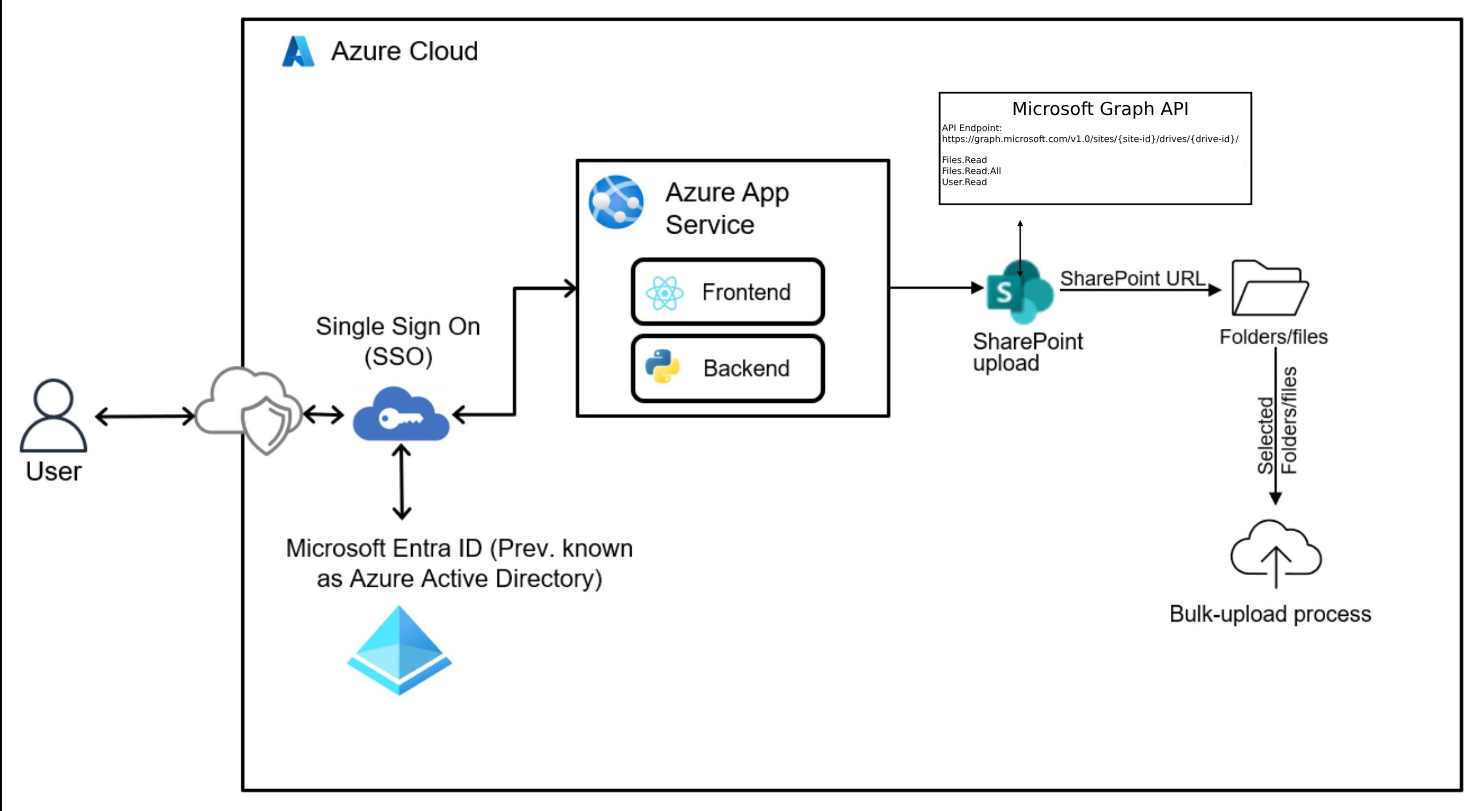
1. Users upload PDF,PPT, DOCX, or TXT files through the interface.
2. The uploaded files are first saved in a local folder on the server.
3. Pdf files are then uploaded to Azure Blob Storage without any conversion.
4. PPTX, DOCX and TXT files are then converted into PDF format for standardized processing.
5. After conversion, all the PDF files are uploaded to Azure Blob Storage.
6. When the user asks a question, the model processes the content and generates an answer.
7. A readable Blob Storage URL with the relevant page number is generated and appended to the answer as a citation.
8. When the user asks a question, the model processes the content and the rele
9. vant citation for the blob file is provided as a pre-viewable link.

## **2.1.9 ARCHITECTURE DIAGRAM FOR TEAMS INTEGRATION**

**Process Flow:**

* The user opens the Teams tab app, which loads in the Teams client context.
* The frontend calls microsoftTeams.authentication.getAuthToken() to obtain a Teams client token (JWT) for the signed-in user.
* This token is sent to the backend via an API call (e.g., storeTeamsToken) for further processing.
* The backend validates the Teams token using Azure AD and extracts the user info (e.g., AAD object ID, tenant ID).
* The backend then performs an **On-Behalf-Of (OBO)** flow using the Teams token to request a new access token for Microsoft Graph.
* Using the Graph token, the backend checks if the user is a member of a specific Azure AD group.
* If the user is part of the group, they're considered authenticated and granted access.
* Otherwise, the backend responds with an unauthorized error or access restriction message.

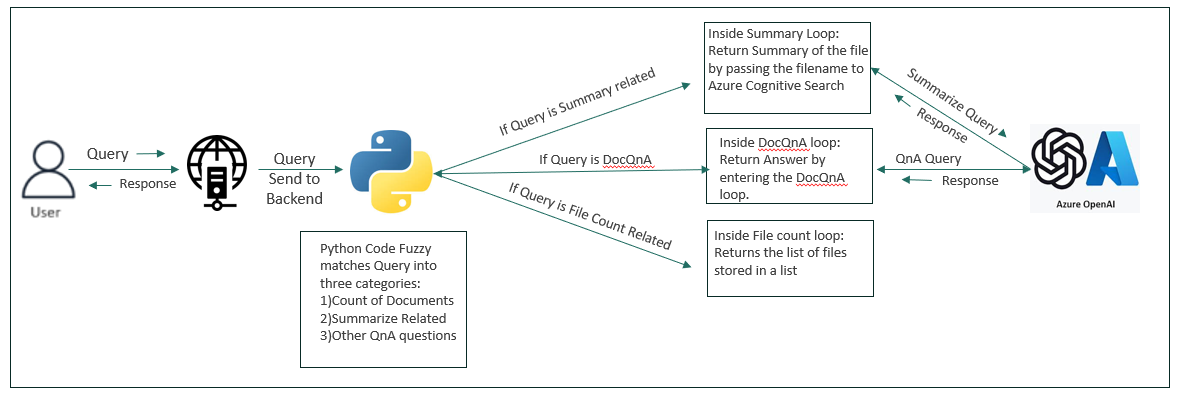
## **2.1.10 ARCHITECTURE DIAGRAM FOR SHAREPOINT UPLOAD**



**System Flow**

1. User logs into ChatDSI using Single Sign-On (SSO).
2. The user uploads a SharePoint folder link through the interface.
3. The system verifies the user's access to the folder using SSO credentials and SharePoint REST APIs.
4. If access is granted, all the files available in the linked SharePoint folder are displayed to the user.
5. The user can select the desired files from the list and proceed.
6. Once selected, the processing of the files follows the same flow as the regular file upload (PDF, PPT, TXT, DOCX).
7. Added support for parsing and handling hierarchical site structures (nested sites and sub-sites), enabling seamless data extraction.
8. Added support for parsing and processing Windows-generated links (shortcuts and URL of folders), ensuring accurate resolution and handling within the application.

## **2.1.11 ARCHITECTURE DIAGRAM FOR SUMMARIZE/COUNT OF FILES:**



* Users upload one or more documents through the interface for QnA.
* When the user asks a question, the system first categorizes the query using keyword detection and fuzzy matching techniques.
* Based on the categorization, the system determines whether the question is related to summarization, list or count of documents, or general querying.
* If the question is about summarization, the user is asked to provide the specific document name they want to summarize, and the system provides the appropriate summary using Azure cognitive search and GPT 4o.
* If the question involves listing or counting, such as "list all documents" or "how many documents," a relevant list or count is returned.
* If the question is not tied to a specific document or type, the system uses Azure Cognitive Search along with the Azure OpenAI GPT-4o model to generate and return a relevant answer.

## **DESIGN DETAILS**

|  |  |
| --- | --- |
| **Layer** | **Description** |
| Frontend | React, Typescript |
| Backend | Python, Quart framework |

## **TECH STACK**

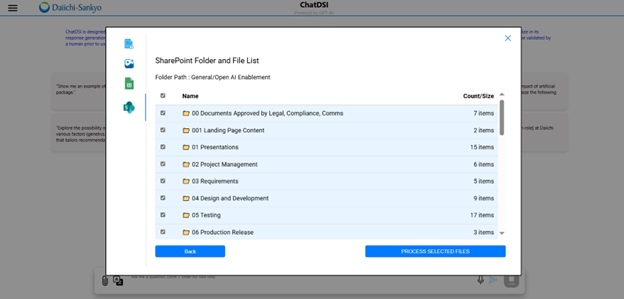
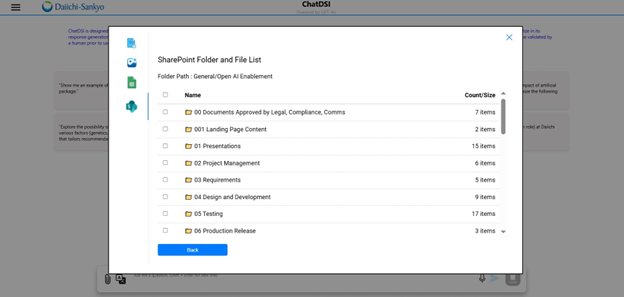
* **GPT-4o Completion API** from Azure OpenAI for generating text responses.
* **Azure DevOps** for continuous integration and continuous deployment (CI/CD), repository management, and application monitoring.
* **Visual Studio Code** with Azure Extensions for development
* **Microsoft Entra ID** (Previously known as Azure Active Directory (AAD)) for identity services, ensuring secure access and authentication to your application.
* **Quart,** an asynchronous web framework for Python, is utilized to streamline the development and management of ChatDSI web application.
* **FAISS:** Used to store embeddings, enhancing quick and efficient similarity searches within the ChatDSI's document querying capabilities.
* **SQLite**, a lightweight database for temporarily storing user uploads (for CSV and Excel).
* **aioodbc**, an asynchronous ODBC library, was used for efficient logging of user information (user’s email, login time, tokens used etc.) into a **SQL database**.
* **Tiktoken** library is used to count number of tokens utilized.
* OpenAI Models are deployed as follows: **GPT-4o** (Limit- 960k TPM [deployed on OpenAI-Prod-East]).
* Azure Translator service is deployed as **‘Prod-OpenAITranslator’** under ‘RG-Prod-Openai’ resource group (Pricing tier: S1) and is utilized for translating both documents and responses. Azure Blob Storage **‘saprodopenai’** is utilized for temporarily storing Uploaded document as well as translated document.
* Azure vision service is deployed as ‘**prod-openai-computervision’** under ‘RG-Prod-Openai’ resource group (Pricing tier: S1) and is utilized for image analysis.
* Azure speech service is deployed as '**prod-openai-speech**' under 'RG-Prod-Openai' resource group and is utilized for voice interactions.
* LibreOffice is used to convert .docx.,.txt,.pptx file to .pdf files.
* Azure storage **saprodopenai** is configured for temporary file storage.
* Azure AI Search service is configured as **bulk-upload-prod**
* Azure Storage container is named as **bulk-upload-qna** for storing documents for citations generation.
* Microsoft Teams Developers Portal for developing and testing ChatDSI teams applications while development.
* Library microsoft/teams-js used for getting the teams authentication access token from teams client.

## **PROCESS FLOW REPRESENTATION**

A diagram of a software application

Description automatically generated

## **UI DESIGN CHANGES – as of April release**Figure 1: Teams integration with ChatDSIFigure 2: SharePoint upload



# **ENVIRONMENT DETAILS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Category** | **Technology Name** | **Instance Name (DEV)** | **Instance Name (QA)** | **Instance Name (PROD)** |
| **Azure Resource manager** | **Resource Group** | RG-PoC-OpenAI | RG-QA-OpenAI | RG-Prod-OpenAI |
| **PaaS (Platform as a service)** | **Azure App Service** | dsi-general-assistant-dev | ChatDSI-General-Assistant-UAT | dsi-general-assistant |
| **Model deployment** | **Azure OpenAI** | OpenAI-PoC-East, OpenAI-PoC-East2 | OpenAI-UAT-East, openai-qa-east2 | OpenAI-Prod-East, openai-prod2-east2 |
| **Azure Service Plan** | **App Service Plan** | **ASP-RGPoCOpenAI-9588 (B2)** | **ASP-DSI-Assistant-0123 (P1v3)** | **ASP-DSI-Assistant-8607 (P1v3)** |
| **Azure Translator Service** | **Translator** | **Poc-DevTranslator** | **QA-OpenAITranslator** | **Prod-OpenAITranslator** |
| **Azure AI Search** | **Search Service** | poc-openai-cogsrch | bulk-upload-uat | bulk-upload-prod |
| **Blob Storage** | **Storage** | sapocopenai | saqachatdsi | saprodchatdsi |

# **DATABASE ARTIFACTS**

## **DATABASE DESIGN**

The following databases are used for logging user information- user email, response date, model used, and number of tokens used. Information like user’s input or output are not being saved.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Database** | **Table Name** | **Server Name** |
| DEV | AI\_ASSTNT | LOG\_TABLE\_KUMO\_AI | mscdtbdev01.8352d9ef34e6.database.windows.net |
| UAT | AI\_ASSTNT | LOG\_TABLE\_AI\_ASSTNT | mscdtbtst01.public.8352d9ef34e6.database.windows.net,3342 |
| PROD | AI\_ASSTNT | LOG\_TABLE\_AI\_ASSTNT | mscdtbprd01.public.3936fb327e22.database.windows.net,3342 |

# **ERROR HANDLING MECHANISM**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 250 | File Upload | csv\_excel\_already\_present\_error | When a user tries to upload a new document while there is already a CSV or Excel file uploaded, this error is shown. To mitigate this, user must click on clear document or delete file to remove already uploaded CSV/Excel file. |
| 2 | 260 | File Upload | unstructured\_data\_already\_uploaded | When a user tries to upload new CSV/Excel document while there is already a document is uploaded, this error is shown, To mitigate this, user must click on clear document or delete file to remove already uploaded document. |
| 3 | 400 | File Upload | csv\_excel\_error | When user tries to upload more than one CSV/Excel file, this error is shown, upload single file to avoid the error. |
| 4 | 500 | File Upload | file\_size\_error | When user upload files which consumes more tokens than set by organization for each user, this error is shown. Kindly upload smaller number of files to avoid this. |
| 5 | 600 | File Upload | try\_after\_some\_time | When concurrent users manage to exceed maximum tokens quota for organization, this error is shown. User can try again after some time. |
| 7 | 500 | GeneralQnA | Server Error | When assistance takes more than 150 seconds, the user gets a message suggesting a time out. User can refresh the page and try again. |
| 8 | 270 | File Upload | image\_unstructured\_error | Do not upload images and unstructured data together. |
| 9 | 290 | File Upload | image\_structured\_error | Do not upload images and structured data together. |
| 10 | 280 | File Upload | structured\_unstructured\_error | Upload either structured or unstructured data. |
| 11 | 291 | File Upload | delete\_previously\_uploaded\_img\_files | Delete stored image files before uploading new file. |
| 12 | 265 | File Upload | unstructured\_data\_already\_uploaded\_image | When a user tries to upload new CSV/Excel document while there is already an image uploaded, this error is shown. To mitigate this, user must click on clear document or delete file to remove already uploaded document. |
| 15 | 400 | DocTranslate | No languages part in the request | Happens when the 'languages' key is missing in the submitted form data. ✅ Ensure the frontend sends 'languages' as a form field in the POST request. |
| 16 | 400 | DocTranslate | No languages selected | Occurs when the 'languages' field is present but empty. ✅ Validate that the form field 'languages' contains a comma-separated list of language values. |
| 17 |  | DocTranslate | Runtime error if session backend is not configured properly | Quart requires a proper session interface. ✅ Ensure session handling (e.g., using SecureCookieSessionInterface or Redis) is configured correctly. |
| 18 | 305 | DocTranslate | translate\_error | Ensure proper OCR/document format is passed. Confirm backend translation service is working as expected. |
| 19 | 400 | SharePoint Upload | Missing selectedItems or sharePointUrl in request body | Ensure the request contains both selectedItems and sharePointUrl fields in JSON format. |
| 20 | 401 | SharePoint Upload | Missing access\_token or user\_name in session | Verify session initialization and login before calling the API. |
| 21 | 500 | SharePoint Upload | Failed to extract site, drive, or folder path | Check if the SharePoint URL is correct and follows the expected format. |
| 22 | 500 | SharePoint Upload | Unable to fetch folder ID from SharePoint | Confirm folder exists and is accessible. Check get\_folder\_id\_recursive() implementation. |
| 23 | 400 | SharePoint Upload | Selected items not found or improperly formatted | Ensure data['data'] contains valid file/folder entries with checked status. |
| 24 | 500 | SharePoint Upload | Error during SharePoint folder/file processing | Check process\_sharepoint\_folder() logic and validate SharePoint permissions. |
| 25 | 500 | SharePoint Upload | Failed to create or access user directory | Ensure server has permissions to create folders under sharetempdir. |
| 26 | 500 | SharePoint Upload | Failed to convert DOCX or TXT to PDF | Check for presence of required tools (e.g., LibreOffice or Python libraries). |
| 27 | 500 | SharePoint Upload | Error while creating index for processed files | Validate the create\_index() function and its inputs. |
| 28 | 500 | SharePoint Upload | Failed to convert PPTX to PDF or upload to Azure | Ensure required packages and Azure credentials are correctly configured. |
| 29 | 500 | SharePoint Upload | File upload to Azure failed | Inspect upload\_file\_to\_azure() and validate connectivity and credentials. |
| 30 | 500 | SharePoint Upload | Unknown internal server error | Enable error logging to capture and debug stack traces. |
| 31 | 400 | SharePoint Upload | Invalid request format. Expected JSON | Ensure the client sends a valid JSON body with Content-Type: application/json. |
| 32 | 400 | SharePoint Upload | Missing required fields (url) | Ensure that the url field is present in the request JSON. |
| 33 | 400 | SharePoint Upload | Invalid SharePoint URL | Make sure the SharePoint URL is a non-empty string and properly structured. |
| 34 | 401 | SharePoint Upload | Authentication required | Confirm user session includes a valid access\_token. Log in again if needed. |
| 34 | 500 | SharePoint Upload | Internal error initializing SharePoint connection | Check token validity, and investigate initialization logic in LoadSharePoint. |
| 36 | 400 | SharePoint Upload | Invalid SharePoint URL structure | Verify the structure of the SharePoint URL — it should contain valid site and drive IDs. |
| 37 | 403 | SharePoint Upload | You are not authorized to access this SharePoint site | Confirm user permissions on the specified SharePoint site. |
| 38 | 500 | SharePoint Upload | Failed to extract SharePoint details | Review the URL and network connectivity; ensure proper parsing logic in extract\_details. |
| 39 | 400 | SharePoint Upload | Invalid folder path | Ensure the specified folder path exists in SharePoint. |
| 40 | 500 | SharePoint Upload | Failed to fetch folder ID | Check folder hierarchy and get\_folder\_id\_recursive implementation. |
| 41 | 404 | SharePoint Upload | No files or folders found | Confirm that the specified folder is not empty. |
| 42 | 500 | SharePoint Upload | Failed to retrieve files from SharePoint | Inspect the get\_folder\_entity\_list function and validate API token/session. |
| 43 | 500 | SharePoint Upload | Invalid response from SharePoint API | Validate the format of the API response (expecting JSON with a status field). |
| 44 | 500 | SharePoint Upload | Internal Server Error | General error — check logs for detailed traceback. |

Limitations document: [Limitations Document.docx](https://dsg2ic.sharepoint.com/:w:/r/teams/DSTeam_CoE_AIML/_layouts/15/Doc.aspx?sourcedoc=%7B167E74F8-8E09-4043-B40B-701505C8B98C%7D&file=Limitations%20-%20Aug%20Release.docx&action=default&mobileredirect=true&DefaultItemOpen=1)

# **api details**

|  |  |  |
| --- | --- | --- |
| **API** | **Endpoint** | **Description** |
| askApi | /ask | Not used. |
| accessApi | /api/auth | This API checks if the user is authenticated |
| logoutApi | /api/logout | Logs out an authenticated user by clearing the session |
| chatApi | /chat\_stream | Handles streaming chat responses based on various approaches like general, translate, readaloud, file upload. |
| chatApi | /chat | Handles chat interactions without streaming, processes user input based on the selected approach, and returns a response. |
| uploadApi | /upload | Handles uploading of files (CSV, Excel, images, or documents (pdf, pptx, docx, txt)). It validates the file types, checks for duplicates or conflicts (like mixing structured and unstructured data), and returns success if the file processing is successful or specific error messages. |
| translateApi | /lang\_sel | It receives a list of user-selected target languages for document translation. It validates and stores the language preferences before initiating the actual file translation process. This API acts as the first step in the document translation workflow. |
| doctranslateApi | /doctranslate | It performs the actual translation of uploaded documents into the user-selected languages. It processes the files, translates the content, and returns translated file(s). |
| getSharePointFileListApi | /get-sharepoint-file-list | Retrieves a list of files from a given SharePoint folder URL. It returns the folder path and file details, enabling users to view and select SharePoint-hosted documents for further processing. |
| processSharePointFilesApi | /process-sharepoint-files | Processes selected documents from a SharePoint folder based on allowed file types (PDF, DOCX, PPTX,TXT). It filters out unsupported files (like images and Excel/CSV), handles validation, returns success if the file processing is successful or specific error messages. |

# **SECURITY CONSIDERATIONS IN DATA PROCESSING**

A diagram of a software development process

Description automatically generated

* ChatDSI ensures authentication by utilizing Azure Active Directory (now known as Microsoft Entra ID), strategically managing user access and permissions through meticulous verification of group memberships.
* ‘SG-DSI-AOAI\_ChatDSI’ serves as the principal group with authorization to access ChatDSI, employing nested groups to facilitate access for additional users by incorporating other groups under this parent group.