Example 5 — SharePoint document analysis

Analyze documents stored in a **SharePoint** document library with Ask Sage, by sending the file to the `/server/query_with_file` endpoint from a Power Automate flow.

> Prerequisites: the Prerequisites section of the Ask Sage Power Automate docs ·
> Auth & secure key storage: the Authentication section of the Ask Sage Power Automate docs.

This is the SharePoint counterpart to the OneDrive example — the flow is the same shape; only the file source changes. SharePoint actions take a **Site Address** (the `dataset` parameter) in addition to the file identifier.

You can get this flow two ways — **import** the ready-made solution (Quick start), or **build it step by step** (the full guide).

> ⚠️ **Not yet portal-verified.** The committed zip was **regenerated from `solution/src/`, not exported from a live environment.** Test-import it into a non-production environment before relying on it (see the Importing a solution section of the Ask Sage Power Automate docs). Screenshots for the build-by-hand steps are a follow-up — capture them during a verified portal build.

Quick start — import the solution

`solution/` holds an importable Power Platform solution (**SharePointIntegration**) that bundles the flow **and** the Ask Sage custom connector it uses. The flow: a manual trigger → *Get file content* + *Get file metadata* from SharePoint → **Query with File** (Ask Sage) → Compose the answer.

1. **Import** `solution/dist/SharePointIntegration_1_0_0_1.zip`.
   Import mechanics (Solutions → Import solution, roles) are in the Importing a solution section of the Ask Sage Power Automate docs.
2. **Create the two connections** when prompted (or under **… More** → **Connections**):
   - **SharePoint Online** — sign in with your Microsoft account.

    - the **Ask Sage custom connector** — paste your **Ask Sage API key** (it goes in the `x-access-tokens` header; see the Authentication section of the Ask Sage Power Automate docs).
3. **Point it at your file.** The flow ships with **placeholders**. Open the flow and, on **Get file content** and **Get file metadata**, set the **Site Address** to your SharePoint site and browse to the document you want to analyze (this replaces `https://YOURTENANT.sharepoint.com/sites/YOUR_SITE` and `REPLACE_WITH_YOUR_SHAREPOINT_FILE_ID`).
4. **Check the model.** The flow ships with `model` set to **`google-claude-46-sonnet`**. If that model isn't enabled in your Ask Sage tenant, open the **Generate completion with file** action and change it to one that is — list available models with `POST /server/get-models`.
5. **Test** → **Manually** → **Run**, enter a prompt (e.g. *"Summarize this document and list key risks"*), and read the result in the **Compose model analysis** step.

> **Why the connector is bundled:** exporting a flow that uses a custom connector fails unless the connector is part of the same solution (*"…requires the custom connector to be added to a dataverse solution"*). This solution includes it, so import is self-contained. If you rebuild from scratch, create the custom connector **inside** the solution (or import the OpenAPI from ../../connector/) — a connector made in the standalone *Custom connectors* area can't be added to a solution after the fact.

> **Reading the response:** `Query with File` returns the answer in the `message` field — `body('<action>')?['message']`. It begins with a `<file-content>…</file-content>` echo of the extracted text; the **Compose model analysis** step strips it with `split(outputs('Compose_full_response'), '</file-content>')[1]` to keep just the analysis.

Overview

This guide walks you through creating a Power Automate flow that analyzes a document stored in a SharePoint document library with Ask Sage. The flow can be triggered manually to process any file and return AI-generated insights. It mirrors the OneDrive example; the difference is that SharePoint actions need the **Site Address** of the library in addition to the file.

Prerequisites

- A **SharePoint Online** site with a document library containing files to analyze
- Power Automate account with appropriate permissions (premium connectors — see the Prerequisites section of the Ask Sage Power Automate docs)
- Ask Sage API key (obtain from your Ask Sage account)
- Basic familiarity with Power Automate flows

Use Cases

This integration is useful for:

- Analyzing proposals, contracts, and policy documents kept in a team site
- Summarizing reports stored in a shared library
- Answering questions about document content without opening each file
- Generating analysis that you can write back to a SharePoint list or column (see Customization)

Step-by-Step Instructions

Step 1: Create a Manual Trigger

1. **Open Power Automate** and create a new instant cloud flow.
2. **Search for and select** the trigger: "*Manually trigger a flow*".
3. **Name your flow** something descriptive like "SharePoint Document Analysis with Ask Sage".
4. (Optional) Add a **Text** input to the trigger so you can type the analysis prompt at run time.

Step 2: Get File Content from SharePoint

1. **Add a new action** and search for "*Get file content*" (**SharePoint** connector).
2. **Configure the action:**
   - **Site Address:** select (or paste) your SharePoint site, e.g. `https://YOURTENANT.sharepoint.com/sites/YOUR_SITE`.
   - **File Identifier:** click the folder icon and browse to the document you want to analyze.
3. **Advanced parameters:** set "Infer Content Type" to **Yes** so Power Automate sends the right content type for the file.

> **Site Address vs. File Identifier:** unlike OneDrive (which only needs the file), every SharePoint action first asks which **site** to act on. That site URL is the `dataset` parameter

> in the flow definition.

Step 3: Get File Metadata

1. **Add a new action** and search for "*Get file metadata*" (**SharePoint** connector).
2. **Configure the action:**
   - **Site Address:** the same site as Step 2.
   - **File Identifier:** the same file as Step 2.
3. This returns the file **Name** (used to name the upload) along with size, path, and dates.

Step 4: Set Up the Ask Sage API Call

You can do this two ways — with the **custom connector** (recommended; what the bundled solution uses) or with a raw **HTTP** action.

Option A — Ask Sage custom connector (recommended):

1. **Add an action** → your **Ask Sage** custom connector → **Generate completion with file** (`query_with_file`). If you don't have the connector yet, import the OpenAPI from `../../connector/` (see example 02).
2. **Configure:**
   - **message:** your prompt (the trigger **Text** input, or a static string like "Analyze this document and provide key insights").
   - **file (body): File content** from Step 2.
   - **file (fileName): Name** from Step 3.
   - **model:** e.g. `google-claude-46-sonnet` · **temperature:** `0` · **dataset:** `all`.

Option B — raw HTTP action:
Use a **POST** to `https://api.asksage.ai/server/query_with_file` with the `x-access-tokens` header and a `multipart/form-data` body that includes `message`, `model`, `tools` / `tools_to_execute` set to `["read_file"]`, and the `file` part — see example 04, Step 4 for the exact body. Don't paste your key into the action — see the Authentication section of the Ask Sage Power Automate docs.

Step 5: Compose the Full Response

1. **Add a Compose action** after the Ask Sage call.
2. **Inputs:** `@body('Generate_completion_with_file')['message']` — the model's answer.

Step 6: Compose Model Analysis (Optional)

1. **Add another Compose action** to strip the echoed file content: `@split(outputs('Compose_full_response'), '</file-content>')[1]`.
2. The output is just the model's analysis.

Step 7: Save and Test Your Flow

1. **Save** the flow.
2. **Test** → **Manually** → **Test** → **Run flow.** Enter a prompt if you added the Text input.
3. Watch each action turn green and read the **Compose model analysis** output.

Understanding the API Parameters

- **model** — e.g. `google-claude-46-sonnet`. List what's available in your tenant with `POST /server/get-models`.
- **temperature** — `0` for deterministic, focused analysis; higher for more creative output.
- **tools / tools_to_execute** — include `read_file` (raw-HTTP path) so the model reads the attached file. The custom connector handles this for you.
- **dataset** — Ask Sage dataset to ground against (`all` by default). Don't confuse this with the SharePoint `dataset` parameter (the **Site Address**) — same word, different APIs.

Customization Options

Trigger automatically on new files

Replace the manual trigger with the SharePoint trigger **"When a file is created (properties only)"** (or *When a file is created or modified*). Add a **Condition** to process only the file types you want (e.g. PDFs), then continue with Steps 2–6. The flow will analyze documents as they're added to the library.

Write the analysis back to SharePoint

Close the loop by storing the result in SharePoint:

- **Update file properties** — add an **Update file properties** action and write the analysis into a column on the document (e.g. an "AI Summary" text column).
- **Create a list item** — add a **Create item** action to log each analysis (file name + summary + date) to a tracking list.

Send the results elsewhere

Add **Send an email (V2)**, post to **Teams**, or start an **Approval** with the composed analysis.

Troubleshooting Tips

- **File not found / invalid identifier?** Re-pick the **File Identifier**, and confirm the **Site Address** matches the site that holds the file.
- **API authentication error (401)?** Verify your Ask Sage API key (connection, or the `x-access-tokens` header on the HTTP path).
- **Invalid / unavailable model?** The flow ships with `google-claude-46-sonnet`. If that model isn't enabled in your tenant, change the `model` value to one that is — list them with `POST /server/get-models`.
- **Empty response?** On the raw-HTTP path, ensure `read_file` is in both `tools` and `tools_to_execute`. Inspect the raw response body for a `message` field.
- **File too large?** Check Ask Sage's file size limits and consider splitting large documents.

Security Best Practices

- Store your API key securely — **Azure Key Vault** or a **Power Platform environment variable**, and turn on **Secure Inputs** on the action carrying the key. See the Authentication section of the Ask Sage Power Automate docs.
- Use **SharePoint permissions** to control which libraries and files the flow can read.
- Be mindful of the sensitivity of documents you send, and match the model accordingly. Power Automate runs in your Microsoft tenant; Ask Sage processes prompts inside its accredited boundary.

Conclusion

You now have a SharePoint integration with Ask Sage — analyze documents from a team site on demand, and (optionally) write the AI analysis straight back into SharePoint. For more, see the Ask Sage custom connector and the OneDrive example it's modeled on.