PROCUREMENT SPECTRUM
Insight-led. Practice-driven. Globally relevant.

Build Your First Procurement AI Agent

An Implementation Guide — step-by-step build instructions for a Contract Clause Reviewer in Claude Projects and Microsoft Copilot Studio

A Procurement Spectrum Research Report • April 2026

procurement-spectrum.com • consult@procurement-spectrum.com

Executive Summary

Ninety percent of Chief Procurement Officers surveyed by ProcureCon in January 2026 said they intend to deploy at least one AI agent within eighteen months. The gap most of them are stuck in is not a strategy gap. It is a build gap — how, specifically, would a procurement-and-legal team take a blank Claude Project or a blank Microsoft Copilot Studio canvas and turn it into something they will trust with live contracts?

This Implementation Guide closes that gap for one focused use case: a Contract Clause Reviewer. The use case is chosen deliberately. Contract review is a high-volume, high-latency, high-error-cost task that every procurement organisation performs. The agent is narrow enough to build in three to six weeks, valuable enough to protect its own budget, and safe enough to govern — every Red-flag case routes to a human approver before the contract returns to the business.

The guide provides three things end-to-end. First, the architecture, the playbook structure, and the complete copy-paste system prompt. Second, a screen-by-screen walkthrough for Claude Projects with every UI label, every required field, and a verification checkpoint after each step. Third, a screen-by-screen walkthrough for Microsoft Copilot Studio with the same structure — including licensing, knowledge sources, generative-orchestration setting, content moderation, channel publication, and admin governance.

A test set, release thresholds, a shadow-to-supervised rollout plan, and an EU AI Act governance checklist complete the package. Every step in this guide has been verified against current Anthropic and Microsoft documentation as of April 2026. The intent is that a procurement-and-legal team can begin the build the day this guide is read, on either platform, without further reference material.

Glossary — Terms used in this guide

LLM (Large Language Model): the underlying AI model — Claude, GPT, Gemini — that reads and writes natural language.

AI agent: an application built on an LLM that receives instructions, consults knowledge, optionally calls tools, and returns a structured output. An agent is more than a chatbot; it produces work product.

Claude Project: a scoped workspace inside Claude.ai that combines project instructions, uploaded project knowledge, and a chat interface. Available on Free, Pro, Max, Team, and Enterprise plans, with sharing and Team-level controls available on Team and Enterprise.

Project knowledge: the file area inside a Claude Project where source documents are uploaded. Subject to a 30 MB per-file limit; unlimited file count, but total content fits within the model context window with RAG fallback for larger sets.

Microsoft Copilot Studio: Microsoft's low-code agent builder, accessed at copilotstudio.microsoft.com. Builds agents using the Configure form-based authoring path or the Describe natural-language path.

Copilot Credits: the metered tenant-level currency that Copilot Studio agents consume at runtime. Replaced “messages” as the billing unit on 1 September 2025.

Knowledge source (Copilot Studio): a configurable data input — file upload, SharePoint, OneDrive, Dataverse, public website, or Microsoft Copilot connectors — that the agent can ground its responses in. Supports up to 1,000 files per agent for SharePoint and OneDrive uploads.

Generative orchestration: the Copilot Studio runtime mode in which the LLM dynamically selects topics, knowledge, tools, and sub-agents based on natural-language descriptions. The default for new agents in 2026. The alternative is “classic orchestration” (trigger-phrase routing).

System prompt / project instructions / Instructions field: the standing instruction set the agent receives on every run. Defines role, task, output format, guardrails, and refusal conditions. Called “project instructions” in Claude Projects and “Instructions” on the Configure tab in Copilot Studio.

Channel (Copilot Studio): the surface on which an agent is published — Microsoft Teams, Microsoft 365 Copilot, demo website, custom website, Direct Line, etc. The current UI consolidates Teams and M365 into a single tile, “Teams and Microsoft 365 Copilot”.

RAG (Retrieval-Augmented Generation): a technique where the agent retrieves relevant chunks from a knowledge base and uses them as context rather than relying on model memory.

Golden test set: a curated collection of real inputs with known correct outputs, used to evaluate whether the agent meets release thresholds before go-live.

Shadow / Assisted / Supervised: the three rollout stages in which an agent moves from running silently in parallel with humans, to providing humans a starting point, to operating with human-on-the-loop oversight on risk decisions.

Audit log: the complete record of inputs, prompts, model version, outputs, and human decisions for every agent run. Required under EU AI Act Article 12 for high-risk systems from 2 August 2026.

Hallucination: the failure mode where the agent generates content that reads plausibly but is not supported by the source document.

Citation: a reference from the agent's output back to the specific clause, section, and page in the source document that supports a claim.

DLP (Data Loss Prevention): tenant-level Power Platform policies that classify connectors as Business / Non-business / Blocked. The procurement-relevant control surface for Copilot Studio.

Content moderation: Copilot Studio's Azure-AI-Content-Safety-backed filter, set to Low / Medium / High in the Generative AI section. Recommended setting for procurement is “High”.

1. Why a Contract Clause Reviewer — and why start here

Contract review is the most common bottleneck between procurement and legal in almost every organisation. A mid-sized company signs hundreds to thousands of vendor contracts a year. Each one requires someone to read it, compare it against the organisation's negotiating position, flag the deviations, and route them for human decision. The work is repetitive, document-heavy, and pattern-based — which is exactly what current AI models are good at.

The use case meets four criteria that a first agent should meet. It is valuable: a well-built reviewer takes an hour of first-pass human reading down to three to five minutes, with the human still making every final call. It is measurable: citation accuracy, false-negative rate on Red flags, and throughput are all cleanly observable. It is governable: a Red-flag human-approval gate means the agent never unilaterally accepts risk. It is safe to iterate on: the worst failure mode is a missed flag that a human catches on the second pass — it does not execute transactions, move money, or sign contracts.

This is the first agent a procurement team should build. Not the most ambitious. The right one.

2. What the agent does — end-to-end architecture

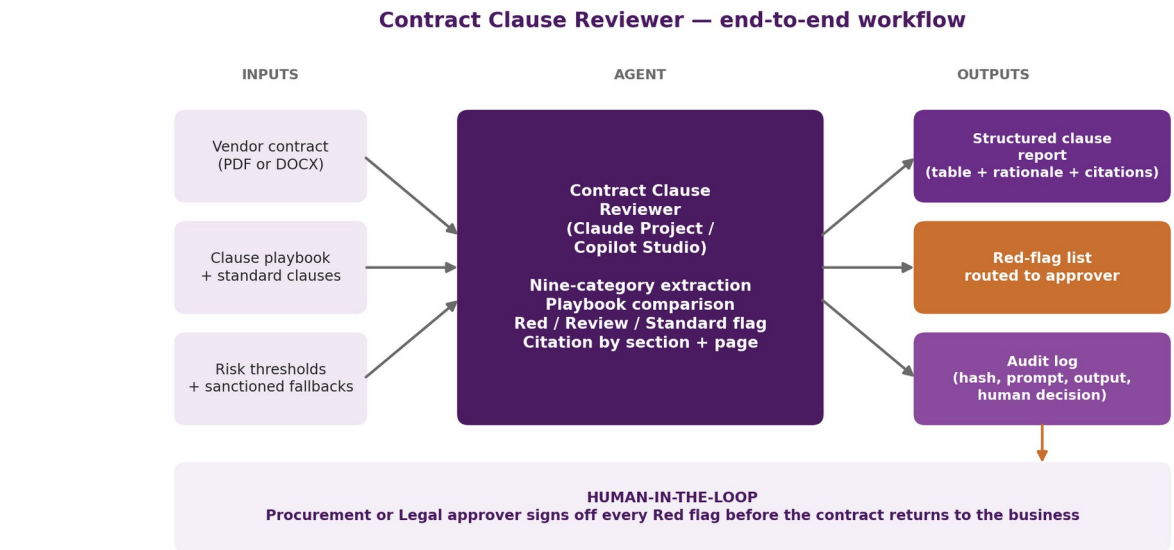

Figure 1 — Contract Clause Reviewer end-to-end workflow with human-in-the-loop approval on Red flags.

The agent takes three inputs: a vendor contract (PDF or DOCX); a clause playbook — the organisation's standard positions on each clause category; and risk thresholds — the pre-agreed points at which a deviation becomes a Red flag.

It performs three operations: it extracts the clauses from the contract into nine standard categories; it compares each clause against the playbook; and it assigns a flag — Standard, Review, or Red — and a citation back to the source document (section and page) for every conclusion.

It returns three outputs: a structured clause-by-clause report (a fixed-column table with rationale and evidence); a Red-flag list routed to a procurement or legal approver; and a complete audit log — input hash, prompt version, agent output, human decision — preserved for compliance.

Every Red-flag case passes through a human approver before the contract returns to the business. The agent does not approve anything. It accelerates the review; humans still decide.

3. The nine clause categories the agent extracts

A Contract Clause Reviewer should target the nine clause categories that cover roughly ninety-five percent of real contract risk in vendor agreements. Narrower scope is a feature, not a limitation — the agent performs better when the extraction target is explicit.

The nine are: payment terms; termination; limitation of liability; intellectual property; data protection and privacy; warranty; service level and service credits; renewal and auto-renewal; governing law and jurisdiction.

Each category has a handful of attributes the agent extracts. For payment terms: net days, currency, late-payment interest, prepayment, invoicing requirements. For termination: termination for convenience notice period, termination for cause cure period, termination fee if any. For limitation of liability: cap expressed as fees paid, cap for specific heads of liability (data, IP, confidentiality), unlimited categories. For data protection: controller-processor clarity, sub-processor flow-down, breach notification period, cross-border transfer mechanism. For warranty: warranty period, service-level warranty, exclusions. For service level: uptime commitment, credit schedule, measurement methodology. For renewal: auto-renewal present, notice period, capped vs. uncapped uplift. For governing law: jurisdiction, dispute resolution forum, arbitration rules if applicable. For intellectual property: ownership of deliverables, licence scope, indemnity.

The playbook specifies for each of these attributes three bands: the Standard band (accept), the Review band (escalate to a named approver), and the Red band (reject unless overridden).

4. The playbook — the single most underestimated artefact

The playbook is the document the agent reads to know what your organisation wants. It is not an AI artefact. It is a procurement artefact. Every organisation has one in some form — a negotiating

handbook, an approved-clause library, a redlining guide. The question is whether it is written explicitly enough for an LLM to use.

Three characteristics distinguish a playbook that works.

It states the positions in unambiguous language. “Net thirty days is standard” is usable. “Payment terms should be reasonable” is not.

It specifies the bands explicitly. Standard, Review, and Red are defined ranges, not judgements. “Net 30-45 days: Standard. Net 46-60 days: Review. Net 61+ days: Red.”

It names the approver for each Review and Red band. “Late-payment interest above 8% APR: Review with Finance Controller.” The agent writes the approver into the output, which removes routing ambiguity downstream.

A usable playbook for all nine categories fits in ten to fifteen pages. Investing in it is the single highest-leverage act in the build. Eighty percent of implementation issues observed in 2025 and 2026 deployments trace to playbook ambiguity, not to LLM behaviour.

5. The system prompt — complete, copy-paste-ready

This is the central artefact. The prompt below is tuned for Claude Projects — it uses the XML-tag structure Anthropic's engineering team recommends. The Microsoft Copilot Studio adaptation follows in Section B.

```
<role>
You are a Contract Clause Reviewer operating on behalf of the Procurement
and Legal functions. Your job is to screen a vendor contract against the
organisation's clause playbook and produce a structured, evidence-grounded
review that a human approver will use to accept, negotiate, or reject each
clause.

You do not approve any clause. You do not sign, commit, or make final
decisions. You prepare the review; humans decide.
</role>

<task>
For the contract provided in this conversation:

1. Extract the nine standard clause categories listed below.
2. For each category, identify the clause (or clauses) in the contract
   that corresponds to it.
3. Compare each clause to the playbook in project knowledge.
4. Assign one of three flags with the exact definitions below.
5. Return a structured report using the output schema defined below.

Nine clause categories:
- Payment terms
```

- Termination (convenience and cause)
- Limitation of liability
- Intellectual property
- Data protection and privacy
- Warranty
- Service level and service credits
- Renewal and auto-renewal
- Governing law and jurisdiction
</task>

<flag_definitions>
Standard: the clause matches the playbook's Standard band. No escalation required. Note briefly why.

Review: the clause falls within the playbook's Review band. Escalate to the approver named in the playbook. State the specific deviation from Standard and the playbook-sanctioned fallback position.

Red: the clause falls within the playbook's Red band OR is materially unfavourable to the organisation in a way not contemplated by the playbook. Route to the Red-flag escalation path.

Not present: the clause category is absent from the contract. State "Not present" — do not infer terms that are not written.
</flag_definitions>

<evidence_and_citation_rules>
Every flag must cite the specific source for the evidence, in the format: [Section X.Y, page Z]. Quote the exact contract language in the Evidence column — do not paraphrase. If the contract is silent on a sub-attribute, state "silent" rather than assuming a default.

If the playbook itself is silent on a clause, state "Playbook silent — use general industry-standard reference only" and flag as Review.

Do not invent terms, defaults, or positions that are not present in either the contract or the playbook.
</evidence_and_citation_rules>

<output_schema>
Return a table with the following columns, in this order:

| # | Category | Flag | Contract Clause (exact quote) | Citation | Playbook Position | Rationale | Approver |

Populate one row per clause category (nine rows). If a category has more than one relevant clause, add rows as needed. Follow the table with a summary section:

```
SUMMARY
- Red flags: [count] — list the row numbers
- Review flags: [count] — list the row numbers
- Standard: [count]
- Not present: [count] — list categories
- Overall assessment: [one paragraph, no more than four sentences]

Do not include any content outside this schema.
</output_schema>

<verification_pass>
After you have produced the table, perform an internal verification pass.
For every clause marked Red, re-locate the exact source language in the
contract and confirm the citation is accurate. If any citation cannot be
verified, change the flag to Review and note "Citation could not be
verified on second pass" in the Rationale column.
</verification_pass>

<refusal_conditions>
If the document provided is not a contract, respond: "This does not
appear to be a contract. Please upload a vendor contract in PDF or DOCX
format." Do not attempt a review.

If the playbook is not available in project knowledge, respond: "No
playbook is present. I cannot perform a review without the organisation's
clause playbook. Please upload it as a project file."

If the contract is in a language not covered by the playbook, respond:
"The contract appears to be in [language]. The playbook covers [language].
I cannot perform a reliable review across languages. Please escalate."

Do not produce speculative advice. Do not suggest negotiation tactics that
are not written in the playbook. Do not commit the organisation to a
position.
</refusal_conditions>

<tone>
Professional, concise, evidence-led. No filler. No hedging language
("it might be" / "you could consider"). State findings directly, with
the citation as evidence. If you are uncertain about a classification,
flag as Review and state the uncertainty in the Rationale column.
</tone>
```

The verification pass — in which the agent re-locates every Red-flag citation in the source document before returning — materially reduces hallucinated citations. It is the most load-bearing piece of the prompt. Do not remove it. The refusal conditions prevent the most common failure

modes: reviewing a non-contract, reviewing without a playbook, or reviewing across a language mismatch. The output schema is deliberately rigid — that is what makes the agent auditable and easy to consume downstream.

Section A — Build it in Claude Projects (step-by-step)

The Claude Projects path is the fastest route to a working Contract Clause Reviewer. A team with a usable playbook can have a configured Project running in approximately ninety minutes. The walkthrough below is verified against Anthropic's Help Center documentation current to April 2026.

A.0 Prerequisites

A Claude Team or Enterprise plan if multiple users will share the Project. Free, Pro, and Max plans support Projects but do not allow sharing — Free is also capped at five Projects total. Confirm the entitlement before starting; the procurement team usually needs Team minimum (5-seat minimum on Team).

A finalised playbook in PDF or DOCX, under 30 MB, covering all nine clause categories with Standard / Review / Red bands and named approvers.

A nominated Project owner — typically the Head of Procurement or a senior Category Manager — who will hold the project instructions, knowledge base, and member roster.

A choice of model. As of April 2026 the current Claude family is Opus 4.7, Sonnet 4.6, and Haiku 4.5. For contract review, Sonnet 4.6 is the right default — it balances accuracy on long-context document work with cost. Move to Opus 4.7 only if the test set surfaces accuracy gaps Sonnet cannot close.

A.1 Step 1 — Sign in and open Projects

Sign in to claude.ai. In the left-hand sidebar, click “Projects” — or navigate directly to claude.ai/projects.

Verify: the Projects page lists existing projects and a button labelled “+ New project” in the upper right.

A.2 Step 2 — Create a new Project

Click “+ New project”. A creation dialog opens.

In the “Name” field, enter: Contract Clause Reviewer.

In the “Description” field, enter a brief human-facing description, e.g.: “Reviews vendor contracts against the organisation's clause playbook. Returns a structured Standard/Review/Red flag report with citations. Human-approval required on all Red flags.” Note: the description is for human reference only — Claude does not read it.

If you are on a Team or Enterprise plan, set visibility. Choose “Keep it private” for the build phase, then switch to “Share with your broader organisation” once the agent has cleared the test set.

Click “Create project”.

Verify: you are taken into the empty project page. The right-hand panel shows the project knowledge base with a “+” button to add content and a “Set project instructions” option below it.

A.3 Step 3 — Set the project instructions (the system prompt)

In the right-hand project knowledge panel, click “Set project instructions”.

A text editor opens. Paste the entire system prompt from Section 5 of this guide — every XML tag, beginning with the role tag and ending with the tone tag. Do not edit, abbreviate, or remove any tag. The verification pass and refusal conditions are load-bearing.

Click “Save instructions”.

Verify: the project knowledge panel now shows the instructions stored, with an option to edit. The first line of the instructions (“You are a Contract Clause Reviewer…”) should be visible in the preview.

A.4 Step 4 — Upload the playbook

In the project knowledge base panel, click the “+” button to add content.

Upload your finalised playbook as a single PDF or DOCX file. The file must be under 30 MB. If your playbook is larger, split it into multiple files by clause category — Claude can read across them.

While the playbook is uploading, name the file clearly using a versioned naming convention. Version control matters under EU AI Act Article 12 record-keeping obligations.

Verify: the file appears in the project knowledge panel with an icon and the filename. Click into the file to confirm the page count and that text is selectable (a scanned PDF without OCR will not be readable by Claude).

Optional but recommended: upload a gold-standard example contract — one fully reviewed by a senior procurement leader, with the expected output written out — as a second knowledge file. This grounds the agent in your organisation's review style and reduces variance on the first ten production runs.

A.5 Step 5 — Choose the model

Open a new chat inside the Project. Above the message box, next to the send button, the model selector displays the currently selected model.

Click the model selector and choose “Claude Sonnet 4.6”. If your test set shows accuracy gaps after Step 7, return here and switch to “Claude Opus 4.7” by clicking “More models”.

Verify: the chat header now displays “Claude Sonnet 4.6” (or your chosen model). Note: model selection is per-chat in Claude.ai, not per-Project — every new chat in this Project will need to be set to the same model. Document this in your Standard Operating Procedure.

A.6 Step 6 — Run a smoke test

In the chat, attach a single, known-clean vendor contract (a contract you have already reviewed manually, with no Red flags) using the paperclip icon in the message composer. Send the message: “Please review this contract using the project instructions and playbook.”

Wait for the response.

Verify: the agent returns the eight-column table with one row per clause category (or more if multiple clauses per category). Every flag has a citation in the format [Section X.Y, page Z]. The summary block at the end gives counts of Red, Review, Standard, and Not present. No content appears outside the schema.

If the table is malformed, return to Step 3, re-paste the prompt verbatim, and confirm the output_schema tag was preserved.

A.7 Step 7 — Run the golden test set

Run all twenty-five cases from the golden test set (Section 6 of this guide) through the Project, one chat per case. Record citation accuracy, Red-flag false-negative rate, false-positive rate, and throughput in a spreadsheet.

Verify: the agent meets all five release gates listed in Section 7. If it does not, the failure mode is almost always one of three: ambiguous playbook language (return to the playbook), missing clause category coverage in the playbook (add it), or an over-large or scanned contract (move scanning out of the agent's path with an OCR pre-step).

A.8 Step 8 — Share with the team

Once the agent clears the release gates, return to the Project page. Click the “Share project” button to the right of the project name.

Add procurement and legal members by name or email. Set role per member: “Can use” (read project instructions and chat — recommended for most reviewers) or “Can edit” (modify instructions, knowledge, members — restrict to the Project owner and a backup).

Click “Share” to send invitations.

Verify: invited members receive an email and see the Project in their Projects list when they sign in. Test by having one nominated reviewer process a contract end-to-end and confirm they can attach files, receive the agent's output, and route the Red flags through your standard escalation channel.

A.9 Step 9 — Privacy, retention, and audit configuration

Confirm with your IT and Legal team:

Training data status. On Team and Enterprise plans, inputs and outputs are not used to train Anthropic models by default — this is the operative assurance for procurement deployments. On Pro and Max, training is opt-in/opt-out via privacy settings at claude.ai/settings/data-privacy-controls.

Zero Data Retention (ZDR). Available contractually on Enterprise. Required if the organisation has classified vendor contracts as "must not leave the tenant".

SOC 2 Type II and HIPAA-eligible configurations. Anthropic publishes these under its Trust Center. Confirm with your security team that the configurations meet your control framework.

Audit log. Claude Projects do not provide a granular audit log of every chat by default — this is a known gap. The standard mitigation is to require reviewers to export the chat as PDF on Red-flag cases and store it in your contract management system, and to run periodic exports of project chat history via the Anthropic API for Enterprise tenants. Document this in your runbook.

Verify: you have a written one-page configuration document covering plan tier, training opt-out status, retention setting, and the chat-export procedure. This document goes into the EU AI Act risk file (Section 9).

Section B — Build it in Microsoft Copilot Studio (step-by-step)

The Copilot Studio path takes longer than Claude Projects (typically two to three weeks for a configured agent versus ninety minutes for a Claude Project), but it integrates cleanly with Microsoft 365 — the playbook lives in SharePoint, the Red-flag routing uses Teams Approvals, audit logging runs through Microsoft Purview, and end users access the agent inside Teams or Microsoft 365 Copilot. For Microsoft-standardised organisations, it is usually the right choice. The walkthrough below is verified against Microsoft Learn documentation current to April 2026.

B.0 Prerequisites

Two layers of licensing. Per-user "Copilot Studio User License" for each maker, and tenant-level capacity via either a "Copilot Studio license subscription" (1 pack = 25,000 Copilot Credits per month, no roll-over) or a "Copilot Studio pay-as-you-go meter" on an Azure subscription. A Microsoft 365 Copilot user licence in the tenant is not strictly required, but it raises the SharePoint per-file limit from 7 MB to 200 MB and unlocks tenant graph grounding with semantic search.

A dedicated environment. Do not build the agent in the default Power Platform environment. Create or request a dedicated "Procurement-Dev" environment in the Power Platform Admin Center, then a "Procurement-Prod" environment for go-live. This separation matters for DLP, audit, and rollback.

Security roles. The Project owner needs "Environment Maker" in the Dev environment. Reviewers will be assigned "Editors" (full edit/share/use) or "Viewers" (use only) once the agent ships.

The playbook, finalised, in PDF or DOCX. If hosted on SharePoint, the file size limit depends on tenant configuration — under 200 MB if M365 Copilot is licensed in the tenant; under 7 MB otherwise.

B.1 Step 1 — Sign in to Copilot Studio

Open https://copilotstudio.microsoft.com in a browser. Sign in with your Microsoft 365 work account.

In the top-right environment selector, switch to your Procurement-Dev environment. Confirm the environment name shown next to your account avatar — this is the most common cause of misconfiguration.

Verify: the Home page shows the Procurement-Dev environment name and a section labelled “Start building from scratch” with a button “Create an agent”.

B.2 Step 2 — Create a blank agent

On the Home page, under “Start building from scratch”, click “Create an agent”. (Equivalently, navigate to the “Agents” tab in the left rail, then click “Create blank agent”.)

The agent creation page opens with two tabs at the top: “Describe” (conversational, AI-generated) and “Configure” (form-based).

Click the “Configure” tab. The form-based path is the recommended route for a governed procurement build — it gives full control over every field and avoids the AI-generated description introducing language you have not approved.

B.3 Step 3 — Set Name, Description, and Instructions

In the “Configure” tab:

“Name”: Contract Clause Reviewer.

“Description”: Reviews vendor contracts against the procurement clause playbook. Returns a Standard / Review / Red flag report with citations. Human-approval required on all Red flags. Do not negotiate; do not commit. Up to 1,024 characters. The description influences how the agent is described to end users in Microsoft 365 Copilot — keep it precise.

“Instructions”: paste the entire system prompt from Section 5, with two adaptations.

Adaptation 1. Convert the XML tags into headed plain-text sections. Copilot Studio's Instructions field accepts natural-language instructions of up to several thousand characters; it does not parse XML tags as semantic markers the way Claude does. Replace the role tag with “## ROLE”, the closing role tag with a blank line, and so on through every tag.

Adaptation 2. In the OUTPUT_SCHEMA section, add this line at the end: “Return the table in Markdown format using pipes for column separators. Do not return JSON unless explicitly requested.” Copilot Studio's default output is Markdown; the explicit instruction prevents drift to JSON for users on the Teams channel.

Verify: the Instructions field shows your full prompt with section headers. Click outside the field to save (Copilot Studio autosaves).

B.4 Step 4 — Add the playbook as a Knowledge source

Scroll to the “Knowledge” section on the Configure tab. Click “Add” then “Add to agent”. The “Add knowledge” dialog opens with a list of source types: “Public website”, “SharePoint”, “Documents” (file upload), “OneDrive”, “Dataverse”, “Microsoft Copilot connectors”, and “Enterprise data”.

The recommended pattern for a procurement playbook is “SharePoint”. Paste the SharePoint URL of the procurement playbook document library or the specific playbook file. Click “Add”. The agent will index the content (this can take from a few minutes to an hour depending on the document set).

If your playbook is not yet on SharePoint, choose “Documents” and upload the PDF or DOCX directly. Per-agent limit is 1,000 files for SharePoint and OneDrive; per-source limit is 1,000 files, 50 folders, 10 layers of subfolders. Files labelled “Confidential” or “Highly Confidential” cannot be indexed — declassify the playbook before adding it, or store it in a SharePoint library without these labels applied.

Verify: the knowledge source appears in the “Knowledge” list with an “Indexed” or “Ready” status. Click into the source to confirm the file count and the index timestamp.

B.5 Step 5 — Confirm Generative Orchestration is ON

Scroll to the “Generative AI” section. Locate the setting “Use generative AI orchestration for your agent's responses?” and confirm it is set to “Yes”.

This is the default for new agents in 2026, but tenant administrators can disable it at the environment level. If it is set to “No” (classic orchestration), the agent will only fire pre-defined topics on trigger phrases — which is the wrong runtime mode for a knowledge-grounded contract reviewer.

If the setting is locked to “No” at the environment level, escalate to the Power Platform admin to enable generative orchestration in the Procurement-Dev environment before continuing.

Verify: the radio is set to “Yes” and the agent description on the Overview page now refers to “agent chooses” trigger behaviour rather than fixed trigger phrases.

B.6 Step 6 — Set Content Moderation to High

Inside the “Generative AI” section, locate the “Content moderation” slider. Move it to “High”.

The slider has three positions: “Low”, “Medium” (default), “High”. High applies the strictest filter from Azure AI Content Safety against the categories of hate, violence, sexual content, self-harm, and against jailbreak / prompt-injection / prompt-exfiltration / copyright detection. Procurement contracts are read-only, internal documents — High is the correct setting and the small reduction in answer latitude does not affect the use case.

Verify: the slider visibly shows “High” and the description below the slider updates to “the highest level applies a stricter filter”.

B.7 Step 7 — Test in the Test pane

In the top-right of the authoring canvas, click “Test”. The “Test your agent” pane opens to the right of the canvas.

Attach a known-clean vendor contract (the same smoke-test contract used in Step A.6). Send: “Please review this contract using the agent instructions and the procurement playbook.”

The Test pane displays the agent's response in real time. With generative orchestration on, an “activity map” appears alongside the conversation — it shows which knowledge source was consulted, which topics fired, and which tools (if any) the agent invoked. Use this to verify the agent is grounding its answers in the playbook rather than the model's general knowledge.

Verify: the agent returns the Markdown table with one row per clause category, every flag carries a citation in the format [Section X.Y, page Z], and the activity map shows the playbook knowledge source was retrieved.

If the agent returns generic answers without grounding in the playbook, the indexing has not completed — return to Step 4 and confirm the Knowledge source shows “Indexed”.

B.8 Step 8 — Run the golden test set

Run all twenty-five cases from the golden test set (Section 6) in the Test pane. Record citation accuracy, Red-flag false-negative rate, false-positive rate, and throughput.

Verify: the agent meets all five release gates in Section 7. Common Copilot Studio-specific failure modes: knowledge files marked confidential (not indexed), instructions field truncated by the character limit (split into the topic-level instructions for an overflow), or generative orchestration silently disabled at environment level (re-check Step 5).

B.9 Step 9 — Publish

In the top menu bar of the authoring canvas, click “Publish”. The “Publish this agent” dialog opens with a confirmation prompt.

Click “Publish” in the dialog. Wait for confirmation that the agent has been published. The first publish takes one to three minutes.

Verify: the “Publish” button now shows a timestamp of the most recent publication. The agent now exists in the Procurement-Dev environment as a versioned, deployable artefact.

B.10 Step 10 — Add the “Teams and Microsoft 365 Copilot” channel

In the top menu bar, click “Channels”. Locate the tile labelled “Teams and Microsoft 365 Copilot” (the 2026 UI consolidates these into a single tile; older Microsoft Learn screenshots may show separate tiles).

Click the tile. Under “Turn on Microsoft 365”, confirm “Make agent available in Microsoft 365 Copilot” is selected (this is the default — clearing it limits the agent to Teams only).

Click “Add channel”. Wait for the configuration to complete.

Verify: the channel tile now displays a green “Configured” indicator and a “See agent in Teams” link.

B.11 Step 11 — Make the agent available to end users

Click “See agent in Teams” to install the agent into your own Teams and Microsoft 365 Copilot client. Test the end-to-end flow as an end user — attach a contract in Teams, send the same review request, confirm the response renders correctly.

To distribute organisation-wide, choose between two paths:

Self-service path. Under “Built with Power Platform” in the Channels configuration, choose “Show the agent in the Teams app store”. Procurement and legal team members can then install the agent themselves from the Teams app catalogue.

Admin-deployed path. Submit the agent for admin approval. Once approved, it appears under “Built for your org” in the Teams app catalogue. Admin approval is granted in the Microsoft Teams admin center under “Manage apps”, or in the Microsoft 365 admin center. This is the recommended path for a governed procurement deployment — it gives admins the opportunity to attach a DLP policy, restrict the user audience, and add the agent to the published procurement runbook before end users see it.

Verify: a nominated end user — not the builder — can find and install the agent in Teams, attach a contract, and receive the structured review response.

B.12 Step 12 — Configure DLP, security roles, and admin governance

Three administrative tasks complete the build.

DLP (Data Loss Prevention). In the Power Platform admin center, navigate to “Data policies”. Create or amend a policy that classifies the connectors the agent uses (SharePoint, Microsoft 365 Copilot, Teams) as “Business”. Block any connector classified as “Non-business” that the agent could otherwise call (e.g. a personal-OneDrive connector). DLP policies enforce in real time and surface as maker-visible error messages — the policy must be published, not draft.

Authentication. On the Configure tab, set “End-user authentication” to “Authenticate with Microsoft Entra ID”. This restricts the agent to authenticated tenant users and writes the user identity into every audit log entry — required under EU AI Act Article 12.

Promote to Procurement-Prod. Once the Dev environment build clears the release gates, export the agent as a Power Platform solution and import it into the Procurement-Prod environment. This is the change-control gate — Prod runs the agent for end users; Dev is for iteration. Re-test the publish-and-channels steps in Prod before announcing the agent to the organisation.

Verify: the DLP policy is published and visible to your tenant admin, the agent's authentication is set to Entra ID, and a working version exists in Procurement-Prod.

6. The golden test set — twenty-five cases before go-live

A golden test set is the set of real inputs whose correct output is already known. It is what you measure the agent against before putting it in front of the business.

Build twenty-five cases minimum before go-live. Ten standard contracts — a mix of SaaS, professional services, hardware, and consulting — that the agent should process cleanly with few or no flags. Ten deviation contracts with specific known Red flags planted in specific clauses. Five adversarial contracts: a scanned PDF with OCR artefacts, a contract in a mixed-language format, a very short contract missing several standard clauses, a very long contract (fifty pages or more), and a contract that has been edited across several redline rounds.

For each case, the expected output is documented in advance: how many Red flags, which rows, what the human reviewer would have flagged. The agent is then run on each, and the output compared.

The measurement is mechanical. Citation accuracy: of the flags the agent raises, what percentage of citations can be verified back to the source. Red-flag false-negatives: of the known Red flags in the test set, what percentage did the agent miss. Red-flag false-positives: of the flags the agent raised, what percentage were not material risks on human review. Throughput: time from contract upload to returned report.

7. Release gates — thresholds that constitute “ready”

Citation accuracy ≥ 95 percent. A flag the agent cannot support with a verifiable citation is worse than no flag.

Red-flag false-negative rate ≤ 2 percent. Missing a true Red flag is the most expensive failure mode. The human-approval gate mitigates it but does not eliminate it.

Red-flag false-positive rate ≤ 5 percent. High false-positive rates swamp the human approvers and destroy the productivity case.

Throughput ≤ 5 minutes per contract. On a standard vendor agreement up to thirty pages.

Audit log completeness: 100 percent. Every run, every input, every output, every human decision is preserved.

An agent that does not clear these thresholds is not ready. Continue building the playbook and the test set rather than putting an underperforming agent in front of the business.

8. Rollout — shadow, assisted, supervised

Shadow mode (weeks 1–4). The agent processes every contract in parallel with the existing human review. Its outputs are logged but not used. Comparisons between agent and human are collected daily. This stage surfaces the failure modes and allows the playbook and the prompt to be tuned without operational risk.

Assisted mode (weeks 5–8). The agent's output is provided to the human reviewer as a starting point. The human is instructed to read the agent's output first, then confirm or override. Throughput gains begin to appear in this stage. Agent and human decisions are compared and divergences are logged.

Supervised automation (week 9 onwards). Standard-flag clauses are automatically accepted without human review, with a five-to-ten-percent random sampling for ongoing quality control. Review-flag and Red-flag clauses continue to go to human approvers. This is the steady-state. The agent is producing value; humans retain all risk decisions.

At no stage does the agent move to fully autonomous review. The governance gate on Red flags remains in place. This is what makes the agent safe to run under EU AI Act high-risk obligations, which apply from 2 August 2026 to AI systems used in procurement and contracting.

9. Governance, audit, and the EU AI Act

From 2 August 2026, AI systems used in procurement decisions that materially affect vendor selection or contract terms are classified as “high-risk” under the EU AI Act. Four obligations matter most.

Risk management (Article 9): a documented risk assessment of the agent's failure modes and the mitigations in place. The Section 7 release gates and Section 8 rollout stages constitute the operative mitigations.

Data governance (Article 10): documentation of training data for any custom model. Not applicable for vendor-hosted Claude or GPT, but applicable to the playbook and any retrieval-augmented examples uploaded as project knowledge.

Logging (Article 12): a complete audit log of inputs, outputs, model version, and human decisions. On Claude Projects this is currently a manual export; on Copilot Studio it integrates with Microsoft Purview audit pipelines.

Human oversight (Article 14): the human-approval gate on Red flags. Build it in from the first release. Do not retrofit.

The Contract Clause Reviewer as described here satisfies all four when the audit log is complete and the Red-flag routing is enforced.

10. What not to build into this agent — on purpose

Four capabilities are deliberately excluded.

Do not let the agent negotiate or generate counter-language for Red flags. The human approver has the organisational context and the authority.

Do not let the agent approve Standard clauses without human sight in the first ninety days. Even after the assisted-to-supervised transition, sample five-to-ten percent for ongoing quality control.

Do not let the agent compare the contract against prior redline rounds. Version comparison is a separate, harder use case with different failure modes. Scope it as a second agent if needed.

Do not let the agent infer missing clauses. “Not present” means not present. Inference is where hallucinations enter.

11. Capability that compounds

Building this agent takes three to six weeks for a team that has a usable playbook and an enterprise Claude or Copilot Studio entitlement. It is a low-risk, high-value first build. More important than the immediate productivity, it is the pattern from which every subsequent procurement agent will be built — supplier risk screener, RFP response evaluator, invoice-to-PO reconciler, contract renewal advisor. Each reuses the same architecture: narrow scope, explicit playbook, evidence-grounded output, human approval on risk decisions, full audit log.

Procurement teams that build this agent in 2026 are not just deploying a tool. They are building an internal capability worth compounding.

Conclusion

The distance between a vendor demo and a procurement AI agent running under governance is much shorter than most teams assume, if the scope is disciplined and the playbook is explicit. The Contract Clause Reviewer is the right place to start: the work is high-volume, the failure modes are known, the governance pattern is clean, and the value is visible within sixty days of go-live.

The system prompt, the playbook structure, the test set, the release gates, the screen-by-screen build for both Claude Projects and Microsoft Copilot Studio, and the rollout plan in this guide are sufficient to begin the build the day this guide is read. The next step, after that, is to build the second agent from the same template.

Sources & references

This guide was verified against the following authoritative sources current to April 2026.

Anthropic / Claude Projects. "How can I create and manage projects?" support.claude.com/en/articles/9519177. "Upload files to Claude" support.claude.com/en/articles/8241126. "How large is the context window on paid Claude plans?" support.claude.com/en/articles/8606394. "How can I change the model version that I'm chatting with?" support.claude.com/en/articles/8664678. "Claude release notes" support.claude.com/en/articles/12138966. "Is my data used for model training?" privacy.claude.com/en/articles/10023580. Claude pricing — claude.com/pricing.

Microsoft Copilot Studio. "Quickstart: Create and deploy an agent" learn.microsoft.com/microsoft-copilot-studio/fundamentals-get-started. "Create and delete agents" learn.microsoft.com/microsoft-copilot-studio/authoring-first-bot. "Microsoft Copilot Studio Licensing Guide — April 2026". "Knowledge sources summary" learn.microsoft.com/microsoft-copilot-studio/knowledge-copilot-studio. "Add SharePoint as a knowledge source" learn.microsoft.com/microsoft-copilot-studio/knowledge-add-sharepoint. "Write agent instructions" learn.microsoft.com/microsoft-copilot-studio/authoring-instructions. "Connect and configure an agent for Teams and Microsoft 365 Copilot" learn.microsoft.com/microsoft-copilot-studio/publication-add-bot-to-microsoft-teams. "Orchestrate agent behavior with generative AI" learn.microsoft.com/microsoft-copilot-studio/advanced-generative-actions. "Configure data policies for agents" learn.microsoft.com/microsoft-copilot-studio/admin-data-loss-prevention. "Resolve responsible AI content filter errors" learn.microsoft.com/microsoft-copilot-studio/troubleshoot-agent-response-filtered-by-responsible-ai. "Security and governance" learn.microsoft.com/microsoft-copilot-studio/security-and-governance.

Other. Anthropic, "Building Effective Agents" (December 2024). ProcureCon CPO survey, January 2026. EU AI Act (Regulation 2024/1689), high-risk obligations effective 2 August 2026.

For advisory support on scoping, building, or governing procurement AI agents — including playbook development, Claude Projects implementation, Copilot Studio agent build, or EU AI Act readiness — contact consult@procurement-spectrum.com | procurement-spectrum.com

Every step in this Implementation Guide has been verified against current Anthropic and Microsoft documentation as of April 2026.