From Proof-of-Concept to Production

From Chat to Repeatable Workflow



The Implementation Ladder

From Simple Queries to Intelligent Automation

Level 1: Basic Chat

Ask individual questions and receive one-off answers

Level 2: Template Prompts

Use structured, reusable prompts for common tasks

Level 3: Verified Workflows

Add quality checks and standardised output formats

Level 4: Persistent Context

Store information in projects and workspaces for continuity

Level 5: Automation & Agents

Agents in n8n or Power Automate, or built into Procore and Microsoft 365

Don't jump to agents if you haven't mastered stable workflows first. Agents amplify **good process** and equally amplify **bad process**. Build your foundation systematically.

The SOP Prompt Structure

Your Copy-Paste Template for Repeatable Results

01

Define the Role

"You are a commercial engineer assistant supporting tender reviews and risk assessment..."

02

Specify Inputs

"Use only: [Document A], [Document B], [Reference sheet C]. Do not use external knowledge."

03

Detail the Steps

"1. Extract key requirements. 2. Identify risks. 3. Map to existing capabilities..."

04

Build in Checks

"Validate against source documents. Flag missing information. Quote sources for all claims."

05

Standardise Output

"Return as: executive summary, risk table with RAG status, recommended actions with owners."

The secret is in the checks. Most people prompt with "summarise this" and wonder why output varies wildly. The Checks section is how you reduce errors, ensure consistency, and make workflows auditable for governance requirements.

Template You Can Reuse

The Skeleton for Reports, Tenders, Programme Updates

```
INPUTS:
Use ONLY the attached documents. If missing info, list it.

STEPS:
Extract X, then compare Y, then write Z.

CHECKS:
Flag contradictions, missing dates, numbers that don't add up.

OUTPUT:
Return a one-page brief: Executive summary, risks, actions, open questions.
```

This is your copy-paste starting point. Adapt the specifics to your use case, but keep the structure intact. The four-part pattern works because it mirrors how engineers already think: what you have, what you do, what you verify, what you deliver.

Beating Token Limits

Three Patterns That Actually Work in Production



1. Chunk & Stitch

Break documents into logical sections, summarise each independently, then combine summaries into a coherent whole. This mirrors how engineers naturally process large specifications.



2. Ask Questions First

Require the AI to ask five clarifying questions before drafting any output. This forces structured thinking and reveals gaps in requirements early in the process.



3. Map Then Reduce

Extract structured data first (dates, numbers, requirements, risks), then write the narrative summary. Structure before synthesis prevents hallucination and improves accuracy.

Token limits aren't blockers when you adapt your methodology. Engineers already work this way mentally: you don't read 300 pages and magically produce answers—you extract what matters, then make decisions based on that distilled information.

The 'Ask-Questions-First' Rule

Instantly Improve Accuracy and Prevent Confident Nonsense

Before answering, ask up to 5 clarifying questions. If you can proceed, state assumptions explicitly.

This single instruction prevents the AI from rushing to output based on incomplete information. By forcing it to ask clarifying questions first, you catch gaps in requirements, surface hidden assumptions, and ensure the final deliverable actually addresses what you need—not what the AI guessed you meant.

Why it matters: This prevents confident nonsense and makes you look professional when you share outputs internally. It's the difference between 'here's my best guess' and 'here's what I need to know before I can give you a quality answer.'

Verification Checklist

Quality Control

Evidence

Show direct quotes and page references for all key claims. Every assertion must be traceable to source material.

Assumptions

List all assumptions explicitly. Don't leave interpretation implicit—name what you've inferred and why.

Numbers

Cross-check totals, units, dates, and calculations. Numerical errors are the fastest way to lose stakeholder trust.

Risks

Flag uncertainties and specify what requires human verification. Identify information gaps before they become problems.

Human Sign-Off

Assign a named reviewer with clear timeframes. Accountability remains with people, not systems.

This approach keeps AI safe and useful without resorting to blanket bans. It also satisfies governance expectations whilst maintaining practical velocity—meeting compliance requirements without drowning in bureaucracy.

Exercise: Build Your Workflow Prompt

Create Something Reusable (10 minutes)

01

Work independently

02

Complete all five fields: Role, Inputs, Steps, Checks, Output

03

Test your prompt in your AI tool immediately

04

Prepare to share one effective check you incorporated

Don't overthink this. The goal is leaving today with one workflow prompt you can reuse tomorrow. Perfection isn't required—practical utility is. Next step: persist this in Projects or workspaces for team-wide access and continuous improvement.

Prompt A: Ask Questions First

Reduce Garbage Output by Forcing Structured Thinking

```
You are my engineering documentation assistant.
Before you write anything, ask me 5 clarifying questions that would materially change the output quality.
After I answer, produce the deliverable using this structure:
• Summary (max 8 bullets)
• Risks & unknowns
• Actions (owner / due date)

Rules: use only provided content; don't invent; list assumptions.
```

This pattern prevents the AI from rushing to output. By requiring questions first, you force it to identify gaps, clarify scope, and understand context before drafting. The result: fewer revisions, better alignment with actual requirements, and outputs that reflect real project constraints rather than generic assumptions.

Use this when: requirements are vague, stakeholders haven't aligned, or you're starting a new workstream where assumptions could derail the outcome.

Prompt B: Progress Report → Executive Summary

Turn Weekly Updates into Consistent Leadership Briefings

```
Role: You are an assistant to a project manager in civil/commercial engineering.
Inputs: I will paste weekly progress notes and issues. Use only that text.
Steps:
1. Extract: milestones achieved, blockers, key decisions, upcoming activities
2. Identify: schedule risk (drivers), cost risk (drivers), H&S items
3. Draft: one-page exec update

Checks:
• If any dates/quantities are missing, flag as "NEEDS CONFIRMATION"
• Separate facts vs assumptions

Output:
Heading + bullets + a table (Issue | Impact | Owner | Next step)
```

This workflow transforms scattered progress notes into standardised executive updates. The structured steps ensure nothing critical is missed, whilst the checks section prevents hallucination and maintains accuracy. The consistent output format means leadership receives the same structure every week—building trust and reducing review time.

Measure it: before vs after, with review time. **Consistency gained:** every update follows the same format, making trends and issues easier to track over time.

Prompt C: Tender Section → Risk & Clarifications Table

Systematic Tender Review for Commercial Teams

```
You are a commercial assistant helping review tender requirements.
Extract only what's explicitly stated.
Output a table: Requirement | Risk/ambiguity | Clarification question | Commercial impact
(low/med/high)
```

This prompt turns tender review from an art into a repeatable process. By constraining the AI to extract only explicit statements, you prevent it from making dangerous assumptions about scope or obligations. The structured table output ensures every requirement is assessed for risk, every ambiguity generates a clarification question, and commercial impact is consistently evaluated.

Perfect for: tender reviews, contract analysis, scope validation. The table format makes it easy to assign follow-up actions and track which clarifications have been resolved before submission deadlines.

Prompt D: Site Minutes → Notices & Early Warnings

Structured Review for Delivery & Commercial Teams

```
Role: You are a contracts assistant to an M&E contracts manager.
Inputs: site meeting minutes, RFI / technical query log, change register. Use only these.
Steps:
1. Extract every instruction, delay, design change and late information
2. Check each event against the change register
3. Draft a dated chronology for anything not yet notified
Checks:
• Flag instructions with no written confirmation
• Flag events that may be close to a contract notice deadline
• Flag duplicated or overlapping register items
• Quote the minute reference for every event
Output:
Table: Date | Event | Source | In register? | Notice needed? + a short chronology
```

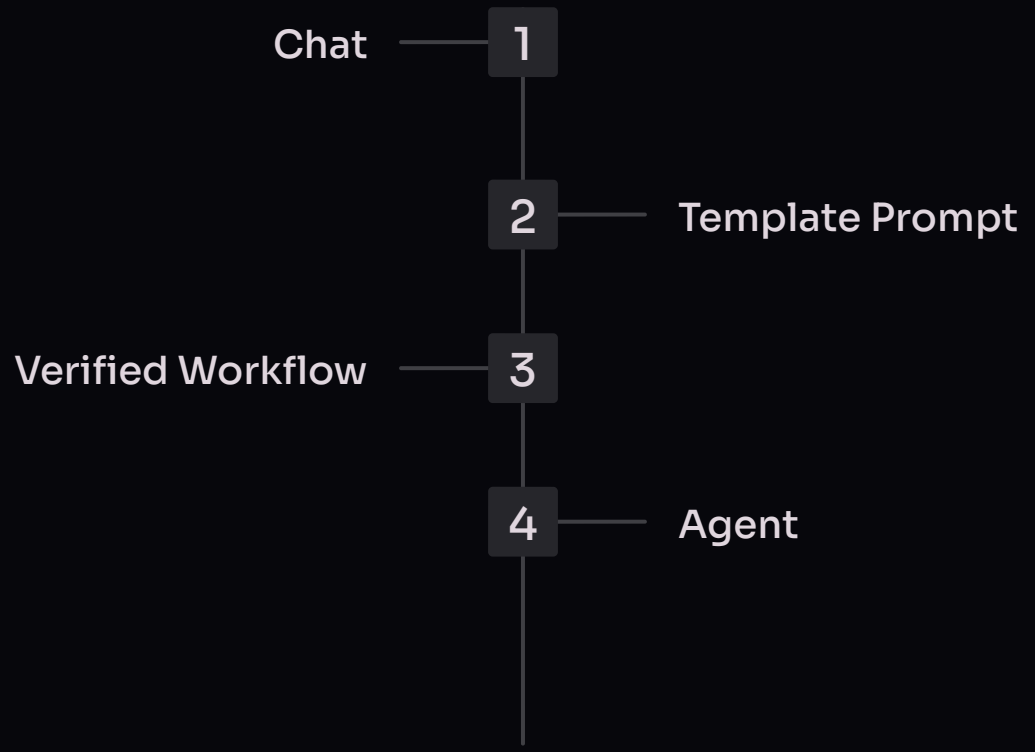
This prompt brings structure to one of the most expensive gaps on a project: events that are discussed in meetings but never formally notified. By forcing every event to be checked against the register and quoted from source, it surfaces late information, verbal instructions and potential delays while there is still time to act.

Perfect for: monthly contract reviews, early-warning checks, preparing payment applications and claims chronologies. The output is a draft for the contracts manager — the decision to notify stays with a person.

From Workflow to Agent Upgrade

How a verified workflow becomes something reusable, persistent, and semi-automated

- A prompt gives you an answer
- A workflow gives you a repeatable output
- An agent gives you repeatable output at the right time, with the right context, using the right tools, under the right controls



Don't jump to agents before the workflow is stable.

The Upgrade Layers

What gets added when you upgrade a workflow into an agent

What a workflow already has

- Role
- Inputs
- Steps
- Checks
- Output

What an agent adds

- Trigger
- Memory / persistent context
- Tool access
- Routing / handoffs
- Approval boundary
- Logging / KPI tracking

Agents are not magic. They are disciplined workflows with context, tools, and control.

Example: Weekly Reporting Agent

Programme Update Workflow → Agent

Current Workflow

- PM pastes programme extract + progress notes
- AI extracts changes and drafts narrative
- Flags risks and creates action list
- Human reviews output

Upgraded Agent

- Trigger: every Thursday at 4pm
- Inputs: programme extract, progress dashboard, issues log
- Memory: previous 3 weekly reports + project context
- Tool use: document folder, spreadsheet, reporting template
- Output: executive summary, risks, actions
- Controls: quotes/sources, needs confirmation flags, human sign-off before sending

The logic stays the same. The system becomes more reliable, timely, and consistent.

Best First Agent Pilots by Track

Where to start based on your role and workflow maturity

Delivery / Contracts

- Weekly CM report agent
- Minutes-to-actions and notices agent
- RFI / technical query tracker agent

Start where the format is repetitive, the inputs are known, and review can stay human-led.

Commercial / QS

- Tender review agent
- Change and early-warning agent
- Payment application check agent

Start where the format is repetitive, the inputs are known, and review can stay human-led.

Leadership

- AI register and usage review
- Vendor agent approval checklist
- Monthly pilot KPI report

Start with decisions you already make: what is in use, what is approved, and what is working.

Before we move into more agents

Most AI failures in companies aren’t because the model is dumb. It’s because the workflow is undefined. If your process isn’t clear enough to teach a new graduate, it’s not clear enough to teach an AI