Prompt Engineering Library

By Mandy Hathaway AI Ethics Specialist & Technical Writer mandyhathaway.com

A structured reference library of prompts for common AI use cases. Each entry includes the prompt, a sample input, a representative output, and a note on why the prompt is structured the way it is.

ABOUT THIS LIBRARY

Prompt engineering is the practice of designing inputs to language models in ways that reliably produce useful outputs. A prompt is a specification: it defines the task, constrains the output, establishes context, and accounts for the ways the model might misread what you actually want.

This library covers five use cases: classification, summarization, structured output, content moderation, and edge case handling. Each entry shows the prompt as written, a sample input, the resulting output, and a note on why specific choices were made.

All prompts in this library were developed and tested against GPT-4 and Claude 3 Sonnet. Performance may vary across models and versions.

01 CLASSIFICATION

Classification prompts ask the model to assign an input to one or more predefined categories. The main challenge is keeping category boundaries tight — given any ambiguity, the model will fill in the gaps, and not always the way you'd expect.

CLASSIFICATION	**Sentiment Classification with Confidence**
PROMPT	*Classify the sentiment of the following customer review. Choose exactly one label from this list: POSITIVE, NEGATIVE, NEUTRAL. Then rate your confidence on a scale of 1–5, where 5 means the label is unambiguous and 1 means the review could reasonably be classified differently. Respond in this format only: Label: [LABEL] Confidence: [1–5] Reason: [one sentence] Review:*

	"{review_text}"
SAMPLE INPUT	`Review: "The build quality is excellent and I love the design, but waiting three weeks for shipping and then getting no response from customer service when I had a problem was really frustrating."`
OUTPUT	Label: NEGATIVE Confidence: 4 Reason: The reviewer praises the product quality but expresses clear frustration with shipping time and customer service, which are the dominant tone of the review.
DESIGN NOTE	*Forcing a single label prevents the model from hedging. The confidence score surfaces ambiguous cases for human review without requiring a separate pass. The reason field catches misclassifications before they reach downstream systems.*

CLASSIFICATION **Multi-Label Topic Tagging**

PROMPT	*You are a content tagger for a technology publication. Assign topic tags to the article excerpt below. Choose all tags that apply from this list only: AI, Machine Learning, Ethics, Policy, Healthcare, Education, Climate, Security, Economics. Rules: - Apply a tag only if the topic is substantively addressed, not merely mentioned. - Return tags as a comma-separated list, ordered from most to least central. - If no tags apply, return: NONE Article excerpt: "{article_text}"*
SAMPLE INPUT	`Article excerpt: "The FTC's proposed rulemaking would require companies deploying automated decision systems in hiring and lending to disclose when AI is used and provide a human review option. Critics argue the rules don't go far enough, pointing to evidence that disclosure alone does little to address algorithmic bias."`

OUTPUT	AI, Ethics, Policy
DESIGN NOTE	*The 'substantively addressed' rule reduces noise from passing references. Ordering by centrality gives downstream ranking systems a usable signal without requiring a separate scoring prompt. Providing an explicit NONE option prevents the model from forcing inappropriate tags.*

02 SUMMARIZATION

Summarization is one of the most common use cases in production AI, and one of the most frequently under-specified. The single most useful thing you can do is define who the summary is for. An executive summary and a research abstract are structurally different documents — the prompt needs to reflect that.

SUMMARIZATION **Executive Summary with Constraints**	
PROMPT	*Summarize the following document for a senior executive who has two minutes to read it. The summary must: - Be 3–5 sentences, no longer. - Lead with the most important finding or decision point. - Avoid jargon. If a technical term is unavoidable, define it in plain language in parentheses. - End with the single most important action item, if one exists. Document: "{document_text}"*
SAMPLE INPUT	`Document: [A 12-page internal audit report examining automated loan decisions made by the company's underwriting system between 2021 and 2023, including statistical analysis of outcomes by demographic group and a review of documentation practices.]`
OUTPUT	The audit found that 34% of loan decisions made by the automated underwriting system between 2021 and 2023 cannot be explained by available data — meaning no documented factor accounts for the outcome. This affects

	approximately 12,000 applications. The system has been flagged for a full bias review (a process that checks whether decisions disadvantage protected groups). The most important action item: suspend automated underwriting for new applications pending review results.
DESIGN NOTE	*Specifying the audience and time constraint forces economy. Leading with the key finding reflects how executive communication actually works. The action item instruction prevents summaries that describe a situation without surfacing what to do about it.*

SUMMARIZATION	**Comparative Summary across Multiple Sources**
PROMPT	*You will be given three short texts on the same topic. Write a comparative summary that: 1. Identifies what all three sources agree on. 2. Identifies the most significant point of disagreement or difference in emphasis. 3. Notes any important claim made by only one source. Keep the total summary under 150 words. Do not attribute claims to specific sources by number — synthesize them. Source 1: "{text_1}" Source 2: "{text_2}" Source 3: "{text_3}"*
SAMPLE INPUT	Source 1: [MIT Technology Review excerpt on LLM risks in healthcare] Source 2: [Brookings Institution policy brief on AI regulation timelines] Source 3: [AI Now Institute report on user over-trust in automated systems]
OUTPUT	All three sources agree that large language models pose measurable risks in high-stakes decision-making contexts, particularly in healthcare and legal settings. The most significant disagreement is on timeline: two sources treat meaningful regulation as a near-term possibility, while one argues that

	legislative capacity is too limited for effective oversight in the next decade. One source uniquely emphasizes that the greatest near-term risk is not model failure but over-trust by users who lack the background to detect errors.
DESIGN NOTE	*The three-part structure (consensus, disagreement, unique claims) is more useful for research synthesis than a flat summary. Prohibiting source attribution forces genuine synthesis rather than sequential paraphrase. The word limit prevents the prompt from becoming a restatement exercise.*

03 STRUCTURED OUTPUT

Structured output prompts ask the model to return data in a specific format: JSON, tables, numbered lists, custom schemas. The problem is that models drift. The longer the output, the more likely they are to ignore format instructions. The prompts below are designed to hold format across varied inputs.

STRUCTURED OUTPUT	**JSON Extraction from Unstructured Text**
PROMPT	*Extract the following fields from the job posting below and return them as a valid JSON object. If a field is not present in the posting, use null. Fields: - title (string) - company (string) - location (string or "Remote") - salary_range (string or null) - required_skills (array of strings) - experience_years_min (integer or null) Return JSON only. No explanation, no markdown formatting, no code fences. Job posting: "{posting_text}"*
SAMPLE INPUT	Job posting: "Vela Systems is hiring a Senior ML Engineer to join our distributed training team. You'll be working on large-scale model training infrastructure. We're looking for 5+

	years of ML experience with strong Python and PyTorch skills, experience with MLOps tooling, and a background in distributed systems. Fully remote. Salary $160k–$195k depending on experience."
OUTPUT	{ "title": "Senior ML Engineer", "company": "Vela Systems", "location": "Remote", "salary_range": "$160,000–$195,000", "required_skills": ["Python", "PyTorch", "distributed training", "MLOps"], "experience_years_min": 5 }
DESIGN NOTE	*Listing exact field names and types reduces hallucination of extra fields. Specifying null for missing values prevents the model from inventing plausible-sounding data. 'No markdown formatting, no code fences' is essential — models default to wrapping JSON in backticks, which breaks downstream parsers.*

04 CONTENT MODERATION

Content moderation is where prompt failures have the most direct impact on real people. Over-moderate and you suppress legitimate speech; under-moderate and you let harmful content through. The prompt below is built around a tiered system that routes ambiguous cases to humans rather than making automated calls on judgment-heavy content.

CONTENT MODERATION	**Tiered Risk Classification**
PROMPT	*You are a content safety reviewer. Classify the following user-submitted text using the risk tiers below. Apply the lowest tier that accurately describes the content — do not escalate unless the criteria are clearly met. Tier 1 — SAFE: No policy concerns. Tier 2 — REVIEW: Potentially sensitive but not clearly violating. Route to human review. Tier 3 — HOLD: Likely policy violation. Suppress pending human*

	review. Tier 4 — REMOVE: Clear, unambiguous policy violation. Respond in this format: Tier: [1/2/3/4] Label: [SAFE/REVIEW/HOLD/REMOVE] Reason: [one sentence describing the specific concern, or "No concern identified" for Tier 1] Text: "{user_text}"
SAMPLE INPUT	Text: "I've been feeling really low lately and I've been reading about what happens if you take too much of [medication name]. Does anyone know the exact amounts that would be dangerous?"
OUTPUT	Tier: 2 Label: REVIEW Reason: The text contains detailed inquiry about medication overdose thresholds in a context that is ambiguous between safety research and potential self-harm intent.
DESIGN NOTE	*Tiered output gives the operations team a graduated response rather than a binary flag. 'Apply the lowest tier' reduces over-moderation. The reason field is required for audit trails and to support appeals. This prompt is intentionally conservative — edge cases go to human review rather than automated action.*

05 EDGE CASE HANDLING

A prompt that works well on clean inputs will often break on real-world data: ambiguous requests, unexpected formats, off-topic questions, adversarial phrasing. Edge cases are where most production prompt failures actually happen. The prompts below address the ones that come up most often.

EDGE CASES **Graceful Handling of Out-of-Scope Requests**	
PROMPT	*You are a customer support assistant for a cloud storage product. Answer questions about account management, billing, file storage, sharing, and technical troubleshooting only. If the*

	user's question falls outside these topics: - Do not attempt to answer it. - Acknowledge the question briefly. - Redirect to the relevant resource if one exists, or explain that this topic is outside your scope. - Do not apologize more than once. User message: "{user_message}"
SAMPLE INPUT	User message: "I think your company is violating GDPR by storing my data in the US. I want to know what my legal options are."
OUTPUT	That question is outside what I can help with here — I'm set up to handle account, billing, and storage questions specifically. For legal and privacy inquiries, your best contact is privacy@example.com. Is there anything about your account or storage settings I can help with in the meantime?
DESIGN NOTE	*Explicit scope definition prevents the model from attempting answers in domains where it may be unreliable or where incorrect information carries liability. The 'do not apologize more than once' instruction prevents the sycophantic over-apologizing that degrades user experience. Offering a redirect maintains helpfulness within constraints.*

EDGE CASES Handling Ambiguous Instructions

PROMPT	*Complete the following task. If the instruction is ambiguous in a way that would significantly affect the output, state the ambiguity in one sentence and ask the single most important clarifying question before proceeding. If you can complete the task without clarification, do so directly. Task: "{task_description}"*
SAMPLE INPUT	Task: "Rewrite this in a more professional tone: 'Hey just checking in

	on that report, any idea when it'll be done? Getting some pressure from above lol'"
OUTPUT	The instruction asks me to rewrite for a more professional tone, but the appropriate register depends on the relationship between sender and recipient — a message to a peer reads differently than one to a direct report or a senior stakeholder. Who is the intended recipient of this message? [Awaiting clarification before rewriting.]
DESIGN NOTE	*Allowing the model to surface ambiguity prevents it from silently choosing an interpretation that may not match user intent. Limiting the clarifying question to one prevents interrogation loops. The instruction to proceed directly when possible avoids unnecessary friction on clear tasks.*

06 SAMPLE OUTPUTS IN CONTEXT

The following examples walk through each prompt type with a real input, the full model output, and a note on what the output is actually doing. The goal is to show not just that the prompts work, but why the outputs look the way they do.

Sentiment Classification: Borderline Case

Input review: "The product itself is great but the app is terrible and crashes constantly. Customer service was helpful when I called but it took forever to get through. Overall I guess it's okay."

Expected output:

```
Label: NEUTRAL
Confidence: 2
Reason: The review contains roughly equal positive and negative content across
different product dimensions, with no dominant emotional tone.
```

The confidence score of 2 is doing real work here. A binary sentiment classifier would call this NEUTRAL and move on. This output flags it for human review instead, which is the right call for a review this mixed. The reason field makes it auditable.

JSON Extraction: Missing Fields

Input posting: "Northlight AI is looking for an ML researcher to join our safety team. Strong Python skills required. This is a hybrid role based in San Francisco."

Expected output:

```
{
  "title": "ML Researcher",
  "company": "Northlight AI",
  "location": "San Francisco",
  "salary_range": null,
  "required_skills": ["Python"],
  "experience_years_min": null
}
```

The null instruction is the critical piece. Without it, the model is likely to either invent a salary figure or skip the field entirely — both of which break downstream processing. Null is clean; a missing key throws an error.

Content Moderation: Clear Safe Case

Input text: "Does anyone know a good recipe for banana bread? I have some really ripe bananas I need to use up."

Expected output:

```
Tier: 1
Label: SAFE
Reason: No concern identified.
```

The 'apply the lowest tier' instruction matters as much for benign content as for harmful content. A system that flags banana bread recipes erodes user trust quickly. This output shows the prompt is calibrated in both directions, not just tuned to catch violations.

Ambiguous Instruction: Proceeding Directly

Input task: "Translate this sentence into Spanish: 'The meeting has been rescheduled to Thursday at 2pm.'"

Expected output:

La reunión ha sido reprogramada para el jueves a las 2pm.

No clarification needed, so none is requested. This is the prompt's conditional logic working as intended. A poorly designed version of this prompt would ask a question anyway, which trains users to expect interrogation before every task.