LLM Prompt Engineering Playbook

A Professional Toolkit Resource by TheConsultAI

This playbook contains advanced prompting patterns, structural system templates, and cost-control guardrails tailored for enterprise environments running LLMs at scale.

Three Core Prompt Engineering Rules

- Rule 1: Use XML tags for structure. Wrap inputs, examples, and outputs in custom XML tags (, ,) to clarify instructions to the attention mechanism.
- Rule 2: Chain of Thought. Force the LLM to output its reasoning step-by-step before producing the final payload (e.g., 'Analyze the data in tags first, then output JSON').
- Rule 3: Enforce JSON Schema. Provide explicit, valid JSON schemas and validate model outputs programmatically at runtime to catch formatting drift.

Production Prompt Template: Structured Data Extractor

System Instructions	You are an expert financial extractor. Output strictly valid JSON. Do not write markdown blocks or text preamble. Follow the schema: { 'revenue': float, 'margin': float }.
Few-Shot Examples	Company A Q3 revenue was $12M with 15% net margin{ 'revenue': 12000000.0, 'margin': 0.15 }
User Payload	{USER_UNSTRUCTURED_DOCUMENT}
Reasoning Trigger	First output your breakdown steps in tags, then return the JSON payload in tags.

Cost & Latency Optimization

Enterprise setups must optimize prompts to save up to 40% in monthly API costs:

- Prompt Caching: Use models that cache system prompts and few-shot examples (saves 50% on input tokens).
- Token Budgeting: Enforce max_tokens at the API level to prevent uncontrolled loops.
- Model Tiering: Route simple queries (sorting, sentiment) to fast, inexpensive models (e.g., Claude Haiku or GPT-4o-mini), and complex tasks to frontier models.