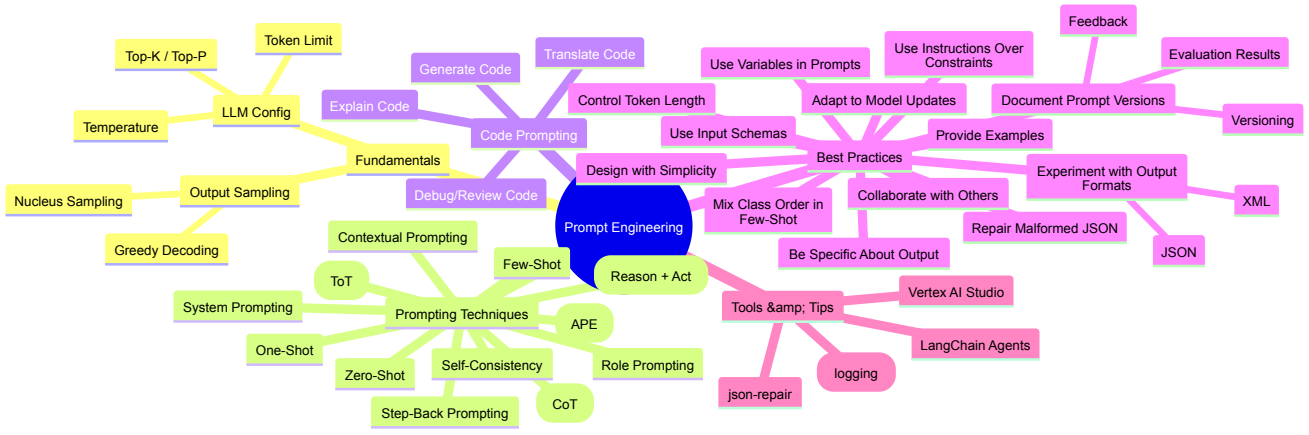
Prompt Engineering Cheatsheet

In the age of generative AI, **prompt writing has become a core engineering skill**. Whether building a chatbot, generating code, or designing autonomous agents, the **prompt is your most powerful interface with a Large Language Model (LLM)**.

This post distills the key ideas from Lee Boonstra's excellent "Prompt Engineering" (Google Cloud) whitepaper, adding my structure and visuals to offer you a **practical cheat sheet and mind map**. It's ideal for AI developers, product engineers, and anyone working with LLMs.

Prompt engineering isn't just writing instructions — it's an iterative, analytical, and creative process. Techniques like **Chain of Thought**, **ReAct**, or **Step-Back Prompting** push LLMs beyond surface-level output and unlock deeper reasoning, precision, and control.



I decided to share this cheat sheet and mind map because I've seen firsthand how powerful prompt engineering can be—but also how easy it is to get lost in the noise of examples, techniques, and model quirks. Whether you're building products with LLMs or just exploring what's possible, having an explicit, structured reference helps cut through trial and error and brings clarity to experimentation. This is my way of consolidating what works, inspired by real-world use cases and backed by proven strategies—hoping it saves others time and helps push ideas forward faster.

The Cheatsheet

Technique / Best Practice	Description	When to Use	Example
Zero-Shot Prompting	Directly instructs the model without examples.	Simple tasks, clear intent.	"Classify this review as POSITIVE, NEUTRAL, or NEGATIVE."
One-Shot Prompting	Includes one example in the prompt.	Slightly ambiguous tasks.	"Translate: Bonjour → Hello" with one example before.

Technique / Best Practice	Description	When to Use	Example
Few-Shot Prompting	Includes 3–5 examples. Helps pattern recognition.	Complex tasks needing structure or format guidance.	Parsing pizza orders into JSON with multiple examples.
System Prompting	Defines model's global behavior or rules.	To shape response style/format.	"You are a JSON API. Always return valid JSON."
Role Prompting	Assigns a persona or role to the model.	For tone, domain knowledge, or consistency.	"Act as a travel guide and recommend places in Rome."
Contextual Prompting	Supplies relevant background or task details.	When the model needs situational context.	"You are writing for a retro gaming blog..."
Step-Back Prompting	Starts with abstraction before solving the task.	When stuck or needing creativity and reasoning.	"What makes a good FPS level? Now write one using that."
Chain of Thought (CoT)	Encourages the model to explain reasoning steps.	Logic, math, planning tasks.	"Let's think step by step..."
Self-Consistency	Samples multiple CoTs and picks most frequent answer.	High-stakes reasoning tasks.	Run same prompt 5 times, then majority vote.
Tree of Thoughts (ToT)	Explores multiple reasoning branches.	Complex problem-solving.	Used in decision-tree like reasoning problems.
ReAct (Reason + Act)	Combines reasoning with tool use (API, search).	Interactive agents.	LangChain agents using search to answer real-world queries.
Automatic Prompt Engineering (APE)	LLM generates variations of a prompt.	Exploratory prompt generation.	"Give 10 ways to phrase: 'Order one Metallica T-shirt size S'"
Use Examples	Provide exemplar inputs and outputs.	To teach format, logic, or behavior.	Few-shot classification, formatting JSON, style guides.
Design with Simplicity	Clear, concise language improves output.	All prompts.	Rewrite verbose questions into actionable commands.
Be Specific About Output	Define output type, format, or structure.	When output must be structured or constrained.	"Return result in 1 paragraph, JSON format, uppercase only."

Technique / Best Practice	Description	When to Use	Example
Prefer Instructions Over Constraints	State what to do instead of what not to do.	General rule — positive prompts perform better.	DO: "Return product list in JSON" vs. DON'T: "Do not use Markdown."
Control Token Length	Use config or text instruction to limit output.	For concise or cost-sensitive output.	"Summarize this in one tweet."
Use Variables in Prompts	Parametrize prompts for reusability.	When integrating into apps or dynamic inputs.	"Tell me a fact about the city: {city}"
Mix Few-Shot Class Orders	Vary label order in classification tasks.	To avoid bias.	Rotate order of POSITIVE, NEUTRAL, NEGATIVE in examples.
Adapt to Model Updates	Tweak prompts when LLMs change.	New model releases or upgrades.	Re-test and iterate prompts after upgrading from GPT-3.5 to GPT-4.
Experiment with Output Formats	Use JSON/XML for structured tasks.	Data extraction, parsing, sorting.	Ask model to "Return JSON with keys: name, price, date"
Repair Malformed JSON	Use tools like `json-repair` to fix broken output.	When outputs are long or truncated.	Useful in pipelines processing LLM-generated data.
Use JSON Schemas for Input	Provide structured input for better LLM understanding.	When prompting with structured data.	Schema: product → name, category, price, features, release_date
Collaborate on Prompt Design	Share prompts across engineers and compare results.	Cross-team prompt optimization.	A/B testing different prompt versions collaboratively.
Document Prompt Versions	Log prompts, configs, and outputs for traceability.	All projects, especially production.	Google Sheets / YAML with fields: prompt, model, temp, outputs, notes.