Think Before You Prompt

Idea In Short

Organizations should stop treating poor generative Artificial Intelligence [GenAI] results as proof that the system failed. In many cases, the deeper problem is upstream: users do not know how models work, what they are good at, where they break, or how to shape inputs so that outputs match real expectations. Prompt patterns matter, but they decay as models, interfaces and context windows change. First-principles thinking lasts longer because it teaches users to reason from the nature of the task, the structure of the model, the risks of error and the evidence required for a usable answer. A course, workshop, or book built around that discipline would strengthen adoption, improve user experience, reduce blame-shifting and help organizations get more value from tools such as ChatGPT, Gemini and Perplexity

Generative Artificial Intelligence has made expert-looking output available at conversational speed. A researcher can ask for a literature map, a scientist can request a plain-language explanation of a method and a business professional can ask for a strategy memo or meeting brief within seconds. This feels like a step change in access to useful information. It is. But the convenience creates its own misunderstanding. Many users now believe that because the interface is simple, the thinking required to use it well must also be simple.

That assumption is proving costly. As large language models spread into daily work, many disappointing results come not from catastrophic system failure but from weak user framing. People ask for too little context, specify the task poorly, fail to define what a good answer looks like and then conclude that the model is shallow or unreliable. Some of those critiques are fair. Many are not. The harder truth is that the user often has more responsibility for output quality than the user experience suggests.

Prompting is not enough

Prompting techniques matter. Clear instructions, examples, output constraints and

structured context all improve model performance. OpenAI's own guidance emphasizes placing instructions early, separating context clearly, being specific about the desired output and using examples when needed1. Similar advice appears across the major model providers.

Yet prompt techniques alone are not a durable foundation. They are often learned as recipes detached from the underlying problem. Users memorize patterns such as role prompting, few-shot examples, chain-of-thought prompting, or structured output requests, then treat those patterns as universal keys. The moment the model changes, the interface evolves, or the task type shifts, the recipe starts to crack.

That fragility is becoming more visible as each model generation changes what works best. Provider guidance now differs in how it recommends ordering instructions, delimiting context, using examples and controlling reasoning. Practical cross-model comparisons show that OpenAI, Anthropic and Google converge on some core principles, but still differ in structure, strengths and failure modes2. A user trained only in fixed prompt patterns will struggle each time the context shifts.

The case for first principles

First-principles thinking is more durable because it begins with the nature of the problem rather than the borrowed shape of the prompt. Instead of asking, "Which trick should I use?" the user asks more basic questions. What is the task? What kind of knowledge does it require? What would make the answer actionable? Where is the error risk highest? What context is essential? What output format would make the result useful in the real workflow?

This way of thinking matters because large language models do not read intention directly. They infer it from the evidence the user supplies. If the user has not clarified the objective, audience, decision context, format, constraints and source expectations, the model fills the gap with statistical plausibility. The answer may sound polished while still missing the user's real need.

First-principles thinking therefore shifts the work upstream. It forces the user to define the problem before asking the model to solve it. That is not a limitation of GenAI alone. It is true of most serious knowledge work. The difference is that GenAI systems can hide poor task definition by producing language that looks complete.

Why users blame the system

When users receive generic or unhelpful answers, they usually see only the output and the model name. They do not see the chain of assumptions that led there. This makes it easy to blame the system rather than inspect the request. The emotional logic is understandable. If a tool feels conversational, users expect it to grasp vague intent the way a skilled colleague might after years of context.

But models do not work that way. They respond to the prompt, the provided context, the accessible tools and the model's training and runtime behavior. If the user says, "Give me a strategy for entering a new market", the model can generate something plausible. If the user instead defines the geography, target segment, time horizon, cost constraints, regulatory risk and required decision format, the output usually improves sharply. The model did not become wiser in the second case. The task became better specified.

Business guidance on GenAI use increasingly reflects this point. Harvard Business Review material on prompting and use-case selection stresses that useful results depend on clarity, context, boundaries and examples rather than vague requests for generic content3. The practical lesson is simple:

> many "AI failures" are actually failures of problem definition

Models change, principles last

This is why a course or workshop focused only on prompt engineering will age quickly unless it is anchored in deeper principles. BERT, ChatGPT, Gemini, Claude, Perplexity and whatever follows do not merely differ in branding. They differ in architecture, context handling, tool use, reasoning controls, latency, modality, retrieval behavior and interface design. Even within one family, model versions can change enough to invalidate old habits.

What lasts across those changes are the basic questions users ask before they prompt. What is the job to be done? Is this a summarization task, a reasoning task, a creative task, a retrieval task, or a decision-support task? Does the answer require current facts, domain expertise, calculation, judgment, or format precision? Should the model be grounded in

sources or documents? Would examples reduce ambiguity? What failure would be most costly?

These are first-principles questions because they come before the wording of the prompt. They help the user choose the right model behavior and the right prompting technique instead of applying techniques blindly. Model providers now recommend building evaluations and monitoring prompt performance over time for exactly this reason:

> behavior shifts and prompt quality has to be judged against the task rather than assumed from habit4.

What a real course should teach

A serious first-principles course on GenAI should not begin with prompt formulas. It should begin with model literacy. Users need a working understanding of what a large language model is, what it predicts, what it does not know, where hallucinations come from and why specificity changes the output distribution. They do not need research-level mathematics for this. They do need enough conceptual grounding to stop treating the model as magic.

From there, the course should move into task decomposition. Users should learn to separate a vague request into parts:

1. objective
2. audience
3. context
4. constraints
5. evidence
6. tone
7. format, and
8. evaluation criteria

This is where prompting techniques become useful, because they can now be taught as instruments serving a defined purpose rather than rituals copied from social media threads.

The course should also teach model-task fit. Not all tasks deserve the same trust. Harvard Business Review's organizational guidance on GenAI argues that the cost of error and the type of knowledge required should shape how aggressively firms apply these tools5. That insight belongs in user training. A person drafting internal brainstorming notes can tolerate more error than a person using GenAI to support legal, scientific, medical, or financial decisions.

Finally, the course should teach evaluation. Users need to know how to interrogate outputs, compare alternatives, spot unsupported claims, request revisions and decide when to leave the model and verify elsewhere. Good GenAI use is not only a prompt skill. It is an inspection skill.

The upstream shift in responsibility

This kind of education changes the user-system relationship. It moves more of the onus upstream to the user, but in a productive way. That is not about blaming users for every bad result. Models do have real limitations. It is about making users more competent in preparing the system to produce outputs aligned with actual expectations.

That matters for user experience. Poor experiences often come from a mismatch between what the user wanted and what the system could infer. When users learn to express the problem more precisely, the number of generic or irrelevant responses falls. The interaction feels less random because the user now understands which levers matter.

This is why first-principles thinking can improve adoption more than another list of prompt hacks. People continue to use tools that help them feel capable. If GenAI systems repeatedly produce generic output, confidence drops. If users learn how to shape the interaction, inspect results and recover from failure, trust grows. McKinsey and other observers of enterprise adoption have repeatedly noted that the bottleneck in GenAI value capture is not just access to tools but the ability to redesign work and build skills around them6.

From skill to flywheel

The strongest case for a first-principles course, book, or workshop is not that it will make everyone a prompt engineer. It is that it will produce better users. Better users define tasks

more clearly, choose techniques more deliberately, evaluate outputs more critically and know when the model is the wrong tool. Those habits raise the quality of outcomes without tying people to one vendor or one version.

That creates the flywheel the user described. Better thinking leads to better prompting. Better prompting leads to more useful outputs. More useful outputs improve trust and willingness to adopt the tools more broadly. Wider use creates more feedback, sharper judgment and better institutional learning. The system improves, but so does the human using it.

1. Teach model literacy before prompt recipes
2. Train users to define the task before they draft the request
3. Match the prompting method to the problem, not the other way around
4. Show users how to inspect outputs, not just generate them
5. Tie GenAI usage to decision quality and workflow outcomes, not novelty alone

GenAI adoption will continue, with or without thoughtful education. The real question is whether organizations want adoption that is shallow and frustrating or adoption that compounds. First-principles thinking is how they shift from one to the other.

- 1Best Practices For Prompt Engineering With The OpenAI API
- 2Prompt Engineering Across The OpenAI, Anthropic And Gemini APIs
- 3An Executive's Guide To Generative AI: Finding The Right Use Cases And Crafting Effective Prompts
- 4OpenAI Prompt Engineering Guide
- 5The Gen AI Playbook For Organizations
- 6The Economic Potential Of Generative AI

Summary

Generative Artificial Intelligence does not eliminate the need for thinking. It raises the premium on it. Users who rely on borrowed prompt formulas without understanding the task, the model, or the required output often produce generic, brittle and disappointing results. They then blame the system for failing to read their intent. First-principles thinking changes

that relationship. It moves preparation upstream by teaching people to define the objective, clarify the constraints, select the right model behavior and inspect the answer against evidence and use case. That shift improves adoption because it gives users more control over outcomes and a clearer sense of responsibility. It also improves experience because higher-quality inputs usually produce more useful outputs. Over time, that stronger experience compounds into trust, repeat usage and the flywheel that real adoption requires