Designing AI Systems: A Framework for AI Capabilities

Introduction

As organizations adopt AI-powered products and agentic systems, design teams face a new challenge: how to translate user goals into capabilities that AI systems can reliably execute.

This isn't primarily a technology problem. It's a design problem.

The challenge isn't deciding which model to use or which framework to implement. The challenge is understanding what people are trying to accomplish, identifying the capabilities required to support that goal, and designing systems that reliably, transparently, and at scale invoke those capabilities.

This framework provides a practical approach for thinking about AI capabilities, experience design, and system architecture from a product and design perspective.

What Is a Capability?

A capability is a discrete action, service, or behavior that an AI system can perform to support a broader user goal.

Capabilities are the building blocks of AI-enabled experiences. They allow teams to think beyond interfaces and focus on what the system must actually be able to do.

Good capabilities have three characteristics:

A clear input
What information is required?

A defined output
What result is produced?

A single responsibility
It does one thing well.

Examples:

- Verify account status
- Retrieve inventory availability
- Calculate shipping costs
- Explain billing changes

- Escalate to a human expert

The capability is not the experience itself. It is one component of the system that makes the experience possible.

The Evaluation Framework: Should This Be a Capability?

When evaluating functionality, ask:

Can it be invoked independently?

Could the system perform this action without requiring several other actions to happen first?

If yes, it's likely a capability.

Does it have a clear success state?

Can you define what "done" looks like?

If success cannot be clearly measured, the capability is probably too broad.

Is it reusable?

Can multiple experiences benefit from the same capability?

High reusability usually indicates high strategic value.

Does it connect to a specific source of truth?

Capabilities often represent clean boundaries between systems, data sources, and experiences.

Foundational Capabilities vs. Orchestrated Experiences

Not everything users experience maps directly to a single capability.

Foundational Capabilities

These are focused actions that perform a specific task.

Examples:

- Retrieve account balance
- Locate nearby stores
- Verify identity
- Check inventory availability

They are highly reusable and relatively easy to govern.

Orchestrated Experiences

These combine multiple capabilities to help users accomplish meaningful goals.

Examples:

- Explain why my bill changed
- Help me choose the right plan
- Compare available products
- Resolve a delivery issue

Users experience these as a single interaction.

Under the hood, the system may be coordinating many capabilities at once.

This distinction matters because users care about experiences, while systems operate through capabilities.

Designers need to understand both.

Designing for Composition

Most user goals require more than one capability.

The design challenge is not simply defining capabilities.

The challenge is understanding:

- Which capabilities need to exist
- How they work together
- What happens when one fails
- How the system communicates outcomes to users

The quality of an AI-enabled experience is often determined less by individual capabilities and more by how effectively they are orchestrated.

Prioritization

Not every capability deserves immediate investment.

High-priority capabilities tend to be:

High Frequency + High Friction

Users encounter them frequently and current solutions create frustration.

Reusable Across Experiences

The same capability unlocks value in multiple journeys.

Dependent on Real-Time Information

The answer changes frequently and accuracy matters.

High Trust or Regulatory Impact

Mistakes create risk for customers or organizations.

Experience Differentiators

Capabilities that unlock outcomes that were previously difficult or impossible.

Designing Beyond Happy Paths

Capabilities rarely fail gracefully on their own.

Designers should consider:

- What happens when data is unavailable?
- What happens when confidence is low?
- What happens when the system produces multiple possible answers?
- When should a human become involved?
- How should uncertainty be communicated?

Designing recovery paths is often as important as designing successful outcomes.

The Questions Designers Should Ask

As AI systems become more capable, the role of design expands.

Questions worth asking include:

- What user goal does this capability support?
- How will people experience the outcome?
- What assumptions is the system making?
- What happens when the capability fails?
- How do we maintain transparency and trust?
- How do we evaluate whether the capability is actually helping?

These questions help connect technical implementation to real human outcomes.

Final Thought: Capabilities Build Systems. Experiences Build Trust.

Capabilities are the building blocks.

Experiences are how people encounter them.

Systems determine how they work together.

The challenge is rarely the AI itself.

The challenge is understanding what people are trying to accomplish, identifying the capabilities required to support that goal, and designing systems that reliably, transparently, and at scale invoke those capabilities.

That's fundamentally a design problem.