PREMIUM GUIDE · LEARN.AIWITHPRIYABRAT.COM

The GenAI System Design Interview Guide

A repeatable framework, the estimation toolkit, four complete case studies and the trade-offs that tie them together, with model answers for every follow-up question.

By **Priyabrat Dalbehera** · GenAI engineer

8 chapters · 4 full case studies · 6 solved practice problems · 25 interview questions

FREE PREVIEW · FULL GUIDE AT LEARN.AIWITHPRIYABRAT.COM

learn.aiwithpriyabrat.com

01 How the GenAI design round works

02 The toolkit: questions, estimates, architecture

03 Case: customer support chatbot

04 Case: document Q&A at scale

05 Case: coding assistant

06 Case: multi-agent invoice workflow

07 Trade-offs, latency and capacity

08 Practice problems and a mock interview

© 2026 Priyabrat Dalbehera

learn.aiwithpriyabrat.com · Edition 1.0 · October 2026

Contents

1 **How the GenAI design round works**
What's tested, scoring, the 7-step framework, mistakes

2 **The toolkit: questions, estimates, architecture**
Questions, estimator, formulas, reference architecture

3 **Case: customer support chatbot**
RAG + tools, routing, permissions, handoff, deflection

4 **Case: document Q&A at scale**
50M documents, index size, tenants, permissions, freshness

5 **Case: coding assistant**
Fast completions, repository context, agent mode, privacy

6 **Case: multi-agent invoice workflow**
Workflow vs agents, durable state, approvals, audit

7 **Trade-offs, latency and capacity**
12 decisions, latency budgets, limits, numbers to remember

8 **Practice problems and a mock interview**
6 problems with full solutions, a condensed mock interview

FREE PREVIEW · FULL GUIDE AT LEARN.AIWITHPRIYABRAT.COM

Learn the framework
Chapters 1–2: the steps, questions, estimates and reference architecture.

Study the cases
Chapters 3–6: requirements to failure modes, with model answers.

Practise aloud
Chapter 8: six timed problems with full solutions, and a mock interview.

learn.aiwithpriyabrat.com

CHAPTER 1

How the GenAI design round works

What interviewers listen for, and a framework you can run in 45 or 30 minutes.

1.1 What the round is really testing

A GenAI system design round sounds like a classic one ("design a support chatbot for ten million users"), but the interviewer is listening for different things. Four questions run through every good answer:

1. **How will we know it's good?** Quality is statistical; you need an evaluation plan.
2. **How long does the user wait?** LLM calls take seconds; streaming and async work matter.
3. **What does each request cost?** Tokens are billed; cost can dominate the design.
4. **What stops it doing harm?** Prompt injection, data leaks and unsafe actions are real.

Classic system design	GenAI system design adds
Deterministic correctness	Statistical quality → **evaluation**
Millisecond latency	Multi-second calls → **streaming, async**
Cost = servers	Cost = **tokens per request**
Your own throughput limits	**Provider rate limits**, GPU capacity
Auth and input validation	**Injection, data leakage, unsafe actions**
Data model and storage	**Retrieval**: chunking, embeddings, indexes, freshness

FREE PREVIEW · FULL GUIDE AT LEARN.AIWITHPRIYABRAT.COM

1.2 How you're scored

Signal	Strong candidate	Weak candidate
Requirements	Asks about users, quality bar, data, latency, budget, risk before drawing	Draws boxes in minute one
Numbers	Estimates tokens, cost, peak load; uses them to decide	"We'll use GPT" with no numbers
Quality	Has an evaluation plan: offline set, online signals	Never mentions how to measure answers
Retrieval	Chunking, hybrid search, reranking, **permissions**, freshness	"Put it in a vector DB"
Safety	Guardrails, approvals for actions, injection defence	Lets the model act freely
Trade-offs	States both sides, the deciding factor, when to revisit	Lists technologies
Communication	Signposts, checks in, manages time	Monologue, runs out of time

1.3 The 7-step framework, with what to say

Step 1 · Clarify (5 min). Say: "*Before designing, let me ask a few questions so I optimise for the right things.*" Then cover users and scale, the 2–3 core use cases, the **quality bar** and cost of a wrong answer, data sources and permissions, latency expectations, whether the system **acts** or only answers, and constraints (budget,

residency, stack). Finish by **writing down assumptions**: *"I'll assume 200k daily users, English and Hindi, first token under a second, and that refunds above ₹2,000 need a human."*

Step 2 · Estimate (5 min). Requests per day and at peak, tokens per request, peak tokens per minute, cost per request and per month, storage for documents and vectors. Say the arithmetic out loud and round aggressively. *"Peak tokens per minute is about 5.5 million, which is above a default quota, so routing and caching matter."*

Step 3 · High-level architecture (8 min). Draw the request path first (client → gateway → orchestration → retrieval / tools / models → response), then the data path (ingestion → index), then async paths (queues, workers). Label arrows.

Step 4 · Data and retrieval (8 min). Sources and connectors, parsing, chunking, embeddings, hybrid search, reranking, permission filters, freshness and deletion.

Step 5 · Model strategy (6 min). API vs self-host; which model where (router, small model for easy turns, large for hard); prompting vs fine-tuning (and why not to fine-tune on day one); prompt versioning.

Step 6 · Evaluation and guardrails (6 min). Offline eval set and CI gate; online metrics; input and output guardrails; approvals for risky actions; handling prompt injection.

Step 7 · Scale, reliability, cost, wrap-up (7 min). Caching, rate limits, fallbacks, monitoring and alerts; the **top bottleneck** at 10× and how you'd handle it; what you'd build next. End with a two-sentence summary that maps back to the requirements.

FREE PREVIEW · FULL GUIDE AT LEARN.AIWITHPRIYABRAT.COM

> *"Let me state my assumptions so we can adjust them."* · *"The bottleneck at 10× will be provider tokens per minute."* · *"We trade X for Y; given the requirement for Z, that's the right call."* · *"Should I go deeper on retrieval, or move on to evaluation?"* · *"For v1 I'd keep it simple with X, and move to Y when we see Z."*

1.4 The 30-minute version

Some rounds are 30 minutes. Compress, don't skip: clarify (3), estimate (3), architecture (7), **one** deep dive the interviewer chooses (10), evaluation + guardrails (4), wrap-up (3). Say up front: *"In 30 minutes I'll keep estimates brief and go deep on one area of your choice."*

1.5 Common mistakes, and the fix

Mistake	Fix
Designing before asking	Five minutes of questions; write assumptions down
No numbers	Always estimate tokens and cost per request
No evaluation story	Offline set + CI gate + online feedback, every time
Ignoring permissions	Filter retrieval and tools as the end user
"The agent will figure it out"	Workflow first; agents only where steps vary
Assuming the provider never fails	Timeouts, retries, fallbacks, rate limits
Ending abruptly	Recap, bottleneck, next steps

1.6 Interview answers

Q How is designing a GenAI system different from a classic system design?

Output quality is statistical, so the design needs an evaluation plan; LLM calls take seconds, so streaming and asynchronous work matter; cost is per token and can dominate the budget; provider rate limits and GPU capacity bound throughput; there are new risks such as prompt injection, data leakage and unsafe actions; and retrieval adds a whole data pipeline with chunking, embeddings, permissions and freshness.

Q What do you do in the first five minutes?

Clarify: users and scale, the core use cases and what's out of scope, the quality bar and the cost of a wrong answer, data sources and permissions, latency targets, whether the system takes actions, and constraints like budget and data residency. Then write down explicit assumptions and a plan for the session.

learn.aiwithpriyabrat.com

FREE PREVIEW · FULL GUIDE AT LEARN.AIWITHPRIYABRAT.COM