System Design — Backend/Full-Stack

System Design | Backend/Full-Stack

Evaluate system design skills: scalability, trade-offs, and architecture decisions.

Evaluation Criteria

Requirements Gathering & Scoping

Evaluates the candidate's ability to clarify ambiguous requirements and define system scope before designing.

Rating: □ 1 □ 2 □ 3 □ 4 □ 5

Sample Questions:

- What questions would you ask before starting the design?
- What are the most important functional and non-functional requirements?
- How many users and what traffic patterns should we design for?

✓ STRONG SIGNAL

Candidate asks targeted questions about scale, consistency requirements, latency expectations, and user patterns. They scope the problem to a manageable size and explicitly state assumptions.

✕ WEAK SIGNAL

Candidate starts drawing boxes without understanding what they're building, makes assumptions without stating them, or cannot distinguish between must-have and nice-to-have requirements.

High-Level Architecture & Component Design

Assesses the ability to design a coherent system architecture with well-defined components and clear data flow.

Rating: □ 1 □ 2 □ 3 □ 4 □ 5

Sample Questions:

- Walk me through the high-level architecture of your design.
- How does data flow through the system from the user to the database?
- Why did you choose a microservices vs. monolithic approach here?

✓ STRONG SIGNAL

Candidate draws a clear architecture diagram with well-defined components (API gateway, services, cache, database, queue). Data flow is logical, and each component has a clear responsibility.

✕ WEAK SIGNAL

Architecture is a random collection of boxes with no clear data flow, candidate cannot explain why components exist, or the design is either over-engineered or overly simplistic for the requirements.

Data Model & Storage Design

Evaluates choices around data modeling, database selection, and storage patterns.

Rating: □ 1 □ 2 □ 3 □ 4 □ 5

Sample Questions:

- How would you model the core data entities and their relationships?

- Why did you choose this database type for this use case?
- How would you handle data that needs to be queried in different ways?

✓ STRONG SIGNAL

Candidate designs a clear data model, selects appropriate databases with justification (SQL for relational data, Redis for caching, Elasticsearch for search), and discusses indexing, partitioning, and replication strategies.

✕ WEAK SIGNAL

Candidate uses a single database for everything without justification, cannot define a basic data model, or ignores data access patterns when choosing storage.

Scalability & Performance

Assesses understanding of horizontal scaling, caching, load balancing, and performance optimization.

Rating: □ 1 □ 2 □ 3 □ 4 □ 5

Sample Questions:

- How would this system handle 10x the current traffic?
- Where would you add caching, and what caching strategy would you use?
- What would be the bottleneck in this design, and how would you address it?

✓ STRONG SIGNAL

Candidate identifies bottlenecks proactively, proposes specific solutions (horizontal scaling, read replicas, CDN, cache layers), and discusses trade-offs of each approach including cache invalidation strategies.

✕ WEAK SIGNAL

Candidate says 'just add more servers' without specifics, doesn't identify bottlenecks, or proposes solutions that introduce new problems without acknowledging them.

Trade-off Analysis & Decision Making

Evaluates the candidate's ability to articulate trade-offs and make justified design decisions.

Rating: □ 1 □ 2 □ 3 □ 4 □ 5

Sample Questions:

- What are the trade-offs of your chosen approach?
- If you had to optimize for cost instead of latency, what would change?
- What would you sacrifice first if you had to simplify this design?

✓ STRONG SIGNAL

Candidate discusses CAP theorem implications, consistency vs. availability trade-offs, cost vs. performance, and build vs. buy decisions. They justify choices with reasoning, not dogma.

✕ WEAK SIGNAL

Candidate insists there are no trade-offs, cannot compare alternatives, or makes every decision based on 'that's what we used at my last job' without analysis.

Reliability & Failure Handling

Assesses whether the candidate designs for failure and considers system reliability.

Rating: □ 1 □ 2 □ 3 □ 4 □ 5

Sample Questions:

- What happens if this service goes down?
- How do you handle partial failures in distributed transactions?
- What monitoring and alerting would you set up for this system?

✓ STRONG SIGNAL

✕ WEAK SIGNAL

Candidate proactively discusses failure modes, circuit breakers, retries with backoff, dead letter queues, and graceful degradation. They design for failure, not just the happy path.

Candidate assumes nothing will fail, has no strategy for handling partial outages, or doesn't consider what happens when dependencies are unavailable.

Red Flags

- Cannot estimate order-of-magnitude capacity (QPS, storage, bandwidth) for common systems
- Designs only for the happy path with no consideration of failures
- Cannot articulate any trade-off — insists every choice they make is optimal in all dimensions

Notes & Overall Recommendation

☐ Strong Hire ☐ Hire ☐ No Hire ☐ Strong No Hire

Notes: _____