# Technical Phone Screen — Backend

Phone Screen | Backend

Assess backend skills: API design, databases, concurrency, and system thinking.

## Evaluation Criteria

### API Design & REST/GraphQL

Evaluates ability to design clean, consistent APIs and understanding of API design principles.

Rating: ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

**Sample Questions:**

- How do you design a RESTful API for a resource with complex relationships?
- When would you choose GraphQL over REST, and what are the trade-offs?
- How do you handle API versioning and backward compatibility?

✓ STRONG SIGNAL

Candidate demonstrates clear understanding of resource modeling, HTTP semantics, idempotency, and pagination. They discuss real trade-offs rather than stating one approach is always better.

✕ WEAK SIGNAL

Candidate cannot explain REST constraints beyond 'use GET and POST', confuses HTTP status codes, or has no strategy for API evolution and versioning.

### Database Design & Query Optimization

Assesses knowledge of data modeling, indexing strategies, and query performance tuning.

Rating: ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

**Sample Questions:**

- How do you decide between a SQL and NoSQL database for a new project?
- Walk me through how you'd design the schema for an e-commerce order system.
- How do you identify and fix slow database queries?

✓ STRONG SIGNAL

Candidate discusses normalization trade-offs, indexing strategies with specifics (composite indexes, covering indexes), and uses EXPLAIN plans. They can reason about when denormalization is appropriate.

✕ WEAK SIGNAL

Candidate cannot explain indexing beyond 'it makes things faster', has no strategy for schema design, or has never profiled a slow query.

### Concurrency & Distributed Systems Basics

Evaluates understanding of concurrency patterns, race conditions, and basic distributed systems concepts.

Rating: ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

**Sample Questions:**

- How do you handle concurrent writes to the same resource?
- What is a race condition, and how have you dealt with one?

- Explain the difference between optimistic and pessimistic locking.

✓ STRONG SIGNAL

Candidate explains locking mechanisms, transaction isolation levels, or queue-based approaches with concrete examples. They understand the trade-offs between consistency and availability.

✕ WEAK SIGNAL

Candidate has never considered concurrency, confuses threads and processes, or cannot describe a situation where data consistency was a concern.

### Error Handling & Observability

Assesses approach to building resilient services with proper error handling, logging, and monitoring.

Rating: □ 1 □ 2 □ 3 □ 4 □ 5

**Sample Questions:**

- How do you design error handling in a backend service?
- What's your approach to logging and monitoring in production?
- Describe how you'd debug an intermittent production issue.

✓ STRONG SIGNAL

Candidate describes structured logging, correlation IDs, health checks, and alerting strategies. They have a systematic debugging approach and understand the difference between errors, warnings, and info logs.

✕ WEAK SIGNAL

Candidate relies on console.log for debugging, has no monitoring strategy, or cannot describe how they would investigate a production incident.

### Security Fundamentals

Evaluates awareness of common backend security concerns and secure coding practices.

Rating: □ 1 □ 2 □ 3 □ 4 □ 5

**Sample Questions:**

- What are the most common security vulnerabilities you guard against in backend code?
- How do you handle authentication and authorization in your services?
- How do you manage secrets and sensitive configuration?

✓ STRONG SIGNAL

Candidate identifies OWASP Top 10 concerns naturally (SQL injection, XSS, CSRF), explains JWT vs. session-based auth trade-offs, and has a strategy for secrets management (vault, env vars, never in code).

✕ WEAK SIGNAL

Candidate has never thought about SQL injection, stores passwords in plain text, or commits secrets to version control without concern.

## Red Flags

- Cannot explain the difference between SQL and NoSQL at a conceptual level
- No awareness of authentication/authorization patterns despite claiming backend experience
- Dismisses testing and monitoring as 'DevOps responsibilities'

## Notes & Overall Recommendation

□ Strong Hire □ Hire □ No Hire □ Strong No Hire

Notes: _____