ROAMINGPIGS

Technical Due Diligence Checklist

50+ questions to evaluate startup technology before you invest. Based on real assessments across dozens of startups, from seed to Series C.

- ✓ Five-Minute Smell Test to spot red flags fast
- ✓ Architecture, code quality, and security checklists
- ✓ Red flags that kill deals vs. yellow flags to discuss
- ✓ Final assessment framework for investment decisions

Cisco Caceres

45+ years building systems. Government agencies to startups.
Technical due diligence for investors and acquirers.
roamingpigs.com | contact@roamingpigs.com

Start Here: The Five-Minute Smell Test

Quick indicators before diving deep. Any "no" warrants deeper investigation. These five questions reveal more about technical health than hours of code review.

- ☐ Can CTO explain architecture in plain English to non-technical stakeholders?
- ☐ Is there code older than 6 months? (Complete rewrites suggest thrashing)
- ☐ Do they deploy at least weekly? (Monthly = cautious or terrified of codebase)
- ☐ Can they clearly describe their last outage and what changed after?
- ☐ Do they know their deployment rollback time?

Architecture Assessment

System Design

- ☐ Architecture matches company stage (monolith for early, services only if needed)
- ☐ Can explain why they chose their architecture (not just "Netflix does it")
- ☐ Service boundaries are clean and logical (if microservices)
- ☐ No premature optimization for scale they haven't reached
- ☐ Clear data flow documentation exists

Database & Data

- ☐ Boring, proven database choices (Postgres usually wins)
- ☐ Each specialized database solves a real problem (not resume-driven)
- ☐ Data backup and recovery process documented and tested
- ☐ No custom database or data layer (unless they're a database company)
- ☐ Schema migrations are version-controlled and reversible

Dependencies

- ☐ Reasonable number of third-party dependencies
- ☐ Critical dependencies are from established vendors
- ☐ Fallback plan exists if key vendor disappears or raises prices 10x
- ☐ No obscure libraries for core business logic
- ☐ Dependencies are regularly updated (check last update date)

Code Quality

Testing

- [ ] Critical paths have test coverage
- [ ] Integration tests exist and run
- [ ] CI pipeline runs tests on every commit
- [ ] Tests actually fail when code breaks

Documentation

- [ ] README explains how to run locally
- [ ] Architecture decisions are recorded
- [ ] API documentation exists
- [ ] Documentation is current (not 2 years stale)

Code Health

- [ ] Git history shows healthy contribution patterns
- [ ] No single files that change every commit (usually broken)
- [ ] Recent refactoring has clear rationale
- [ ] Team can identify their top 3 technical debt areas

Infrastructure & Operations

Disaster Readiness

- [ ] Clear answer for: "What if primary database goes down?"
- [ ] Documented recovery time for complete data loss
- [ ] Deployment rollback tested and timed
- [ ] Credential rotation process exists
- [ ] Incident response runbook documented

Cloud & Costs

- [ ] Cloud spend is proportional to usage/revenue
- [ ] Cost per user/transaction is known
- [ ] No obvious waste (idle resources, over-provisioning)
- [ ] Growth in cloud costs won't outpace revenue

Team & Process

- [ ] Bus factor > 1 for critical systems
- [ ] Knowledge is documented, not just in heads
- [ ] New engineer productive within 2-4 weeks
- [ ] Team acknowledges technical debt (denial is a red flag)
- [ ] Clear process for security vulnerability response

Security Basics

- [ ] Secrets in environment variables, not in code repo
- [ ] Authentication and authorization logic exists
- [ ] No known-vulnerable versions of critical dependencies
- [ ] Someone has done at least one security review
- [ ] Credential rotation is possible if compromised
- [ ] Access controls exist (not everyone has admin)

Red Flags (Deal Killers)

- [ ] Fundamental scaling limitations that require complete rewrite
- [ ] Security disasters: plaintext passwords, public S3 with customer data
- [ ] Key person dependency with no mitigation plan
- [ ] Misrepresentation: claims that don't match reality
- [ ] Dangerous vendor lock-in with unfavorable terms

Yellow Flags (Need Discussion)

- [ ] Technical debt exists but team knows where it is
- [ ] Missing tests (fixable with time)
- [ ] Junior team (requires mentorship expectations)
- [ ] Unusual technology choices (may be innovative or problematic)

Final Assessment

- [ ] Can this technology support the business plan?
- [ ] What could go wrong technically, and how likely?
- [ ] Is this team capable of building what they're proposing?
- [ ] What should be addressed in the first 90 days post-investment?

Need a Professional Technical Assessment?

This checklist covers the basics. For investment decisions, M&A transactions, or partnership evaluations, a comprehensive assessment identifies risks that checklists miss.

Schedule a Consultation Read the full article