The 47-Point Technical Due Diligence Checklist

The same framework used in every engagement—yours free.

Protect your investment from hidden technical risks

Code Quality • Security • Infrastructure • Scalability • Team • Technical Debt • Data & IP

Why This Checklist Matters

Technology investments have never been riskier. In an era where AI reshapes industries overnight and technical landscapes evolve at breakneck speed, the margin for error in due diligence has narrowed dramatically. A company can look exceptional on paper—strong revenue, impressive growth metrics, charismatic founders—yet harbor critical vulnerabilities that only surface post-investment.

We've seen it happen repeatedly: promising ventures that implode not from market forces, but from three silent killers lurking in their technical foundations:

- **Lax Security Practices** — Data breaches don't just cost money; they destroy trust, invite regulatory scrutiny, and can render an entire business model untenable overnight.
- **Compliance Gaps** — In an increasingly regulated landscape, yesterday's shortcuts become tomorrow's lawsuits. GDPR, SOC 2, HIPAA—the alphabet soup of compliance isn't optional.
- **Excessive Technical Debt** — That 'move fast and break things' mentality often leaves behind a codebase that's expensive to maintain, impossible to scale, and resistant to the very innovation that made the company attractive.

This is why we developed this 47-point checklist. It's the exact framework we use in every technical due diligence engagement—battle-tested across hundreds of evaluations, refined through hard-won lessons, and designed to surface the risks that matter before they become your problem.

How to Use This Checklist

Each item represents a critical checkpoint in technical evaluation. Use this as your roadmap during due diligence—whether you're conducting the assessment yourself, briefing your technical advisors, or evaluating reports from third-party auditors. Not every item will apply to every company, but every item deserves consideration.

1. Code Quality & Architecture

[ ] 1. Codebase uses consistent coding standards and style guidelines

[ ] 2. Architecture documentation exists and reflects current system state

[ ] 3. Clear separation of concerns (frontend, backend, data layer)

[ ] 4. Appropriate use of design patterns without over-engineering

[ ] 5. Code review process is established and consistently followed

[ ] 6. Automated code quality tools (linters, static analysis) in place

[ ] 7. Test coverage meets industry standards (unit, integration, e2e)

[ ] 8. Technical documentation is current and accessible

2. Security & Compliance

[ ] 9. Security audit completed within the last 12 months

[ ] 10. Vulnerability scanning integrated into CI/CD pipeline

[ ] 11. Authentication system follows current best practices (MFA, OAuth 2.0)

[ ] 12. Data encryption at rest and in transit

[ ] 13. Access controls follow principle of least privilege

[ ] 14. Incident response plan documented and tested

[ ] 15. Compliance certifications current (SOC 2, GDPR, HIPAA as applicable)

[ ] 16. Third-party security dependencies regularly updated

3. Infrastructure & DevOps

[ ] 17. Infrastructure as Code (IaC) implemented

[ ] 18. CI/CD pipeline fully automated

[ ] 19. Environment parity (dev, staging, production)

[ ] 20. Monitoring and alerting systems operational

[ ] 21. Disaster recovery plan documented and tested

[ ] 22. Backup systems verified with regular restore testing

[ ] 23. Cloud resource costs tracked and optimized

4. Scalability & Performance

- [ ] 24. Load testing performed with documented results
- [ ] 25. Database queries optimized and indexed appropriately
- [ ] 26. Caching strategy implemented where beneficial
- [ ] 27. Horizontal scaling capability demonstrated
- [ ] 28. Performance benchmarks established and tracked
- [ ] 29. CDN and edge computing utilized where appropriate

5. Team & Engineering Processes

- [ ] 30. Engineering team retention rate above industry average
- [ ] 31. Knowledge distribution—no single points of failure
- [ ] 32. Onboarding documentation enables rapid contributor ramp-up
- [ ] 33. Agile/development methodology consistently applied
- [ ] 34. Technical decision-making process is documented
- [ ] 35. Post-mortem culture exists for learning from incidents

6. Technical Debt & Risk Assessment

- [ ] 36. Technical debt inventory maintained and prioritized
- [ ] 37. Legacy system migration plan exists (if applicable)
- [ ] 38. Dependency audit completed—no abandoned libraries
- [ ] 39. License compliance verified for all third-party code
- [ ] 40. End-of-life technology identified with replacement timeline
- [ ] 41. Refactoring roadmap aligned with business priorities

7. Data & Intellectual Property

- [ ] 42. Data architecture documented with clear ownership
- [ ] 43. Data retention and deletion policies implemented
- [ ] 44. IP ownership clear for all code and algorithms
- [ ] 45. Proprietary technology properly protected (patents, trade secrets)

[ ] 46. Data lineage traceable for regulatory compliance

[ ] 47. Machine learning models (if any) documented with training data provenance

Next Steps

This checklist is your starting point, not your finish line. Technical due diligence is both an art and a science—it requires not just checking boxes, but understanding context, asking the right follow-up questions, and knowing which risks are dealbreakers versus manageable concerns.

- **Score your findings** — Consider a simple Red/Yellow/Green rating for each item to quickly visualize overall technical health.
- **Prioritize deep dives** — Not all items carry equal weight. Security and compliance issues typically warrant immediate attention; code quality concerns may be acceptable with a remediation plan.
- **Engage specialists** — For areas outside your expertise, bring in domain experts. A cloud infrastructure review requires different skills than a security audit.
- **Document everything** — Your due diligence findings become part of the investment thesis. Clear documentation protects all parties and informs post-investment priorities.

"The best time to discover a technical liability is before you write the check."

For professional technical due diligence services, contact us at your convenience.