TECH DEBT ONBOARDING GUIDE

What Every New Team Member Should Know
About Our Technical Debt

For Engineering Managers and New Hires

WHAT'S INSIDE

- Why we talk about tech debt openly
- Current debt landscape and tracking
- Your first 30 days navigating debt
- Our processes and tools for managing debt
- Quick reference decision tree

TechDebt.fail
https://techdebt.fail/

Why We Talk About Tech Debt Openly

Every codebase has technical debt. Acknowledging it is a sign of engineering maturity, not failure. This guide helps you understand our current debt landscape, how we manage it, and how you can contribute to keeping our codebase healthy.

Our Tech Debt Philosophy

- We track debt intentionally, not accidentally
- We allocate sprint capacity for debt reduction
- We never shame the original authors - context changes
- We document decisions, even (especially) when we choose to defer
- We measure progress with metrics, not feelings

Our Current Debt Profile

Known Debt Areas

Area 1: _______________ | Status: ___ | Owner: ___

Area 2: _______________ | Status: ___ | Owner: ___

Area 3: _______________ | Status: ___ | Owner: ___

Area 4: _______________ | Status: ___ | Owner: ___

Area 5: _______________ | Status: ___ | Owner: ___

Active Remediation Efforts

Project 1: _______________ | Timeline: ___ | Contact: ___

Project 2: _______________ | Timeline: ___ | Contact: ___

Project 3: _______________ | Timeline: ___ | Contact: ___

Note: *This section should be filled in by your manager or tech lead during onboarding.*

How to Navigate Debt as a New Team Member

Week 1: Observe and Learn

* Read the architecture decision records (ADRs)
* Ask "why is it this way?" without judgment
* Identify the patterns used in your area of the codebase
* Note things that surprise or confuse you (fresh eyes are valuable)

Week 2-3: Start Contributing

* Follow the Boy Scout Rule: leave code a little better than you found it
* Add tests for code you touch that lacks coverage
* Update documentation that was wrong or missing
* Ask before refactoring - context matters

Week 4+: Engage Actively

* Share your "fresh eyes" observations with the team
* Propose debt items for the backlog with impact estimates
* Volunteer for debt reduction stories in sprint planning
* Help document tribal knowledge you had to learn the hard way

Our Processes and Tools

How We Track Debt

* Tracking system: ________________ (Jira, Linear, etc.)
* Label/tag convention: ________________
* Priority levels: P1 (security), P2 (blocking), P3 (slowing), P4 (cosmetic)
* Review cadence: ________________ (monthly, quarterly)

How We Prioritize

* Impact/effort matrix for scoring
* Security and compliance items are always P1
* Customer-facing impact trumps internal convenience
* Team consensus for borderline items

How We Communicate

* Sprint planning: present debt items alongside features
* Retrospectives: discuss debt-related blockers
* Monthly report to leadership on debt metrics

Cheat Sheet: What to Do When You Find Debt

Decision Tree

Is it a security vulnerability?
-> Flag immediately, P1

Can you fix it in < 30 minutes?
-> Fix it now (Boy Scout Rule)

Does it block your current work?
-> Create ticket, discuss in standup

Is it annoying but not blocking?
-> Add to debt backlog with notes

Is it a major architecture issue?
-> Document in ADR, discuss with tech lead

Key Contacts

Tech Debt Owner: ________________

Architecture Lead: ________________

Your Manager: ________________

Full resources at
https://techdebt.fail/