(Credit: Jenda Kubeš/Pexels)

There’s no excuse for technical debt – here’s why

Formula One. Race day. As the cars scream around the circuit, one driver finally gets the call: it’s time for a pit stop. They’re frustrated, of course, but they know if they want to survive the remaining laps (let alone win the race), their car needs servicing.

If only every tech leader thought like an F1 driver.

Technical debt (tech debt) is what forces companies to make a pit stop when pushing their product to market. Your software and hardware need servicing. The longer you wait to do this, the bigger problems you’ll have to deal with down the line.

Many companies are waiting too long. 90% of chief technical officers see tech debt as their biggest challenge right now and companies waste up to 42% of their time trying to keep on top of it.

In the end, there's no excuse for incurring tech debt. Forget the trade-offs, the management plans, or the glowing financial loan analogies. Tech debt is something you should routinely prevent, contain, and mitigate if you want your product to cross the finish line.

What is technical debt?

Like the wheels on a race car, your software naturally wears down over time as you develop new products and scale your company. The faster you move (by taking shortcuts during development, for instance), the sooner your software will need servicing. This 'wear and tear' is known as **technical debt**.

The more tech debt you incur, the more time, money, and resources you'll have to spend fixing it in the future. Ignore your tech debt for too long and your 'pit stop' may force you to hire experts, upgrade your tech stack, or even halt progress on innovative, profit-making projects.

Where does technical debt come from?

Although wear and tear in software development is natural, there are several key culprits that cause tech debt to pile up more quickly than usual. These include:

Poor planning

Major tech debt build-ups are almost always a result of management making the wrong decisions early on – whether that's due to them lacking an understanding of strong agile methodologies, skipping over technical knowledge gaps, or rushing an MVP to market before it's actually ready. If you have no clear way of defining whether code is finished and ready to release, for instance, then you risk delivering a product that's buggy, unstable, and difficult to maintain.

Weak documentation

Good quality control is key to avoiding tech debt. Detailed documentation reduces the chance that developers write poor quality, confusing, and messy code that future employees struggle to understand. Without it, engineers won't know how to fix bugs, figure out edge cases, or find ways to scale existing architecture.

Over-reliance on manual processes

Manual processes are slow and prone to human error. If you're relying too much on your employees and not implementing enough automation (for code review, testing, and generally speeding up workflows), you're bound to be slowed down by tech debt.

Vendor lock-in

Proprietary software is convenient, but it stifles your ability to innovate or pivot to market changes without incurring significant time and resources. This makes you easy prey for tech debt build-ups.

Legacy tech stack

Redundant tools, frameworks, libraries, codebases, and databases lead to compatibility issues, bloat your system, and confuse employees, making it harder to avoid or remediate tech debt.

Why you can’t ‘manage’ technical debt

Traditional wisdom says you can separate tech debt into two camps: managed and unmanaged (or intentional and unintentional). It goes that incurring technical debt on purpose is fine as long as it’s a considered business decision that suits your short term goals and offers a reasonable long-term repayment plan.

This is a giant pitfall.

The thing is, even if you’ve documented your tech debt and drawn up a comprehensive remediation strategy, there’s still no surefire way to contain it.

Codebases evolve rapidly, tech debt documentation (if it exists) is often long and complex, which leaves plenty of room for human error, and ultimately, no matter how ‘managed’ your tech debt is, it can always be quickly lost or forgotten in the shadow of a new client or project.

While there is a case for emergencies, such as pushing last-minute patches, incurring tech debt *on purpose* is a slippery slope. It’s highly unpredictable and it can surprise you at any moment during the vital stages of your company’s growth.

The hidden costs of technical debt

Whether you can't see it or you're ignoring it on purpose, tech debt is a ticking time bomb. Here's why.

1. Tech debt **blocks innovation**

If you're incurring tech debt, you'll eventually be forced to make a pit stop – whether that's in the form of a mammoth refactoring project, rapid hiring spree, or expensive tech stack upgrade. This all eats up precious time and money, preventing you from stepping in time with the market, working on (or investing in) new software, and staying agile enough to take on more ambitious projects.

What will you do if a new mega contract lands on your desk? Could your software scale to meet its demands? If tech debt is ruling your life, it'll be near-impossible to onboard clients that could take your company to the next level. You may even have to turn down existing clients who come to you with bigger projects, sending them to your competitors instead.

2. Tech debt **drives away innovative people**

Tech debt is a leading cause of low productivity and dissatisfaction among developers. Devs want to work with clean code and modern, scalable systems, and they want to *innovate*. Give them a soul-crushing tech debt mountain climb and sprinkle some of the usual business pressures on top, like stakeholder demands and tight deadlines, and you'll lose passionate, talented employees who only widen the knowledge gap when they leave.

3. Tech debt **leads to data breaches**

Symptoms of tech debt such as complex code and outdated legacy software lead to vulnerabilities that hackers can exploit to infiltrate your network, steal critical data, and extort large amounts of money. That's not to mention the legal and reputational consequences of a successful attack. Ultimately, good cybersecurity relies on visibility. If your teams don't understand your software architecture, they likely can't see where it's exposing sensitive resources or where certain users or machines are behaving suspiciously. They could even accidentally damage the system themselves from within and leak sensitive information.

4. Tech debt **hurts your reputation**

You can't hide your tech debt forever. Eventually, it'll bleed through into the front end of your software as bugs, defects, and downtime, and it could prevent your customer support teams from resolving issues. Tech debt can even reach critical mass and cause your systems to fail at key moments like launch events (a true story). None of this is great news for your reputation, though it will be for your competitors.

Is it time for your pit stop?

When the pit crew spots a problem, they don't wait for the driver to make ten more laps of the race course – they pull them in. Tackling tech debt requires the same mentality.

It's entirely possible to deliver value in every sprint without incurring any tech debt. Rather than accepting tech debt as 'manageable' and pushing through with your fingers crossed, the key is to foster good engineering habits and consistently fix your software as you scale. This way, you'll avoid the nasty tech debt pile-ups and constant firefighting, keeping your business in the race at all times.

We'll help you tackle technical debt

We'll perform a deep review of your tech stack and operations to locate technical debt and then provide you with a high-impact remediation plan – all with minimal disruption to your business. Get in touch today and schedule a call to get started.

About us

HI is a technology and innovation partner empowering tech-first businesses, consumer brands, and consultancies to pioneer, evolve, and grow through engineering, resourcing, and white labelled solutions.