



THE MODERNIZATION GAP

What Legacy Applications
Are Costing Your Business

Table of Contents

➤	Executive Summary	01
➤	The Hidden Cost of Standing Still	01
➤	Why Modernization Gets Delayed Anyway	02
➤	What Modernization Actually Looks Like	02
➤	Proof in Practice	03
➤	A Quick Framework for Assessing Your Own Gap	03
➤	Where to Start	04



Executive Summary

Legacy applications don't send you a bill. There's no line item for the risk of an unpatched framework, or for the feature that shipped a quarter late because two engineers had to relearn a system nobody had touched in years. The cost shows up in pieces: support tickets, stalled hiring, deals that go to a faster competitor.

This ebook covers where that cost hides, why companies keep paying it instead of fixing it, and what a realistic path to modernization looks like, based on Forgeahead's work as an AWS Advanced Tier Services Partner.

Legacy costs build up as a slow tax on speed, security, and talent, paid quietly every quarter until someone adds it up.

The Hidden Cost of Standing Still

Legacy systems wear an organization down gradually in four consistent ways.



Maintenance drag

Old systems eat a disproportionate share of engineering time; patching undocumented dependencies, working around abandoned frameworks, testing by hand because there's no pipeline. Teams that spend 40% or more of a sprint on upkeep like this aren't unusual; they're just not tracking it as a line item, so leadership never sees it as a decision they're making. Every hour there is an hour not spent building what the business asked for.

Security exposure

Unpatched dependencies and configurations nobody fully understands are doors left open. Compliance frameworks like SOC 2, HIPAA, and PCI DSS keep adding requirements that legacy systems were never built to meet, and retrofitting security into old architecture is slower and more expensive than building it in from the start. Security teams end up managing a risk they can name but can't fully close.

Talent friction

Engineers want current tools. Hiring for a decade-old stack takes longer and pays a premium, since fewer people want that experience on their resume. The engineers who do understand the system well are also the ones best positioned to leave, which means the organization's institutional knowledge sits with a small, shrinking group.

Opportunity cost

Often the largest of the four, and the easiest to miss. While your team keeps the lights on, a competitor on modern infrastructure ships faster and gets AI features into their product first. It shows up gradually: a slightly slower release cadence, a slightly longer sales cycle, a market position that's noticeably weaker after a few years of it.

Why Modernization Gets Delayed Anyway

Most leaders know their legacy footprint is a liability. It gets pushed to next year anyway, for reasons that hold up fine in isolation:



- **Fear of disruption**

A payments platform can't go down, and a failed migration is a visible, career-affecting mistake in a way that a slow, expensive status quo never quite is.

- **Unclear scope**

Without a real map of the environment, it's hard to price the work or promise a timeline, so the project stays a "someday" line on a roadmap.

- **Budget structure**

Operating budgets fund keeping the lights on; transformation competes for a separate, harder-to-win pool of money that has to be justified every cycle.

- **No sequencing plan**

Teams know the destination, not the steps, so the project stalls before it starts, or gets scoped so big that nobody wants to own it.

Each of these is fixable on its own, which is why the right response is treating modernization as a sequence of smaller, assessed moves instead of one large, high-risk project.

What Modernization Actually Looks Like

Modernization done well is a sequenced process that starts with visibility and moves in stages the business can actually absorb.



Assess first

An AWS Well-Architected review checks the environment against operational excellence, security, reliability, performance, and cost. That produces a prioritized list of real risk, ranked by what actually matters, instead of a general sense that things feel outdated. Most engagements start here because it's the only way to scope the rest of the work accurately.

Migrate in sequence

The highest-risk or highest-ROI systems move first. Everything else follows in an order built around keeping the business running, so core operations don't have to pause while the architecture changes underneath them.

Let AI take the grunt work

Agentic AI tools now handle a real share of modernization work: code analysis, dependency mapping, scaffolding replacement components. That's cut timelines that used to run years down to months in several recent engagements, though it doesn't replace the judgment calls about what to migrate first and why.

Proof in Practice

Forgeahead has run this playbook across fintech, life sciences, and ed-tech. Three results from recent engagements:



- **40%** less downtime for a payments platform, after automating CI/CD and monitoring
- **35%** lower cloud costs for an ed-tech research platform, after an AWS infrastructure optimization
- **99.99%** uptime for a life sciences data platform, after a DevOps-led cloud modernization

Each of these started with an assessment of the existing environment, moved through a sequenced modernization plan, and ended with infrastructure the client's own team could operate and extend going forward, without staying dependent on the team that built it.

A Quick Framework for Assessing Your Own Gap

Answer these honestly and the size of your gap gets obvious fast:



1

How much engineering time went to maintenance versus new development last quarter?

2

Which systems run on frameworks that are no longer actively supported?

3

How long would a new hire need to become productive on your oldest production system?

3

What's the last feature a competitor shipped that your architecture would struggle to support?

4

If your most senior engineer on a legacy system left tomorrow, who else understands it?



Where to Start

The gap between what a legacy application costs and what it delivers only grows with time. Closing it takes an accurate picture of where the risk actually sits and a sequenced plan to address it, built around what the business can and can't afford to pause and not necessarily a full replatform on day one.

Forgeahead's AWS Expert Assessment gives enterprise teams that picture: a clear view of legacy risk, cost, and a modernization roadmap sequenced around business continuity.

forgeahead 



**Schedule an Expert
AWS Assessment**



www.forgeahead.io



+1-650-948-1787