DataShelf Hub

CLOUD & RELIABILITY

Shipping with Confidence

CI/CD, feature flags and deploys that don't need a prayer

FREE SAMPLE CHAPTER

Omar El Alaoui
Data & systems engineer

PDF · CH. 01

ABOUT THIS SAMPLE

A free first chapter

What follows is Chapter 1 of *Shipping with Confidence*, reproduced in full and provided at no cost by DataShelf Hub. It is unabridged — nothing has been trimmed or summarised. If it's useful to you, the remaining seven chapters continue exactly where this one leaves off.

This sample is for your personal use while you decide whether the book is right for you. Please don't republish or redistribute it — share a link to the book's page instead.

TITLE	Shipping with Confidence
AUTHOR	Omar El Alaoui
SHELF	Cloud & Reliability
FORMAT	PDF, DRM-free, 254 pages
THIS SAMPLE	Chapter 1 of 8 — 14 pages
PUBLISHER	DataShelf Hub · datashelfhub.com

WHAT'S IN THE FULL BOOK

Table of contents

8 chapters · 24 sections

Continued on the next page →

TABLE OF CONTENTS, CONTINUED

Chapters 5–8

CHAPTER ONE

01

Making Deploys Boring

Deploying shouldn't feel like a decision you have to work up the nerve to make — and if it does, that's a design problem, not a personality problem.

Someone on the team says "I'll deploy that on Monday, don't want to do it Friday afternoon," and everyone nods, because everyone has the same instinct. Nobody questions it. But sit with that instinct for a second: the code is done, it's reviewed, it's tested, it's ready. The only reason to delay is a quiet, well-earned belief that deploying it might break something, and that if it does, fixing it will take longer than anyone wants to spend on a Friday evening. That belief isn't paranoia. For most teams, it's an accurate read of their own pipeline.

This chapter is about why that fear exists and, more usefully, about the specific engineering that makes it go away — not by getting braver, but by making a bad deploy cheap, fast, and narrow enough in its blast radius that nobody needs courage to ship on a Friday, or any other day. Confidence isn't a mindset here. It's a property of the pipeline, and like any property of a system, it's built, not willed into existence.

1.1 Why Releases Feel Scary

Fear of deploying is almost always a rational response to one of a small number of concrete problems, not a vague anxiety. Either the pipeline is slow, so a mistake takes a long time to even detect. Or the blast radius is large, so a bad change reaches every user at once instead of a small fraction. Or rollback is slow or manual, so once something's wrong, fixing it takes longer than anyone's comfortable with. Usually it's some combination of all three.

None of those are personality problems, and none of them are fixed by asking engineers to be braver about shipping. They're fixed by changing the actual mechanics: making detection fast, making blast radius small on purpose, and making rollback fast enough that a bad deploy is a minor, boring event instead of the start of an incident.

> *"Confidence isn't a feeling you build up before pressing deploy. It's what you have left over once a bad deploy genuinely can't do much damage before you notice and undo it."*

This reframing matters because it points at concrete engineering work instead of vague culture change. You can't tell a team to "just be more confident" and expect Friday deploys to feel different. You can build a pipeline where a bad change reaches one percent of traffic, gets caught by an automated check within ninety seconds, and rolls back without a human needing to page anyone at all — and that pipeline changes how the team feels about shipping whether or not anyone talks about confidence at all.

It's worth being honest that this isn't free. Fast detection, small blast radius, and fast rollback are all real engineering investments — in your CI pipeline, in your deployment strategy, in how your application is built to support partial rollouts. This book is that investment, chapter by chapter. But it's an investment with an unusually direct payoff: every piece of it makes the next deploy less stressful, in a way the whole team feels immediately.

THE THREE LEVERS THAT ACTUALLY REDUCE DEPLOY FEAR

Fast detection	how quickly a bad change is noticed — seconds, not hours
Small blast radius	how many users a bad change reaches before it's caught
Fast rollback	how quickly a bad change can be undone once found

Every remaining chapter maps onto one of those three levers. Chapter 2's fast, trustworthy CI and Chapter 7's testing-in-production techniques improve detection. Chapter 5's canary and progressive delivery strategies shrink blast radius on purpose, by design, rather than by luck. Chapter 5 and Chapter 8 both cover rollback — fast, automatic where possible, and never a manual scramble through runbooks nobody's tested since they were written.

None of these levers require heroics. They require deciding, in advance, what "safe to ship" actually means for your system, and building the pipeline that enforces it automatically — which is a very different thing from trusting everyone to be careful, every time, forever.

1.2 Deploy Versus Release

A huge amount of deploy anxiety comes from a single conflated idea: that deploying code and releasing a feature to users are the same event. They don't have to be, and separating them is one of the highest-leverage ideas in this entire book — the subject of Chapter 4 in full, but worth introducing now because it reframes everything that follows.

Deploying is getting new code running in production. **Releasing** is deciding that code is now active and visible to users. Conflate them, and every deploy is automatically a release — high-stakes, all-or-nothing, exactly the kind of event that earns a team's fear. Separate them, and deploying becomes routine: new code ships behind a flag, inert, running in production but doing nothing different yet.

THE SAME CHANGE, TWO DIFFERENT EVENTS

Deploy	new code reaches production, flag off, behaviour unchanged — can happen any day, any time
Release	flag flipped on, for 1% of traffic, then 10%, then everyone — a deliberate, reversible decision

Once deploy and release are separate, most of the Friday-afternoon anxiety simply evaporates — you can deploy Friday afternoon, because nothing user-visible changes until someone deliberately flips a flag, on a day and at a time the team chooses on purpose, with the ability to flip it back in seconds rather than needing a rollback at all.

WORTH REMEMBERING

A rollback undoes a deploy. Turning off a flag undoes a release. The second one is almost always faster, safer, and available to more people on the team — which is exactly why decoupling the two is worth the upfront investment.

1.3 A Pipeline Overview

With those two ideas in hand — fear traces to three concrete, fixable levers, and deploy doesn't have to equal release — the rest of this book is a tour of the pipeline that puts them into practice, roughly in the order code travels through it. It starts with CI, because nothing downstream matters if the thing you're about to promote to production hasn't been verified fast and reliably.

From there: building one immutable artifact and promoting the identical thing through every environment, so "it worked in staging" actually means something. Feature flags, to decouple deploy from release properly. Progressive rollout strategies, to make blast radius a design decision instead of an accident. Database migrations, handled with the same care, because a schema change that isn't backward-compatible can undo every other safeguard in one step.

The book closes with testing in production — treating your live system as a source of verification, not just a destination for changes — and with what to do on the days it still goes wrong anyway, because it sometimes will, and a mature pipeline is judged as much by how gracefully it recovers as by how rarely it fails.

None of this is about eliminating risk entirely. It's about making the remaining risk small, fast to detect, and fast to undo — which, it turns out, is the actual definition of a team that can ship on a Friday afternoon without anyone needing to feel brave about it.

Chapter 2 starts at the beginning of the pipeline: what makes CI fast enough and trustworthy enough that everything built on top of it — the rest of this book — actually holds.

§ Chapter Summary

Seven things worth carrying into Chapter 2.

BEFORE YOU MOVE ON

01	Deploy fear traces to three concrete, fixable problems — slow detection, large blast radius, slow rollback — not to a lack of nerve.
02	Confidence is what's left over once a bad deploy genuinely can't do much damage before it's caught and undone.
03	Deploying and releasing don't have to be the same event — separating them turns most deploys into routine, low-stakes events.
04	A rollback undoes a deploy; turning off a feature flag undoes a release, and it's almost always faster and safer.
05	The whole pipeline — CI, artifacts, flags, progressive rollout, migrations — exists to make the three fear-reducing levers real, in order.

BEFORE CHAPTER 2

Think about the last deploy your team was nervous about. Which of the three levers — slow detection, large blast radius, or slow rollback — was actually the reason? Naming it precisely is the first step toward fixing the real problem, which Chapters 2 through 5 do directly.

END OF SAMPLE

Seven chapters still to go.

Chapter 1 names what actually causes deploy anxiety and introduces the deploy/release split; the rest of the book builds the pipeline that makes shipping routine — trustworthy CI, immutable artifacts, feature flags, progressive rollout, safe migrations, and graceful recovery when a release goes wrong anyway.