🚨 2-Week Project Recovery Playbook
Real Systems for When Things Break Mid-Sprint

When your developer disappears or your sprint derails, you don't need panic — **you need a plan.**

This playbook gives you the exact 2-week recovery framework we use internally to rebuild momentum fast: scope, code, and cadence.
It's the same process founders at Reddit, Goldman Sachs, and multiple seed-stage startups have used to ship testable products after chaos.

🕓 72-Hour Containment Plan (Pre-Week)

Before touching code, stop the bleeding. This first stage stabilizes your project and protects what's left.

Checklist

- Freeze your repo — revoke all old credentials immediately.
- Export backlog — mark what's done, half-done, untouched.
- Define your "minimum viable ship" — the smallest version that proves value.
- Run a quick Risk Ledger (template link) to list unknowns, dependencies, blockers.

Friend tip: Don't assume "it's fine" until you've seen every commit and permission. Containment saves weeks later.

WEEK 1 — Rebuild Scope and Stability

Day 1–2: Audit

Review the entire codebase and documentation.
Tag features as usable, broken, or unknown.
Map dependencies (APIs, plugins, environment).
Create a one-page summary: "What exists / What's missing."

Day 3–4: Define a New Scope (1-Pager)

One activation path only (the first thing users must do).
3–5 "must-have" features that can be completed in 10 days.
Define measurable acceptance criteria (e.g., "User can register and submit one booking").
Cut or delay all else.
Deliverable: Your new Scope 1-Pager, ready to brief any developer in minutes.
Use Drexus' format: activation path + acceptance + cut-lines + non-functional targets.

Day 5–7: Reset Cadence

Implement a Friday Progress Receipt ritual:

What shipped ✅
What's in progress ⚙️
What's next ➡️
Known risks ⚠️

Setup daily check-ins (15 min) or async Slack updates.
Confirm ownership of code, environments, and documentation.

Goal by end of Week 1:

You have a stable base, a new scope, and visible rhythm. **Nothing is "vague" anymore.**

WEEK 2 — Ship, Demo, and Regain Control

Day 8–10: Execution Sprint

Rebuild or fix broken flows first (not new features).

Prioritize usability over polish.

Implement analytics or error tracking (Firebase, Sentry, Mixpanel).

Document each fix and decision.

Day 11–13: QA and Demo Prep

Test every core flow from the new scope.

Run a small internal or closed beta.

Record one demo video or host a live walkthrough.

Prepare your second Friday Progress Receipt showing measurable progress.

Day 14: Handoff and Next Steps

Deliver a clean repo, documentation, and credentials list.

Review what was learned and what remains open.

Decide whether to extend to next sprint, onboard new devs, or pause.

Celebrate progress — chaos is now data.

📊 Output Deliverables

By Day 14, you should have:

✅ A stable, working product slice

✅ Documented code ownership and environment access

✅ Clear scope for next sprint

✅ Transparent progress receipts

✅ Renewed confidence with investors or internal stakeholders

🚫 Common Mistakes to Avoid

❌ Rewriting everything — fix what works first.

❌ Skipping documentation — future you will hate that.

❌ Letting emotion drive priorities.

❌ Ignoring communication structure — cadence > code quality in recovery.

❌ Not defining “done” clearly — if it’s not testable, it’s not done.

🧠 Tools Mentioned

Risk Ledger: Identify unknowns and blockers fast.

Scope 1-Pager: Redefine your project with clarity and cuts.

Friday Progress Receipt: Replace panic with rhythm.

2-Week Recovery Template: Combine all of the above into one repeatable framework.

💬 Final Word

You can’t control who ghosts you, but you can control your system.

Run this playbook once, and you’ll never lose a sprint again.

“Recovery isn’t about code, it’s about visibility.
When you can see progress, you can fix anything.”

FROM
YOUR FRIENDS
@ DREXUS