What to Do When Your Only Developer Quits (First 90 Days)

0) FIRST 24 HOURS, CONTROL THE BLEEDING

- ❑ Pause non-critical changes (new features, “quick fixes,” refactors)
- ❑ Confirm who is decision owner for this plan (one person)
- ❑ Book a 60–90 min “Systems Risk Huddle” within 48 hours
- ❑ Create a shared folder: **“System Rescue – 90 Days”** (company-owned)

1) WITHIN 48 HOURS, SYSTEMS RISK HUDDLE (TRIAGE)

Attendees:

- ❑ Departing developer
- ❑ Ops owner (daily user / process owner)
- ❑ IT lead (if any)
- ❑ Decision-maker

Deliverables (must leave the meeting with these):

- ❑ List every system/app touched weekly (even “temporary” tools)
- ❑ Rank top 5 systems by business impact if down
- ❑ Identify single points of failure (servers, scripts, credentials, laptops)
- ❑ Create a **Risk Register** (template below)

Questions to answer (write down):

- ❑ What breaks most often?
- ❑ What breaks worst (highest impact)?
- ❑ What only you know how to fix?
- ❑ What runs on a personal machine?
- ❑ What has no tested restore?

2) WEEK 1, SECURE OWNERSHIP (CODE / DATA / ACCESS)

Code & repos

- ❑ Move all code into a company-owned Git org/repo
- ❑ Confirm **2+ admins** have access
- ❑ Export a repo backup (mirror/zip) to company storage
- ❑ Document “how to run/build” in 10 bullets max

Servers & environments

- ❑ Inventory environments: dev / test / prod
- ❑ List where apps run (server name, host, VM, cloud account)
- ❑ Capture configs + scheduled jobs (Task Scheduler/Cron/Services)

Credentials & access

- ❑ Inventory all production credentials (DB/app/server/cloud)
- ❑ Move credentials to company password manager/vault
- ❑ Remove personal accounts from critical access paths
- ❑ Rotate production credentials (planned)

3) WEEK 1–2, BACKUPS THAT ACTUALLY RESTORE

- ❑ Identify “must restore” assets:
  - ❑ Databases
  - ❑ App binaries/builds
  - ❑ Configs/secrets
  - ❑ Scheduled jobs/scripts
  - ❑ File shares / imports / exports

- ❑ Confirm backup frequency + retention + storage location
- ❑ Perform a **test restore** into a safe environment
- ❑ Record restore steps + time + what failed

- ❑ Define:
  - ❑ RPO (max data loss allowed)
  - ❑ RTO (max downtime allowed)

4) WEEK 2–4, CAPTURE “HOW IT REALLY WORKS” (LEAN DOCS)

One-page system map (required)

- ❏ Components (UI, DB, scripts, integrations)
- ❏ Data flows (what talks to what)
- ❏ Where logs live / where failures show up
- ❏ Vendor accounts/licenses/certs dependencies

Top 10 issues list (required)

For each:

- ❏ Symptom users see
- ❏ Likely cause
- ❏ First place to check (log/table/service/script)
- ❏ Quick workaround (if any)

Safe change / deployment checklist (required)

- ❏ Pre-checks
- ❏ Steps in order
- ❏ Rollback steps
- ❏ “Never do this” list

Recordings (high value)

- ❏ Record 2–5 walkthrough sessions (30–60 min)
- ❏ Label recordings clearly (workflow + date + topic)
- ❏ Store in shared folder + link from system map

5) WEEK 3–6, STABILIZE BEFORE ANY “REBUILD”

- ❏ Add basic monitoring/alerts (uptime, disk space, job failures)
- ❏ Add logging where it’s missing (top failure areas)
- ❏ Add input validation for known bad entries
- ❏ Reduce corruption risk (concurrency rules, lock handling, guardrails)
- ❏ Create a safe test environment that resembles production

6) WEEK 4–8, SAFE HANDOFF (NEW HIRE OR EXTERNAL TEAM)

Create a “Starter Pack” folder:

- ❑ System map (1 page)
- ❑ Top 10 issues list
- ❑ Deployment/maintenance checklist
- ❑ Restore procedure notes
- ❑ Access guide (who grants what)
- ❑ Recordings + links

Onboarding rules:

- ❑ First 2 weeks = observation + low-risk work only
- ❑ No direct production changes without review/checklist
- ❑ Start with: small bug fixes, non-critical reports, tests, logging

7) WEEK 6–12, KILL THE “SINGLE HERO” PATTERN

- ❑ Assign an ops owner + technical owner for each critical system
- ❑ Require 2-person knowledge coverage for critical workflows
- ❑ Schedule monthly 30-min knowledge transfer (recorded)
- ❑ Store all artifacts in one place (searchable, company-owned)
- ❑ Add a quarterly “restore test” to the calendar

RISK REGISTER

System/App:	
Business owner:	
Technical owner:	
What it supports:	
Where it runs:	
If it fails, impact is:	
Single points of failure:	
Backups exist? (Y/N):	
Restore tested? (Y/N):	
Access risk (one person only)? (Y/N):	
Dependencies (vendors/certs/licenses):	
Next actions (3 max):	