ALGOSHRED TECHNOLOGIES

DEVELOPER PLATFORMS · HELPING APP BUILDERS WAIT LESS

Let your app builders build, not wait.

We help you build one ready, shared setup for the people who build your apps: ready-made starting points, a help-yourself tool wall and the same automatic checks on every change, so routine requests no longer have to wait in a queue and risky ones still get a person's yes.

Worth knowing: It's a journey, not a quick fix: we build it with your teams, one lane at a time. A platform helps only when it is the easier way for the people using it, so we build it with them.

Get the pit ready. Let them drive.

Start by following one change: email contact@algoshred.com with what your team builds and where work seems to wait, and we'll suggest where to start.

Platform & developer engineering as a service
Vignesh T.V., CEO & CTO, Algoshred Technologies Pvt. Ltd.

contact@algoshred.com

The problem, and the idea.

- Our best people spend more time waiting on requests than building.
- Every team set up its own tools, and none of them match.
- Release day is a big, nervous event.
- Something broke after the update, and nobody could tell which change did it.
- New joiners spend their first weeks asking who to ask.
- We bought a shiny new tool, and the waiting stayed the same.

> **A developer platform is one ready, shared setup for the people who build your apps, so they spend their day building instead of asking, waiting and setting things up from scratch.**

The story, decoded. In our story, Kofi and the other drivers are your app builders (developers), the race is getting work to your customers, Lento is the team that runs your shared tools, the ready pit is the developer platform, the bright, painted lane is your team's usual way to build and release, the tool wall is where everyday tools are ready to take, and the check gate is the automatic checks every change goes through.

01 One ready pit for every team

Instead of each team building its own tools, one crew keeps a shared set ready and looked after, and listens to the people who use it.

Experts call this: internal developer platform (IDP)

Worth knowing A platform helps only when it is the easier way for the people using it, so we build it with them.

02 A painted lane for the common jobs

For the jobs you do most, the quick way and the careful way become the same way: a ready-made start with the checks already in it.

Experts call this: golden paths, paved roads

Worth knowing The painted lane is a recommendation, not a rule. Teams can step off it for unusual work.

03 Help yourself, within the lines

Routine things hang on the wall for anyone to take, and the wall records who took what.

Experts call this: self-service with guardrails, developer portal

Worth knowing Everyday tools: no form, no queue. Risky tools still need a person's yes.

04 The same checks on every change, in small steps

Each change is built and tested automatically and can go out in small steps, to a few customers first where that makes sense, with a way back for most changes.

Experts call this: CI/CD (continuous integration and delivery), release engineering

Worth knowing Automatic checks catch many mistakes, not all of them. Small releases don't replace testing, and some problems still get through.

From idea to customer: ask and wait, or a ready pit.

STATION	ASK AND WAIT	READY PIT
Idea	An idea is ready. First, a form to ask for somewhere to build it.	An idea is ready. A starting template is already on the wall.
A place to build	The request joins the queue. The stopwatch keeps sweeping.	A test copy of the app comes off the wall. Risky tools still need a person's yes.
Checks	Checks are done by hand and saved up for the end.	The same automatic checks run on this change. Even the ready car stops here.
Release	The change waits for the next big release day, packed with everything else.	This one small change goes out on its own, to a few customers first.
Customers	Customers finally see it, bundled with many other changes.	Customers can see it without waiting for the next big release, and it can be switched off if something looks wrong.

An illustration, not a measurement. Even the ready car stops at the check gate.

Pick a job. Watch the lane paint itself. Comes ready on the lane, and what still asks a person.

A new page for customers

A starting template for customer pages · A test copy of the app to try it in · The same automatic checks · A way to release to a few customers first, and undo · A view of how the page is doing · Team access, set up the house way

Still asks a person: The first release to every customer

A new behind-the-scenes service

A starting template for services · A place to try it, away from customers · The same automatic checks · A way to release and undo · A way to watch it once it runs · Team access, set up the house way

Still asks a person: Access to live customer records

A scheduled report

A starting template for scheduled jobs · Sample data to try it on · The same automatic checks · A way to switch it on and off · A note when a run fails · Team access, set up the house way

Still asks a person: Access to live customer data

A mobile app update

A starting template for app updates · Test phones to try it on · The same automatic checks · A way to send it to a few people first · A view of crashes after it ships · Team access, set up the house way

Still asks a person: Sending the update to everyone

Something nobody here has built before

Off the painted lane. That's fine: talk to the platform crew, and if it comes up often, it becomes the next painted lane.

One big, scary pit stop, or small stops often?

One big pit stop

Every change waits in one big crate for release day, then everything goes out together.

Something breaks: The whole crate rolls back, and nobody can tell which change did it.

Small stops, often

Each change rolls through the check gate on its own, as soon as it is ready.

Something breaks: Usually only that one change is put back; the rest keep rolling.

Worth knowing
Small, frequent releases are easier to check and to undo. They don't replace testing, and some problems still get through.

Worth knowing
Everyday tools: no form, no queue. Risky tools still need a person's yes.

Worth knowing
The painted lane is a recommendation, not a rule. Teams can step off it for unusual work.

contact@algoshred.com

What we help with.

Start with one lane, or bring the pieces together into one shared setup for every team.

Follow one change from idea to customer

We ride along with a real change, talk to the people who build, and run a short survey about waiting, interruptions and confusion.

A map of where the waits and hand-offs are, and which to fix first.

Experts call this: developer experience review, value-stream mapping

A crew that treats your builders as customers

Who looks after the shared setup, how they hear from builders, what they build first and how it is funded.

A platform with an owner and a plan that follows what builders ask for.

Experts call this: platform as a product, team topologies

Ready-made starting points for common jobs

Templates for the jobs you do most, such as a new web service or a scheduled report, with checks, a view of how it is running and access set up the house way.

New work starts the house way, with the checks built in.

Experts call this: golden paths, software templates

A help-yourself tool wall and front door

One place to find what exists and who looks after it, and to pick up routine things like a test copy of the app. A person still approves the risky ones.

Routine requests no longer have to wait in a queue.

Experts call this: developer portal, service catalogue, self-service

The same automatic checks on every change

Build, test and safety checks that run on every change, with less waiting on them and clearer messages when they fail.

Problems often show up earlier, while they are small.

Experts call this: continuous integration (CI), build pipelines

Small releases you can roll out and undo

Small changes, on/off switches for new features, rollouts to a few customers first where that makes sense, and a way back for most changes.

Release day can become an ordinary day.

Experts call this: continuous delivery (CD), release engineering, feature flags, gradual rollout

Measuring the waits, and asking the builders

A few plain delivery measures plus a regular builder survey, agreed with you at the start. Experts call what the survey asks about developer experience; in plain words: less waiting, fewer interruptions, less confusion.

You can see whether it is getting smoother, in your own numbers.

Experts call this: DORA measures (four standard delivery measures), developer experience surveys

Room for AI helpers on the same road

Giving coding assistants and agents (AI helpers that take steps on their own) the same paths, the same written instructions, checks and limits as people, plus a person reviewing what they write.

AI helpers that go through the same checks as everyone else, which catch many mistakes, not all, plus a person's review.

Experts call this: AI-assisted engineering, agent guardrails

> For your technical team: we work with what you already use, for example GitHub Actions, GitLab CI, Jenkins, Argo CD, Flux, Backstage, Port, Terraform, OpenTofu, Crossplane, Kubernetes and OpenFeature.

CAPABILITIES

Capabilities, from the first ride-along to small releases you can undo.

From listening to the people who build, to ready-made starting points, to one front door for tools, to small releases you can undo, and room for AI helpers on the same road.

Listen and plan

Where your builders get stuck, and what to fix first.

- Developer journey mapping: following real changes from idea to customer
- Builder surveys on waiting, interruptions and confusion
- A plan for the platform, and for who looks after it
- A plan that starts with the smallest useful platform

Pave the paths

Ready-made starting points with the checks already in them.

- Golden-path templates for common jobs
- Help-yourself test copies of the app
- Safety, cost and policy checks built into the paths

Open the front door

One place to find things, take things and see who looks after them.

- A developer portal and service catalogue (who looks after what)
- Written guides kept next to the work they describe
- Scorecards: a simple checklist of how each service measures up to the house rules

Ship small and measure

Small, checked releases, and a plain view of whether it is getting smoother.

- Automatic build and test steps that keep builders waiting less and explain failures clearly (continuous integration, CI: every change is built and tested automatically)
- Feature switches and gradual rollouts with a way back (continuous delivery, CD: every checked change is ready to release)
- Plain delivery measures plus builder feedback
- Room for AI helpers inside the same checks

WHERE IT BECOMES REAL · ILLUSTRATIVE EXAMPLES

Retail and e-commerce

A small change to the checkout goes out to a few shoppers first, through the same checks as every other change, and can be switched off if something looks wrong.

Financial services

For example, each new service could start from a template with your own safety and record-keeping checks already in it, so reviews look at the work, not the setup.

SaaS and product companies

A new joiner picks a ready-made path, gets a test copy of the app without filing a request, and ships a small, checked change early on.

Healthcare

For example, teams could help themselves to routine test setups, while anything touching patient records waits for a named person’s approval.

Media and publishing

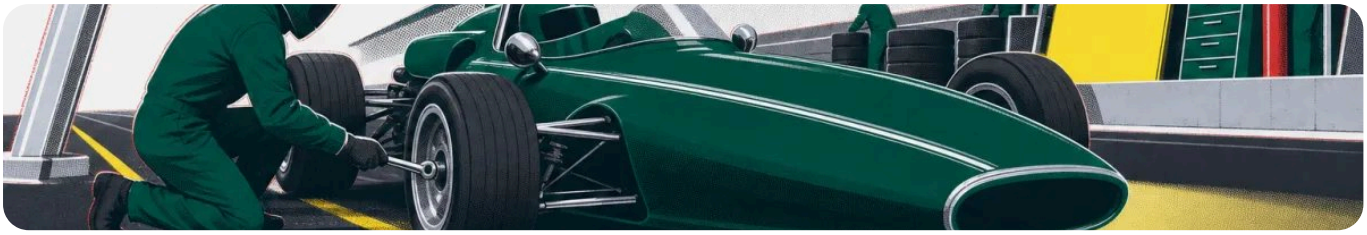
Many small releases replace one big Friday launch, so when something breaks it is clear which change did it.

Any team using AI coding helpers

Code written with AI help goes through the same paths and checks as everyone else’s, plus a person reviewing it, so speed doesn’t skip the safety nets.

HOW WE WORK

Listen first, then paint one lane at a time.



01 LISTEN

Listen to the people who build

We ride along with one real change, from idea to customers, and talk to the people who build. A short survey asks where work waits, what interrupts and what confuses.

02 STUCK SPOTS

Find the stuck spots

We mark every wait, hand-off and “ask the one person who knows” on one map. Then we agree with you what smoother means, and which stuck spot to fix first.

03 ONE LANE

Paint one lane

We build the smallest useful platform for your most common job, with one pilot team (the first team to try it), even if part of it is good written steps. Experts call this the thinnest viable platform.

04 TOOL WALL

Open the tool wall

Routine requests come within reach through one front door, and the wall records who took what. A person still says yes to the risky ones.

05 YOUR TEAM RUNS IT

Your team keeps improving it

We hand over the know-how so your team can keep improving the platform. When a new kind of job keeps coming up, we help paint the next lane if you want us to.

Ways to start

Follow one change. A ride-along with one real change, and a map of where it waits.

First painted lane. One ready-made starting point for your most common job, built with one team.

Release-day tune-up. Help one product start moving from big, rare releases towards small ones you can undo.

What you get

- A map of one change from idea to customer, with every wait marked
- A short list of stuck spots to fix first, agreed with you
- One painted lane for your most common job, built with a pilot team
- Written steps your team can follow and change
- A few plain measures and a builder survey, agreed at the start
- Hand-over sessions for the people who will look after the platform

NEXT STEP

Questions, myths and the next step.

What is a developer platform, in plain words?

A shared, ready-to-use setup for the people who build your software, with ready-made starting points, help-yourself tools and the same checks on every change.

Is this the same as DevOps?

It grows out of it. DevOps asks builders and operators (the people who keep systems running) to work together. A platform packages what they agree on into something every team can pick up and use.

Do we need a big team, or Kubernetes (a popular system for running apps), to start?

No. The first version can be a few good written paths and some automation on tools you already have. It grows only when it needs to.

Won't it take freedom away from our teams?

Painted lanes are the easy option, not a fence. Teams can step off them for unusual work, and the common detours become new lanes.

How do you make sure app builders want to use it?

We build it with them, start with what slows them down most, treat them as customers and keep asking. If a path isn't easier than their own way, we fix the path.

Do we have to replace our current tools?

No. We start from what you have, including the build and release tools you already run, and connect it. A portal or new tool comes in only where it helps.

How will we know it's working?

We agree a few measures with you at the start, in plain words: how long finished changes wait before customers get them, how often you release, how often a release needs a rescue, and how quickly you recover (experts call these DORA measures). We also ask the builders. The numbers stay with you, and we make no promises about them.

How long does it take, and what happens after?

It depends on your teams and what you already have. We start with one ride-along, then agree the next step together, one lane at a time. Afterwards your platform crew keeps the paths, the written steps and the measures, and we train them to run and grow it. We can stay on to help paint the next lanes if you prefer.

MYTHS, ANSWERED

"A platform is a tool you buy."

It's a way of working plus a shared setup built around your teams. A portal or tool is just the front door. In our story, a new tool alone didn't fix it; watching the drivers did.

"We don't have a platform, so we don't need one."

You already have one: forms, queues and the one person who knows. The question is whether it's a good one.

"Releasing less often is safer."

One big release mixes many changes, so when something breaks it is hard to tell which one did it. Small releases are easier to check and to put back. They still need testing, and some problems still get through.

"Build it once and you're done."

It's a product: start small, listen to your builders and keep improving it. It's a journey, not a quick fix: we build it with your teams, one lane at a time.

Is your fastest driver stuck in the garage?

Start by following one change: email contact@algoshred.com with what your team builds and where work seems to wait, and we'll suggest where to start.

contact@algoshred.com

WATCH THE FILM

algoshred.com/developer-platforms