APPETIZER LABS

Platform Bottleneck Checklist

This checklist makes twenty-four statements about your path to production, grouped into six areas. You tick what is true today. What stays unticked is the list of things slowing your delivery down — and the basis for deciding whether a platform would fix any of it.

Reviewed: 2026-08-09

Who it is for

For engineering leads, platform teams, and anyone who has to explain why releases take longer than they should. Work through it with two or three people who see the path to production from different sides: development, operations, and whoever answers for the budget.

How to use it

Only tick what is already true — not what is planned, requested, or described in a ticket. If the room disagrees about a statement, it counts as open: the disagreement is itself a finding. At the end you count the open items, not the ticked ones.

01

Path to production

- [ ] A small change ships the same working day, without anyone hunting for a release window.
- [ ] Nothing between merge and production is a manual step only one named person is allowed to run.
- [ ] A failed deployment can be rolled back without anyone improvising.
- [ ] Every team can see for itself whether its change reached production, without asking.

02

Environments and test data

- [ ] A team gets a new environment without raising a ticket with another team.
- [ ] Environments come from code, not from a hand-tuned configuration that grew over the years.
- [ ] Usable test data is available without copying real personal data.
- [ ] The differences between staging and production are known and written down.

03

Operations and on-call

- [ ] Every service has a team that takes it during an incident — named, not “the IT department”.
- [ ] Alerts that wake someone at night are built so that the person woken can act on them.
- [ ] The most common failures have runbooks someone unfamiliar with the system can follow.
- [ ] After an incident the cause gets fixed, not just the ticket closed.

04

Platform as a product

- [ ] Developers do the most common tasks themselves instead of requesting them.
- [ ] There is documentation someone can ship with on their first working day.
- [ ] What the platform offers — and what it explicitly does not — has been decided and written down.
- [ ] The platform team knows who its users are, and whether they come voluntarily.

05

Cost and capacity

- [ ] Every team can see the cloud spend it causes itself.
- [ ] Unused environments and resources are cleaned up automatically.
- [ ] Capacity decisions rest on measurements, not on estimates from the early days.
- [ ] When a bill surprises you, the source is clear the same day.

06

Knowledge and dependencies

- [ ] No critical part of the system depends on a single person.
- [ ] Architecture decisions can be read back, including the reasons against them.
- [ ] Version and security updates run as routine, not as a project.
- [ ] Switching provider would be expensive, but the price is known and named.

Reading the result

Count the open items — everything you could not tick. How they cluster matters more than the total: four open items inside one area is a different problem from four spread across all six.

0 to 4 open

Isolated gaps. That is ordinary maintenance, not a platform question. Take the open items one at a time.

5 to 12 open

A visible bottleneck, usually concentrated in one or two areas. Worth finding the cause before buying tooling.

13 or more open

The bottleneck is structural. Individual fixes tend to evaporate here; the order you tackle them in decides the outcome.

What it does not do

The checklist measures nothing. It is no substitute for reading your delivery data or for looking inside clusters, pipelines and invoices. What it does is reorganise a conversation that usually starts with tooling, and point it at the path to production instead. Hard numbers need the data itself.

What to do with the open items

Take the areas with the most empty boxes and work out whether they share a cause. If you want a second opinion, write to info@appetizers.io with the areas and a few sentences of context. You get an assessment of whether a platform would change anything here — and the answer is allowed to be no.