

Who does what? Sort 23 real tasks

Score _____ / 23

Tick one box per task. **DevOps practice**: a habit the whole team follows. **SRE**: site reliability engineers. **Platform team**: the team that builds self-service tools for the company's own developers. **Shared**: it depends on the company.

The answer key, with a reason and a source for every task: antonputra.com/exercises/devops-sre-platform-engineering/

TASK	DEVOPS PRACTICE	SRE	PLATFORM TEAM	SHARED it depends
1 Merge your code into the team's shared main copy at least once a day, with automated tests running every time.	☐	☐	☐	☐
2 Ship one small change at a time instead of a big release every few months.	☐	☐	☐	☐
3 Build a "new service" starter template, so every team starts with tests, a pipeline, logs and alerts already set up.	☐	☐	☐	☐
4 Wake a human only when users are actually hurt (errors or slowness), not every time one server is busy.	☐	☐	☐	☐
5 Maintain the shared build-and-deploy pipeline (CI/CD) that turns a code change into a running release.	☐	☐	☐	☐
6 Let low-risk changes go live after a colleague's review and automated checks, instead of waiting for a weekly approval meeting.	☐	☐	☐	☐
7 Give developers a button or command that creates a database in minutes, without filing a ticket.	☐	☐	☐	☐
8 Review a new service before it launches: is it monitored, and can it survive a traffic spike?	☐	☐	☐	☐
9 Interview developers about what slows them down, then plan the platform's roadmap from their answers.	☐	☐	☐	☐
10 Write code that creates the company's cloud servers and networks ("infrastructure as code").	☐	☐	☐	☐
11 Run the developer portal: one web page that lists every service, who owns it, and its docs.	☐	☐	☐	☐
12 Lead the response while checkout is failing: decide who does what and keep everyone updated.	☐	☐	☐	☐
13 Roll back the change that broke checkout, to stop the damage before digging for the cause.	☐	☐	☐	☐
14 Put security and compliance checks into the default pipeline, so every team gets them without extra work.	☐	☐	☐	☐
15 The developers who built a feature also get alerted when it breaks in production ("you build it, you run it").	☐	☐	☐	☐
16 Agree on a reliability target, such as "99.9% of checkout requests succeed each month", and measure it.	☐	☐	☐	☐
17 Release a new version to a small slice of users first (a "canary"), watch the errors, then roll it out to everyone.	☐	☐	☐	☐
18 Write a blameless report after the outage: what happened, why, and which fixes will stop it happening again.	☐	☐	☐	☐
19 After the outage, add an automatic safety check (roll back when errors jump) to the shared pipeline template, so every team gets it.	☐	☐	☐	☐
20 When this month's error budget is used up, pause risky launches until reliability recovers.	☐	☐	☐	☐
21 Developers and operations people watch the same dashboards and fix problems together, instead of blaming each other.	☐	☐	☐	☐
22 Keep repetitive manual work under half of your team's time, and spend the rest writing code that removes it.	☐	☐	☐	☐
23 Write the guides and run demos so other teams actually choose to use the platform.	☐	☐	☐	☐

The key shows the usual split described by Google's SRE books, the CNCF and DORA. Your company may differ.

ANTON PUTRA