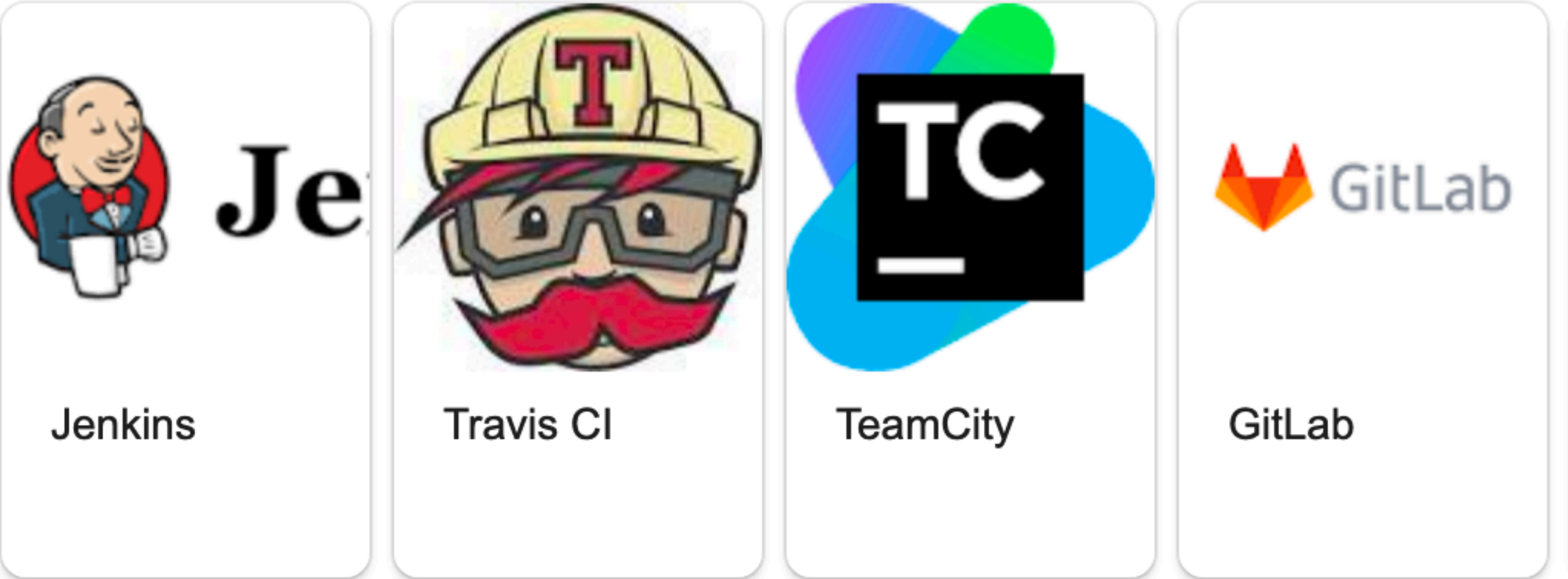
Continuous Integration как практика

Андрей Александров

CI? Что приходит в голову?

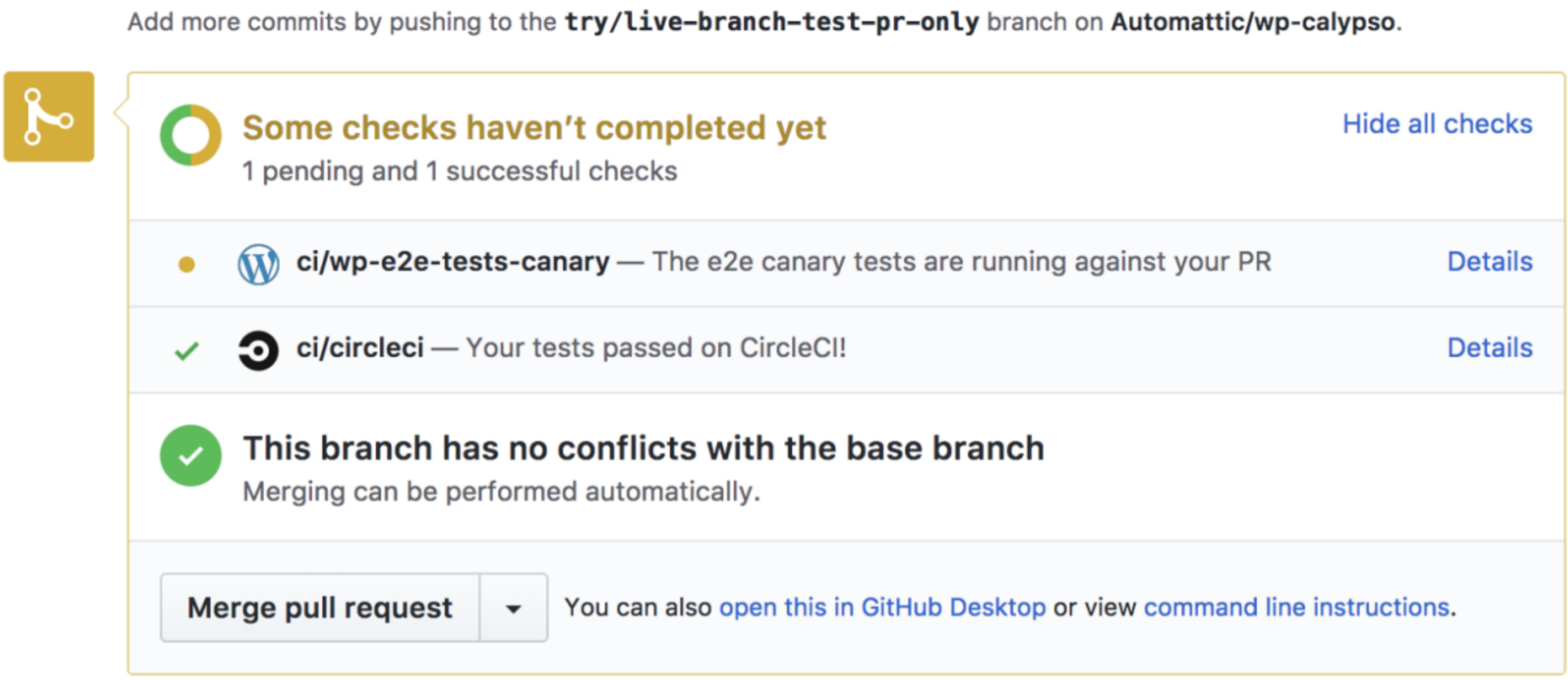


Jenkins

Travis CI

TeamCity

GitLab



Add more commits by pushing to the `try/live-branch-test-pr-only` branch on **Automattic/wp-calypso**.

Some checks haven't completed yet

1 pending and 1 successful checks

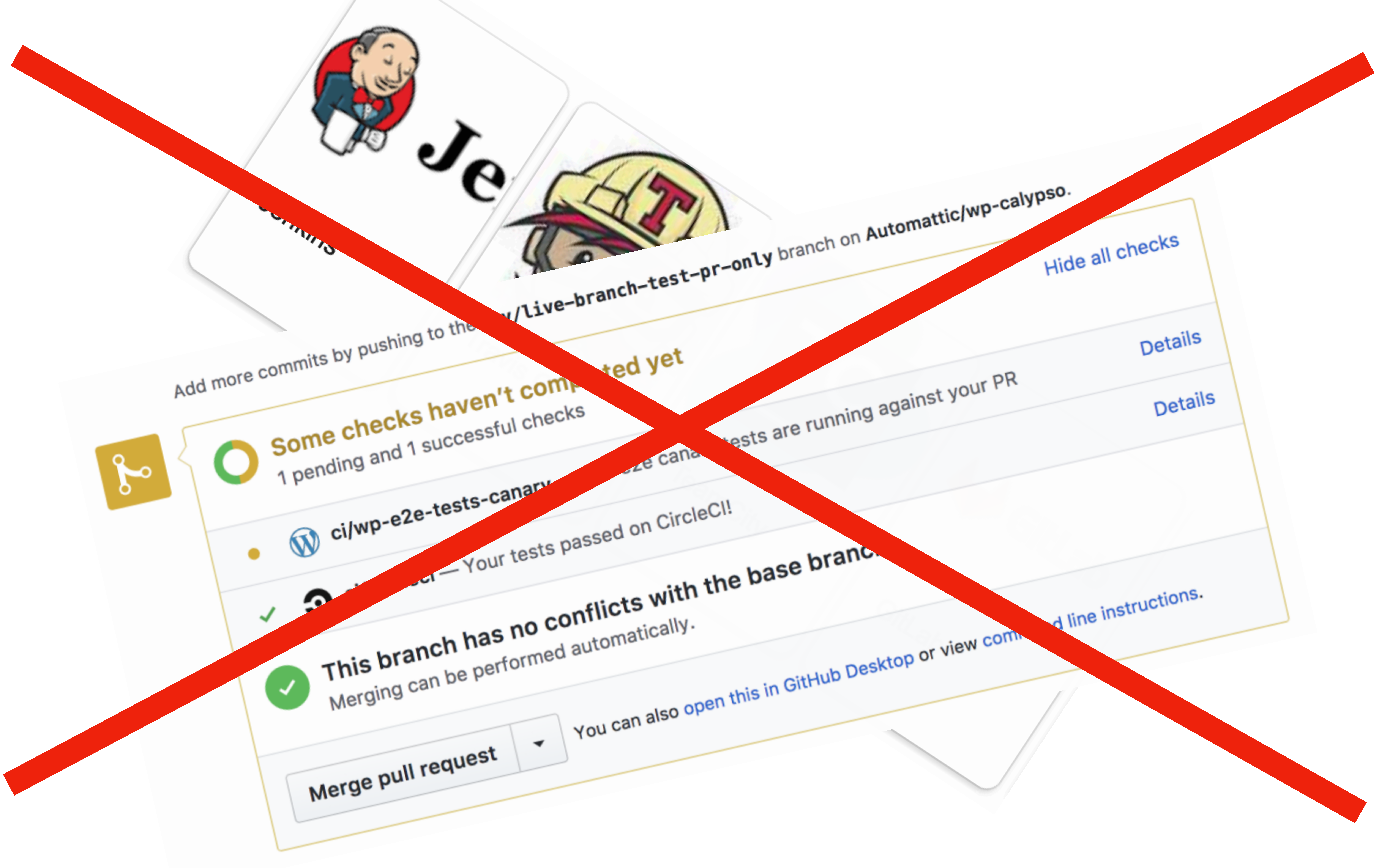
● **ci/wp-e2e-tests-canary** — The e2e canary tests are running against your PR

✓ **ci/circleci** — Your tests passed on CircleCI!

This branch has no conflicts with the base branch

Merging can be performed automatically.

You can also open this in GitHub Desktop or view command line instructions.



Add more commits by pushing to the v/live-branch-test-pr-only branch on Automattic/wp-calypso.

Some checks haven't completed yet

1 pending and 1 successful checks

Hide all checks

● ci/wp-e2e-tests-canary — e2e canary tests are running against your PR Details

✓ ci/circleci — Your tests passed on CircleCI! Details

This branch has no conflicts with the base branch

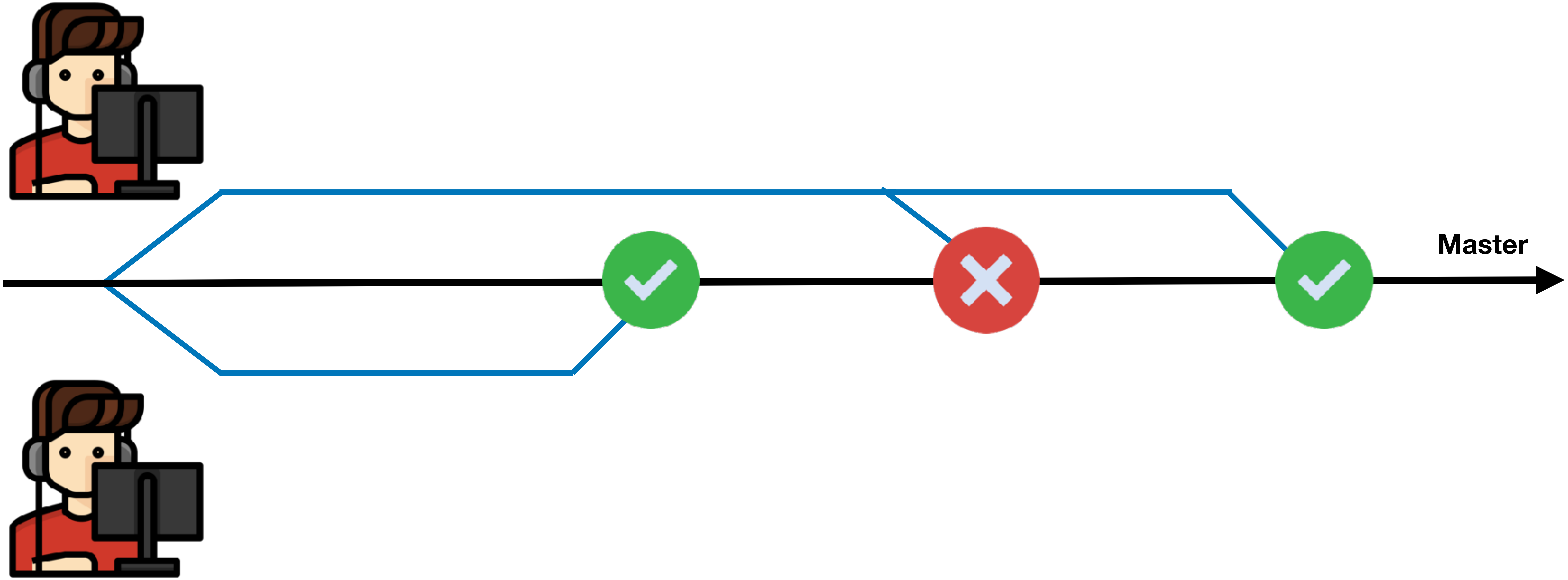
Merging can be performed automatically.

You can also open this in GitHub Desktop or view command line instructions.

Какую боль изначально решали?

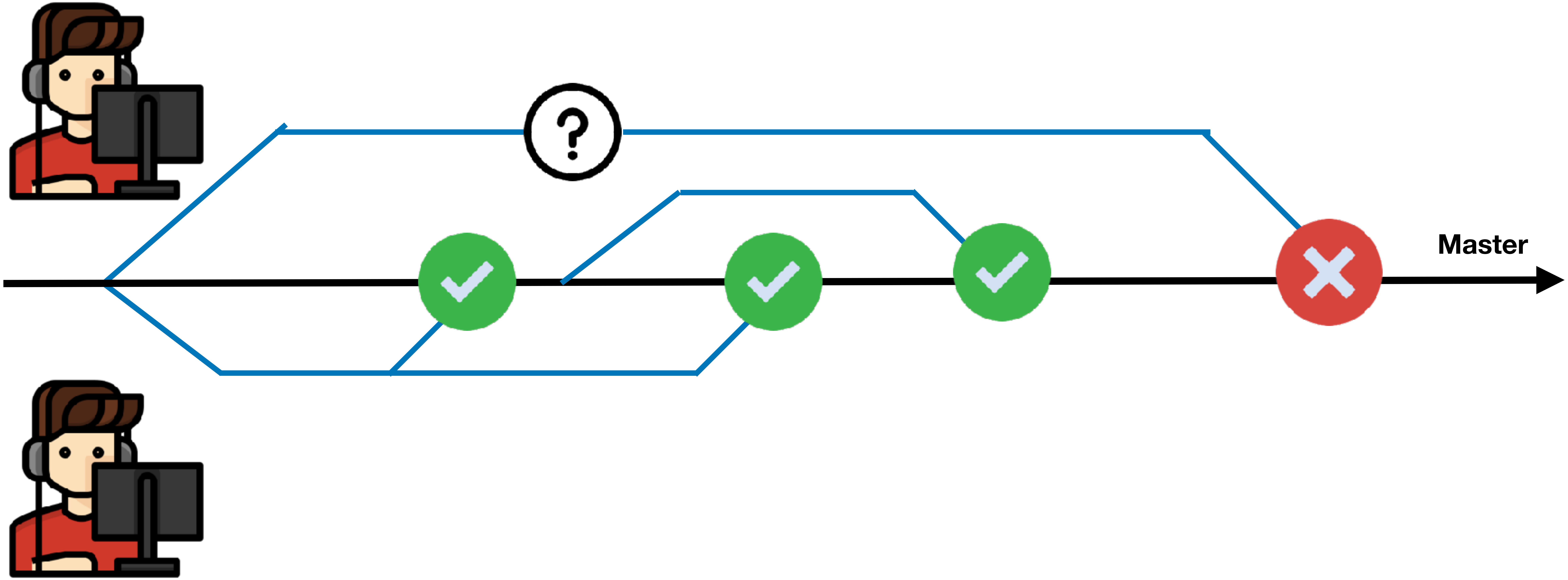
Совместной работы

Командная работа



Icon made by dDara from www.flaticon.com
Icon made by kiranshastry from www.flaticon.com

Командная работа



Icon made by dDara from www.flaticon.com
Icon made by kiranshastry from www.flaticon.com

Совместная работа — боль!

Extreme Programming empowers your developers to confidently respond to changing customer requirements, even late in the life cycle.

The first Extreme Programming project was started March 6, 1996.

Integration

Everyone needs to work with the latest version. Changes should not be made to obsolete code causing integration head aches.

if you integrate throughout the project in small amounts you will not find your self trying to integrate the system for weeks at the project's end while the deadline slips by.

Continuous

Developers should be integrating and commiting code into the code repository every few hours, when ever possible. In any case never hold onto changes for more than a day.

Each development pair is responsible for integrating their own code when ever a reasonable break presents itself.

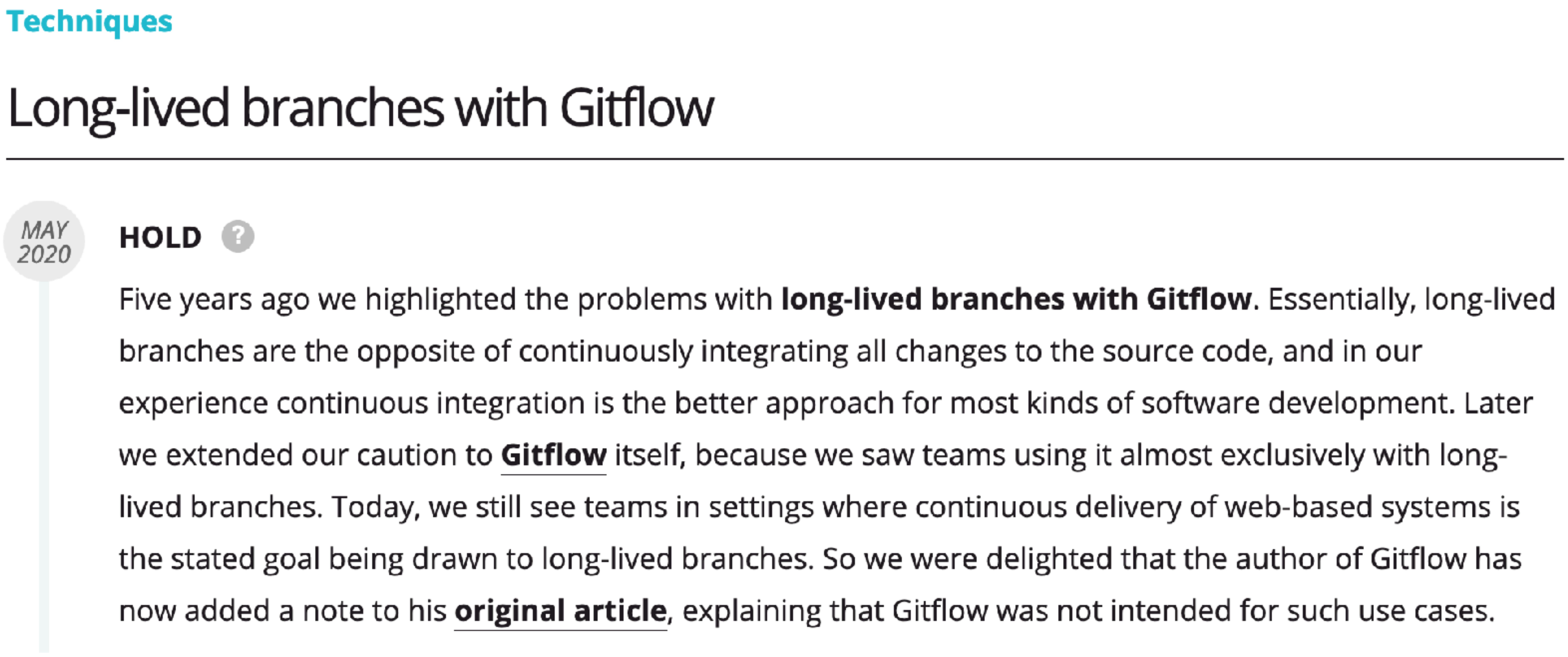
Пруфы!

State of DevOps 2018 (Accelerate)

teams using branches that live a short amount of time (integration times less than a day) combined with short merging and integration periods (less than a day) do better in terms of software delivery performance than teams using longer-lived branches

Deployment frequency   Lead time for changes   Time to restore service   Change failure rate

TECHNOLOGY RADAR



Techniques

Long-lived branches with Gitflow

MAY 2020

HOLD ?

Five years ago we highlighted the problems with **long-lived branches with Gitflow**. Essentially, long-lived branches are the opposite of continuously integrating all changes to the source code, and in our experience continuous integration is the better approach for most kinds of software development. Later we extended our caution to **Gitflow** itself, because we saw teams using it almost exclusively with long-lived branches. Today, we still see teams in settings where continuous delivery of web-based systems is the stated goal being drawn to long-lived branches. So we were delighted that the author of Gitflow has now added a note to his **original article**, explaining that Gitflow was not intended for such use cases.

Тест от Jez Humble на наличие CI

1. Код инженера попадает в мастер **ежедневно**
2. На каждый коммит запускаются unit-тесты
3. Если билд упал, его чинят примерно за 10 минут

https://www.continuousdelivery.com/foundations/continuous-integration

Вопросики?