Twitter Thread by Alberto Gimeno ■■■■■■■■



Alberto Gimeno ■■■■■■■■
@gimenete

I'm going to tell you a little bit how we work on features at GitHub. It's simple, but very powerful in my opinion. At GitHub we have a high performance culture.

We are deploying changes almost all the time while keeping the service running and the deployment failure ratio needs to be super low! We are constantly working on new medium/big features, we have frequent deploys and required peer reviews.

How do we know that big features are not conflicting with each other in functionality and in version control. How do we merge the work done in these non-trivial features? How do we do prevent peer reviews slow down our productivity?

With basically two things: split the work in small batches and feature flags. We don't have long-lived feature branches. We do very small changes all the time and deploy them.

We could split a feature in tens or hundreds of PRs, but until it is beta or GA, we don't expose it to users by using feature flags.

A change in the data model, a new endpoint, a change in one screen in the UI,... all those are small changes: easy to reason about, easy to review, harmless to deploy, very low chance that it conflicts with something else.

Also these batches can be worked in parallel so you can have a few PRs waiting for review, but typically you can keep doing work on another small part of the feature.

I'm writing about this because when talking about feature flags most people think about features already finished. We use feature flags for features under development too! And it is super helpful and changes the way we work with git.

Like I said, we don't use long-lived feature branches, which simplifies everything because allows us to do incremental changes to our features and reduce the deployment failure.