

team can become your cheerleader and help convince other teams to migrate, too. Fixing the specific deployment issue can make the CEO happy and get you support to use IaC for more projects. This will allow you to go for another quick win, and another one after that. And if you can keep repeating this process—delivering value early and often—you’ll be far more likely to succeed at the larger migration effort. But even if the larger migration doesn’t work out, at least one team is more successful now and one deployment process works better, so it was still worth the investment.

## **Give Your Team the Time to Learn**

I hope that, at this point, it’s clear that adopting IaC can be a significant investment. It’s not something that will happen overnight. It’s not something that will happen magically, just because the manager gives you a nod. It will happen only through a deliberate effort of getting everyone on board, making learning resources (e.g., documentation, video tutorials, and, of course, this book!) available, and providing dedicated time for team members to ramp up.

If your team doesn’t get the time and resources that it needs, then your IaC migration is unlikely to be successful. No matter how nice your code is, if your entire team isn’t on board with it, here’s how it will play out:

1. One developer on the team is passionate about IaC and spends a few months writing beautiful Terraform code and using it to deploy lots of infrastructure.
2. The developer is happy and productive, but unfortunately, the rest of the team did not get the time to learn and adopt Terraform.
3. Then, the inevitable happens: an outage. One of your team members needs to deal with it, and they have two options: either (A) fix the outage the way they’ve always done it, by making changes manually, which takes a few minutes, or (B) fix the outage by using Terraform, but they aren’t familiar with it, so this could take hours or days. Your