section, when I say "infrastructure code," I mean code written with any IaC tool (including, of course, Terraform) that you can use to deploy arbitrary infrastructure changes beyond a single application: for example, deploying databases, load balancers, network configurations, DNS settings, and so on.

Here's what the infrastructure code workflow looks like:

1. Use version control
2. Run the code locally
3. Make code changes
4. Submit changes for review
5. Run automated tests
6. Merge and release
7. Deploy

On the surface, it looks identical to the application workflow, but under the hood, there are important differences. Deploying infrastructure code changes is more complicated, and the techniques are not as well understood, so being able to relate each step back to the analogous step from the application code workflow should make it easier to follow along. Let's dive in.

Use Version Control

Just as with your application code, all of your infrastructure code should be in version control. This means that you'll use `git clone` to check out your code, just as before. However, version control for infrastructure code has a few extra requirements:

- *Live* repo and *modules* repo
- Golden Rule of Terraform
- The trouble with branches