industry, so you'll probably be familiar with parts of it. Later in this chapter, I'll talk about a workflow for taking infrastructure code (e.g., Terraform modules) from development to production. This workflow is not nearly as well known in the industry, so it will be helpful to compare that workflow side by side with the application workflow to understand how to translate each application code step to an analogous infrastructure code step.

Here's what the application 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.

Let's go through these steps one at a time.

Use Version Control

All of your code should be in version control. No exceptions. It was the #1 item on the classic Joel Test when Joel Spolsky created it more than 20 years ago, and the only things that have changed since then are that (a) with tools like GitHub, it's easier than ever to use version control and (b) you can represent more and more things as code. This includes documentation (e.g., a README written in Markdown), application configuration (e.g., a config file written in YAML), specifications (e.g., test code written with RSpec), tests (e.g., automated tests written with JUnit), databases (e.g., schema migrations written in ActiveRecord), and of course, infrastructure.

As in the rest of this book, I'm going to assume that you're using Git for version control. For example, here is how you can check out the code repo