```
  # (...)
}
```

If you are a customer of HashiCorp's Terraform Cloud or Terraform Enterprise, you can have this same experience with a Private Terraform Registry—that is, a registry that lives in your private Git repos and is only accessible to your team. This can be a great way to share modules within your company.

Beyond Terraform Modules

Although this book is all about Terraform, to build out your entire production-grade infrastructure, you'll need to use other tools, too, such as Docker, Packer, Chef, Puppet, and, of course, the duct tape, glue, and work horse of the DevOps world, the trusty Bash script.

Most of this code can reside in the *modules* folder directly alongside your Terraform code: e.g., you might have a *modules/packer* folder that contains a Packer template and some Bash scripts you use to configure an AMI right next to the *modules/asg-rolling-deploy* Terraform module you use to deploy that AMI.

However, occasionally, you need to go further and run some non-Terraform code (e.g., a script) directly from a Terraform module. Sometimes, this is to integrate Terraform with another system (e.g., you've already used Terraform to configure User Data scripts for execution on EC2 Instances); other times, it's to work around a limitation of Terraform, such as a missing provider API, or the inability to implement complicated logic due to Terraform's declarative nature. If you search around, you can find a few "escape hatches" within Terraform that make this possible:

- Provisioners
- Provisioners with `null_resource`
- External data source