```
      # Run Terraform using HashiCorp's setup-terraform Action
      - uses: hashicorp/setup-terraform@v1
          with:
            terraform_version: 1.1.0
            terraform_wrapper: false
        run: |
          terraform init
          terraform apply -auto-approve
```

Now, when you run this build in one of the repos and branches in the `allowed_repos_branches` variable, GitHub will be able to assume your IAM role automatically, using temporary credentials, and Terraform will authenticate to AWS using that IAM role, all without having to manage any credentials manually.

Resources and Data Sources

The next place you'll run into secrets with your Terraform code is with resources and data sources. For example, you saw earlier in the chapter the example of passing database credentials to the `aws_db_instance` resource:

```
resource "aws_db_instance" "example" {
  identifier_prefix   = "terraform-up-and-running"
  engine              = "mysql"
  allocated_storage   = 10
  instance_class      = "db.t2.micro"
  skip_final_snapshot = true
  db_name             = var.db_name

  # DO NOT DO THIS!!!
  username = "admin"
  password = "password"
  # DO NOT DO THIS!!!
}
```

I've said it multiple times in this chapter already, but it's such an important point that it's worth repeating again: storing those credentials in the code, as plain text, is a bad idea. So, what's a better way to do it?