# Terraform S3 Backend

“Stores the state as a given key in a given bucket on [Amazon S3](https://aws.amazon.com/s3/). This backend also supports state locking and consistency checking via [Dynamo DB](https://aws.amazon.com/dynamodb/)” (<https://developer.hashicorp.com/terraform/language/settings/backends/s3>).

***Resources used for the S3 backend and DynamoDB configuration:***

* [***https://technology.doximity.com/articles/terraform-s3-backend-best-practices***](https://technology.doximity.com/articles/terraform-s3-backend-best-practices)
* [***https://github.com/hashicorp/terraform/issues/17663***](https://github.com/hashicorp/terraform/issues/17663)
* [***https://jhooq.com/terraform-error-config-s3/***](https://jhooq.com/terraform-error-config-s3/)

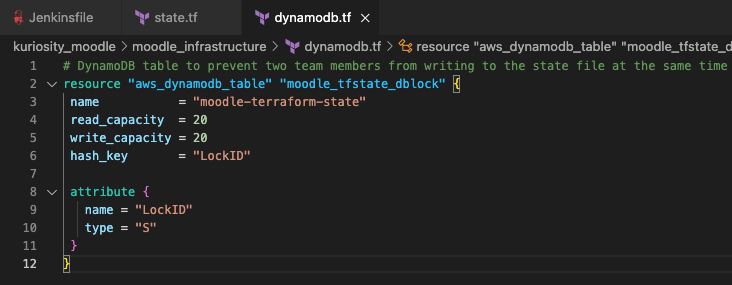
Prior to configuring an S3 backend, terraform.tfstate and terraform.tfstate.backup files lived locally on the Jenkins Agent:



Followed the resources mentioned above to set up S3 bucket and DynamoDB in our infrastructure to configure a Terraform backend – S3 and associated resource blocks:

Text

Description automatically generated

DynamoDB resource block:

After committing the changes to the repository – first ran terraform init with just regular AWS access and secret keys, but that was not enough information to initialize the S3 backend:

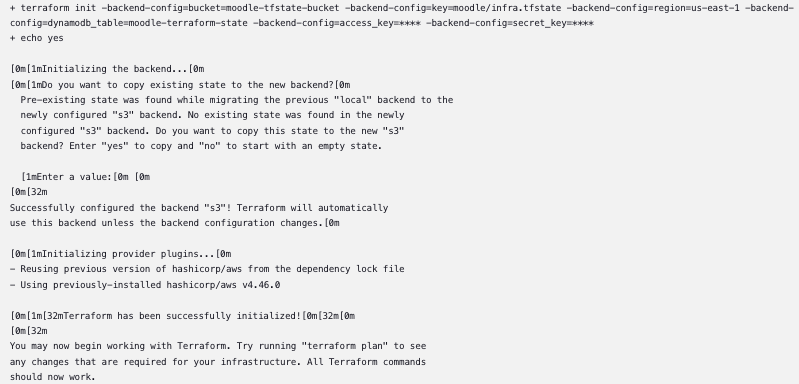
Graphical user interface, text, application, email

Description automatically generated

Provided all backend config information but the terraform init command returned an error because it required user input of “yes” for copying existing state file to the new s3 bucket:

Text, letter

Description automatically generated

To address this, I echoed “yes” and piped it to the terraform init command and this worked:

Confirm on AWS that the new S3 bucket has been created:

Graphical user interface, application, email

Description automatically generated

Navigate to the moodle folder inside the bucket:Graphical user interface, text, application, email

Description automatically generated

The infra.tfstate object contains the terraform.tfstate file for the infrastructure:

Graphical user interface, text, application, email

Description automatically generated

Content of the infra.tfstate object:

Text

Description automatically generated

In DynamoDB > Items > moodle-terraform-state, you’ll see the LockID created:

Graphical user interface, text, application, email

Description automatically generated

To confirm it can store multiple states, I ran the pipeline again with some changes – the object version has changed in S3:Graphical user interface, text, application, email

Description automatically generated

DynamoDB item has updated as well:

Graphical user interface, text, application, email

Description automatically generated

Confirming that we were able to successfully store the terraform.tfstate file in S3, and lock down the infrastructure using DynamoDB, which keeps the metadata of resources and state file to lock in place.