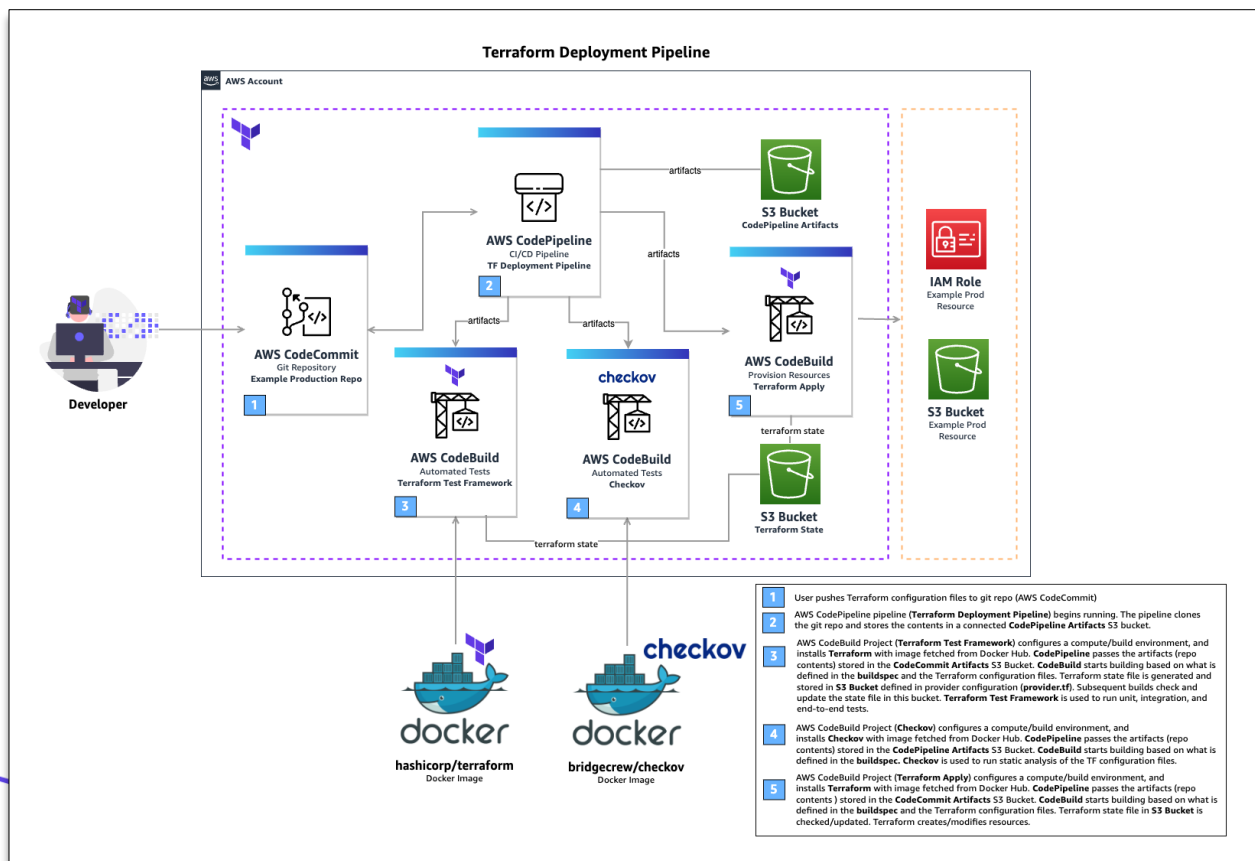


# TERRAFORM CI/CD AND TESTING ON AWS



Joey Acosta | AWS Cloud Engineer





**JOEY ACOSTA**

**AWS Cloud Engineer**

[linkedin.com/in/joeyacosta](https://www.linkedin.com/in/joeyacosta) | [github.com/joeycloudio](https://github.com/joeycloudio)

2

## Problem Statement

---

Manual Terraform deployments and lack of testing increase the risk of misconfigured infrastructure, security issues, and unscalable cloud environments. This project solves that by building a CI/CD system that automates testing and deployment of Terraform modules.

## Business Impact

---

- Reduced risk of misconfigured infrastructure through automated testing and validation
- Improved deployment speed and consistency with full CI/CD pipeline
- Strengthened security with Checkov integration
- Centralized Terraform state storage for team collaboration using S3 + DynamoDB
- Eliminated manual provisioning, enabling repeatable, production-ready deployments

## Cloud Architecture

---

Services Used:

- **Terraform** – Infra automation, validation, testing
- **AWS CodePipeline** – CI/CD pipeline orchestration
- **AWS CodeBuild** – Executes Terraform tests, Checkov scans, and apply
- **S3** – Stores artifacts and remote backend state
- **DynamoDB** – State lock management
- **IAM** – Secure access roles and permissions
- **GitHub (via CodeStar Connections)** – Source repo and trigger for pipelines

Structure:

- Testing Pipeline (module validation)
- Deployment Pipeline (example workload)
- Remote state setup across two pipelines
- Refactored out CodeCommit, replaced with GitHub + CodeStar Connection





**JOEY ACOSTA**

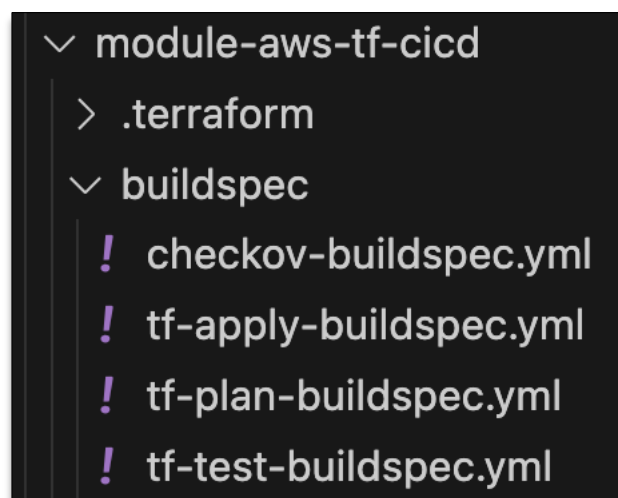
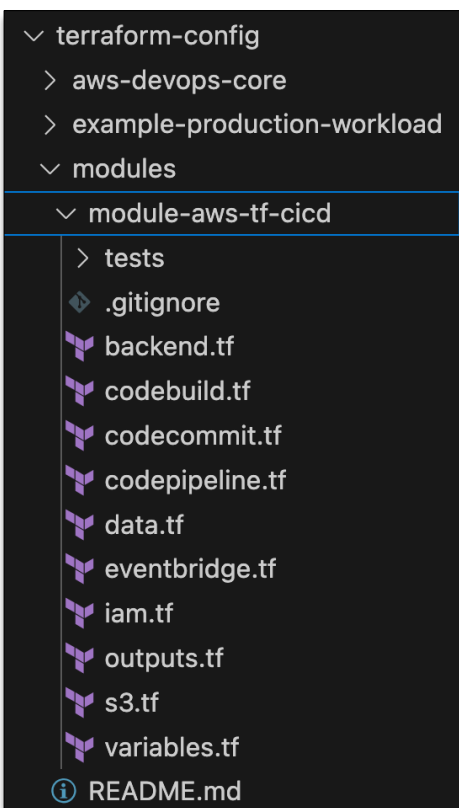
**AWS Cloud Engineer**

[linkedin.com/in/joeyacosta](https://linkedin.com/in/joeyacosta) | [github.com/joeycloudio](https://github.com/joeycloudio)

3

## Technical Implementation

1. Developed a reusable Terraform module to provision AWS infrastructure using IaC best practices, including IAM roles, S3 buckets, and CodePipeline.





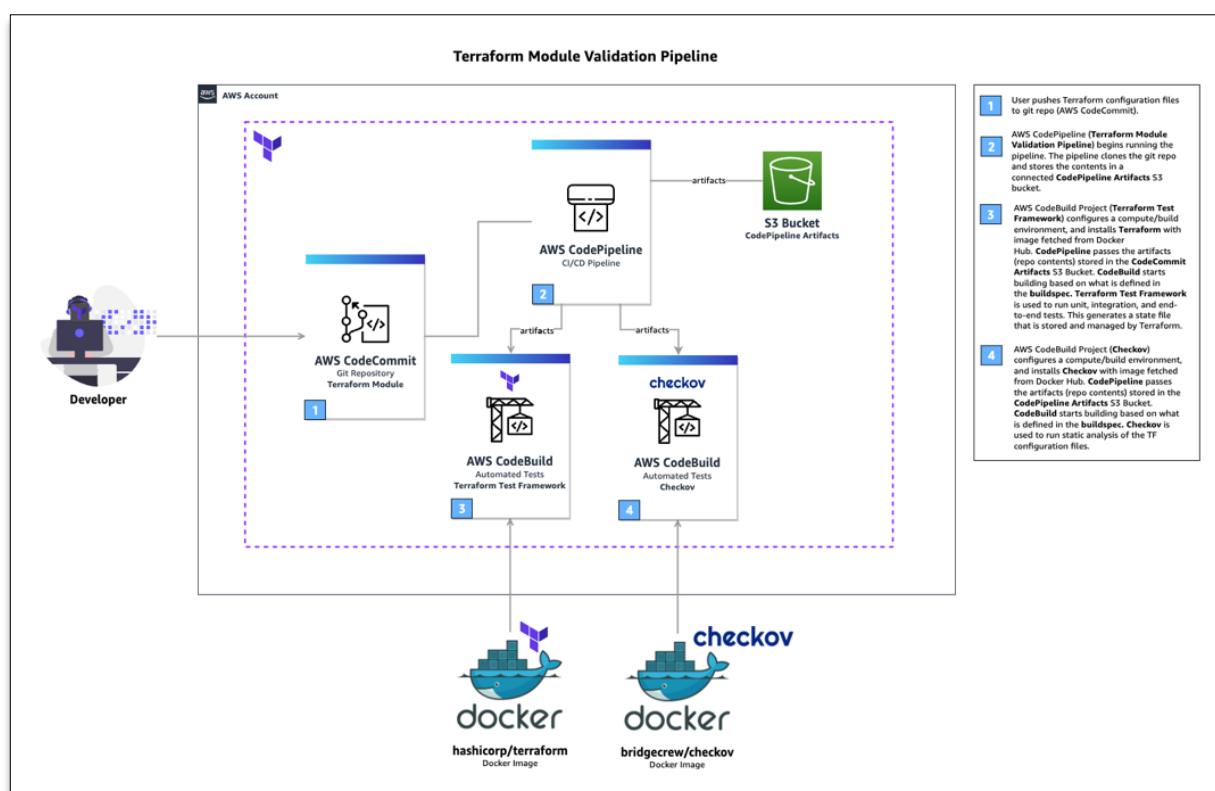
JOEY ACOSTA

AWS Cloud Engineer

[linkedin.com/in/joeyacosta](https://www.linkedin.com/in/joeyacosta) | [github.com/joeycloudio](https://github.com/joeycloudio)

4

## 2. Integrated Terraform Test Framework to validate module functionality with unit, integration, and end-to-end tests.



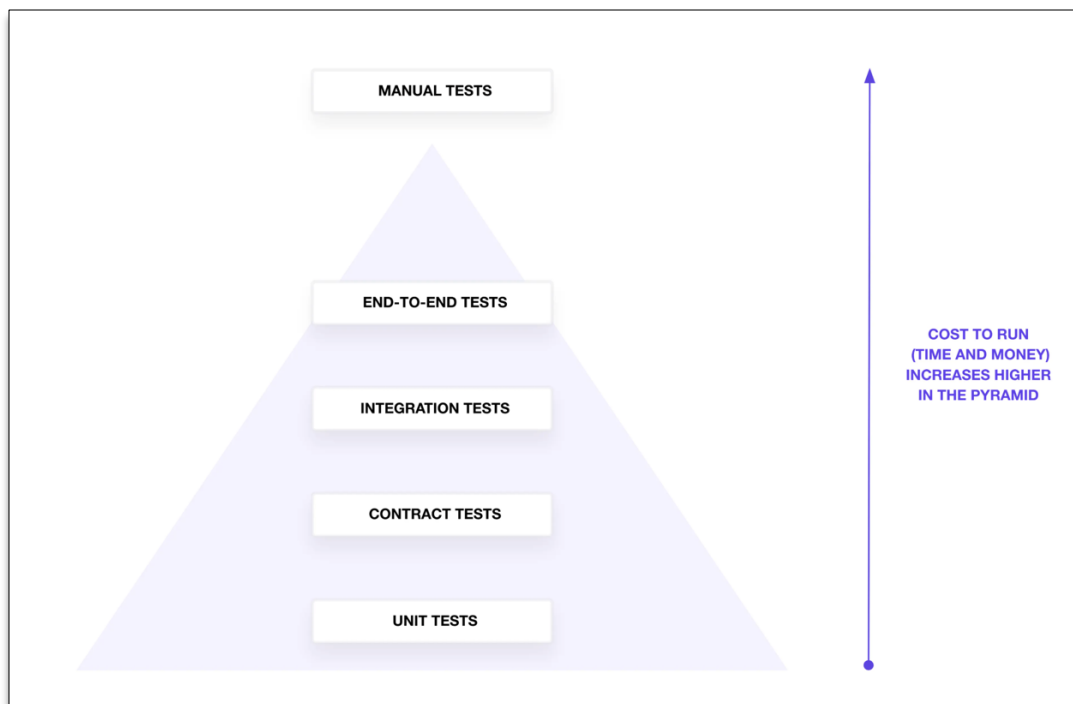


JOEY ACOSTA

AWS Cloud Engineer

[linkedin.com/in/joeyacosta](https://linkedin.com/in/joeyacosta) | [github.com/joeycloudio](https://github.com/joeycloudio)

5



```
joeyacosta@Joeys-MacBook-Pro module-aws-tf-cicd % terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Reusing previous version of hashicorp/random from the dependency lock file
- Using previously-installed hashicorp/aws v5.94.1
- Using previously-installed hashicorp/random v3.7.1

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
joeyacosta@Joeys-MacBook-Pro module-aws-tf-cicd % terraform test
tests/main.tfctest.hcl... in progress
  run "input_validation"... pass
  run "e2e_test"... pass
tests/main.tfctest.hcl... tearing down
tests/main.tfctest.hcl... pass

Success! 2 passed, 0 failed.
```





JOEY ACOSTA

AWS Cloud Engineer

[linkedin.com/in/joeyacosta](https://www.linkedin.com/in/joeyacosta) | [github.com/joeycloudio](https://github.com/joeycloudio)

6

3. Added security scanning with Checkov and enforced linting with TFLint to ensure clean, secure, and compliant Terraform code.

```
joeyacosta@Joeys-MacBook-Pro module-aws-tf-cicd % checkov --directory /Users/joeyacosta/orm-config/modules/module-aws-tf-cicd
[ terraform framework ]: 100%|██████████| [10/10], Current File Scanned=variable
[ secrets framework ]: 100%|██████████| [10/10], Current File Scanned=/Users/joeyacosta/orm-config/modules/module-aws-tf-cicd

checkov

By Prisma Cloud | version: 3.2.400
Update available 3.2.400 -> 3.2.403
Run pip3 install -U checkov to update

terraform scan results:

Passed checks: 94, Failed checks: 0, Skipped checks: 17

Check: CKV_AWS_93: "Ensure S3 bucket policy does not lockout all but root user. (Prevent
PASSED for resource: aws_s3_bucket.tf_remote_state_s3_buckets
File: /backend.tf:10-21
Guide: https://docs.prismacloud.io/en/enterprise-edition/policy-reference/aws-po
Check: CKV_AWS_53: "Ensure S3 bucket has block public ACLS enabled"
PASSED for resource: aws_s3_bucket_public_access_block.tf_remote_state_s3_bucket
File: /backend.tf:31-39
Guide: https://docs.prismacloud.io/en/enterprise-edition/policy-reference/aws-po
Check: CKV_AWS_54: "Ensure S3 bucket has block public policy enabled"
PASSED for resource: aws_s3_bucket_public_access_block.tf_remote_state_s3_bucket
File: /backend.tf:31-39
```

```
joeyacosta@Joeys-MacBook-Pro aws-devops-core % tflint
1 issue(s) found:

Warning: terraform "required_version" attribute is required (terraform_required_version)

on provider.tf line 3:
3: terraform {

Reference: https://github.com/terraform-linters/tflint-ruleset-terraform/blob/v0.11.0/docs/sion.md

joeyacosta@Joeys-MacBook-Pro aws-devops-core % terraform -version
Terraform v1.11.3
on darwin_arm64
+ provider registry.terraform.io/hashicorp/aws v5.94.1
+ provider registry.terraform.io/hashicorp/random v3.7.1
joeyacosta@Joeys-MacBook-Pro aws-devops-core % tflint
joeyacosta@Joeys-MacBook-Pro aws-devops-core %
```





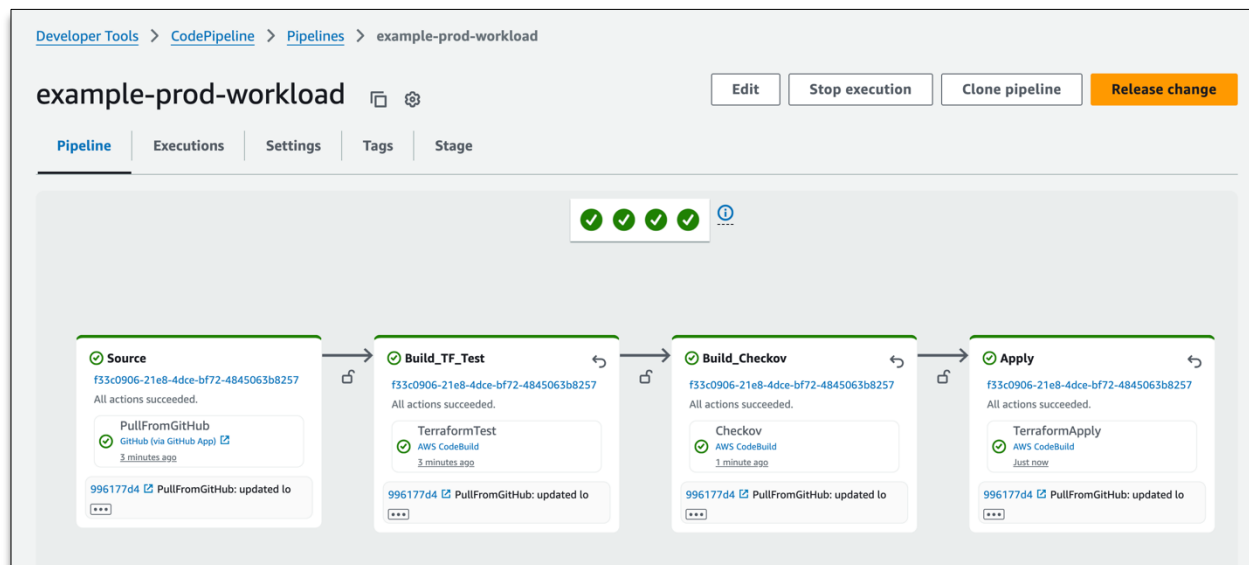
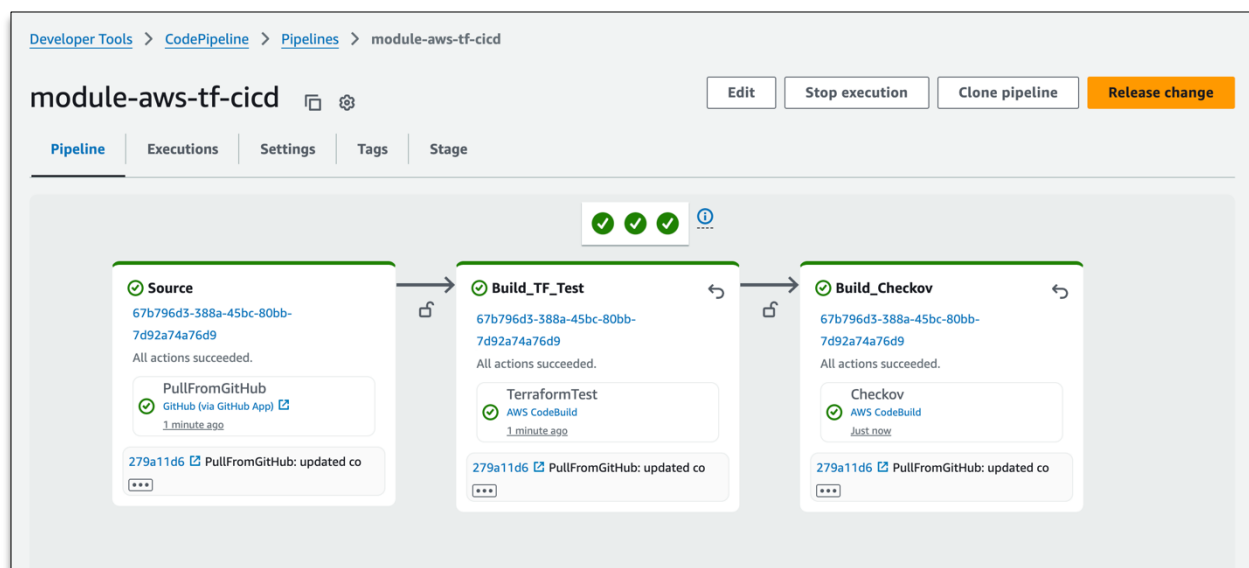
JOEY ACOSTA

AWS Cloud Engineer

[linkedin.com/in/joeyacosta](https://www.linkedin.com/in/joeyacosta) | [github.com/joeycloudio](https://github.com/joeycloudio)

7

4. Built two automated CI/CD pipelines using Terraform, CodePipeline, and CodeBuild to test and deploy infrastructure triggered by GitHub commits.






JOEY ACOSTA

AWS Cloud Engineer

[linkedin.com/in/joeyacosta](https://www.linkedin.com/in/joeyacosta) | [github.com/joeycloudio](https://github.com/joeycloudio)

8



## AWS Connector for GitHub

🕒 Installed 5 days ago    👤 Developed by [aws](#)    📄 <https://docs.aws.amazon.com/dtconsole/latest/userguide/welcome-connections.html>

Enables you to connect GitHub with AWS


### Permissions

- ✓ **Read** access to deployments, issues, and metadata
- ✓ **Read and write** access to administration, code, commit statuses, pull requests, and repository hooks



### Repository access

☐ **All repositories**  
This applies to all current and future repositories owned by the resource owner. Also includes public repositories (read-only).

☒ **Only select repositories**  
Select at least one repository. Also includes public repositories (read-only).

 **Select repositories** ▾

Selected 2 repositories.

 joeycloudio/module-aws-tf-cicd	×
 joeycloudio/example-prod-workload	×







JOEY ACOSTA

AWS Cloud Engineer

[linkedin.com/in/joeyacosta](https://linkedin.com/in/joeyacosta) | [github.com/joeycloudio](https://github.com/joeycloudio)

9

5. Configured S3 remote backend and DynamoDB state locking, enabling collaboration and safe, versioned state management.

```
joeyacosta@Joeys-MacBook-Pro aws-devops-core % terraform init
Initializing the backend...
Do you want to copy existing state to the new backend?
Pre-existing state was found while migrating the previous "local" backend to the
newly configured "s3" backend. No existing state was found in the newly
configured "s3" backend. Do you want to copy this state to the new "s3"
backend? Enter "yes" to copy and "no" to start with an empty state.

Enter a value: yes

Successfully configured the backend "s3"! Terraform will automatically
use this backend unless the backend configuration changes.
Initializing modules...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Reusing previous version of hashicorp/random from the dependency lock file
- Using previously-installed hashicorp/aws v5.94.1
- Using previously-installed hashicorp/random v3.7.1

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

state/ [Copy S3 URI](#)

[Objects](#) | [Properties](#)

**Objects (1)**

[Refresh](#) [Copy S3 URI](#) [Copy URL](#) [Download](#) [Open](#) [Delete](#) [Actions](#) [Create folder](#)

[Upload](#)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

☐ Show versions < 1 > [Settings](#)

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	<a href="#">terraform.tfstate</a>	tfstate	April 11, 2025, 23:35:11 (UTC-07:00)	127.8 KB	Standard





**JOEY ACOSTA**

**AWS Cloud Engineer**

[linkedin.com/in/joeyacosta](https://linkedin.com/in/joeyacosta) | [github.com/joeycloudio](https://github.com/joeycloudio)

10

6. Refactored the project to replace AWS CodeCommit with GitHub and CodeStar Connections, resolving all repo trigger issues and enabling automated pipeline execution.

```
variable "codestar_connection_arn" {  
  description = "CodeStar Connection ARN for GitHub integration"  
  type        = string
```





JOEY ACOSTA

AWS Cloud Engineer

[linkedin.com/in/joeyacosta](https://www.linkedin.com/in/joeyacosta) | [github.com/joeycloudio](https://github.com/joeycloudio)

11

```
stages = [
  # Clone from GitHub, store contents in artifacts S3 Bucket
  {
    name = "Source"
    action = [
      {
        name      = "PullFromGitHub"
        category = "Source"
        owner     = "AWS"
        provider  = "CodeStarSourceConnection"
        version   = "1"
        configuration = {
          ConnectionArn      = var.codestar_connection_arn
          FullRepositoryId   = "joeycloudio/module-aws-tf-cicd"
          BranchName        = "main"
        }
      }
    ]
    input_artifacts = []
    # Store the output of this stage as 'source_output_artifacts'
    output_artifacts = ["source_output_artifacts"]
    run_order        = 1
  },
]
```

```
stages = [
  # Clone from GitHub, store contents in artifacts S3 Bucket
  {
    name = "Source"
    action = [
      {
        name      = "PullFromGitHub"
        category = "Source"
        owner     = "AWS"
        provider  = "CodeStarSourceConnection"
        version   = "1"
        configuration = {
          ConnectionArn      = var.codestar_connection_arn
          FullRepositoryId   = "joeycloudio/example-prod-workload"
          BranchName        = "main"
        }
      }
    ]
    input_artifacts = []
    # Store the output of this stage as 'source_output_artifacts'
    output_artifacts = ["source_output_artifacts"]
    run_order        = 1
  },
]
```





**JOEY ACOSTA**

**AWS Cloud Engineer**

[linkedin.com/in/joeyacosta](https://www.linkedin.com/in/joeyacosta) | [github.com/joeycloudio](https://github.com/joeycloudio)

12

## Key Accomplishments

---

- Replaced CodeCommit with GitHub integration across both pipelines
- Built modular, testable Terraform infrastructure using `terraform test` framework
- Integrated security scanning (Checkov) and linting (TFLint)
- Configured and tested remote S3/DynamoDB backend
- Migrated local Terraform state to remote backend with zero drift
- Troubleshoot and resolved CodePipeline ARN issues, buildspec errors, and webhook bugs
- Successfully deployed and verified 50+ AWS resources end-to-end

## Key Learnings

---

- Real-world Terraform debugging is often reverse-engineering someone else's design
- CI/CD pipelines require precise wiring: GitHub → CodeStar → CodePipeline → CodeBuild
- Remote state config must be handled carefully, especially post-deploy
- Modular Terraform doesn't mean every folder should be a root module
- Git and Terraform interaction (rebase, state drift, force-unlock) matter more than tutorials admit

## Why This Matters in Production

---

This project reflects actual practices used in infrastructure teams—testing Terraform modules, automating secure deployments, and managing state with best practices. It's built for scale, collaboration, and real-world reliability, not just a local sandbox.





**JOEY ACOSTA**

**AWS Cloud Engineer**

[linkedin.com/in/joeyacosta](https://www.linkedin.com/in/joeyacosta) | [github.com/joeycloudio](https://github.com/joeycloudio)

13

## Next Steps

---

- Add a manual approval stage + SNS notifications (optional stretch goal)
- Practice deploying to a second AWS account via cross-account IAM roles
- Reuse modules and pipelines for future real-world Terraform repos

