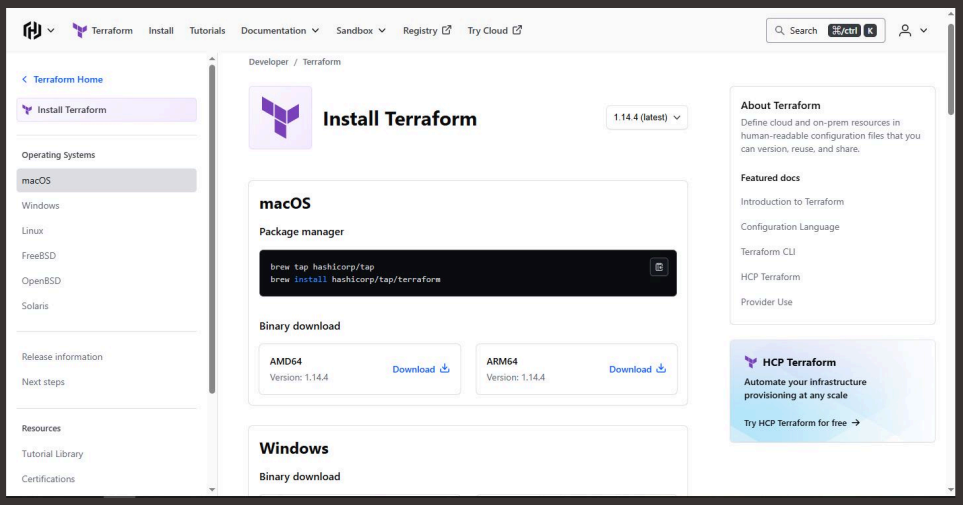
nextwork.org

Create S3 Buckets with Terraform

M Mohammed Dawoud Mota



Mohammed Dawoud M...
NextWork Student

Introducing Today's Project!

Today I’m setting up Terraform on my machine and using it to deploy an S3 bucket in AWS. The goal is to learn the basics of Infrastructure as Code by writing a simple Terraform configuration, initializing it, planning it, and then applying it to create real resources in my AWS account. This project helps me understand how automation replaces manual console work and builds a foundation for more advanced Terraform projects later.

Tools and concepts

In this project, I learned how to use Terraform to deploy infrastructure to AWS from my local machine. I set up the AWS CLI, configured access keys, and connected Terraform to my AWS account. I also learned how Terraform’s workflow fits together — writing the configuration, initializing the provider, reviewing the execution plan, and applying the changes. The main focus today was understanding how authentication works, how Terraform interacts with AWS, and how to troubleshoot credential issues so the deployment can run successfully.

Project reflection

It took me a little over an hour to complete this project. Most of the time went into fixing the AWS authentication issue and making sure the CLI was configured correctly. Once the credentials were set up, the Terraform workflow itself—initializing, planning, and applying—went smoothly and wrapped up pretty quickly.

I chose to do this project today because I wanted to strengthen my Terraform fundamentals and make sure I could deploy infrastructure from my local environment without running into roadblocks. Something that would make learning with NextWork even better is adding more step-by-step support for setup and debugging, so learners can move through the workflow with more

Mohammed Dawoud M...
NextWork Student

Introducing Terraform

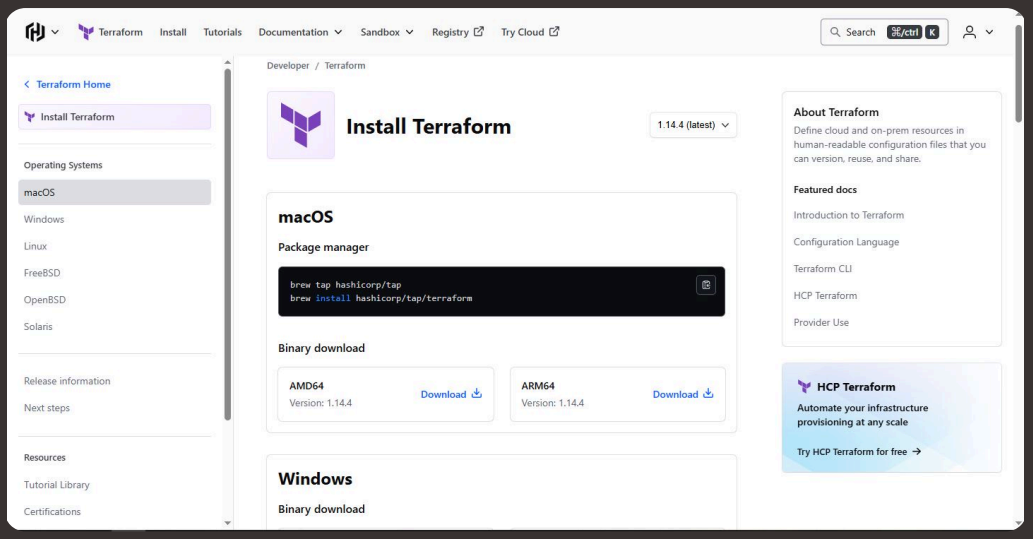
Terraform is a tool that lets you build and manage cloud infrastructure using code instead of clicking through the AWS console. You write simple configuration files that describe the resources you want, and Terraform creates them for you. It helps keep your setup consistent, repeatable, and easy to version control, which is why it is widely used in cloud engineering, DevOps, and security.

Infrastructure as Code is the practice of managing cloud resources using configuration files instead of clicking around in the AWS console. You describe your infrastructure in code, and tools like Terraform use that code to create and manage everything for you. This makes your setup consistent, repeatable, and easy to version control, so you can rebuild the same environment anytime and avoid manual mistakes.

main.tf is the main Terraform configuration file where I define the infrastructure I want to build. This is the file Terraform reads first, and it’s where I write the resources, providers, and settings that tell Terraform what to create in AWS. It acts as the core of the project, and everything else builds around it.

Mohammed Dawoud M...
NextWork Student

nextwork.org



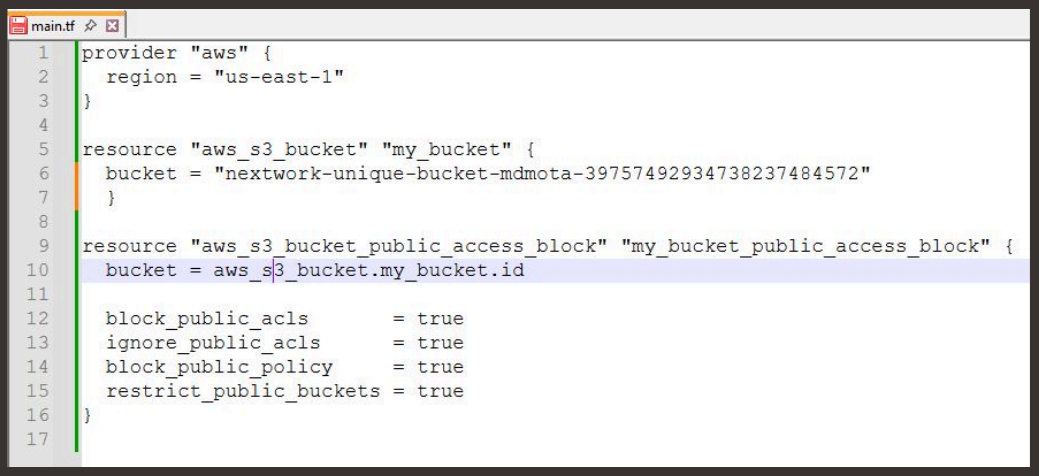
Configuration files

My configuration is defined using a .tf file where I write the code that describes the infrastructure I want Terraform to build. In this case, I'm defining the AWS provider and the resources inside main.tf. Terraform reads this file to understand what to create, update, or destroy, so everything about my setup is expressed as code in that configuration file.

My main.tf configuration has three blocks

The three blocks in my main.tf file describes the core parts of my Terraform setup. The provider block tells Terraform which cloud platform I'm using—in this case, AWS. The resource block defines the actual infrastructure I want to create, like the S3 bucket for this project. And the backend or configuration block sets up how Terraform manages its state. Together, these blocks tell Terraform what cloud to connect to, what to build, and how to track the changes it makes.

Mohammed Dawoud M...
NextWork Student



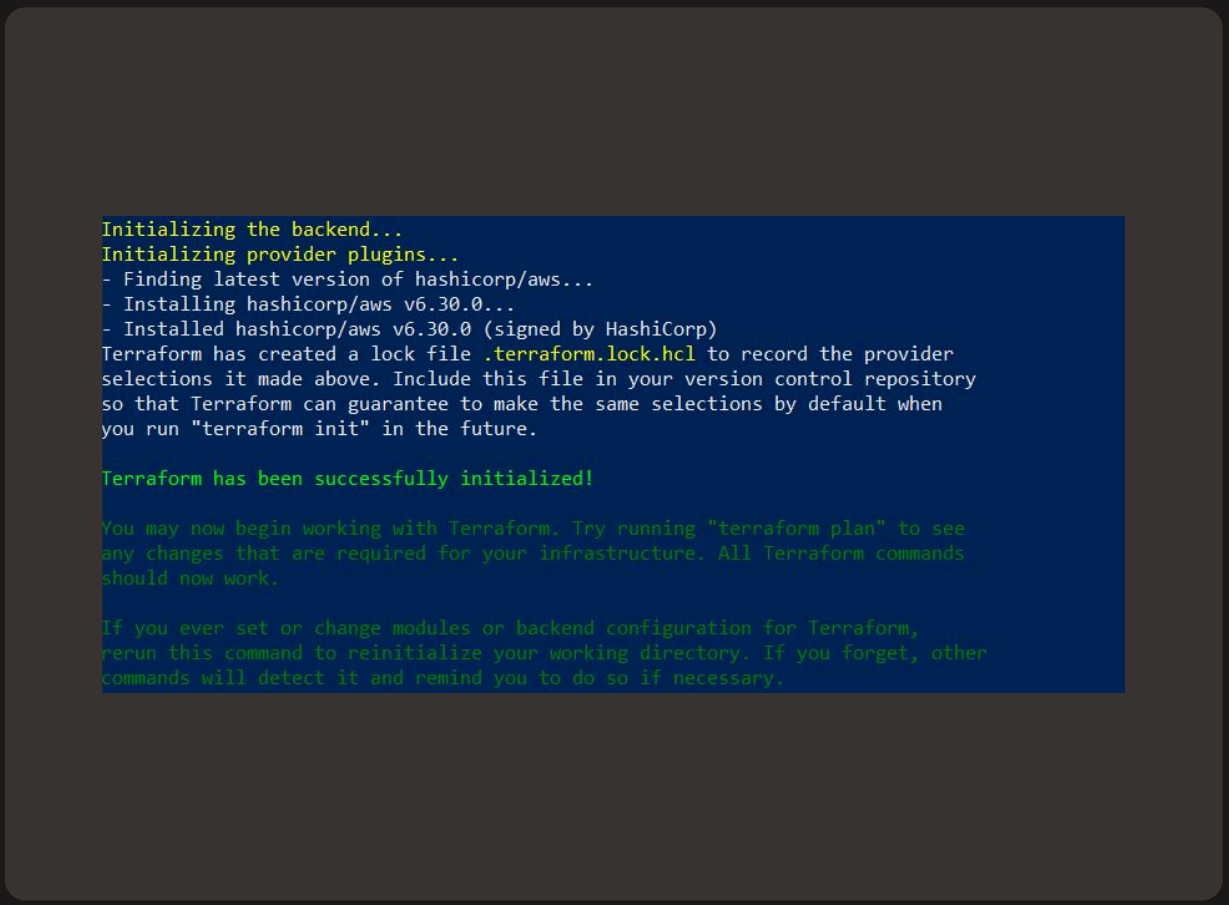
Mohammed Dawoud M…
NextWork Student

Terraform commands

terraform init sets up the working directory so Terraform can run your configuration. It downloads the necessary provider plugins, initializes the backend, and prepares the folder to manage state. Basically, it gets everything ready so Terraform can understand your code and connect to AWS.

terraform plan shows me what Terraform is going to do before it actually makes any changes. It compares my configuration to what already exists in AWS and then gives me a preview of what will be created, updated, or destroyed. It’s basically a safety check that lets me review the changes and confirm everything looks right before I apply them.

Mohammed Dawoud M...
NextWork Student



Mohammed Dawoud M...
NextWork Student

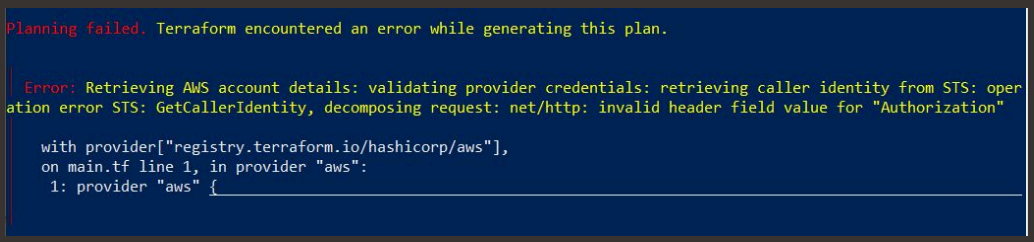
AWS CLI and Access Keys

I got an error because the AWS CLI wasn't configured in my local terminal, so Terraform couldn't authenticate to my AWS account. Terraform relies on the AWS CLI credentials to connect, and since those weren't set up in this session, the command failed. Once I configure the AWS CLI with my access keys, Terraform will be able to run without errors.

The AWS CLI is a command-line tool that lets me interact with my AWS account directly from my terminal. Instead of clicking around in the AWS console, I can run commands to configure credentials, manage resources, and connect Terraform to my account. It basically gives me a way to control AWS from my local machine using simple commands.

I set up access keys so Terraform can authenticate to my AWS account from my local machine. These keys act like credentials that let Terraform make API calls on my behalf, such as creating or managing resources. Without access keys, Terraform wouldn't be able to connect to AWS or apply any of the configuration I wrote.

Mohammed Dawoud M...
NextWork Student

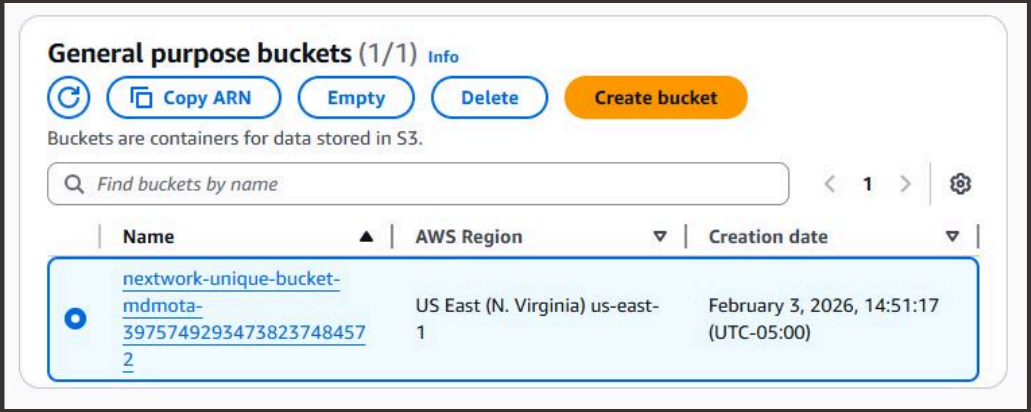


Lanching the S3 Bucket

terraform apply takes the plan that Terraform generated and actually builds the resources in AWS. It’s the command that turns my configuration into real infrastructure. After I review the changes, Terraform goes ahead and creates, updates, or deletes whatever is needed to match the code I wrote.

Terraform commands follow a specific flow because each one builds on the previous step. First, I write my configuration in .tf files. Then I run terraform init to set up the working directory and download the AWS provider. After that, I use terraform plan to preview what Terraform is going to create or change in my AWS account. Once everything looks correct, I run terraform apply to actually build the infrastructure. So the commands work together in a sequence: write the code, initialize Terraform, review the plan, and then apply the changes to AWS.

Mohammed Dawoud M...
NextWork Student



nextwork.org

The place to learn & showcase your skills

Check out nextwork.org for more projects