# This workflow installs the latest version of Terraform CLI and configures the Terraform CLI configuration file

# with an API token for Terraform Cloud (app.terraform.io). On pull request events, this workflow will run

# `terraform init`, `terraform fmt`, and `terraform plan` (speculative plan via Terraform Cloud). On push events

# to the "main" branch, `terraform apply` will be executed.

#

# Documentation for `hashicorp/setup-terraform` is located here: https://github.com/hashicorp/setup-terraform

#

# To use this workflow, you will need to complete the following setup steps.

#

# 1. Create a `main.tf` file in the root of this repository with the `remote` backend and one or more resources defined.

# Example `main.tf`:

# # The configuration for the `remote` backend.

# terraform {

# backend "remote" {

# # The name of your Terraform Cloud organization.

# organization = "example-organization"

#

# # The name of the Terraform Cloud workspace to store Terraform state files in.

# workspaces {

# name = "example-workspace"

# }

# }

# }

#

# # An example resource that does nothing.

# resource "null\_resource" "example" {

# triggers = {

# value = "A example resource that does nothing!"

# }

# }

#

#

# 2. Generate a Terraform Cloud user API token and store it as a GitHub secret (e.g. TF\_API\_TOKEN) on this repository.

# Documentation:

# - https://www.terraform.io/docs/cloud/users-teams-organizations/api-tokens.html

# - https://help.github.com/en/actions/configuring-and-managing-workflows/creating-and-storing-encrypted-secrets

#

# 3. Reference the GitHub secret in step using the `hashicorp/setup-terraform` GitHub Action.

# Example:

# - name: Setup Terraform

# uses: hashicorp/setup-terraform@v1

# with:

# cli\_config\_credentials\_token: ${{ secrets.TF\_API\_TOKEN }}

name: 'Terraform'

on:

push:

branches: [ "main" ]

pull\_request:

permissions:

contents: read

jobs:

terraform:

name: 'Terraform'

runs-on: ubuntu-latest

environment: production

# Use the Bash shell regardless whether the GitHub Actions runner is ubuntu-latest, macos-latest, or windows-latest

defaults:

run:

shell: bash

steps:

# Checkout the repository to the GitHub Actions runner

- name: Checkout

uses: actions/checkout@v4

# Install the latest version of Terraform CLI and configure the Terraform CLI configuration file with a Terraform Cloud user API token

- name: Setup Terraform

uses: hashicorp/setup-terraform@v1

with:

cli\_config\_credentials\_token: ${{ secrets.TF\_API\_TOKEN }}

# Initialize a new or existing Terraform working directory by creating initial files, loading any remote state, downloading modules, etc.

- name: Terraform Init

run: terraform init

# Checks that all Terraform configuration files adhere to a canonical format

- name: Terraform Format

run: terraform fmt -check

# Generates an execution plan for Terraform

- name: Terraform Plan

run: terraform plan -input=false

# On push to "main", build or change infrastructure according to Terraform configuration files

# Note: It is recommended to set up a required "strict" status check in your repository for "Terraform Cloud". See the documentation on "strict" required status checks for more information: https://help.github.com/en/github/administering-a-repository/types-of-required-status-checks

- name: Terraform Apply

if: github.ref == 'refs/heads/"main"' && github.event\_name == 'push'

run: terraform apply -auto-approve -input=false