Git Debugging Journey: Fixing Large File Push Errors

Introduction

This document explains the issues encountered when pushing our Terraform project to GitHub, what broke, how we fixed it, and the final outcome.

The Problem

When attempting to push our local commits to GitHub, the push was rejected with an error: 'GH001: Large files detected... The file terraform-provider-aws_v6.12.0_x5 is 740 MB.' This happened because the `.terraform` folder (which contains provider binaries) had been committed to Git history. Even after removing the files locally, they remained in the branch history, so GitHub still blocked the push.

Why This Happens

- The `.terraform/` directory stores downloaded provider plugins and should never be tracked. - Once a large file is committed, removing it in later commits doesn't erase it from history. - GitHub scans all commits being pushed and blocks any history containing files larger than 100 MB.

Debugging Steps

1. Identified the offending file: the AWS provider binary inside `.terraform/providers`. 2. Added `.terraform/` and Terraform state files to `.gitignore`. 3. Tried normal commits/removals, but push still failed because the large file was in history. 4. Attempted to use `git filter-repo` to rewrite history, but ran into environment/installation issues. 5. Finally, reset the branch against `origin/main`, staged only the desired files, and recreated a clean commit without `.terraform/` tracked.

The Fix

- Ran `git reset --mixed origin/main` to drop local commit history while keeping working files. - Added `.terraform/` and `terraform.tfstate*` to `.gitignore`. - Re-staged and committed all clean files. - Force pushed (`git push -f origin main`) to overwrite the branch on GitHub with the cleaned history.

Outcome

■ Push succeeded. The project is now on GitHub without large provider binaries. The repo contains only the files we actually need: Terraform configs, modules, scripts, docs, and site code. Terraform state and provider binaries are excluded. This ensures the repo is lightweight, compliant with GitHub's limits, and clean for future collaboration.

Lessons Learned

- Always ignore `.terraform/` and `*.tfstate` in `.gitignore`. - If a large file sneaks in, normal removal isn’t enough—you must rewrite history or reset clean. - Force pushing is sometimes required after cleaning history, but should be used carefully. - Having a clean, consistent repo makes CI/CD and collaboration much easier.