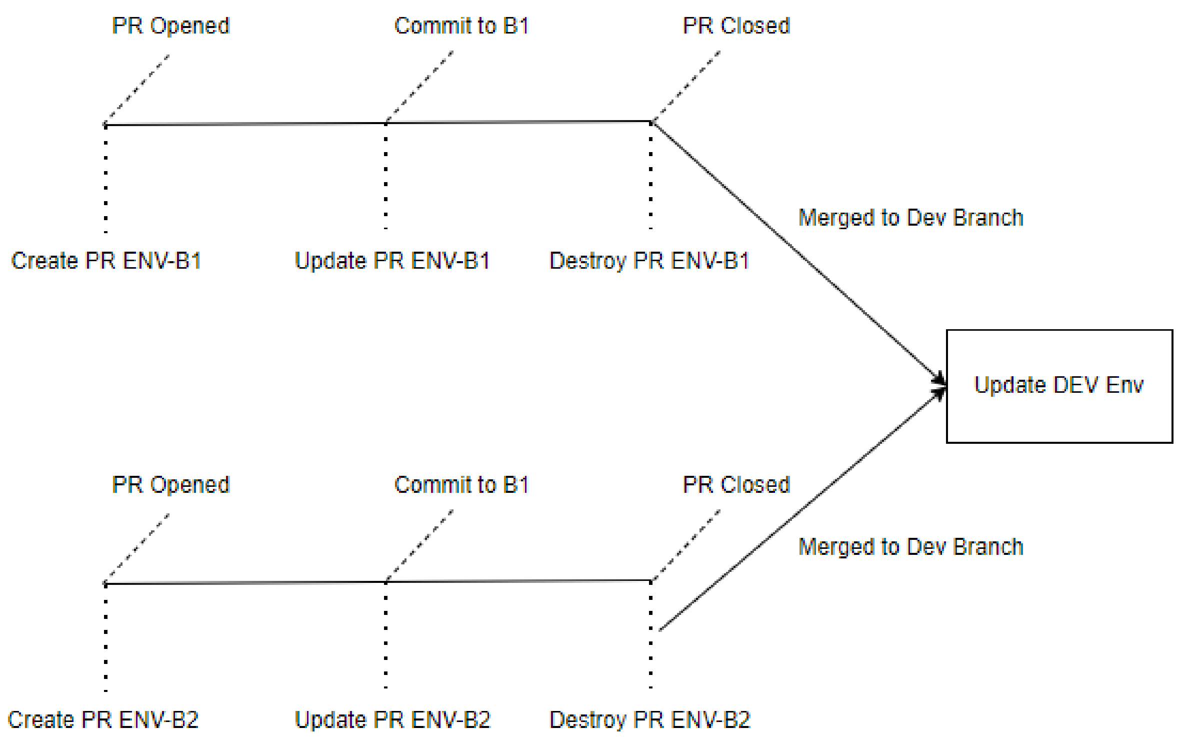

To achieve this, I would use terraform and integrate it with the CI tool.

The key point is that we need to have different state files for each created environment. It can be done with a dynamic variable which inherit from CI variables like branch name.

Also, we need a variable in our terraform configuration file to provide safety for our production environments to not destroy them (like safe_to_destroy=true/false). So, with this information, our jobs look like this,

- Create a job/action for Creating PR Environment :

We need to define when this action runs like PR is opened, reponed, edited etc.

Also, we need to provide necessary secrets and variables(branch_name.) to create a unique environment.

It's also important to state a unique backend config in this job.

When defining "Terraform Plan" step, it's crucial to define "safe_to_destroy" variable as true.

The steps sequence will be like this:

- Checkout from Git
- Build App
- Terraform Init
- Terraform Plan
- Terraform Apply
- Comment on PR with updated information

- Destroy PR Environment

Will be triggered when PR is closed.

- Checkout from Git
- Terraform Init (with differentiator of state files “branch name”)
- Terraform Destroy
- Comment on PR with “The PR Env is destroyed”

Also, there are several tools to make easier and more stable this process like env0, harness etc.

Note: Also, this rough pipeline can be differentiated by application-specific build pipelines.