Comment:
Also the local environment setup was not optimal since it does not uses volumes for easy development.

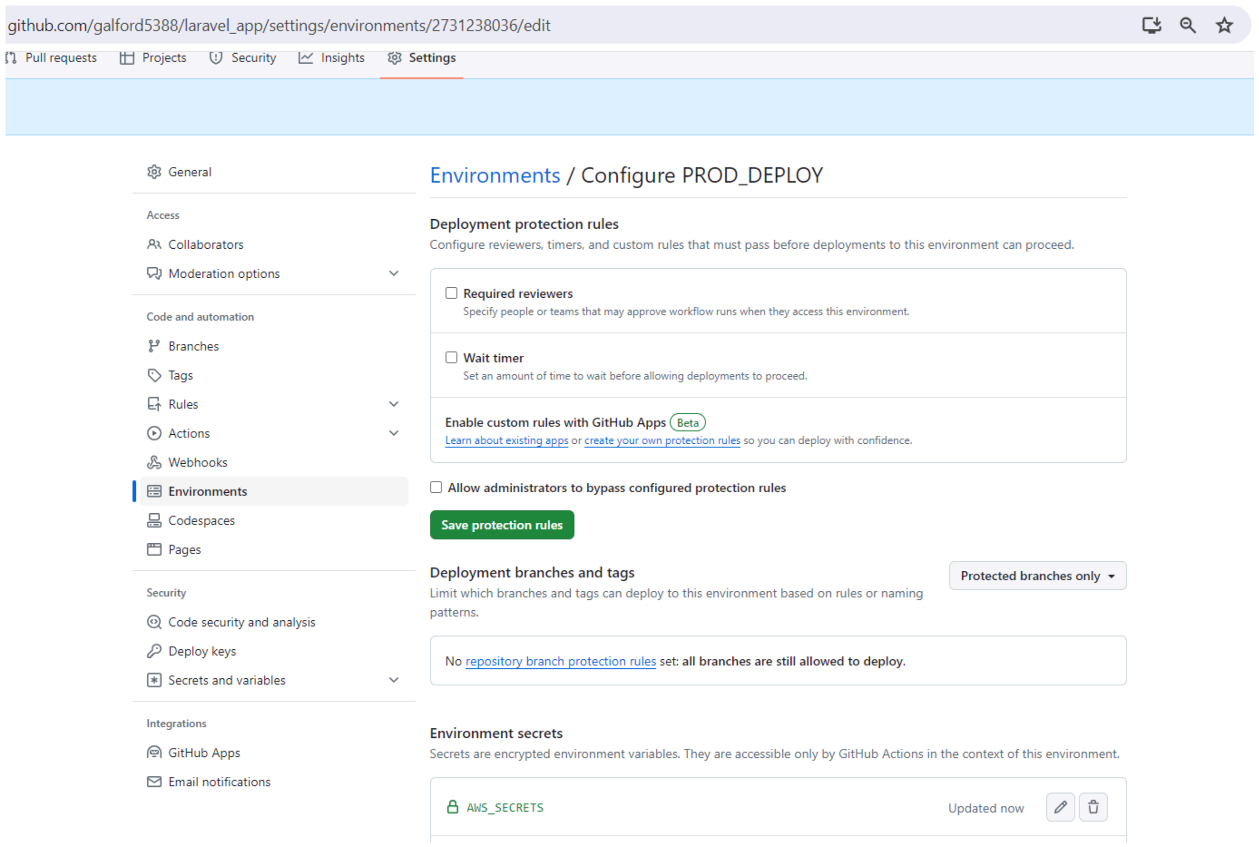
A:
It doesnt neeed to mount the volume as the Dockerfile holds the volume this is to make sure that all builds are mirror and to provide no differences between production and local development it eliminates the dependencies problem. I also updated the readme so that you can walkthrough the app and navigate within the container. Kindly read the Dockerfile.

Comment:
And Mysql 5.7 is used while this has reached EOL October 2023 and should not be used anymore.
A:
I adjusted the docker-compose.yml to use the latest and mount a volume, this needs volume as it holds persistent data over local development. Ive added the .gitignore also to exclude the database volume footprints.

Comment:
How was it delivered:
Delivery was ok, installation instructions are there but very simple and still missing some information such as where the site will be located and which other command can be used to stop the application, run tests, etc.
A:
I updated the readme kindly read it carefully

Comment:
Bonus points:
I was not able to see any of the bonus points being delivered.
A:

1. Ive added container insights, cloudwatch and logroups for app observability within infrastructure directory.
2. Ive added aws vpn client endpoint see infrastructure.
3. Automated deployment if theres a Merge request and testing is done on the status
4. Added synthetics canaries and alarms on the productions environments.
5. Added protection branch but this requires github manual configuration.


github.com/galford5388/laravel_app/settings/environments/2731238036/edit

Environments / Configure PROD_DEPLOY

Deployment protection rules

Configure reviewers, timers, and custom rules that must pass before deployments to this environment can proceed.

☐ Required reviewers
Specify people or teams that may approve workflow runs when they access this environment.

☐ Wait timer
Set an amount of time to wait before allowing deployments to proceed.

Enable custom rules with GitHub Apps (Beta)
Learn about existing apps or create your own protection rules so you can deploy with confidence.

☐ Allow administrators to bypass configured protection rules

Save protection rules

Deployment branches and tags — Protected branches only
Limit which branches and tags can deploy to this environment based on rules or naming patterns.

No repository branch protection rules set: all branches are still allowed to deploy.

Environment secrets
Secrets are encrypted environment variables. They are accessible only by GitHub Actions in the context of this environment.

AWS_SECRETS — Updated now

Bonus points:

- Blue green deployments.
This requires separate task and deployments.
Requires careful planning.

- Seeded or reduced (production) database.

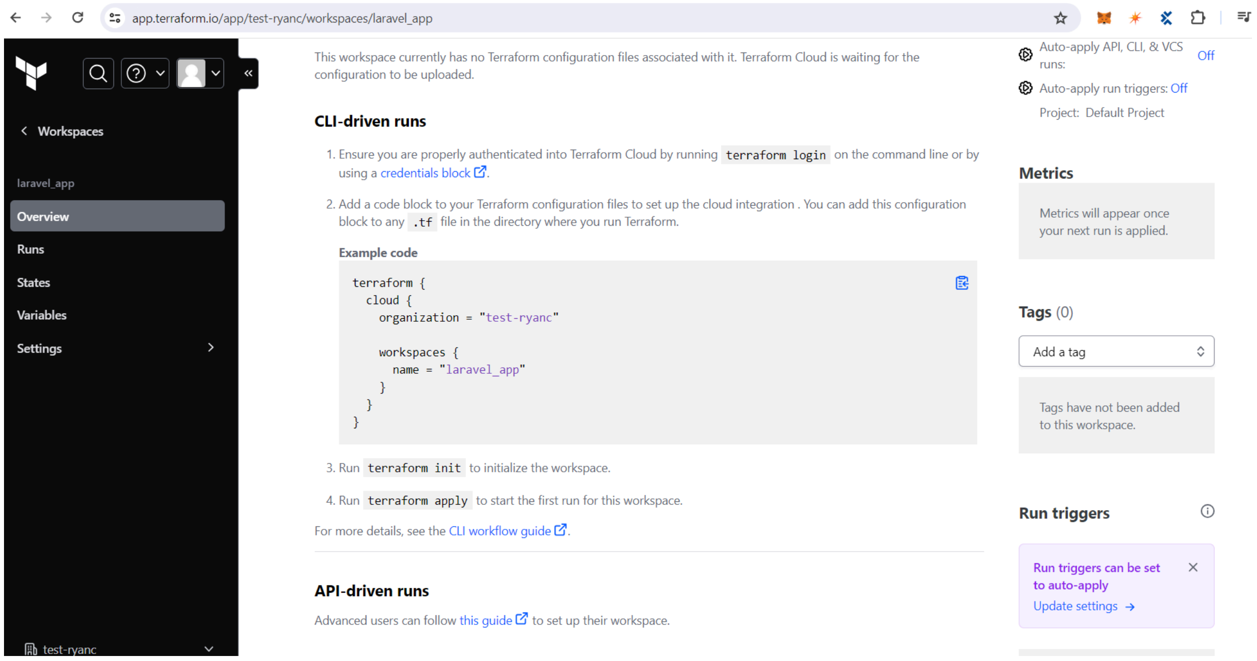
This can be done by added on the pipeline
```
php artisan make:seeder YourSeederName
```

- Automated dependency updates.
This can be manipulated in the Dockerfile
- Observability metrics.

- Automated production deployment (e.g. after MR acceptance)

See below TF Cloud Config:

Workspace Configuration


app.terraform.io/app/test-ryanc/workspaces/laravel_app

This workspace currently has no Terraform configuration files associated with it. Terraform Cloud is waiting for the configuration to be uploaded.

CLI-driven runs

1. Ensure you are properly authenticated into Terraform Cloud by running `terraform login` on the command line or by using a credentials block.
2. Add a code block to your Terraform configuration files to set up the cloud integration . You can add this configuration block to any `.tf` file in the directory where you run Terraform.

Example code

```
terraform {
  cloud {
    organization = "test-ryanc"

    workspaces {
      name = "laravel_app"
    }
  }
}
```

3. Run `terraform init` to initialize the workspace.
4. Run `terraform apply` to start the first run for this workspace.

For more details, see the CLI workflow guide.

API-driven runs

Advanced users can follow this guide to set up their workspace.

Auto-apply API, CLI, & VCS runs: Off

Auto-apply run triggers: Off

Project: Default Project

Metrics

Metrics will appear once your next run is applied.

Tags (0)

Add a tag

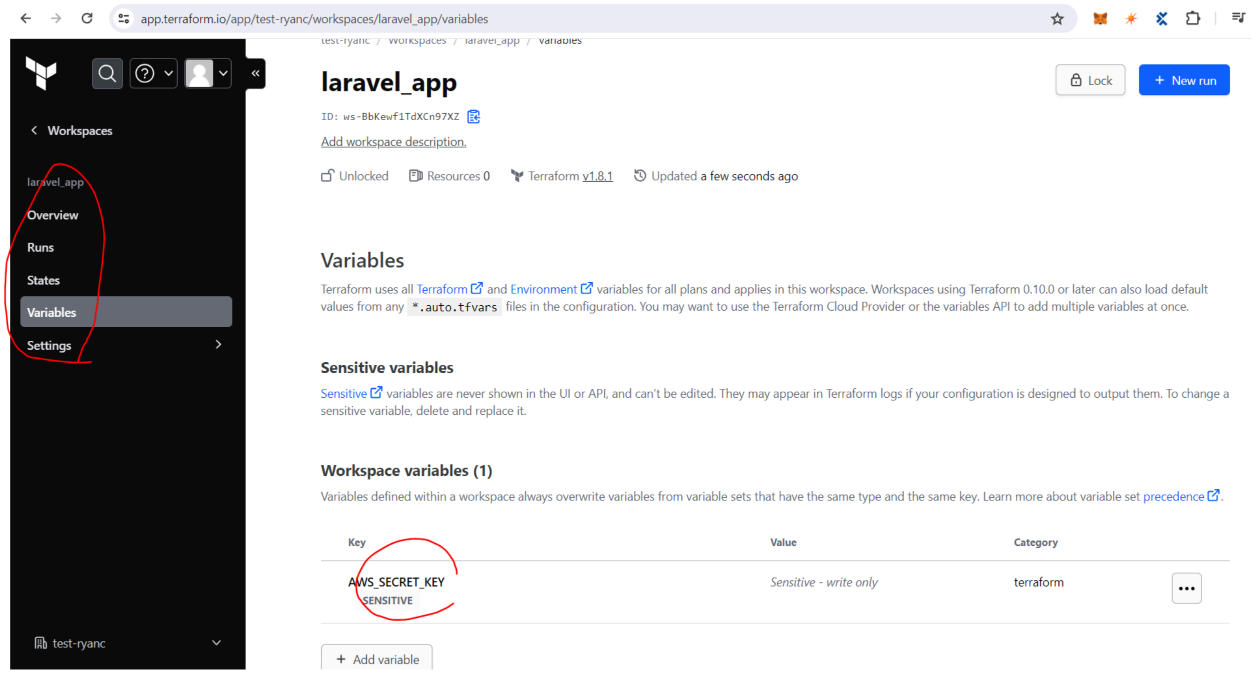
Tags have not been added to this workspace.

Run triggers

Run triggers can be set to auto-apply

Update settings →

Adding Variables:


app.terraform.io/app/test-ryanc/workspaces/laravel_app/variables

laravel_app

ID: ws-BbKewf1TdXCn97XZ

Add workspace description.

Unlocked   Resources 0   Terraform v1.8.1   Updated a few seconds ago

Variables

Terraform uses all Terraform and Environment variables for all plans and applies in this workspace. Workspaces using Terraform 0.10.0 or later can also load default values from any `*.auto.tfvars` files in the configuration. You may want to use the Terraform Cloud Provider or the variables API to add multiple variables at once.

Sensitive variables

Sensitive variables are never shown in the UI or API, and can't be edited. They may appear in Terraform logs if your configuration is designed to output them. To change a sensitive variable, delete and replace it.

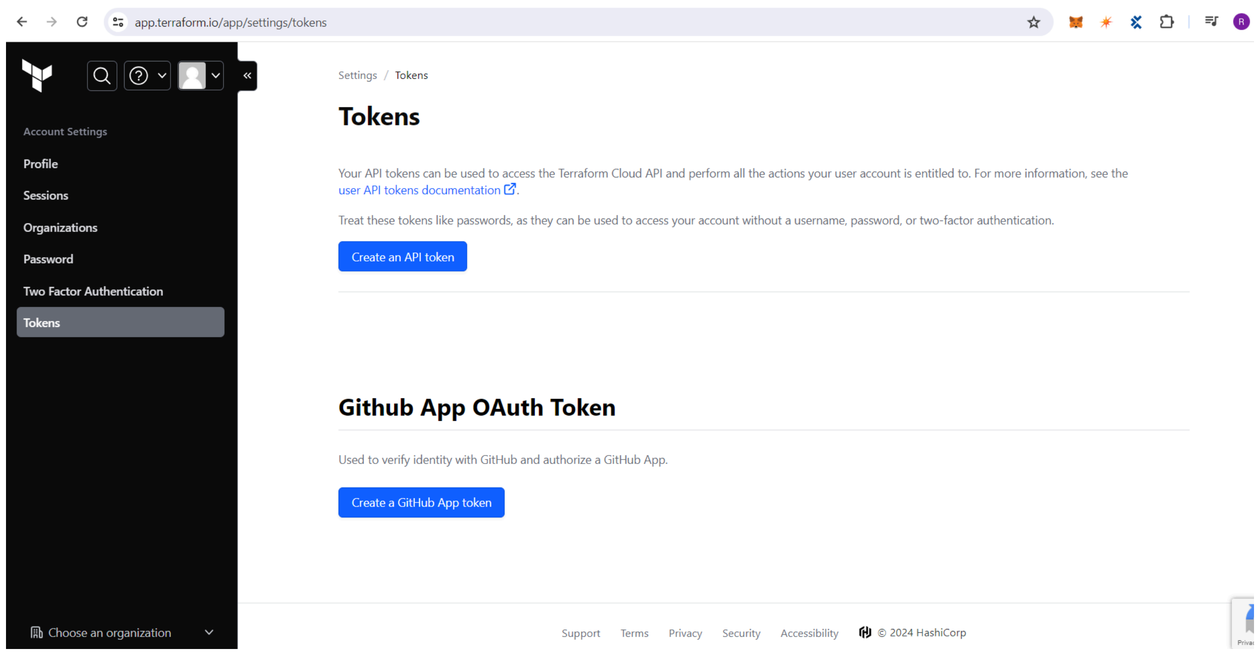
Workspace variables (1)

Variables defined within a workspace always overwrite variables from variable sets that have the same type and the same key. Learn more about variable set precedence.

Key	Value	Category
AWS_SECRET_KEY SENSITIVE	*Sensitive - write only*	terraform

+ Add variable

Creating an API Token


app.terraform.io/app/settings/tokens

Settings / Tokens

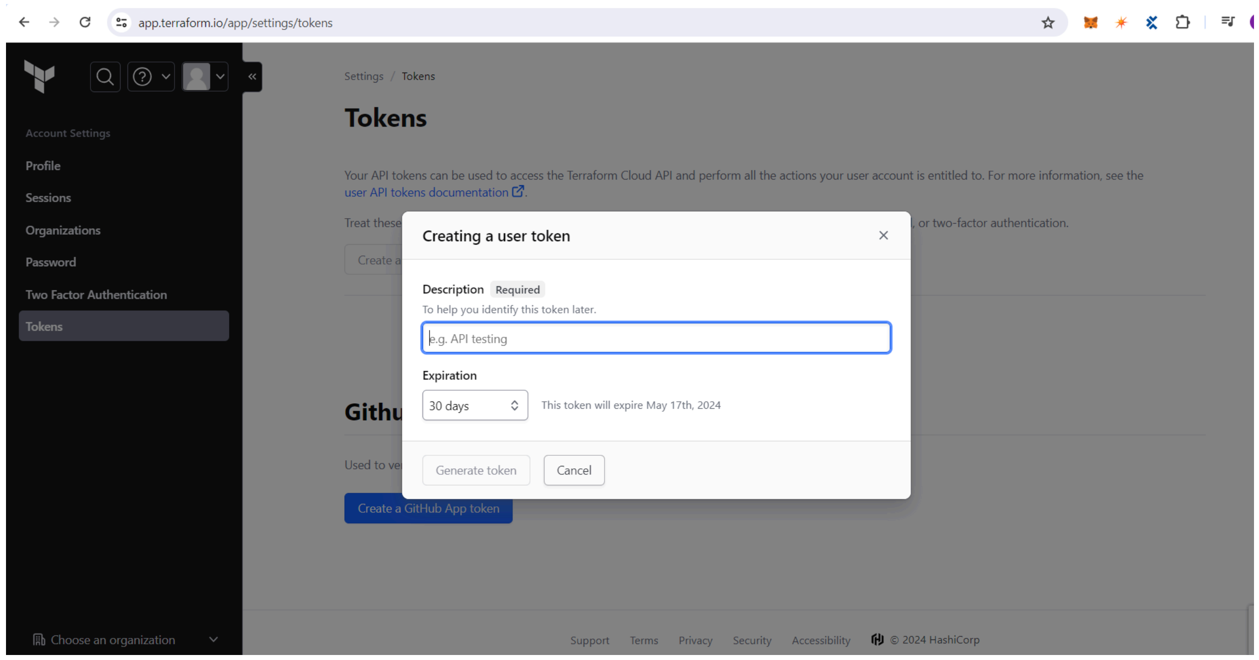
Tokens

Your API tokens can be used to access the Terraform Cloud API and perform all the actions your user account is entitled to. For more information, see the user API tokens documentation.

Treat these tokens like passwords, as they can be used to access your account without a username, password, or two-factor authentication.

Github App OAuth Token

Used to verify identity with GitHub and authorize a GitHub App.


app.terraform.io/app/settings/tokens

Creating a user token

Description Required

To help you identify this token later.

e.g. API testing

Expiration

30 days

This token will expire May 17th, 2024

After Token creation paste it in your github
${{ secrets.COMPANY_SVC_DEV_TF_API_TOKEN }}
This can be found on the .github folder each of the environment have separate token Dev, uat, prod .