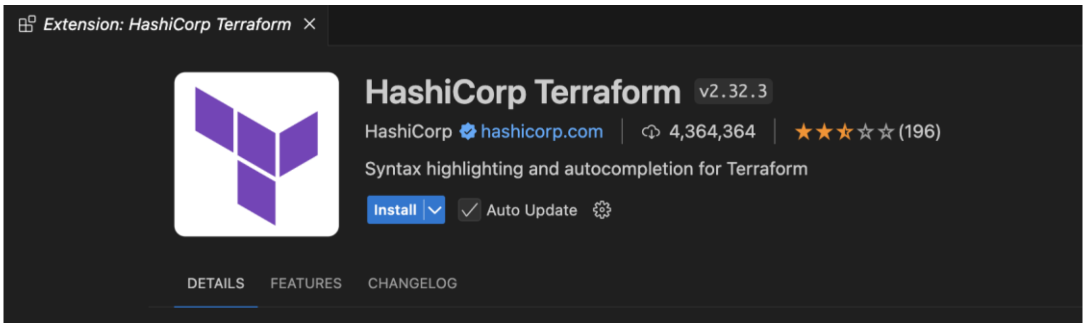
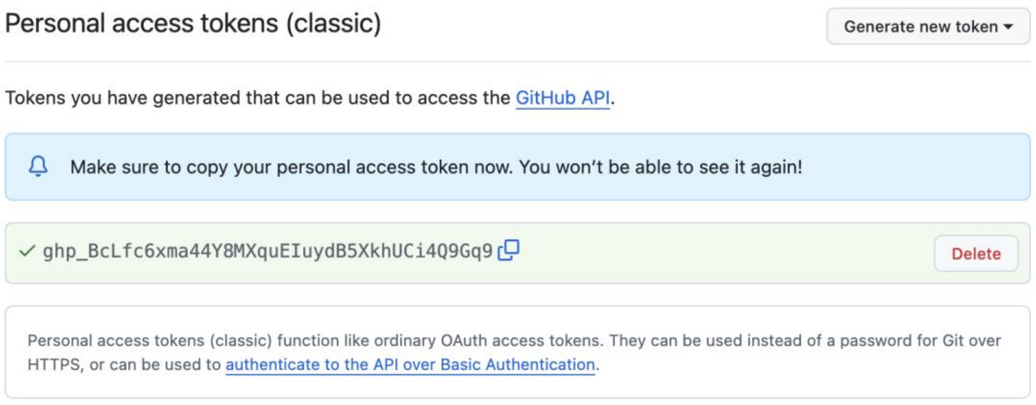
1: Install HashiCorp Terraform extension in VS Code.



Extension: HashiCorp Terraform
HashiCorp Terraform v2.32.3
HashiCorp hashicorp.com 4,364,364 (196)
Syntax highlighting and autocompletion for Terraform
Install
Auto Update
DETAILS
FEATURES
CHANGELOG

2. Obtain the github token from your github account settings and grant all necessary permissions.



Personal access tokens (classic)

Generate new token

Tokens you have generated that can be used to access the GitHub API.

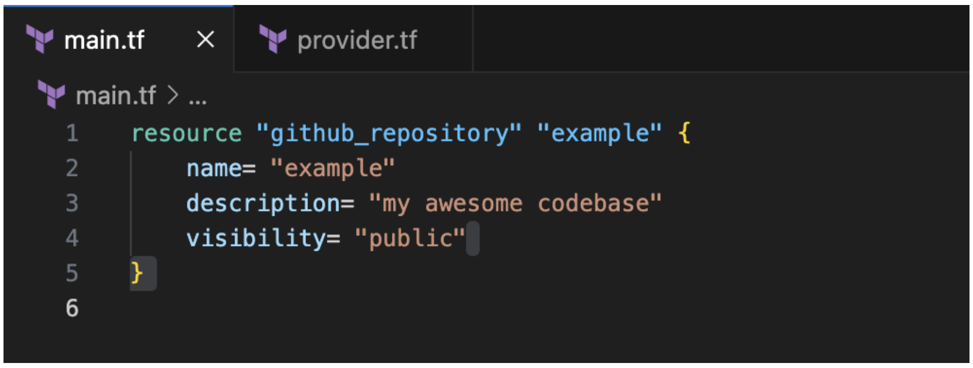
Make sure to copy your personal access token now. You won't be able to see it again!

✓ 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

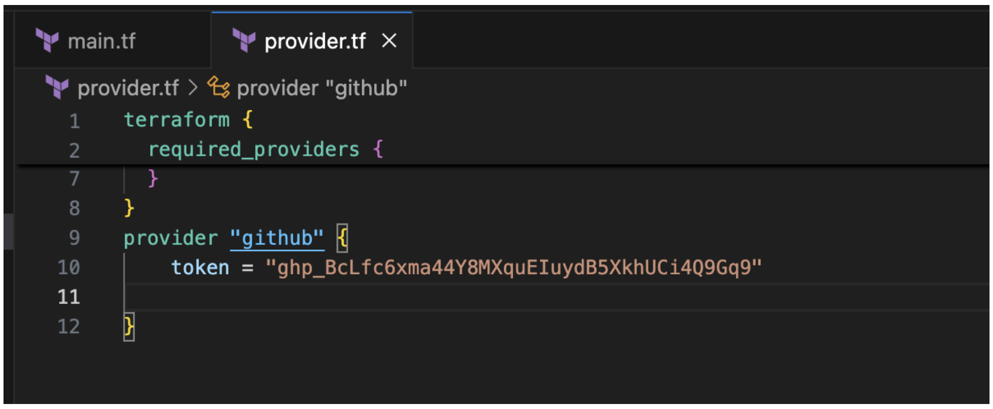
Delete

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

3. Write the code in main.tf and provider.tf

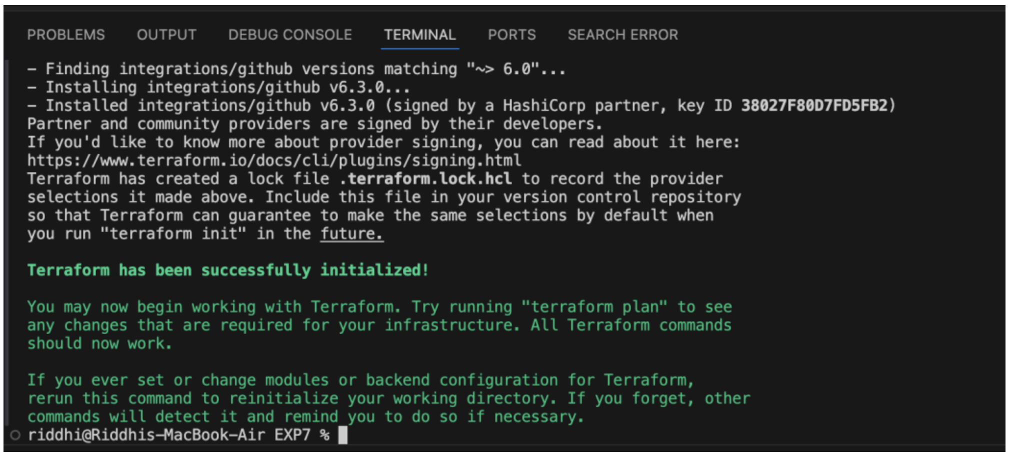


```
resource "github_repository" "example" {
    name= "example"
    description= "my awesome codebase"
    visibility= "public"
}
```



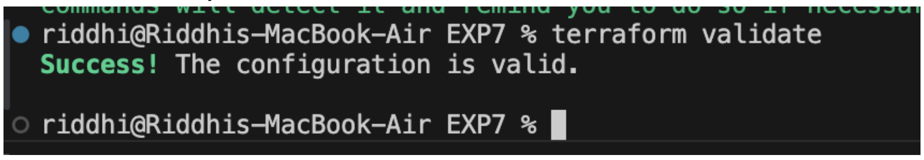
```
terraform {
  required_providers {
  }
}
provider "github" {
    token = "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"

}
```

4. In terminal, perform terraform init command.



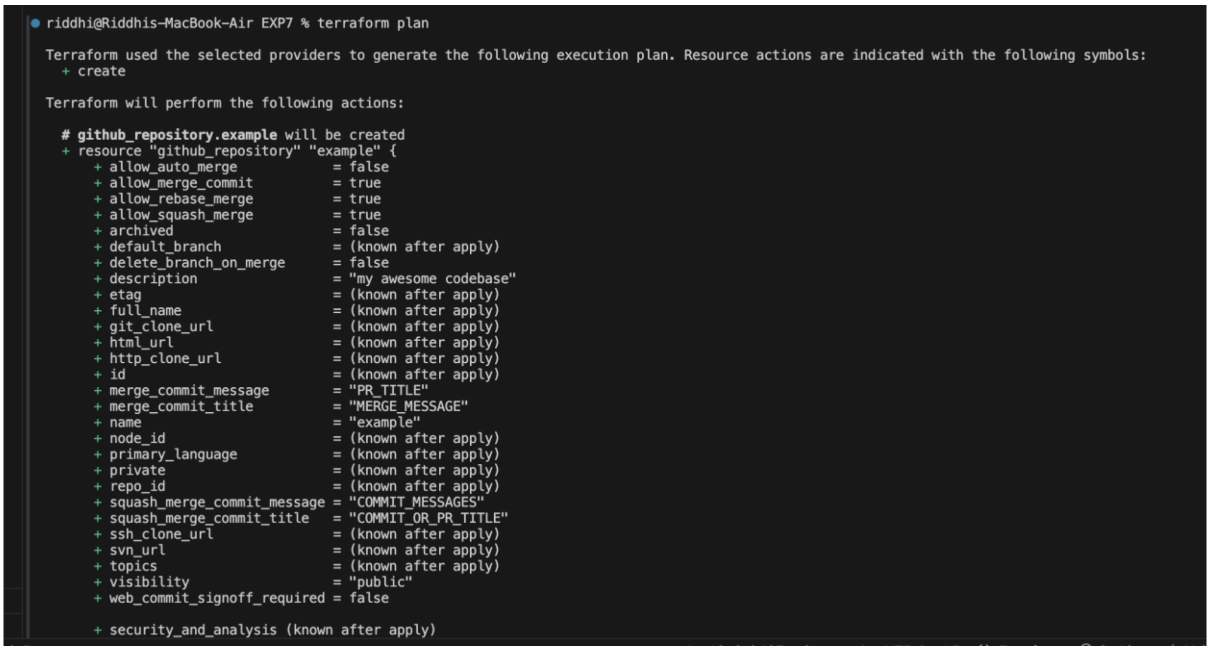
```
- Finding integrations/github versions matching "~> 6.0"...
- Installing integrations/github v6.3.0...
- Installed integrations/github v6.3.0 (signed by a HashiCorp partner, key ID 38027F80D7FD5FB2)
Partner and community providers are signed by their developers.
If you'd like to know more about provider signing, you can read about it here:
https://www.terraform.io/docs/cli/plugins/signing.html
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
riddhi@Riddhis-MacBook-Air EXP7 %
```

5. In terminal, perform terraform validate command.



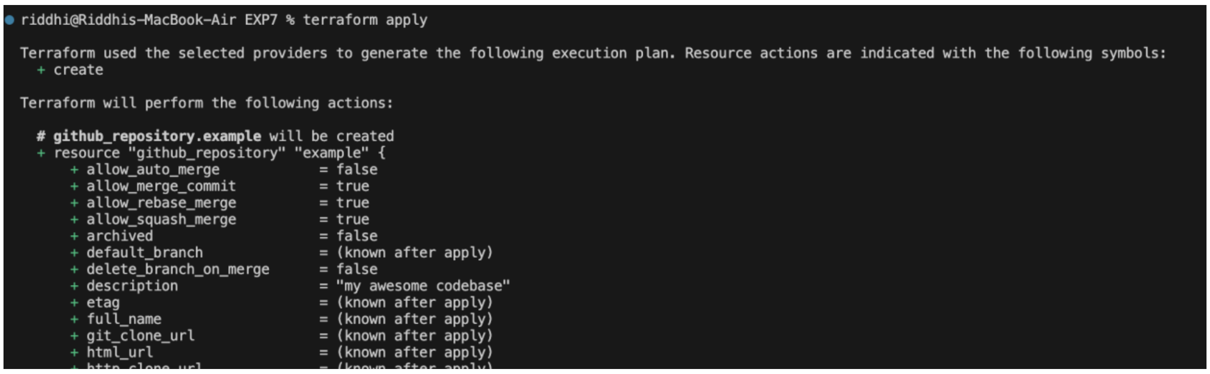
```
riddhi@Riddhis-MacBook-Air EXP7 % terraform validate
Success! The configuration is valid.

riddhi@Riddhis-MacBook-Air EXP7 %
```

6. In terminal, perform terraform plan command.

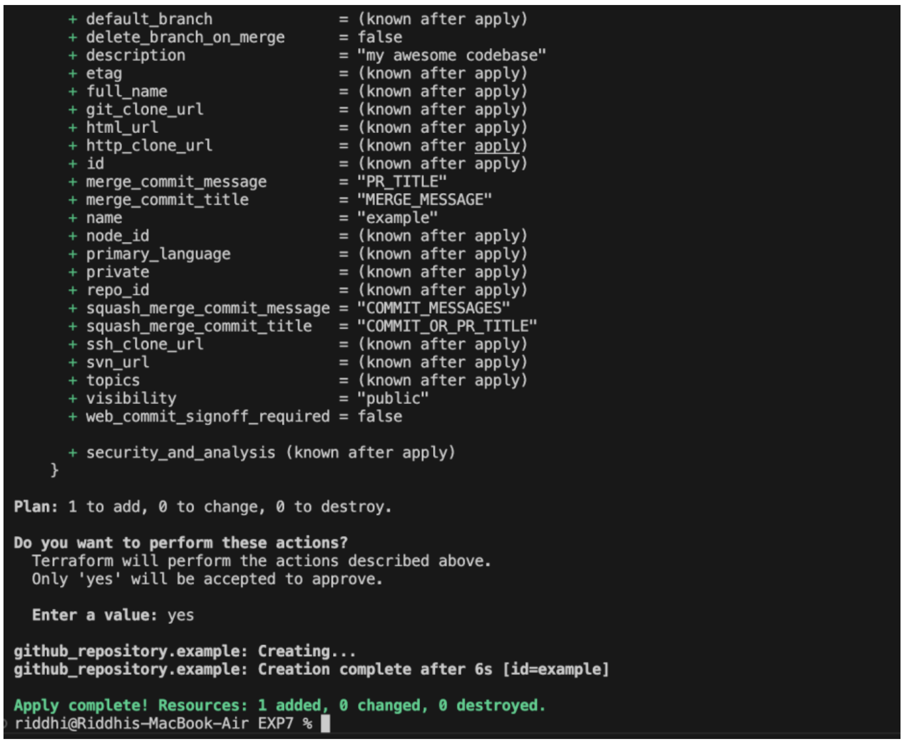


```
riddhi@Riddhis-MacBook-Air EXP7 % terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # github_repository.example will be created
  + resource "github_repository" "example" {
      + allow_auto_merge            = false
      + allow_merge_commit          = true
      + allow_rebase_merge          = true
      + allow_squash_merge          = true
      + archived                    = false
      + default_branch              = (known after apply)
      + delete_branch_on_merge      = false
      + description                 = "my awesome codebase"
      + etag                        = (known after apply)
      + full_name                   = (known after apply)
      + git_clone_url               = (known after apply)
      + html_url                    = (known after apply)
      + http_clone_url              = (known after apply)
      + id                          = (known after apply)
      + merge_commit_message        = "PR_TITLE"
      + merge_commit_title          = "MERGE_MESSAGE"
      + name                        = "example"
      + node_id                     = (known after apply)
      + primary_language            = (known after apply)
      + private                     = (known after apply)
      + repo_id                     = (known after apply)
      + squash_merge_commit_message = "COMMIT_MESSAGES"
      + squash_merge_commit_title   = "COMMIT_OR_PR_TITLE"
      + ssh_clone_url               = (known after apply)
      + svn_url                     = (known after apply)
      + topics                      = (known after apply)
      + visibility                  = "public"
      + web_commit_signoff_required = false

      + security_and_analysis (known after apply)
```

7. In terminal, perform terraform apply command.

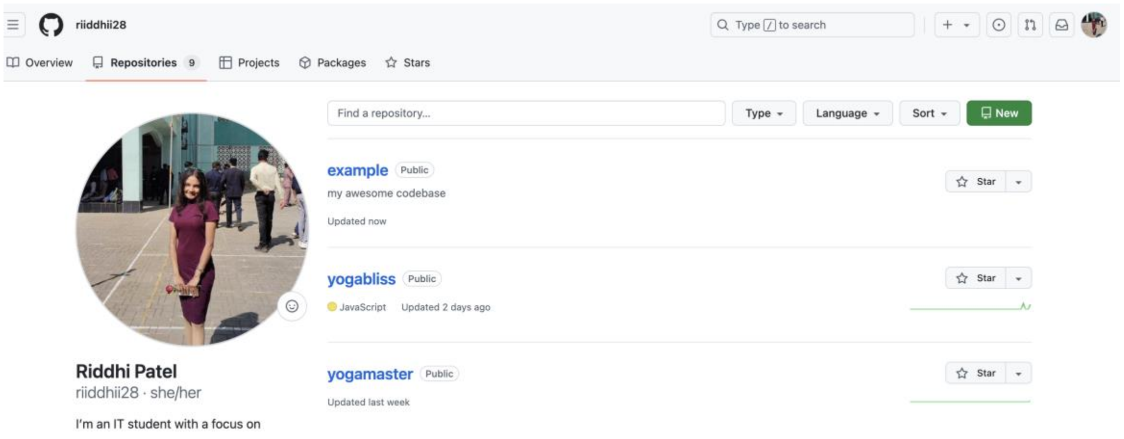


```
riddhi@Riddhis-MacBook-Air EXP7 % terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # github_repository.example will be created
  + resource "github_repository" "example" {
      + allow_auto_merge            = false
      + allow_merge_commit          = true
      + allow_rebase_merge          = true
      + allow_squash_merge          = true
      + archived                    = false
      + default_branch              = (known after apply)
      + delete_branch_on_merge      = false
      + description                 = "my awesome codebase"
      + etag                        = (known after apply)
      + full_name                   = (known after apply)
      + git_clone_url               = (known after apply)
      + html_url                    = (known after apply)
      + http_clone_url              = (known after apply)
```



```
      + default_branch              = (known after apply)
      + delete_branch_on_merge      = false
      + description                 = "my awesome codebase"
      + etag                        = (known after apply)
      + full_name                   = (known after apply)
      + git_clone_url               = (known after apply)
      + html_url                    = (known after apply)
      + http_clone_url              = (known after apply)
      + id                          = (known after apply)
      + merge_commit_message        = "PR_TITLE"
      + merge_commit_title          = "MERGE_MESSAGE"
      + name                        = "example"
      + node_id                     = (known after apply)
      + primary_language            = (known after apply)
      + private                     = (known after apply)
      + repo_id                     = (known after apply)
      + squash_merge_commit_message = "COMMIT_MESSAGES"
      + squash_merge_commit_title   = "COMMIT_OR_PR_TITLE"
      + ssh_clone_url               = (known after apply)
      + svn_url                     = (known after apply)
      + topics                      = (known after apply)
      + visibility                  = "public"
      + web_commit_signoff_required = false

      + security_and_analysis (known after apply)
    }

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

github_repository.example: Creating...
github_repository.example: Creation complete after 6s [id=example]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
riddhi@Riddhis-MacBook-Air EXP7 % 
```

8. Repository is added to github.



Riddhi Patel
riiddhii28 · she/her

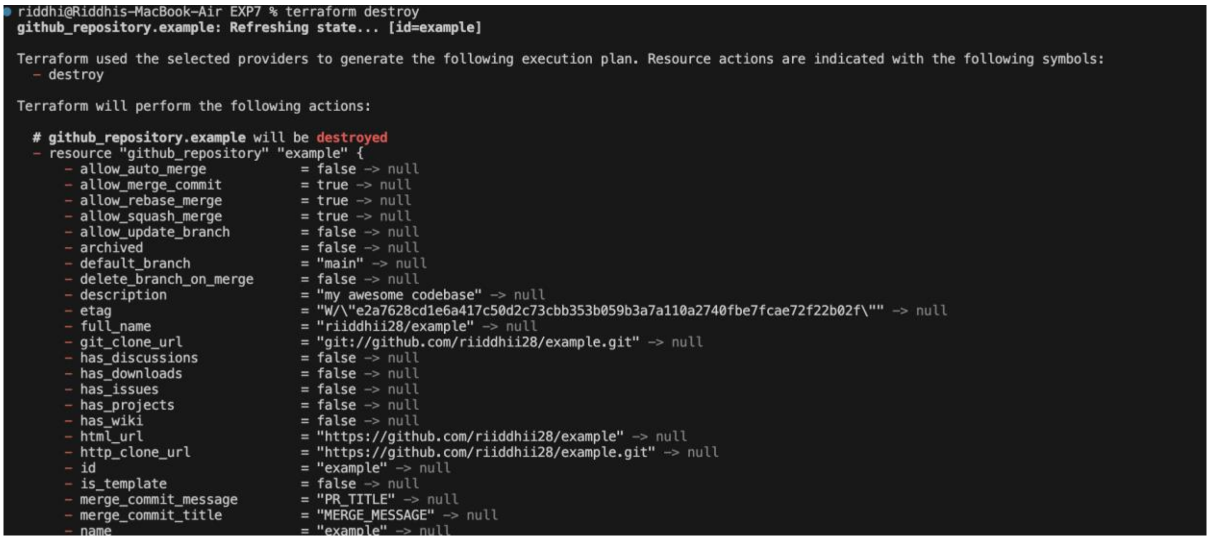
I'm an IT student with a focus on

example Public
my awesome codebase
Updated now

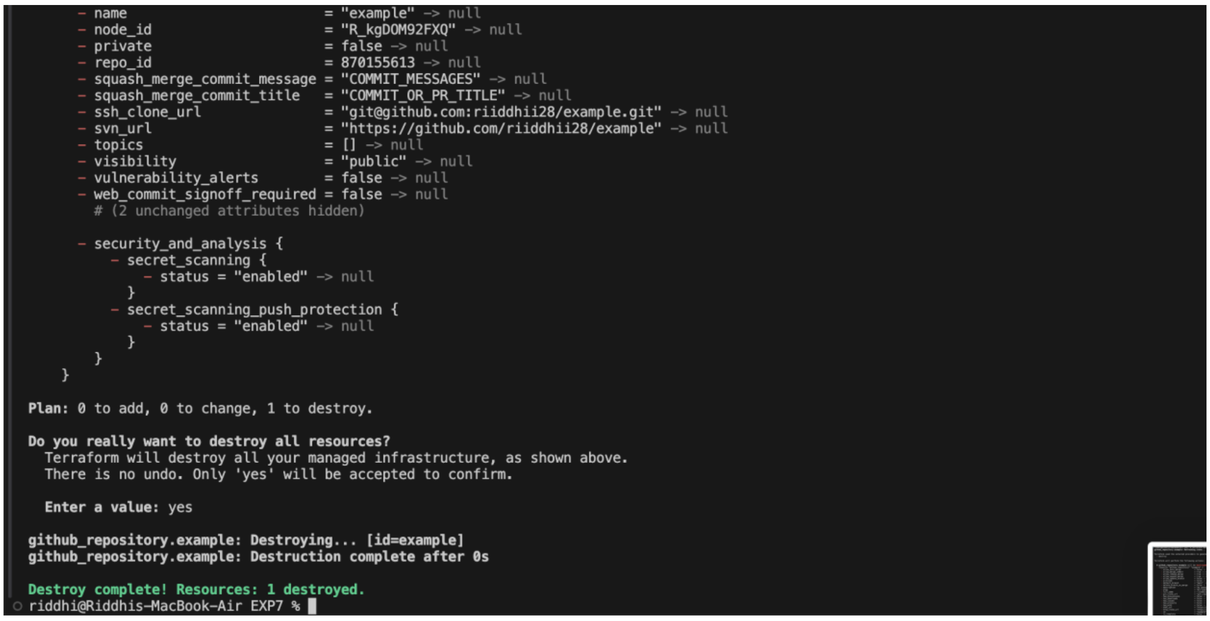
yogabliss Public
JavaScript Updated 2 days ago

yogamaster Public
Updated last week

9. To delete repository, perform terraform destroy command.

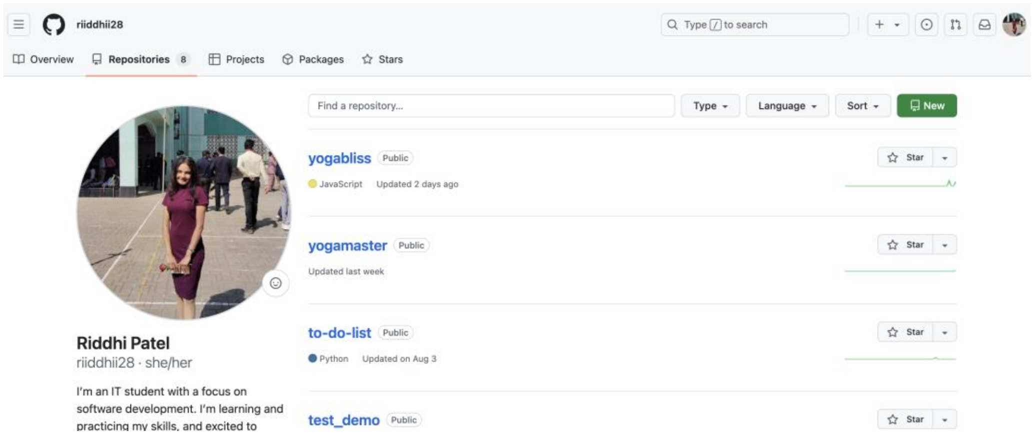


```
riddhi@Riddhis-MacBook-Air EXP7 % terraform destroy
github_repository.example: Refreshing state... [id=example]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

  # github_repository.example will be destroyed
  - resource "github_repository" "example" {
      - allow_auto_merge            = false -> null
      - allow_merge_commit          = true -> null
      - allow_rebase_merge          = true -> null
      - allow_squash_merge          = true -> null
      - allow_update_branch         = false -> null
      - archived                    = false -> null
      - default_branch              = "main" -> null
      - delete_branch_on_merge      = false -> null
      - description                 = "my awesome codebase" -> null
      - etag                        = "W/\"e2a7628cd1e6a417c50d2c73cbb353b059b3a7a110a2740fbe7fcae72f22b02f\"" -> null
      - full_name                   = "riiddhii28/example" -> null
      - git_clone_url               = "git://github.com/riiddhii28/example.git" -> null
      - has_discussions             = false -> null
      - has_downloads               = false -> null
      - has_issues                  = false -> null
      - has_projects                = false -> null
      - has_wiki                    = false -> null
      - html_url                    = "https://github.com/riiddhii28/example" -> null
      - http_clone_url              = "https://github.com/riiddhii28/example.git" -> null
      - id                          = "example" -> null
      - is_template                 = false -> null
      - merge_commit_message        = "PR_TITLE" -> null
      - merge_commit_title          = "MERGE_MESSAGE" -> null
      - name                        = "example" -> null
```



```
      - name                        = "example" -> null
      - node_id                     = "R_kgDOM92FXQ" -> null
      - private                     = false -> null
      - repo_id                     = 870155613 -> null
      - squash_merge_commit_message = "COMMIT_MESSAGES" -> null
      - squash_merge_commit_title   = "COMMIT_OR_PR_TITLE" -> null
      - ssh_clone_url               = "git@github.com:riiddhii28/example.git" -> null
      - svn_url                     = "https://github.com/riiddhii28/example" -> null
      - topics                      = [] -> null
      - visibility                  = "public" -> null
      - vulnerability_alerts        = false -> null
      - web_commit_signoff_required = false -> null
        # (2 unchanged attributes hidden)

      - security_and_analysis {
          - secret_scanning {
              - status = "enabled" -> null
            }
          - secret_scanning_push_protection {
              - status = "enabled" -> null
            }
        }
    }

Plan: 0 to add, 0 to change, 1 to destroy.

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

github_repository.example: Destroying... [id=example]
github_repository.example: Destruction complete after 0s

Destroy complete! Resources: 1 destroyed.
riddhi@Riddhis-MacBook-Air EXP7 %
```

10. Repository has been deleted.



riiddhii28

Overview · Repositories 8 · Projects · Packages · Stars

Riddhi Patel
riiddhii28 · she/her

I'm an IT student with a focus on software development. I'm learning and practicing my skills, and excited to

yogabliss Public
JavaScript · Updated 2 days ago

yogamaster Public
Updated last week

to-do-list Public
Python · Updated on Aug 3

test_demo Public