Setup branch rules in the repositories using terraform

Feature/Task JIRA	Setup branch rules in the repositories using terraform
Document Status	Completed
Assignee	Nidhi Airi
Review	Mr. Mukul Srivastava
Approval	Mr Mukul Srivastava

Description

This task is related to set up branching rules for your repository using terraform

Findings/Prerequisite

The given task requires some prerequisites that are as follows:

- The terraform needs to be installed in your system
- Then the operator must have an access token of the organization

Strategy

- First we will write main.tf file that will be entry point in order to create branching rules
- Then we will run the following commands:

```
terraform init
terraform plan
```

```
3  terraform apply
```

➢ Then we will look at our github organization and we will look at whether we were able to set up the branch rules for our repositories

Changes Implemented on Present Infra

Implementation Details

- **Setting up Terraform**

At first we will install terraform in our system using the following commands

```
● wget
  https://releases.hashicorp.com/terraform/0.12.24/terraform_0.1
  2.24_linux_amd64.zip
● sudo apt-get install zip -y
● unzip terraform*.zip
● sudo mv terraform /usr/local/bin
```

- **Accessing the token**

Then we will access the token of the github organization in which repositories we want to add branch rules

- **Creation of files**

1. **First we will create main.tf file**

The configuration of main.tf file is below

```
# here the provider github is used to use github plugin using terraform
provider github {
  version      = "~> 2.1"
  organization = "${var.org_name}"
  token        = "${var.token_name}"
}
resource "github_membership" "member" {
  username     = "${var.git_user}"
  role         = "${var.git_user_role}"
}
# Add a repository to you organisation
resource "github_repository" "repository" {
  for_each             = var.repo_name
  name                 = each.value.name
  description          = "This repository is created using terraform."
  private              = false
  allow_merge_commit = true
  auto_init            = true # to produce an initial commit in the repository.
  license_template     = "mit" #(optional)
}

# Configure a branch protection for the repository
resource "github_branch_protection" "repository" {
  for_each        = var.repo_name
  repository      = each.value.name
  branch          = "${var.branch_name}" # here u need to update the branch you want to set the rules for
  enforce_admins = true # enforces status checks for repository administrators

  required_status_checks { # Require branches to be up to date before merging.
    strict    = true
    contexts = ["ci/travis"]
  }

  required_pull_request_reviews { # Enforce restrictions for pull request reviews.
    dismiss_stale_reviews = true
  }
  depends_on = [
    github_repository.repository,
  ]
}
```

2. Creation of variables.tf file

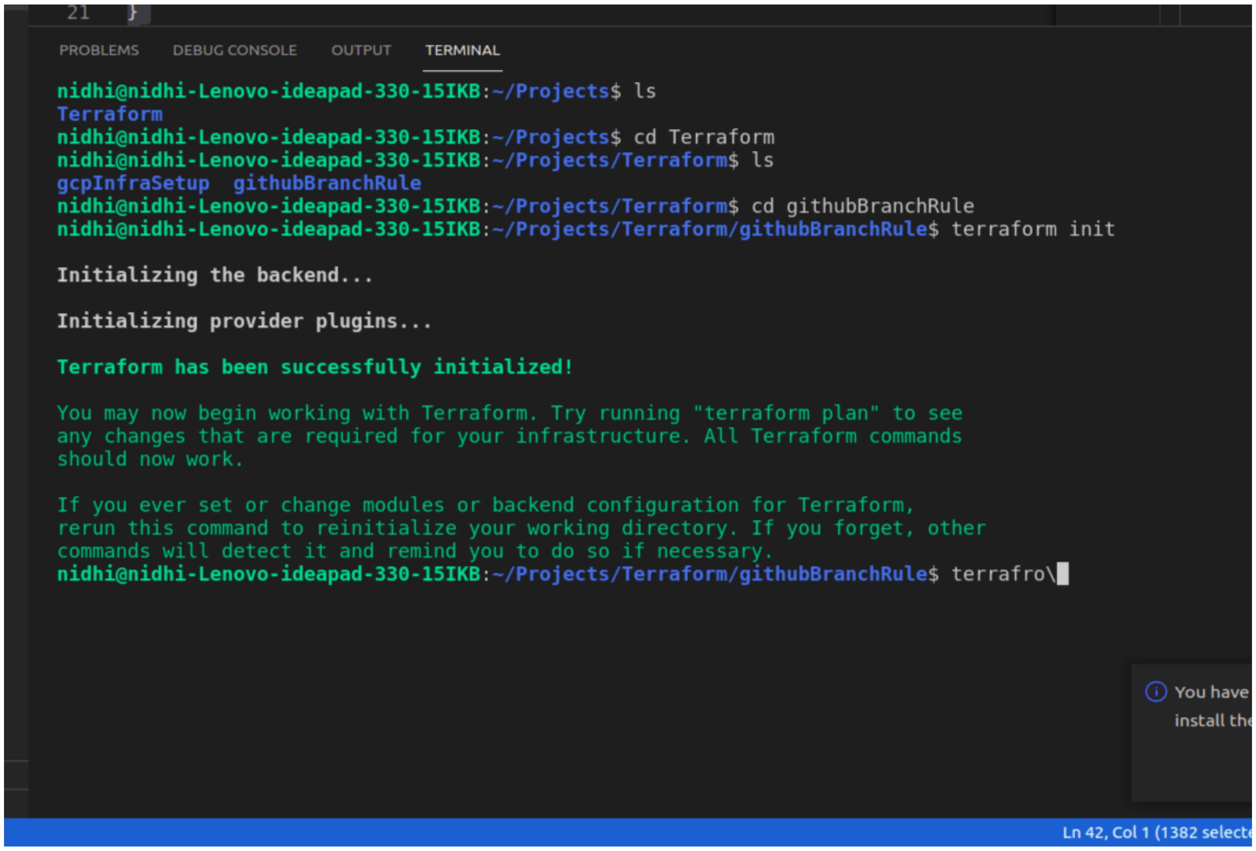
```
# here we will define our variables to be used in main.tf
variable "repo_name" {
    default = {
        "one" = {
            "name" = " add repostory name here"
        },
        "two" = {
            "name" = " add other repostory name here",
        }

    }
}
variable "branch_name"{
    type    =  string
    default = " add branch name here "
}


```

```
variable "org_name"{
    type    = string
    default = " add organisation name here"
}

variable "token_name"{
    type    = string
    default =  "add token here"
}


variable "git_user"{
    type    = string
    default = "add username here"
}

variable "git_user_role"{
    type    = string
    default = "add user role here"
}

```

- After successfully creating the two files we will run the following terraform commands

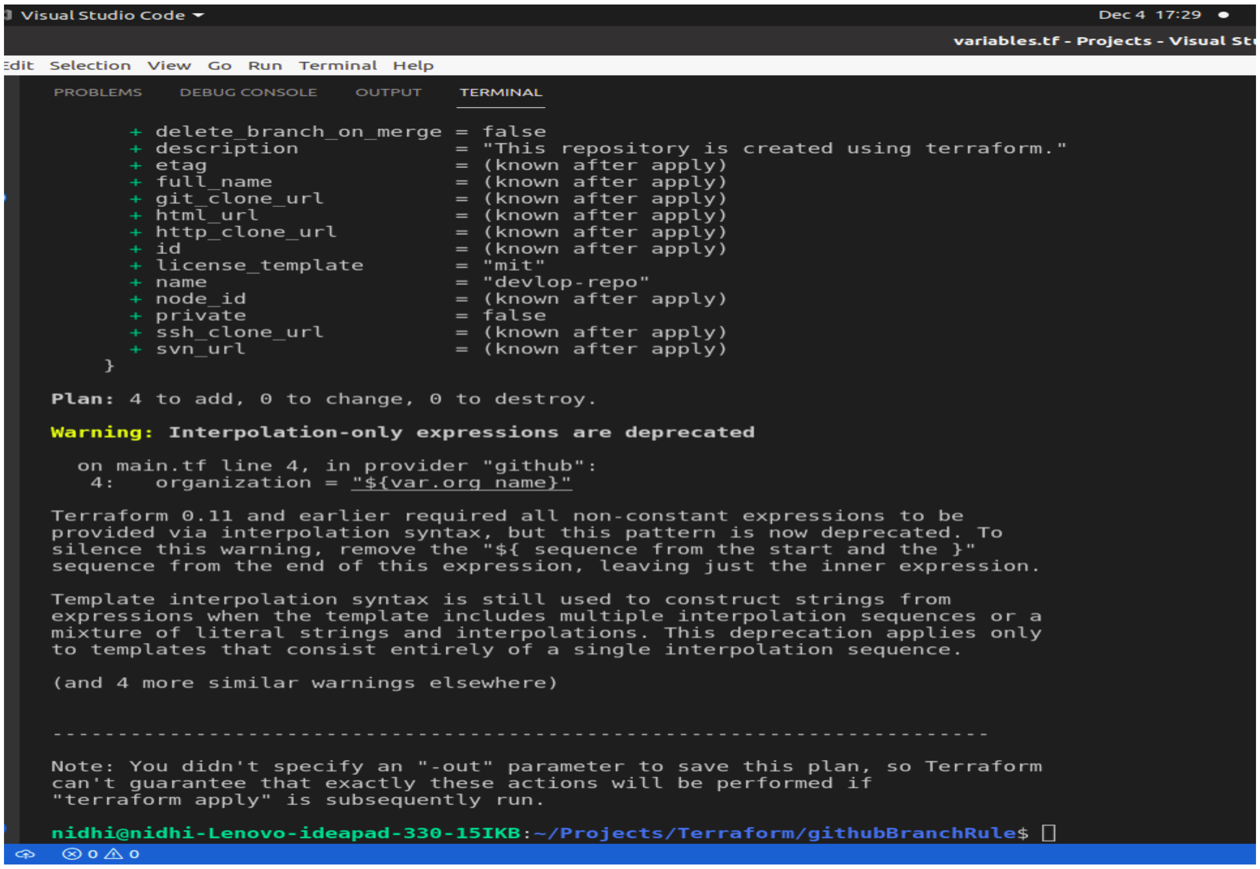
1. **terraform init**

It is used to initialize a working directory containing necessary provider plugins.



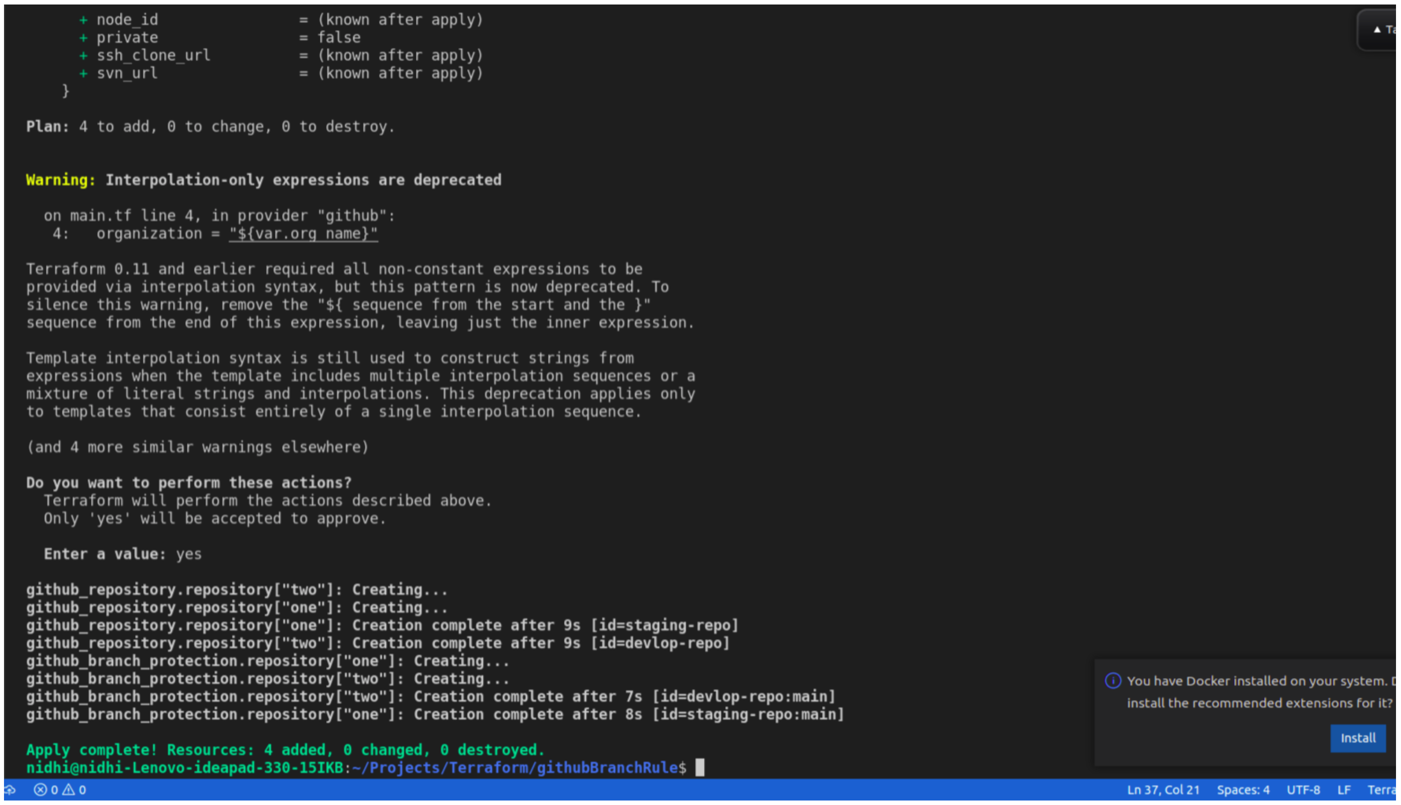
```
nidhi@nidhi-Lenovo-ideapad-330-15IKB:~/Projects$ ls
Terraform
nidhi@nidhi-Lenovo-ideapad-330-15IKB:~/Projects$ cd Terraform
nidhi@nidhi-Lenovo-ideapad-330-15IKB:~/Projects/Terraform$ ls
gcpInfraSetup  githubBranchRule
nidhi@nidhi-Lenovo-ideapad-330-15IKB:~/Projects/Terraform$ cd githubBranchRule
nidhi@nidhi-Lenovo-ideapad-330-15IKB:~/Projects/Terraform/githubBranchRule$ terraform init

Initializing the backend...

Initializing provider plugins...

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.
nidhi@nidhi-Lenovo-ideapad-330-15IKB:~/Projects/Terraform/githubBranchRule$ terrafro\
```

2. **terraform plan** It basically creates an execution plan


Visual Studio Code
Dec 4 17:29
variables.tf - Projects - Visual St
Edit Selection View Go Run Terminal Help
PROBLEMS DEBUG CONSOLE OUTPUT TERMINAL

```
      + delete_branch_on_merge = false
      + description            = "This repository is created using terraform."
      + 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)
      + license_template       = "mit"
      + name                   = "devlop-repo"
      + node_id                = (known after apply)
      + private                = false
      + ssh_clone_url          = (known after apply)
      + svn_url                = (known after apply)
    }

Plan: 4 to add, 0 to change, 0 to destroy.

Warning: Interpolation-only expressions are deprecated

  on main.tf line 4, in provider "github":
   4:   organization = "${var.org_name}"

Terraform 0.11 and earlier required all non-constant expressions to be
provided via interpolation syntax, but this pattern is now deprecated. To
silence this warning, remove the "${ sequence from the start and the }"
sequence from the end of this expression, leaving just the inner expression.

Template interpolation syntax is still used to construct strings from
expressions when the template includes multiple interpolation sequences or a
mixture of literal strings and interpolations. This deprecation applies only
to templates that consist entirely of a single interpolation sequence.

(and 4 more similar warnings elsewhere)


------------------------------------------------------------------------

Note: You didn't specify an "-out" parameter to save this plan, so Terraform
can't guarantee that exactly these actions will be performed if
"terraform apply" is subsequently run.

nidhi@nidhi-Lenovo-ideapad-330-15IKB:~/Projects/Terraform/githubBranchRule$
```

3. **terraform apply** It applies the plan created



```
      + node_id                = (known after apply)
      + private                = false
      + ssh_clone_url          = (known after apply)
      + svn_url                = (known after apply)
    }

Plan: 4 to add, 0 to change, 0 to destroy.


Warning: Interpolation-only expressions are deprecated

  on main.tf line 4, in provider "github":
   4:   organization = "${var.org_name}"

Terraform 0.11 and earlier required all non-constant expressions to be
provided via interpolation syntax, but this pattern is now deprecated. To
silence this warning, remove the "${ sequence from the start and the }"
sequence from the end of this expression, leaving just the inner expression.

Template interpolation syntax is still used to construct strings from
expressions when the template includes multiple interpolation sequences or a
mixture of literal strings and interpolations. This deprecation applies only
to templates that consist entirely of a single interpolation sequence.

(and 4 more similar warnings elsewhere)

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.repository["two"]: Creating...
github_repository.repository["one"]: Creating...
github_repository.repository["one"]: Creation complete after 9s [id=staging-repo]
github_repository.repository["two"]: Creation complete after 9s [id=devlop-repo]
github_branch_protection.repository["one"]: Creating...
github_branch_protection.repository["two"]: Creating...
github_branch_protection.repository["two"]: Creation complete after 7s [id=devlop-repo:main]
github_branch_protection.repository["one"]: Creation complete after 8s [id=staging-repo:main]

Apply complete! Resources: 4 added, 0 changed, 0 destroyed.
nidhi@nidhi-Lenovo-ideapad-330-15IKB:~/Projects/Terraform/githubBranchRule$
```

You have Docker installed on your system.
install the recommended extensions for it?

Validating the above steps

Now we will visit our github orgamisation and check whether the required branch rules are set on the repositories

Pricing

In setting terraform

- No cost involved in installing terraform

In setting up branch rules

We have used Github free account which will allow us to run build for 2000 minutes per month and 500 MB of storage

Questions

Questions	Outcome
How can we connect the github with the terraform?	We will require the github providers plugins so that after running terraform init we will able to provide necessary plugins

	to the terraform
How to connect terraform with our organization?	By giving the token of the organization in variable.tf file
How to create repositories using terraform?	By giving the resource github_repository under main.tf file
How to set branch rules using terraform?	By giving the resource github_branch_protection under main.tf file
How to iterate over the resources?	By using the meta argument for_each inside resource block

Challenges/Blockers or Not Doing

- **Challenge 1**: We were not able to connect the terraform with the github

Resolved: We have resolved the issue by providing necessary provider github in main.tf file

- **Challenge 2:** We were not able to setup rules on branches of multiple repositories

Resolved: We have resolved this issue by using the for_each meta argument inside the resource block

Referred Links

➤ https://stackoverflow.com/questions/58594506/how-to-for-each-through-a-listobjects-in-terraform-0-12

➤ https://registry.terraform.io/providers/integrations/github/latest/docs/resources/branch

➤ https://www.mineiros.io/blog/how-to-manage-your-github-organization-with-terraform

➤ https://docs.github.com/en/organizations/collaborating-with-groups-in-organizations/creating-a-new-organization-from-scratch