

# **Empower everyone to manage Infrastructure with GitLab and Terraform**

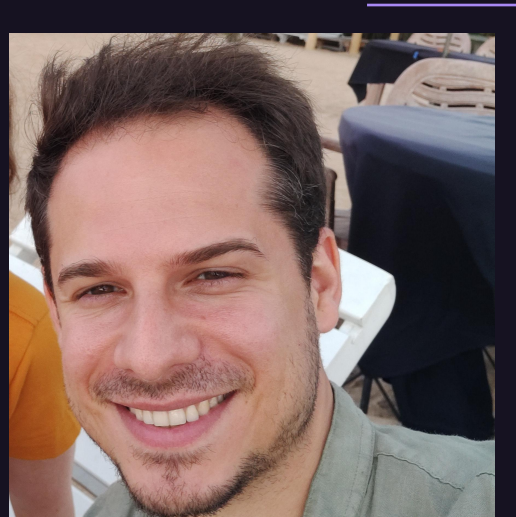
Timo Furrer

---



# Timo Furrer

Senior Backend Engineer  
@timofurrer





*Once upon a time ...*

# What do we want?

- Low-Friction setup
- Extend to a “complex” setup
- Confidence
- Reusability



# Storing State

```
resource "random_pet" "main"  
{  
  length = 2  
}  
  
output "pet_name" {  
  value = random_pet.main.id  
}
```



# Storing State



```
$ terraform init
```

```
$ terraform apply
```

```
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

```
$ cat terraform.tfstate
```

# Storing State



```
{
  "version": 4,
  "terraform_version": "1.3.6",
  "serial": 2,
  "lineage": "972c2eb1-46ab-e29b-2cda-14baacd0e2fa",
  "outputs": {
    "pet_name": {
      "value": "welcome-antelope",
      "type": "string"
    }
  },
  "resources": [
    {
      "mode": "managed",
      "type": "random_pet",
      "name": "main",
      "provider":
"provider[\\\"registry.terraform.io/hashicorp/random\\\"]",
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "id": "welcome-antelope",
            "keepers": null,
            "length": 2,
            "prefix": null,
            "separator": "-"
          },
          "sensitive_attributes": []
        }
      ]
    }
  ],
  "check_results": null
}
```

# Storing State



```
{
  "version": 4,
  "terraform_version": "1.3.6",
  "serial": 2,
  "lineage": "972c2eb1-46ab-e29b-2cda-14baacd0e2fa",
  "outputs": {
    "pet_name": {
      "value": "welcome-antelope",
      "type": "string"
    }
  },
  "resources": [
    {
      "mode": "managed",
      "type": "random_pet",
      "name": "main",
      "provider":
"provider[\"registry.terraform.io/hashicorp/random\"]",
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "id": "welcome-antelope",
            "keepers": null,
            "length": 2,
            "prefix": null,
            "separator": "-"
          },
          "sensitive_attributes": []
        }
      ]
    }
  ],
  "check_results": null
}
```



# Storing State



```
{
  "version": 4,
  "terraform_version": "1.3.6",
  "serial": 2,
  "lineage": "972c2eb1-46ab-e29b-2cda-14baacd0e2fa",
  "outputs": {
    "pet_name": {
      "value": "welcome-antelope",
      "type": "string"
    }
  },
  "resources": [
    {
      "mode": "managed",
      "type": "random_pet",
      "name": "main",
      "provider":
"provider[\\\"registry.terraform.io/hashicorp/random\\\"]",
      "instances": [
        {
          "schema_version": 0,
          "attributes": {
            "id": "welcome-antelope",
            "keepers": null,
            "length": 2,
            "prefix": null,
            "separator": "-"
          },
          "sensitive_attributes": []
        }
      ]
    }
  ],
  "check_results": null
}
```

# Storing State



Let's store the state remotely:

```
terraform {  
  backend "http" {}  
}  
  
resource "random_pet" "main"  
{  
  length = 2  
}  
  
output "pet_name" {  
  value = random_pet.main.id  
}
```

# Storing State



We need to configure the backend:

```
terraform {  
  backend "http" {  
    // address?  
    // (un)lock address?  
    // username & password?  
    // timeouts?  
  }  
}  
  
resource "random_pet" "main"  
{  
  ..  
}
```

# GitLab-managed state

We can configure it manually in the config:

```
terraform {  
  backend "http" {  
    address      = "https://gitlab.com/.../state-name"  
    lock_address =  
"https://gitlab.com/.../state-name/lock"  
    unlock_address =  
"https://gitlab.com/.../state-name/lock"  
    username     = "__token__"  
    password     = "glpat-..."  
    lock_method  = "POST"  
    unlock_method = "DELETE"  
    ...  
  }  
}  
  
resource "random_pet" "main" {  
  ...  
}
```



# GitLab-managed state

... or in the terminal during initialization:

```
PROJECT_ID="<gitlab-project-id>"
TF_USERNAME="<gitlab-username>"
TF_PASSWORD="<gitlab-personal-access-token>"
TF_ADDRESS="https://gitlab.com/api/v4/projects/{PROJECT_ID}/terraform/state/state-name"
```

```
terraform init \  
  -backend-config=address=${TF_ADDRESS} \  
  -backend-config=lock_address=${TF_ADDRESS}/lock \  
  -backend-config=unlock_address=${TF_ADDRESS}/lock \  
  -backend-config=username=${TF_USERNAME} \  
  -backend-config=password=${TF_PASSWORD} \  
  -backend-config=lock_method=POST \  
  -backend-config=unlock_method=DELETE \  
  -backend-config=retry_wait_min=5
```



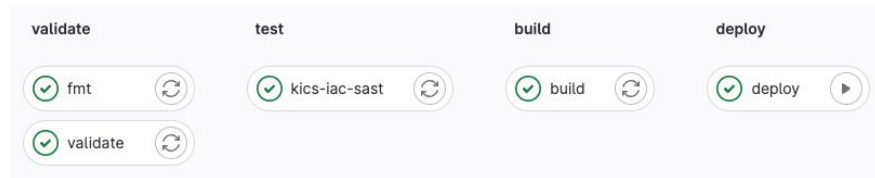
# *Terraform CI/CD Templates*

# Pipeline



Include the template in the `.gitlab-ci.yml`:

```
include:  
  template: Terraform.latest.gitlab-ci.yml
```



## Terraform.latest.gitlab-ci.yml

### Infrastructure

- `random_pet`

### TF States

- `default`

### Environments

- `default`



*Show me! ...*



# Terraform State

The screenshot shows the GitLab web interface. The top navigation bar includes the GitLab logo, a 'Next' button, a search bar with the text 'Search GitLab', and several notification icons (9, 6, 6) and a user profile icon. The left sidebar contains a navigation menu with the following items: 'empower-terraform-gitlab', 'Project information', 'Repository', 'Issues' (0), 'Merge requests' (1), 'CI/CD', 'Security & Compliance', 'Deployments', 'Packages and registries', 'Infrastructure', 'Kubernetes clusters', 'Terraform' (selected), 'Google Cloud', 'Monitor', and 'Analytics'. The main content area shows the breadcrumb 'Timo Furrer > empower-terraform-gitlab > Terraform'. Below this, there is a section titled 'States' with a count of 1. A table displays the state information:

| Name    | Pipeline  | Details                       | Actions |
|---------|---|-------------------------------|---------|
| default | <a href="#">#758197639</a><br><span>passed</span> | Timo Furrer updated 1 day ago | ⋮       |



# Environment

The screenshot displays the GitLab web interface. At the top, there is a navigation bar with the GitLab logo, a 'Next' button, a search bar containing 'Search GitLab', and several utility icons including a plus sign, a notification bell with '9', a search icon with '6', a checkmark with '6', a help icon, and a user profile icon.

The main content area is titled 'empower-terraform-gitlab' and 'Environments'. It shows a summary of environment status: 'Available 1' and 'Stopped 0'. There are two buttons: 'Enable review app' and 'New environment'. Below this is a search bar for environment names.

The 'default' environment is expanded, showing a deployment status of 'Success' (Latest Deployed) with commit #4 (cca2b26a) deployed 1 day ago. A message 'Allow Developers to plan' is visible. Below the message is a table with columns for Triggerer, Job, and Branch.

| Triggerer   | Job    | Branch |
|-------------|--------|--------|
| @timofurrer | deploy | main   |

The left sidebar contains a navigation menu with the following items: Project information, Repository, Issues (0), Merge requests (1), CI/CD, Security & Compliance, Deployments, Environments (highlighted), Feature Flags, Releases, Pages, and Packages and registries.



# Environment Deployments

The screenshot displays the GitLab CI/CD interface for the project 'empower-terraform-gitlab'. The left sidebar contains navigation options: Project information, Repository, Issues (0), Merge requests (1), CI/CD, Security & Compliance, Deployments, Environments (selected), Feature Flags, Releases, Pages, and Packages and registries. The main content area shows the 'default' environment with a table of recent deployments. The table has columns for Status, ID, Triggerer, Commit, Job, Created, and Deployed. Three successful deployments are listed, each with a 'success' status, an ID (#4, #3, #2), a triggerer profile, a commit hash and message, a job name 'deploy (#366...', and timestamps '1 day ago'. Each row includes a refresh icon. At the top right of the environment view are 'Edit' and 'Stop' buttons.

Timo Furrer > empower-terraform-gitlab > Environments > default

## default

Edit Stop

| Status  | ID | Triggerer | Commit                                      | Job             | Created   | Deployed  |  |
|---------|----|-----------|---|-----------------|-----------|-----------|--|
| success | #4 |           | main → cca2b26a<br>Allow Developers to plan | deploy (#366... | 1 day ago | 1 day ago |  |
| success | #3 |           | main → 5b448594<br>Fix formatting           | deploy (#366... | 1 day ago | 1 day ago |  |
| success | #2 |           | main → 59f0e354<br>Set TF_ROOT              | deploy (#366... | 1 day ago | 1 day ago |  |



# What do we want?

- Low-Friction setup
- Extend to a “complex” setup
- Confidence
- Reusability



# Customize Pipeline

Give the state and environment a name:

```
include:  
  template: Terraform.latest.gitlab-ci.yml  
  
variables:  
  TF_STATE_NAME: production
```



# Customize Pipeline

Place the Terraform config in a directory:

```
include:  
  template: Terraform.latest.gitlab-ci.yml  
  
variables:  
  TF_STATE_NAME: production  
  TF_ROOT: terraform/
```



# Extend Template Jobs

## Setup Vault Authentication:

```
include:
  template: Terraform.latest.gitlab-ci.yml

.vault-auth: &vault_auth
- apk add --no-cache vault
- export VAULT_ADDR=http://vault.example.com:8200
- export VAULT_TOKEN="$(
  vault write -field=token auth/jwt/login
  role=$CI_PROJECT_NAME jwt=$CI_JOB_JWT
)"

build:
  before_script:
    - *vault_auth

deploy:
  before_script:
    - *vault_auth
```





# Extend Template Jobs

## Setup Vault Authentication:

```
include:
  template: Terraform.latest.gitlab-ci.yml

.vault-auth: &vault_auth
- apk add --no-cache vault
- export VAULT_ADDR=http://vault.example.com:8200
- export VAULT_TOKEN="$(
  vault write -field=token auth/jwt/login
  role=$CI_PROJECT_NAME jwt=$CI_JOB_JWT
)"

build: # override job from the template
  before_script:
  - *vault_auth

deploy: # override job from the template
  before_script:
  - *vault_auth
```



# Extend Template Jobs

## Setup Vault Authentication:

```
include:
  template: Terraform.latest.gitlab-ci.yml

.vault-auth: &vault_auth # Keep it DRY
- apk add --no-cache vault
- export VAULT_ADDR=http://vault.example.com:8200
- export VAULT_TOKEN="$(
  vault write -field=token auth/jwt/login
  role=$CI_PROJECT_NAME jwt=$CI_JOB_JWT
)"

build: # override job from the template
  before_script:
  - *vault_auth

deploy: # override job from the template
  before_script:
  - *vault_auth
```



# Multi Environment

## Extend to multiple environments:

```
.environment:
  stage: environments
  variables:
    TF_STATE_NAME: $environment
    TF_CLI_ARGS_plan:
"-var-file=environments/$TF_STATE_NAME.tfvars"
  trigger:
    include:
      template: Terraform.latest.gitlab-ci.yml
    strategy: depend

staging:
  extends: .environment
  variables:
    environment: staging

production:
  extends: .environment
  variables:
    environment: production
  rules:
    - if: $CI_COMMIT_BRANCH == $CI_DEFAULT_BRANCH
```



# Multi Environment

## Extend to multiple environments:

```
.environment:
  stage: environments
  variables:
    TF_STATE_NAME: $environment
    TF_CLI_ARGS_plan: "-var-file=environments/$TF_STATE_NAME.tfvars"
  trigger:
    include:
      template: Terraform.latest.gitlab-ci.yml
    strategy: depend

staging: # Job to trigger `staging` environment
  extends: .environment
  variables:
    environment: staging

production: # Job to trigger `production` environment
  extends: .environment
  variables:
    environment: production
  rules:
    - if: $CI_COMMIT_BRANCH == $CI_DEFAULT_BRANCH
```



# Multi Environment

## Extend to multiple environments:

```
.environment:
  stage: environments
  variables:
    TF_STATE_NAME: $environment
    TF_CLI_ARGS_plan: "-var-file=environments/$TF_STATE_NAME.tfvars"
  trigger:
    include:
      template: Terraform.latest.gitlab-ci.yml
    strategy: depend

staging: # Job to trigger `staging` environment
  extends: .environment
  variables:
    environment: staging

production: # Job to trigger `production` environment
  extends: .environment
  variables:
    environment: production
  rules:
    # only for default branch
    - if: $CI_COMMIT_BRANCH == $CI_DEFAULT_BRANCH
```



# Multi Environment

## Extend to multiple environments:

```
.environment:
  stage: environments
  variables:
    TF_STATE_NAME: $environment
    TF_CLI_ARGS_plan: "-var-file=environments/$TF_STATE_NAME.tfvars"
  trigger:
    include:
      template: Terraform.latest.gitlab-ci.yml
    strategy: depend

staging: # Job to trigger `staging` environment
  extends: .environment
  variables:
    environment: staging

production: # Job to trigger `production` environment
  extends: .environment
  variables:
    environment: production
  rules:
    # only for default branch
    - if: $CI_COMMIT_BRANCH == $CI_DEFAULT_BRANCH
```



# Multi Environment

## Extend to multiple environments:

```
.environment: # base job template
  stage: environments
  variables:
    TF_STATE_NAME: $environment
    TF_CLI_ARGS_plan: "-var-file=environments/$TF_STATE_NAME.tfvars"
  trigger:
    include:
      template: Terraform.latest.gitlab-ci.yml
    strategy: depend

staging: # Job to trigger `staging` environment
  extends: .environment
  variables:
    environment: staging

production: # Job to trigger `production` environment
  extends: .environment
  variables:
    environment: production
  rules:
    # only for default branch
    - if: $CI_COMMIT_BRANCH == $CI_DEFAULT_BRANCH
```



# Multi Environment

## Extend to multiple environments:

```
.environment: # base job template
  stage: environments
  variables:
    # Environment specific variables:
    TF_STATE_NAME: $environment
    TF_CLI_ARGS_plan: "-var-file=environments/$TF_STATE_NAME.tfvars"
  trigger:
    include:
      template: Terraform.latest.gitlab-ci.yml
    strategy: depend

staging: # Job to trigger `staging` environment
  extends: .environment
  variables:
    environment: staging

production: # Job to trigger `production` environment
  extends: .environment
  variables:
    environment: production
  rules:
    # only for default branch
    - if: $CI_COMMIT_BRANCH == $CI_DEFAULT_BRANCH
```





# Multi Environment

## Extend to multiple environments:

```
.environment: # base job template
  stage: environments
  variables:
    # Environment specific variables:
    TF_STATE_NAME: $environment
    TF_CLI_ARGS_plan: "-var-file=environments/$TF_STATE_NAME.tfvars"
  trigger: # trigger child-pipeline
    include:
      template: Terraform.latest.gitlab-ci.yml # include the template
    strategy: depend

staging: # Job to trigger `staging` environment
  extends: .environment
  variables:
    environment: staging

production: # Job to trigger `production` environment
  extends: .environment
  variables:
    environment: production
  rules:
    # only for default branch
    - if: $CI_COMMIT_BRANCH == $CI_DEFAULT_BRANCH
```



## Child Pipeline: Terraform.latest.gitlab-ci.yml

### Infrastructure

- `random_pet [len=3]`
- `random_pet [len=2]`

### TF States

- `staging`
- `production`

### Environments

- `staging`
- `production`

# What do we want?

- Low-Friction setup
- Extend to a “complex” setup
- Confidence
- Reusability



# Confidence



- Use Multiple Environments
- Use Review Apps
- Use Merge Request Terraform Widget
- Use static code analysis report
- Run tests
  - simple curl commands `~\_(\ツ)\_/~`
  - Terratest
  - ...

# Collaboration

09 Collaboration with Con...

Project information

Repository

Issues 0

Merge requests 1

CI/CD

Security & Compliance

Deployments

Packages and registries

Infrastructure

Monitor

Analytics

Wiki

Snippets

Settings

GitLab.org > Empower everyone to manage Infrastructure with GitLab and Terraform > 09 Collaboration with Confidence > Merge requests > 110

## Close gap between review and production env

Edit Code

Open Timo Furrer requested to merge review/dec-name into main 29 minutes ago

Overview 0 Commits 1 Pipelines 1 Changes 1

This change set decreases the name of the `pet infrastructure` to a length of `4`.  
It used to be `5`, which was evaluated to be too far away from the `production` value of `2`.

0 0 0

Pipeline #76661149 passed for bdd63929 on review/dec-name 24 minutes ago

Deployed to review-dec-name 25 minutes ago View app

Approval is optional

View eligible approvers

2 Terraform reports were generated in your pipelines

- The job `build:production` generated a report. Reported Resource Changes: 0 to add, 0 to change, 0 to delete Full log
- The job `build:review` generated a report. Reported Resource Changes: 6 to add, 0 to change, 0 to delete Full log

Security scanning detected no new vulnerabilities. Full report

SAST detected no new vulnerabilities.

Ready to merge!

Squash commits  Edit commit message

1 commit and 1 merge commit will be added to main.

Merge

Mark as done

Assignee: Timo Furrer Edit

0 Reviewers: None - assign yourself Edit

Labels: Infrastructure Edit

Milestone: None Edit

Time tracking: No estimate or time spent +

Lock merge request: Unlocked Edit

Notifications: On

1 participant

Reference: gitlab-org/configure/...  
Source branch: review/dec-na...



Demo

# Permissions



Who has access to the state?

|        | Maintainers                         | Developers                          |
|--------|-------------------------------------|-------------------------------------|
| read   | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| write  | <input checked="" type="checkbox"/> | <input type="checkbox"/>            |
| lock   | <input checked="" type="checkbox"/> | <input type="checkbox"/>            |
| unlock | <input checked="" type="checkbox"/> | <input type="checkbox"/>            |

# Permissions



Use state-only project to better control state access:

```
include:
  template: Terraform.latest.gitlab-ci.yml

variables:
  TF_STATE_NAME: production
  TF_PROJECT_ID: 123456
  # define TF_STATE_ADDRESS
  TF_STATE_ADDRESS:
  $CI_API_V4_URL/projects/$TF_PROJECT_ID/ter
  raform/state/${TF_STATE_NAME}}
```

[Open Issue: Protected Terraform States](#)

# What do we want?

- Low-Friction setup
- Extend to a “complex” setup
- Confidence
- Reusability





# *Sharing Modules*

# Modules



## What is a Module again?

“A Terraform module is a set of Terraform configuration files in a single directory.”

- [Terraform Docs](#)

```
$ ls pet-site/  
.  
├── LICENSE  
├── README.md  
├── main.tf  
├── variables.tf  
└── outputs.tf
```

# Modules



We can use a Terraform Module:

```
$ cat production/main.tf
```

```
module "pet_site" {  
  source = "../pet-site"  
  
  pet_name_length = 2  
}
```

```
$ cat staging/main.tf
```

```
module "pet_site" {  
  source = "../pet-site"  
  
  pet_name_length = 4  
}
```

# Modules



We can use a remote Terraform Module:

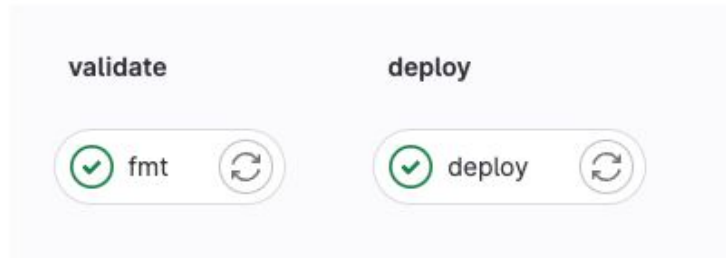
```
module "pet_site" {  
  source = "gitlab.com/gitlab-org/pet-site/aws"  
  version = "~> 1.0"  
  
  pet_name_length = 2  
}
```

# Publish Module



Include module template in `.gitlab-ci.yml`:

```
include:  
  template: Terraform-Module.gitlab-ci.yml  
  
variables:  
  TERRAFORM_MODULE_DIR: ${CI_PROJECT_DIR}  
  TERRAFORM_MODULE_NAME: ${CI_PROJECT_NAME}  
  TERRAFORM_MODULE_SYSTEM: local  
  TERRAFORM_MODULE_VERSION: ${CI_COMMIT_TAG}
```



# Infrastructure Registry

The screenshot displays the GitLab Infrastructure Registry interface. The left sidebar contains navigation options: Project information, Repository, Issues (0), Merge requests (0), CI/CD, Security & Compliance, Deployments, Packages and registries (Package Registry, Container Registry, Infrastructure Registry), Infrastructure, Monitor, Analytics, Wiki, Snippets, and Settings. The main content area shows the details for the 'pet-site/aws' Terraform module, version 1.0.0, published 1 day ago. It includes a 'Delete' button, a 'Detail' tab, and a 'History' section with a timeline of events: creation, commit creation, pipeline build, and publication. Below this is the 'Provision instructions' section with a code block for Terraform configuration. The 'Registry setup' section provides a code block for authorization credentials. At the bottom, an 'Assets' table lists the available assets.

04 - Deploy Terraform Mo...

GitLab... > Empower everyone to manage Infrastructure with GitLab and Te... > 04 - Deploy Terraform Mo... > Infrastructure Regi... > pet-site/a... > 1.0.0

## pet-site/aws

v 1.0.0 published 1 day ago

Terraform 1.24 KiB 04 - Deploy Terraform... 1.0.0

Detail Other versions

### History

- pet-site/aws version 1.0.0 was first created 1 day ago
- Created by commit d6de7554 on branch 1.0.0
- Built by pipeline #762986703 triggered 1 day ago by Timo Furrer
- Published to the 04 - Deploy Terraform Module Package Registry 1 day ago

### Provision instructions

Copy and paste into your Terraform configuration, insert the variables, and run Terraform init:

```
module "my_module_name" {
  source = "gitlab.com/gitlab-org/pet-site/aws"
  version = "1.0.0"
}
```

### Registry setup

To authorize access to the Terraform registry:

```
credentials "gitlab.com" {
  token = "<TOKEN>"
}
```

For more information on the Terraform registry, [see our documentation](#).

### Assets

| Name                   | Commit | Size     | Created   |
|------------------------|--------|----------|-----------|
| pet-site-aws-1.0.0.tgz |        | 1.24 KiB | 1 day ago |



# What do we want?

- Low-Friction setup
- Extend to a “complex” setup
- Confidence
- Reusability





*One more thing ...*



# GitLab Provider



```
terraform {  
  required_providers {  
    gitlab = {  
      source = "gitlabhq/gitlab"  
      version = "~> 15.8"  
    }  
  }  
}  
  
provider "gitlab" {  
  # Configuration options  
}
```

# GitLab Provider



```
terraform {  
  required_providers {  
    gitlab = {  
      source = "gitlabhq/gitlab"  
      version = "~> 15.8"  
    }  
  }  
}  
  
provider "gitlab" {  
  # Configuration options  
  base_url = "gitlab.example.com"  
  token    = var.gitlab_token  
}
```

67+ Resources  
40+ Data Sources



```
terraform {  
  required_providers {  
    gitlab = {  
      source  = "gitlabhq/gitlab"  
      version = "~> 15.8"  
    }  
  }  
}  
  
provider "gitlab" {  
  # Configuration options  
}
```

# Popular Use Cases



- User onboarding

```
data "gitlab_group" "backend_engineers" {  
  full_path = "roles/backend-engineers"  
}
```

```
resource "gitlab_user" "jessica_drew" {  
  name      = "Jessica Drew"  
  username  = "jessicadrew"  
  email     = "jessica.drew@marvel.example.com"  
}
```

```
resource "gitlab_group_membership" "jessica_drew_backend" {  
  user_id      = gitlab_user.jessica_drew.id  
  group_id     = data.gitlab_group.backend_engineers.id  
  access_level = "maintainer"  
}
```

# Popular Use Cases



- User onboarding
- **(compliant) group / project creation**
  - Protected Branch, Tag and Environment settings
  - Standardized permission settings
  - Approval rules
  - CODEOWNER settings
  - Integrations (slack, ...)
  - CI/CD variables
  - Issue Boards
  - ...

# Popular Use Cases



- User onboarding
- (compliant) group / project creation
- **Instance management**
  - Application Settings
  - Topic management
  - System Hooks
  - ...

# Popular Use Cases

- User onboarding
- (compliant) group / project creation
- Instance management
- **Runner management**

```
resource "gitlab_group" "my_group" {
  name          = "my-group"
  description   = "Group that holds the runners"
}

resource "gitlab_runner" "main" {
  registration_token = gitlab_group.my_group.runners_token
}
```



# Popular Use Cases



- User onboarding
- (compliant) group / project creation
- Instance management
- Runner management
- **GitLab Agent for Kubernetes management**

```
resource "gitlab_cluster_agent" "this"{
  project = "prd-cluster"
  name    = "my-agent"
}

resource "gitlab_cluster_agent_token" "this"{
  project      = gitlab_cluster_agent.this.project
  agent_id    = gitlab_cluster_agent.this.agent_id
  name        = "prd"
}

resource "helm_release" "gitlab_agent"{
  name           = "gitlab-agent"
  namespace      = "gitlab-agent"
  create_namespace = true
  repository     = "https://charts.gitlab.io"
  chart          = "gitlab-agent"
  version        = "1.10.0"

  set {
    name = "config.token"
    value = gitlab_cluster_agent_token.this.token
  }
}
```





**Thank you!**



# Timo Furrer

Senior Backend Engineer  
@timofurrer

