

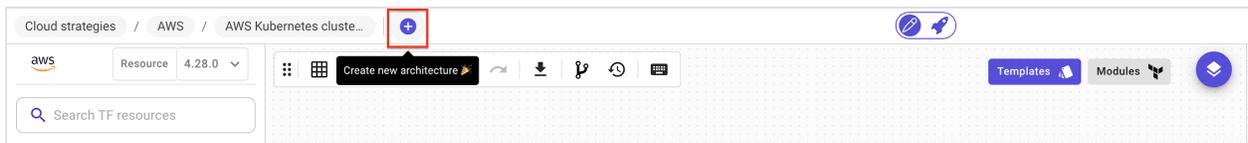
Getting Started

Fast track

1. Create an account

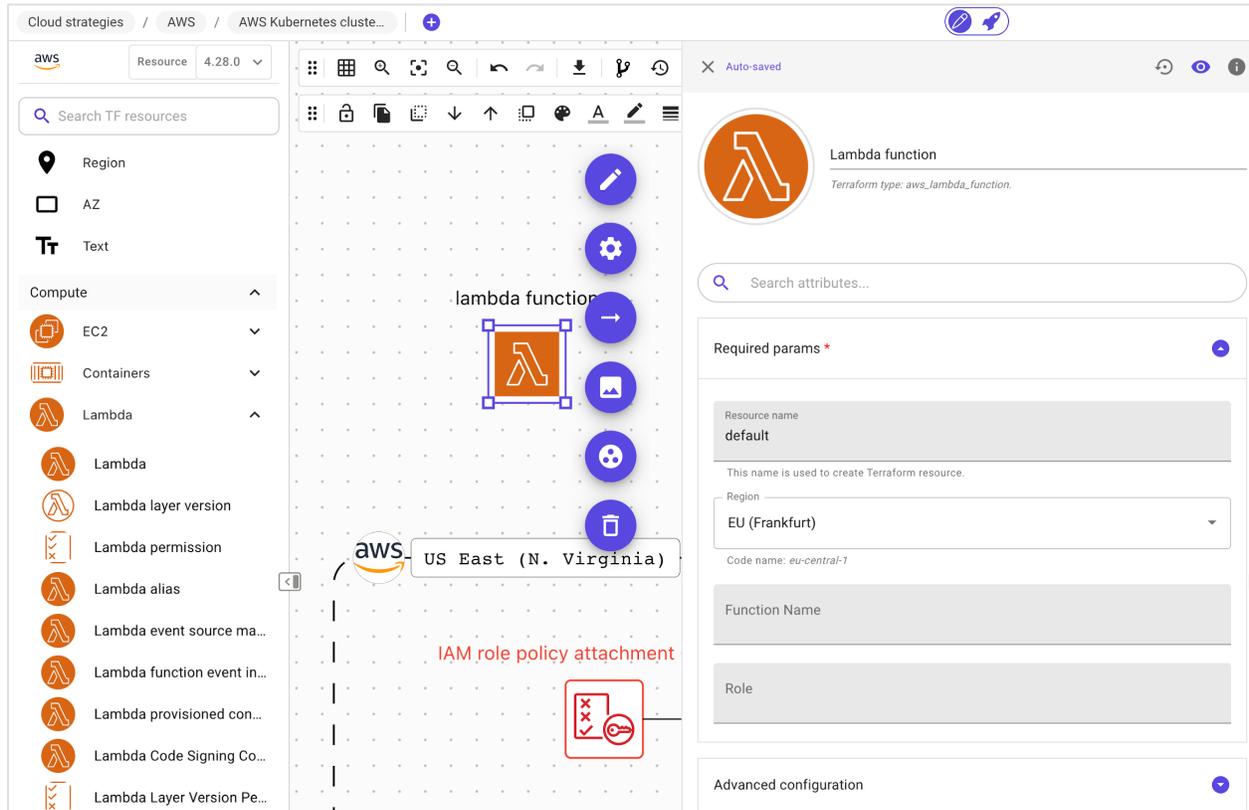
Register [here](#) to create your account. You can sign up with your Google or Microsoft login.

2. Create a new architecture



3. Add cloud resources

Drag and drop cloud resources from the leftbar to the design area to build your architecture. Customize the cloud configuration of the resources



4. Auto-generate the Terraform code

See the auto-generated Terraform code on the right pane.

The screenshot displays the Brainboard interface for configuring a Terraform resource. On the left, a configuration panel for a 'Lambda function' is shown. It includes a search bar for attributes, a 'Required params' section with fields for 'Resource name' (set to 'default'), 'Region' (set to 'US East (N. Virginia)'), 'Function Name' (set to 'docs'), and 'Role' (set to 'aws_iam_role.iam_for_lambda.arn'). Below this is an 'Advanced configuration' section. On the right, a code editor shows the Terraform code for the 'main.tf' file. The code defines an 'aws_eks_cluster' resource and an 'aws_lambda_function' resource. The 'aws_lambda_function' resource is highlighted with a red box, showing its configuration: 'provider = aws.us-east-1', 'tags = merge(var.tags, {})', 'role = aws_iam_role.iam_for_lambda.arn', and 'function_name = "docs"'. At the top right of the code editor, there are buttons for 'CREATE PULL REQUEST' and 'PLAN'.

We support providers such as AWS, Azure, GCP and more.

5. Add you cloud credentials

Add your preferred cloud provider credentials into Brainboard [here](#).

Here are examples for AWS and Azure.

ORGANIZATION

- General
- Members
- Teams
- Projects
- Billing
- Support

SECURITY

- Cloud Providers
- Git Apps
- Data

MY ACCOUNT

- Information
- Personal Git tokens

CONFIGURATION

Cloud providers configuration

aws Amazon Web Services

Credentials
The only accepted AWS credentials is the pair of access key id and its associated secret.

DDCS + Add new credentials

Name	Creation time	Last update	Access key id	Secret access key	Actions
Main	2020-01-09 14:05	2021-10-21 07:45	[REDACTED]	*****	
[REDACTED]	2022-04-02 14:00	2022-12-07 08:50	[REDACTED]	*****	
[REDACTED]	2022-12-06 10:38	2022-12-06 10:40	[REDACTED]	*****	

Microsoft Azure RM

Google Cloud Platform

Scaleway

Oracle Cloud Infrastructure

ORGANIZATION

- General
- Members
- Teams
- Projects
- Billing
- Support

SECURITY

- Cloud Providers
- Git Apps
- Data

MY ACCOUNT

- Information
- Personal Git tokens

CONFIGURATION

Cloud providers configuration

aws Amazon Web Services

Microsoft Azure RM

Credentials
The only accepted Azure credential method is based on the client id and its associated secret.

DDCS + Add new credentials

Name	Creation time	Last update	Subscription id	Client id	Tenant id	Client secret	Actions
Training	2022-02-22 19:48	2022-02-28 14:20	[REDACTED]	[REDACTED]	[REDACTED]	*****	
Demos	2022-02-23 15:43	2022-08-26 20:59	[REDACTED]	[REDACTED]	[REDACTED]	*****	
Brainboard Production	2022-04-02 08:20	2022-04-02 08:20	[REDACTED]	[REDACTED]	[REDACTED]	*****	

Google Cloud Platform

Scaleway

Oracle Cloud Infrastructure

6. Trigger a plan

After adding your cloud credentials, you can trigger the Terraform plan directly from the design area and get the output in real time.

Cloud strategies / Azure / Azure Landing Zone Ba...

Resource 3.20.0

main.tf

CREATE PULL REQUEST PLAN

```

1 resource "azurerm_resource_group" "resource_group_hub" {
2   tags = merge(var.tags, {})
3   name = "rg_hub"
4   location = var.location
5 }
6
7 resource "azurerm_resource_group" "resource_group_spoke" {
8   tags = merge(var.tags, {})
9   name = "rg_spoke"
10  location = var.location
11 }
12
13 resource "azurerm_virtual_network" "virtual_network_hub" {
14   tags = merge(var.tags, {})
15   resource_group_name = azurerm_resource_group.resource_group_hub.name
16   name = "vnet_hub"
17   location = var.location
18   address_space = [
19     var.vnet_hub_addr_space,
20   ]
21 }
22
23 resource "azurerm_virtual_network" "virtual_network_spoke" {
24   tags = merge(var.tags, {})
25   resource_group_name = azurerm_resource_group.resource_group_spoke.name
26   name = "vnet_spoke"
27   location = var.location
28   address_space = [
29     var.vnet_spoke_addr_space,
30   ]
31 }
32
33 resource "azurerm_virtual_network_peering" "virtual_network_peering" {
34   virtual_network_name = azurerm_virtual_network.virtual_network_hub.name
35   resource_group_name = azurerm_resource_group.resource_group_hub.name
36   remote_virtual_network_id = azurerm_virtual_network.virtual_network_spoke.id
37   name = "peerhubtospoke"
38   allow_virtual_network_access = true
39   allow_forwarded_traffic = true
40 }
41
42 resource "azurerm_subnet" "subnet_firewall" {
43   virtual_network_name = azurerm_virtual_network.virtual_network_hub.name
44   resource_group_name = azurerm_resource_group.resource_group_hub.name
45   name = "AzureFirewallSubnet"
46   address_prefixes = [
47     var.vnet_firewall_addr_space,
48   ]
49 }
50
51 resource "azurerm_subnet" "subnet_jumphost" {
52   virtual_network_name = azurerm_virtual_network.virtual_network_hub.name
53   resource_group_name = azurerm_resource_group.resource_group_hub.name
54   name = "JumpHostSubnet"
55   address_prefixes = [
56     var.vnet_jumphost_addr_space,
57   ]
58 }
59
60

```

Cloud strategies / Azure / Azure Landing Zone Ba...

CI/CD ENGINE

VALIDATE PLAN

APPLY DESTROY

```

+ custom_dns_configs = (known after apply)
+ id = (known after apply)
+ location = "eastus"
+ name = "pe-keyvault"
+ network_interface = (known after apply)
+ private_dns_zone_configs = (known after apply)
+ resource_group_name = "rg_spoke"
+ subnet_id = (known after apply)
+ tags = {
+   "archuid" = "004cd02e-5895-4e66-8ce4-1665b3112232"
+   "env" = "Azure"
+ }
+ private_service_connection {
+   is_manual_connection = false
+   name = "connectiontokv"
+   private_connection_resource_id = (known after apply)
+   private_ip_address = (known after apply)
+   subresource_names = ["Vault"]
+ }
}

# azurerm_public_ip.public_ip_app will be created
resource "azurerm_public_ip" "public_ip_app" {
+ allocation_method = "Static"
+ fqdn = (known after apply)
+ id = (known after apply)
+ idle_timeout_in_minutes = 4
+ ip_address = (known after apply)
+ ip_version = "IPv4"
+ location = "eastus"
+ name = "pip_firewall"
+ resource_group_name = "rg_hub"
+ sku = "Standard"
+ sku_tier = "Regional"
+ tags = {
+   "archuid" = "004cd02e-5895-4e66-8ce4-1665b3112232"
+   "env" = "Azure"
+ }
}

# azurerm_public_ip.public_ip_vp will be created
resource "azurerm_public_ip" "public_ip_vp" {
+ allocation_method = "Dynamic"
+ fqdn = (known after apply)
+ id = (known after apply)
+ idle_timeout_in_minutes = 4
+ ip_address = (known after apply)
+ ip_version = "IPv4"
+ location = "eastus"
+ name = "pip_vp"
+ resource_group_name = "rg_hub"
+ sku = "Basic"
+ sku_tier = "Regional"
+ tags = {
+   "archuid" = "004cd02e-5895-4e66-8ce4-1665b3112232"
+   "env" = "Azure"
+ }
}

# azurerm_resource_group.resource_group_hub will be created
resource "azurerm_resource_group" "resource_group_hub" {
+ id = (known after apply)
+ location = "eastus"
}

```

Succeeded - 2022-12-15 20:01 (UTC)

Account Setup

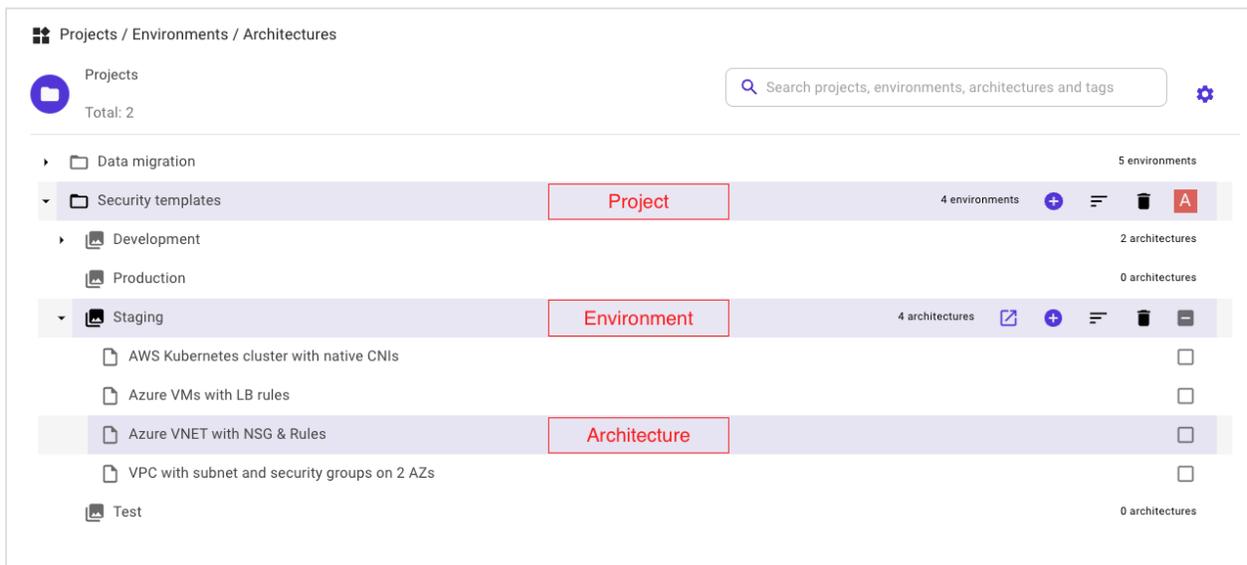
Here below are setup actions that you need to do for your account to be able to:

- Do pull requests to your repo
- Organize how you work with your team collaboratively
- Do plan, apply and/or destroy

How data is organized

Before you start, you need to understand how information are organized in Brainboard.

There are 3 levels of information:



The screenshot displays the Brainboard interface for managing cloud infrastructure. At the top, there's a breadcrumb trail: "Projects / Environments / Architectures". Below this, a "Projects" section shows a search bar and a "Total: 2" indicator. The main content area is a tree view of folders and items. The "Security templates" folder is highlighted as a "Project" and contains 4 environments. The "Staging" folder is highlighted as an "Environment" and contains 4 architectures. The "Architecture" level is highlighted for "Azure VNET with NSG & Rules". Other folders include "Data migration" (5 environments), "Development" (2 architectures), "Production" (0 architectures), and "Test" (0 architectures). Each folder has icons for adding, deleting, and refreshing.

1. **Projects:** this is the topmost level of the hierarchy and it is equivalent to a folder. To better organize your work within Brainboard, you can consider the the project as your client or team's folder, so you can rename it in way that makes sens to you.
2. **Environments:** inside any given project, you have multiple environments. By default Brainboard suggests to create 5 environments when you create a project. These environments are the stages of production systems, like test, development, QA, staging, production... and they are containers for architectures.
3. **Architectures:** is the last element in the hierarchy and contains the diagram of your cloud infrastructure with the auto-generated code. The architecture is the one that accepts actions like: plan, apply and/or destroy, pull requests, versioning...

TERRAFORM STATE

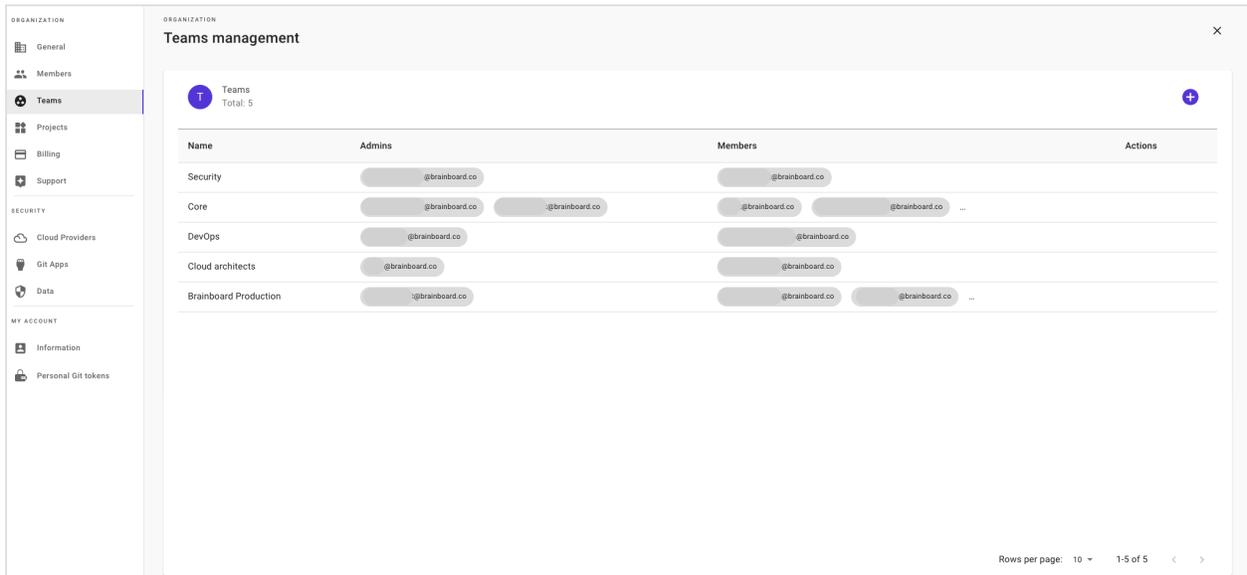
The architecture is associated to one Terraform state to better isolate and secure your infrastructure. This is a Terraform best practice and you can specify a different remote backend in the setting page.

Please refer to the right section of every level if you need detailed information.

1. Team organization

When you invite your colleagues to build cloud infrastructures within Brainboard, it's better to put them into teams to reflect your internal organization & processes. For e.g. DevOps team, Cloud Architect team, Security team, Project managers...

Here is the [link](#) to access the team settings.



Name	Admins	Members	Actions
Security	@brainboard.co	@brainboard.co	
Core	@brainboard.co @brainboard.co	@brainboard.co @brainboard.co	...
DevOps	@brainboard.co	@brainboard.co	
Cloud architects	@brainboard.co	@brainboard.co	
Brainboard Production	@brainboard.co	@brainboard.co @brainboard.co	...

2. Add cloud credentials

Add your preferred cloud provider credentials to be able to do plan, apply or destroy and also to trigger the CI/CD pipelines.

Here is the [link](#) to access the cloud credentials page.

Examples of configuration for AWS and Azure:

Cloud providers configuration

aws Amazon Web Services

Credentials
The only accepted AWS credentials is the pair of access key id and its associated secret.

Name	Creation time	Last update	Access key id	Secret access key	Actions
Main	2020-01-09 14:05	2021-10-21 07:45	[REDACTED]	*****	[REDACTED]
[REDACTED]	2022-04-02 14:00	2022-12-07 08:50	[REDACTED]	*****	[REDACTED]
[REDACTED]	2022-12-06 10:38	2022-12-06 10:40	[REDACTED]	*****	[REDACTED]

Buttons: DOCS, Add new credentials

Cloud providers configuration

Microsoft Azure RM

Credentials
The only accepted Azure credential method is based on the client id and its associated secret.

Name	Creation time	Last update	Subscription id	Client id	Tenant id	Client secret	Actions
Training	2022-02-22 19:48	2022-02-28 14:20	[REDACTED]	[REDACTED]	[REDACTED]	*****	[REDACTED]
Demos	2022-02-23 15:43	2022-08-26 20:59	[REDACTED]	[REDACTED]	[REDACTED]	*****	[REDACTED]
Brainboard Production	2022-04-02 08:20	2022-04-02 08:20	[REDACTED]	[REDACTED]	[REDACTED]	*****	[REDACTED]

Buttons: DOCS, Add new credentials

3. Git configuration

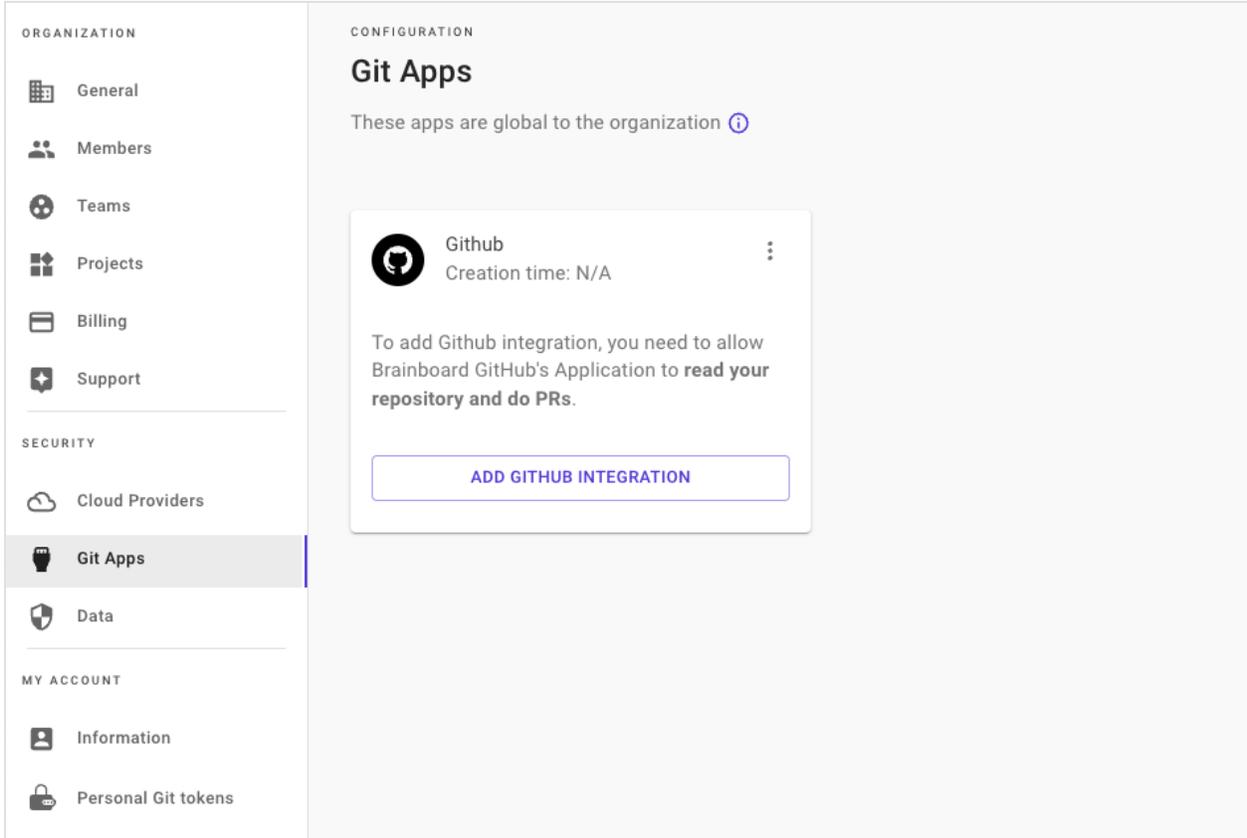
Brainboard supports 2 types of Git connections:

3.1. Git apps

This type of connections is done through the app registration, and the user management is done at your provider level and not in Brainboard
INFO

The only provider supported is Github. Azure DevOps will be added in the near future.

Here is the [link](#) to access Git apps settings page.



ORGANIZATION

- General
- Members
- Teams
- Projects
- Billing
- Support

SECURITY

- Cloud Providers
- Git Apps**
- Data

MY ACCOUNT

- Information
- Personal Git tokens

CONFIGURATION

Git Apps

These apps are global to the organization ⓘ

Github
Creation time: N/A

To add Github integration, you need to allow Brainboard GitHub's Application to **read your repository and do PRs.**

[ADD GITHUB INTEGRATION](#)

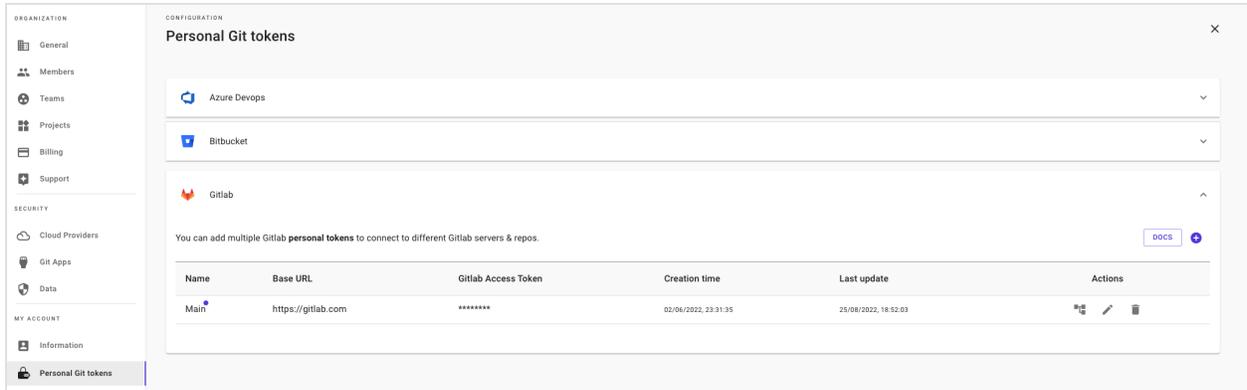
3.2. Add personal git tokens

Git personal tokens page allows you to store the tokens that you generate from your Git provider and set their scope (in which projects they will be used).

The Git providers supported are:

- Gitlab
- Azure DevOps
- Bitbucket

Here is the [link](#) to access Git apps settings page.



4. Set remote backend

It's a best practice to store the Terraform state generated after you provision your cloud infrastructure into a remote backend.

Brainboard allows you to set and use your own remote backend.

Here is the [link](#) to configure it.

ORGANIZATION

- General
- Members
- Teams
- Projects
- Billing
- Support

SECURITY

- Cloud Providers
- Git Apps
- Data**

MY ACCOUNT

- Information
- Personal Git tokens

SECURITY

Data Management

 Terraform Backend
Last update time: 2022-10-07 10:24

Please select a **Remote Backend** for your tfstate files.
N.B: it may take few minutes to migrate all your Terraform states into the target backend.

AWS S3

Region *
eu-west-3

Bucket Name *
security-topologies

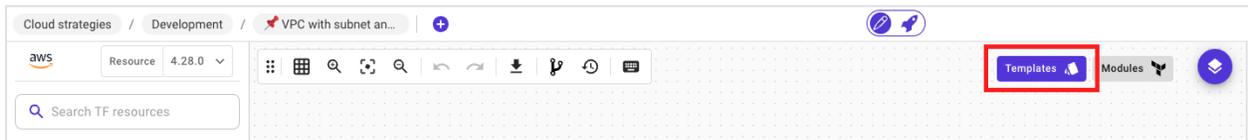
Azure Blob Storage

Brainboard backend (default)

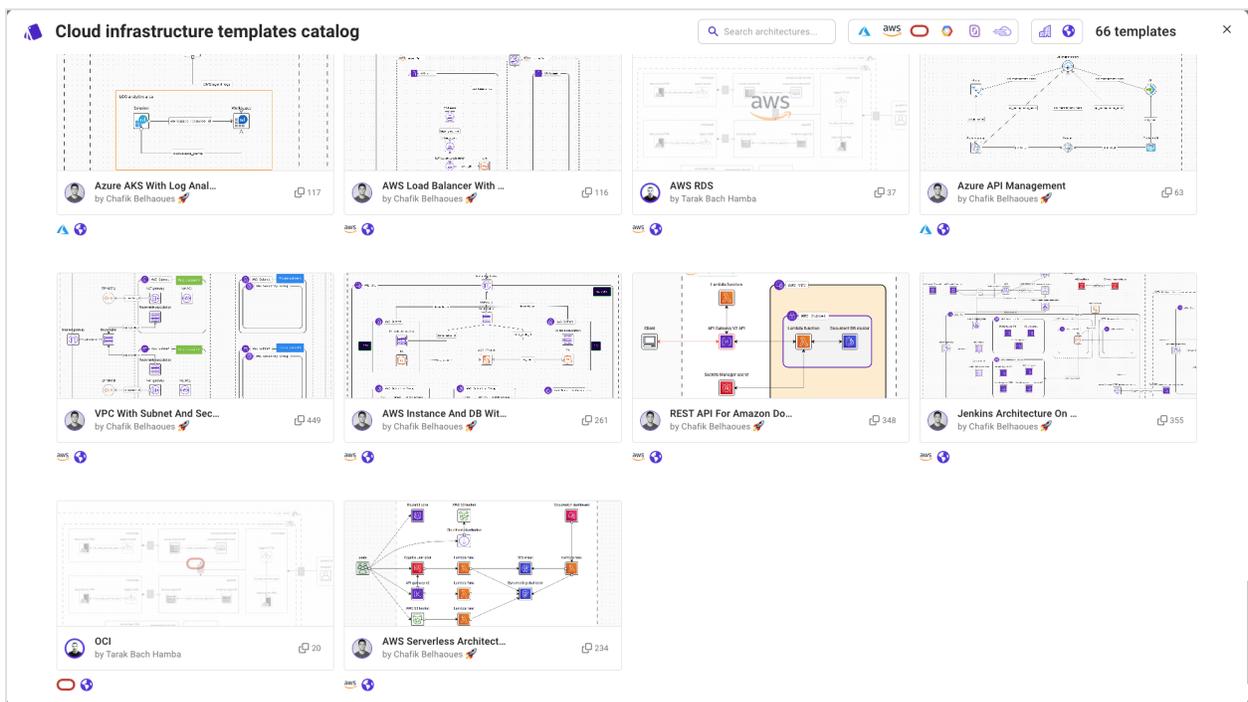
Start with a template

One of the fastest way to start with Brainboard, is to start by a template.

You can access the template catalogue from the design area with the button Templates:



You can search by keyword any architecture, for e.g. landing zone, kubernetes, security... or select your preferred cloud provider to only see the templates architectures for this specific provider:



TIP

There are 2 types of templates: public and private.

When you first create your account, you only see the public ones but you can convert any architecture that you make into a private template. Which allows you to build your internal library of templates to use off the shelf.