Projet DevOps

Azure

Terraform

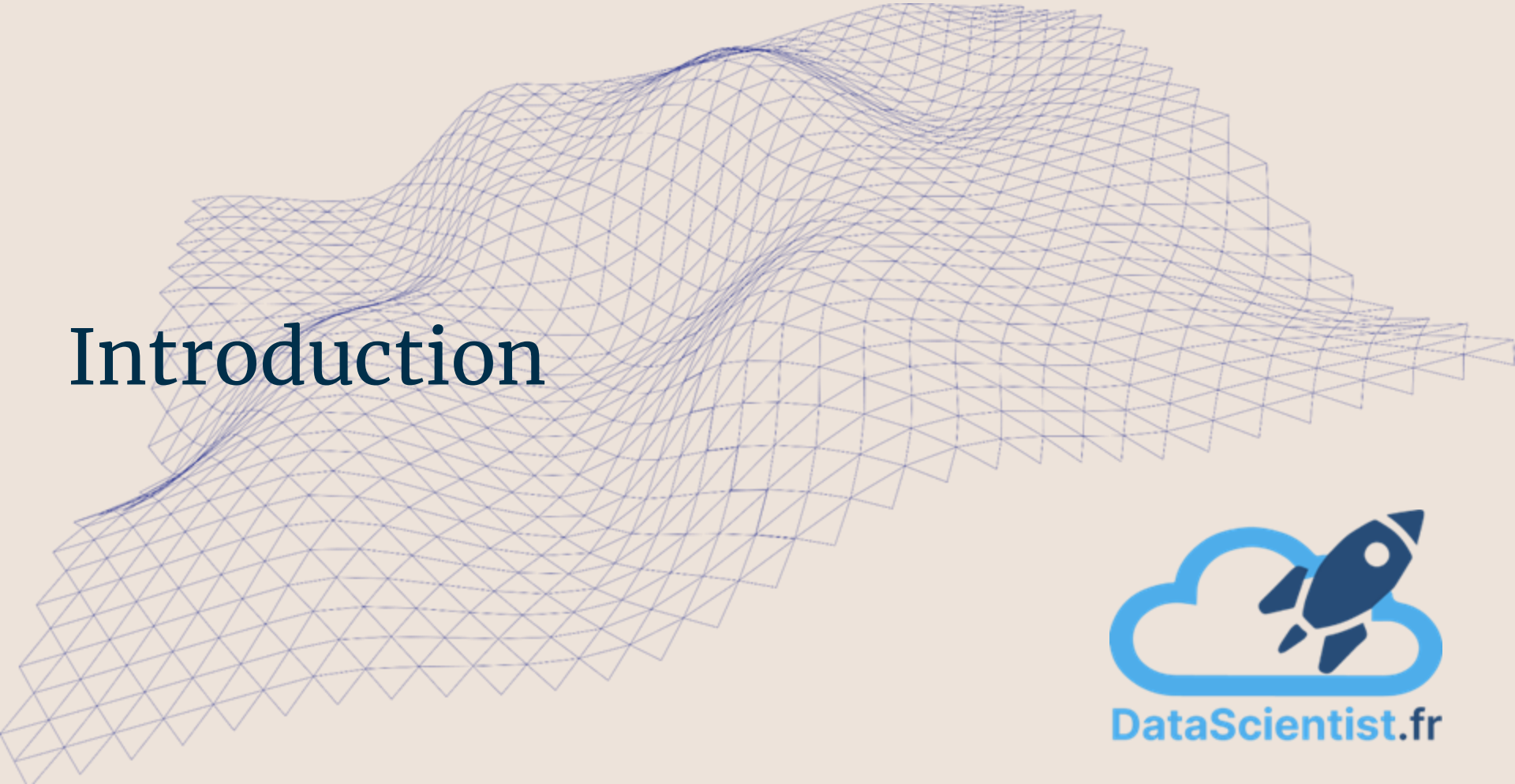
Kubernetes

DataScientist.fr

Sommaire

- Introduction
- Terraform : déploiement de l'infra
- Docker : push de l'image sur le Container Registry
- Kubernetes : déploiement de l'application
- Test
- Notation

Introduction



DataScientist.fr

Introduction

Le but de ce projet est de déployer une application Flask sur Azure avec Terraform et Kubernetes.

Vous devrez dans un premier temps déployer une infrastructure Azure avec Terraform, pour avoir un cluster Kubernetes et un registre Docker.

Ensuite, vous devrez créer une image Docker de l'application Flask, l'envoyer sur le registry Docker pour l'utiliser par la suite avec Kubernetes.

Enfin, vous devrez déployer les ressources nécessaires sur Kubernetes pour que l'application flask tourne et soit exposer sur internet.

Installer Azure CLI

Utilisez votre adresse mail étudiante pour créer un compte étudiant sur Microsoft Azure :

- https://azure.microsoft.com/fr-fr/free/students/

Installez Azure CLI :

- https://learn.microsoft.com/en-us/cli/azure/install-azure-cli

Connectez vous avec azure CLI :

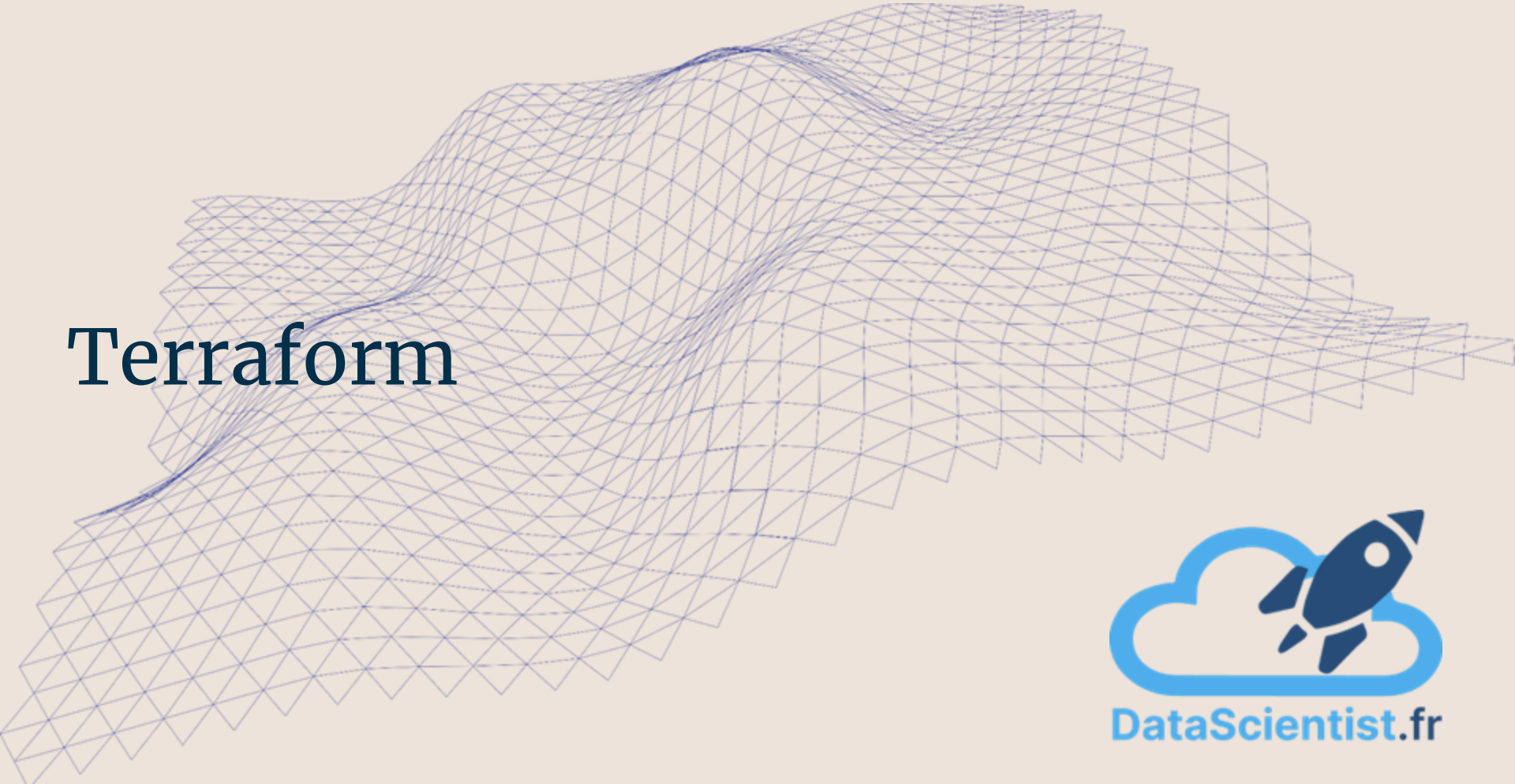
- `$ az login`

Téléchargement du projet

Téléchargez le fichier **Projet-DevOps.zip** qui contient l'application **flask-app** que vous devrez déployer.

Vous devrez completer les dossiers **kubernetes** et **terraform** ainsi que le fichier **README**.

Terraform



DataScientist.fr

Terraform : ressources à déployer

Groupe de ressource (resource group) :
- Nom du groupe dans Azure : *rg-ESGI-<student-name>*

Registre de conteneur (container registry) :
- Dans le groupe : *rg-ESGI-<student-name>*
- Sku : "Standard"

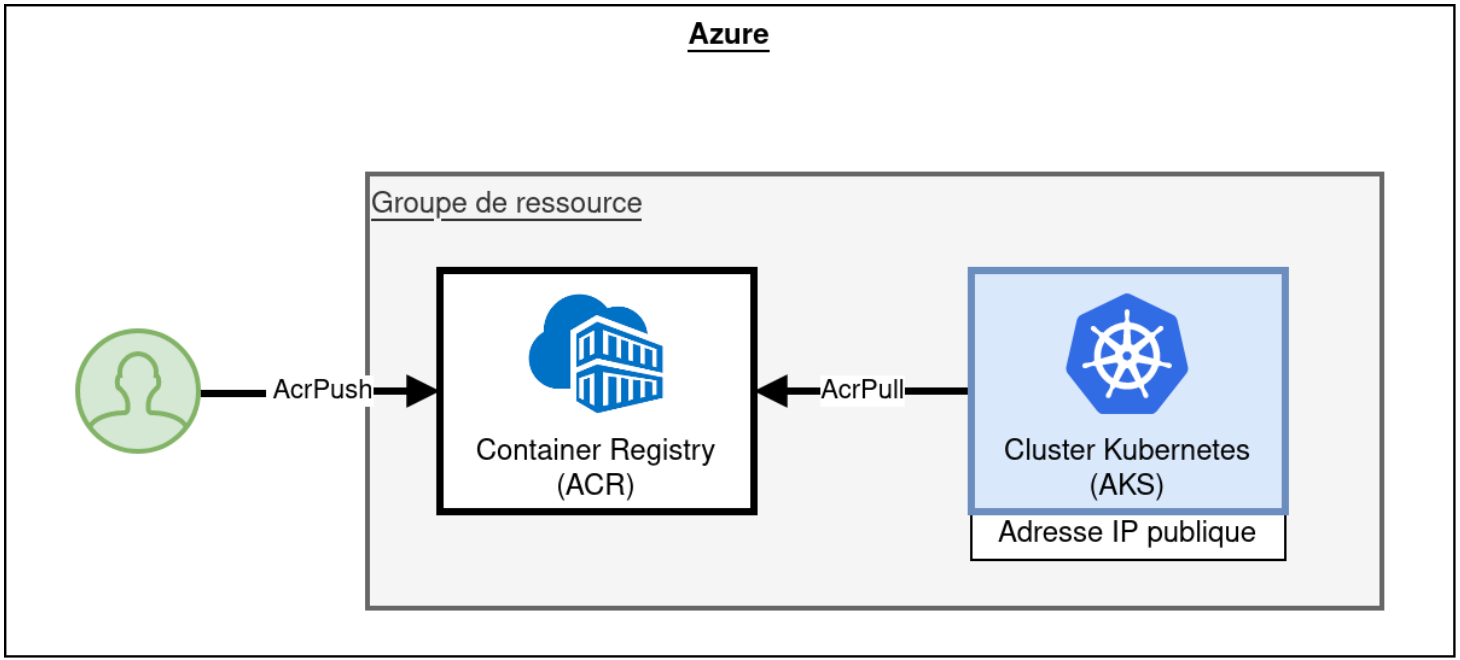
Cluster Kubeternes (kubernetes cluster) :
- Dans le groupe : *rg-ESGI-<student-name>*
- default_node_pool.vm_size : "Standard_B2s"

Adresse IP publique (public ip) :
- Dans le groupe créer pour les ressource kubernetes (*MC_<...>*)
- Devra être afficher dans l'output terraform (***output.ft***)

Des droits d'accès au registre de conteneur (role assignement) :
- **AcrPull** pour le cluster Kubernetes (Pour que le cluster puisse récupérer des images depuis votre registre)
- **AcrPush** pour votre utilisateur (Pour que vous puissiez envoyer vos images vers le registre)

DataScientist.fr

Terraform : schema d'infrastructure

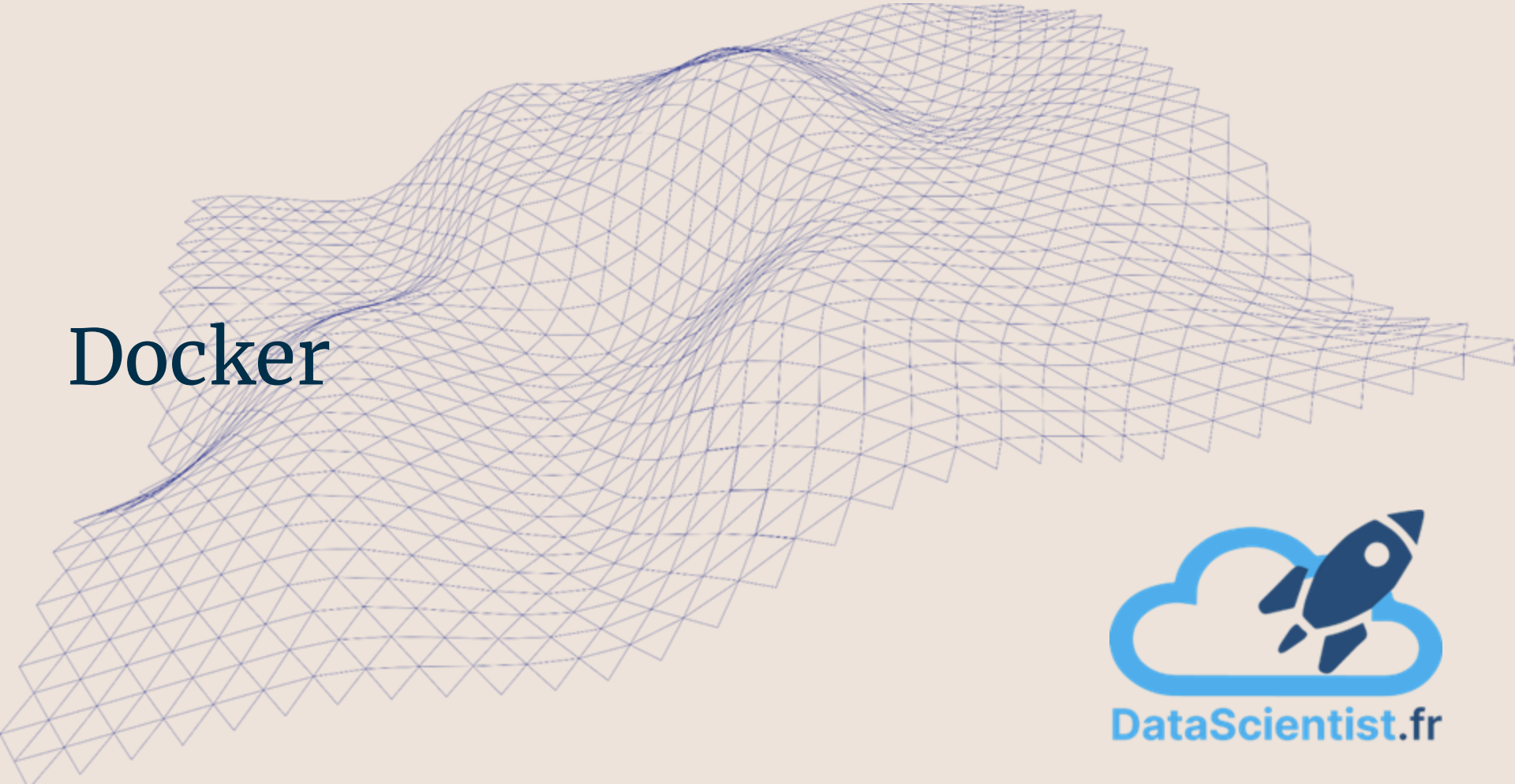


Terraform : documentation

Documentation :

- https://learn.microsoft.com/fr-fr/azure/developer/terraform/overview
- https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs

Docker



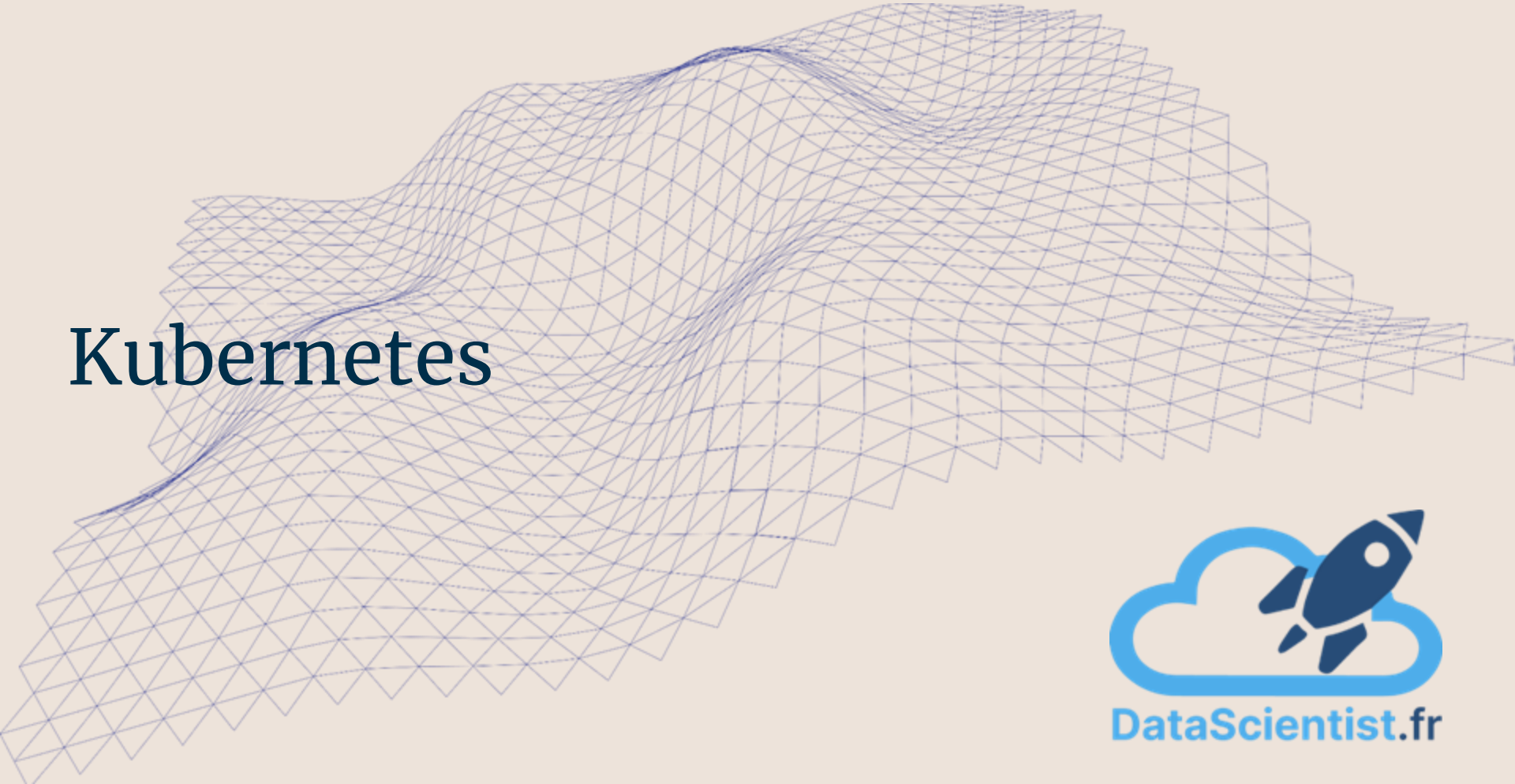
DataScientist.fr

Docker : push de l'image sur le Container Registry

Créez et pushez l'image Docker vers le registre de conteneur que vous avez créé dans Azure :

- Connectez vous à votre conteneur de registre : `az acr login --name <acr-name>`
- Créez l'image à partir du Dockerfile : `docker build <...>`
- Pushez l'image : `docker push <...>`

Kubernetes



DataScientist.fr

Kubernetes : déploiement de l'application

Connectez vous au cluster kubernetes :

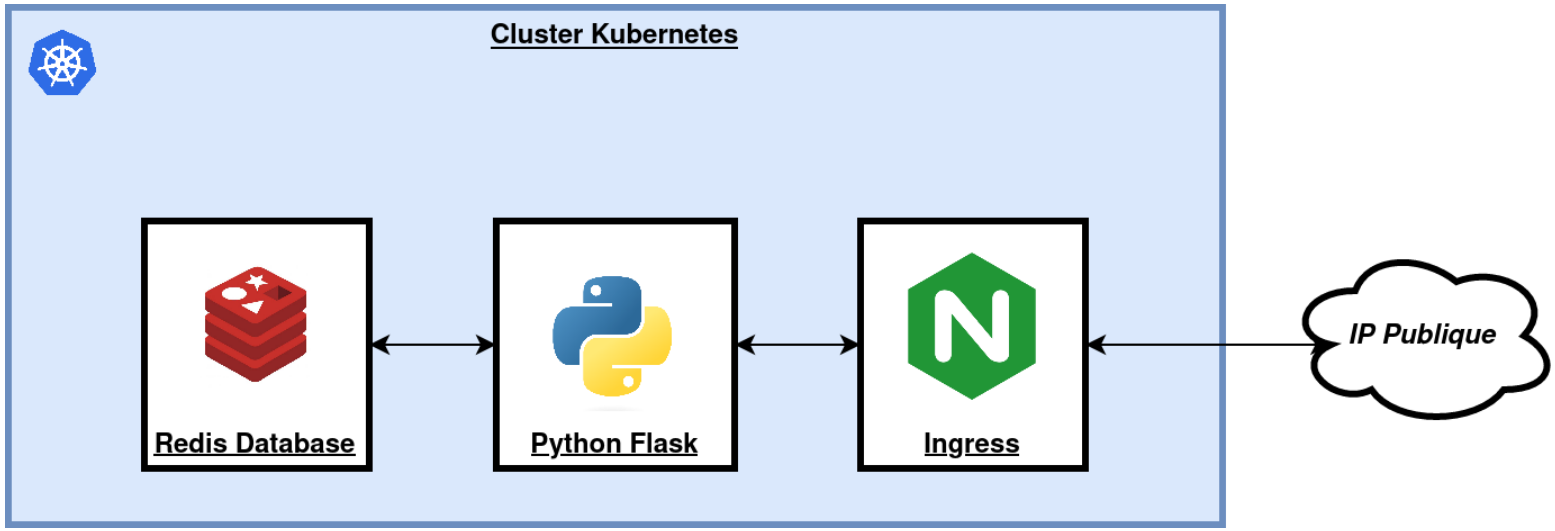
```
az aks get-credentials --overwrite-existing -n <cluster name>  -g <resource group name>
```

Avant de déployer l'application vous devrez **déployer un ingress controller (en utilisant helm)** sur votre cluster K8s.
Votre ingress controller doit être connecté à l'adresse ip publique que vous avez créer avec Terraform.
https://learn.microsoft.com/en-us/azure/aks/ingress-basic?tabs=azure-cli

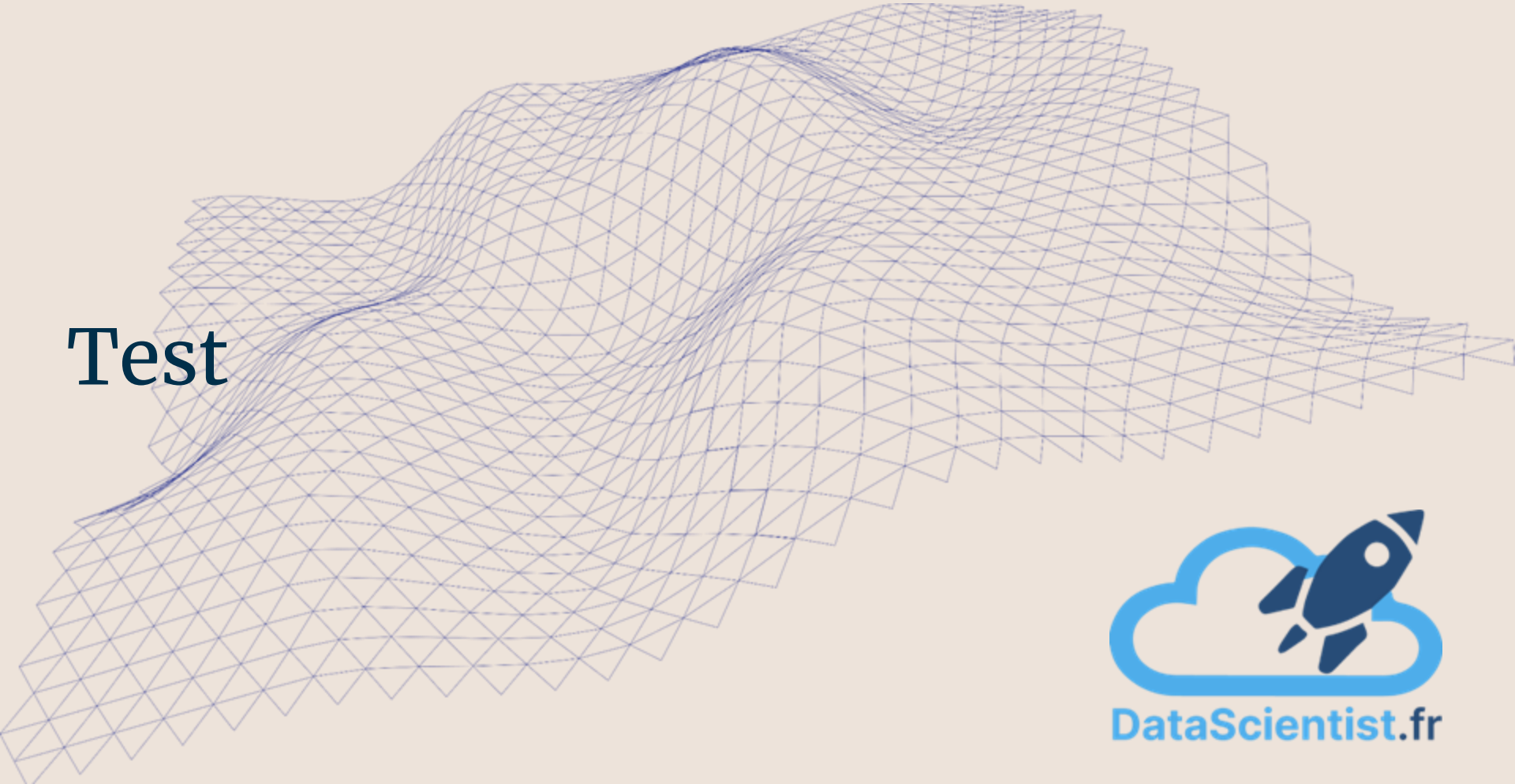
Vous devrez déployer :

- **L'application Flask** (qui sera connecté à la base de donnée redis et sera exposer sur l'ip publique via l'ingres.
- **Une base de données Redis.**
- **Un ingress** qui permet de rediriger le traffic de votre ip publique vers votre application flask.

Kubernetes : déploiement de l'application



Test



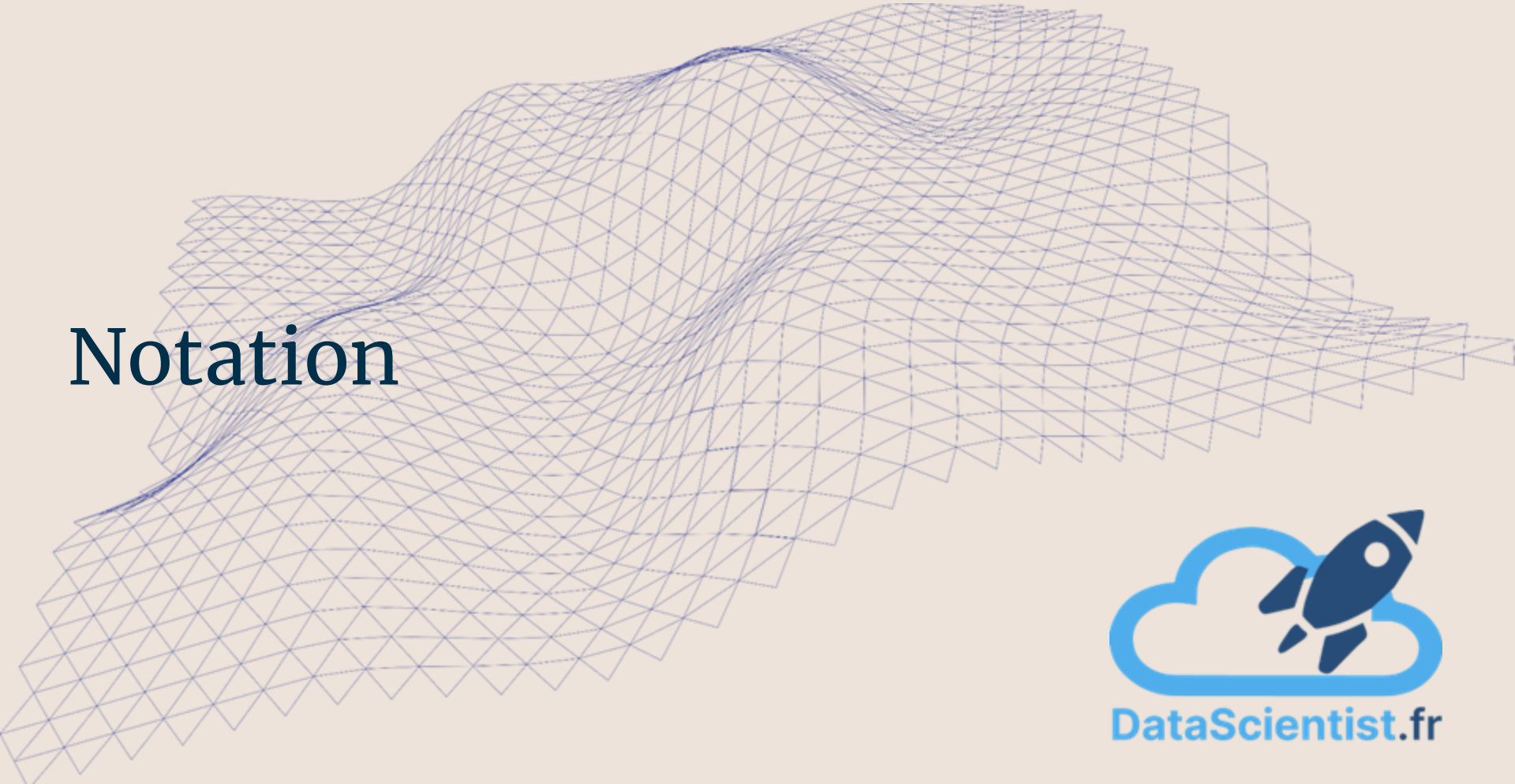
DataScientist.fr

Test

Pour tester votre infrastructure + application, il vous suffit simplement de faire une requête à l'ip publique que vous avez créer : `curl <PUBLIC-IP>`

Vous devriez voir le résultats : `This webpage has been viewed <X> time(s)`

Notation



DataScientist.fr

Notation

Créez un fichier zip qui contient les éléments suivants :

- ***Le dossier flask-app*** qui sera utilisé pour créer et push l'image docker vers le registre.
- Un ***dossier qui contient vos fichiers terraform*** (Inclut seulement les fichiers ***.tf***, ne doit pas contenir les fichiers ***.tfstate*** et ***.lock.hcl***)
- Un ***dossier qui contient vos fichier kubernetes*** (Inclut seulement les fichiers ***.yaml***)
- Un fichier ***README*** expliquant comment déployer l'infrastructure, build & push l'image doker et enfin comment appliquer la config kubernetes. Pensez à y inclure toutes les étapes pour déployer votre application sur votre infrastructure Azure, et notament les commandes pour :
    - Vous connecter à azure / terraform / docker
    - Déployer avec terraform / kubernetes / helm / docker

Envoyez votre fichier zip à l'adresse : romain@axitechnologies.fr avant le Vendredi 08/12/2023

Correction : Votre correcteur se contentera de suivre à la lettre les commandes indiquées dans le README pour tout déployer de A à Z. Si vous oubliez des étapes, vous serez pénalisé en conséquence.

DataScientist.fr

Suppression des ressources

Pensez à supprimer vos ressources sur Auzre pour ne pas dépenser vos crédits inutilement.

Pour supprimer vos ressource, vous devrez dans l'ordre :

- Supprimez vos ressource kubernetes: `kubectl delete -f kubernetes`
- Désinstallez le ingress controller: `helm uninstall ingress-nginx --namespace ingress-nginx`
- Supprimez vos ressources azure avec terraform: `terraform destroy`