|  |  |  |
| --- | --- | --- |
| Node | Un nœud est une machine de travail dans Kubernetes (sans lui pas de déploiement d'app) Chaque nœud contient les services nécessaires à l'exécution de pods et est géré par les composants du master. Ex : Dans notre cas on a qu’**UN** Node (puisqu'on qu'on a que 1 serveur : 129.194.10.126), nommé 'pinfo1'. Tous nos services/pods/ingress etc sont déployé dans ce Node. | /var/folders/36/lvgdw3zd2yvdd8k7yl139mf40000gn/T/com.microsoft.Word/WebArchiveCopyPasteTempFiles/cidE9F76F71-5E84-8C4D-BA2E-2B1BC80E3461.png |
| Pod | Il s'agit de la plus petite unité d'une application Kubernetes  Un pod peut contenir plusieurs conteneurs étroitement couplés (cas d'utilisation avancé) ou un conteneur unique (cas d'utilisation plus courant). Les conteneurs sont regroupés dans des pods Kubernetes afin d'optimiser le partage des ressources  Cette organisation des conteneurs en pods constitue la base de l'une des fonctions réputées de Kubernetes : la réplication | /var/folders/36/lvgdw3zd2yvdd8k7yl139mf40000gn/T/com.microsoft.Word/WebArchiveCopyPasteTempFiles/cidE9F76F71-5E84-8C4D-BA2E-2B1BC80E3461.png |
| Service | Une manière **abstraite** d'exposer une application s'exécutant sur un ensemble de Pods en tant que service réseau. Ex : notre Web-ui est un service, nos base de données sont des services. |  |
| KubeAdm | **sets up a minimal viable cluster.** -> grâce à lui on a un Node sur lequel on peut déployer nos application |  |
| Kubectl | The Kubernetes **command-line** tool, **kubectl**, allows you to **run commands** against Kubernetes clusters. You can **use** kubectl **to deploy** applications, inspect and manage cluster resources, and view logs |  |
| Kubelet | Kubelet **surveille** l'état d'un pod et s'il n'est pas dans l'état voulu, le pod sera redéployé sur le même node |  |
| Kubernetes | Kubernetes est une plate-forme pour la gestion de charges de travail (workloads) et de services conteneurisés |  |
| Helm charts | Regroupe tous les fichiers .yaml pour un déployment « d’un coup » de tous les microservices (fichiers .yaml) |  |
| ConfigMap | Fichier .yaml. C'est une configuration générale (externe) de nos variables, si on change quelque chose dans la config on va le changer pour toutes les (par ex :) bases de données qui pointent vers cette config | /var/folders/36/lvgdw3zd2yvdd8k7yl139mf40000gn/T/com.microsoft.Word/WebArchiveCopyPasteTempFiles/cid7C3C3D6D-692C-484B-A46E-79D154D4B6DE.png |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
| NodePort | Un NodePort est 1 des 3 moyens d'exposer (créer un accès depuis l'extérieur du cluster) un service. Ouvre un port spécifique sur tous les Nœuds (les VMs), et tout trafic envoyé vers ce port est transféré vers le service | /var/folders/36/lvgdw3zd2yvdd8k7yl139mf40000gn/T/com.microsoft.Word/WebArchiveCopyPasteTempFiles/cid9D9D40C8-7775-0246-BD37-95156B38154F.png |
| LoadBalancer | Un LoadBalancer est 1 des 3 moyens d'exposer (créer un accès depuis l'extérieur du cluster) un service. Dans le fichier yaml, on va attacher un loadbalancer au service et celui-ci sera accessible depuis l'extérieur du server. Ex : notre web-ui est de type LoadBalancer | /var/folders/36/lvgdw3zd2yvdd8k7yl139mf40000gn/T/com.microsoft.Word/WebArchiveCopyPasteTempFiles/cid13FE4C2A-0410-3C47-A4EC-3231F4B4BD98.png |
| Ingress | Une Ingress est 1 des 3 moyens d'exposer (créer un accès depuis l'extérieur du cluster) un service. Il agit comme un 'smart router'. Permet entre autre de créer des 'path' dans l'url. Ex /Groups, /Profile | /var/folders/36/lvgdw3zd2yvdd8k7yl139mf40000gn/T/com.microsoft.Word/WebArchiveCopyPasteTempFiles/cid12862A60-1179-B04D-9A83-49D49A0F0D1D.png |
|  |  |  |
|  |  |  |
|  |  |  |
| Docker | Permet de lancer des applications dans des conteneurs linux. Permet d'empaqueter une application et ses dépendances dans un conteneur isolé, qui pourra être exécuté sur n'importe quel serveur |  |
| Image (docker) | Une image docker représente le système de fichiers, sans les processus. Elle contient tout ce que vous avez décidé d'y installer (Java, une base de donnée, un script que vous allez lancer, etc…), mais est dans un état inerte  (~fichier exécutable) |  |
| Dockerfile | Les Dockerfiles sont des fichiers qui permettent de construire une image Docker adaptée à nos besoins, étape par étape. (~Makefile pour docker) |  |
| DockerHub | Site web où on va push nos images docker, pour pouvoir ensuite les récupérer depuis une autre machine. (Comme Github mais pour les images docker) Ex : nous on push nos images sur dockerHub et on va ensuite les récupérer sur le serveur pour les déployer dessus en tant que services. |  |
| Fichiers .yaml | Fichier de configuration. Utile pour indiqué les étapes à reproduire à notre CI (Travis). Utile pour l'automatisation des images (build depuis dockerfile, tag + push image sur DockerHub). Utile pour donner un rôle (service, pod, ingress, deployment etc..) à nos image docker pour le déploiement. Ex : le web-ui est un service, on le dit dans le .yaml avant le déploiement de celui-ci |  |
| Travis (continuous integration) | (Site Web) Travis CI est un logiciel d'intégration continue. Permet de compiler, tester et déployer le code source des logiciels développés. |  |
| GitHub | Site Web qui permet de stocker notre code dans un lieu commun et accessible à l'équipe. |  |
| SonarCloud | Site Web qui permet de vérifier que notre code est bien sécurisé en terme de coverage, code smells, security, bugs etc… |  |