

# PROJET FINAL DEVOPS

## Groupe 3

- Majd EL KHATIB (OP CONSULTING)
- Christophe GARCIA (SPIE ICS Toulouse)
- Romain REVEL (SPIE ICS Grenoble)

# Introduction

# Contexte

## Formation

- chez AJC Formation
- du 25/07/2022 au 13/10/2022
- sur le **DevOps**

Ce document est le rendu du projet final

# Rappel du sujet

La société IC GROUP souhaite mettre sur pied un site **web vitrine** devant permettre d'accéder à ses 02 applications phares: **Odoo** et **pgAdmin**

# Analyse du sujet

## Partie 1

- Conteneurisation de la web app python (Flask) avec **Docker**
- Déploiement des 3 produits sur un cluster **Kubernetes**

## Partie 2

- Mise en place d'un pipeline CI/CD à l'aide de **Jenkins** et de **Ansible**

# Méthodologie

2 possibilités:

- Coopération: chacun traite un point particulier de son côté puis mise en commun
- Collaboration: tout le monde se concentre sur le même point particulier

Source: <https://www.votre-it-facile.fr/travail-collaboratif-et-travail-cooperatif-difference/>

# Choix des outils

- Communication (voix / texte / partage d'écran)
  - Discord
- Versionnement
  - Git
  - Github
  - Gitkraken: client Git **beau et ergonomique**
- Infrastructure
  - Virtualbox
  - Vagrant (outil de provisioningning)

Viewing 5/5 Show All

Filter (Ctrl + Alt + f)

LOCAL 2/2

 dev

main

Cloud

REMOTE 3/3

STASHES 1

PULL REQUESTS 0

ISSUES

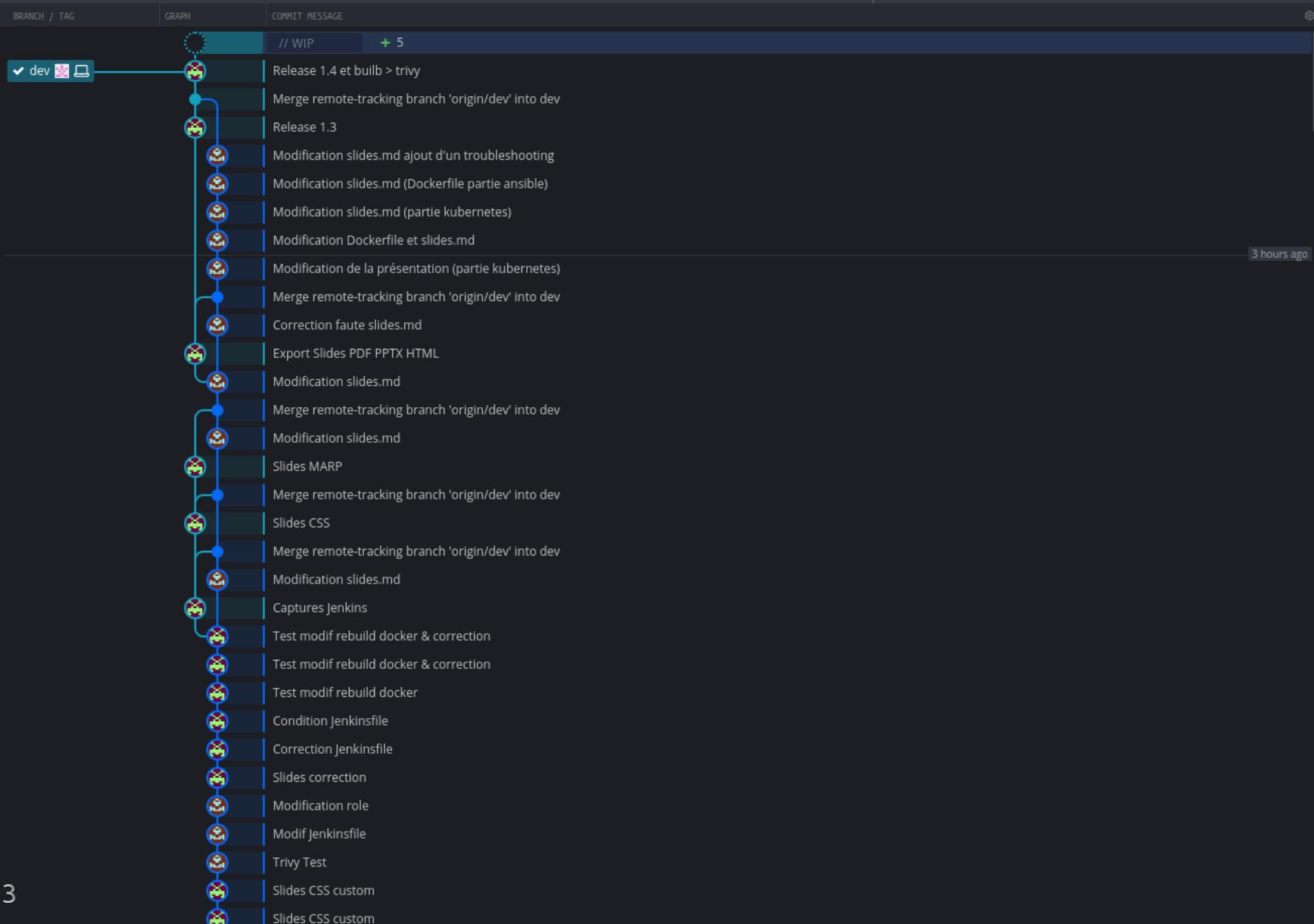
TEAMS

TAGS 0/0

SUBMODULES 0

GITHUB ACTIONS 0

Groupe 3



- 5 file changes on dev
- Path Tree
- Unstaged Files (5) Stage all changes
- + d/2022-10-12-20-59-14.039-VBoxHeadless-99543.log
  - + d/2022-10-12-20-59-14.041-VBoxHeadless-21103.log
  - + d/2022-10-12-20-59-14.049-VBoxHeadless-15815.log
  - + d/2022-10-12-20-59-14.052-VBoxHeadless-14239.log
  - + presentation/test.md
- Staged Files (0)

Commit Message  Amend

Summary

Description

Slide 8 sur 115

Stage files changes to commit

100% Support FREE 8.10.0

# Partie 1

Approche Kubernetes

# Infrastructure

- Github  
<https://github.com/Romain-Revel/ajc-projet-final>
- Docker  
<https://hub.docker.com/repository/docker/sh0t1m3/ic-webapp>
- Kubernetes / Minikube
- Postes: **Windows 10 et Linux Mint** -> complications

# Vagrant

Utilisation d'un fichier Vagrant fourni par **Dirane** lors de notre formation, adapté pour répondre à notre besoin.

```
# Version initiale fonctionnant uniquement sous Windows
Vagrant.configure("2") do |config|
  config.vm.define "docker" do |docker|
    docker.vm.box = "geerlingguy/centos7"
    docker.vm.network "private_network", type: "static", ip: "192.168.99.11"
    docker.vm.hostname = "docker"
    docker.vm.provider "virtualbox" do |v|
      v.name = "docker"
      v.memory = 1024
      v.cpus = 2
    end
    docker.vm.provision :shell do |shell|
      shell.path = "install_docker.sh"
      shell.env = { 'ENABLE_ZSH' => ENV['ENABLE_ZSH'] }
    end
  end
end
```

Comme nous utilisons deux environnements différents, nous avons fait un module dans le fichier vagrant (en ruby). Ce module sera utilisé pour tous les fichiers vagrant qui suivront.

```
# Module pour gérer l'OS hôte
module OS
    def OS.windows?
        (/cygwin|mswin|mingw|bccwin|wince|emx/ =~ RUBY_PLATFORM) != nil
    end
    def OS.mac?
        (/darwin/ =~ RUBY_PLATFORM) != nil
    end
    def OS.unix?
        !OS.windows?
    end
    def OS.linux?
        OS.unix? and not OS.mac?
    end
end
```

# Voici comment utiliser le module.

```
# Ajout
docker.vm.box = "geerlingguy/centos7"
if OS.linux?
  # Sous linux, il FAUT préciser le nom du réseau hôte
  # https://www.vagrantup.com/docs/providers/virtualbox/networking
  # Dans Virtualbox > Fichier > Gestionnaire de réseau hôte (CTRL + H):
  # - Vérifier la présence de vboxnet0, sinon le créer
  # - Vérifier l'adresse IPv4 et le masque, sinon les modifier (à faire 2 fois pour être pris en compte)
  #
  # Vérifier avec "ip -a" le nom, l'IP et le masque
  docker.vm.network "private_network", type: "static", ip: "192.168.99.11", name: "vboxnet0"
elsif OS.windows?
  docker.vm.network "private_network", type: "static", ip: "192.168.99.11"
else
  puts 'OS not managed'
end
# ...
```

# Conteneurisation de la web app

La première étape est de procéder à la conteneurisation de l'application web vitrine.

<https://github.com/sadofrazer/ic-webapp.git>

Comment l'intégrer dans notre repo git ?

- Copier-coller -> le plus simple
- Git submodules -> trop compliqué et risqué
- Git subtrees -> pas le temps

Pour cela un fichier Dockerfile a été créé enfin de générer une image.

```
# Dockerfile ic-webapp
FROM python:3.6-alpine

ENV ODOO_URL=""
ENV PGADMIN_URL=""

# Install python and pip
RUN apk add --no-cache --update bash && \
    # Install dependencies
    pip3 --no-cache-dir install Flask && \
    # Add a user icwebapp
    addgroup -S icwebapp && \
    adduser -S icwebapp -G icwebapp
# Add our code
COPY --chown=icwebapp:icwebapp ic-webapp /opt/ic-webapp/
USER icwebapp
WORKDIR /opt/ic-webapp

EXPOSE 8080
CMD [ "python3", "app.py" ]
```

On peut voir que pour des raisons de bonnes pratiques, nous avons créé un USER (icwebapp) qui lancera l'application.

Bien entendu, il faut tester l'application. Afin de répéter les commandes un script a été créé.

```
#!/bin/bash
# Script avec les commandes séquentielles
image="ic-webapp"
name="test-ic-webapp"
port="8080"
docker stop ${name} && docker rm ${name}
# Build soit en taggant directement soit en retaggant:
# docker tag SOURCE_IMAGE[:TAG] TARGET_IMAGE[:TAG]
docker build -t sh0t1m3/${image}:1.0 .
docker run -d -p ${port}:${port} \
-e OD00_URL='https://www.odoo.com/' \
-e PGADMIN_URL='https://www.pgadmin.org/' \
--name=${name} sh0t1m3/${image}:1.0
sleep 3
curl http://localhost:${port}
```



Intranet Applications

Rechercher

Search



Une fois l'image construite, et validée après le test d'un conteneur. Nous pouvons publier l'image sur le registry dockerhub.

```
#!/bin/bash
# Script de publication et de nettoyage
image="ic-webapp"
name="test-ic-webapp"

docker stop ${name}
docker rm ${name}

docker login
docker push sh0t1m3/${image}:1.0

# docker rmi ${image}
# docker rm $(docker ps -aq)
# docker rmi $(docker images -aq)
```

Docker Hub — Mozilla Firefox

AJC-PROJET-FINAL [Jenkins] Docker Hub

https://hub.docker.com/repository/docker/sh0t1m3/ic-webapp

docker hub Search for great content (e.g., mysql) Explore Repositories Organizations Help Upgrade sh0t1m3

sh0t1m3 Repositories ic-webapp Using 1 of 1 private repositories. Get more

General Tags Builds Collaborators Webhooks Settings

**i** Add a short description for this repository  
The short description is used to index your content on Docker Hub and in search engines. It's visible to users in search results. [Update](#)

**sh0t1m3 / ic-webapp**

Description  
*This repository does not have a description* [Edit](#)

Docker commands  
To push a new tag to this repository,  
`docker push sh0t1m3/ic-webapp:tagname` [Public View](#)

Last pushed: 9 minutes ago

Tags and scans VULNERABILITY SCANNING - DISABLED [Enable](#)

This repository contains 1 tag(s).

Tag	OS	Pulled	Pushed
1.0	Ubuntu	--	9 minutes ago

[See all](#) [Go to Advanced Image Management](#)

Automated Builds  
Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.  
Available with Pro, Team and Business subscriptions.

[Upgrade](#) [Learn more](#)

README [Edit](#)

Repository description is empty. Click [here](#) to edit.

Groupe 3 Slide 19 sur 115

# Déploiement avec Kubernetes

Afin de pouvoir déployer la totalité de l'environnement, dans cette approche nous allons utiliser kubernetes (minikube)

On a installé longhorn qui est chargé de gérer les PVs (Persistent Volume)

- Installation de longhorn (avec helm)
- Création des manifests
- Application des manifests

# Helm

Kubernetes package manager : <https://helm.sh/>

```
#!/bin/bash
# Helm installation
HELM_PATH=$(which helm)

if [ "${HELM_PATH}" == "" ]; then
    # Get HELM
    curl https://get.helm.sh/helm-v3.10.0-linux-amd64.tar.gz -o helm-v3.10.0-linux-amd64.tar.gz
    tar xfz helm-v3.10.0-linux-amd64.tar.gz
    sudo mv linux-amd64/helm /usr/local/bin && chmod +x /usr/local/bin/helm
    rm -Rf tar xfz helm-v3.10.0-linux-amd64.tar.gz linux-amd64/
    HELM_PATH=/usr/local/bin/helm
fi
```

# Longhorn

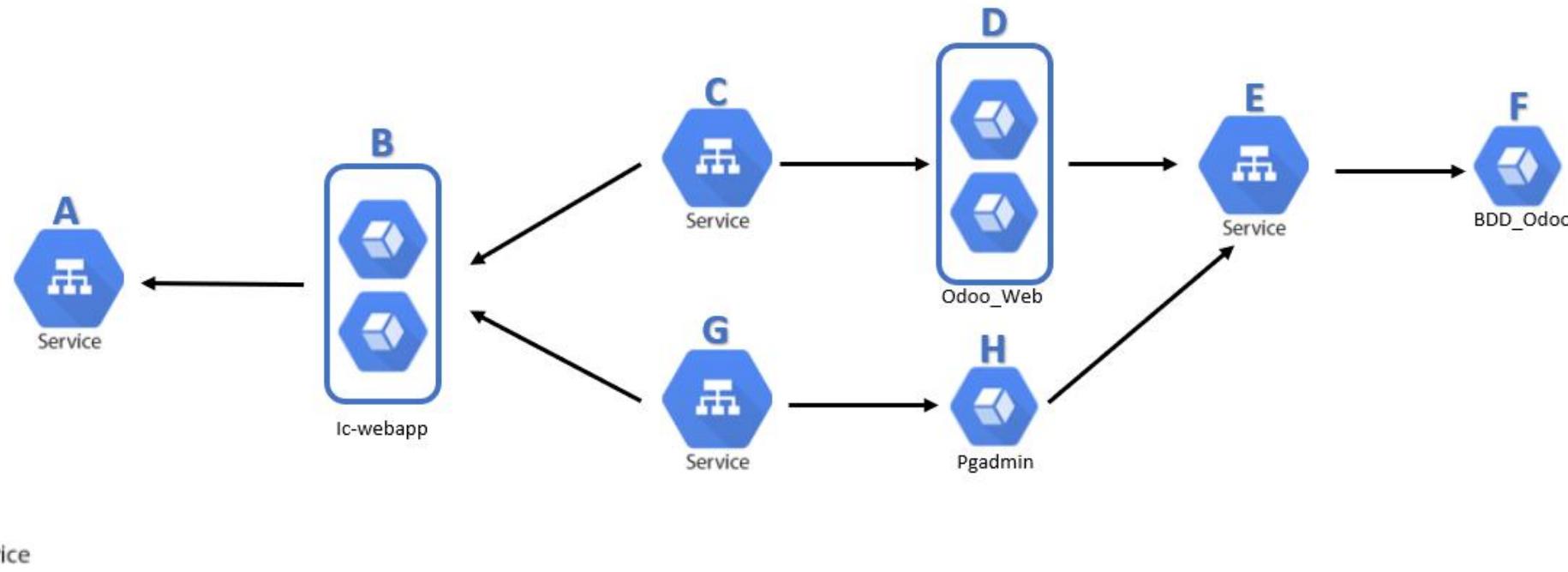
## Cloud native distributed block storage for Kubernetes

```
# Longhorn installation with helm
su vagrant -c "${HELM_PATH} repo add longhorn https://charts.longhorn.io"
if [ $? -ne 0 ]; then
    echo "Impossible d'ajouter le repo à helm"
    exit 1
fi

su vagrant -c "${HELM_PATH} repo update"

su vagrant -c "${HELM_PATH} install longhorn longhorn/longhorn --namespace longhorn-system --create-namespace"
if [ $? -ne 0 ]; then
    echo "Impossible d'installer longhorn"
    exit 1
fi
```

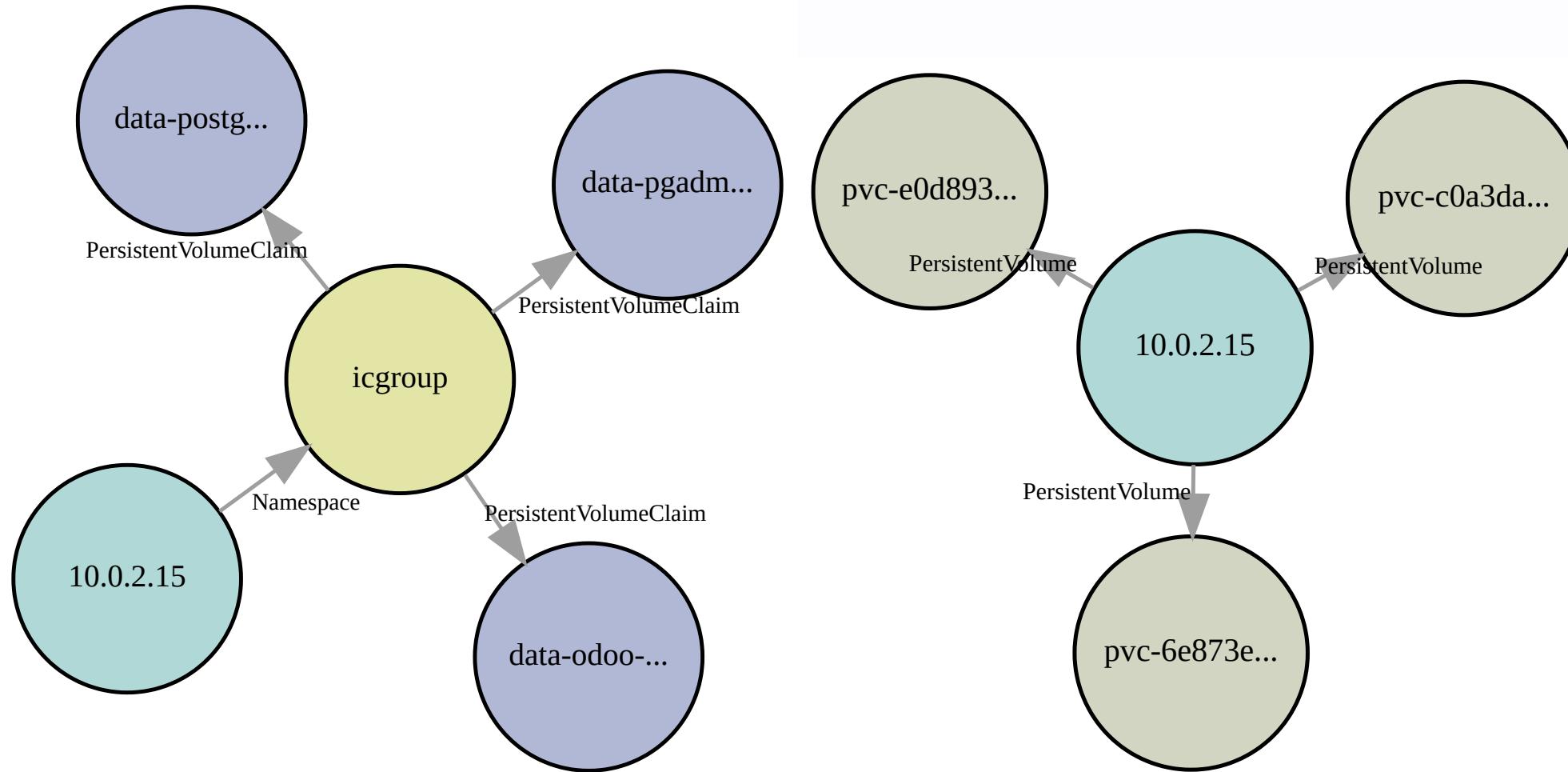
# Architecture du namespace **icgroup**



# Les fichiers descriptifs yaml (manifestes) pour kubernetes

```
# Le namespace
kubectl apply -f 01-create-namespace.yml
kubectl create secret generic odoo-pgsql-password --from-literal=odoo=YOUR_PASSWORD -n icgroup
kubectl create secret generic pgadmin --from-literal=pgadmin-password=YOUR_PASSWORD -n icgroup
# La base de données
kubectl apply -f 02-pvc-postgresl.yml
kubectl apply -f 04-BDD_Odoo.yml
kubectl apply -f 05-service-pgsql.yml
# L'ERP Odoo
kubectl apply -f 06-pvc-odoo.yml
kubectl apply -f 07-Odoo_Web.yml
kubectl apply -f 08-odoo-service.yml
# PGadmin
kubectl apply -f 09-pvc-pgadmin.yml
kubectl apply -f 11-config_map-pgadmin.yml
kubectl apply -f 12-pgadmin.yml
kubectl apply -f 13-pgadmin-service.yml
# IC Webapp (site vitrine)
kubectl apply -f 14-ic-webapp.yml
kubectl apply -f 15-ic-webapp-service.yml
```

# Liste des PVC



- **Deployments**

Un **Deployment** fournit des mises à jour déclaratives pour les **Pods** et les **ReplicaSets**.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: pgadmin
  namespace: icgroup
spec:
  replicas: 1
  selector:
    matchLabels:
      app: pgadmin
      env: prod
```

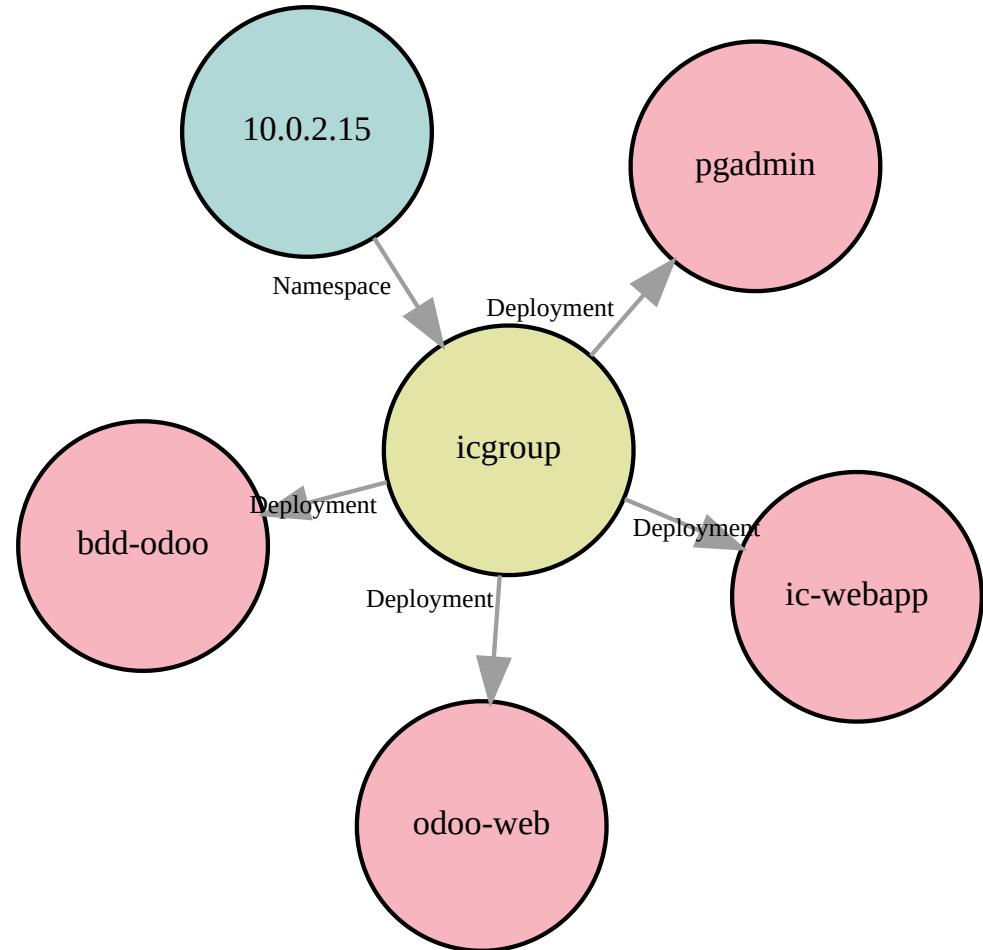
```
template:  
  metadata:  
    labels:  
      app: pgadmin  
      env: prod  
  spec:  
    securityContext:  
      runAsUser: 5050  
      runAsGroup: 5050  
      fsGroup: 5050  
      fsGroupChangePolicy: "OnRootMismatch"  
  volumes:  
    - name: pgadmin-config  
      configMap:  
        name: pgadmin-config  
    - name: pgadmin-data  
      persistentVolumeClaim:  
        claimName: data-pgadmin-claim
```

```
containers:
  - name: pgadmin
    #securityContext:
      #readOnlyRootFilesystem: true
    image: dpage/pgadmin4:6.14
    env:
      - name: PGADMIN_LISTEN_ADDRESS
        value: 0.0.0.0
      - name: PGADMIN_PORT
        value: "80"
      - name: PGADMIN_DEFAULT_EMAIL
        value: user@domain.com
      - name: PGADMIN_DEFAULT_PASSWORD
        valueFrom:
          secretKeyRef:
            name: pgadmin
            key: pgadmin-password
    ports:
      - name: http
        containerPort: 80
        protocol: TCP
    volumeMounts:
      - name: pgadmin-config
        mountPath: /pgadmin4/servers.json
        subPath: servers.json
        readOnly: true
      - name: pgadmin-data
        mountPath: /var/lib/pgadmin
    readinessProbe:
      httpGet:
        path: /
        port: 80
    resources: {}
      #requests:
        #memory: "300Mi"
        #cpu: "100m"
      #limits:
        #memory: "300Mi"
        #cpu: "200m"
```

# Le ConfigMap pour le fichier servers.json

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: pgadmin-config
  namespace: icgroup
data:
  servers.json: |
    {
      "Servers": {
        "1": {
          "Name": "Minimally Defined Server",
          "Group": "Server Group 1",
          "Port": 5432,
          "Username": "odoo",
          "Host": "odoo-postgres",
          "SSLMode": "prefer",
          "MaintenanceDB": "postgres"
        }
      }
    }
```

# Liste des déploiements



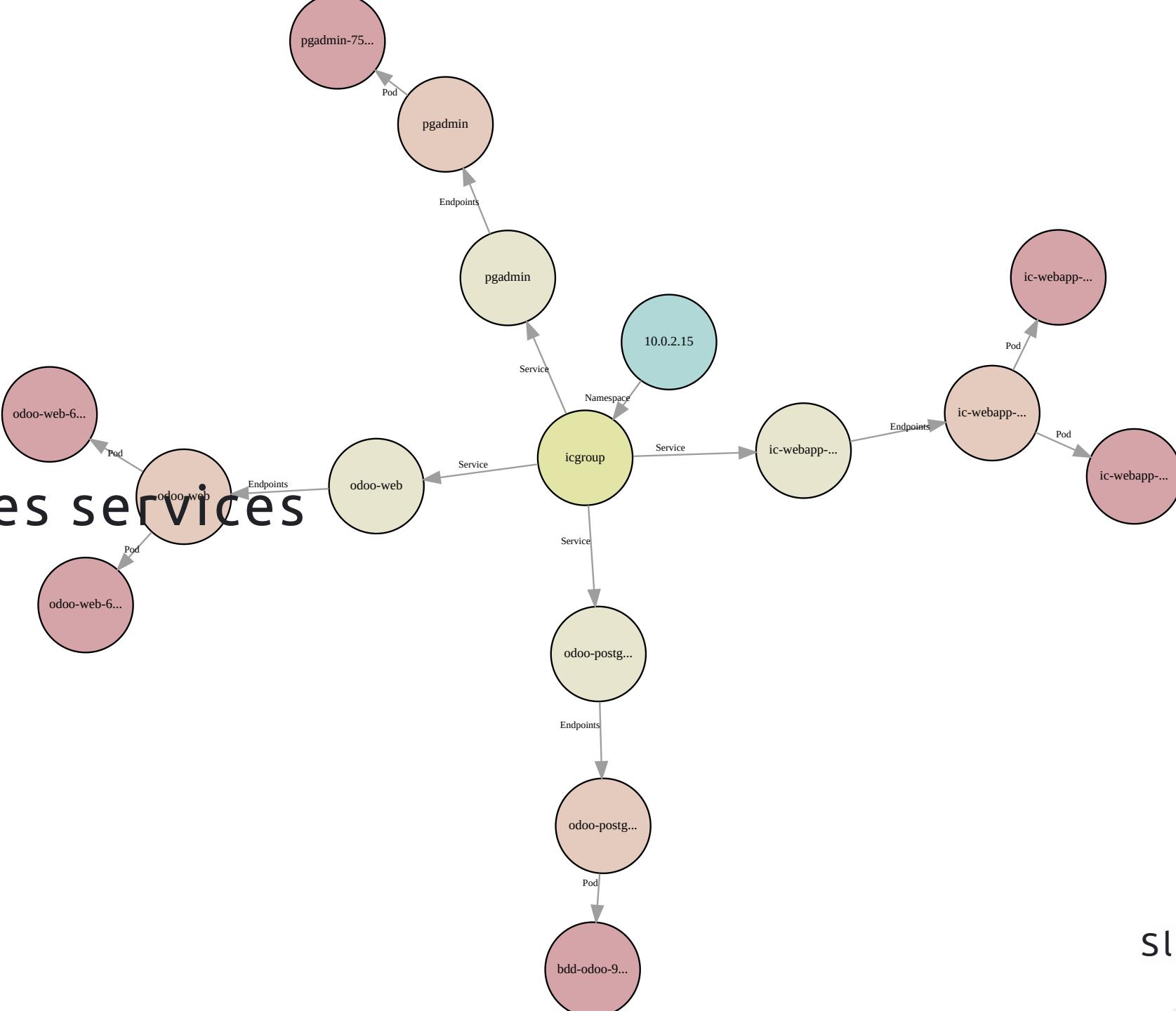
- Services
  - ClusterIP
  - NodePort

## Exemple de génération de fichier en ligne de commande

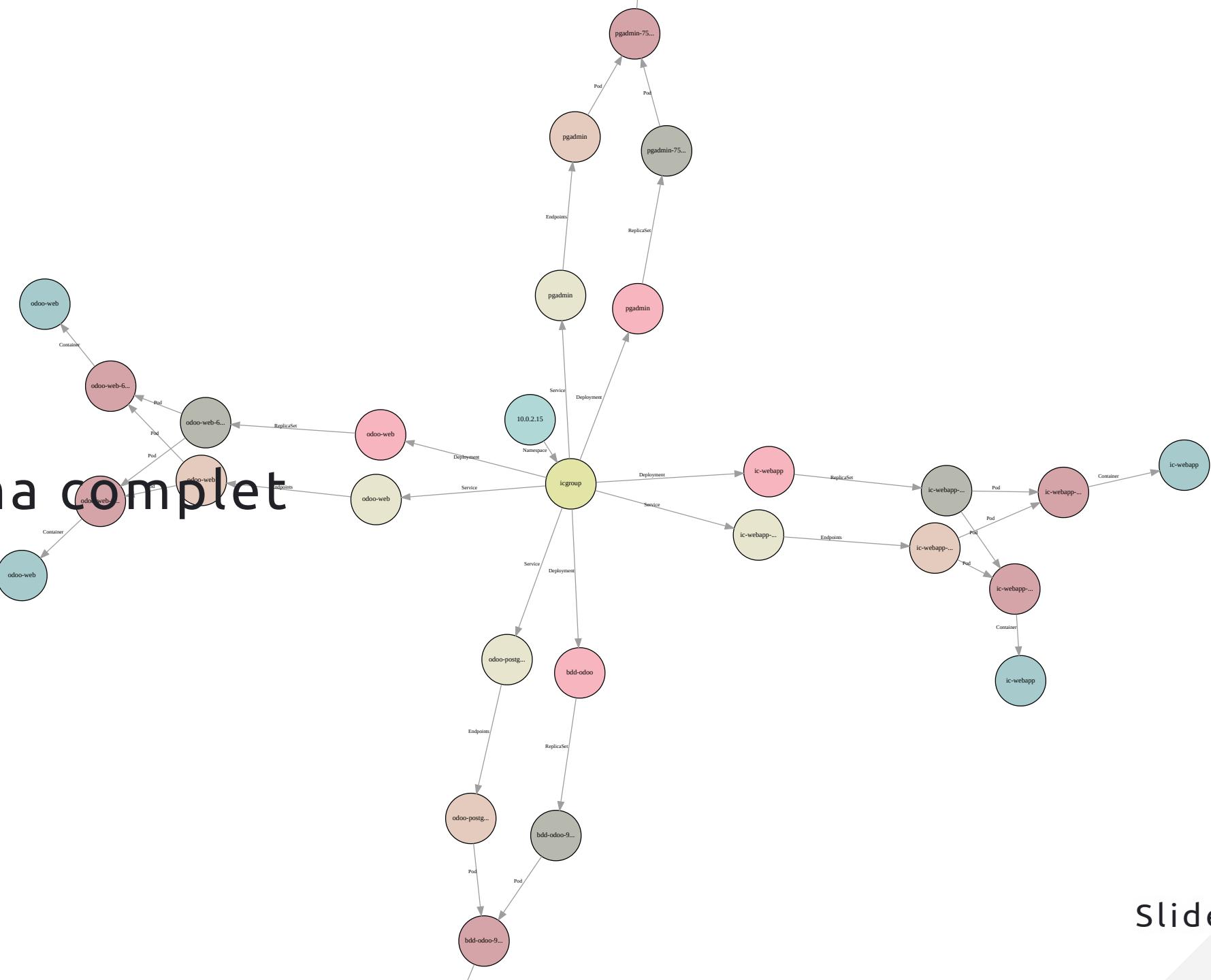
```
kubectl expose deploy bdd-odoo --port=5432 --type=ClusterIP \
--target-port=5432 --name=odoo-postgres -n icgroup -o yaml --dry-run=client
```

```
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: null
  labels:
    app: odoo-pgsql
    env: prod
  name: odoo-postgres
  namespace: icgroup
spec:
  ports:
  - port: 5432
    protocol: TCP
    targetPort: 5432
  selector:
    app: odoo-pgsql
    env: prod
    type: ClusterIP
status:
  loadBalancer: {}
```

# Liste des services



# Schéma complet



## Architecture fonctionnelle :

```
[vagrant@icwebapp manifests]$ kubectl get all -n icgroup
```

NAME	READY	STATUS	RESTARTS	AGE
pod/bdd-odoo-9457f8b55-2pwnb	1/1	Running	0	57s
pod/ic-webapp-c6cd4866c-fnhhg	1/1	Running	2 (7h59m ago)	15h
pod/ic-webapp-c6cd4866c-p9tns	1/1	Running	2 (8h ago)	15h
pod/odoo-web-6bb458fdcf-4wzdc	1/1	Running	0	57s
pod/odoo-web-6bb458fdcf-v9hqp	1/1	Running	0	57s
pod/pgadmin-75454655f7-8tp9t	1/1	Running	0	57s

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
service/ic-webapp-service	NodePort	10.104.163.187	<none>	25000:31500/TCP	15h
service/odoo-postgres	ClusterIP	10.98.234.144	<none>	5432/TCP	15h
service/odoo-web	NodePort	10.99.27.236	<none>	25000:31000/TCP	15h
service/pgadmin	NodePort	10.110.169.153	<none>	25000:32000/TCP	15h

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
deployment.apps/bdd-odoo	1/1	1	1	15h
deployment.apps/ic-webapp	2/2	2	2	15h
deployment.apps/odoo-web	2/2	2	2	15h
deployment.apps/pgadmin	1/1	1	1	15h

NAME	DESIRED	CURRENT	READY	AGE
replicaset.apps/bdd-odoo-9457f8b55	1	1	1	15h
replicaset.apps/ic-webapp-c6cd4866c	2	2	2	15h
replicaset.apps/odoo-web-6bb458fdcf	2	2	2	15h
replicaset.apps/pgadmin-75454655f7	1	1	1	15h

HOME | IC GROUP

192.168.99.9:31500

# IC GROUP

Intranet Applications

Rechercher  Search

Odoo

192.168.99.9:31000/web/database/selector

# odoo

Warning, your Odoo database manager is not protected. To secure it, we have generated the following master password for it:

**3dng-j3hn-f5t6**

You can change it below but be sure to remember it, it will be asked for future operations on databases.

Master Password

Database Name

Email

Password

Phone number

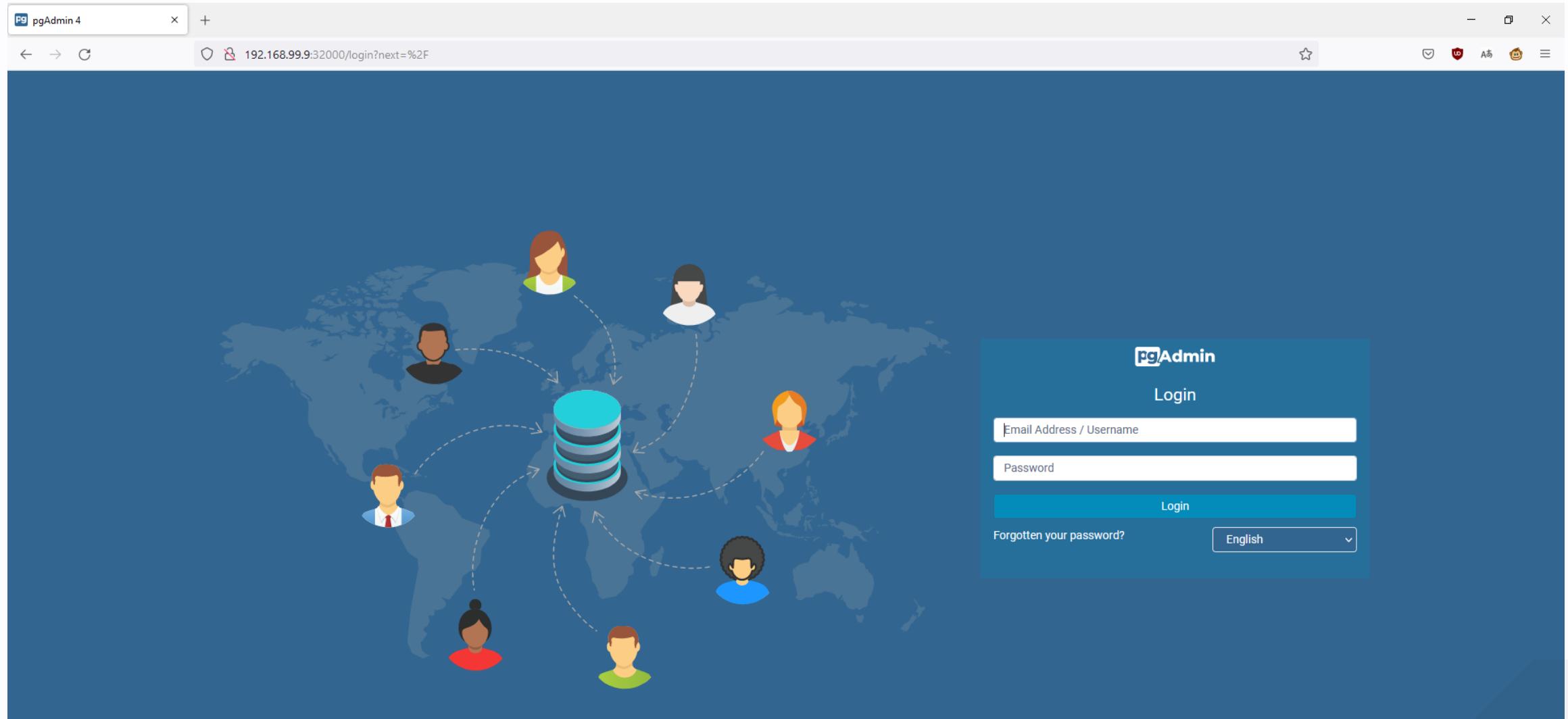
Language

Country

Demo data

**Create database** or restore a database

This screenshot shows the Odoo database selector interface. It features a large 'odoo' logo at the top center. Below the logo is a yellow warning box containing text about an unsecured database manager and a generated master password. The main form below has fields for Master Password, Database Name, Email, Password, Phone number, Language (set to English (US)), and Country. There is also a checkbox for Demo data and buttons for Create database and restore a database. The browser address bar shows the URL 192.168.99.9:31000/web/database/selector.



# Procédure de déploiement

- Kubernetes existant

```
git clone https://github.com/Romain-Revel/ajc-projet-final-2.git  
cd ajc-projet-final-2/manifests
```

Vous pouvez personnaliser l'environnement en modifiant les manifests que ce soit pour les paramètres des conteneurs ou la configuration des services.

Une fois configurée, il suffit de faire :

```
./install_app.sh
```

Ce script demandera de rentrer les mots de passes qui devront être utilisés. Celui de la base de données `postgres` et de l'utilisateur `pgadmin`

- Avec vagrant

```
git clone https://github.com/Romain-Revel/ajc-projet-final-2.git  
cd ajc-projet-final-2/infrastructure/ic-webapp  
vagrant up
```

Cela installera une VM contenant minikube, longhorn et tout l'environnement de production avec comme mot de passe pour la BDD : odoo et pour pgadmin : pgadmin

Si vous avez besoin de changer ces mots de passe, il faut éditer le script `install_app.sh` présent dans le répertoire.

# La validation des fichiers

- hooks de pre-commit

<https://pre-commit.com/>

```
#!/bin/bash

echo "Installation de pre-commit:"
pip install pre-commit

echo "Version de pre-commit:"
pre-commit --version

echo "Activation des hooks de pre-commit:"
pre-commit install

echo "Pour lancer pre-commit sur tous les fichiers:"
echo "pre-commit run --all-files"
echo "Fin du script d'installation de pre-commit"
```

## <https://pre-commit.com/hooks.html>

```
# See https://pre-commit.com for more information
# See https://pre-commit.com/hooks.html for more hooks
repos:
  - repo: https://github.com/pre-commit/pre-commit-hooks
    rev: v4.3.0
    hooks:
      - id: trailing-whitespace
      - id: end-of-file-fixer
      - id: check-yaml
      - id: check-added-large-files
      - id: check-byte-order-marker
      - id: fix-byte-order-marker
      #-
      # id: check-executables-have-shebangs
      # id: check-shebang-scripts-are-executable
      - id: check-symlinks
      - id: destroyed-symlinks
      - id: forbid-new-submodules
      - id: check-json
      - id: check-xml
      - id: mixed-line-ending
        args: [--fix=lf']
      - id: check-case-conflict
      - id: check-merge-conflict
      - id: check-vcs-permalinks
      - id: detect-private-key
      - id: requirements-txt-fixer
```

# • shellcheck pour les scripts d'installation

```
shellcheck $(find . -type f -name "*.sh")
```

```
In ./infrastructure/docker/install_docker.sh line 14:  
if [[ !(-z "$ENABLE_ZSH") && ($ENABLE_ZSH == "true") ]]  
^-- SC1035: You are missing a required space here.
```

```
In ./infrastructure/docker/install_docker.sh line 19:  
su - vagrant -c 'echo "Y" | sh -c "$(curl -fsSL https://raw.githubusercontent.com/ohmyzsh/ohmyzsh/master/tools/install.sh)"'  
^-- SC2016: Expressions don't expand in single quotes, use double quotes for that.
```

```
In ./infrastructure/minikube/install_minikube.sh line 15:  
sudo curl -L0 https://storage.googleapis.com/kubernetes-release/release/`curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt`/bin/linux/amd64/kubectl  
^-- SC2046: Quote this to prevent word splitting.  
^-- SC2006: Use $(..) instead of legacy `..`.
```

```
In ./infrastructure/minikube/install_minikube.sh line 18:  
sudo echo '1' > /proc/sys/net/bridge/bridge-nf-call-iptables  
^-- SC2024: sudo doesn't affect redirects. Use ..| sudo tee file
```

```
In ./infrastructure/minikube/install_minikube.sh line 22:  
echo 'source <(kubectl completion bash)' >> ~vagrant/.bashrc  
^-- SC2129: Consider using { cmd1; cmd2; } >> file instead of individual redirects.
```

```
In ./infrastructure/minikube/install_minikube.sh line 26:  
if [[ !(-z "$ENABLE_ZSH") && ($ENABLE_ZSH == "true") ]]  
^-- SC2039: In POSIX sh, [[ ]] is not supported.  
^-- SC1035: You are missing a required space here.
```

```
In ./infrastructure/minikube/install_minikube.sh line 31:  
su - vagrant -c 'echo "Y" | sh -c "$(curl -fsSL https://raw.githubusercontent.com/ohmyzsh/ohmyzsh/master/tools/install.sh)"'  
^-- SC2016: Expressions don't expand in single quotes, use double quotes for that.
```

- kube-linter pour les manifests

```
docker run --rm -v /home/vagrant/ajc-projet-final/manifests:/dir stackrox/kube-linter lint /dir
KubeLinter 0.5.0-0-g5b90391fdf
kdir/04-BDD_Odoo.yml: (object: icgroup/bdd-odoo apps/v1, Kind=Deployment) container "bdd-odoo" does not have a read-only
root file system (check: no-read-only-root-fs, remediation: Set readOnlyRootFilesystem to true in the container
securityContext.)
kdir/04-BDD_Odoo.yml: (object: icgroup/bdd-odoo apps/v1, Kind=Deployment) container "bdd-odoo" is not set to runAsNonRoot
(check: run-as-non-root, remediation: Set runAsUser to a non-zero number and runAsNonRoot to true in your pod or container
securityContext. Refer to https://kubernetes.io/docs/tasks/configure-pod-container/security-context/ for details.)
kdir/07-Odoo_Web.yml: (object: icgroup/odoo-web apps/v1, Kind=Deployment) object has 2 replicas but does not specify
inter pod anti-affinity (check: no-anti-affinity, remediation: Specify anti-affinity in your pod specification to
ensure that
the orchestrator attempts to schedule replicas on different nodes. Using podAntiAffinity, specify a
labelSelector that
matches pods for the deployment, and set the topologyKey to kubernetes.io/hostname. Refer to https://kubernetes.io/docs/
concepts/scheduling-eviction/assign-pod-node/#inter-pod-affinity-and-anti-affinity for details.)
```

- hadolint pour le Dockerfile

```
docker run --rm -i hadolint/hadolint <ajc-projet-final/ic-webapp/Dockerfile
-:7 DL3013 warning: Pin versions in pip. Instead of `pip install <package>` use `pip install <package>==<version>` or `pip install --requirement <requirements file>`  
-:7 DL3018 warning: Pin versions in apk add. Instead of `apk add <package>` use `apk add <package>=<version>`
```

# Partie 2

Mise en place d'un pipeline CI/CD

# **Ansible**

# Création des Rôles Ansible

Déployer des conteneurs docker avec 2 rôles :

- `odoo_role` : lance 1 conteneur **odoo** et 1 **postgres**
- `pgadmin_role` : lance le site vitrine **ic-webapp** et un conteneur **pgadmin**

**NB** : Toutes les données sont variabilisées donc pourront être surchargée par ansible

**NB** : Il faudrait par la suite passer par un rôle de ansible galaxy  
<https://galaxy.ansible.com/geerlingguy/docker>

# Rôle Odoo

Déploie 2 conteneurs avec le template docker-compose :

- Conteneur odoo
- Conteneur postgres

# templates/docker-compose.yml.j2

```
# Template docker-compose for odoo
version: '3.3'
services:
    {{ SERVICE_POSTGRES }}:
        environment:
            - 'POSTGRES_USER={{ DB_USER }}'
            - 'POSTGRES_PASSWORD={{ DB_PASS }}'
            - 'POSTGRES_DB={{ DB_NAME }}'
        networks:
            - {{ NETWORK_NAME }}
        volumes:
            - 'pgdata:{{ MOUNT_POINT_POSTGRES }}'
        container_name: {{ CONTAINER_NAME_POSTGRES }}
        image: 'postgres:13'
        ports:
            - '{{ POSTGRES_PORT }}:5432'
```

```
{{ SERVICE_ODOO }}:  
  depends_on:  
    - {{ SERVICE_POSTGRES }}  
  ports:  
    - '{{ OD00_PORT }}:8069'  
  container_name: {{ CONTAINER_NAME_ODOO }}  
  networks:  
    - {{ NETWORK_NAME }}  
  volumes:  
    - 'odoo-web-data:/var/lib/odoo'  
  environment:  
    - 'USER={{ DB_USER }}'  
    - 'PASSWORD={{ DB_PASS }}'  
    - 'HOST={{ DB_NAME }}'  
  image: odoo:13  
  
volumes:  
  odoo-web-data:  
  pgdata:  
  
networks:  
  {{ NETWORK_NAME }}:  
    driver: bridge
```

# defaults/main.yml

```
# defaults file for odoo_role

DB_USER: "odoo"
DB_PASS: "odoo"
DB_NAME: "postgres"
POSTGRES_PORT: "5432"
ODOO_PORT: "8081"
IC_PORT: "80"
HOST_IP: "192.168.99.20"
SERVICE_POSTGRES: "postgres"
SERVICE_ODOO: "odoo"
NETWORK_NAME: "ic_network"
CONTAINER_NAME_POSTGRES: "postgres"
CONTAINER_NAME_ODOO: "odoo"
MOUNT_POINT_POSTGRES: "/var/lib/postgresql/data"
```

# tasks/main.yml

```
# tasks file for odoo_role

- name: creation un repertoire files
  file:
    path: "/home/{{ ansible_user }}/files/"
    recurse: yes
    state: directory
- name: generer un fichier docker-compose
  template:
    src: "docker-compose.yml.j2"
    dest: "/home/{{ ansible_user }}/files/docker-compose.yml"
- name: "Deploiemment"
  command: "docker-compose up -d"
  args:
    chdir: "/home/{{ ansible_user }}/files"
```

# Rôle PgAdmin

Déploie deux conteneurs via les templates docker-compose et servers :

- pgadmin
- ic-webapp

## templates/docker-compose.yml.j2

```
# Template docker-compose for pgadmin and ic-webapp
version: '3.3'
services:
  {{ SERVICE_PGADMIN }}:
    container_name: {{ CONTAINER_NAME_PGADMIN }}
    image: dpage/pgadmin4
    networks:
      - {{ NETWORK_NAME }}
    environment:
      - 'PGADMIN_DEFAULT_EMAIL={{ PGADMIN_EMAIL }}'
      - 'PGADMIN_DEFAULT_PASSWORD={{ PGADMIN_PASS }}'
    ports:
      - "{{ PGADMIN_PORT }}:80"
    volumes:
      - /home/{{ ansible_user }}/files/servers.json:/pgadmin4/servers.json
      - 'pgadmin_data:/var/lib/pgadmin'
  {{ SERVICE_ICWEBAPP }}:
    container_name: {{ CONTAINER_NAME_ICWEBAPP }}
    ports:
      - "{{ IC_PORT }}:8080"
    environment:
      - "ODOO_URL=http://{{ POSTGRES_IP }}:{{ ODOO_PORT }}/"
      - "PGADMIN_URL=http://{{ HOST_IP }}:{{ PGADMIN_PORT }}/"
    image: '{{ IMAGE_NAME }}:{{ IMAGE_TAG }}'
    networks:
      - {{ NETWORK_NAME }}
volumes:
  pgadmin_data:
networks:
  {{ NETWORK_NAME }}:
    driver: bridge
```

# servers.json.j2

```
{  
  "Servers": {  
    "1": {  
      "Name": "{{ DB_NAME }}",  
      "Group": "docker_postgres_group_1",  
      "Port": {{ POSTGRES_PORT }},  
      "Username": "{{ DB_USER }}",  
      "Host": "{{ POSTGRES_IP }}",  
      "SSLMode": "prefer",  
      "MaintenanceDB": "{{ DB_NAME }}"  
    }  
  }  
}
```

**ATTENTION : Le port de Postgre DOIT être numérique**

# defaults/main.yml

```
# defaults file for pgadmin_role
PGADMIN_EMAIL: "user@domain.com"
PGADMIN_PASS: "odoo_pgadmin_password"
PGADMIN_PORT: "8082"
DB_USER: "odoo"
DB_PASS: "odoo"
DB_NAME: "postgres"
POSTGRES_PORT: "5432"
ODOO_PORT: "8081"
IC_PORT: "80"
HOST_IP: "192.168.99.21"
POSTGRES_IP: "192.168.99.20"
IMAGE_NAME: "sh0t1m3/ic-webapp"
IMAGE_TAG: "1.0"
SERVICE_PGADMIN: "pgadmin"
SERVICE_ICWEBAPP: "ic-webapp"
NETWORK_NAME: "ic_network"
CONTAINER_NAME_PGADMIN: "pgadmin"
CONTAINER_NAME_ICWEBAPP: "ic-webapp"
```

# tasks/main.yml

```
- name: creation repertoire files
  file:
    path: "/home/{{ ansible_user }}/files/"
    recurse: yes
    state: directory
- name: générer docker-compose
  template:
    src: "docker-compose.yml.j2"
    dest: "/home/{{ ansible_user }}/files/docker-compose.yml"
- name: pgadmin config file servers
  template:
    src: "servers.json.j2"
    dest: "/home/{{ ansible_user }}/files/servers.json"
- name: "Deploiemt"
  command: "docker-compose up -d"
  args:
    chdir: "/home/{{ ansible_user }}/files"
```

# Le playbook Ansible

Un playbook Ansible est un modèle de tâches d'automatisation. Les playbooks Ansible sont exécutés sur un **ensemble**, un **groupe** ou une **classification** d'hôtes, qui forment ensemble un inventaire.

Source:

<https://www.redhat.com/fr/topics/automation/what-is-an-ansible-playbook>

# play.yml

```
# Notre playbook
- name: "installation de odoo"
  hosts: prod-odoo
  roles:
    - role: odoo_role
- name: "Installation de pgadmin"
  hosts: prod-pgadmin
  roles:
    - role: pgadmin_role
```

## prods.yml

```
# L'inventaire
all:
  children:
    prod-odoo:
      hosts:
        docker-odoo:
    prod-pgadmin:
      hosts:
        docker-pgadmin-icwebapp:
```

# La structure de notre répertoire Ansible

```
└── group_vars
    ├── prod-odoo.yml
    └── prod-pgadmin.yml
└── host_vars
    ├── docker-odoo.yml
    └── docker-pgadmin-icwebapp.yml
├── play.yml
└── prods.yml
└── roles
    ├── odoo_role
    └── pgadmin_role
```

```
└── roles
    ├── odoo_role
    │   ├── defaults
    │   │   └── main.yml
    │   ├── handlers
    │   │   └── main.yml
    │   ├── meta
    │   │   └── main.yml
    │   ├── README.md
    │   ├── tasks
    │   │   └── main.yml
    │   ├── templates
    │   │   └── docker-compose.yml.j2
    │   ├── tests
    │   │   └── inventory
    │   │       └── test.yml
    │   └── vars
    │       └── main.yml
```

Tableau de bord > essai3 > #5

```
[Pipeline] ansiblePlaybook
[essai3] $ ansible-playbook ansible/play.yml -i ansible/prods.yml --extra-vars "NETWORK_NAME=network
IMAGE_TAG=1.2
ansible_user=**** ansible_ssh_pass=****
ansible_sudo_pass=**** PGADMIN_EMAIL=pgadmin_user@pgadmin.net
PGADMIN_PASS=**** DB_USER=pgsql_user DB_PASS=****"
[WARNING]: Invalid characters were found in group names but not replaced, use
-vvvv to see details

PLAY [installation de odoo] ****
TASK [Gathering Facts] ****
ok: [docker-odoo]

TASK [odoo_role : creation un repertoire files] ****
ok: [docker-odoo]

TASK [odoo_role : generer un fichier docker-compose] ****
ok: [docker-odoo]

TASK [odoo_role : Deploiement] ****
changed: [docker-odoo]

PLAY [Installation de pgadmin] ****
TASK [Gathering Facts] ****
ok: [docker-pgadmin-icwebapp]
```



Tableau de bord → essai3 → #5

```
TASK [odoo_role : Deploiement] ****
changed: [docker-odoo]

PLAY [Installation de pgadmin] ****

TASK [Gathering Facts] ****
ok: [docker-pgadmin-icwebapp]

TASK [pgadmin_role : creation repertoire files] ****
ok: [docker-pgadmin-icwebapp]

TASK [pgadmin_role : g?n?rer docker-compose] ****
ok: [docker-pgadmin-icwebapp]

TASK [pgadmin_role : pgadmin config file servers] ****
changed: [docker-pgadmin-icwebapp]

TASK [pgadmin_role : Deploiement] ****
changed: [docker-pgadmin-icwebapp]

PLAY RECAP ****
docker-odoo          : ok=4    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
docker-pgadmin-icwebapp : ok=5    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

# Jenkins

# Infrastructure

- Serveur 1 : **192.168.99.12** Jenkins  
<https://github.com/sadofrazer/jenkins-frazer.git>
- Serveur 2 : **192.168.99.21** Applications web site  
vitrine + pgadmin4
- Serveur 3 : **192.168.99.20** Application Odoo  
(PostgreSQL)

# Serveurs 2 et 3 - boucle

```
Vagrant.configure("2") do |config|
  array = ["odoo", "pgadmin-icwebapp"]
  # Boucle pour créer les 2
  array.each_with_index do |val, index|
    config.vm.define "docker-#{val}" do |docker|
      #...
      docker.vm.network "private_network", type: "static", ip: "192.168.99.2#{index}", name: "vboxnet0"
      #...
      docker.vm.network "private_network", type: "static", ip: "192.168.99.2#{index}"
      #...
      docker.vm.hostname = "docker-#{val}"
      #...
      v.name = "docker-#{val}"
      #...
```

# Modification du fichier releases.txt

Il est demandé de pouvoir reconstruire l'image de l'application web vitrine lors de la modification du fichier releases.txt

ODOO\_URL: http://192.168.99.20:8081

PGADMIN\_URL: http://192.168.99.21:8082

version: 1.2

Ce fichier alimentera le script de lancement de l'application web vitrine.

(Il faudrait le tester dans la pipeline pour vérifier que les paires champs:valeur soient cohérentes)

# Modification du dockerfile

```
FROM python:3.6-alpine

ENV ODOO_URL=""
ENV PGADMIN_URL=""

# Install python and pip
RUN apk add --no-cache --update bash && \
    # Install dependencies
    pip3 install Flask && \
    # Add a Group and user icwebapp
    addgroup -S icwebapp && \
    adduser -S icwebapp -G icwebapp

# Add our code
COPY --chown=icwebapp:icwebapp ic-webapp /opt/ic-webapp/
COPY --chown=icwebapp:icwebapp releases.txt /opt/ic-webapp/releases.txt

USER icwebapp
# Create the entry point script
RUN echo -e "#!/bin/bash\nexport ODOO_URL=$(cat /opt/ic-webapp/releases.txt | \
grep ODOO_URL | sed -e 's/^ODOO_URL: \(.*\)$/\1/') && export PGADMIN_URL=$(cat /opt/ic-webapp/releases.txt | grep PGADMIN_URL | \
sed -e 's/^PGADMIN_URL: \(.*\)$/\1/') && python3 app.py" > /opt/ic-webapp/start.sh \
&& chmod +x /opt/ic-webapp/start.sh

WORKDIR /opt/ic-webapp
EXPOSE 8080

ENTRYPOINT [ "./start.sh" ]
```

# Jenkins - Script d'installation

```
#!/bin/bash
# Script d'installation de Jenkins fourni par Dirane
yum -y update
yum -y install epel-release
# install ansible
yum -y install ansible
# retrieve ansible code
yum -y install git
git clone https://github.com/diranetafen/cursus-devops.git
cd cursus-devops/ansible
ansible-galaxy install -r roles/requirements.yml
ansible-playbook install_docker.yml
sudo usermod -aG docker vagrant
cd ../jenkins
/usr/local/bin/docker-compose up -d
echo "For this Stack, you will use $(ip -f inet addr show enp0s8 | \
sed -En -e 's/.inet ([0-9.]+).*/\1/p') IP Address"
```

# Interface

Tableau de bord [Jenkins] — Mozilla Firefox

Tableau de bord [Jenkins] +  
192.168.99.12:8080

Jenkins

rechercher Romain REVEL se déconnecter

Nouveau item Utilisateurs Historique des constructions Administrer Jenkins Mes vues Crée une Vue

Ajouter une description

Bienvenue sur Jenkins !

Vos jobs Jenkins seront affichés sur cette page. Pour commencer, vous pouvez mettre en place un build distribué ou commencer à créer un projet.

Commencer à créer votre projet

Créer un job →

Configurer un build distribué

Mettre en place un agent →

Configurer un cloud →

En apprendre plus sur les builds distribués ↗

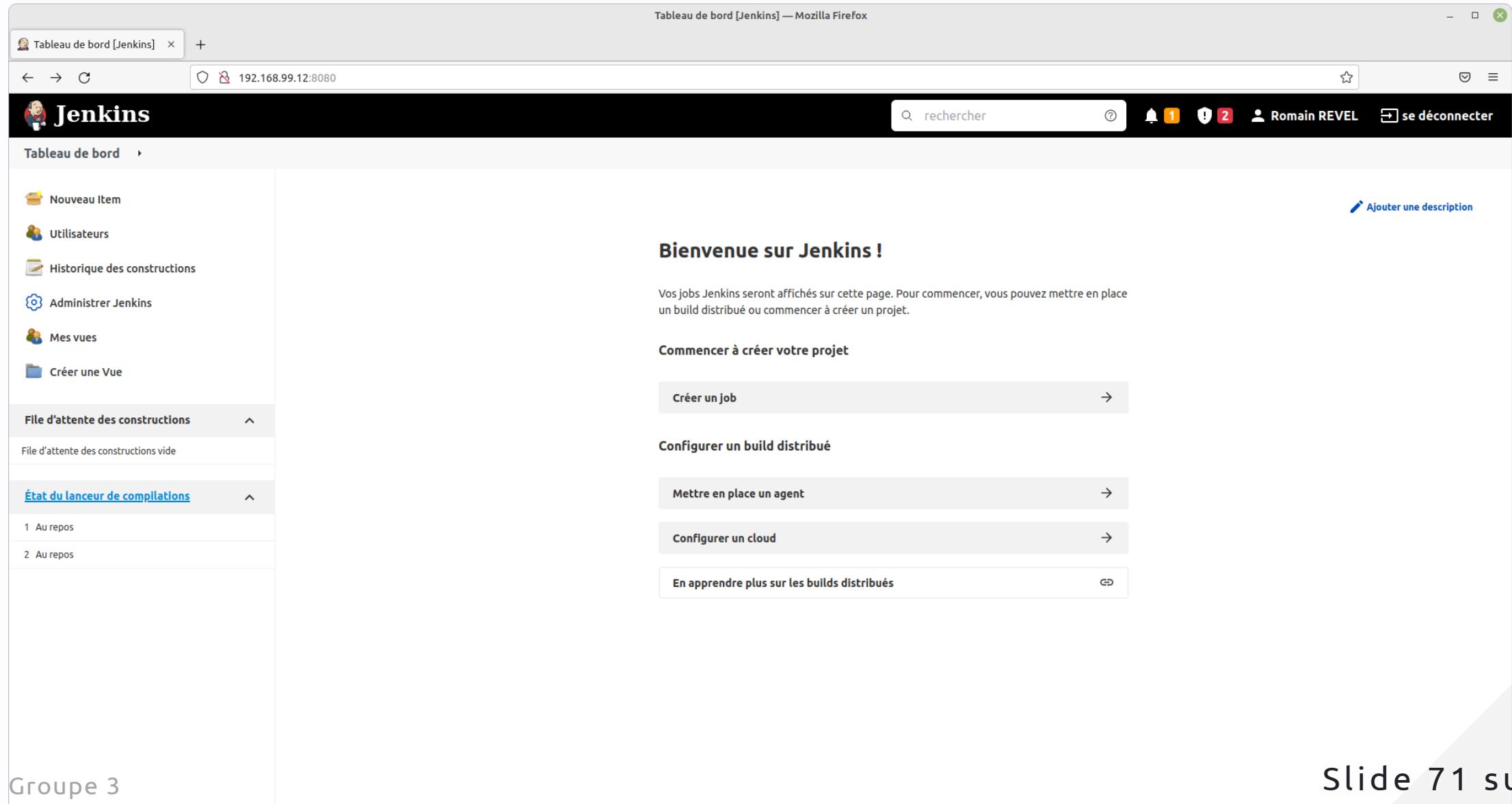
File d'attente des constructions ^  
File d'attente des constructions vide

État du lanceur de compilations ^  
1 Au repos  
2 Au repos

Groupe 3

Slide 71 sur 115

REST API Jenkins 2.339



# Jenkins - Inconvénients

- Version datée
- Interface web **laide et non ergonomique**
- Les installations ne partagent pas:
  - Comptes
  - Plugins
  - Jobs
  - Secrets
  - Configuration globale

# Automatisation de l'installation

jenkins-custom :

- Docker
- jenkins/jenkins:lts-jdk11

```
FROM jenkins/jenkins:lts-jdk11
USER root
ENV JAVA_OPTS -Djenkins.install.runSetupWizard=false
ENV CASC_JENKINS_CONFIG /var/jenkins_home/jenkins.casc.yml
# Installation
RUN apt-get update && \
    apt-get install -qy curl python3 python3-pip sshpass shellcheck && \
    pip3 install ansible && \
    curl -sSL https://get.docker.com/ | sh
USER jenkins
# Plugins Jenkins
COPY jenkins.plugins.txt /usr/share/jenkins/ref/jenkins.plugins.txt
RUN jenkins-plugin-cli --list && \
    jenkins-plugin-cli --plugin-file /usr/share/jenkins/ref/jenkins.plugins.txt && \
    jenkins-plugin-cli --list
# Configuration as code Jenkins
COPY jenkins.casc.yml /var/jenkins_home/jenkins.casc.yml
```

# Plugins par défaut

```
antisamy-markup-formatter:latest  
build-timeout:latest  
cloudbees-folder:latest  
credentials-binding:latest  
email-ext:latest  
git:latest  
github-branch-source:latest  
mailer:latest  
pam-auth:latest  
pipeline-github-lib:latest  
pipeline-stage-view:latest  
ssh-slaves:latest  
timestamper:latest  
workflow-aggregator:latest  
ws-cleanup:latest
```

# Plugins supplémentaires

```
ansible:latest  
authorize-project:latest  
configuration-as-code:latest  
docker-plugin:latest  
docker-workflow:latest  
matrix-auth:latest
```

Tableau de bord &gt;

+ Nouveau item

[Ajouter une description](#) Utilisateurs

Tous

+

 Historique des constructions Relations entre les builds Vérifier les empreintes numériques Administrer Jenkins Mes vues

S	M	Nom du projet ↓	Dernier succès	Dernier échec	Dernière durée
		ic-webapp-pipeline	19 h #3	18 h #7	38 s
		ic-webapp-pipeline-manuel	S. O.	S. O.	ND

Icône: S M L

Légende

 Atom feed pour tout Atom feed de tous les échecs Atom feed juste pour les dernières compilations**File d'attente des constructions**

File d'attente des constructions vide

**État du lanceur de compilations**

1 Au repos

2 Au repos

# jenkins-cli

- <https://www.jenkins.io/doc/book/managing/cli/>
- <https://medium.com/@muku.hbt/export-import-jenkins-job-and-their-plugins-53cafa5869fa>
- <https://www.digitalocean.com/community/tutorials/how-to-automate-jenkins-setup-with-docker-and-jenkins-configuration-as-code>

```
#/bin/bash
# Téléchargement de jenkins-cli
if [[ -z "${JENKINS_USERNAME}" ]]; then
    echo "La variable JENKINS_USERNAME n'existe pas. Veuillez l'exporter.";
    exit 1;
fi
if [[ -z "${JENKINS_PASSWORD}" ]]; then
    echo "La variable JENKINS_PASSWORD n'existe pas. Veuillez l'exporter.";
    exit 1;
fi
# Valeurs par défaut
[[ ! -z "${JENKINS_IP}" ]] || JENKINS_IP="192.168.99.13";
[[ ! -z "${JENKINS_PORT}" ]] || JENKINS_PORT="8080";
URL="http://${JENKINS_IP}:${JENKINS_PORT}";
JOB_NAME="TEST";
# Téléchargement de la cli jenkins
if [[ ! -f "jenkins-cli.jar" ]]; then
    wget "${URL}/jnlpJars/jenkins-cli.jar";
fi
```

```

# Pour lister les jobs
java -jar jenkins-cli.jar -s "${URL}" -auth "${JENKINS_USERNAME}":"${JENKINS_PASSWORD}" list-jobs;

# Récupérer la liste de tous les jobs et les exporter
JOBS=$(java -jar jenkins-cli.jar -s "${URL}" -auth "${JENKINS_USERNAME}":"${JENKINS_PASSWORD}" list-jobs);
mkdir -p "jobs";
for JOB_NAME in $JOBS
do
    java -jar jenkins-cli.jar -s "${URL}" -auth "${JENKINS_USERNAME}":"${JENKINS_PASSWORD}" get-job "${JOB_NAME}" > "jobs/${JOB_NAME}.xml";
done

# Restaurer tous les jobs
for JOB_FILE in $(cd "jobs"; ls *.xml)
do
    # echo $JOB_FILE
    # echo ${JOB_FILE%.xml}
    java -jar jenkins-cli.jar -s "${URL}" -auth "${JENKINS_USERNAME}":"${JENKINS_PASSWORD}" create-job "${JOB_FILE%.xml}" < "jobs/${JOB_FILE}"
done

```

# Configuration de Jenkins

Plugin Configuration as code

- Compte(s) admin
- URL
- Credentials
- Security (bonus)

# Compte admin

```
jenkins:  
  securityRealm:  
    local:  
      allowsSignup: false  
      enableCaptcha: false  
    users:  
      - id: admin  
        password: password  
        properties:  
          - "apiToken"  
          - mailer:  
              emailAddress: "admin@hotmail.fr"  
          - preferredProvider:  
              providerId: "default"
```

# URL

unclassified:

location:

url: <http://192.168.99.13:8080/>

# Credentials

```
credentials:
  system:
    domainCredentials:
      - credentials:
          - string:
              description: "Token dockerhub"
              id: "dockerhub"
              scope: GLOBAL
              secret: "{AQAAABAAAAAw632BD8V0u3j00s90yYiMwllBr60zIrtmGMqWAEvtIDcqXa2XCyH2WBJPrmSdH9fPnShuX2v4AMjjUbicqwo2Ag==}"
            - usernamePassword:
                id: "ansible_user_credentials"
                password: "vagrant"
                scope: GLOBAL
                username: "vagrant"
                usernameSecret: true
            - usernamePassword:
                id: "pgadmin_credentials"
                password: "pgadmin"
                scope: GLOBAL
                username: "pgadmin@local.domain" // @ Très important
                usernameSecret: true
```

# Sécurité

```
jenkins:  
  //...  
  authorizationStrategy:  
    globalMatrix:  
      permissions:  
        - "USER:Overall/Administer:admin"  
        - "GROUP:Overall/Read:authenticated"  
  remotingSecurity:  
    enabled: true  
  
security:  
  queueItemAuthenticator:  
    authenticators:  
      - global:  
          strategy: triggeringUsersAuthorizationStrategy
```

# Rendre Jenkins accessible avec Ngrok

Ngrok est un reverse-proxy qui permet d'ouvrir sur internet des ports d'une machine

Utilisé pour la partie **webhook**

Alternatives à ngrok:

- Vagrant share

<https://www.vagrantup.com/docs/share>

- Localtunnel

<https://github.com/localtunnel/localtunnel>

```
#!/bin/bash
# Script d'installation et de lancement de ngrok
FILE_NAME="ngrok-v3-stable-linux-amd64.tgz";
if [[ ! -f "/usr/local/bin/ngrok" ]]; then
    # Téléchargement
    # https://ngrok.com/download
    if [[ ! -f "${FILE_NAME}" ]]; then
        wget "https://bin.equinox.io/c/bNyj1mQVY4c/${FILE_NAME}";
    fi

    # Décompression et installation
    sudo tar xvzf "${FILE_NAME}" -C /usr/local/bin;
fi
sleep 3
# Enregistrement
ngrok config add-authtoken $(cat token.txt);
# Lancement
nohup ngrok http 8080 &
# Récupération de l'URL
curl "http://localhost:4040/api/tunnels";
```



You are about to visit:  
**3b40-176-130-113-152.eu.ngrok.io**

Website IP: 176.130.113.152

- This website is served for free through [ngrok.com](#).
- You should only visit this website if you trust whoever sent the link to you.
- Be careful about disclosing personal or financial information like passwords, phone numbers, or credit cards.

[Visit Site](#)

**Are you the developer?**

We display this page to prevent abuse. Visitors to your site will only see it once.

**To remove this page:**

- Set and send an `ngrok-skip-browser-warning` request header with any value.
- Or, set and send a custom/non-standard browser `User-Agent` request header.
- Or, please [upgrade](#) to any paid ngrok account.

**ngrok** Learn how ngrok [fights abuse](#)

S'identifier [Jenkins]



← → ↻

https://3b40-176-130-113-152.eu.ngrok.io/login?from=%2F



≡

**Bienvenue dans Jenkins !**

utilisateur

Mot de passe

 Garder ma session ouverte**S'identifier**

# Pipeline(s)

# Structure

```
// Jenkinsfile
pipeline {

    environment {
        IMAGE_NAME = "ic-webapp"
        IMAGE_TAG = "${sh(returnStdout: true, script: 'cat ic-webapp/releases.txt \
|grep version | cut -d\\: -f2|xargs')}"
        CONTAINER_NAME = "ic-webapp"
        USER_NAME = "sh0t1m3"
    }

    agent any

    stages {
        // stage 1...
    }
}
```

# Lint YAML

```
stage('Lint yaml files') {
    when { changeset "**/*.yml"}
    agent {
        docker {
            image 'sdesbure/yamllint'
        }
    }
    steps {
        sh 'yamllint --version'
        sh 'yamllint ${WORKSPACE} >report_yml.log || true'
    }
    post {
        always {
            archiveArtifacts 'report_yml.log'
        }
    }
}
```

# Lint markdown

```
stage('Lint markdown files') {
    when { changeset "**/*.md" }
    agent {
        docker {
            image 'ruby:alpine'
        }
    }
    steps {
        sh 'apk --no-cache add git'
        sh 'gem install mdl'
        sh 'mdl --version'
        sh 'mdl --style all --warnings --git-recurse ${WORKSPACE} > md_lint.log || true'
    }
    post {
        always {
            archiveArtifacts 'md_lint.log'
        }
    }
}
```

# Lint ansible

```
stage("Lint ansible playbook files") {
    when { changeset "ansible/**/*.yml"}
    agent {
        docker {
            image 'registry.gitlab.com/robconnolly/docker-ansible:latest'
        }
    }
    steps {
        sh '''
            cd "${WORKSPACE}/ansible/"
            ansible-lint play.yml > "${WORKSPACE}/ansible-lint.log" || true
        '''
    }
    post {
        always {
            archiveArtifacts "ansible-lint.log"
        }
    }
}
```

# Lint shell scripts

```
stage('Lint shell script files') {
    when { changeset "**/*.sh" }
    agent any
    steps {
        sh 'shellcheck */*.sh >shellcheck.log || true'
    }
    post {
        always {
            archiveArtifacts 'shellcheck.log'
        }
    }
}
```

# Lint shell scripts - checkstyle

```
stage('Lint shell script files - checkstyle') {
    when { changeset "**/*.sh" }
    agent any
    steps {
        catchError(buildResult: 'SUCCESS') {
            sh """#!/bin/bash
                # The null command `:` only returns exit code 0 to ensure following task executions.
                shellcheck -f checkstyle */*.sh > shellcheck.xml || true
            """
        }
    }
    post {
        always {
            archiveArtifacts 'shellcheck.xml'
        }
    }
}
```

# Lint docker files

```
stage ("Lint docker files") {
    when { changeset "**/Dockerfile" }
    agent {
        docker {
            image 'hadolint/hadolint:latest-debian'
        }
    }
    steps {
        sh 'hadolint $PWD/**/Dockerfile | tee -a hadolint_lint.log'
    }
    post {
        always {
            archiveArtifacts 'hadolint_lint.log'
        }
    }
}
```

# Push docker image

```
stage ('Login and push docker image') {
    when { changeset "ic-webapp/releases.txt"}
    agent any
    environment {
        DOCKERHUB_PASSWORD = credentials('dockerhub')
    }
    steps {
        script {
            sh '''
                echo "${DOCKERHUB_PASSWORD}" | docker login -u ${USER_NAME} --password-stdin;
                docker push ${USER_NAME}/${IMAGE_NAME}:${IMAGE_TAG};
            '''
        }
    }
}
```



## Tableau de bord &gt; AJC-PROJET-FINAL &gt;

Full Stage View

Renommer

Pipeline Syntax

 Historique des builds tendance ^

Filter builds...

#14 10 oct. 2022 13:18

#13 10 oct. 2022 13:10

#12 10 oct. 2022 13:08

#11 10 oct. 2022 13:03

#10 10 oct. 2022 12:56

#9 10 oct. 2022 12:54

#8 10 oct. 2022 12:51

#7 10 oct. 2022 12:49

#6 10 oct. 2022 12:46

#5 10 oct. 2022 12:41

#4 10 oct. 2022 07:49

#3 10 oct. 2022 07:44

#2 10 oct. 2022 07:43

#1 10 oct. 2022 07:43

Atom feed des builds

groupe 3

Atom feed des échecs

## Stage View

Average stage times:  
(Average full run time: ~1min

14s)

#14 Oct 10 13:18

1 commit

14s

Declarative: Checkout SCM	Lint yaml files	Lint markdown files	Lint ansible playbook files	Lint shell script files	Lint shell script files - checkstyle	Lint docker files	Build docker image	Test docker image	Login and push docker image
1s	4s	8s	6s	30s	1s	3s	624ms	15s	6s
#14 Oct 10 13:18	1s	5s	9s	6s	28s	1s	3s		
#13 Oct 10 13:10	1s	4s	8s	6s	30s	1s	3s	630ms	15s
#12 Oct 10 13:08	910ms	4s	8s	6s	28s	953ms	3s	619ms	5s
#11 Oct 10 13:03	2s	4s	8s	6s	33s	1s	3s	623ms	13s

## Liens permanents

- Dernier build (#7), il y a 49 s
- Dernier build stable (#4), il y a 5 h 1 mn

## Tableau de bord &gt; AJC-PROJET-FINAL &gt;

Full Stage View

Renommer

Pipeline Syntax

Historique des builds tendance ^

Filter builds...

## Stage View

Average stage times:  
(Average full run time: ~1min  
15s)

	Declarative: Checkout SCM	Lint yaml files	Lint markdown files	Lint ansible playbook files	Lint shell script files	Lint shell script files - checkstyle	Lint docker files	Build docker image	Test docker image	Login and push docker image
#17	2s	4s	8s	6s	29s	1s	3s	950ms	15s	7s
#16	Oct 10 13:26	981ms	4s	8s	6s	29s	3s	1s	15s	12s
#15	Oct 10 13:21	5s	6s	9s	6s	26s	986ms	3s		
#14	Oct 10 13:18	953ms	4s	8s	6s	27s	969ms	3s	1s	15s
#13	Oct 10 13:10	1s	5s	9s	6s	28s	1s	3s		
#12	Oct 10 13:08	1s	4s	8s	6s	30s	1s	3s	630ms	15s
#11	Oct 10 13:03	1s	4s	8s	6s	30s	1s	3s	630ms	6s
#10	Oct 10 12:56	1s	5s	9s	6s	30s	1s	3s	630ms	6s
#9	Oct 10 12:54	1s	5s	9s	6s	30s	1s	3s	630ms	6s
#8	Oct 10 12:51	1s	5s	9s	6s	30s	1s	3s	630ms	6s
#7	Oct 10 12:49	1s	5s	9s	6s	30s	1s	3s	630ms	6s
#6	Oct 10 12:46	1s	4s	8s	6s	30s	1s	3s	630ms	6s
#5	Oct 10 12:41	1s	4s	8s	6s	30s	1s	3s	630ms	6s
#4	Oct 10 07:49	1s	4s	8s	6s	30s	1s	3s	630ms	15s
#3	Oct 10 07:44	1s	4s	8s	6s	30s	1s	3s	630ms	15s
#2	Groupe 3	1s	4s	8s	6s	30s	1s	3s	630ms	6s
	Oct 10 07:43	1s	4s	8s	6s	30s	1s	3s	630ms	6s

Docker Hub — Mozilla Firefox

AJC-PROJET-FINAL [Jenkins] Docker Hub

https://hub.docker.com/repository/docker/sh0t1m3/ic-webapp

docker hub Search for great content (e.g., mysql) Explore Repositories Organizations Help Upgrade sh0t1m3

sh0t1m3 Repositories ic-webapp Using 1 of 1 private repositories. Get more

General Tags Builds Collaborators Webhooks Settings

**i** Add a short description for this repository  
The short description is used to index your content on Docker Hub and in search engines. It's visible to users in search results. [Update](#)

**sh0t1m3 / ic-webapp**

Description  
*This repository does not have a description* [Edit](#)

Docker commands  
To push a new tag to this repository,  
`docker push sh0t1m3/ic-webapp:tagname` [Public View](#)

Last pushed: 9 minutes ago

Tags and scans  VULNERABILITY SCANNING - DISABLED [Enable](#)

This repository contains 1 tag(s).

Tag	OS	Pulled	Pushed
1.0		--	9 minutes ago

[See all](#) [Go to Advanced Image Management](#)

Automated Builds  
Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.  
Available with Pro, Team and Business subscriptions.

[Upgrade](#) [Learn more](#)

README   
Repository description is empty. Click [here](#) to edit.

Groupe 3 Slide 101 sur 115

Docker Hub — Mozilla Firefox

AJC-PROJET-FINAL [Jenkins] Docker Hub

https://hub.docker.com/repository/docker/sh0t1m3/ic-webapp

docker hub Search for great content (e.g., mysql) Explore Repositories Organizations Help Upgrade sh0t1m3

sh0t1m3 Repositories ic-webapp Using 1 of 1 private repositories. Get more

General Tags Builds Collaborators Webhooks Settings

**i** Add a short description for this repository  
The short description is used to index your content on Docker Hub and in search engines. It's visible to users in search results. [Update](#)

**sh0t1m3 / ic-webapp**

Description  
*This repository does not have a description* [Edit](#)

Docker commands  
To push a new tag to this repository,  
`docker push sh0t1m3/ic-webapp:tagname` [Public View](#)

Last pushed: a few seconds ago

Tags and scans  VULNERABILITY SCANNING - DISABLED [Enable](#)

This repository contains 2 tag(s).

Tag	OS	Pulled	Pushed
1.2		--	a few seconds ago
1.0		--	8 minutes ago

[See all](#) [Go to Advanced Image Management](#)

Automated Builds  
Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.  
Available with Pro, Team and Business subscriptions.

[Upgrade](#) [Learn more](#)

README   
Repository description is empty. Click [here](#) to edit.

Groupe 3 Slide 102 sur 115

# Déploiement avec Ansible

Credentials à déclarer dans Jenkins pour déployer par Ansible

## Credentials

T	P	Store ↓	Domain	ID	Name
		System	(global)	dockerhub	<a href="#">Token dockerhub</a>
		System	(global)	ansible_user_credentials	<a href="#">ansible_user_credentials</a>
		System	(global)	pgadmin_credentials	<a href="#">pgadmin_credentials</a>
		System	(global)	pgsql_credentials	<a href="#">pgsql/*****</a>
		System	(global)	ansible_sudo_pass	<a href="#">ansible_sudo_pass</a>

## Stores scoped to Jenkins

P	Store ↓	Domains
	System	(global)

Icône:   

```

stage ('Deploy to prod with Ansible') {
    steps {
        withCredentials([
            usernamePassword(credentialsId: 'ansible_user_credentials', usernameVariable: 'ansible_user', passwordVariable: 'ansible_user_pass'),
            usernamePassword(credentialsId: 'pgadmin_credentials', usernameVariable: 'pgadmin_user', passwordVariable: 'pgadmin_pass'),
            usernamePassword(credentialsId: 'pgsql_credentials', usernameVariable: 'pgsql_user', passwordVariable: 'pgsql_pass'),
            string(credentialsId: 'ansible_sudo_pass', variable: 'ansible_sudo_pass')
        ])
        {
            ansiblePlaybook (
                disableHostKeyChecking: true,
                installation: 'ansible',
                inventory: 'ansible/prods.yml',
                playbook: 'ansible/play.yml',
                extras: '--extra-vars "NETWORK_NAME=network \
                    IMAGE_TAG=${IMAGE_TAG} \
                    ansible_user=${ansible_user} \
                    ansible_ssh_pass=${ansible_user_pass} \
                    ansible_sudo_pass=${ansible_sudo_pass} \
                    PGADMIN_EMAIL=${pgadmin_user} \
                    PGADMIN_PASS=${pgadmin_pass} \
                    DB_USER=${pgsql_user} \
                    DB_PASS=${pgsql_pass}"'
            )
        }
    }
}

```

# Test final

```
stage ('Test full deployment') {
    steps {
        sh '''
            sleep 10;

            curl -LI http://192.168.99.21 | grep "200";
            curl -L http://192.168.99.21 | grep "IC GROUP";

            curl -LI http://192.168.99.20:8081 | grep "200";
            curl -L http://192.168.99.20:8081 | grep "Odoo";

            curl -LI http://192.168.99.21:8082 | grep "200";
            curl -L http://192.168.99.21:8082 | grep "pgAdmin 4";
        '''

    }
}
```

# Troubleshooting

- Attention aux redirections HTTP
- Attention au délai entre la fin d'Ansible et la disponibilité du site web
- Attention si on change le login et mot de passe de la base de données il faut supprimer le volume pgadmin avant de relancer le docker-compose up -d, bien entendu cette action supprime les données donc il faut éviter ce changement.

# Trivy

Scanner de vulnérabilité

- code source (fs, repo)
- dépendances
- images conteneur
- ...

<https://semaphoreci.com/blog/continuous-container-vulnerability-testing-with-trivy>

```
stage("Trivy scan") {
    agent any
    steps {
        sh 'rm -Rf ./trivy || true'
        sh 'curl -sfL https://raw.githubusercontent.com/aquasecurity/trivy/main/contrib/install.sh | sh -s -- -b . v0.18.3'
        sh './trivy --exit-code 0 image -o trivy-icwebapp.log ${USER_NAME}/${IMAGE_NAME}:${IMAGE_TAG}'
        sh './trivy --exit-code 1 image --severity=CRITICAL -o trivy-icwebapp_CRITICAL.log ${USER_NAME}/${IMAGE_NAME}:${IMAGE_TAG}'
        sh './trivy --exit-code 0 image --severity=CRITICAL -o trivy-odoo13.log odoo:13'
        sh './trivy --exit-code 0 image --severity=CRITICAL -o trivy-postgres13.log postgres:13'
        sh './trivy --exit-code 0 image --severity=CRITICAL -o trivy-pgadmin4.log dpage/pgadmin4'
    }
    post {
        always {
            archiveArtifacts "trivy-icwebapp.log"
            archiveArtifacts "trivy-icwebapp_CRITICAL.log"
            archiveArtifacts "trivy-odoo13.log"
            archiveArtifacts "trivy-postgres13.log"
            archiveArtifacts "trivy-pgadmin4.log"
        }
    }
}
```

# Déclenchement automatique

- Webhook Github
- Tâche planifiée qui vérifie régulièrement les modifications du SCM
- Tâche planifiée qui build régulièrement (MAUVAIS)

General

Access

Collaborators

Moderation options

Code and automation

Branches

Tags

Actions

Webhooks

Environments

Pages

Security

Code security and analysis

Deploy keys

Secrets

Integrations

GitHub apps

Email notifications

Webhooks / Add webhook

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information can be found in [our developer documentation](#).

Payload URL \*

`https://9513-176-130-113-152.eu.ngrok.io/github-webhook/`

Content type

application/json

Secret

(empty)

SSL verification

By default, we verify SSL certificates when delivering payloads.

Enable SSL verification  Disable (not recommended)

Which events would you like to trigger this webhook?

Just the push event.

Send me everything.

Let me select individual events.

Active

We will deliver event details when this hook is triggered.

Add webhook

# Problèmes rencontrés avec Jenkins

- Version pas à jour
- Agent docker ne fonctionne pas pour (shellcheck, trivy)
- Jenkins difficile à configurer automatiquement
- Documentation non officielle plus forcément bonne

# Astuces

- Commencer simplement, par des étapes qui fonctionnent
- `when { changeset "**/*.ext"}` pour éviter de relancer inutilement certains stages
- Il est possible de rejouer un build à partir d'une étape mais sur le **même** commit
- Créer un **job manuel** pour faire des tests...
- ...et qui ne teste que l'étape souhaitée
- Exporter, versionner, importer les jobs

# TODO

- Healthcheck sur les conteneurs
- Auto-merge sur main à la réussite du pipeline
- Utiliser un master Jenkins et un ou plusieurs slaves
- Tester l'installation **Jenkins** avec un role ansible  
<https://github.com/geerlingguy/ansible-role-jenkins>
- Job qui exporte ET versionne automatiquement les pipelines

# Conclusion

- Approche kubernetes :
  - Avantages
  - Inconvénients
- Approche pipeline CI/CD avec ansible et docker
  - Avantages
  - Inconvénients