



# PROJET DEVOPS - Orchestration (SAGNANE Saïdou - 5SRC2)

## Objectif du projet

Au cours de ce TP, j'ai été chargé de conteneuriser et de déployer, au sein d'un cluster Kubernetes, plusieurs services essentiels pour l'entreprise IC GROUP :

Un site vitrine développé avec Flask, dans lequel j'ai utilisé des variables d'environnement afin d'adapter dynamiquement les liens vers les autres services ;

L'ERP Odoo 13 (édition communautaire), connecté à une base de données PostgreSQL persistante que j'ai configurée via Kubernetes ;

pgAdmin, que j'ai utilisé pour gérer la base de données d'Odoo, avec une configuration automatisée grâce à un fichier **servers.json**.

# Lancement Minikube

```
minikube start
```

```
ssagnane@ubuntu:~$ minikube start
● minikube v1.35.0 sur Ubuntu 24.04
  Choix automatique du pilote Docker. Autres choix: none, ssh
  Utilisation du pilote Docker avec le privilège root
  Démarrage du nœud "minikube" primary control-plane dans le cluster "minikube"
  Extraction de l'image de base v0.0.46...
  Téléchargement du préchargement de Kubernetes v1.32.0...
    > preloaded-images-k8s-v10...: 333.57 MiB / 333.57 MiB 100.00% 36.12 M
    > gcr.io/k8s-minikube/kicbase...: 500.31 MiB / 500.31 MiB 100.00% 45.72 M
  Création de docker container (CPU2, Memory:2200Mi) ...
  Préparation de Kubernetes v1.32.0 sur Docker 27.4.1...
    * Génération des certificats et des clés
    * Démarrage du plan de contrôle ...
    * Configuration des règles RBAC ...
  Configuration de bridge CNI (Container Networking Interface)...
  Vérification des composants Kubernetes...
    * Utilisation de l'image gcr.io/k8s-minikube/storage-provisioner:v5
  Modules activés: default-storageclass, storage-provisioner
  Terminé ! kubectl est maintenant configuré pour utiliser "minikube" cluster et espace de noms "default" par défaut.
ssagnane@ubuntu:~$ |
```

## Cloner le repo Github

Pour le clonage, j'ai fait un fort sur ce repo:

<https://github.com/OlivierKouokam/mini-projet-5esgi>

```
git clone https://github.com/ssagnane1/mini-projet-5esgi
```

```
ssagnane@ubuntu:~$ git clone https://github.com/ssagnane1/mini-projet-5esgi.git
Cloning into 'mini-projet-5esgi'...
remote: Enumerating objects: 46, done.
remote: Counting objects: 100% (46/46), done.
remote: Compressing objects: 100% (42/42), done.
remote: Total 46 (delta 11), reused 32 (delta 2), pack-reused 0 (from 0)
Receiving objects: 100% (46/46), 906.77 KiB | 8.47 MiB/s, done.
Resolving deltas: 100% (11/11), done.
ssagnane@ubuntu:~$ cd
.cache/          .kube/          mini-projet-5esgi/ .ssh/
.docker/         .minikube/     projet-esgi/
ssagnane@ubuntu:~$ cd mini-projet-5esgi/
ssagnane@ubuntu:~/mini-projet-5esgi$ |
```

## Étape 1 - Conteneurisation de l'application Flask

Dockerfile utilisé:

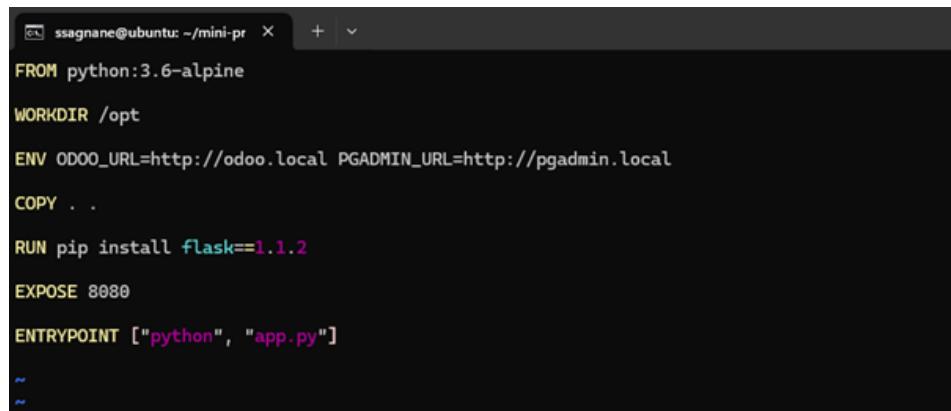
```
FROM python:3.6-alpine
WORKDIR /opt
ENV ODOO_URL=http://odoo.local PGADMIN_URL=http://pgadmin.local
```

```
COPY ..  
RUN pip install flask==1.1.2  
EXPOSE 8080  
ENTRYPOINT ["python", "app.py"]
```

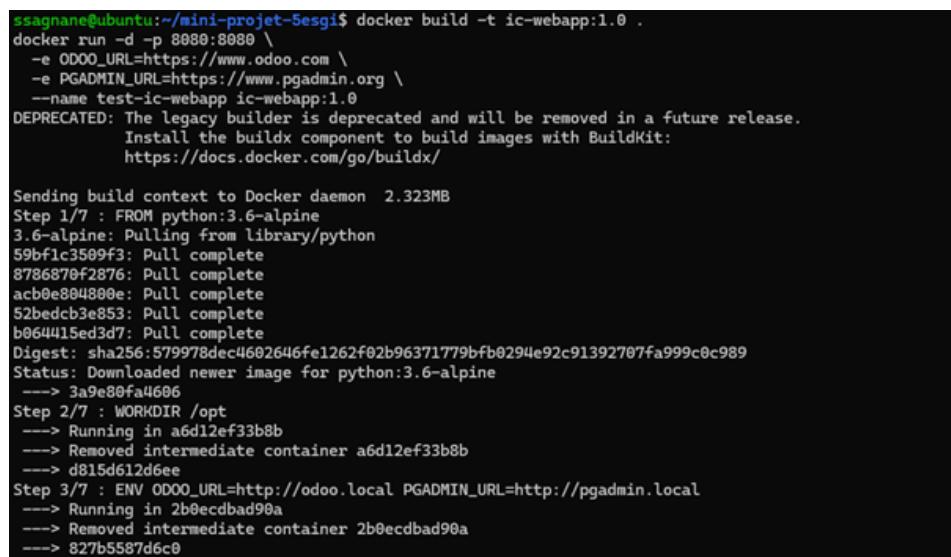
Commandes exécutées:

```
docker build -t ic-webapp:1.0 .  
docker run -d -p 8080:8080 \  
-e OODOO_URL=https://www.odoo.com \  
-e PGADMIN_URL=https://www.pgadmin.org \  
--name test-ic-webapp ic-webapp:1.0
```

Screens:



```
ssagnane@ubuntu: ~/mini-pr x + v  
FROM python:3.6-alpine  
WORKDIR /opt  
ENV OODOO_URL=http://odoo.local PGADMIN_URL=http://pgadmin.local  
COPY ..  
RUN pip install flask==1.1.2  
EXPOSE 8080  
ENTRYPOINT ["python", "app.py"]  
~  
~
```



```
ssagnane@ubuntu:~/mini-projet-5esgi$ docker build -t ic-webapp:1.0 .  
docker run -d -p 8080:8080 \  
-e OODOO_URL=https://www.odoo.com \  
-e PGADMIN_URL=https://www.pgadmin.org \  
--name test-ic-webapp ic-webapp:1.0  
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.  
    Install the buildx component to build images with BuildKit:  
    https://docs.docker.com/go/buildx/  
  
Sending build context to Docker daemon 2.323MB  
Step 1/7 : FROM python:3.6-alpine  
3.6-alpine: Pulling from library/python  
59bf1c3589f3: Pull complete  
8786870f2876: Pull complete  
ac0e804800e: Pull complete  
52bedcb3e853: Pull complete  
b064415ed3d7: Pull complete  
Digest: sha256:579978dec46b2646fe1262f02b96371779bfb0294e92c91392707fa999c0c989  
Status: Downloaded newer image for python:3.6-alpine  
--> 3a9e80fa4606  
Step 2/7 : WORKDIR /opt  
--> Running in a6d12ef33b8b  
--> Removed intermediate container a6d12ef33b8b  
--> d815d612d6ee  
Step 3/7 : ENV OODOO_URL=http://odoo.local PGADMIN_URL=http://pgadmin.local  
--> Running in 2b0ecdbad90a  
--> Removed intermediate container 2b0ecdbad90a  
--> 827b5587d6c0
```

Suite à ça, j'ai pousser l'image sur le docker hub

Commande:

```
docker login  
docker tag ic-webapp:1.0 ssagnane/ic-webapp:1.0  
docker push ssagnane/ic-webapp:1.0
```

Screens:

```
ssagnane@ubuntu:~/mini-projet-5esgi$ docker login  
Authenticating with existing credentials...  
WARNING! Your password will be stored unencrypted in /home/ssagnane/.docker/config.json.  
Configure a credential helper to remove this warning. See  
https://docs.docker.com/engine/reference/commandline/login/#credentials-store  
  
Login Succeeded  
ssagnane@ubuntu:~/mini-projet-5esgi$ docker tag ic-webapp:1.0 ssagnane/ic-webapp:1.0  
ssagnane@ubuntu:~/mini-projet-5esgi$ docker push ssagnane/ic-webapp:1.0  
The push refers to repository [docker.io/ssagnane/ic-webapp]  
41bd9fabf449: Pushed  
5baef78557edc: Pushed  
3156423bd38f: Mounted from library/python  
efa76becf38b: Mounted from library/python  
671e3248113c: Mounted from library/python  
1965cfbef2ab: Mounted from library/python  
8d3ac3489996: Mounted from library/python  
1.0: digest: sha256:aae4224c7ba4db5c028decf8b3445371le1d66d2765173287c4f4184ae71bde4 size: 1790  
ssagnane@ubuntu:~/mini-projet-5esgi$ |
```

The screenshot shows the Docker Hub interface for the repository `ssagnane/ic-webapp`. The repository has 1 tag, `1.0`, which was pushed 1 minute ago. The Docker commands section shows the command to push a new tag: `docker push ssagnane/ic-webapp:tagname`. The `General` tab is selected, showing the following table of tags:

Tag	OS	Type	Pulled	Pushed
1.0		Image	less than 1 day	1 minute

## Étape 2 - Configuration Kubernetes

Lors de la configuration Kubernetes, j'ai déployé plusieurs fichiers. Voici les fichiers de configuration .yaml .

```
ssagnane@ubuntu:~/mini-projet-5esgi/kubernetes$ tree
.
├── namespace.yaml
├── odoo.yaml
├── pgadmin-configmap.yaml
├── pgadmin.yaml
├── postgres.yaml
└── webapp-config.yaml
    └── webapp.yaml
```

Voici les configurations de quelques fichiers .yaml .

#### Fichier **namespace.yaml**

```
apiVersion: v1
kind: Namespace
metadata:
  name: icgroup
  labels:
    env: prod
```

#### Fichier **webapp.yaml**

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: ic-webapp
  namespace: icgroup
  labels:
    env: prod
spec:
  replicas: 1
  selector:
    matchLabels:
      app: ic-webapp
  template:
    metadata:
      labels:
        app: ic-webapp
```

```

env: prod
spec:
  containers:
    - name: ic-webapp
      image: ssagnane/ic-webapp:1.0
      ports:
        - containerPort: 8080
      envFrom:
        - configMapRef:
            name: webapp-config
---
apiVersion: v1
kind: Service
metadata:
  name: ic-webapp
  namespace: icgroup
  labels:
    env: prod
spec:
  type: NodePort
  selector:
    app: ic-webapp
  ports:
    - port: 8080
      targetPort: 8080
      nodePort: 30080

```

## Etape 3 - Déploiement Kubernetes

Lors du déploiement j'ai appliqué mes fichiers .yaml dans le dossier **kubernetes/**

```

ssagnane@ubuntu:~/mini-projet-5esgi$ sudo kubectl apply -f kubernetes/namespace.yaml
namespace/icgroup created
ssagnane@ubuntu:~/mini-projet-5esgi$ sudo kubectl apply -f kubernetes/webapp-config.yaml
configmap/webapp-config created
ssagnane@ubuntu:~/mini-projet-5esgi$ sudo kubectl apply -f kubernetes/postgres.yaml
persistentvolumeclaim/postgres-pvc created
service/postgres created
deployment.apps/postgres created
ssagnane@ubuntu:~/mini-projet-5esgi$ sudo kubectl apply -f kubernetes/odoo.yaml
deployment.apps/odoo created
service/odoo created
ssagnane@ubuntu:~/mini-projet-5esgi$ sudo kubectl apply -f kubernetes/pgadmin-configmap.yaml
configmap/pgadmin-config created
ssagnane@ubuntu:~/mini-projet-5esgi$ sudo kubectl apply -f kubernetes/pgadmin.yaml
persistentvolumeclaim/pgadmin-pvc created
deployment.apps/pgadmin created
service/pgadmin created
ssagnane@ubuntu:~/mini-projet-5esgi$ sudo kubectl apply -f kubernetes/webapp.yaml
deployment.apps/ic-webapp created
service/ic-webapp created

```

Commande pour la vérification des pods, services, deployments; etc...

```
sudo kubectl get all -n icgroup
```

```

ssagnane@ubuntu:~/mini-projet-5esgi/kubernetes$ sudo kubectl get all -n icgroup
NAME                         READY   STATUS    RESTARTS   AGE
pod/ic-webapp-74889cbc67-5p858  1/1    Running   0          23m
pod/odoo-747567b7bc-vvjm8       1/1    Running   0          23m
pod/pgadmin-68585c79db-wxs2t    1/1    Running   0          34s
pod/postgres-6bbb89459c-7m7hn  1/1    Running   0          23m

NAME            TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
service/ic-webapp  NodePort  10.43.9.26    <none>        8080:30080/TCP  23m
service/odoo     NodePort  10.43.207.143  <none>        8069:30069/TCP  23m
service/pgadmin  NodePort  10.43.234.192  <none>        80:30050/TCP   23m
service/postgres ClusterIP  10.43.144.246  <none>        5432/TCP        23m

NAME           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/ic-webapp  1/1     1           1           23m
deployment.apps/odoo       1/1     1           1           23m
deployment.apps/pgadmin   1/1     1           1           23m
deployment.apps/postgres  1/1     1           1           23m

NAME          DESIRED   CURRENT   READY   AGE
replicaset.apps/ic-webapp-74889cbc67  1         1         1      23m
replicaset.apps/odoo-747567b7bc        1         1         1      23m
replicaset.apps/pgadmin-68585c79db    1         1         1      34s
replicaset.apps/pgadmin-c856bdcc       0         0         0      23m
replicaset.apps/postgres-6bbb89459c   1         1         1      23m
ssagnane@ubuntu:~/mini-projet-5esgi/kubernetes$ |

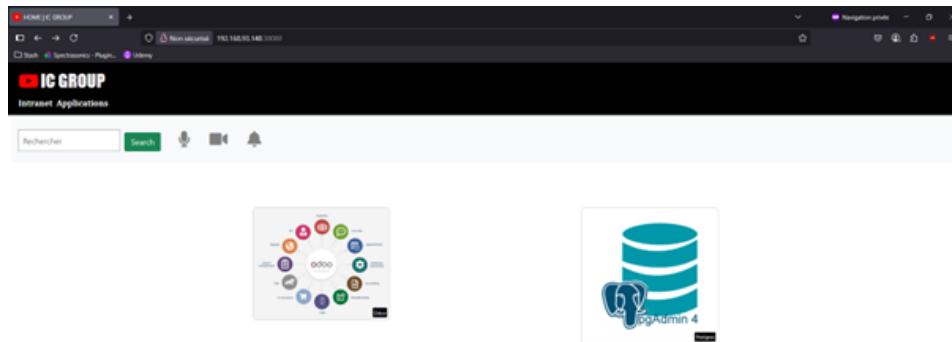
```

## Test

Pour le test c'était sur ma machine virtuel @IP: **192.168.93.148**

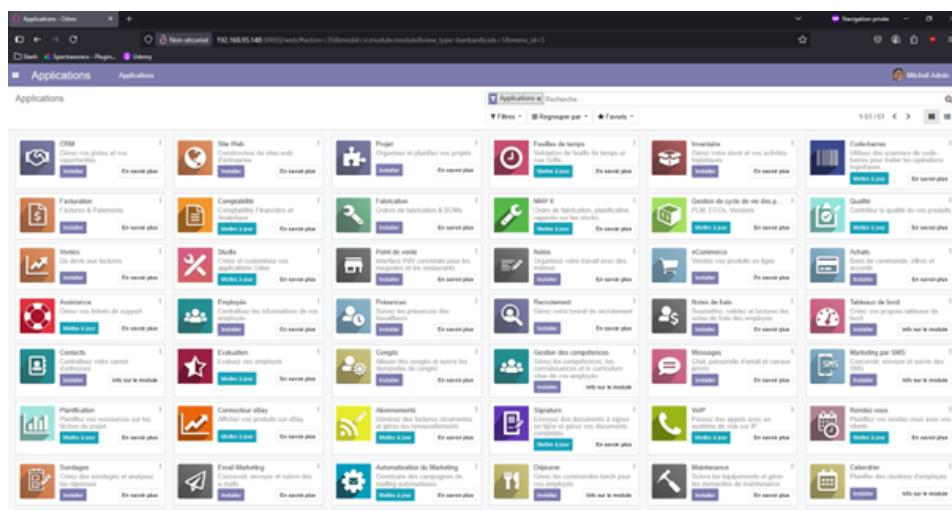
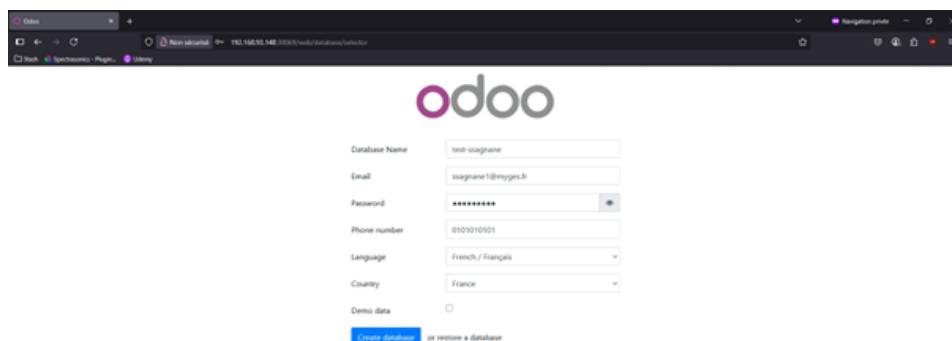
Application	URL
Vitrine Flask	<b>192.168.93.148:30080</b>
Odoo	<b>192.168.93.148:30069</b>
pgAdmin	<b>192.168.93.148:30050</b>

## Vitrine Flask:



## Odoo:

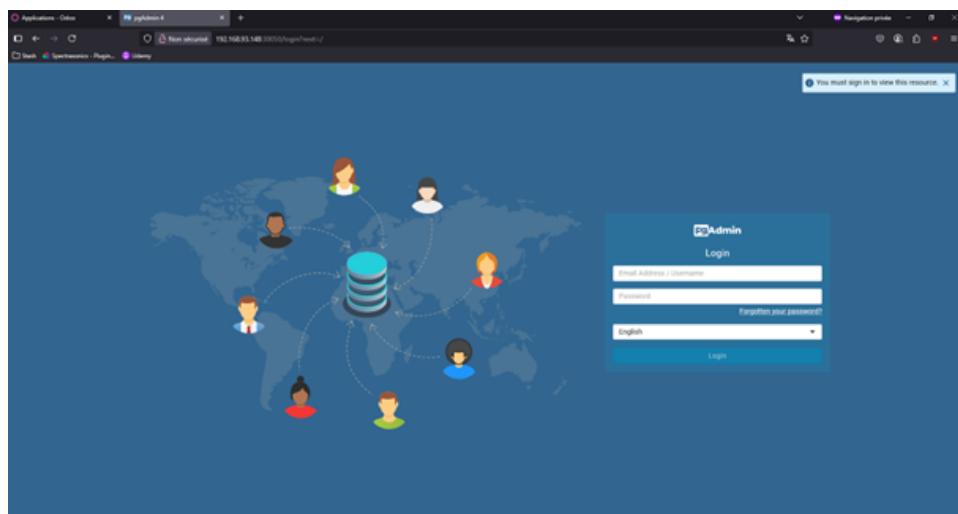
Pour ça j'ai crée une database, et j'ai entrer les identifiants pour créer la base de données.



## pgAdmin:

Pour pgAdmin j'ai utiliser les identifants d'admin que j'ai configuré dans le fichier pgadmin.yaml

```
spec:  
  containers:  
    - name: pgadmin  
      image: dpage/pgadmin4  
      ports:  
        - containerPort: 80  
      env:  
        - name: PGADMIN_DEFAULT_EMAIL  
          value: "ssagnane1@myges.fr"  
        - name: PGADMIN_DEFAULT_PASSWORD  
          value: "admin123"  
      volumeMounts:
```



Pour accéder à la base de donnée Odoo, j'ai renseigné l'identifiant admin odoo/odoo pour me connecter.

Fichier odoo.yaml

```
apiVersion: apps/v1  
kind: Deployment  
metadata:  
  name: odoo
```

```
namespace: icgroup
labels:
  env: prod
spec:
  replicas: 1
  selector:
    matchLabels:
      app: odoo
template:
  metadata:
    labels:
      app: odoo
      env: prod
spec:
  containers:
    - name: odoo
      image: odoo:13.0
      env:
        - name: HOST
          value: "postgres"
        - name: USER
          value: "odoo"
        - name: PASSWORD
          value: "odoo"
  ports:
    - containerPort: 8069
```

