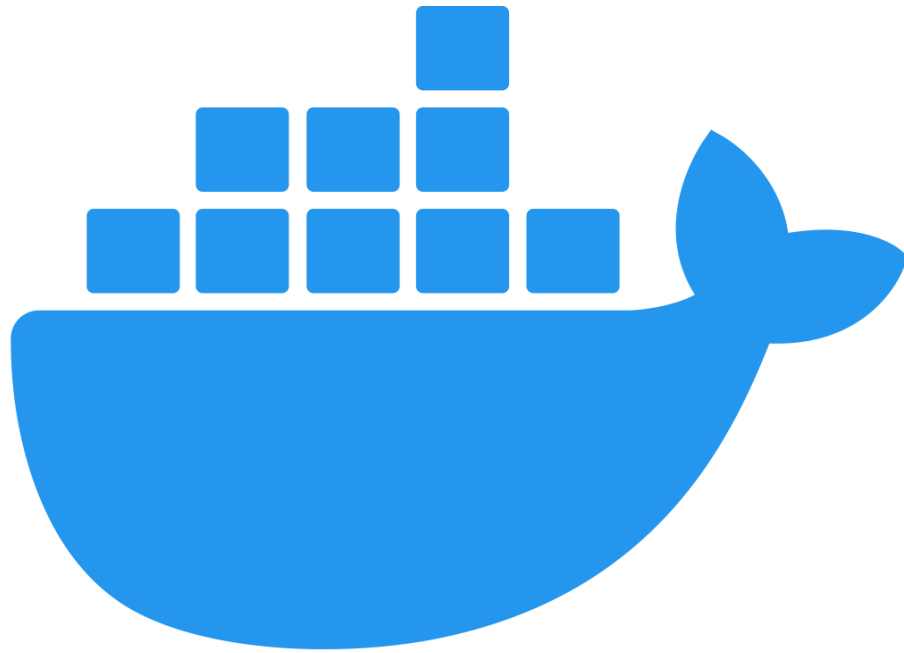


Projet E3 3MSCOR



docker®

Sommaire

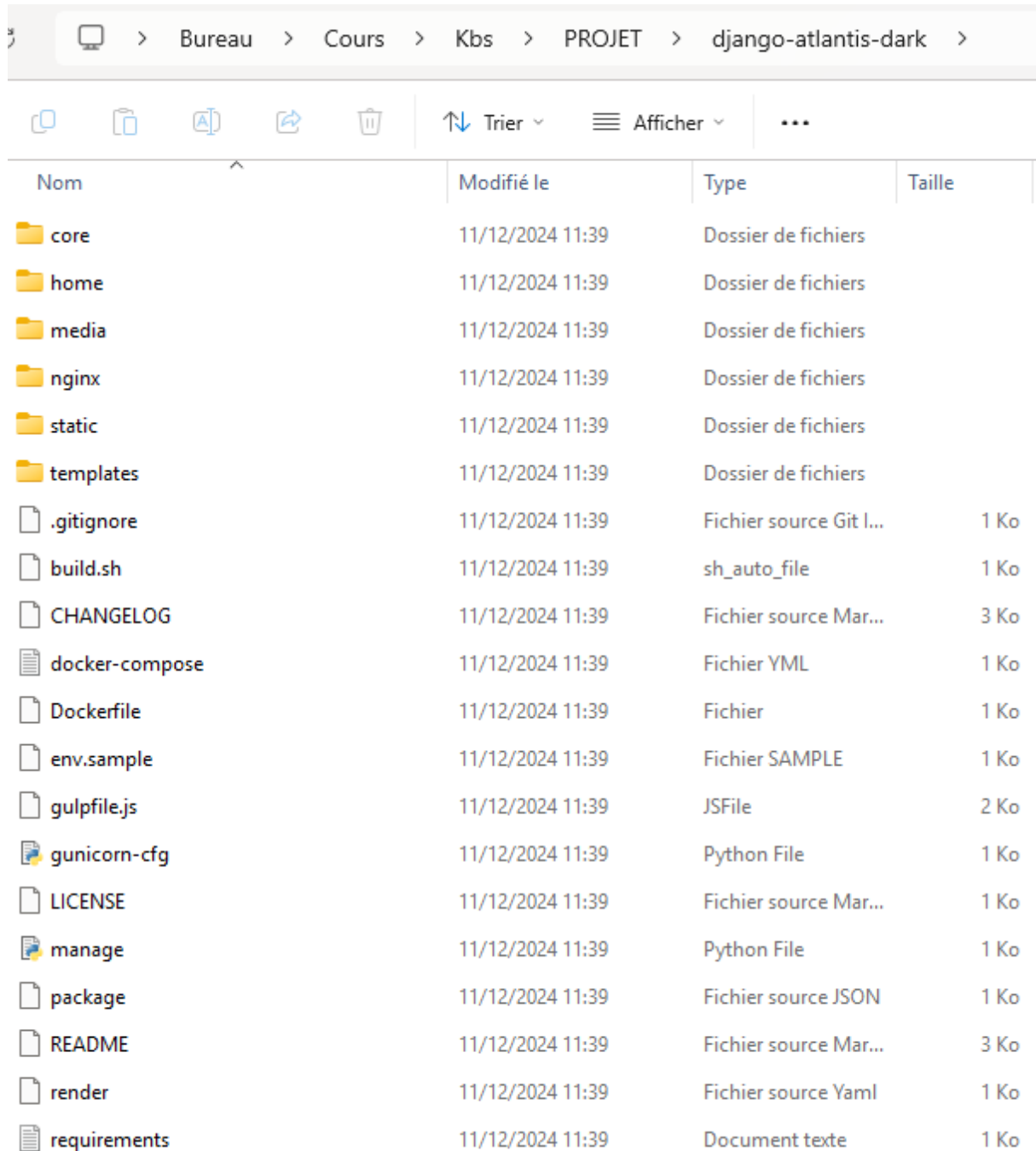
Table des matières

PARTIE : A.....	3
1. Trouver les différentes Applications	3
A. Django-atlantis-dark.....	3
B. Django-argon-dashboard	4
C. Django-soft-ui-dashboard	5
D. Django-adminlte.....	6
E. Django-modernize.....	7
F. Flask-material-dashboard.....	8
G. Django-corporate-dashboard.....	9
2. Création d'un git de dépôt pour rassembler toutes les applications	10
3. Modification des docker-compose.yaml.....	11
4. Modification des dockerfiles.	13
5. Configuration de Nginx.....	14
PARTIE B :	16
1. Répartitions des taches dans l'équipe	16
2. Schéma représentant l'architecture du projet	17
3. Explication des frameworks utilisé	17
4. Système pour héberger notre projet sur docker hub.....	18
PARTIE C :	19
1. Qu'est-ce que le DevOps ?.....	19
2. En quoi le DevOps vous a aidé dans cette mission ?	19
3. Votre chef de projet vous demande de comparer l'utilisation de Docker avec des machines virtuelles. Que répondrez-vous	20
4. À l'aide de vos connaissances personnelles, citer un ou plusieurs autres outils similaires à docker qui pourrait aider votre équipe à livrer et maintenir ce projet client	20

PARTIE : A

1. Trouver les différentes Applications































A. Django-atlantis-dark
































The screenshot shows a file explorer window with the breadcrumb path: Bureau > Cours > Kbs > PROJET > django-atlantis-dark. The file list is as follows:

Nom	Modifié le	Type	Taille
core	11/12/2024 11:39	Dossier de fichiers	
home	11/12/2024 11:39	Dossier de fichiers	
media	11/12/2024 11:39	Dossier de fichiers	
nginx	11/12/2024 11:39	Dossier de fichiers	
static	11/12/2024 11:39	Dossier de fichiers	
templates	11/12/2024 11:39	Dossier de fichiers	
.gitignore	11/12/2024 11:39	Fichier source Git l...	1 Ko
build.sh	11/12/2024 11:39	sh_auto_file	1 Ko
CHANGELOG	11/12/2024 11:39	Fichier source Mar...	3 Ko
docker-compose	11/12/2024 11:39	Fichier YML	1 Ko
Dockerfile	11/12/2024 11:39	Fichier	1 Ko
env.sample	11/12/2024 11:39	Fichier SAMPLE	1 Ko
gulpfile.js	11/12/2024 11:39	JSFile	2 Ko
gunicorn-cfg	11/12/2024 11:39	Python File	1 Ko
LICENSE	11/12/2024 11:39	Fichier source Mar...	1 Ko
manage	11/12/2024 11:39	Python File	1 Ko
package	11/12/2024 11:39	Fichier source JSON	1 Ko
README	11/12/2024 11:39	Fichier source Mar...	3 Ko
render	11/12/2024 11:39	Fichier source Yaml	1 Ko
requirements	11/12/2024 11:39	Document texte	1 Ko

B. Django-argon-dashboard

 > Bureau > Cours > Kbs > PROJET > django-argon-dashboard >				
      Trier ▾  Afficher ▾ ...				
Nom	Modifié le	Type	Taille	
 core	11/12/2024 11:39	Dossier de fichiers		
 home	11/12/2024 11:39	Dossier de fichiers		
 nginx	11/12/2024 11:39	Dossier de fichiers		
 static	11/12/2024 11:39	Dossier de fichiers		
 templates	11/12/2024 11:39	Dossier de fichiers		
 .env	11/12/2024 11:39	Fichier ENV	1 Ko	
 .gitignore	11/12/2024 11:39	Fichier source Git l...	1 Ko	
 build.sh	11/12/2024 11:39	sh_auto_file	1 Ko	
 CHANGELOG	11/12/2024 11:39	Fichier source Mar...down	4 Ko	
 db.sqlite3	11/12/2024 11:39	Fichier SQLITE3	128 Ko	
 docker-compose	11/12/2024 11:39	Fichier YML	1 Ko	
 Dockerfile	11/12/2024 11:39	Fichier	1 Ko	
 env.sample	11/12/2024 11:39	Fichier SAMPLE	1 Ko	
 gulpfile.js	11/12/2024 11:39	JSFile	2 Ko	
 gunicorn-cfg	11/12/2024 11:39	Python File	1 Ko	
 LICENSE	11/12/2024 11:39	Fichier source Mar...	1 Ko	
 manage	11/12/2024 11:39	Python File	1 Ko	
 package	11/12/2024 11:39	Fichier source JSON	1 Ko	
 README	11/12/2024 11:39	Fichier source Mar...	3 Ko	
 README_deploy	11/12/2024 11:39	Fichier source Mar...	1 Ko	
 render	11/12/2024 11:39	Fichier source Yaml	1 Ko	
 requirements	11/12/2024 11:39	Document texte	1 Ko	

C. Django-soft-ui-dashboard

 > Bureau > Cours > Kbs > PROJET > django-soft-ui-dashboard >				
      Trier ▾  Afficher ▾ ...				
Nom	Modifié le	Type	Taille	
 core	11/12/2024 11:39	Dossier de fichiers		
 home	11/12/2024 11:39	Dossier de fichiers		
 nginx	11/12/2024 11:39	Dossier de fichiers		
 static	11/12/2024 11:39	Dossier de fichiers		
 templates	11/12/2024 11:39	Dossier de fichiers		
 .env	11/12/2024 11:39	Fichier ENV	1 Ko	
 .gitignore	11/12/2024 11:39	Fichier source Git l...	1 Ko	
 build.sh	11/12/2024 11:39	sh_auto_file	1 Ko	
 CHANGELOG	11/12/2024 11:39	Fichier source Mar...	7 Ko	
 db.sqlite3	11/12/2024 11:39	Fichier SQLITE3	128 Ko	
 docker-compose	11/12/2024 11:39	Fichier YML	1 Ko	
 Dockerfile	11/12/2024 11:39	Fichier	1 Ko	
 env.sample	11/12/2024 11:39	Fichier SAMPLE	1 Ko	
 gulpfile.js	11/12/2024 11:39	JSFile	2 Ko	
 gunicorn-cfg	11/12/2024 11:39	Python File	1 Ko	
 LICENSE	11/12/2024 11:39	Fichier source Mar...	1 Ko	
 manage	11/12/2024 11:39	Python File	1 Ko	
 package	11/12/2024 11:39	Fichier source JSON	1 Ko	
 README	11/12/2024 11:39	Fichier source Mar...	3 Ko	
 render	11/12/2024 11:39	Fichier source Yaml	1 Ko	
 requirements	11/12/2024 11:39	Document texte	1 Ko	

D. Django-adminlte

Bureau > Cours > Kbs > PROJET > django-adminlte >				
<div> </div> <div> Trier ▾ Afficher ▾ ... </div>				
Nom	Modifié le	Type	Taille	
core	11/12/2024 11:39	Dossier de fichiers		
home	11/12/2024 11:39	Dossier de fichiers		
nginx	11/12/2024 11:39	Dossier de fichiers		
static	11/12/2024 11:39	Dossier de fichiers		
templates	11/12/2024 11:39	Dossier de fichiers		
.env	11/12/2024 11:39	Fichier ENV	1 Ko	
.gitignore	11/12/2024 11:39	Fichier source Git I...	1 Ko	
build.sh	11/12/2024 11:39	sh_auto_file	1 Ko	
CHANGELOG	11/12/2024 11:39	Fichier source Mar...	6 Ko	
db.sqlite3	11/12/2024 11:39	Fichier SQLITE3	128 Ko	
docker-compose	11/12/2024 11:39	Fichier YML	1 Ko	
Dockerfile	11/12/2024 11:39	Fichier	1 Ko	
env.sample	11/12/2024 11:39	Fichier SAMPLE	1 Ko	
gunicorn-cfg	11/12/2024 11:39	Python File	1 Ko	
LICENSE	11/12/2024 11:39	Fichier source Mar...	1 Ko	
manage	11/12/2024 11:39	Python File	1 Ko	
media	11/12/2024 11:39	Fichier source Mar...	1 Ko	
README	11/12/2024 11:39	Fichier source Mar...	3 Ko	
render	11/12/2024 11:39	Fichier source Yaml	1 Ko	
requirements	11/12/2024 11:39	Document texte	1 Ko	

E. Django-modernize

Navigation: Bureau > Cours > Kbs > PROJET > django-modernize >









Actions: [Copier] [Coller] [Renommer] [Partager] [Supprimer] | Trier [v] | Afficher [v] | ...

Nom	Modifié le	Type	Taille
core	11/12/2024 11:39	Dossier de fichiers	
home	11/12/2024 11:39	Dossier de fichiers	
nginx	11/12/2024 11:39	Dossier de fichiers	
static	11/12/2024 11:39	Dossier de fichiers	
templates	11/12/2024 11:39	Dossier de fichiers	
.gitignore	11/12/2024 11:39	Fichier source Git l...	1 Ko
build.sh	11/12/2024 11:39	sh_auto_file	1 Ko
CHANGELOG	11/12/2024 11:39	Fichier source Mar...	1 Ko
docker-compose	11/12/2024 11:39	Fichier YML	1 Ko
Dockerfile	11/12/2024 11:39	Fichier	1 Ko
env.sample	11/12/2024 11:39	Fichier SAMPLE	1 Ko
gunicorn-cfg	11/12/2024 11:39	Python File	1 Ko
LICENSE	11/12/2024 11:39	Fichier source Mar...	1 Ko
manage	11/12/2024 11:39	Python File	1 Ko
README	11/12/2024 11:39	Fichier source Mar...	4 Ko
render	11/12/2024 11:39	Fichier source Yaml	1 Ko
requirements	11/12/2024 11:39	Document texte	1 Ko

F. Flask-material-dashboard

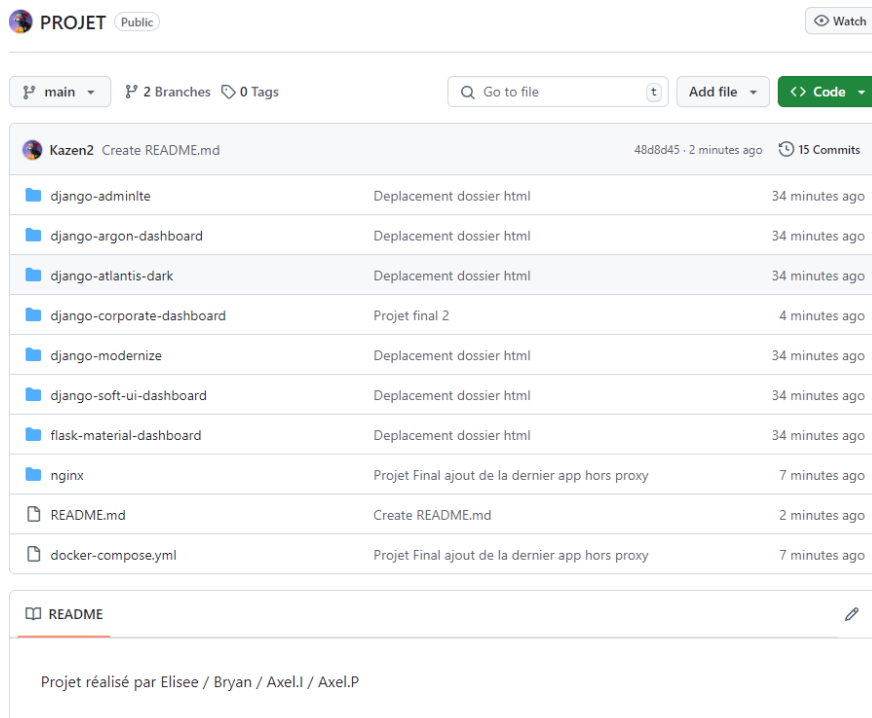
Nom	Modifié le	Type	Taille
apps	11/12/2024 11:39	Dossier de fichiers	
media	11/12/2024 11:39	Dossier de fichiers	
nginx	11/12/2024 11:39	Dossier de fichiers	
.dockerignore	11/12/2024 11:39	Fichier DOCKERIG...	1 Ko
.env	11/12/2024 11:39	Fichier ENV	1 Ko
.gitignore	11/12/2024 11:39	Fichier source Git l...	1 Ko
build.sh	11/12/2024 11:39	sh_auto_file	1 Ko
CHANGELOG	11/12/2024 11:39	Fichier source Mar...	4 Ko
docker-compose	11/12/2024 11:39	Fichier YML	1 Ko
Dockerfile	11/12/2024 11:39	Fichier	1 Ko
env.sample	11/12/2024 11:39	Fichier SAMPLE	1 Ko
gunicorn-cfg	11/12/2024 11:39	Python File	1 Ko
LICENSE	11/12/2024 11:39	Fichier source Mar...	1 Ko
README	11/12/2024 11:39	Fichier source Mar...	9 Ko
render	11/12/2024 11:39	Fichier source Yaml	1 Ko
requirements	11/12/2024 11:39	Document texte	1 Ko
run	11/12/2024 11:39	Python File	2 Ko

G. Django-corporate-dashboard

<div>  > Bureau > Cours > Kbs > PROJET > django-corporate-dashboard </div>			
<div>      <div>  Trier ▾  Afficher ▾ ... </div> </div>			
Nom	Modifié le	Type	Taille
core	11/12/2024 16:07	Dossier de fichiers	
home	11/12/2024 16:07	Dossier de fichiers	
static	11/12/2024 16:07	Dossier de fichiers	
templates	11/12/2024 16:07	Dossier de fichiers	
.gitignore	11/12/2024 16:07	Fichier source Git l...	1 Ko
build.sh	11/12/2024 16:07	sh_auto_file	1 Ko
CHANGELOG	11/12/2024 16:07	Fichier source Mar...	2 Ko
Dockerfile	11/12/2024 16:07	Fichier	1 Ko
entrypoint.sh	11/12/2024 16:07	sh_auto_file	1 Ko
env.sample	11/12/2024 16:07	Fichier SAMPLE	1 Ko
gulpfile.js	11/12/2024 16:07	JSFile	2 Ko
gunicorn-cfg	11/12/2024 16:07	Python File	1 Ko
LICENSE	11/12/2024 16:07	Fichier source Mar...	1 Ko
manage	11/12/2024 16:07	Python File	1 Ko
package	11/12/2024 16:07	Fichier source JSON	1 Ko
README	11/12/2024 16:07	Fichier source Mar...	3 Ko
render	11/12/2024 16:07	Fichier source Yaml	1 Ko
requirements	11/12/2024 16:07	Document texte	1 Ko

Cette application ne fonctionne pas avec le reverse proxy alors que toutes les autres applications fonctionnent.

2. Création d'un git de dépôt pour rassembler toutes les applications



Pour créer ce projet il faut télécharger toutes les applications en local sur nos pcs pour les pushes sur le git de dépôts. Une fois toutes les applications déposer, on peut travailler ensemble sur le projet

Voici les commandes utilisées pour envoyer toutes les applications sur le git:

- git init
- git add html5up-phantom
- git commit -m "ajout html"
- git remote add origin

<https://github.com/Kazen2/PROJET.git>

- git push -u origin mai

Les membres de notre groupe ont juste à Pull pour récupérer le projet pour travailler dessus

3. Modification des docker-compose.yaml

Suppression de tous les dockercompose.yaml pour en créer un seul à la racine du projet, ce fichier va permettre de gérer toutes les applications avec la partie network notamment.

```

1  version: '3.8'
2
3  services:
4    django_atlantis_dark:
5      build: './django-atlantis-dark'
6      container_name: django-atlantis-dark
7      restart: always
8
9    django_corporate_dashboard:
10     build: './django-corporate-dashboard'
11     container_name: django-corporate-dashboard
12     restart: always
13     ports:
14       - '5005:5005'
15
16    django_argon_dashboard:
17     build: './django-argon-dashboard'
18     container_name: django-argon-dashboard
19     restart: always
20
21    django_soft_ui_dashboard:
22     build: './django-soft-ui-dashboard'
23     container_name: django-soft-ui-dashboard
24     restart: always
25
26    django_adminlte:
27     build: './django-adminlte'
28     container_name: django-adminlte
29     restart: always
30
31    django_modernize:
32     build: './django-modernize'
33     container_name: django-modernize
34     restart: always
35
36    flask_material_dashboard:
37     build: './flask-material-dashboard'
38     container_name: flask-material-dashboard
39     restart: always
40    db:
41     image: postgres:13
42     container_name: postgres_db
43     restart: always

```

```
44     environment:
45         POSTGRES_USER: user
46         POSTGRES_PASSWORD: password
47         POSTGRES_DB: app_db
48     volumes:
49         - postgres_data:/var/lib/postgresql/data
50
51     nginx:
52         image: nginx:alpine-slim
53         container_name: nginx_proxy
54         restart: always
55         ports:
56             - "80:80"
57             - "8081:8081"
58             - "8082:8082"
59             - "8083:8083"
60             - "8084:8084"
61             - "8085:8085"
62             - "8086:8086"
63         volumes:
64             - ./nginx:/etc/nginx/conf.d
65
66     volumes:
67         postgres_data:
```

Voici le docker compose global qui gère toutes les applications. Cela permet d'avoir qu'un fichier au lieu d'un fichier par application.

4. Modification des dockerfiles.

Optimisation des dockerfile pour alléger les applications, suppression des commandes run en trop.

```
FROM python:3.9

# set environment variables
ENV PYTHONDONTWRITEBYTECODE 1
ENV PYTHONUNBUFFERED 1

COPY requirements.txt .
COPY env.sample .env

# install python dependencies
RUN pip install --upgrade pip && pip install --no-cache-dir -r requirements.txt

COPY . .

# running migrations
RUN python manage.py migrate

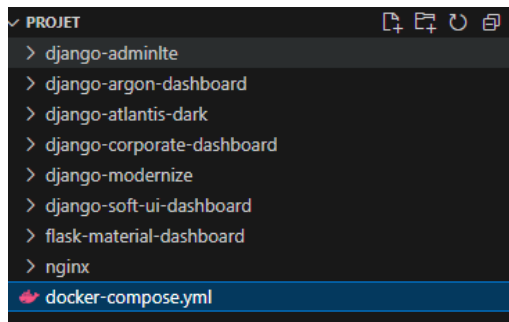
# gunicorn
CMD ["gunicorn", "--config", "gunicorn-cfg.py", "core.wsgi"]
```

Voici ci-dessus un dockerfile de la première application, ce fichier a été optimiser par les actions run qui on était limité. Cela permet de rendre le fichier ma lourd donc plus optimiser

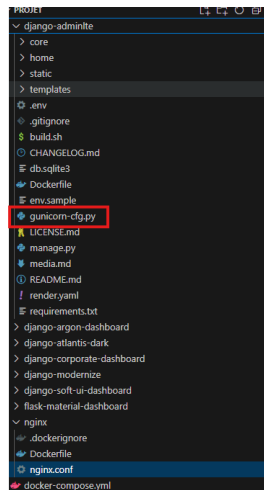
De plus on a rajouté la ligne COPY env.sample .env, cela permet de charger ses variables d'environnement lors de lancement du conteneur.

5. Configuration de Nginx

Pour la configuration du Nginx j'ai supprimé tous les nginx de tous les micro-service afin d'en créer un seul a la racine du projet



Ensuite j'ai fait par service, je me suis rendu dans leur unicorn afin de modifier les ports en mettant 8081, 8082, ..., 8086.



Une fois modifier j'ai créé un dossier nginx.conf ou dedans j'ai renseigné chaque service avec leur port ou il y a écrit en rouge le micro service et en jaune le port situé dans le unicorn **django-adminlte** :8081 le listen et en réalité le port de sortie du microservice.

```

1 upstream django-adminlte {
2     server django-adminlte:8081;
3 }
4
5 server {
6     listen 8081;
7     server_name localhost;
8
9     location / {
10        proxy_pass http://django-adminlte;
11        proxy_set_header Host $host;
12        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
13    }
14
15 }
16

```

Une fois créé je l'es renseigner dans le docker-compose

```
0
1  nginx:
2    image: nginx:alpine-slim
3    container_name: nginx_proxy
4    restart: always
5    ports:
6      - "80:80"
7      - "8081:8081"
8      - "8082:8082"
9      - "8083:8083"
0      - "8084:8084"
1      - "8085:8085"
2      - "8086:8086"
3    volumes:
4      - ./nginx:/etc/nginx/conf.d
5
```

PARTIE B :

1. Répartitions des tâches dans l'équipe

Axel IRSUTTI :

- Conception du rapport final de projet
- Choix de 2 applications
- Test du fonctionnement des applications

Axel PEIFFER :

- Gestion et création d'un github de dépôt
- Configuration du Nginx
- Mise en place de la connexion en local de la 7eme application
- Debug sur docker desktop

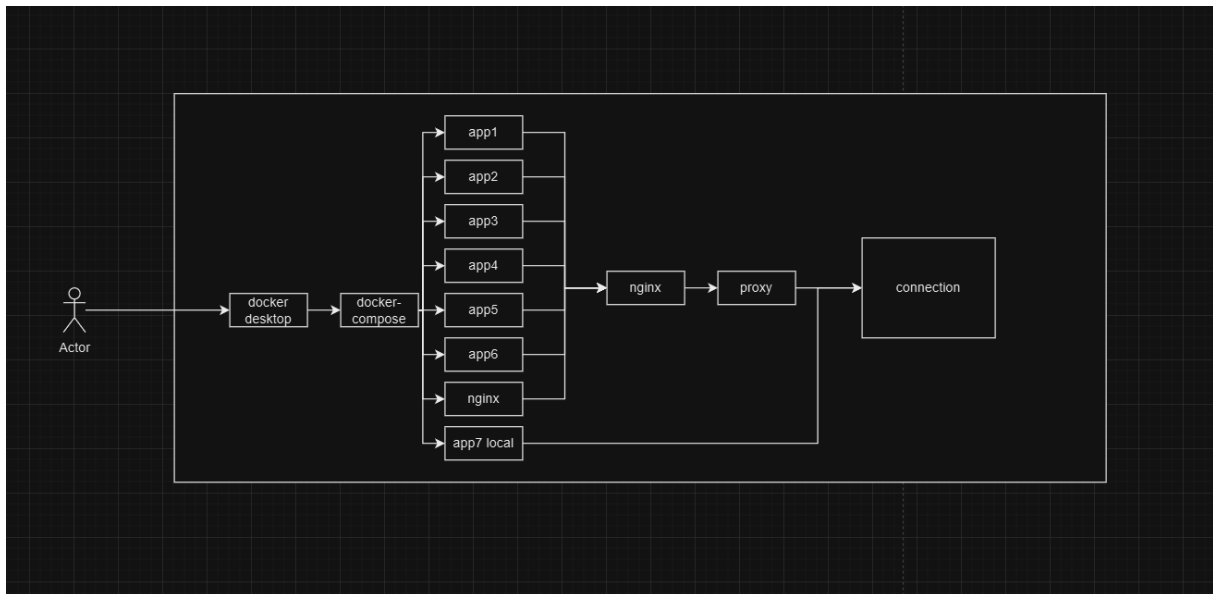
Bryan MAHAUT :

- Configuration et optimisation des fichiers dockercompose.yaml
- Réponse aux questions de la partie C
- Déploiement de 2 Applications

Elisee AZONSI :

- Configuration et optimisation des fichiers dockerfile
- Débug sur dockercompose

2. Schéma représentant l'architecture du projet



3. Explication des frameworks utilisé

Pour le développement de nos applications, nous avons choisi les frameworks Django et Flask en raison de leurs avantages complémentaires.

Django est idéal pour les projets complexes grâce à sa structure complète ("batteries incluses"), sa sécurité intégrée, sa scalabilité, et sa riche documentation. Il offre des fonctionnalités prêtes à l'emploi, accélérant le développement tout en garantissant une base solide et sécurisée.

Flask, plus léger et flexible, est parfait pour les projets nécessitant une personnalisation accrue et un développement rapide. Son approche minimaliste et son extensibilité permettent de créer des solutions sur mesure sans surcharge inutile.

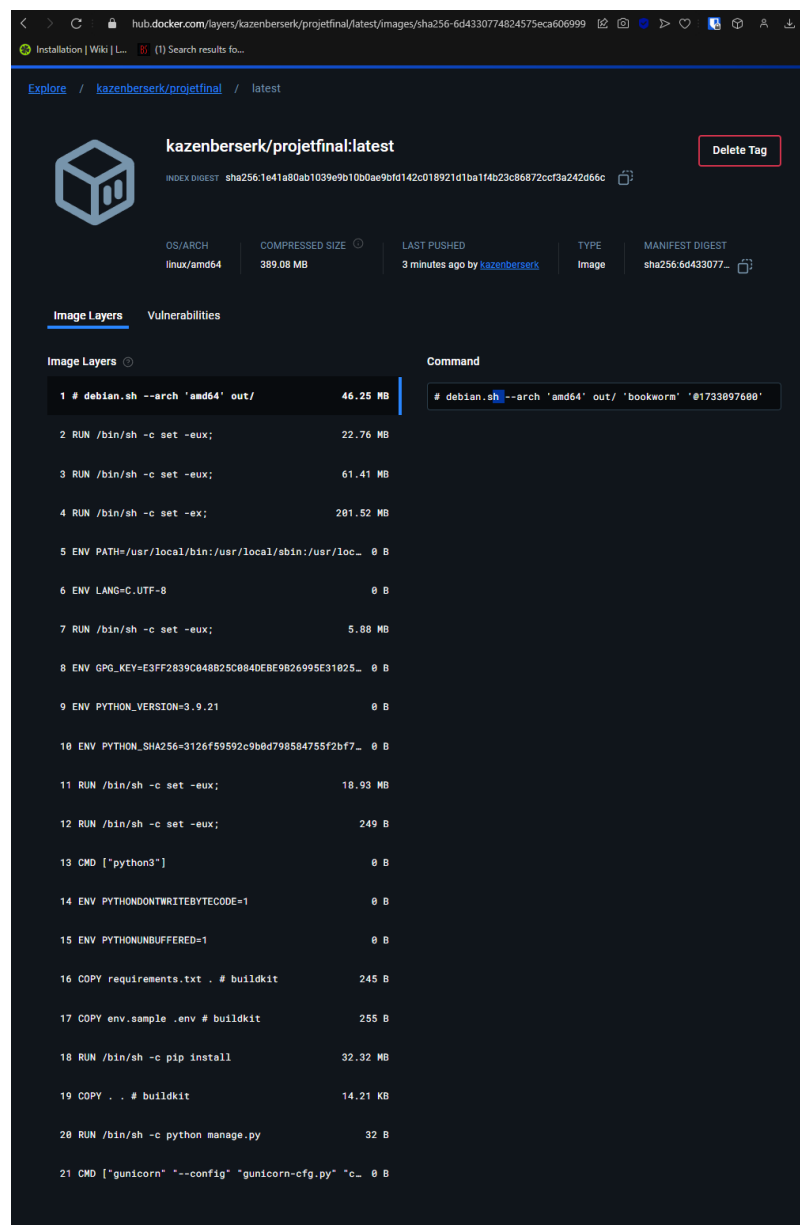
4. Système pour héberger notre projet sur docker hub

Création d'un docker hub

```
PS F:\03--ESTIAME3\3MSCOR\PROJET> docker tag projet-django_corporate_dashboard:latest kazenberserk/projetfinal:latest
PS F:\03--ESTIAME3\3MSCOR\PROJET> docker tag projet-django_modernize:latest kazenberserk/projetfinal:latest
PS F:\03--ESTIAME3\3MSCOR\PROJET> docker tag projet-django_atlantis_dark:latest kazenberserk/projetfinal:latest
PS F:\03--ESTIAME3\3MSCOR\PROJET> docker tag projet-django_soft_ui_dashboard:latest kazenberserk/projetfinal:latest
PS F:\03--ESTIAME3\3MSCOR\PROJET> docker tag projet-django_argon_dashboard:latest kazenberserk/projetfinal:latest
PS F:\03--ESTIAME3\3MSCOR\PROJET> docker tag projet-flask_material_dashboard:latest kazenberserk/projetfinal:latest
PS F:\03--ESTIAME3\3MSCOR\PROJET> docker tag projet-django_adminlte:latest kazenberserk/projetfinal:latest
PS F:\03--ESTIAME3\3MSCOR\PROJET> docker push kazenberserk/projetfinal:latest
The push refers to repository [docker.io/kazenberserk/projetfinal]
afcba355cf81: Pushed
```

Voici le lien du docker hub pour retrouver notre projet :

<https://hub.docker.com/r/kazenberserk/projetfinal/tags>



hub.docker.com/layers/kazenberserk/projetfinal/latest/images/sha256-6d4330774824575eca606999

Installation | Wiki | Search results for...

Explore / kazenberserk/projetfinal / latest

kazenberserk/projetfinal:latest Delete Tag

INDEX DIGEST sha256:1e41a80ab1039e9b10b0ae9bfd142c018921d1ba1f4b23c86872ccf3a242d66c

OS/ARCH	COMPRESSED SIZE	LAST PUSHED	TYPE	MANIFEST DIGEST
linux/amd64	389.08 MB	3 minutes ago by kazenberserk	Image	sha256:6d433077...

Image Layers Vulnerabilities

Image Layers

Layer	Command	Size
1 # debian.sh --arch 'amd64' out/	# debian.sh --arch 'amd64' out/ 'bookworm' '#1733897680'	46.25 MB
2 RUN /bin/sh -c set -eux;		22.76 MB
3 RUN /bin/sh -c set -eux;		61.41 MB
4 RUN /bin/sh -c set -ex;		281.52 MB
5 ENV PATH=/usr/local/bin:/usr/local/sbin:/usr/local...		0 B
6 ENV LANG=C.UTF-8		0 B
7 RUN /bin/sh -c set -eux;		5.88 MB
8 ENV GPG_KEY=E3FF2839C848B25C084DEB9B26995E31025...		0 B
9 ENV PYTHON_VERSION=3.9.21		0 B
10 ENV PYTHON_SHA256=3126f59592c9b8d798584755f2bf7...		0 B
11 RUN /bin/sh -c set -eux;		18.93 MB
12 RUN /bin/sh -c set -eux;		249 B
13 CMD ["python3"]		0 B
14 ENV PYTHONDONTWRITEBYTECODE=1		0 B
15 ENV PYTHONUNBUFFERED=1		0 B
16 COPY requirements.txt . # buildkit		245 B
17 COPY env.sample .env # buildkit		255 B
18 RUN /bin/sh -c pip install		32.32 MB
19 COPY . . # buildkit		14.21 KB
20 RUN /bin/sh -c python manage.py		32 B
21 CMD ["unicorn" "--config" "unicorn-cfg.py" "c_...		0 B

PARTIE C :

1. Qu'est-ce que le DevOps ?

DevOps est une approche qui combine le développement logiciel (Dev) et les opérations informatiques (Ops) pour améliorer la collaboration et la productivité en automatisant les processus de développement, de test et de déploiement. L'objectif est de livrer des applications plus rapidement et de manière plus fiable.

2. En quoi le DevOps vous a aidé dans cette mission ?

Dans cette mission, DevOps a été crucial pour plusieurs raisons :

Automatisation : Grâce à des outils comme Docker et Docker Compose, nous avons pu automatiser le déploiement des applications, ce qui a réduit les erreurs humaines et accéléré le processus de mise en production.

Intégration continue : DevOps nous a permis de mettre en place des pipelines d'intégration continue (CI) pour tester et valider les modifications de code en continu, assurant ainsi une qualité constante du logiciel.

Collaboration : En facilitant la communication entre les équipes de développement et d'opérations, DevOps a amélioré la collaboration et la réactivité face aux problèmes.

3. Votre chef de projet vous demande de comparer l'utilisation de Docker avec des machines virtuelles. Que répondez-vous

Léger : Les conteneurs partagent le noyau du système d'exploitation hôte, ce qui les rend plus légers et plus rapides à démarrer.

Isolation : Chaque conteneur fonctionne de manière isolée, mais ils partagent le même noyau, ce qui réduit l'empreinte mémoire.

Portabilité : Les conteneurs peuvent être facilement déplacés entre différents environnements (développement, test, production).

Isolation complète : Chaque VM inclut un système d'exploitation complet, offrant une isolation totale mais au prix d'une plus grande consommation de ressources.

Démarrage plus lent : Les VMs prennent plus de temps à démarrer en raison de la nécessité de charger un système d'exploitation complet.

Utilisation des ressources : Les VMs consomment plus de ressources (CPU, mémoire) car elles incluent un système d'exploitation complet.

4. À l'aide de vos connaissances personnelles, citer un ou plusieurs autres outils similaires à docker qui pourrait aider votre équipe à livrer et maintenir ce projet client

Kubernetes : Un système de gestion de conteneurs qui automatise le déploiement, la mise à l'échelle et la gestion des applications conteneurisées.

Podman : Un outil de gestion de conteneurs qui fonctionne sans démon, offrant une alternative à Docker avec une compatibilité similaire.

OpenShift : Une plateforme de conteneurs basée sur Kubernetes, offrant des fonctionnalités supplémentaires pour le développement et le déploiement d'applications.