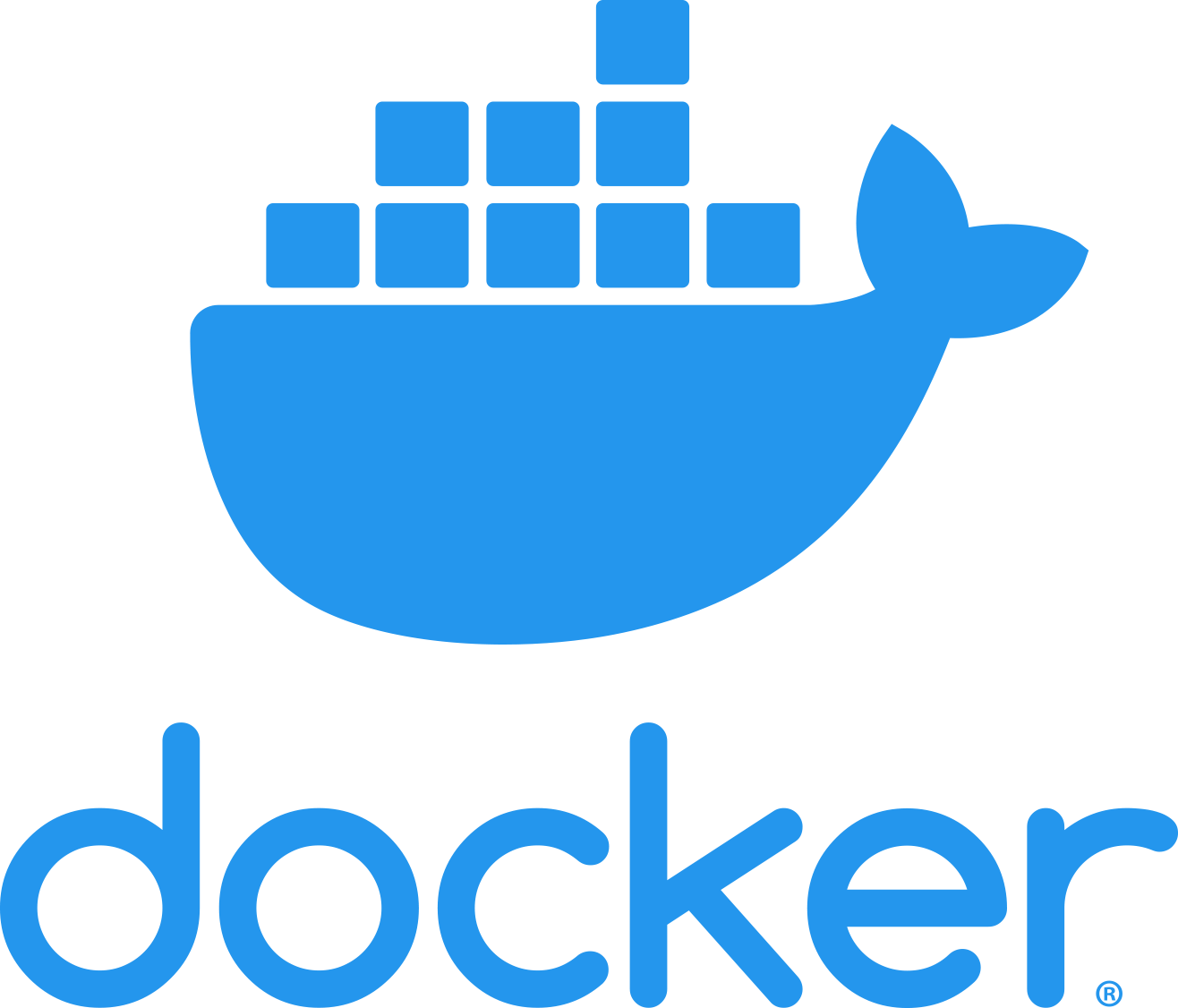
**Projet E3 3MSCOR**



**Sommaire**

Table des matières

[PARTIE : A 3](#_Toc184829392)

[1. Trouver les différentes Applications 3](#_Toc184829393)

[A. Django-atlantis-dark 3](#_Toc184829394)

[B. Django-argon-dashboard 3](#_Toc184829395)

[C. Django-soft-ui-dashboard 3](#_Toc184829396)

[D. Django-adminlte 3](#_Toc184829397)

[E. Django-modernize 3](#_Toc184829398)

[F. Flask-material-dashboard 3](#_Toc184829399)

[G. Django-corporate-dashboard 3](#_Toc184829400)

[2. Création d’un git de dépôt pour rassembler toutes les applications 3](#_Toc184829401)

[3. Modification des docker-compose.yaml 3](#_Toc184829402)

[4. Modification des dockerfiles. 3](#_Toc184829403)

[5. Configuration de Nginx 3](#_Toc184829404)

[PARTIE B : 3](#_Toc184829405)

[1. Répartissions des taches dans l’équipe 3](#_Toc184829406)

[2. Schéma représentant l’architecture du projet 3](#_Toc184829407)

[3. Explication des frameworks utilisé 3](#_Toc184829408)

[4. Système pour héberger notre projet sur docker hub 3](#_Toc184829409)

[PARTIE C : 3](#_Toc184829410)

[1. Qu'est-ce que le DevOps ? 3](#_Toc184829411)

[2. En quoi le DevOps vous a aidé dans cette mission ? 3](#_Toc184829412)

[3. Votre chef de projet vous demande de comparer l'utilisation de Docker avec des machines virtuelles. Que répondrez-vous 3](#_Toc184829413)

[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 3](#_Toc184829414)

# PARTIE : A

## Trouver les différentes Applications

1. Django-atlantis-dark

Une image contenant texte, capture d’écran, menu, nombre

Description générée automatiquement

1. Django-argon-dashboard

Une image contenant texte, capture d’écran, nombre, menu

Description générée automatiquement

1. Django-soft-ui-dashboard

Une image contenant texte, capture d’écran, menu, nombre

Description générée automatiquement

1. Django-adminlte

Une image contenant texte, capture d’écran, nombre, menu

Description générée automatiquement

1. Django-modernize

Une image contenant texte, capture d’écran, nombre, Police

Description générée automatiquement

1. Flask-material-dashboard

Une image contenant texte, capture d’écran, nombre, Police

Description générée automatiquement

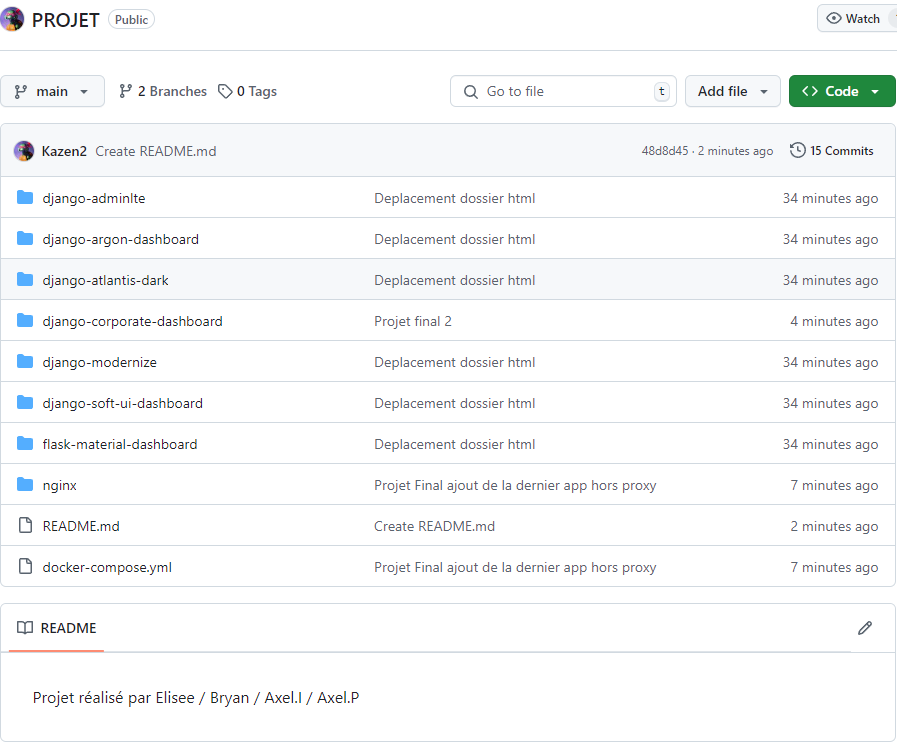
1. Django-corporate-dashboard

Une image contenant texte, capture d’écran, nombre, menu

Description générée automatiquement

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

## 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 pushs 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

## 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.

Une image contenant texte, document, capture d’écran, menu

Description générée automatiquement

Une image contenant texte, capture d’écran, Police, document

Description générée automatiquement

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

## Modification des dockerfiles.

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

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

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.

## 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

Une image contenant texte, capture d’écran, logiciel, Police

Description générée automatiquement

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

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Description générée automatiquement

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

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Description générée automatiquement

Une fois créé je l’es renseigner dans le docker-compose Une image contenant texte, capture d’écran, Police

Description générée automatiquement

# PARTIE B :

## Répartissions des taches 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

## Schéma représentant l’architecture du projet

Une image contenant capture d’écran, diagramme

Description générée automatiquement

## 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.

## Système pour héberger notre projet sur docker hub

Création d’un docker hub

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

Voici le lien du docker hub pour retrouver notre projet : <https://hub.docker.com/r/kazenberserk/projetfinal/tags>

Une image contenant texte, capture d’écran, logiciel

Description générée automatiquement

# PARTIE C :

## 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.

## 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.

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

## 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.