



ÉCOLE
D'INGÉNIEURS
PARIS-LA DÉFENSE

CLOUD APPLICATION DEVELOPMENT

RAPPORT DE PROJET

MATTHIEU LANVERT – VICTOR QUERETTE

IBO 5A – JANVIER 2019

SOMMAIRE

1) Architecture de l'application.....	3
a) Fonctionnement général.....	3
B) Schéma des données.....	3
C) Choix d'implémentation.....	4
2) La webapp.....	5
a) Installation et utilisation.....	5

1) ARCHITECTURE DE L'APPLICATION

A) FONCTIONNEMENT GÉNÉRAL

Notre projet consiste en une application web, basée sur une API écrite en python et utilisant le framework Flask.

Cette API fait l'intermédiaire entre le front-end web réalisé en HTML et js, et le cluster de serveurs mis en disposition par le service Atlas de mongoDB.

B) SCHÉMA DES DONNÉES

L'import des données dans notre base se fait en deux étapes. La première est une copie des tables SQL du dataset choisi (disponible [ici](#)) dans mongoDB. Cette copie construit une base de données *imports*, contenant le même schéma « relationnel » que la DB originale. Ce schéma est fournie dans l'archive du dépôt au format svg.

Cet import se fait en utilisant conjointement les programmes *mysqldump* et *mongoimport*. Ce dernier supporte nativement le format csv que sort *mysqldump* et effectue la conversion JSON directement.

Dans un second temps, ces données sont remaniées en un schéma noSQL. Cette étape est effectuée par la webapp directement : au démarrage, si la base de données *data* est absente du serveur, la dénormalisation est effectuée.

C) CHOIX D'IMPLÉMENTATION

Les scripts python constituant la webapp sont exécutés dans un environnement virtuel afin de n'installer les dépendances requises (le driver *pymongo* notamment) que dans cet environnement. Cet environnement est généré via *make*. Le lancement de la webapp et des fonctionnalités de gestion de la base de données s'effectue via un script bash *run* situé à la racine du dépôt.

Le script *app.py* est le point d'entrée de la webapp, et est celui exécuté par Flask. Il contient la définition de toutes les routes de la webapp et de l'API. Le script *mongo_ops.py* définit une classe *MongoOps* contenant le code nécessaire à l'interaction avec la base de données. C'est lors de l'instanciation de cette classe que l'application vérifie la présence de la DB dénormalisée, et, le cas échéant, se charge des opérations de passage à un schéma noSQL.

2) LA WEBAPP

A) INSTALLATION ET UTILISATION

Le projet peut être trouvé sur le dépôt git hébergé sur [cloud_app_dev_tp](#), par clonage git ou en téléchargeant une archive dans l'onglet *releases*.

Le fichier README.md contient toutes les instructions nécessaires au build et au run de l'application. Il est à noter que les dépendances de cette dernière y sont listées et sont requises.

La projet a été testé au cours du développement, au build et à l'exécution, sur les plateformes Linux suivantes : Debian 9, Majaro 18, Alpine 3.8, Ubuntu 18 et Gentoo.