Distributed Programing report

LAI Khang Duy - Lylia SOMETHINGHERE

29-01-2022

Contents

1 Bonus work Qwiklabs

Please find the link here

- LAI KHANG DUY
- LYLIA

2 Resource

Please find the source code of the project here.

CLICK HERE

3 Présentation du projet

The purpose of the project is to prove the concept of using multiple technologies for distributed programming. With this project, we have built a full web server with 2 backends, 1 frontend and 1 database to showcase the problems.

Each of it is under a Docker container and defined to comunicate with each other using Docker Compose.

The front end fetch from 2 backends with 2 independent tasks.

Notre projet consiste à réaliser une application web tout en utilisant les technologies acquises lors du cours Programmation distribuée.

Et pour cela nous avons créé une application qui affiche la température pour n'importe quelle ville au monde ainsi qu'un enregistrement des températures précédentes récupéré depuis la base de données.

4 Introduction about the technology

4.1 Django

Django is a backend framework written in Python

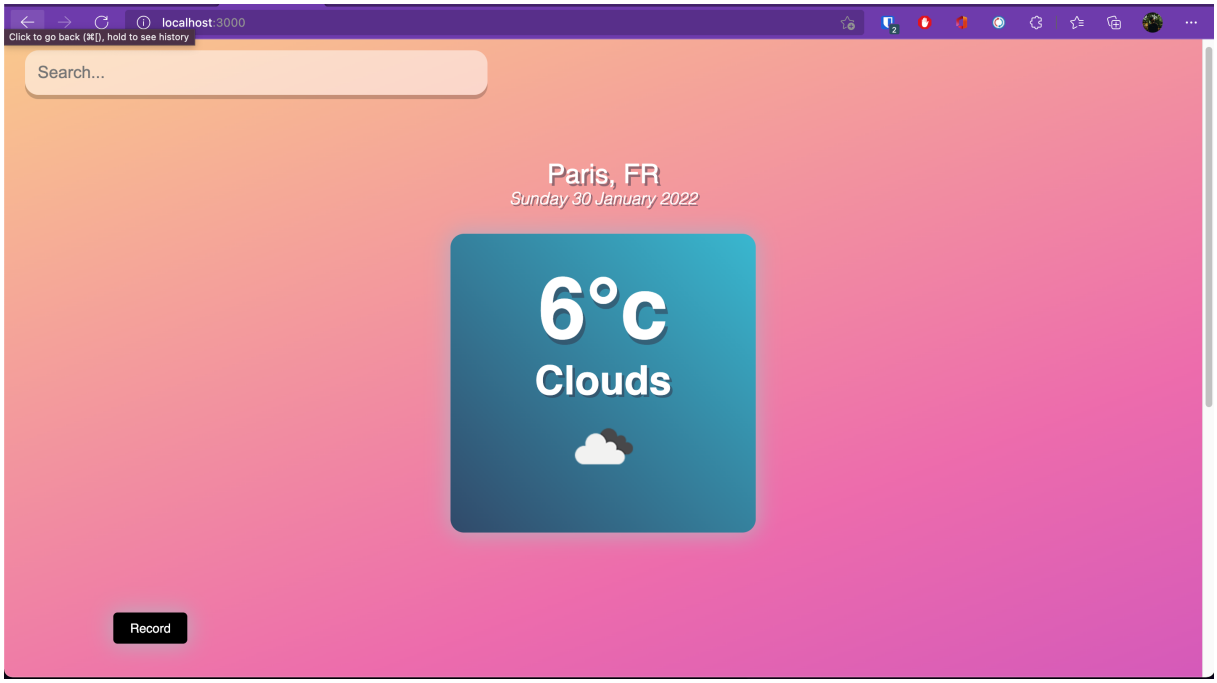

Figure 1: Show weather module

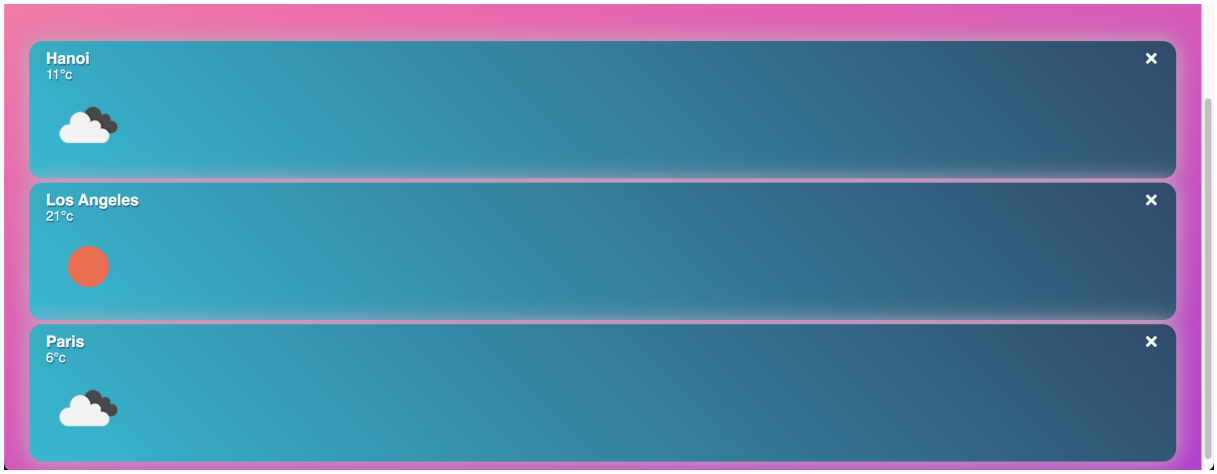

Figure 2: Show record module

4.2 React.JS

Please write some here

4.3 PostgreSQL

Please write some here

5 Architecture de l'application

Pour expliquer le fonctionnement de notre application web, nous avons schématisé l'architecture logicielle de notre application comme on peut le voir sur la figure ci-dessous:

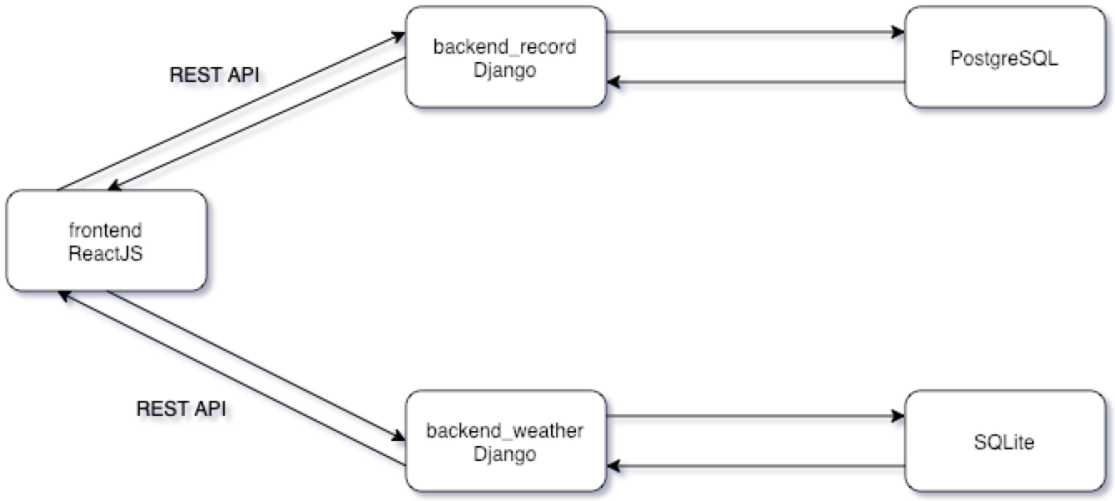

Figure 3: Architecture de l'application

Notre application web utilisant le web service REST est composé de :

- Un front end (qui nous sert de navigateur) codé en REACT JS ,
- Deux back end : un qui sert à stocker et à extraire les données depuis notre base donnée codé en python en utilisant DJANGO , le deuxième qui sert à récupérer la température d'une autre API codé en python en utilisant DJANGO
- Base de donnée POSTGRESQL

6 Distributed programming used in the project

6.1 Docker

Les conteneurs fonctionnent un peu comme les VM, mais de manière beaucoup plus spécifique et granulaire. Ils isolent une seule application et ses dépendances - toutes les bibliothèques logicielles externes dont l'application a besoin pour fonctionner - à la fois du système d'exploitation sous-jacent et des autres conteneurs.

Dans notre application chaque application tourne et contenue dans un docker container, Ce qui permet une utilisation plus efficace des ressources du système, des cycles de livraison de logiciels plus rapides, mais surtout très efficace dans architecture micro-service telle que la nôtre.

6.2 Docker compose

Compose est un outil permettant de définir et d'exécuter des applications Docker multi-conteneurs. Avec Compose, on utilise un fichier YAML pour configurer les services de votre application. Ensuite, avec une seule commande, on crée et démarre tous les services à partir de notre configuration.

6.3 Kubernetes

minikube est un outil qui nous permet d'exécuter Kubernetes localement. minikube exécute un cluster Kubernetes à un seul nœud sur notre ordinateur. Dans notre cas , comme on a utilisé minikube , tous nos conteneurs sont orchestrés par un seul node.

7 Conclusion

Ce projet nous a initié aux notions Docker ainsi que Kubernetes d'ou Docker aide à "créer" des conteneurs, et Kubernetes permet de les "gérer" au moment de l'exécution. Utiliser Docker pour emballer et expédier l'application et utiliser Kubernetes pour déployer et mettre à l'échelle l'application. Utilisés ensemble, Docker et Kubernetes servent de facilitateurs de la transformation numérique et d'outils pour une architecture cloud moderne.