

16/01/2019

Léo Guilpain

TD 3

Galerie de photos

*« J’atteste que ce travail est original, qu’il indique de façon appropriée tous les emprunts, et qu’il fait référence de façon appropriée à chaque source utilisée »*

Table des matières

[Partie I - "Hello World" Node.js + Express avec Docker 2](#_Toc535441429)

[Création du "Hello World" 2](#_Toc535441430)

[Question n° 1 et 2 : 2](#_Toc535441431)

[Question n°3 : 2](#_Toc535441432)

[Conteneurisation de notre app 3](#_Toc535441433)

[Question n° 1 3](#_Toc535441434)

[Question n°2 3](#_Toc535441435)

[Question n °3 4](#_Toc535441436)

[Partie II - Création d'une galerie d'images statique 5](#_Toc535441437)

Partie I - "Hello World" Node.js + Express avec Docker

Création du "Hello World"

Question n° 1 et 2 :

Après avoir initialisé le projet, nous pouvons tester sur l’adresse « http://localhost:3000 » . On obtient ceci :

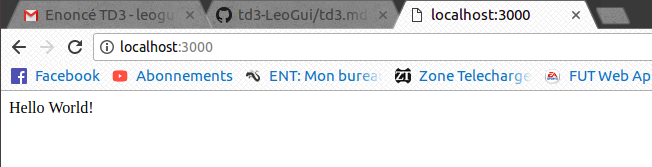
****

Figure : Hello World

Question n°3 :

On modifie le package-json comme ceci :

****

Figure : package-json

Lorsque l’on lance la commande « npm start », cela signifie que l’on fait « node index.js ».

Conteneurisation de notre app

Question n° 1

La taille de l’image est de 674 MB. En faisant « docker pull node » on récupère la dernière image qui est de 898 MB.

Question n°2

Pour générer le DockerFile, je me suis aidé de ce site : <https://nodejs.org/en/docs/guides/nodejs-docker-webapp/>

On obtient donc ceci pour le Dockerfile :

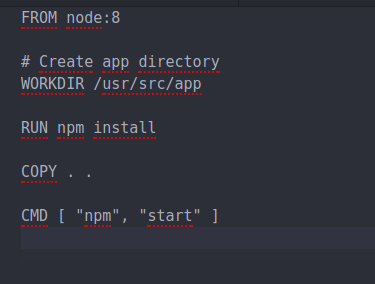


Figure : Dockerfile

On exécute ensuite la commande suivante :

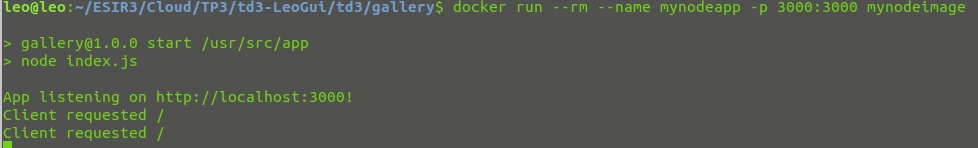


Figure : docker run

En exécutant la commande “docker images”, on voit bien que l’image a été créée.

https://lh3.googleusercontent.com/wupVFAEq9_nGrtXIzLjjVh3iFfggrbbIxB91pyivijOnyR-138hCChKTJ0NruSThF3G3K_k4cLQ03tgY3VkeVtzM-Z2ebiS7B-WEGdWh9_JJGQB838YORCuoC5LnQRtWurgMb1I

Figure : docker images

Question n°3

Pour que les modules soient installés lors du build de l'image Docker il faut rajouter ceci dans le fichier DockerFile :

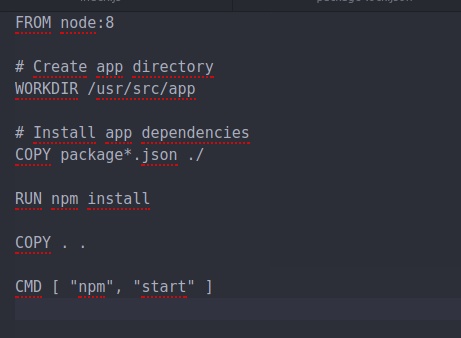


Figure : Dockerfile

Je me suis servi du site suivant :

<https://www.sitepoint.com/create-new-express-js-apps-with-express-generator/>

Partie II - Création d'une galerie d'images statique

La difficulté de cette partie a été de rendre dynamique l’affichage des albums et des photos

J’ai utilisé l’arborescence des images pour gérer leur titre ainsi que les albums. J’obtiens donc ceci :

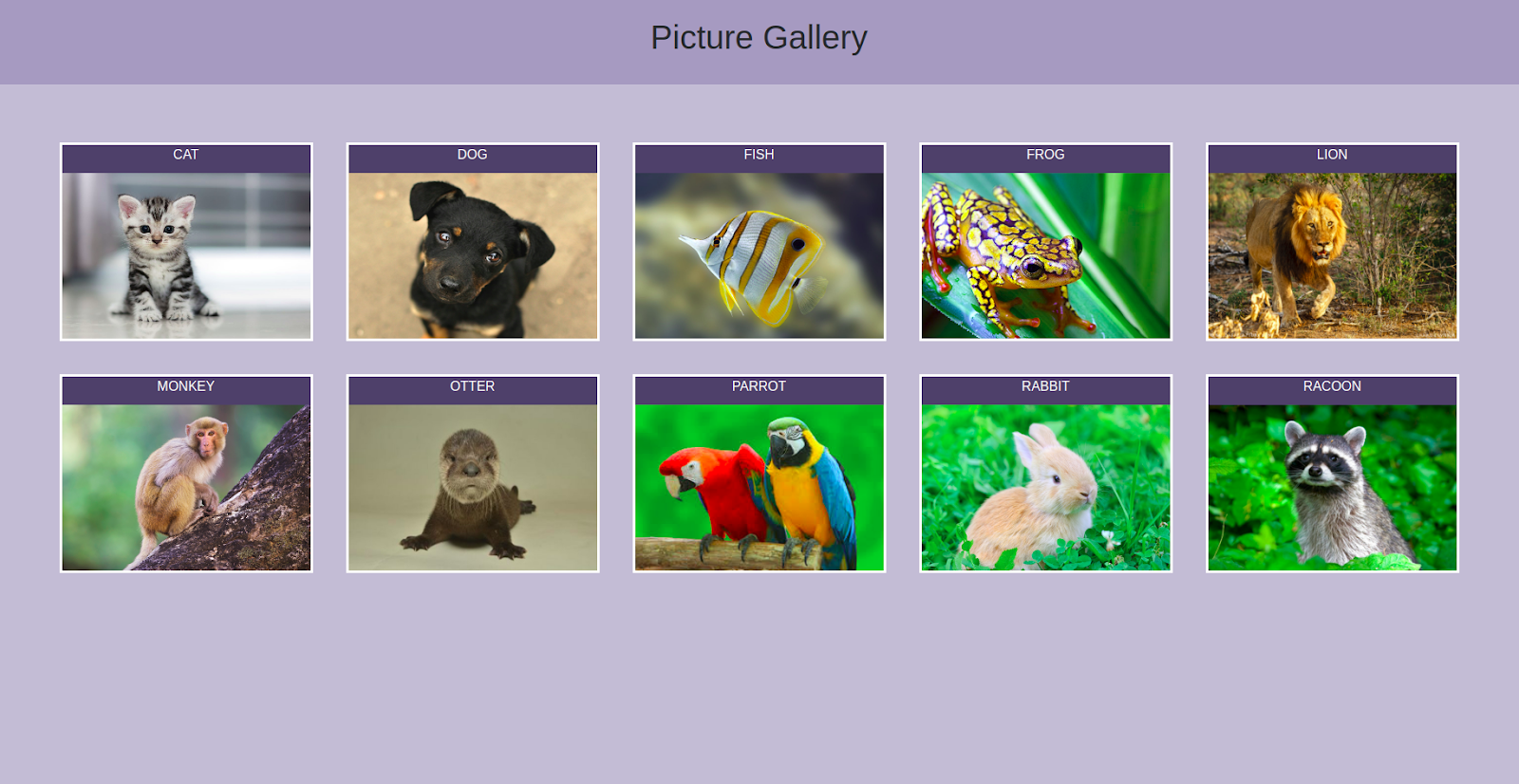


Figure : page d'accueil / index.js

Ici on peut voir 4 albums possédant une miniature. Si on ajoute ou supprime un album cela l’ajoutera ou le supprimera automatiquement sur cette page.

Ensuite lorsque l’on clique sur une des miniatures, on arrive sur la page correspondant à l’album (celle-ci est dynamique et s’adapte aux photos de l’album). Un carrousel se trouve sur cette page permettant de faire défiler les photos.



Figure : Carrousel

Ensuite pour déployer la galerie avec Docker, j’ai créé un Dockerfile dans mon projet. Je l’ai ensuite complété comme ceci :

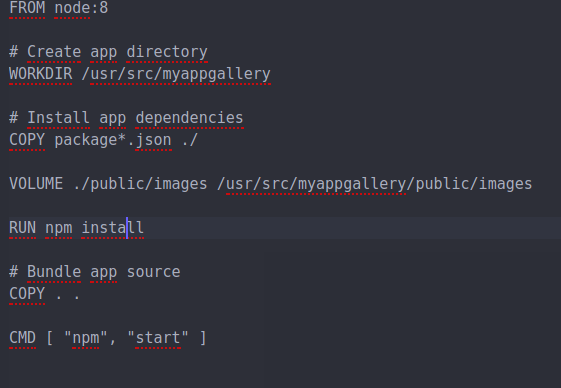


Figure : Dockerfile

Après avoir build, on run l’image avec la commande suivante :

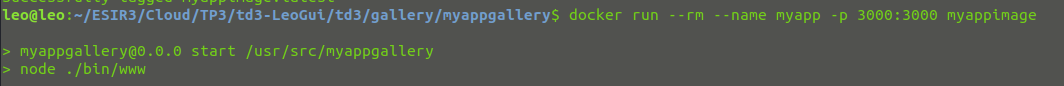


Figure : docker run