TOLERANCIA A FALLAS

DOCKER

Computación Tolerante a Fallas

Murillo Cortes, Jeanette.

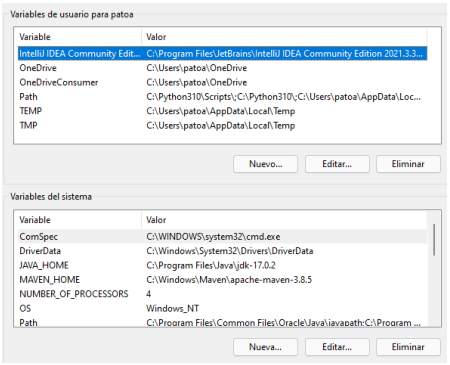
Computación Tolerante a Fallas
Dr. Michel Emanuel López Franco

Sección D06
Calendario 2022A

Lunes 21 de Marzo, 2022
Guadalajara, Jalisco.

Docker

PREINSTALACIONES



Para el desarrollo de esta práctica, se hizo uso de:

- Docker, el cuál se puede descargar desde el siguiente url: https://www.docker.com/products/docker-desktop/
  Y que se agregó a las rutas del sistema para poder trabajar con él. El desarrollo de su instalación se encuentra en el archivo 'Instalar Docker.txt'.

DESARROLLO DE LA PRÁCTICA

Para recordar, Docker permite que las aplicaciones se puedan desarrollar, ejecutar y desplegar mediante contenedores de software.
Para crear uno, se deben compilar una serie de comandos que crean una imagen local o remotamente. Las imágenes se generan en un archivo llamado Dockerfile, el cuál contiene instrucciones. Algunas de ellas son como las que se muestran en la imagen, que fueron utilizadas para la realización del contenedor de esta práctica.

```
Dockerfile > FROM
1   FROM node:12-alpine3.12 AS build
2   WORKDIR C:\Users\patoa\Desktop\system
3   COPY package.json ./
4   RUN npm install
5   COPY . .
6   CMD npm start
7   RUN npm run build
```

En ella se puede observar que se especifican los siguientes datos:

IMAGEN DE DOCKER

- FROM: Especifica una imagen padre donde normalmente se escoge con las dependencias y librerías que se ocuparán para el trabajo. Para crear en este caso una imagen de Node, las distribuciones más comunes de Linux son Alpine3, búster y stretch, sin embargo, Alpine3 es la más comúnmente utilizada por su poco peso (5.57MB).
- WORKDIR: Especifica el directorio donde se va a crear, y si no existe, se crea.
- COPY. . : Copea todos los archivos que hay dentro del directorio
- RUN: Corre el código
- CMD: Pasa mediante argumentos el comando que va a correr

Ahora, para la creación de la imagen y contenedor, se usan algunos de los siguientes comandos, los cuáles son básicos para el uso de Docker:
Para construir la imagen se utiliza el comando:

- "`docker build . -t` *`nombreimagen`*"

Para correr el contenedor, se utiliza:

- "`Docker run` *`nombreimagen`*"

Y para especificar el puerto en el que se está escuchando (de manera local):

- “`Docker run -p 3000:3000 nombreimagen`”

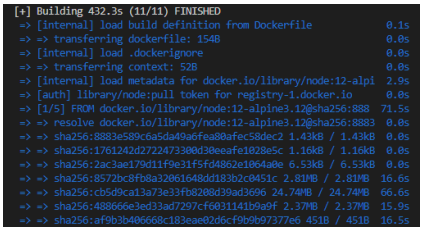
Aunque también se puede correr desde background con -d o -dp

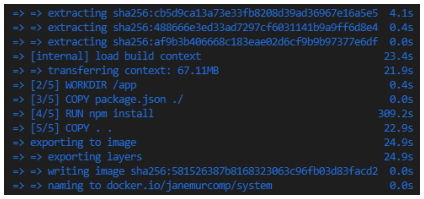
Para subirlo a internet, se utiliza el comando

- “`Docker tag imagenid cuentadocker/nombreimagen`”

Subir la imagen

- “`Docker push cuentadocker/nombreimagen`”





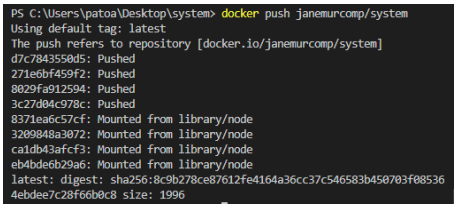
De manera, para la práctica primeramente, se utilizó el comando de Docker para la creación de la imagen directamente a internet, en este caso con se utilizó con mi cuenta, por lo que se utilizó el siguiente comando:

- “`docker build -t janemurcomp/system .`”

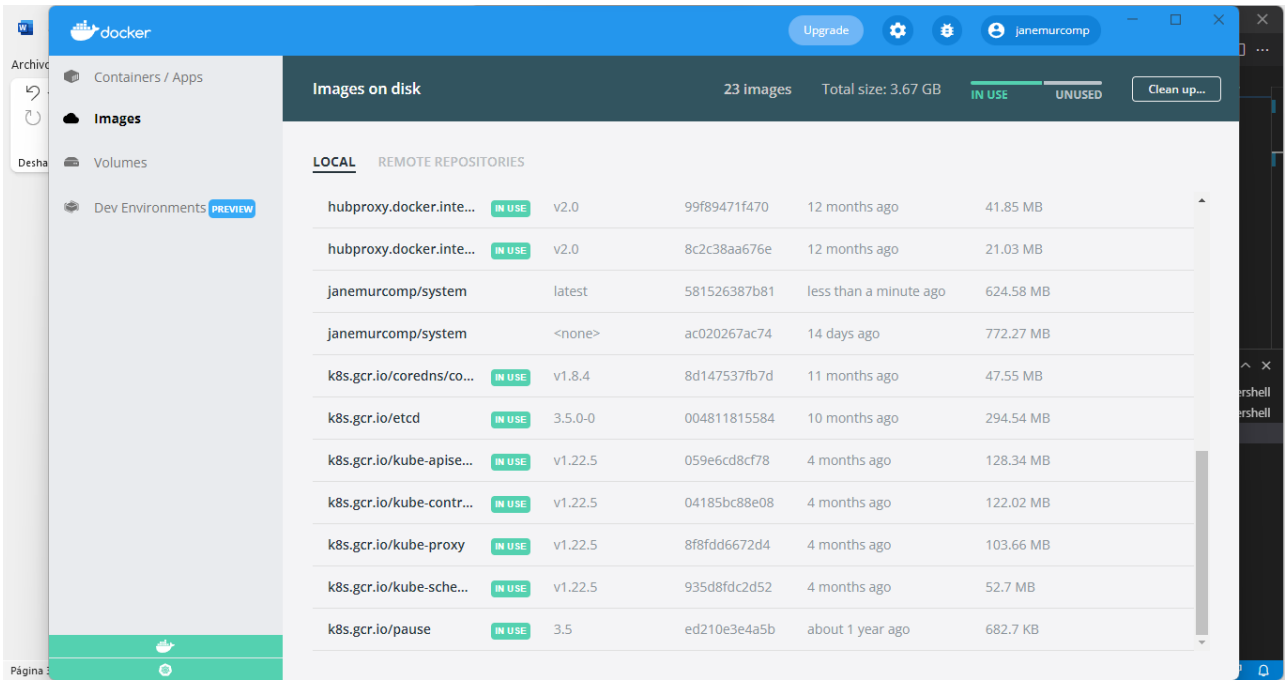
En las imágenes de la izquierda se puede observar cómo se creó la imagen desde la terminal de la consola de comandos.

Luego, se publicó en Docker mediante el comando:

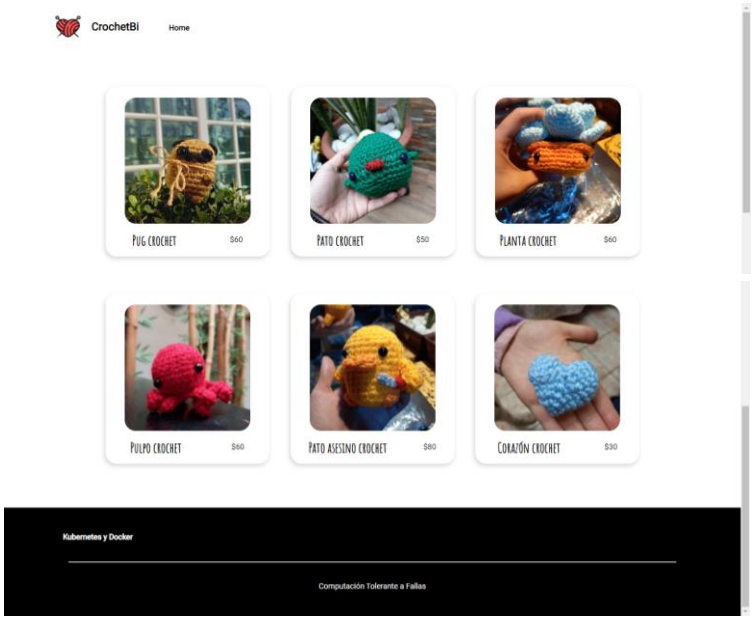
- “`docker push janemurcomp/system`”

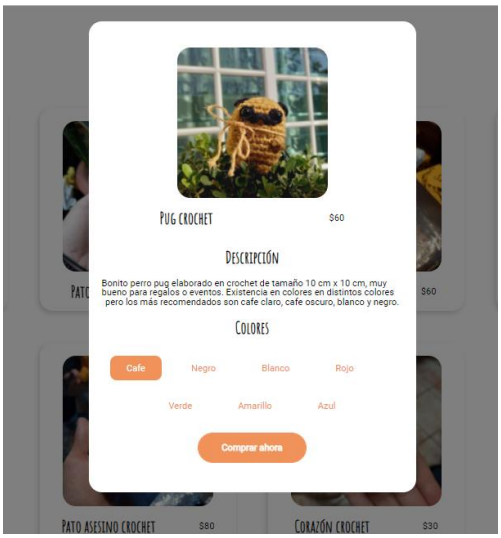


De esta manera podemos observar que el Docker se ha creado correctamente:



El Docker publicado, contiene una aplicación para la muestra de productos de un pequeño negocio de piezas de crochet, el cuál se muestra en las siguientes imágenes:





CONCLUSIONES

Docker es una aportación muy importante porque gracias a ello nos permite correr nuestras aplicaciones de una manera más sencilla y conterinizada con mejor eficiencia desde distintas máquinas, por lo que conocer acerca de su uso nos permite hacer mejores desarrollos de producción de programas que pueden ser utilizados para futuro, gracias a la gran ventaja de Docker de poder tener todo lo necesario para que nuestras aplicaciones funcionen desde un solo contenedor.

BIBLIOGRAFIA

- Deploy a react app in Docker – YouTube. Recuperado del URL: https://www.youtube.com/watch?v=JL6RTbtS9p0&list=PLeOcwkjDsoExvynzi_TvC69gnXyzZ9M--&index=3&t=37s