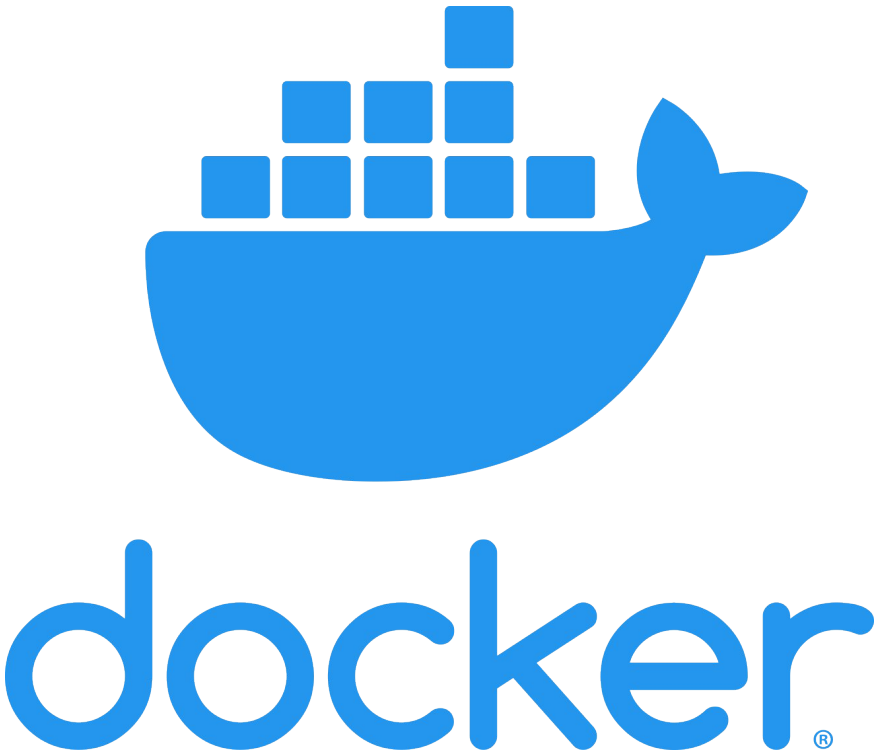
Universidad de San Carlos de Guatemala
Faculta de Ingenieria
Practicas Iniciales F-

Manual de Docker



Integrantes:

Jennifer Yulissa Lourdes Taperio Manuel – 202103763
Harold Benjamin Oxlaj Mangandi – 202100543
Luis Fernando Gomez Rendon – 201801391
Adam Jose Miguel Navas Garcia – 201213547
Roberto Adolfo Ramirez Cos – 201900999

Indice

Que es Docker

Docker es una plataforma de software que le permite crear, probar e implementar aplicaciones rápidamente. Docker empaqueta software en unidades estandarizadas llamadas contenedores que incluyen todo lo necesario para que el software se ejecute, incluidas bibliotecas, herramientas de sistema, código y tiempo de ejecución. Con Docker, puede implementar y ajustar la escala de aplicaciones rápidamente en cualquier entorno con la certeza de saber que su código se ejecutará.

Como funciona:

Docker le proporciona una manera estándar de ejecutar su código. Docker es un sistema operativo para contenedores. De manera similar a cómo una máquina virtual virtualiza (elimina la necesidad de administrar directamente) el hardware del servidor, los contenedores virtualizan el sistema operativo de un servidor. Docker se instala en cada servidor y proporciona comandos sencillos que puede utilizar para crear, iniciar o detener contenedores.

Instalación de Docker

Antes de poder instalar docker correctamente, debemos verificar que todos los paquetes de ubuntu estén actualizados correctamente, procedemos a abrir una terminal y escribimos:

```
~$ sudo apt update
```

Luego de haber actualizado los paquetes, procedemos a instalar algunos paquetes de requisitos previos que permitan a apt usar paquetes a través de HTTPS, en la terminal escribimos:

```
:~$ sudo apt install apt-transport-https ca-certificates curl software-properties-common
```

Luego debemos añadir la clave de GPG para el repositorio oficial de Docker en su sistema, escribimos en la terminal:

```
~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

Ya que hemos añadido las claves, debemos agregar el repositorio de Docker a las fuentes de APT:

```
~$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu focal stable"
```

Ya que hemos agregado las claves y el repositorio de docke, procedemos a actualizar todos los paquetes utilizando:

```
~$ sudo apt update
```

Procedemos a verificar que tengamos seleccionado el repositorio predeterminado de docker para ver la versión que instalaremos en Ubuntu para mayor facilidad, para esto utilizamos:

```
~$ apt-cache policy docker-ce
```

Al usar el comando nos aparecerá lo siguiente:

```
docker-ce:
  Instalados: (ninguno)
  Candidato:  5:23.0.2-1~ubuntu.20.04~focal
  Tabla de versión:
     5:23.0.2-1~ubuntu.20.04~focal 500
        500 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages
     5:23.0.1-1~ubuntu.20.04~focal 500
        500 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages
     5:23.0.0-1~ubuntu.20.04~focal 500
        500 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages
     5:20.10.23~3-0~ubuntu-focal 500
        500 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages
     5:20.10.22~3-0~ubuntu-focal 500
        500 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages
     5:20.10.21~3-0~ubuntu-focal 500
        500 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages
     5:20.10.20~3-0~ubuntu-focal 500
        500 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages
     5:20.10.19~3-0~ubuntu-focal 500
        500 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages
     5:20.10.18~3-0~ubuntu-focal 500
        500 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages
     5:20.10.17~3-0~ubuntu-focal 500
        500 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages
     5:20.10.16~3-0~ubuntu-focal 500
        500 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages
     5:20.10.15~3-0~ubuntu-focal 500
        500 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages
```

Ahí nos mostrara una tabla con todas las versiones disponibles de docker por si se desea utilizar alguna versión en especifico, la versión que se recomienda en candidato y el estado de instalados (en este caso dice ninguno) para que verifiquemos que versión se encuentra actualmente en nuestro ubuntu

Para este caso instalaremos la versión candidato, para ello escribimos el comando:

```
~$ sudo apt install docker-ce
```

Luego de haber instalado docker, docker se iniciara automáticamente, debemos verificar su estado para ver si funciona bien, para ello utilizamos:

```
~$ sudo systemctl status docker
```

Al utilizar el comando, nos aparecerá una ventana:

En esta ventana podremos ver que docker se encuentra funcional y activo actualmente, para salir del mensaje oprimimos CTRL + C

DockerFile

Dockerfile es un archivo de texto plano que contiene instrucciones para construir una imagen de Docker. El archivo especifica las capas de la imagen, que incluyen desde la base del sistema operativo hasta las dependencias y configuraciones específicas de la aplicación.

A este conjunto de instrucciones se le conoce como línea de comandos y serán los encargados de indicar los pasos a seguir para el ensamblaje de una imagen en Docker, es decir, los elementos necesarios para el desarrollo de un contenedor en Docker.

De manera que las imágenes en Dockerfile se crean a partir de un comando en específico denominado **docker build**, que se encargará de ofrecer las herramientas para que el sistema siga las instrucciones que el usuario haya indicado en la línea de comandos.

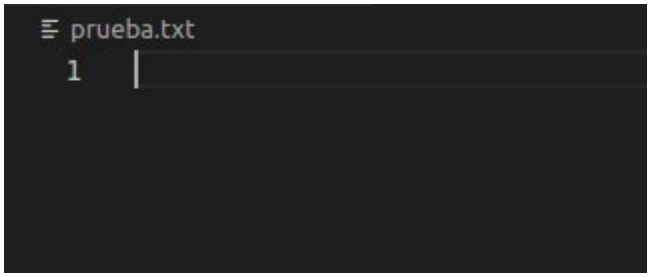
Dentro de las opciones más relevantes para un Dockerfile se encuentran herramientas encargadas de labores, como el establecimiento de la imagen base, el cambio de usuario o los elementos preestablecidos para el arranque de un contenedor en Docker, entre otros.

Algunas de estas opciones son:

- **FROM:** es una opción de Dockerfile que debe presentarse como la primera instrucción. Cumple con la función de establecer la imagen sobre la que los pasos e imágenes siguientes se desarrollan en el sistema. La imagen mínima que da origen al resto de imágenes en Docker es llamada scratch.
- **ENV:** hace referencia a la opción que indica las variables de entorno que se necesitan en el proceso de construcción de una imagen en Docker y permite la ejecución de los contenedores y sus labores en el sistema.
- **USER:** esta herramienta se utiliza en los archivos de instrucciones de Dockerfile con el objetivo de cambiar el usuario y su pertenencia a un grupo determinado. Una vez se ejecute esta opción, se aplicará a la totalidad de instrucciones siguientes.
- **RUN:** es una de las opciones de mayor importancia y popularidad en Dockerfile. Cumple la labor de ejecutar una instrucción incluida en la línea de comandos de la imagen durante su proceso de construcción. Dockerfile RUN puede escribirse en formato SHELL o bajo la opción de escritura EXEC.
- **ADD:** este elemento se encarga de las tareas relacionados con la copia de ficheros, directorios y archivos de una imagen en Dockerfile. Se debe tener en cuenta que el uso de la instrucción ADD implica la creación de una nueva capa de imagen, por lo que debes ser cuidadoso al implementar esta opción.
- **EXPOSE:** es la opción que tiene como labor la definición de las asignaciones referentes a los puertos para los contenedores de la plataforma que se encuentren en su etapa de ejecución.

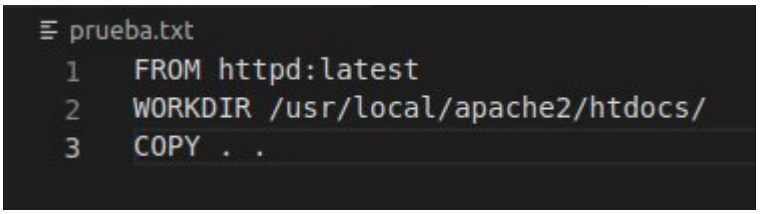
Para poder crear un Dockerfile para levantar aplicaciones hacemos lo siguiente:

Creamos un archivo de texto en un editor de texto como notepad o visual studio code

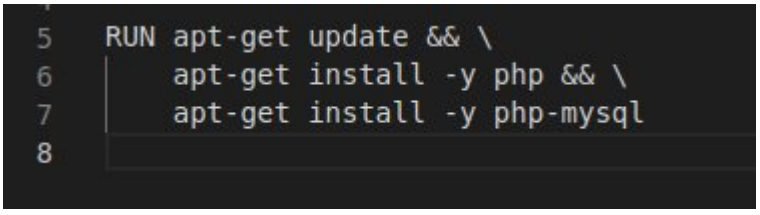


Luego escribimos en la primera linea, utilizamos el comando FROM, seguido de la imagen base que se utilizara para construir la imagen en docker, como ejemplo utilizaremos una imagen de un servidor web de apache, este se especificara de la siguiente manera:

Luego para definir el directorio donde se estara trabajando y copiar los archivos de la aplicacion a la imagen, debemos agregar lo siguiente:



Es necesario tambien especificar las dependencias y configuraciones que estara usando la aplicacion, por ejemplo, si nuestra aplicacion necesita PHP y diferentes extensiones para funcionar, se agregan las siguientes lineas para instalar las dependencias y extensiones necesarias



Tambien es necesario especificar el puerto donde se ejecutara la aplicacion, para este ejemplo utilizaremos el puerto 1000, agregamos la siguiente linea:

Por ultimo agregamos la linea CMD con el cual especificaremos al contenedor que comando se ejecutara cuando se inicialice

```
11   CMD ["php", "-S", "0.0.0.0:80", "index.php"]
```

Luego de que ya hemos escrito los cambios, se debe guardar el archivo con el nombre de “Dockerfile” en la raiz del directorio donde estara la aplicacion

Ya con el dockerfile creado, podemos construiir la imagen del docker utilizando el comando de “docker build” y luego ejecutamos el contenedor utilizando el comando “docker run”, en este caso, si la aplicacion se llama “prueba_pagina”, ejecutamos el siguiente comando para construir la imagen y ejecutar le contenedor:

```
hunterzxu@Adam-Ubuntu:~$ docker build -t prueba_pagina .
docker run -p 80:80 mi_aplicacion
```

Y listo, asi es como se utiliza un dockerfile para levantar y construir una aplicacion

DockerHub

Docker Hub es un registro público y privado de imágenes de Docker. Es un servicio en línea que permite a los usuarios almacenar, distribuir y compartir imágenes de Docker en un repositorio centralizado. La biblioteca de Docker Hub cuenta, además, con una amplia variedad de fuentes para sus imágenes, dentro de las que se incluyen grandes proveedores de software independientes, algunos proyectos de tipo open source que tienen el objetivo de desarrollar y distribuir su código en contenedores para estos repositorios y la comunidad de desarrolladores.

Docker Hub también destaca por ser el repositorio establecido por defecto o predeterminado de la plataforma de Docker Engine, debido a las capacidades y propiedades de este sistema.

El repositorio de docker hub cuenta con una serie de elementos que lo caracterizan y permiten su funcionamiento, dentro de los que se incluyen:

Imágenes oficiales

Docker Hub se caracteriza por contener una alta cantidad de imágenes oficiales y de buena calidad para contenedores, es decir, son las imágenes que la propia plataforma de Docker proporciona de forma directa.

Imágenes de proveedores

Docker Hub también permite la extracción y el uso de imágenes de contenedores proporcionadas por algunos proveedores externos al sistema. En el caso de las imágenes de estos proveedores que se encuentre certificadas, cuentan, además, con un soporte y una garantía que asegura su compatibilidad con la versión de Docker Enterprise.

Compilaciones automáticas

Otra de las características de docker hub es que contribuye a la creación de forma automática de imágenes de contenedores desde plataformas como GitHub y Bitbucker, para después enviarlas de manera directa al repositorio.

Repositorios públicos

Docker Hub se caracteriza también por ofrecerle a sus usuarios el acceso a una serie de repositorios gratuitos y públicos, donde pueden almacenar y compartir imágenes de contenedores.

Repositorios privados

Esta biblioteca también ofrece la posibilidad de unos repositorios de tipo privado que se encargan de almacenar imágenes de contenedores push y pull. Además de esto, los equipos de trabajo y organizaciones cuentan con la capacidad para gestionar el acceso a estos repositorios.. Para utilizar esta función, es necesario que el cliente se suscriba a la plataforma.

Para poder subir una imagen al Docker Hub:

Primero debemos registrarnos en Docker hub utilizando el link:

https://hub.docker.com/

Luego de que ya nos hayamos registrado, iniciamos sesion en docker hub, para ello podes hacerlo en al terminal utilizando el comando de **“docker login”**

```
~$ docker login --username=yourhubusername --email=youremail@company.com
```

Donde “yourhubusername” lo reemplazara con el nombre de usuario que utilizo para registrarse en docker hub y “youremail@company.com” es el correo electronico que se utilizo en el registro

Ya que hemos ingresado correctamente desde nuestra terminal, procedemos a crear una etiqueta para nuestra imagen utilizando el comando **“docker tag”** especificando nuestro nombre de usuario, y el nombre de la imagen

```
~$ docker tag myimage yourhubusername/myimage
```

Ahora procedemos a subir la imagen etiquetada al registro de Docker Hub utilizando el comando **“docker push”**

```
~$ docker push yourhubusername/myimage
```

Y listo, de esta manera ya habremos subido una imagen al Docker Hub.

Para poder Obtener imagenes de Docker Hub:

Para obtener imagenes a Docker Hub, primero procedemos a buscar la imagen que queramos, para ello utilizamos el comando **“docker search”** junto con el nombre de la imagen que queremos

```
~$ docker search imagename
```

Ya que hayamos encontrado la imagen que necesitamos, procedemos a descargarla para poder utilizarla, para ello utilizamos el comando “**docker pull”**, especificando el nombre de nuestro usuario de docker hub y el nombre de la imagen que necesitamos

```
~$ docker pull yourhubusername/imagename
```

Ya que hayamos descargado la imagen que queriamos, podemos ejecutar un contenedor utilizando la imagen que hayamos descargado utilizando el comando **“docker run”**

```
~$ docker run -it yourhubusername/imagename
```