**¿Que es Docker?**

Docker es una plataforma abierta para que desarrolladores y administradores de sistemas desarrollen, envíen y ejecuten aplicaciones distribuidas, ya sea en computadoras portátiles, maquinas virtuales de centros de datos o en la nube.



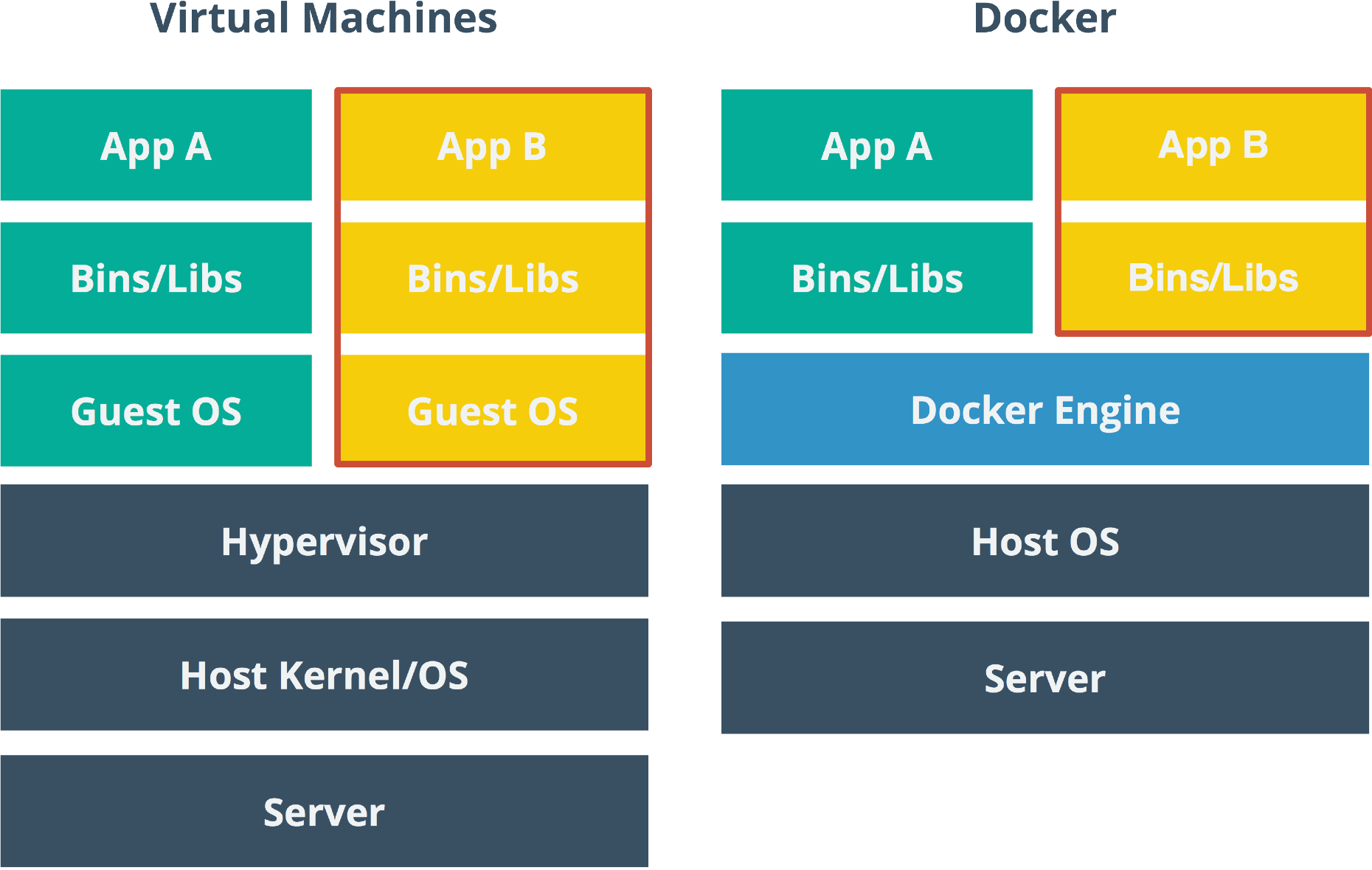
Docker empaqueta software en “contenedores” que incluyen en ellos todo lo necesario para que dicho software se ejecute, incluidas librerías. Con Docker se puede implementar y ajustar la escala de aplicaciones de una forma rápida en cualquier entorno con la garantía de que el código se ejecutará.

A primera vista se piensa en Docker como una especie de máquina virtual “liviana”, pero la verdad no lo es. En Docker no existe un hypervisor que virtualice hardware sobre el cual corra un sistema operativo completo. En Docker lo que se hace es usar las funcionalidades del Kernel para encapsular un sistema, de esta forma el proyecto que corre dentro de el no tendrá conocimiento que está en un contenedor. Los contenedores se encuentran aislados entre sí y se comportaran como máquinas independientes.

Iniciar un contenedor no tiene un gran impacto a diferencia de iniciar una máquina virtual ya que no tiene que iniciar un sistema operativo completo (desde cero). Gracias al uso de contenedores la demanda de recursos baja limitándose sólo al consumo de la aplicación que contenga. Un contenedor inicia en milisegundos.

Algunas de las características más notables de un contenedor son:

* Los contenedores son más livianos (ya que trabajan directamente sobre el Kernel) que las maquinas virtuales.
* No es necesario instalar un sistema operativo por contenedor.
* Menor uso de los recursos de la máquina.
* Mayor cantidad de contenedores por equipo físico.
* Mejor portabilidad.



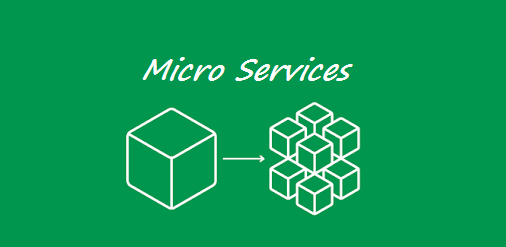
**¿Que es una Imagen Docker?**

Las imagenes Docker son plantillas (que incluyen una aplicación, los binarios y las librerias necesarias) que se utilizan para construir contenedores Docker y ejecutarlos (los contenedores ejecutarán una imagen previamente compilada).

**¿Por qué usar Docker?**

Docker le permite entregar código con mayor rapidez, estandarizar las operaciones de las aplicaciones, transferir el código con facilidad y ahorrar dinero al mejorar el uso de recursos. Con Docker, obtiene un solo objeto que se puede ejecutar de manera fiable en cualquier lugar. La sintaxis sencilla y simple de Docker le aporta un control absoluto. La amplia adopción significa que existe un gran ecosistema de herramientas y aplicaciones listas para su uso que puede utilizar con Docker.

**¿Que son los microservicios?**

 Los microservicios son un sistema de desarrollo software que propone su propia arquitectura. Mientras que en una arquitectura monolítica la aplicación es desarrollada como una única unidad, una arquitectura de microservicios funciona con un conjunto de pequeños servicios que se ejecutan de manera independiente y autónoma. Incluso cada uno de ellos puede estar en un lenguaje de programación diferente. Este tipo de servicios nos permite contar con infraestructuras IT más flexibles y adaptables, ya que para modificar un único servicio no es necesario alterar el resto de la infraestructura.

**Explicacion de la solucion Implementada:**

La solución consistió en la eliminación del EScaw Viejo, creando un sistema de reportes bajo la arquitectura de microservicios, el cual trabaja de manera autónoma pero interactua con eScaw Nuevo sin ningún tipo de problema ni usando ningún tipo de sincronización ya que comparten el mismo origen de datos (SQL SERVER).

El equipo de desarrollo decidió por implementar una arquitectura de microservicios debido a las ventajas presentadas por esta entre ellas:

* Equipo de trabajo mínimo
* Escalabilidad
* Funcionalidad modular, módulos independientes.
* Libertad del desarrollador de desarrollar y desplegar servicios de forma independiente
* Uso de contenedores permitiendo el despliegue y el desarrollo de la aplicación rápidamente

Imagenes docker:

1. **Imagen mssql para entorno de desarrollo:**

Se creó un repositorio en gitlab <https://gitlab.com/sistemaspolybarq/mssql-polybarq> para los archivos de la imagen y para el uso del registry de imagenes docker de gitlab.

El dockerfile esta hecho a partir de la imagen <https://hub.docker.com/_/microsoft-mssql-server> ,se usó la version **mcr.microsoft.com/mssql/server:2019-CTP3.0-ubuntu** se configuro la contraseña en el dockerfile através de las variables de entorno especificadas para la imagen base, luego se hace un restore de la base de datos sql server <https://docs.microsoft.com/en-us/sql/linux/tutorial-restore-backup-in-sql-server-container?view=sql-server-ver15> para luego activar el sql server con /opt/mssql/bin/sqlservr

1. **Imagen Apache PHP Composer Reportico:**

Se creó un repositorio en gitlab <https://gitlab.com/sistemaspolybarq/reportico> para los archivos de la imagen y para en uso del registry de images docker de gitlab

Dentro de este repositorio se encuentra el dockerfile y el archivo de configuracion para el reportico

El dockerfile esta hecho a partir de la imagen <https://hub.docker.com/_/php> la version **php:7.3.3-apache**, dicha imagen esta basada en **debian**, al cual se le instalan las siguientes dependencias (zip unzip libpng-dev zlib1g-dev libbz2-dev curl apt-transport-https gnupg), luego se le instala ODBC Driver para SQL Server <https://docs.microsoft.com/en-us/sql/connect/odbc/linux-mac/installing-the-microsoft-odbc-driver-for-sql-server?view=sql-server-ver15> Luego se instalan las siguientes extensiones para php (pdo pdo\_mysql gd bz2 sqlsrv pdo\_sqlsrv) Luego se instala composer <https://getcomposer.org/download/> Luego instalamos reportico y copiamos el archivo de configuracion en la carpeta reportico/projects/admin/

1. **Imagen Para produccion:**

El dockerfile esta hecho a partir de la imagen registry.gitlab.com/sistemaspolybarq/reportico imagen a la que se le copian los archivos de reportico, este dockerfile esta ubicado en el repositorio de desarrollo https://gitlab.com/sistemaspolybarq/reportico-polybarq.git

**Docker-Compose para entorno de Desarrollo:**

Se creo un archivo docker-compose.yml en el cual se declaran 2 servicios,

el primer servicio cuya imagen es **registry.gitlab.com/sistemaspolybarq/reportico** tiene expuesto el puerto 8000 el cual vincula con el puerto 80 del contenedor, tambien se especifica un volumen para el desarrollo reactivo y poder ver reflejado de inmediato lo que se modifica, en el contenedor, dicho volumen vincula la carpeta **./reportico** de la carpeta de desarrollo, con la carpeta **/var/www/html/reportico** dentro del contenedor; Del mismo modo se emplean diferentes variables de entorno para la configuracion de conexiones a las Bases de Datos (HOST, USER, DB, PASSWORD) especificadas en el archivo config.php de cada proyecto reportico (polybarq, cordeleria, andisacos); De igual forma se establece un unlace entre el primer servicio y el segundo a traves de la propiedad links de docker-compose.

El segundo servicio cuya imagen es **registry.gitlab.com/sistemaspolybarq/mssql-polybarq** es una imagen de mssql server la cual contiene la base de datos de Andisacos, se le expone el puerto 1433 el cual vincula con el puerto 1433 del contenedor

**¿Que es git?**

Git es una herramienta que realiza una función del control de versiones de código de forma distribuida,

del cual destaca el no depender de un repositorio central, la rapidez y el trabajo en ramas.

**Repositorios GIT:**

1. **Repositorio MSSQL-Polybarq (**[https://gitlab.com/sistemaspolybarq/mssql-polybarq](https://gitlab.com/sistemaspolybarq/mssql-polybarq.git)**):**

Dentro de este repositorio se encuentra el dockerfile, el archivo de backup (andisacos.bak), y 2 archivos de scripts shell. Se hace uso del registry para la creacion de la imagen registry.gitlab.com/sistemaspolybarq/mssql-polybarq

1. **Repositorio Reportico (**https://gitlab.com/sistemaspolybarq/reportico**):**

Dentro de este repositorio se encuentra el dockerfile y el archivo de configuracion de reportico. Se hace uso del registry para la creacion de la imagen registry.gitlab.com/sistemaspolybarq/reportico

1. **Repositorio Reportico-Polybarq (**[https://gitlab.com/sistemaspolybarq/reportico-polybarq](https://gitlab.com/sistemaspolybarq/reportico-polybarq.git)**):**

Este es el principal repositorio de desarrollo, en este se encuentran el dockerfile para la imagen de produccion, el docker-compose.yml para el entorno de desarrollo, la carpeta de reportico en desarrollo, la carpeta de documentacion y la carpeta de solucion excel. Se debe hacer uso del registry de gitlab para la utilizacion de la imagen de produccion