**Documentación del entorno de Jupyter**

**Introducción**

La imagen del contenedor creado para el entorno de laboratorio está basada en la oficial de Ubuntu, con una serie de herramientas instaladas y configuradas, además de modificaciones en el entorno que se listarán en este documento.

La imagen de Ubuntu utilizada está descargada del repositorio de dockerhub, con la etiqueta **latest**.

Los subtítulos con el prefijo **Archivo** en este documento hacen referencia a los archivos utilizados en la construcción de la imagen del laboratorio. En estas secciones se va a describir para qué es utilizado el archivo, su estructura y como modificarlo si se desea hacer un cambio.

**Archivo Dockerfile**

Es el archivo donde se lista la serie de instrucciones que darán paso a la construcción de la imagen del laboratorio.

Actualiza los repositorios y comienza con la instalación de las herramientas necesarias. A continuación se hará la lista de las herramientas importantes, las que no se mencionen aquí y sin embargo estén en el archivo dockerfile, es porque son únicamente dependencias y no es la intención de este documento hacer un detallado de ellas.

* R: Para poder instalar R en modo no interactivo se hacen configuraciones como la zona horaria, es por ello que estas instrucciones forman parte de las primeras líneas. Cabe mencionar que existe un archivo rjuypter.r que instala paquetes de R como dplyr, tidiverse, devtools y RJDBC, además de dejar a R disponible para Jupyter.
* CPU limit: Herramienta utilizada para limitar el CPU por usuario. CPU limit tiene una serie de utilerías para el manejor del poder de procesamiento, pero la que utilizamos para la imagen es por porcentaje, donde 100% equivale a 1 CPU completo. Si quisiéramos usar 5 CPUs entonces el porcentaje sería 500%, por ejemplo.
* Java11: Versión de java actualizada para funcionar correctamente con JupyterHub y todas sus herramientas. Podría revisarse a futuro si conviene cambiar a Java8 para mayor compatibilidad con el ecosistema Hadoop.
* Python: Con librerías como pandas, numpy, scikit-learn, conexión con hive y livy para conetarse a spark.
* Sistema de monitoreo de Jupyter: Agrega gráficas al entorno de Jupyter Lab para poder monitorear los recursos (CPU y RAM) que utiliza un usuario.
* Herramientas para la creación de certificado HTTPS autofirmado.

Este archivo por último copia algunos archivos de configuración y de cambios visuales que se irán detallando a lo largo de este documento.

**Archivo monitor.py**

Este programa de python es el encargado de monitorear la memoria por cada uno de los usuarios y matar el proceso en dado caso que supere la cantidad configurada. También limita el porcentaje de CPU permitido. Se ejecuta desde que se arranca el contenedor. Utiliza un archivo de configuración llamado **users.conf** ubicado en **/home/app/**. En el vendrás las especificaciones de porcentaje de CPU y memoria RAM que X usuario tendrá permitido utilizar.

Este archivo de configuración consta de 3 partes, el nombre del usuario, la cantidad de memoria RAM permitida y el porcentaje de CPU por usuario. Esto en una estructura separada por “|”. Al momento de agregar un usuario mediante **adduser.sh** automáticamente se agregarán las configuraciones para ese usuario en este archivo, pero si se desea modificar, la estructura es la siguiente:

**Usuario|memoria|cpu** en donde por ejemplo, si queremos que el usuario **lalo** tengacapacidad máxima de **32GB** y poder de procesamiento de 25 cores, quedaría de la siguiente forma:

**lalo|32G|2500**

La configuración de cada usuario deberá ir separada por salto de línea.

Por último cabe mencionar que este programa deja registros de los procesos realizados en **/home/app/mataproceso.log** donde vienen datos como el usuario, el proceso y si es que se excedió, entre otros

**Archivo login.html**

Contiene los cambios en el texto que viene por default del login.

**Archivo page.html**

Aquí vienen los cambios más visuales en el login del laboratorio, como la imagen (que está en un formato de base64), los colores y el agregado de un botón para el laboratorio de Hue.

**Archivo start\_container\_processes.sh**

Inicia los servicios necesarios para arrancar el contenedor. Es decir, monitor.py, jupyterhub (con certificado https) y SSH.

**Archivo adduser.sh**

Ejecuta un conjunto de comandos al momento de crear un usuario. Su forma de utilizarse es:

**bash adduser.sh nombreUsuario passwd GBram CPU**

Básicamente, con este script se agrega un usuario y se realizan las configuraciones asociadas para dicho usuario (límites y visualización de monitoreo de recursos).

Por ejemplo, si queremos crear un usuario **edaurdo** con contraseña **root123** con límite de memoria RAM de **5GB** y 5 procesadores, entonces tendríamos que usar el comando:

**bash adduser.sh lalo root123 5G 500**

Por último, cabe mencionar que en este script se pueden agregar configuraciones necesarias para realizarse al momento de que se cree un usuario nuevo (como una instalación de paquete por usuario, por ejemplo).