**Documentación**

**AGESIC**

**VERSIONADO DEL DOCUMENTO**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Fecha | Versión | | Autor | Descripción |
| 10/09/2021 | 1.0 |  | Ivan Viamonte | Solicitud 37 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**INDICE**

[Resumen 4](#_Toc82012004)

[Ambiente de pruebas utilizado 5](#_Toc82012005)

[Instalando Docker 6](#_Toc82012006)

[\*\*Importante: 6](#_Toc82012007)

[Docker principales comandos 7](#_Toc82012008)

[Analizando el conector SPIKE en WildFly 11 8](#_Toc82012009)

[Nota 8](#_Toc82012010)

[Migración de WildFly 11 a WildFly 24 9](#_Toc82012011)

[Copia de archivos 9](#_Toc82012012)

[Modificación de archivos 9](#_Toc82012013)

[Creando Dockerfile 12](#_Toc82012014)

[Contenido del Dockerfile 12](#_Toc82012015)

[Creando la imagen 14](#_Toc82012016)

[Ejecución del contenedor 14](#_Toc82012017)

[Validación 14](#_Toc82012018)

[Solicitudes a Soporte Agesic: 15](#_Toc82012019)

[Referencias 16](#_Toc82012020)

[Adjuntos a la entrega 17](#_Toc82012021)

## Resumen

El objeto de esta solicitud fue pasar a **Docker** el conector **SPIKE** en su más reciente versión.

Este documento describe las siguientes tareas:

* Instalación de **Docker**.
* Análisis del conector **SPIKE en Wildfly**.
* Migración **WildFly 11** a **WildFly 24.**
* Armando el **Dockerfile**.
* Creando la imagen.
* Realizando el deploy.

Tenga en cuenta que este documento fue elaborado para un servidor con los siguientes requisitos:

**SO:** CentOS 8.3.2011

**CPU:** 4

**RAM:** 8 GB

**HDD:** 20 GB

**Docker**: 20.10.8

## Ambiente de pruebas utilizado

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **IPs** | **Puertos** | **Dominio** | **Usuario** | **Clave** |

## Instalando Docker

Procederemos a instalar **Docker** en un **CentOS 8** para poder realizar las tareas de contenerización del conector **SPIKE**.

* **Agregando el repositorio**:
  + *sudo dnf config-manager --add-repo=https://download.docker.com/linux/centos/docker-ce.repo*
* **Instalando Docker**:
  + *sudo dnf install docker-ce --nobest*
* **Habilitando Docker**:
  + *sudo systemctl enable --now docker*
* **Verificando estado**:
  + *systemctl status docker*
* **Agregando usuario actual al grupo docker**:
  + *sudo usermod -aG docker $USER*
* **Verificando grupos del usuario**:
  + *id $USER*
* **\*\*Deshabilitando Firewall del CentOS**:
  + *sudo systemctl disable firewalld*

### \*\*Importante:

No se recomienda deshabilitar las funciones de seguridad críticas. Sin embargo, en **CentOS 8** el firewall evita la resolución de **DNS** dentro de los contenedores de **Docker**.

## Docker principales comandos

A continuación, veremos brevemente los principales comandos usados en la ejecución de las tareas.

* **Compilar imagen:**
  + *docker build -t <etiqueta> .*
* **Iniciar imagen:**
  + *docker run -p <puerto local:puerto externo> <nombre imagen:version>*
* **Listar imágenes locales:**
  + *docker images -a*
* **Eliminar una imagen:**
  + *docker rmi -f <image-id>*
* **Eliminar todas las imagenes:**
  + *docker rmi -f $(docker images -q)*
* **Eliminar imagenes rotas:**
  + *docker image prune*
* **Listar contenedores:**
  + *docker container ls -a*
* **Detener contenedor:**
  + *docker stop <container-id>*
* **Detener todos los contenedores:**
  + *docker container stop $(docker container ls -aq)*
* **Accediendo al contenedor:**
  + *docker container exec -it <container-id> /bin/bash*
* **Eliminar contenedores detenidos:**
  + *docker container prune*
* **Eliminar todos los contenedores:**
  + *docker container rm $(docker container ls -aq)*

## Analizando el conector SPIKE en WildFly 11

Previo a contenerizar el conector fue necesario realizar un profundo análisis con el objetivo de comprender las configuraciones existentes en **WildFly 11** y con este conocimiento poder realizar la migración a **WildFly 24**. Durante este análisis detectamos que la documentación existente no contemplaba la totalidad de las configuraciones a migrar, lo que requirió una revisión detallada de todos los archivos de configuración para poder determinar todas las configuraciones necesarias para tener en cuenta durante la migración. Para llevar a cabo estas tareas se utilizo una instalación fresca de **WildFly 24** en un **CentOS 8**, donde se fueron aplicando distintas configuraciones hasta lograr la formula correcta que podía levantar el conector en su totaliad.

#### Nota

Todas las pruebas realizadas fueron sobre una instalación limpia de **CentOS 8** en su última versión 8.3-2011.

## Migración de WildFly 11 a WildFly 24

A continuación, describiremos paso a paso las tareas necesarias para migrar correctamente el conector desde la versión 11 de **WildFly** a la 24. El directorio raíz del **WildFly 11** a migrar le llamaremos **wildfly11-path** y de igual forma al directorio raíz del nuevo **WildFly 24** le llamaremos **wildfly24-path**.

### Copia de archivos

Agregar lo siguiente al nuevo servidor manteniendo los mismos path del servidor anterior:

1. Copiar los siguientes archivos desde *wildfly11-path\standalone\configuration*:
   1. *connector.properties*
   2. *connector-web.properties*
   3. *agesic\_v2.0.keystore*
   4. *pge-client.properties*
2. Copiar el directorio *wildfly11-path\standalone\connector.*
3. Copiar desde *standalone\deployments*:
   1. *connector-runtime.war*
   2. *connector-web.war*
4. Copiar el directorio *wildfly11-path\derby.*
5. Copiar el directorio *wildfly11-path\modules\org*

### Modificación de archivos

Se deberán modificar los siguientes archivos con el contenido descrito a continuación:

* ***wildfly24-path\bin\standalone.conf:***
  + Reemplazar la linea 38 por:
    - JAVA\_HOME=”***jdk-13-path***”
  + *Reemplazar la linea 76 por:*
    - JBOSS\_HOME="***wildfly24-path***"
    - JAVA\_OPTS="$JAVA\_OPTS -Dfile.encoding=UTF-8 -Dderby.system.home=$JBOSS\_HOME/derby -DuploadFolder=$JBOSS\_HOME/standalone/connector -Dconnector.integration.configLocation=$JBOSS\_HOME/standalone/configuration -Dconnector.web.configLocation=$JBOSS\_HOME/standalone/configuration"
* ***wildfly24-path\standalone\configuration\standalone.xml*:**
  + **Agregar las siguientes System Properties**:

<system-properties>

<property name="derby.drda.startNetworkServer" value="true"/>

<property name="derby.drda.portNumber" value="1527"/>

<property name="derby.stream.error.file" value="${jboss.server.log.dir}/derby.log"/>

<property name="derby.infolog.append" value="true"/>

</system-properties>

* **Agregar el siguiente Security Realm**:

<security-realm name="SslRealm">

<server-identities>

<ssl>

<keystore path="${jboss.server.config.dir}/agesic\_v2.0.keystore" alias="testing.hg.red.uy" keystore-password="conector" />

</ssl>

</server-identities>

</security-realm>

* **Agregar el siguiente DataSource**:

<datasource jndi-name="java:jboss/jdbc/DerbyEmbeddedDS" pool-name="DerbyEmbeddedDS" enabled="true">

<connection-url>jdbc:derby:${jboss.server.base.dir}/connectorpge;create=true</connection-url>

<driver>derby-embedded</driver>

<security>

<user-name>user</user-name>

<password>pass</password>

</security>

</datasource>

* **Agregar el siguiente Driver:**

<driver name="derby-embedded" module="org.apache.derby">

<driver-class>org.apache.derby.jdbc.EmbeddedDriver</driver-class>

<xa-datasource-class>org.apache.derby.jdbc.EmbeddedXADataSource</xa-datasource-class>

<datasource-class>org.apache.derby.jdbc.EmbeddedDataSource</datasource-class>

</driver>

* **Sustituir los http-listeners por los siguientes:**

<http-listener name="default" socket-binding="http-testing" max-post-size="30000000" redirect-socket="https-testing"/>

<http-listener name="produccion" socket-binding="http-prod" max-post-size="30000000" redirect-socket="https-prod"/>

<https-listener name="default-ssl" socket-binding="https-testing" max-post-size="30000000" security-realm="SslRealm"/>

<https-listener name="produccion-ssl" socket-binding="https-prod" max-post-size="30000000" security-realm="SslRealm"/>

* **Sustituir los Socket Binding Group (http y https) por:**

<socket-binding name="http-testing" port="${jboss.http.port:9800}"/>

<socket-binding name="http-prod" port="${jboss.http.prod.port:9700}"/>

<socket-binding name="https-testing" port="${jboss.https.port:8553}"/>

<socket-binding name="https-prod" port="${jboss.https.prod.port:8443}"/>

## Creando Dockerfile

A continuación, veremos como fue armado el **Dockerfile** para la creación de la imagen. Como imagen base se procedió a utilizar **CentOS 8** con las siguientes consideraciones:

* Se actualiza la imagen al hacer el **build**.
* Se descarga el JDK 13 y se ubica en /opt**.**
* Se descarga el WildFly 24.0.1 y se ubica en /opt.
* Se asignan permisos de ejecución sobre los scripts del **WildFly 24**.
* Se exponen los puertos 8080, 9800 y 9700.
* Se copian los archivos necesarios para levantar el conector.
  + Es importante destacar que estos deben estar preparados previa y preferiblemente en un directorio al mismo nivel del **Dockerfile**. Junto con la entrega de este documento se entrega además dicho directorio.
* Se procede a dejar la instrucción de ejecución del **WildFly 24.**

#### Contenido del Dockerfile

FROM centos:centos8.3.2011

LABEL Author="Ivan Viamonte"

#Actualizar servidor

USER root

RUN dnf update -y && dnf upgrade -y && dnf clean all

#Java JDK-13

RUN cd /tmp

RUN curl -O https://download.java.net/java/GA/jdk13/5b8a42f3905b406298b72d750b6919f6/33/GPL/openjdk-13\_linux-x64\_bin.tar.gz

RUN tar xf openjdk-13\_linux-x64\_bin.tar.gz

RUN mv jdk-13 /opt/

#Variables de entorno

ENV JAVA\_HOME=/opt/jdk-13

ENV WILDFLY\_VERSION=24.0.1.Final

ENV JBOSS\_HOME=/opt/wildfly

#WildFly instalación y permisos

RUN curl -O https://download.jboss.org/wildfly/$WILDFLY\_VERSION/wildfly-$WILDFLY\_VERSION.tar.gz

RUN tar xf wildfly-$WILDFLY\_VERSION.tar.gz

RUN mv wildfly-$WILDFLY\_VERSION $JBOSS\_HOME

RUN rm -f wildfly-$WILDFLY\_VERSION.tar.gz

RUN chmod -R g+rw ${JBOSS\_HOME}

#Exponer puertos

EXPOSE 8080 9800 9700

#Copia de archivos de configuración

RUN cd $HOME

COPY confWildfly/bin/\* $JBOSS\_HOME/bin/

COPY confWildfly/conf/\* $JBOSS\_HOME/standalone/configuration/

COPY confWildfly/connector $JBOSS\_HOME/standalone/connector

COPY confWildfly/deployments/\* $JBOSS\_HOME/standalone/deployments/

COPY confWildfly/derby $JBOSS\_HOME/derby

COPY confWildfly/modules/org $JBOSS\_HOME/modules/org

#Ejecucion

CMD cd /opt/wildfly/bin/ && sh standalone.sh -b 0.0.0.0

## Creando la imagen

Una vez elaborado el **Dockerfile** estamos en condiciones de crear la imagen con los siguientes pasos:

1. Copiar la carpeta “**confWildfly**” al mismo nivel del archivo **Dockerfile**.
2. Hacer el build:
   1. *docker build -t conector:latest .*
3. Comprobar que la imagen fue creada:
   1. *docker images -a*

#### Ejecución del contenedor

1. Una vez compilada la imagen, ejecutarla con el siguiente comando:
   1. *docker run -p 9800:9800 -p 9700:9700 conector:latest*
2. Comprobar que el contenedor está corriendo:
   1. *docker container ls*

#### Validación

Siguiendo los pasos anteriores tendremos el conector funcionando en el puerto habitual 9800, lo cual podemos comprobar accediendo en el navegador web a:

* http://<ip-nodo-docker>:<puerto>/connector-web

Pudiendo observar la siguiente pantalla:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

## Solicitudes a Soporte Agesic:

En este apartado se documentan las solicitudes que fueron realizadas al soporte de Agesic para lograr las implementaciones detalladas y los involucrados en las mismas.

|  |  |  |  |
| --- | --- | --- | --- |
| **Solicitud** | **Descripción** | **Atendida por** | **Comentarios** |

## Referencias

A continuación, dejamos algunas referencias que fueron usadas para la elaboración del presente documento y las tareas descritas.

* Docker documentación: <https://docs.docker.com/>
* Docker imágenes: <https://hub.docker.com/>
* WildFly 24: <https://docs.wildfly.org/24/>
* CentOS 8: <https://docs.centos.org/en-US/8-docs/>

## Adjuntos a la entrega

Se proceden a entregar los siguientes archivos:

* Dockerfile
* Directorio **confWildfly:**
  + Directorio **bin:**
    - standalone.conf
  + Directorio **conf:**
    - agesic\_v2.0.keystore
    - connector-web.properties
    - connector.properties
    - pge-client.properties
    - standalone.xml
  + Directorio **connector.**
  + Directorio **deployments:**
    - connector-runtime.war
    - connector-web.war
  + Directorio **derby.**
  + Directorio **modules:**
    - Directorio **org.**