UNIVERSIDAD DE LA REPÚBLICA URUGUAY

FACULTAD DE INGENIERÍA

Manual de configuración de la estación de trabajo

Informe presentado por

Dara Leslie Silvera Martínez y Nicolás Cámera López

en cumplimiento parcial de los requerimientos para la graduación de la carrera de Ingeniería en Computación de Facultad de Ingeniería de la Universidad de la República

Supervisores

Federico Gómez Frois
Sylvia Rita da Rosa Zipitría

Montevideo, 30 de julio de 2023

Índice

Objetivo

La idea de este documento es mostrar cómo configurar una estación de trabajo desde cero para poder desarrollar la aplicación MateFun. Adicionalmente, para configurar el intérprete MateFun se recomienda leer el readme que se encuentra en el repositorio Matefun.

1. Configuración previa

Todos estos pasos fueron probados en una computadora con Windows y sobre la misma se empleó una máquina virtual (en el virtualizador VMWare) con sistema operativo Linux con distribución Fedora KDE Plasma 35 de 64 bits. En otras palabras, toda la configuración de la estación de trabajo se realizó sobre una máquina virtual.

Notar que para esta guía se requiere permisos de root para realizar varios de las instalaciones y configuraciones que se verán.

2. Requerimientos necesarios para empaquetar y desplegar la aplicación MateFun

A continuación, se va a exponer una lista de dependencias y programas que son necesarios para poder empaquetar el repositorio Frontend.

2.1. Instalar y configurar Java JDK 11

En la distribución de Fedora 38 que se usó para configurar la estación de trabajo ya venía instalado Java JDK 17.0.7. Para chequear la versión de java, ejecutar en la terminal:

```
java --version
```

Como la versión de Java JDK es incompatible con la versión de los plugins que usa el Backend, es necesario instalar Java 11

```
sudo dnf install java-11-openjdk-devel.x86_64
```

Cambiar la versión de Java seleccionada a la **11**, usando el siguiente comando

```
sudo alternatives --config java
```

Una vez que tenemos instalado Java JDK, hay que configurar la variable de entorno `JAVA_HOME`. Dependiendo del shell que se está usando (por ejemplo, Bash o Zsh), el archivo puede ser `~/.bashrc`, `~/.bash_profile`, `~/.zshrc`, o algún otro archivo similar. En el caso de test, se editó el archivo `~/.bashrc`

```
sudo nano ~/.bashrc
```

En dicho archivo se agregó el export de la variable con formato export `JAVA_HOME=/path/to/java`

```
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-11.0.19.0.7-1.fc38.x86_64
```

Para guardar los cambios en Nano, se apreta **Ctrl + O**, luego **Enter** para confirmar y finalmente para cerrar Nano con **Ctrl + X**.

Para que los cambios surtan efecto en la sesión actual, hay que ejecutar

```
source ~/.bashrc
```

Referencia: https://docs.fedoraproject.org/es/quick-docs/installing-java/

2.2. Instalar Maven

Antes de empezar, asegurarse de que la distribución de Fedora está actualizada con lo último. Este proceso de actualización puede tardar mucho, dependiendo de la velocidad de conexión y de la capacidad de los repositorios para proveer las dependencias a instalar.

```
sudo dnf upgrade --refresh -y
```

Instalar Maven:

```
sudo dnf install maven
```

Modificar la versión de Java que usa Maven en el siguiente archivo

```
sudo nano ~/.mavenrc
```

En dicho archivo, exportar la variable `JAVA_HOME`

```
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-11.0.19.0.7-1.fc38.x86_64
```

Confirmar la instalación y la versión de Maven (debería decir que está usando la versión de Java 11.0.X):

```
mvn -version
```

Referencia:
https://www.linuxcapable.com/how-to-install-apache-maven-on-fedora-linux/

2.3. Instalar MySQL

Ejecutar el comando

```
sudo dnf install community-mysql-server -y
```

Si da problemas con las dependencias, probar agregar alguna de las flags **`--allowerasing`** o **`--skip-broken`** al comando anterior.

Iniciar el servicio de MySQL

```
sudo systemctl start mysqld
```

Comprobar el estado del servicio

```
systemctl status mysqld
```

[Opcional] Iniciar el servicio de MySQL en cada boot del sistema operativo

```
sudo systemctl enable mysqld
```

Establecer las normativas de seguridad de MySQL y crear una contraseña para acceder al DBMS

```
sudo mysql_secure_installation
```

Referencia: https://www.tecmint.com/install-mysql-fedora-linux/

2.4. Instalar Wildfly

Descargar la versión **10.1.0** de Wildfly

```
wget http://download.jboss.org/wildfly/10.1.0.Final/wildfly-10.1.0.Final.zip
```

Crear el directorio `/data` y copiar el zip del Wildfly al mismo

```
sudo mkdir /data
sudo cp wildfly-10.1.0.Final.zip /data
cd /data
```

Extraer el comprimido del Wildfly en el directorio `/data`

```
sudo unzip wildfly-10.1.0.Final.zip
```

[Opcional] En la referencia se explica como cambiar la IP a la que está enlazada el Wildfly, para así permitir acceder al servidor desde cualquier lugar de la LAN.

Agregar un nuevo usuario administrador (**Management user**) ejecutando el siguiente script

```
sudo sh /data/wildfly-10.1.0.Final/bin/add-user.sh
```

Finalmente, para iniciar la instancia de Wildfly ejecutar

```
sudo sh /data/wildfly-10.1.0.Final/bin/standalone.sh
```

Para acceder a la instancia de Wildfly desde el navegador, ir hasta **localhost:8080**

Referencia: https://linuxtechlab.com/wildfly-10-10-1-0-installation/

2.5. Crear la base de datos Matefun

Acceder al DBMS MySQL.

```
mysql -u root -p
```

Crear la base de datos Matefun

```
CREATE DATABASE Matefun;
```

Salir del DBMS

```
exit;
```

Pedir el dump de la base de datos Matefun al profesor **Marcos Viera**.

Ubicarse en el directorio donde está el dump de la base de datos Matefun y hacer el dump para crear y poblar las tablas

```
mysql -u USUARIO -p Matefun < Matefun_inicial.sql
```

[Opcional] Estando en la consola del DBMS MySQL, para seleccionar la base de datos Matefun y mirar sus tablas hacer

```
use Matefun;
show tables;
```

Referencia:
https://www.digitalocean.com/community/tutorials/how-to-import-and-export-databases-in-mysql-or-mariadb

2.6. Instalar y configurar el driver MySQL JDBC en Wildfly

Descargar conector JDBC (**mysql-connector-java-8.0.25.jar**) desde acá
https://repo1.maven.org/maven2/mysql/mysql-connector-java/8.0.25/

En una **terminal 1**, levantar el servidor Wildfly

```
sudo sh /data/wildfly-10.1.0.Final/bin/standalone.sh
```

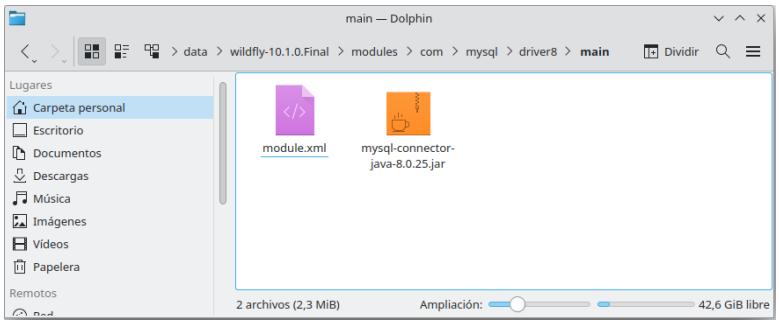
En una **terminal 2** (otra terminal distinta a la anterior), conectarse a la consola de jboss

```
sudo sh /data/wildfly-10.1.0.Final/bin/jboss-cli.sh -c
```

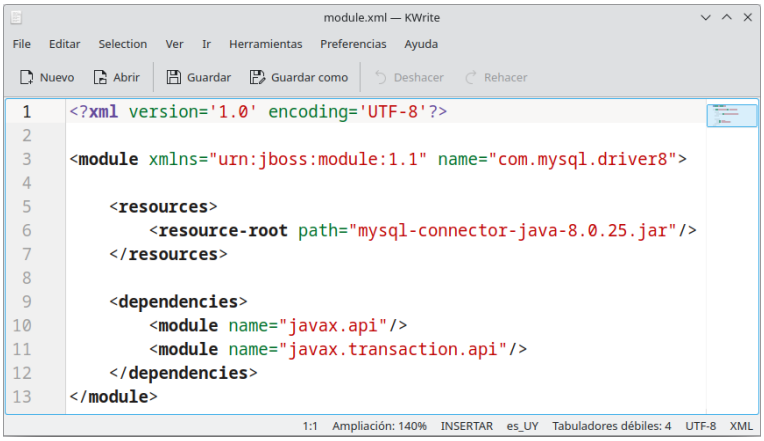
En la **terminal 2**, ejecutar (donde en **`--resources`** va el path al mysql-connector-java descargado anteriormente)

`module add --name=com.mysql.driver8 --dependencies=javax.api,javax.transaction.api --resources=/home/username/Descargas/mysql-connector-java-8.0.25.jar`

El comando debió crear el siguiente directorio, copiando el mysql-connector y creando el archivo module.xml.



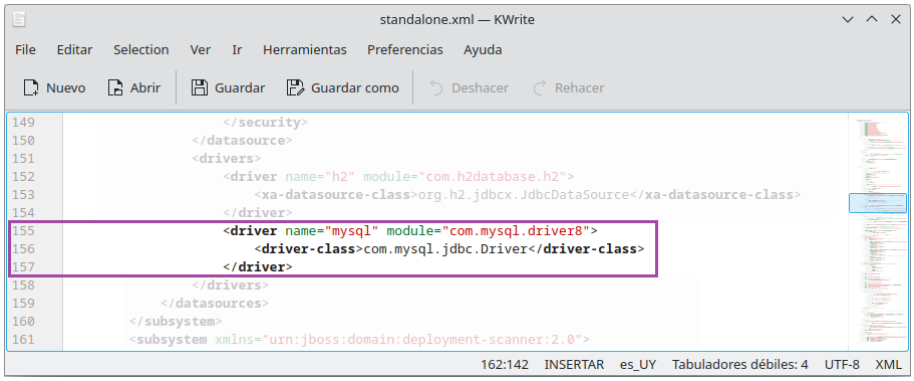
El contenido del archivo module.xml debe ser el siguiente



En la **terminal 2**, instalar el driver JDBC

`/subsystem=datasources/jdbc-driver=mysql/:add(driver-module-name=com.mysql.driver8,driver-name=mysql,driver-class-name=com.mysql.jdbc.Driver)`

En el archivo `/data/wildfly-10.1.0.Final/standalone/configuration/standalone.xml` se puede ver el resultado del anterior comando.



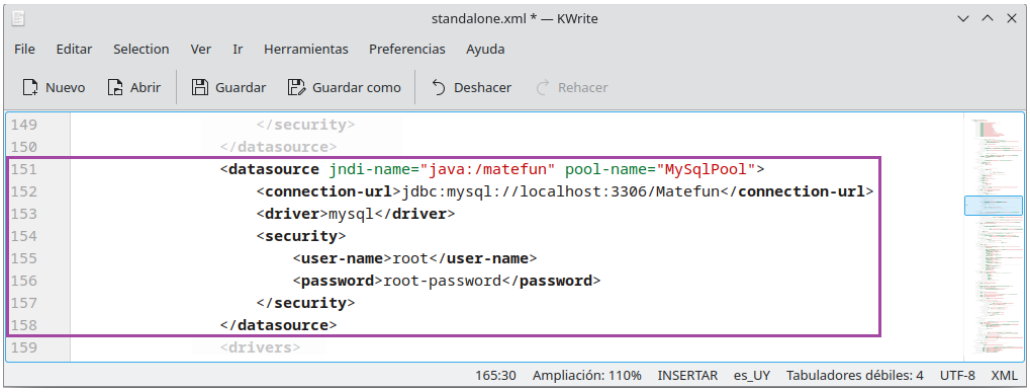
En la **terminal 2**, crear el data-source (tener en cuenta las consideraciones siguientes)

```
data-source add --jndi-name=java:/matefun --name=MySqlPool --connection-url=jdbc:mysql://localhost:3306/Matefun --driver-name=mysql --user-name=root --password=root-password
```

Consideraciones del comando anterior:

- `matefun` es el nombre del data-source. Este nombre se encuentra ubicado en el tag jta-data-source del archivo `/Servidor JEE/src/main/resources/META-INF/persistence.xml` del repositorio Frontend.
- `Matefun` es el nombre de la base de datos creada en la sección 2.1.5.
- `root` y `root-password` son el usuario y contraseña con los cuales se puede acceder a la base de datos `Matefun` del DBMS MySQL

En el archivo `/data/wildfly-10.1.0.Final/standalone/configuration/standalone.xml` se puede ver el resultado del anterior comando.



Finalmente, cerrar la **terminal 2** y reiniciar el servidor Wildfly (por ejemplo, detener la ejecución de la **terminal 1**) para que queden impactados los cambios.

Referencia:
https://reachmnadeem.wordpress.com/2021/05/13/install-and-configure-mysql-jdbc-driver-on-jboss-wildfly/

2.7. Instalar NodeJS

Se requiere una versión de NodeJS **^16.10.0**[1] para desarrollar en modo development y empaquetar para producción el Frontend Angular de MateFun.

Instalar NodeJS

```
sudo dnf module install nodejs:16/common
```

Para corroborar la versión instalada de NodeJS

```
node --version
```

Referencia:
https://developer.fedoraproject.org/tech/languages/nodejs/nodejs.html

[1] Versión **^16.10.0** es lo mismo que 16.**X**.**Y** con **X** >= 10 e **Y** cualquiera

3. Empaquetar la aplicación en un War

Ubicarse en el directorio deseado para almacenar el repositorio y clonarlo

```
git clone https://gitlab.fing.edu.uy/matefun/Frontend
```

Si no está instalado git, aparecerá un mensaje preguntando por instalar `git-core`. Responder de manera afirmativa.

Acceder al directorio que contiene el Frontend Angular

```
cd Frontend/Frontend\ Angular\ 4/
```

Instalar todas las dependencias del proyecto

```
npm install
```

Editar el archivo `config.ts` para actualizar desde cuáles URL consume los servicios el Frontend

```
kwrite src/app/shared/config.ts
```

Reemplazar el contenido del archivo por lo siguiente:

```
// psico
// export const SERVER = 'https://matefun.math.psico.edu.uy';
// export const GHCI_URL = 'wss://matefun.math.psico.edu.uy/endpoint';

// fing
// export const SERVER = 'https://www.fing.edu.uy/proyectos/matefun';
// export const GHCI_URL =
'wss://www.fing.edu.uy/proyectos/matefun/endpoint';

// local
export const SERVER = 'http://localhost:8080/proyectos/matefun';
export const GHCI_URL =
'ws://localhost:8080/proyectos/matefun/endpoint';
```

Nota importante: Notar que este cambio es solo para trabajar en local. Este archivo no debería ser actualizado en el servidor remoto Frontend, porque de otro modo, quedarían mal configuradas las URLs que utiliza el Frontend para consumir los servicios.

Editar el siguiente archivo para actualizar la ruta de despliegue de la aplicación

```
kwrite ../Servidor\ JEE/WebContent/WEB-INF/jboss-web.xml
```

Reemplazar el contenido del archivo por lo siguiente

```
<jboss-web>
   <context-root>/proyectos/matefun</context-root>
</jboss-web>
```

Nota importante: Al igual que sucede con el archivo `config.ts`, este archivo no debería ser actualizado en el repositorio remoto Frontend.

Volver al root del repositorio

```
cd ..
```

Generar el war asociado al repositorio, ejecutando

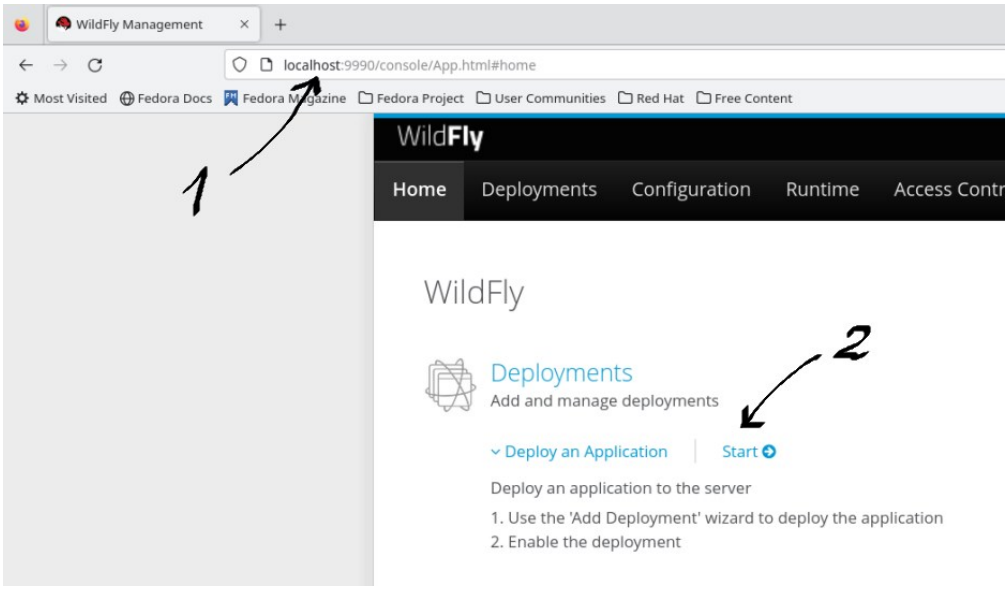
```
./generate-war.sh
```

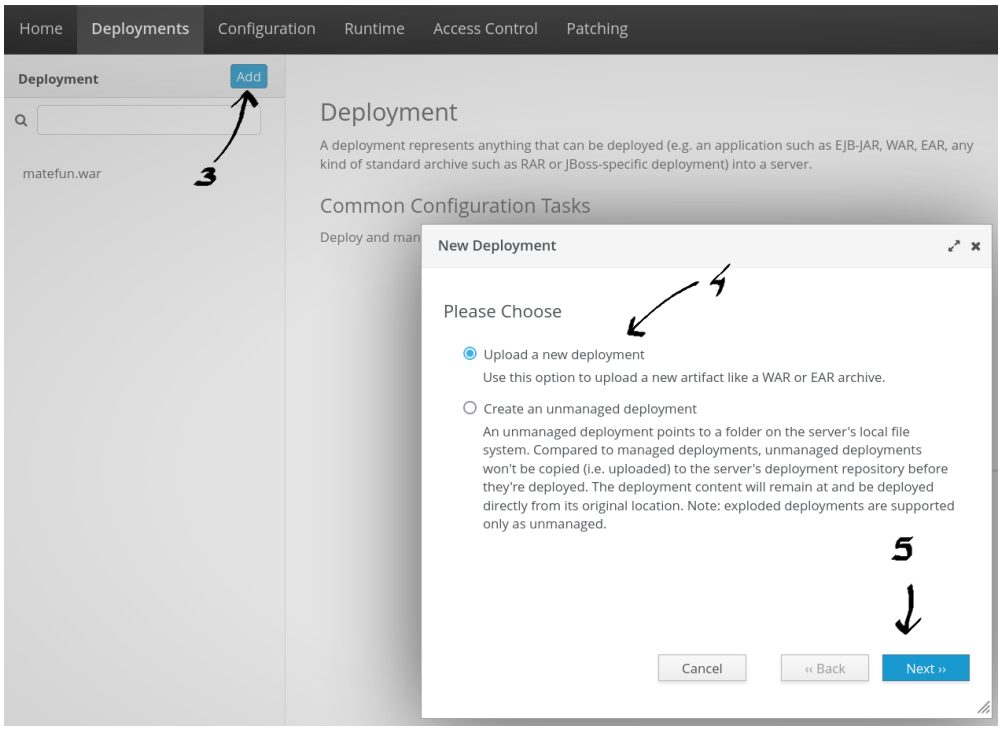
De aquí en más, a menos que se actualice alguna librería y por ende se tenga que actualizar las dependencias con `npm install`, solo debería hacer falta ejecutar el script anterior (**`./generate-war.sh`**) para actualizar el war a desplegar en el servidor local Wildfly.

El war asociado a la aplicación empaquetada es el archivo
`<Path al repositorio Frontend>/Servidor JEE/target/`**`matefun.war`**

Para desplegar el war en el servidor Wildfly, levantar el servicio y dirigirse a la **consola de administración**. Si no se cambió la IP, esta se encuentra **localhost:9990**

Luego ir a Deployments → Start → Add → Upload a new deployment → Next, seleccionar el matefun.war y dar Finish





Finalmente, si no se cambió la IP donde se aloja el servidor Wildfly, la aplicación Matefun se encontrará accesible en **localhost:8080/proyectos/matefun**

4. Configurar modo desarrollo para Frontend Angular

Recomendamos usar el IDE Visual Studio Code, ya que es un IDE muy liviano, completo y con muchas opciones de extensión fáciles de instalar.

Para iniciar el modo desarrollo del Frontend Angular, basta con estar parados sobre el directorio `<Path al repositorio Frontend>/Frontend\ Angular\ 4/` y ejecutar

```
npm start
```

Esto levantará la consola de desarrollo Angular, la cual consumirá los servicios del Backend que están desplegados sobre el servidor Wildfly. Notar que el servidor Wildfly tiene que estar levantado para que los servicios sean accesibles por el Frontend. Además, el archivo `config.ts` tiene que estar configurado como se menciona en esta guía, de otro modo, es muy posible que no estén correctamente definidas las URLs de los servicios.

5. Troubleshooting

Algunos problemas comunes que pueden impedir que la estación de trabajo sea configurada exitosamente:

- Asegurarse de tener instalada la versión 10.1.0 de Wildfly. Otras versiones de Wildfly dan problemas al momento de desplegar el war de la aplicación en el servidor.
- No tener la base de datos MateFun creada.
- No tener levantado el servicio de MySQL.
- No tener levantado el servicio asociado al Wildfly.
- No tener la versión correcta de NodeJS instalada. Al momento de escribir este documento, la versión de Angular que se usa para el Frontend es la versión 13, pero en un futuro, se puede actualizar la versión usada de Angular la cual puede exigir otros requerimientos superiores de NodeJS. **Tip**: Usar nvm para mantener múltiples versiones de NodeJS instaladas a la vez.
- Leer la terminal del Wildfly al momento de levantar el servicio. Esta suele dar indicios de los problemas que puede haber en el proceso. En algunas ocasiones, hay que desplegar de nuevo el war.
- No tener bien configurado el archivo `config.ts` de Angular, para hacer que apunte a los servicios de la estación de trabajo.
- No tener bien colocado el nombre del data-source en el archivo `persistence.xml` del Backend.