

TAREA DAW03

1 - Estructura de directorios de una aplicación web: Su directorio raíz contendrá una carpeta **WEB-INF** (que contendrá a su vez, el fichero **web.xml** y las subcarpetas **class** y **lib**) y el resto de carpetas contendrán los archivos estáticos.

Un ejemplo sería:

Aplic_Web(Directorio raíz):

- WEB-INF(carpetas):

- web.xml

- class(carpetas): archivos compilados(servlets y beans)

- lib(carpetas): librerías adicionales

-Carpetas de archivos estáticos

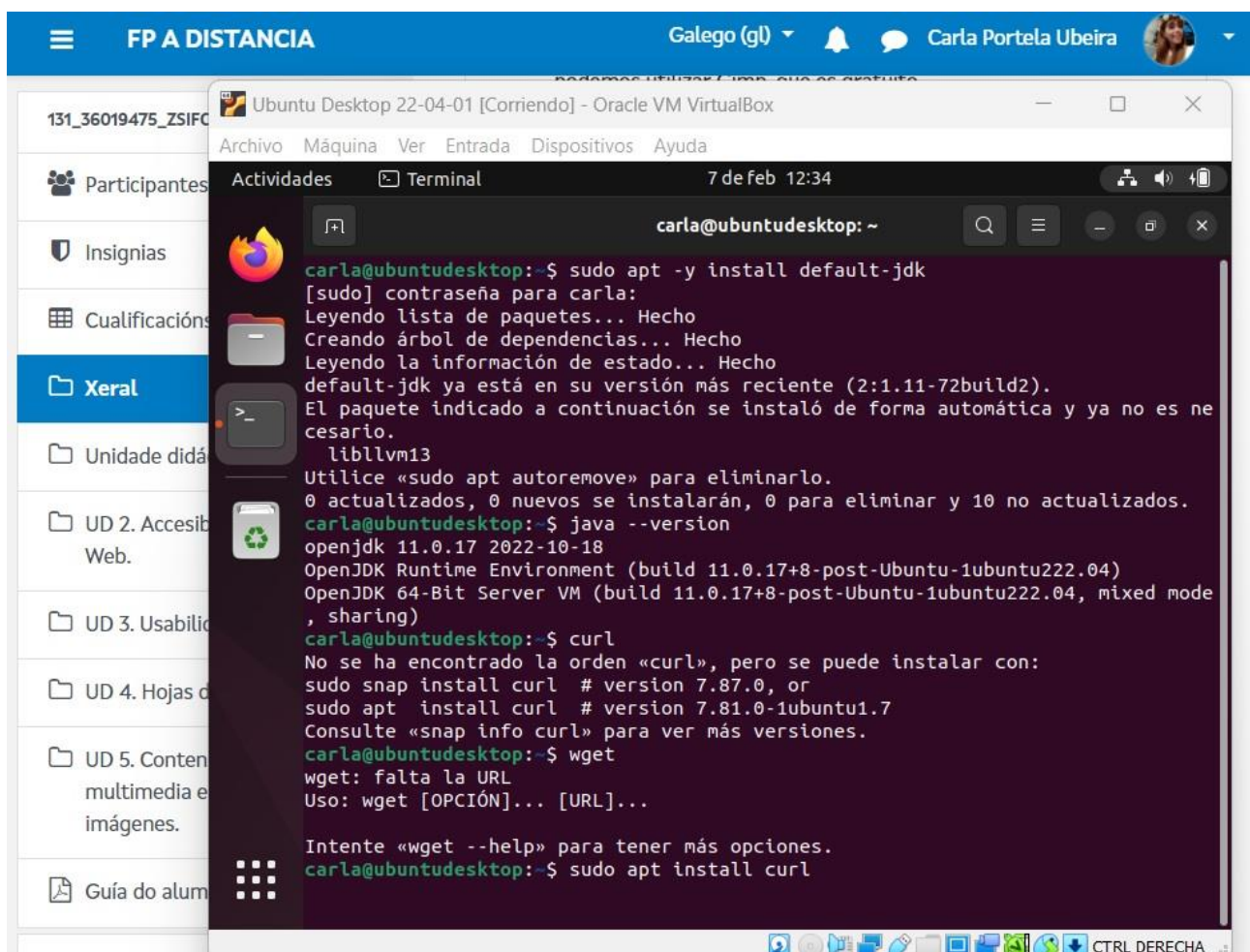
2 - Etiquetas de contenido de build.xml:

- Project: Elemento raíz del fichero xml (sólo puede haber uno en todo el fichero).
- Property: Parámetro que se precisa para procesar la aplicación. Ant incluye las básicas como BaseDir, ant.file y ant.java.version
- Target: Objetivo o conjunto de tareas que se desean aplicar a la aplicación. Se puede hacer que unos objetivos dependan de otros.
- Task: Tarea o código ejecutable que se aplicará a la aplicación y que pueden contener distintas propiedades.

3 - 1.Instalación de JDK8

Instalamos **JDK**, como usuario root, mediante el comando *apt install default-jdk*

Comprobamos que estén instalados tanto **wget** como **curl**(sino lo están, los instalamos).



```
carla@ubuntudesktop: ~  
carla@ubuntudesktop:~$ sudo apt -y install default-jdk  
[sudo] contraseña para carla:  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias... Hecho  
Leyendo la información de estado... Hecho  
default-jdk ya está en su versión más reciente (2:1.11-72build2).  
El paquete indicado a continuación se instaló de forma automática y ya no es necesario.  
liblvm2  
Utilice «sudo apt autoremove» para eliminarlo.  
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 10 no actualizados.  
carla@ubuntudesktop:~$ java --version  
openjdk 11.0.17 2022-10-18  
OpenJDK Runtime Environment (build 11.0.17+8-post-Ubuntu-1ubuntu222.04)  
OpenJDK 64-Bit Server VM (build 11.0.17+8-post-Ubuntu-1ubuntu222.04, mixed mode, sharing)  
carla@ubuntudesktop:~$ curl  
No se ha encontrado la orden «curl», pero se puede instalar con:  
sudo snap install curl # version 7.87.0, or  
sudo apt install curl # version 7.81.0-1ubuntu1.7  
Consulte «snap info curl» para ver más versiones.  
carla@ubuntudesktop:~$ wget  
wget: falta la URL  
Uso: wget [OPCIÓN]... [URL]...  
  
Intente «wget --help» para tener más opciones.  
carla@ubuntudesktop:~$ sudo apt install curl
```

3 - 2.Crear usuario de WildFly

Creamos, como usuario no root, el **grupo wildfly** y un **usuario** llamado **wildfly** que se agrega al grupo creado mediante los comandos:

```
sudo groupadd -r wildfly
```

```
sudo useradd -r -g wildfly -s /bin/false -p $(cat /dev/urandom | tr -dc 'x' | fold -n 12 | xargs printf '%s\n') wildfly
```

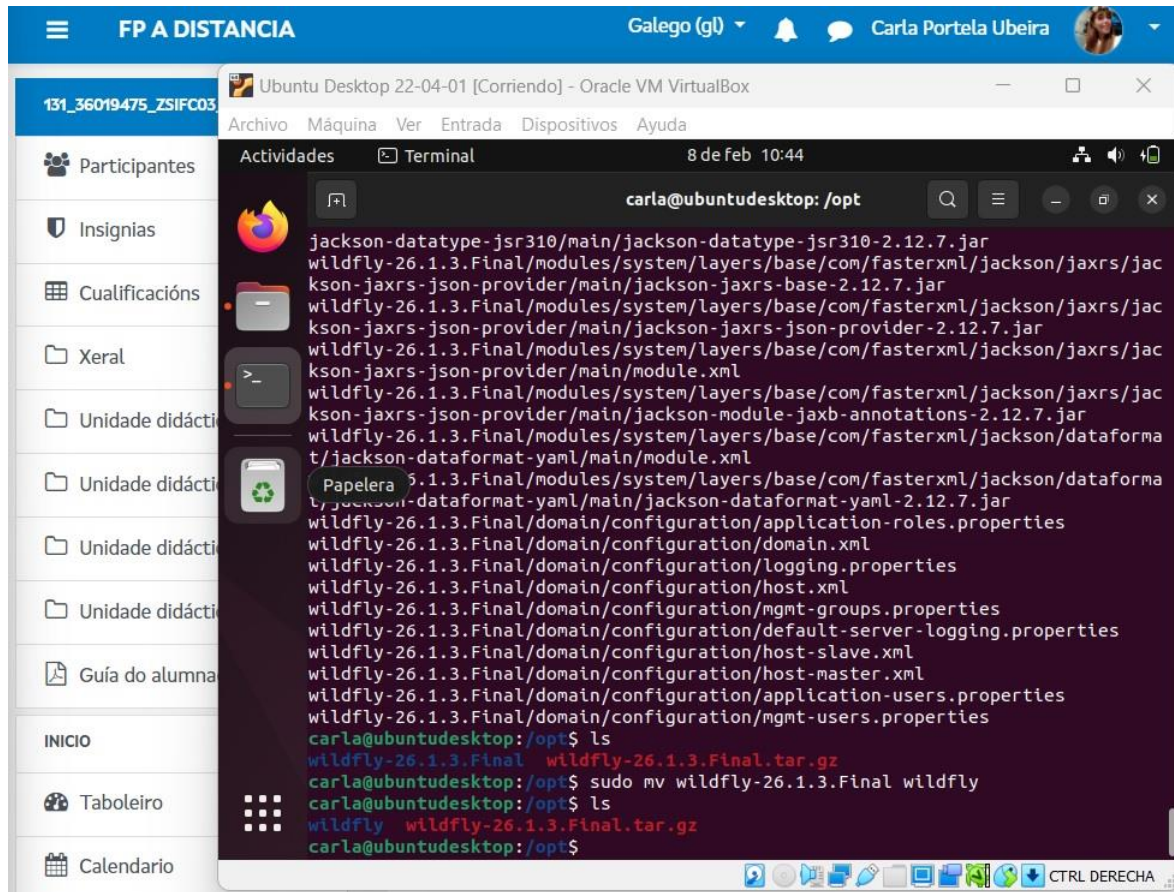
3 - 3.Descargar e instalar Wildfly 26.1.3.Final

Descargamos el paquete wildfly-26.1.3.Final.tgz en la carpeta /tmp

Lo descomprimos y lo movemos al directorio /opt

Creamos el enlace simbólico para Wildfly

Y, le pasamos la propiedad del directorio al usuario wildfly



3 - 4.Configurar systemd y el archivo wildfly.conf

Creamos el directorio para guardar el archivo de configuración:

```
sudo mkdir -p etc/wildfly
```

Copiamos el archivo de configuración al directorio etc/wildfly

Copiamos el script **launch.sh** al directorio opt/wildfly/bin/

Le ponemos el atributo ejecutable a los dicheros del directorio bin

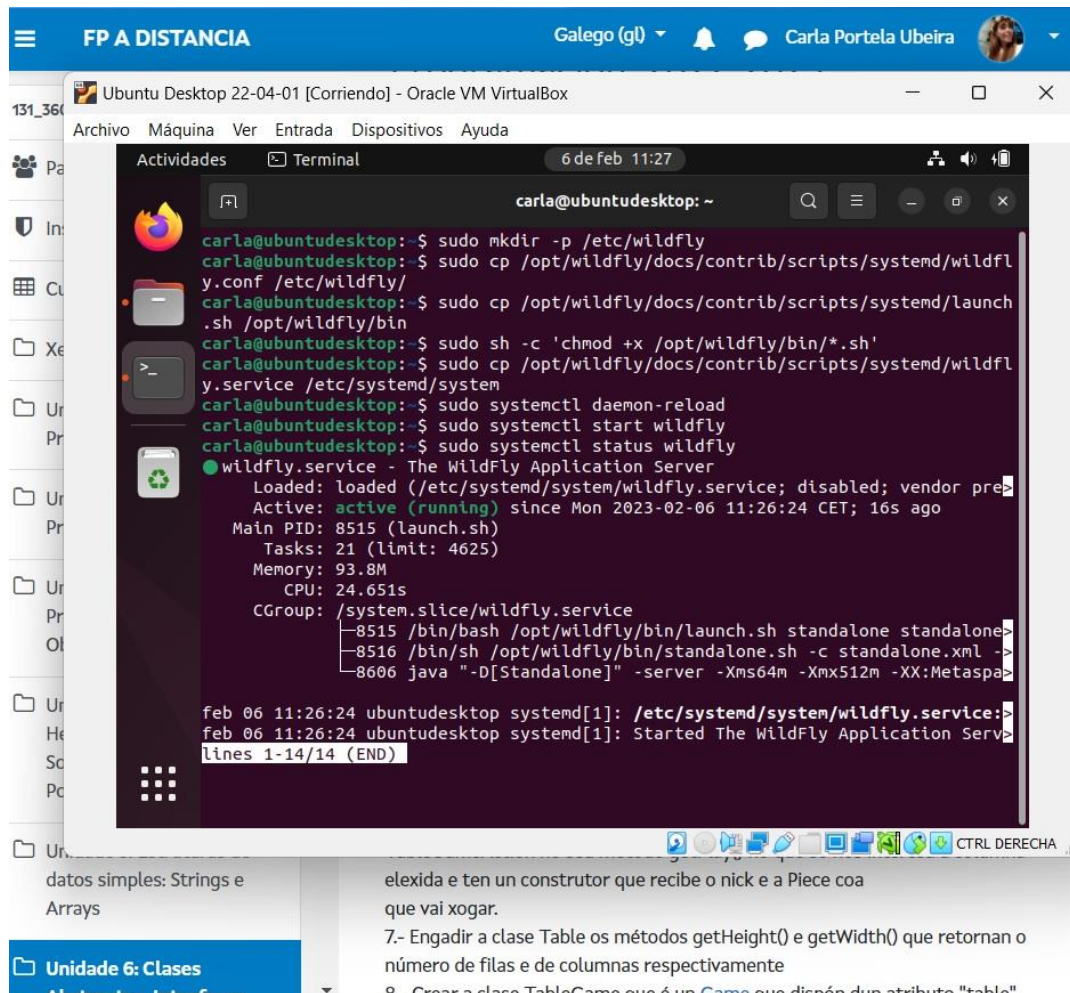
Finalmente, copiamos el archivo del servicio al directorio etc/systemd/system/

Recargamos systemd, arrancamos el servicio Wildfly y comprobamos que esté funcionando:

```
sudo systemctl daemon-reload
```

```
sudo systemctl start wildfly
```

```
sudo systemctl status wildfly
```



3-5. Configurar la autenticación de Wildfly

Agregamos un usuario ejecutando el script **add.user.sh**, seleccionamos la opción a y añadimos un usuario con nombre “usuario” y contraseña “usuario”.

Por último, verificamos la instalación de Wildfly mediante el comando:

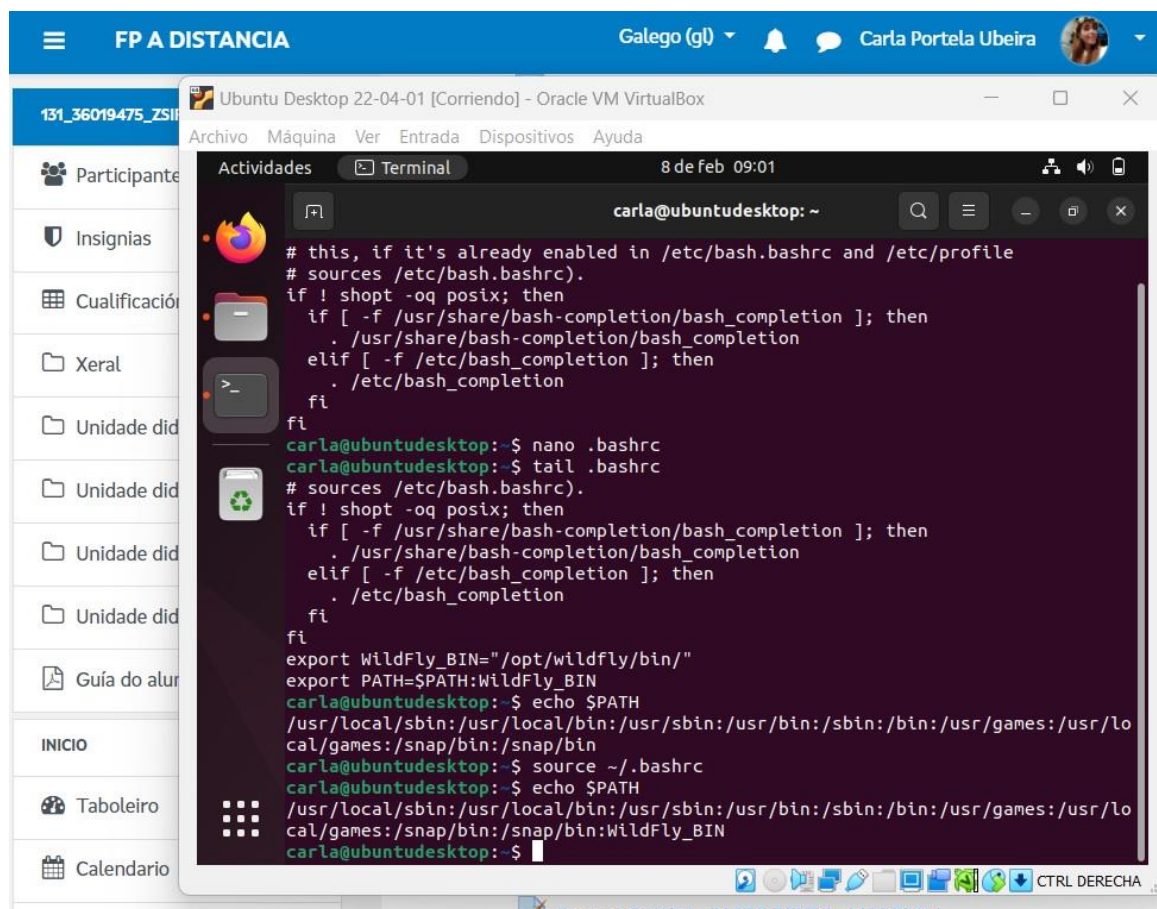
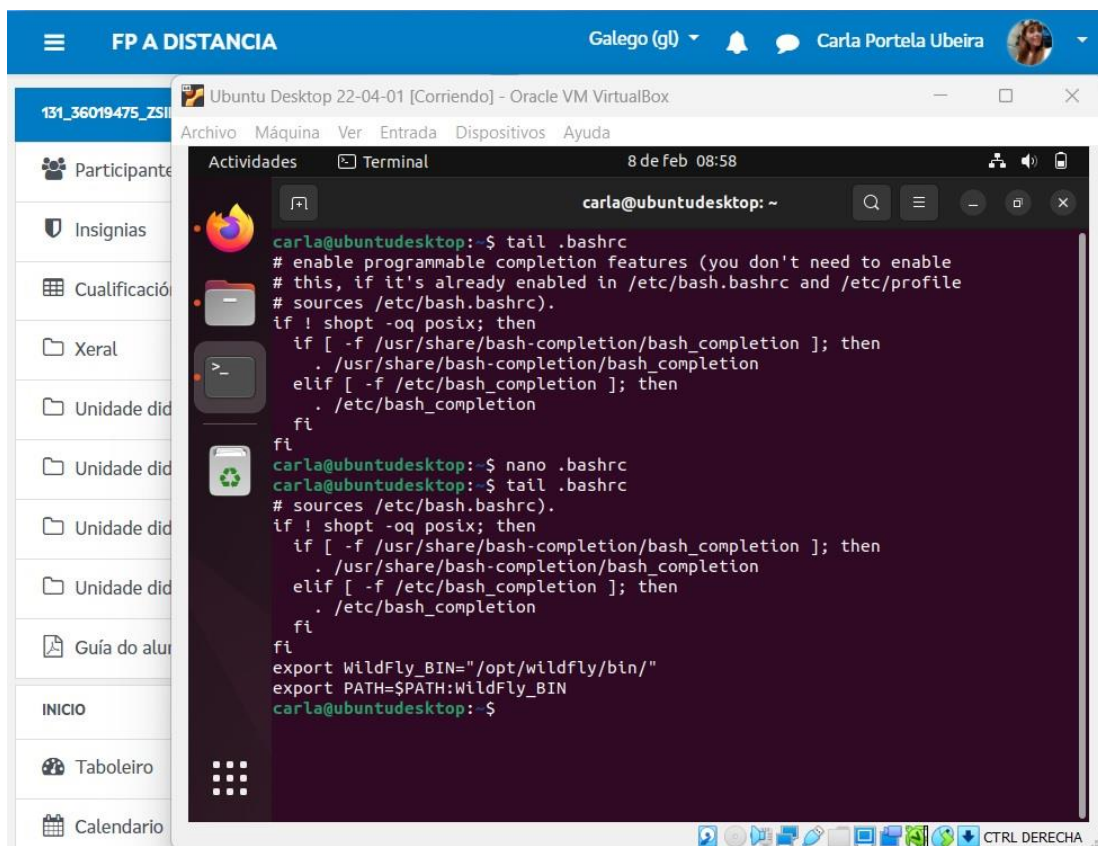
sudo systemctl status wildfly


```
carla@ubuntudesktop: ~  
carla@ubuntudesktop: $ sudo /opt/wildfly/bin/add-user.sh  
[sudo] contraseña para carla:  
  
What type of user do you wish to add?  
a) Management User (mgmt-users.properties)  
b) Application User (application-users.properties)  
(a): a  
  
Enter the details of the new user to add.  
Using realm 'ManagementRealm' as discovered from the existing property files.  
Username : usuario  
Password recommendations are listed below. To modify these restrictions edit the  
add-user.properties configuration file.  
- The password should be different from the username  
- The password should not be one of the following restricted values {root, admin,  
in, administrator}  
- The password should contain at least 8 characters, 1 alphabetic character(s)  
, 1 digit(s), 1 non-alphanumeric symbol(s)  
Password :  
WFLYDM0098: The password should be different from the username  
Are you sure you want to use the password entered yes/no? yes  
Re-enter Password :  
What groups do you want this user to belong to? (Please enter a comma separated  
list, or leave blank for none)[ ]: ManagementRealm  
About to add user 'usuario' for realm 'ManagementRealm'  
Is this correct yes/no? yes  
Added user 'usuario' to file '/opt/wildfly-19.1.0.Final/standalone/configuration/  
n/mgmt-users.properties'
```

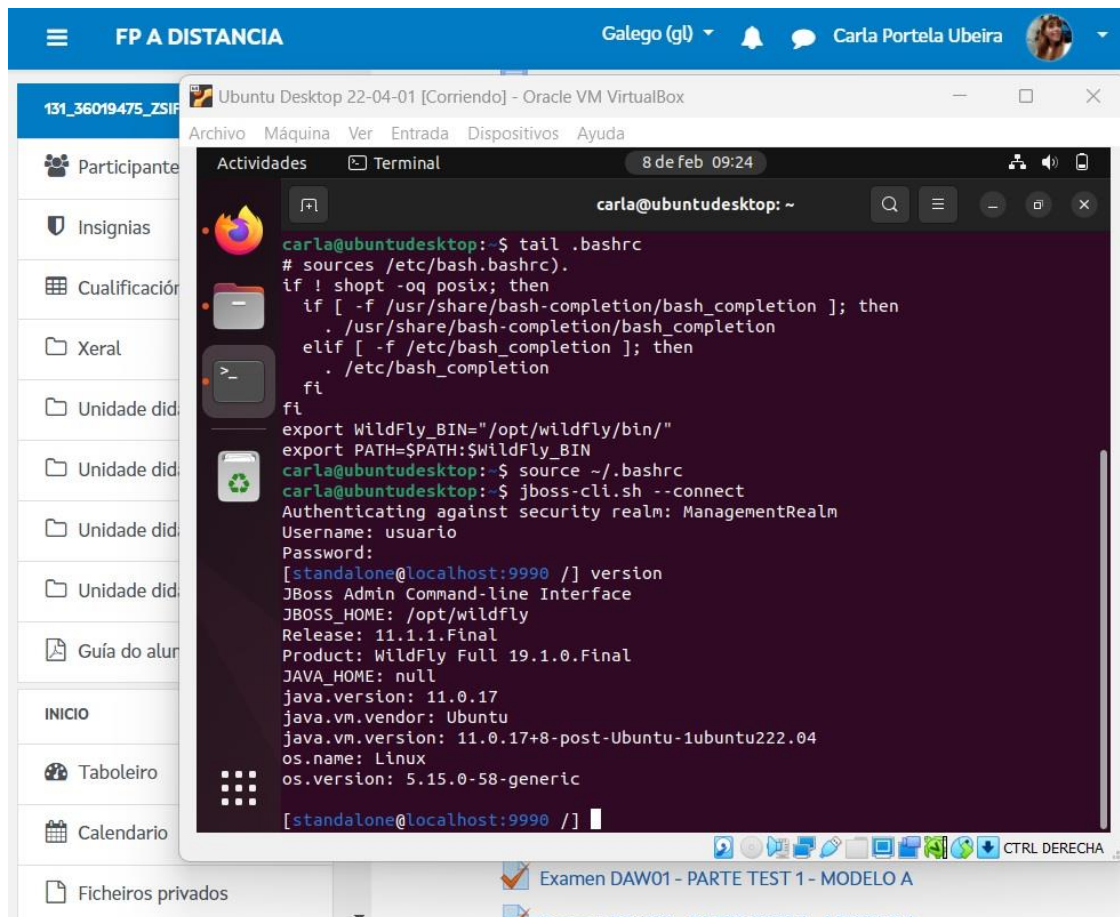
```
Added user 'usuario' to file '/opt/wildfly-19.1.0.Final/domain/configuration/mg  
mt-users.properties'  
Added user 'usuario' with groups ManagementRealm to file '/opt/wildfly-19.1.0.F  
inal/standalone/configuration/mgmt-groups.properties'  
Added user 'usuario' with groups ManagementRealm to file '/opt/wildfly-19.1.0.F  
inal/domain/configuration/mgmt-groups.properties'  
Is this new user going to be used for one AS process to connect to another AS p  
rocess?  
e.g. for a slave host controller connecting to the master or for a Remoting con  
nection for server to server EJB calls.  
yes/no? yes  
To represent the user add the following to the server-identities definition <se  
cret value="dXN1YXJpbW==" />  
carla@ubuntudesktop: $ sudo systemctl status wildfly  
● wildfly.service - The WildFly Application Server  
   Loaded: loaded (/etc/systemd/system/wildfly.service; disabled; vendor pre  
   Active: active (running) since Mon 2023-02-06 11:26:24 CET; 14min ago  
   Main PID: 8515 (launch.sh)  
     Tasks: 61 (limit: 4625)  
    Memory: 270.7M  
       CPU: 1min 44.858s  
    CGroup: /system.slice/wildfly.service  
            └─8515 /bin/bash /opt/wildfly/bin/launch.sh standalone standalone  
            └─8516 /bin/sh /opt/wildfly/bin/standalone.sh -c standalone.xml ->  
            └─8606 java "-D[Standalone]" -server -Xms64m -Xmx512m -XX:Metaspa  
  
feb 06 11:26:24 ubuntudesktop systemd[1]: /etc/systemd/system/wildfly.service:  
feb 06 11:26:24 ubuntudesktop systemd[1]: Started The WildFly Application Serv  
lines 1-14/14 (END)
```

3 - 6. Acceso a la Wildfly Admin Console

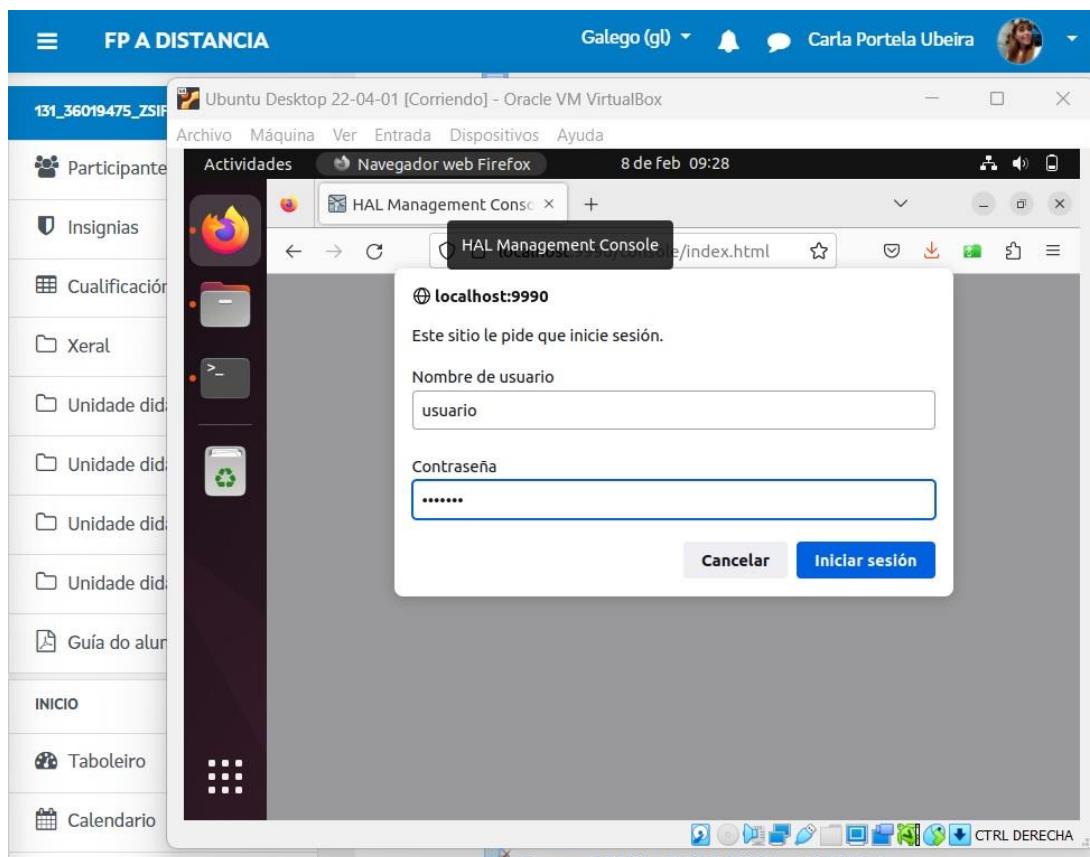
Añadimos /opt/wildfly/bin a la variable PATH y hacemos source del archivo bashrc



Y comprobamos que todo esta correcto conectándonos a la **Wildfly Admin Console** mediante el comando `jboss-cli.sh` ;e introducimos usuario y contraseña.



Cerramos la consola con el comando `exit` y accedemos a la **Wildfly Web Console** a través del navegador a través de: <http://localhost:9990> ; e introducimos usuario y contraseña.



DESPLIGUE DE APLICACIONES EN WILDFLY

Tenemos previamente que instalar **maven**, que es el gestor de paquetes:

```
apt install maven
```

Y nos descargamos las aplicaciones de ejemplo; para ello tecleamos en una nueva pestaña del navegador *localhost:8080* que nos lleva a la página por defecto de Wildfly y seleccionamos el enlace de Quickstarts donde aparecen ejemplos ya creados.

Tenemos que asegurar a la hora de descargar, que las versiones del código sean adecuadas para la versión del Jakarta que tenemos instalado y la del Wildfly; para ello en las **Quickstarts** le damos a *main* y en las *branches* (bifurcaciones, seleccionamos la nuestra que en este caso es 26.x).
descargamos el archivo en formato zip en la pestaña *code*.

Desde la consola vamos al directorio Descargas donde se encontrará el fichero zip:

```
cd Descargas/  
ls
```

Y descomprimos el archivo: *unzip quickstart-26.x.zip*

Comprobamos que se realizó la descompresión correctamente haciendo *ls* en Descargas y m̃nos movemos al directorio de los ejemplos descargados:

```
cd Descargas/quickstart-26.x  
ls
```

Y empezamos con la primera de ellas “helloworld”. Para ello nos movemos al directorio mediante *cd helloworld* y listamos su contenido mediante *ls*

Utilizando maven teclamos: *mvn package wildfly:deploy* (para construir el fichero war y lo va a subir al servidor) y empieza a descargar las dependencias, sobretudo la primera vez que lo ejecutamos.

Si queremos desinstalar la aplicación sería *mvn package wildfly:undeploy*

Si tenemos alguna duda, en Quickstart, seleccionando previamente la versión que tenemos instalada en *main* (en nuestro caso 26.x), existe el fichero **guide** que nos explica como lanzar las aplicaciones.

Una vez construido el fichero war, nos solicita usuario y clave para subir el archivo al servidor.

Podemos comprobar la correcta instalación en la consola web de Wildfly en **Deployments**(Ahí nos aparecen las aplicaciones instaladas).

Picnhamos sobre la aplicación y la habilitamos (Enable) y comprobamos que funciona en el navegador con la siguiente dirección en la barra de direcciones: *localhost:8080/helloworld*

Podemos ver el fichero con sus contenidos cliclando en la pestaña *View* al lado de la aplicación instalada en **Deployments**.

En Classes está el código fuente de java que ejecuta(aparece ya compilado).

Accedemos a el desde la consola situandonos en

```
cd Descargas/quickstart-26.x/helloworld/src/main/java/org/jboss/as/quickstart/helloworld  
y listamos los archivos .java mediante ls
```

Se puede modificar dicho código fuente mediante *nano HelloService.java* y *nano*

HelloWorldServlet.java. Guardamos y una vez modificado el código se vuelve a hacer deploy de la aplicación.

