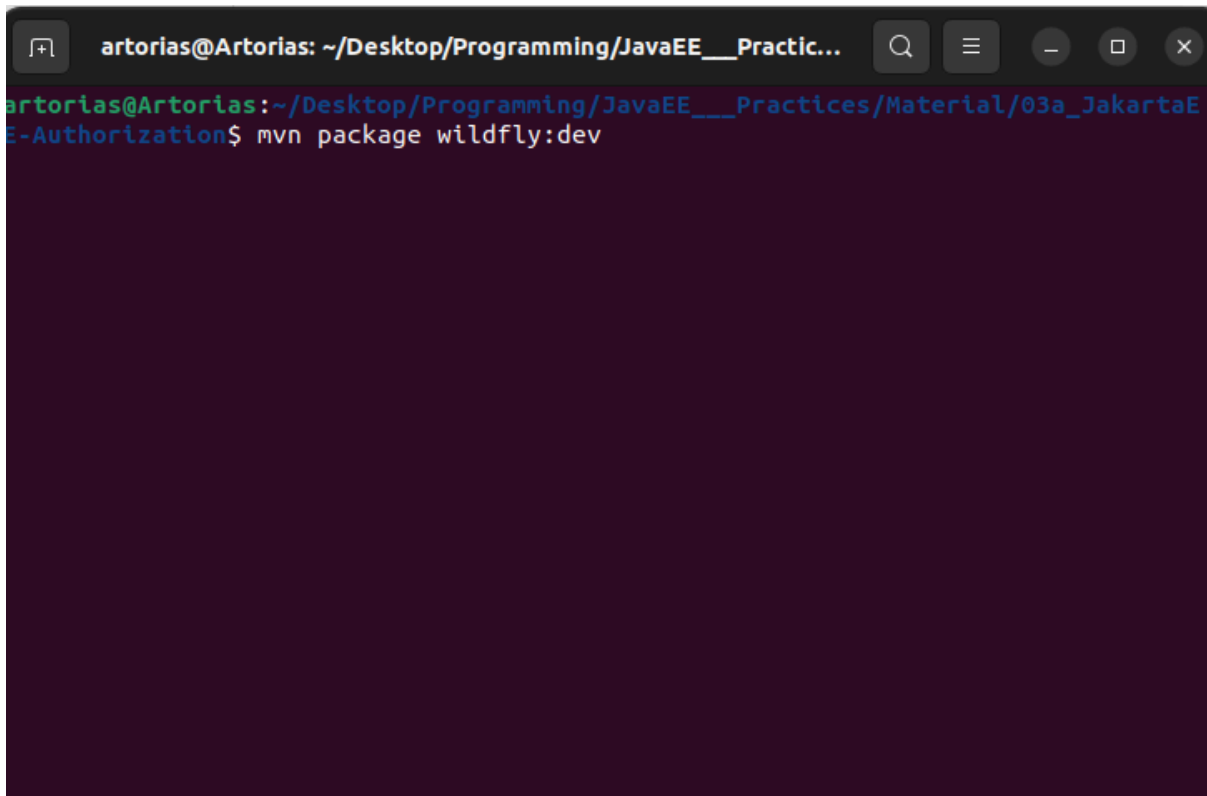


# Ejercicio 4

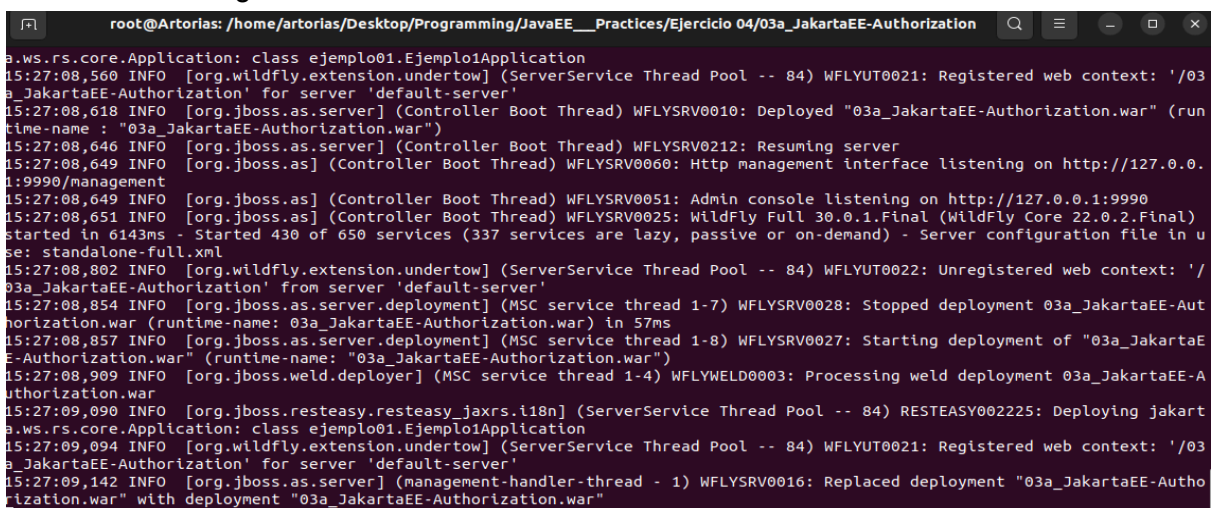
## Jakarta Security

1) Primero debemos bajar el código que el docente nos provee.  
Una vez bajado, podemos proceder a levantar el servidor.  
Debemos ejecutar el comando `mvn package wildfly:dev`.



```
artorias@Artorias: ~/Desktop/Programming/JavaEE___Practic...
artorias@Artorias:~/Desktop/Programming/JavaEE___Practices/Material/03a_JakartaE
E-Authorization$ mvn package wildfly:dev
```

Podemos ver lo siguiente:



```
root@Artorias: /home/artorias/Desktop/Programming/JavaEE___Practices/Ejercicio 04/03a_JakartaEE-Authorization
a.ws.rs.core.Application: class ejemplo01.Ejemplo1Application
15:27:08,560 INFO [org.wildfly.extension.undertow] (ServerService Thread Pool -- 84) WFLYUT0021: Registered web context: '/03
a_JakartaEE-Authorization' for server 'default-server'
15:27:08,618 INFO [org.jboss.as.server] (Controller Boot Thread) WFLYSRV0010: Deployed "03a_JakartaEE-Authorization.war" (run
time-name : "03a_JakartaEE-Authorization.war")
15:27:08,646 INFO [org.jboss.as.server] (Controller Boot Thread) WFLYSRV0212: Resuming server
15:27:08,649 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0060: Http management interface listening on http://127.0.0.
1:9990/management
15:27:08,649 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0051: Admin console listening on http://127.0.0.1:9990
15:27:08,651 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0025: WildFly Full 30.0.1.Final (WildFly Core 22.0.2.Final)
started in 6143ms - Started 430 of 650 services (337 services are lazy, passive or on-demand) - Server configuration file in u
se: standalone-full.xml
15:27:08,802 INFO [org.wildfly.extension.undertow] (ServerService Thread Pool -- 84) WFLYUT0022: Unregistered web context: '/'
03a_JakartaEE-Authorization' from server 'default-server'
15:27:08,854 INFO [org.jboss.as.server.deployment] (MSC service thread 1-7) WFLYSRV0028: Stopped deployment 03a_JakartaEE-Aut
horization.war (runtime-name: 03a_JakartaEE-Authorization.war) in 57ms
15:27:08,857 INFO [org.jboss.as.server.deployment] (MSC service thread 1-8) WFLYSRV0027: Starting deployment of "03a_JakartaE
E-Authorization.war" (runtime-name: "03a_JakartaEE-Authorization.war")
15:27:08,909 INFO [org.jboss.weld.deployer] (MSC service thread 1-4) WFLYWELD0003: Processing weld deployment 03a_JakartaEE-A
uthorization.war
15:27:09,090 INFO [org.jboss.resteasy.resteasy_jaxrs.i18n] (ServerService Thread Pool -- 84) RESTEASY002225: Deploying jakart
a.ws.rs.core.Application: class ejemplo01.Ejemplo1Application
15:27:09,094 INFO [org.wildfly.extension.undertow] (ServerService Thread Pool -- 84) WFLYUT0021: Registered web context: '/03
a_JakartaEE-Authorization' for server 'default-server'
15:27:09,142 INFO [org.jboss.as.server] (management-handler-thread - 1) WFLYSRV0016: Replaced deployment "03a_JakartaEE-Aut
horization.war" with deployment "03a_JakartaEE-Authorization.war"
```

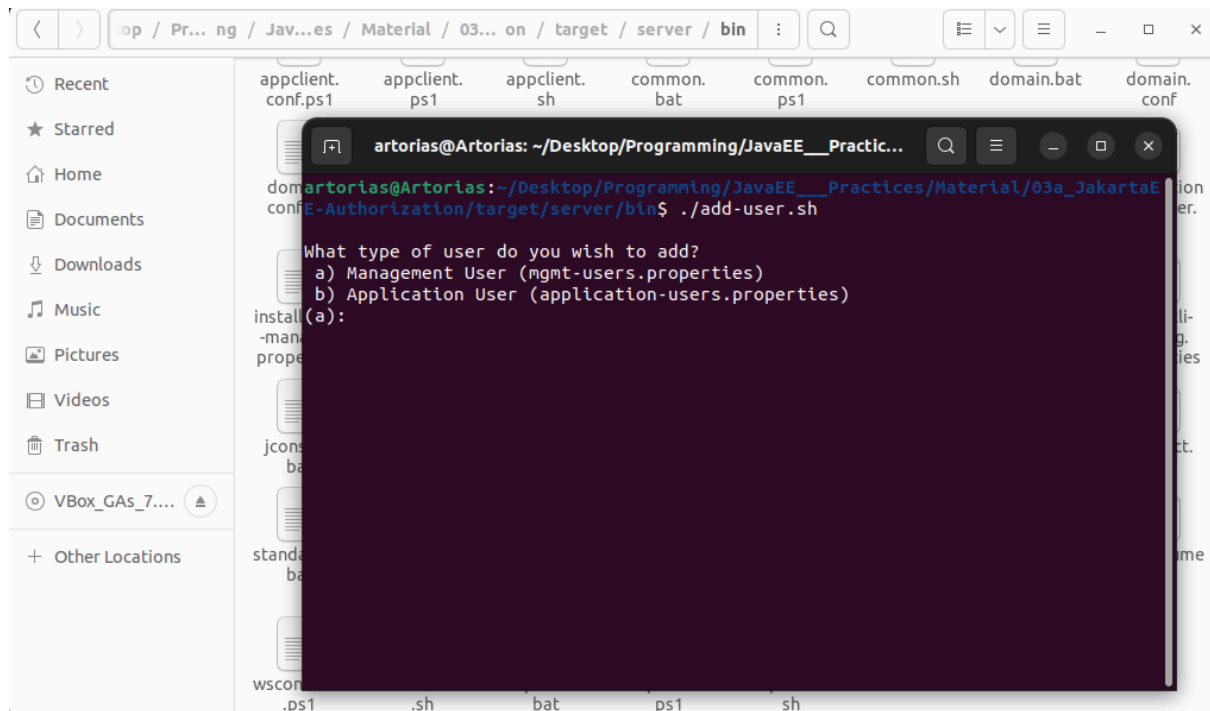
Dentro del código del servidor deployado, se puede ver como esta deployado en modo seguro. De ahí vemos como se deployea en modo seguro con HTTPS.

```
root@Artorias: /home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/03a_JakartaEE-Authorization
[Fault]
15:27:05,431 INFO [org.jboss.as.naming] (MSC service thread 1-3) WFLYNAM0003: Starting Naming Service
15:27:05,470 INFO [org.jboss.as.ejb3] (MSC service thread 1-7) WFLYEJB0481: Strict pool slsb-strict-max-pool is using a max i
stance size of 96 (per class), which is derived from thread worker pool sizing.
15:27:05,476 INFO [org.jboss.as.ejb3] (MSC service thread 1-2) WFLYEJB0482: Strict pool mdb-strict-max-pool is using a max in
stance size of 24 (per class), which is derived from the number of CPUs on this host.
15:27:05,501 INFO [org.wildfly.extension.undertow] (MSC service thread 1-6) WFLYUT0003: Undertow 2.3.10.Final starting
15:27:05,633 INFO [org.wildfly.extension.undertow] (ServerService Thread Pool -- 79) WFLYUT0014: Creating file handler for pa
th '/home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/03a_JakartaEE-Authorization/target/server/welcome-conte
nt' with options [directory-listing: 'false', follow-symlink: 'false', case-sensitive: 'true', safe-symlink-paths: '[]']
15:27:05,635 INFO [org.wildfly.extension.undertow] (MSC service thread 1-5) WFLYUT0012: Started server default-server.
15:27:05,636 INFO [org.wildfly.extension.undertow] (MSC service thread 1-5) Queuing requests.
15:27:05,636 INFO [org.wildfly.extension.undertow] (MSC service thread 1-5) WFLYUT0018: Host default-host starting
15:27:05,729 INFO [org.wildfly.extension.undertow] (MSC service thread 1-3) WFLYUT0006: Undertow HTTP listener default listen
ing on 127.0.0.1:8080
15:27:05,788 INFO [org.wildfly.extension.undertow] (MSC service thread 1-1) WFLYUT0006: Undertow HTTPS listener https listeni
ng on 127.0.0.1:8443
15:27:05,831 INFO [org.jboss.as.ejb3] (MSC service thread 1-5) WFLYEJB0493: Jakarta Enterprise Beans subsystem suspension com
plete
15:27:05,873 INFO [org.jboss.as.connector.subsystems.datasources] (MSC service thread 1-5) WFLYJCA0001: Bound data source [ja
va:jboss/datasources/ExampleDS]
15:27:05,984 INFO [org.wildfly.iop.openjdk] (MSC service thread 1-6) WFLYIIOP0009: CORBA ORB Service started
15:27:06,100 INFO [org.jboss.as.server.deployment] (MSC service thread 1-1) WFLYSRV0027: Starting deployment of "03a_JakartaE
E-Authorization.war" (runtime-name: "03a_JakartaEE-Authorization.war")
15:27:06,126 INFO [org.wildfly.extension.messaging-activemq] (MSC service thread 1-7) WFLYMSGAMQ0075: AIO wasn't located on t
his platform, it will fall back to using pure Java NIO. Your platform is Linux, install LibAIO to enable the AIO journal and a
chieve optimal performance
```

2) Ahora debemos modificar los usuarios y sus respectivos grupos.

Para esto, debemos irnos a la carpeta /target/server/bin/ y de ahí ejecutar el archivo que nos dejara modificar los mismos, llamado add-user.sh.

Ya dentro del directorio, ejecutaremos el comando add-user.sh.



Seguido de eso nos preguntara que tipo de user queremos agregar, al cual le daremos application user.

```
root@Artorias: /home/artorias/Desktop/Programming/JavaE...
root@Artorias:/home/artorias/Desktop/Programming/JavaEE___Practices/Material/03a
_JakartaEE-Authorization/target/server/bin# ./add-user.sh

What type of user do you wish to add?
a) Management User (mgmt-users.properties)
b) Application User (application-users.properties)
(a): b

Enter the details of the new user to add.
Using realm 'ApplicationRealm' as discovered from the existing property files.
Username : 
```

Aca nos pide user name, y luego nos va a pedir password, para agregarlo al ApplicationRealm.

```
root@Artorias: /home/artorias/Desktop/Programming/JavaE...
root@Artorias:/home/artorias/Desktop/Programming/JavaEE___Practices/Material/03a
_JakartaEE-Authorization/target/server/bin# ./add-user.sh

What type of user do you wish to add?
a) Management User (mgmt-users.properties)
b) Application User (application-users.properties)
(a): b

Enter the details of the new user to add.
Using realm 'ApplicationRealm' as discovered from the existing property files.
Username : Juan
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, administrator}
- The password should contain at least 8 characters, 1 alphabetic character(s),
1 digit(s), 1 non-alphanumeric symbol(s)
Password : WFLYDM0099: Password should have at least 8 characters!
Are you sure you want to use the password entered yes/no? 
```

En este caso, queremos agregar el usuario al grupo gerente, pero estaremos realizando lo demas pedido en la propuesta, de igual manera.

```
root@Artorias: /home/artorias/Desktop/Programming/JavaE...
What type of user do you wish to add?
a) Management User (mgmt-users.properties)
b) Application User (application-users.properties)
(a): b
Enter the details of the new user to add.
Using realm 'ApplicationRealm' as discovered from the existing property files.
Username : Juan
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, administrator}
- The password should contain at least 8 characters, 1 alphabetic character(s), 1 digit(s), 1 non-alphanumeric symbol(s)
Password :
WFLYDM0099: Password should have at least 8 characters!
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)[ ]: gerente
About to add user 'Juan' for realm 'ApplicationRealm'
Is this correct yes/no? y
```

De esta manera quedo agregado el usuario Juan al rol gerente.

```
root@Artorias: /home/artorias/Desktop/Programming/JavaE...
- The password should contain at least 8 characters, 1 alphabetic character(s), 1 digit(s), 1 non-alphanumeric symbol(s)
Password :
WFLYDM0099: Password should have at least 8 characters!
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)[ ]: gerente
About to add user 'Juan' for realm 'ApplicationRealm'
Is this correct yes/no? y
Added user 'Juan' to file '/home/artorias/Desktop/Programming/JavaEE___Practices/Material/03a_JakartaEE-Authorization/target/server/standalone/configuration/application-users.properties'
Added user 'Juan' to file '/home/artorias/Desktop/Programming/JavaEE___Practices/Material/03a_JakartaEE-Authorization/target/server/domain/configuration/application-users.properties'
Added user 'Juan' with groups gerente to file '/home/artorias/Desktop/Programming/JavaEE___Practices/Material/03a_JakartaEE-Authorization/target/server/standalone/configuration/application-roles.properties'
Added user 'Juan' with groups gerente to file '/home/artorias/Desktop/Programming/JavaEE___Practices/Material/03a_JakartaEE-Authorization/target/server/domain/configuration/application-roles.properties'
root@Artorias: /home/artorias/Desktop/Programming/JavaEE___Practices/Material/03a_JakartaEE-Authorization/target/server/bin#
```

Siguientemente, podemos intentar lo mismo, pero esta vez, agregando el us1 al grupo empleado.

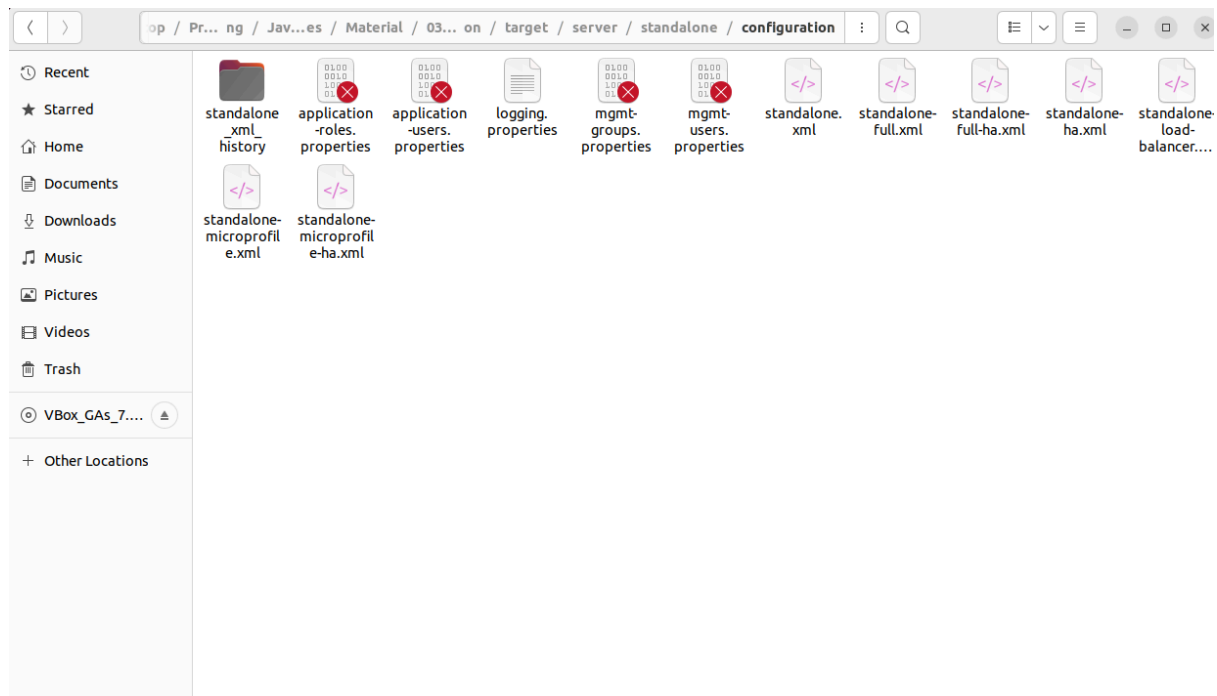
```
root@Artorias: /home/artorias/Desktop/Programming/JavaE...
root@Artorias:/home/artorias/Desktop/Programming/JavaEE___Practices/Material/03a
JakartaEE-Authorization/target/server/bin# ./add-user.sh

What type of user do you wish to add?
a) Management User (mgmt-users.properties)
b) Application User (application-users.properties)
(a): b






Enter the details of the new user to add.
Using realm 'ApplicationRealm' as discovered from the existing property files.
Username : usr1
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, admini
n, administrator}
- The password should contain at least 8 characters, 1 alphabetic character(s),
1 digit(s), 1 non-alphanumeric symbol(s)
Password :
WFLYDM0099: Password should have at least 8 characters!
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)[ ]: empleado
```

Y así replicaremos con los demás solicitados.

Una vez hecho esto, nos dirigiremos a la carpeta /server/standalone/configuration para chequear los archivos application-roles.properties y el archivo application-users.properties.



Aquí podemos ver los usuarios y sus respectivos roles

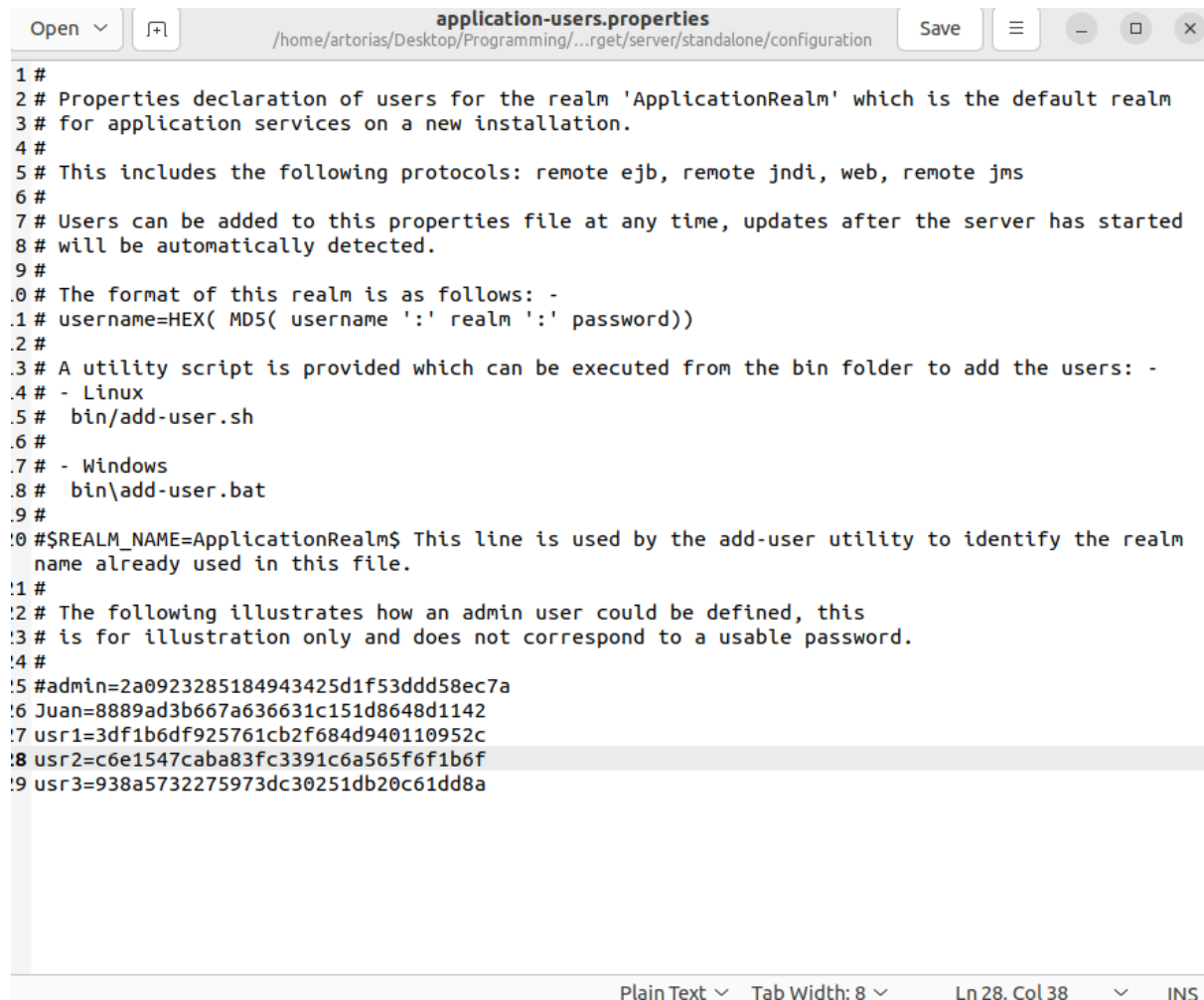
Open ▾  application-roles.properties Save    

/home/artorias/Desktop/Programming/...rget/server/standalone/configuration

```
1 #
2 # Properties declaration of users roles for the realm 'ApplicationRealm' which is the default
  realm
3 # for application services on a new installation.
4 #
5 # This includes the following protocols: remote ejb, remote jndi, web, remote jms
6 #
7 # Users can be added to this properties file at any time, updates after the server has started
8 # will be automatically detected.
9 #
10 # The format of this file is as follows: -
11 # username=role1,role2,role3
12 #
13 # A utility script is provided which can be executed from the bin folder to add the users: -
14 # - Linux
15 #   bin/add-user.sh
16 #
17 # - Windows
18 #   bin\add-user.bat
19 #
20 # The following illustrates how an admin user could be defined.
21 #
22 #admin=PowerUser,BillingAdmin,
23 #guest=guest
24 Juan=gerente
25 usr1=empleado
26 usr2=empleado
27 usr3=gerente
```

Plain Text ▾ Tab Width: 8 ▾ Ln 23, Col 13 ▾ INS





```
1 #
2 # Properties declaration of users for the realm 'ApplicationRealm' which is the default realm
3 # for application services on a new installation.
4 #
5 # This includes the following protocols: remote ejb, remote jndi, web, remote jms
6 #
7 # Users can be added to this properties file at any time, updates after the server has started
8 # will be automatically detected.
9 #
10 # The format of this realm is as follows: -
11 # username=HEX( MD5( username ':' realm ':' password))
12 #
13 # A utility script is provided which can be executed from the bin folder to add the users: -
14 # - Linux
15 #   bin/add-user.sh
16 #
17 # - Windows
18 #   bin\add-user.bat
19 #
20 # $REALM_NAME=ApplicationRealm$ This line is used by the add-user utility to identify the realm
21 # name already used in this file.
22 #
23 # The following illustrates how an admin user could be defined, this
24 # is for illustration only and does not correspond to a usable password.
25 #
26 #admin=2a0923285184943425d1f53ddd58ec7a
27 # Juan=8889ad3b667a636631c151d8648d1142
28 #usr1=3df1b6df925761cb2f684d940110952c
29 #usr2=c6e1547caba83fc3391c6a565f6f1b6f
30 #usr3=938a5732275973dc30251db20c61dd8a
```

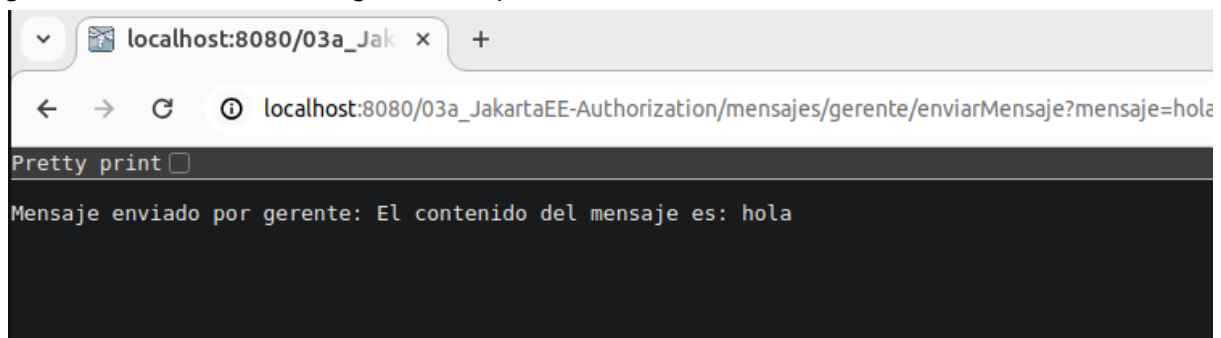
Y aqui en el user properties, podemos ver lo que seria el password hasheado, para cada user respectivo.

3) Para el siguiente punto debemos utilizar un curl sin credenciales e invocar el endpoint gerente/enviarMensaje?mensaje=hola.  
Para esto invocamos el endpoint con un curl curl -v  
[http://localhost:8080/03a\\_JakartaEE-Authorization/mensajes/gerente/enviarMensaje?mensaje=hola](http://localhost:8080/03a_JakartaEE-Authorization/mensajes/gerente/enviarMensaje?mensaje=hola)  
Obtendremos la siguiente respuesta.

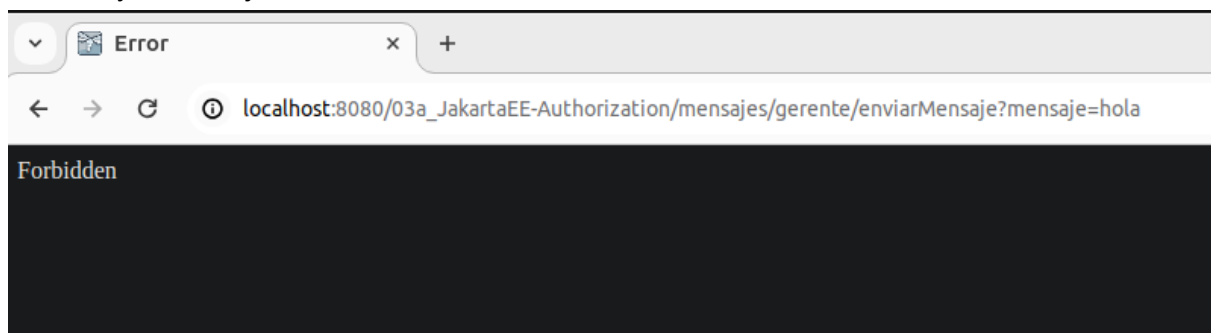
```
root@Artorias: /home/artorias/Desktop/Programming/JavaEE__Practices/Material/03a_JakartaEE-Authorization
JakartaEE-Authorization# curl -v http://localhost:8080/03a_JakartaEE-Authorization/mensajes/gerente/enviarMensaje?mensaje=hol
* Trying 127.0.0.1:8080...
* Connected to localhost (127.0.0.1) port 8080 (#0)
> GET /03a_JakartaEE-Authorization/mensajes/gerente/enviarMensaje?mensaje=hol HTTP/1.1
> Host: localhost:8080
> User-Agent: curl/7.81.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 401 Unauthorized
< Expires: 0
< Connection: keep-alive
< WWW-Authenticate: Basic realm="ApplicationRealm"
< Cache-Control: no-cache, no-store, must-revalidate
< Pragma: no-cache
< Content-Type: text/html; charset=UTF-8
< Content-Length: 71
< Date: Wed, 17 Apr 2024 20:05:31 GMT
<
* Connection #0 to host localhost left intact
<html><head><title>Error</title></head><body>Unauthorized</body></html>root@Artoriasroot@Artoriasroot@Artoriasroot@Artorias
root@Artorias: /home/artorias/Desktop/Programming/JavaEE__Practices/Material/03a_JakartaEE-Authorization#
```

Como se puede ver, nos arroja 401 Unauthorized, esto es, por el grupo al que intentamos acceder, que es gerente, no tenemos los permisos.

4) Intentamos nuevamente tomar ese endpoint pero esta vez en el browser. En este momento nos sale una pantalla de login, la cual usaremos la del us3 que era gerente, obtendremos la siguiente respuesta.



5) En el siguiente punto haremos lo mismo, pero con credenciales de usr1 y usr2. Para ambos casos, vemos que esos grupos de usuarios no tienen permisos para poder acceder y nos arroja un Forbidden.



6) Para el siguiente trataremos de alcanzar el siguiente endpoint,  
[https://localhost:8443/03a\\_JakartaEE-Authorization/mensajes/gerente/enviarMensajeSeguro?mensaje=hola](https://localhost:8443/03a_JakartaEE-Authorization/mensajes/gerente/enviarMensajeSeguro?mensaje=hola)



Obteniendo el siguiente resultado, donde nos arroja el código 302 Found y nos deja acceder.

```
root@Artorias: /home/artorias/Desktop
root@Artorias:/home/artorias/Desktop# curl --user usr3:1212 -v http://localhost:8080/03a_JakartaEE-Authorization/mensajes/gerente/enviarMensajeSeguro?mensaje=hola
* Trying 127.0.0.1:8080...
* Connected to localhost (127.0.0.1) port 8080 (#0)
* Server auth using Basic with user 'usr3'
* GET /03a_JakartaEE-Authorization/mensajes/gerente/enviarMensajeSeguro?mensaje=hola HTTP/1.1
* Host: localhost:8080
* Authorization: Basic dXNyMzoxMjEy
* User-Agent: curl/7.81.0
* Accept: */*
*
* Mark bundle as not supporting multiuse
* HTTP/1.1 302 Found
* Connection: keep-alive
* Location: https://localhost:8443/03a_JakartaEE-Authorization/mensajes/gerente/enviarMensajeSeguro?mensaje=hola
* Content-Length: 0
* Date: Wed, 17 Apr 2024 23:46:52 GMT
*
* Connection #0 to host localhost left intact
root@Artorias:/home/artorias/Desktop# S
```

Primero debemos bajar el código que el docente nos provee.  
Una vez bajado, podemos proceder a levantar el servidor.  
Debemos ejecutar el comando `mvn package wildfly:dev`.  
Podemos ver lo siguiente:

```
artorias@Artorias: ~/Desktop/Programming/JavaEE___Practic...
artorias@Artorias:~/Desktop/Programming/JavaEE___Practices/Material/03b_JakartaE
ESecurity$ mvn package wildfly:dev
```

Y al deployar se vería lo siguiente:

```

root@Artorias: /home/artorias/Desktop/Programming/JavaEE__Practices/Materi...
14:40:53,499 INFO [stdout] (ServerService Thread Pool -- 18) Hibernate: insert into Usuario (us
ername,passwordHash) values ('usr4','0af6d47d5944b3fdfec60c23a9b83224a989605633102aee0bf9cb0e6e
48ea6')
14:40:53,499 INFO [stdout] (ServerService Thread Pool -- 18) Hibernate: insert into Grupo (nomb
re) values ('grupo1')
14:40:53,500 INFO [stdout] (ServerService Thread Pool -- 18) Hibernate: insert into Grupo (nomb
re) values ('grupo2')
14:40:53,500 INFO [stdout] (ServerService Thread Pool -- 18) Hibernate: insert into Grupo (nomb
re) values ('admin')
14:40:53,501 INFO [stdout] (ServerService Thread Pool -- 18) Hibernate: insert into Usuario_Gru
po (Usuario_username, grupos_nombre) values ('usr1','grupo1')
14:40:53,501 INFO [stdout] (ServerService Thread Pool -- 18) Hibernate: insert into Usuario_Gru
po (Usuario_username, grupos_nombre) values ('usr2','grupo2')
14:40:53,502 INFO [stdout] (ServerService Thread Pool -- 18) Hibernate: insert into Usuario_Gru
po (Usuario_username, grupos_nombre) values ('usr3','grupo1')
14:40:53,502 INFO [stdout] (ServerService Thread Pool -- 18) Hibernate: insert into Usuario_Gru
po (Usuario_username, grupos_nombre) values ('usr3','grupo2')
14:40:53,503 INFO [stdout] (ServerService Thread Pool -- 18) Hibernate: insert into Usuario_Gru
po (Usuario_username, grupos_nombre) values ('usr4','admin')
14:40:53,825 INFO [org.wildfly.security.soteria.original.CdiExtension] (MSC service thread 1-8)
Activating jakarta.security.enterprise.authentication.mechanism.http.BasicAuthenticationMechani
smDefinition authentication mechanism from ejemplo00.infraestructura.seguridad.SeguridadConfigur
acion class
14:40:53,999 INFO [org.wildfly.security.soteria.original.SamRegistrationInstaller] (ServerServi
ce Thread Pool -- 18) Initializing Soteria 3.0.3.Final for context '/03b_JakartaEESecurity'
14:40:54,037 INFO [org.wildfly.extension.undertow] (ServerService Thread Pool -- 18) WFLYUT0021
: Registered web context: '/03b_JakartaEESecurity' for server 'default-server'
14:40:54,060 INFO [org.jboss.as.server] (management-handler-thread - 1) WFLYSRV0010: Deployed "
03b_JakartaEESecurity.war" (runtime-name : "03b_JakartaEESecurity.war")

```

7) Para el siguiente punto, debemos utilizar la herramienta OpenSSL. Podemos instalar

OpenSSL con el siguiente comando `sudo apt install openssl -y`

Una vez instalada, podemos correr el comando `openssl s_client -showcerts -connect localhost:8443`

Podemos verificar el resultado del certificado en consola:

```
root@Artorias: /home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/03a...
artorias@Artorias:~/Desktop/Programming/JavaEE__Practices/Ejercicio 04/03a_JakartaEE-Authorization$ su
Password:
root@Artorias:/home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/03a_JakartaEE-Authorization# openssl s_client -showcerts -connect localhost:8443
CONNECTED(000000003)
Can't use SSL_get_servername
depth=0 CN = localhost
verify error:num=18:self-signed certificate
verify return:1
depth=0 CN = localhost
verify return:1
---
Certificate chain
 0 s:CN = localhost
  i:CN = localhost
  a:PKKEY: rsaEncryption, 2048 (bit); sigalg: RSA-SHA256
  v:NotBefore: Apr 19 18:26:48 2024 GMT; NotAfter: Apr 17 18:26:48 2034 GMT
-----BEGIN CERTIFICATE-----
MIICzDCCAbagAwIBAgIJAJaG+u/AbZ1wMAsGCSqGSIb3DQEBCzAUMRIwEAYDVQQD
Ewlsb2NhbgHvc3QwIhgPMjAyNDA0MTkxODI2NDhaGA8yMDM0MDQxNzE4MjY0OFow
FDESMBAGA1UEAxMJbG9jYXRob3N0MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAAQZCsa0FwxPubqNB89s3cBTdF798sv6BD4fiYLnQWjpx0syyRwk1AUxw
udUgWmhDSLZMMtJQHlO1MfI17DyHmLn0iktY7jn3pAEUVY74yM0fidebdcXnlVZZ
499Nprc0Y1g3v119GtA02NctY0bWLz+Ch9nRispwJZGqdwzxcM1jRpD1BN/E5a1q
Pky5/BLgRYC95iQAoqJ4RmnLE/f4G88Sbg5q3t00iSDsDw6qHbA0K16fSvfIvICS
LVx7Lrt0SSo9e/9Bu7iQ5F3IG6DrNpz53FJAp5/50QHtttoyuyhjePDLZFIn6yD4h
9ZSkgM9V0ic3bCMGaGnykKbIoD0nQIDAQABoyEwHzAdBgNVHQ4EFgQUCjBld7yk
ugZibB9/FogCg2gtkhUwCwYJKoZIhvcNAQELA4IBAQCe+UJeQbFaWv9K358z3rFW
SMeyfNpIoubfP0zX0pIeY23snPhNfIorq1wE3+E/22IrOoqcFjbjrXj5ve5ZUxu/
yfi+5YNFo+i1Gcydr0F2RhcGpx6+p2mZ/g60MNnszi2EUxZOMRW+5eR1joVtLko7
z1FCvUa2YvMnHF0KEpK0RDs7cIww2MQvSFGPVHvdw/SugD98gEfeP+Bg/375Ti30
20EbhMvbBCdJ+Cj9bWMnKXC/g8HJTtXQsJkV1ZN495uAQPP/zzf7Px8G0Fy4t2FL
IiS6dqgku8G6RPYFhQcKglUyatLNKSbCXU3WToos2a7DBW2II13qzdPFL+YWL4SW
-----END CERTIFICATE-----
---
Server certificate
subject=CN = localhost
issuer=CN = localhost
---
No client certificate CA names sent
Peer signing digest: SHA256
Peer signature type: RSA-PSS
Server Temp Key: X25519, 253 bits
---
SSL handshake has read 2177 bytes and written 386 bytes
Verification error: self-signed certificate
---
```

```

0190 - 3c bd f1 65 73 ac 03 db-9b 40 70 6d fe 2e 26 f8 <..es....@pm..&.
01a0 - 06 87 d0 e1 89 8b 8c d9-09 b4 ac 84 d6 ad e7 3d .....=
01b0 - 7e 25 ff 99 ff 8a c2 18-81 c7 58 5d 2c 3c db fb ~%.....X],<..
01c0 - c8 c4 15 55 c8 fc 15 48-b5 5b 22 5b 4d 50 62 9b ...U...H.["[MPb.
01d0 - ae 04 5a 42 f2 97 6b b5-20 94 eb 04 cc 32 7a bc ..ZB..k. ....2z.
01e0 - f5 e2 0e 79 79 33 1a 51-1b b8 2b ff 41 27 ef 2c ...yy3.Q...+A'..
01f0 - ac 42 01 dc 89 0d 52 8a-34 9e ee 1f e6 d2 68 97 .B....R.4.....h.
0200 - 82 86 03 3b 8e 2e ce 91-13 22 f3 4c 7f bc 52 c5 ...;.....".L..R.
0210 - 6c 75 34 8b 74 e1 38 7e-ca 2b 89 f7 97 c2 5a 91 lu4.t.8~+....Z.
0220 - 3a aa a5 02 c1 22 75 3a-a8 62 7a 29 37 bc d3 91 :...."u:.bz)7...
0230 - 84 d1 68 77 76 4c 5d f8-12 80 88 90 15 cb 1d 55 ..hvvL].....U
0240 - 30 bc 49 25 99 da 2a 21-d4 32 68 f2 db 75 3e 43 0.I%...*!.2h..u>C
0250 - 7a 9d 1a 9d e8 85 88 c4-58 cb 79 b8 ff 32 83 fe Z.....X.y..2..
0260 - a9 d0 bf 22 e6 d4 76 ab-b3 1c 09 4b ce 3e 4e f2 ...".v....K.>N.
0270 - 4b 25 f6 58 f1 f1 28 53-d5 fb 73 8f 8c 24 b9 09 K%.X..(S...$.
0280 - 52 07 41 47 b0 37 8b 4b-f8 db 2e f2 a6 4b 53 35 R.AG.7.K.....KS5
0290 - 8e 7d 7c 83 ba dc ff ec-7d c3 8f 09 95 88 0d 3b .}|.....}......;
02a0 - 10 09 73 1f 2b c4 1c 25-e8 35 db 1f cb 52 1d c6 ..s.+...%.5...R..
02b0 - ff 54 59 36 c7 8f 04 1e-d6 0a 49 ca ba 7e 44 56 .TY6.....I...~DV
02c0 - e8 d3 4e 91 f4 94 ea 2e-4e 0a 34 a3 f5 bb 37 4a ..N....N.4....7J
02d0 - ea eb 20 3e b5 0f 78 2b-a7 39 8a d3 5e c1 4b fa ..>...x+.9..^..K.
02e0 - 36 72 99 21 f9 30 88 cb-07 eb 2b 6e bc b9 86 ae 6r.!.0....+n....
02f0 - 9e 1d 41 3b 96 22 b0 a6-d8 51 b7 35 82 f1 f2 60 ..A;"....Q.5....`
0300 - 34 7b ca b0 14 8d 77 cc-81 1a ff 6d df ee 7c 8d 4{...w....m...|.
0310 - f8 e1 ea 43 8e 6b 1f fb-03 57 e4 7f 74 bd 62 14 ...C.k...W..t.b.
0320 - f5 a4 04 35 26 c5 d5 8a-da 8b 30 de 74 cb e6 74 ...5&.....0.t..t
0330 - aa fe 1b 1c 2f 22 b1 c9-5e 7b d6 92 2a 28 f3 ff ...."/"..^{...*(..
0340 - f2 b6 59 e5 fb 88 57 fb-f6 1d a4 74 77 1b d0 8f ..Y...W....tw...
0350 - bc 1e 0f 15 e1 ea d2 1b-d9 05 85 d0 3e b0 0b 12 .....>...
0360 - ce 83 83 ff 92 9f eb 03-29 3d 26 12 e0 c7 ec 30 .....)=&.....0
0370 - 07 5b b3 10 cd d1 7f 1a-b6 a3 a9 46 e2 85 20 d2 .[.....F...
0380 - 40 a9 8b 36 22 a2 13 e7-90 3c 7a 55 64 53 9e 98 @..6".....<zUdS..
0390 - 45 dc d9 c1 9d f6 eb 60-be 93 03 c3 9e 02 f9 6c E.....`.....l
03a0 - aa 9f e1 62 24 02 1f bb-6b ac ad 76 ff 5b 09 20 ...b$...k..v.[.
03b0 - 2b c2 a4 14 19 e3 ed b9-c8 93 e6 ae 85 00 1b 45 +.....E
03c0 - 7d a0 92 c1 f4 82 e8 91-1f d2 51 2e a7 fd f4 ea }......Q.....
03d0 - 59 0a 36 a7 c5 b8 17 Y.6....

```

```

Start Time: 1713551208
Timeout : 7200 (sec)
Verify return code: 18 (self-signed certificate)
Extended master secret: yes

```

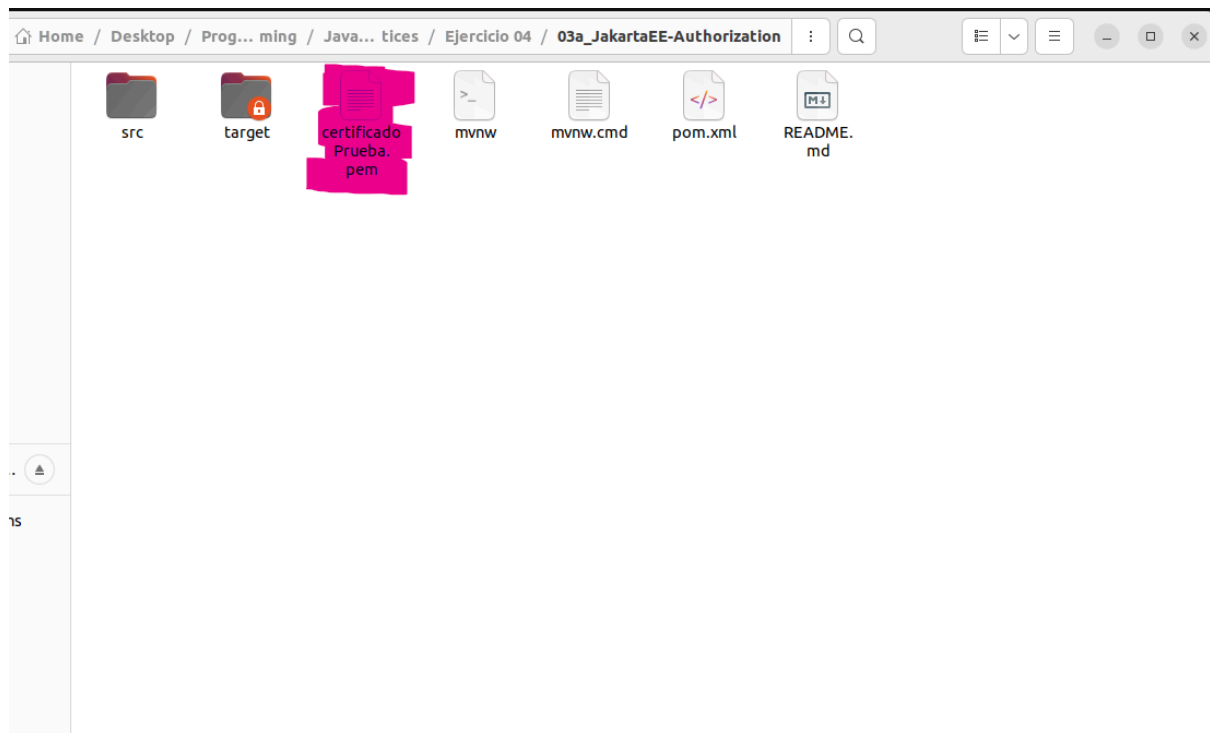
Seguidamente, podemos pasar a generar un archivo de certificado.pem con el siguiente comando: **openssl s\_client -showcerts -connect localhost:8443 </dev/null | sed -n -e '/-.BEGIN/,/-.END/ p' > certificadoPrueba.pem**

```

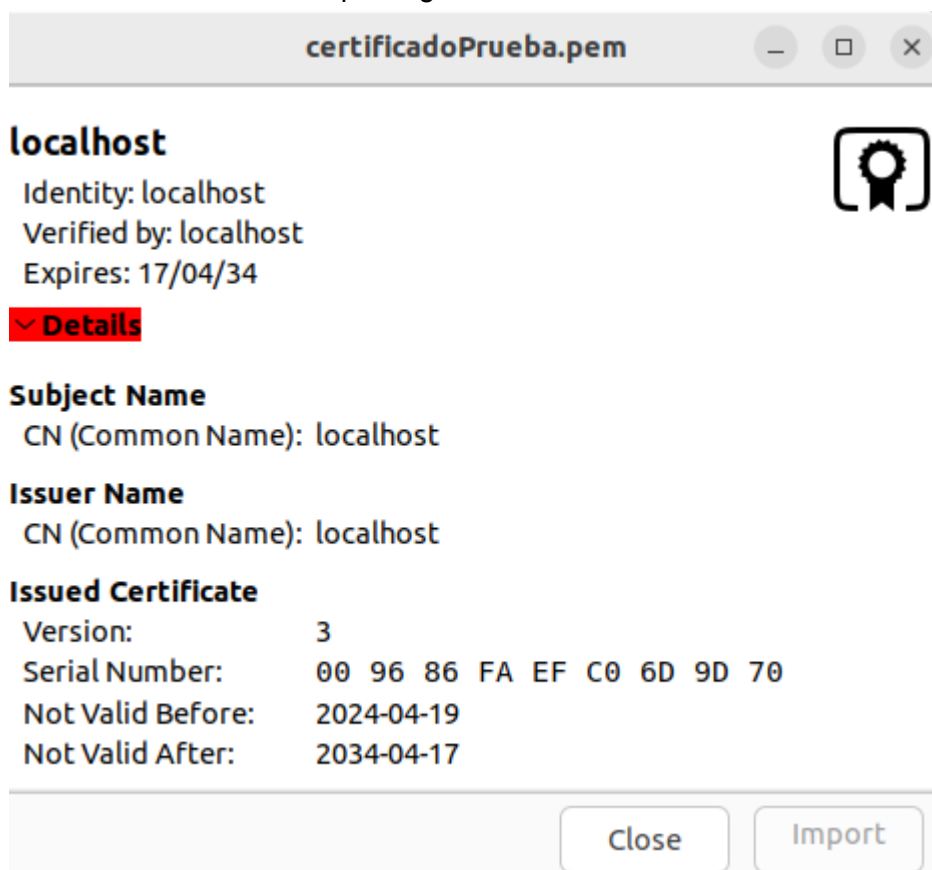
root@Artorias:/home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/03a_JakartaEE-Authorization# openssl s_client -showcerts -connect localhost:8443 </dev/null | sed -n -e '/-.BEGIN/,/-.END/ p' > certificadoPrueba.pem
Can't use SSL_get_servername
depth=0 CN = localhost
verify error:num=18:self-signed certificate
verify return:1
depth=0 CN = localhost
verify return:1
DONE
root@Artorias:/home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/03a_JakartaEE-Authorization# s

```

Ahi vemos que termina de crearlo y de ahi, podemos chequear si el archivo .pem fue creado.



Y de esa manera vemos que lo genero.



Podemos ver los detalles del certificado , la identidad, verificado por, y cuando expira.

8) Una vez aqui, tratamos de ejecutar el comando pedido y podremos ver lo siguiente.

```
root@Artorias: /home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/03a_JakartaEE-Authorizat...
root@Artorias:/home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/03a_JakartaEE-Authorization# curl --cert
certificadoPrueba.pem --user usr3:1212 -v http://localhost:8080/03a_JakartaEE-Authorization/mensajes/gerente/enviarMensa
je?mensaje=hola
```

Una vez puesto el comando, lo ejecutamos.

```
root@Artorias: /home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/03a_JakartaEE-Authorizat...
root@Artorias:/home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/03a_JakartaEE-Authorization# curl --cert
certificadoPrueba.pem --user usr3:1212 -v http://localhost:8080/03a_JakartaEE-Authorization/mensajes/gerente/enviarMensa
je?mensaje=hola
  Trying 127.0.0.1:8080...
  Connected to localhost (127.0.0.1) port 8080 (#0)
  Server auth using Basic with user 'usr3'
  GET /03a_JakartaEE-Authorization/mensajes/gerente/enviarMensaje?mensaje=hola HTTP/1.1
  Host: localhost:8080
  Authorization: Basic dXNyMzoxMjEy
  User-Agent: curl/7.81.0
  Accept: */*

  Mark bundle as not supporting multiuse
  HTTP/1.1 200 OK
  Expires: 0
  Connection: keep-alive
  Cache-Control: no-cache, no-store, must-revalidate
  Pragma: no-cache
  Content-Type: application/json
  Content-Length: 63
  Date: Tue, 23 Apr 2024 14:27:57 GMT

mensaje enviado por gerente: El contenido del mensaje es: hola
  Connection #0 to host localhost left intact
root@Artorias:/home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/03a_JakartaEE-Authorization#
```

Vemos que nos devuelve 200OK con el usr3 que tiene rol de gerente.

## Parte B)

- 1) Para realizar esta parte, podemos crear algo similar a MensajeGerenteApi donde quedara de la siguiente manera. MensajeEmpleadoApi:



```

1 package ejemplo01;
2
3 import ejemplo00.aplicacion.MensajeServicios;
4 import jakarta.annotation.security.RolesAllowed;
5 import jakarta.enterprise.context.ApplicationScoped;
6 import jakarta.inject.Inject;
7 import jakarta.ws.rs.GET;
8 import jakarta.ws.rs.Path;
9 import jakarta.ws.rs.Produces;
10 import jakarta.ws.rs.QueryParam;
11 import jakarta.ws.rs.core.MediaType;
12
13 @ApplicationScoped
14 @Path("/Empleado")
15 public class MensajeEmpleadoApi {
16
17     @Inject
18     private MensajeServicios servicio;
19
20     //curl -v http://localhost:8080/03a_JakartaEE-Authorization/mensajes/Empleado/enviarMensaje?mensaje=HolaMundo
21     //con la url anterior devuelve 401 no autorizado ya que no se envían las credenciales
22
23     //curl --user pepe:contrasenia2024 -v http://localhost:8080/03a_JakartaEE-Authorization/mensajes/Empleado/enviarMensaje?mensaje=HolaMundo
24     //observar que se envían las credenciales
25
26     @GET
27     @Path("/enviarMensaje")
28     //invocar este método
29     @Produces({ MediaType.APPLICATION_JSON })
30     public String enviarMensajeComoEmpleado(@QueryParam("mensaje") String mensaje) {
31         return servicio.enviarMensajeComoEmpleado(mensaje);
32     }
33
34
35     //curl --cacert certificadoPrueba.pem --user pepe:contrasenia2024 -v https://localhost:8443/03a_JakartaEE-Authorization/mensajes/Empleado/enviarMensaje?mensaje=HolaMundo
36     //Nota 1: observar como se incluye el certificado .pem para poder conectarse con ssl
37     //Este archivo contiene la llave pública del servidor
38     //Nota 2: notar que el puerto de comunicación es 8443
39     //Nota 3: los certificados del servidor (.pem) los obtuvimos con la herramienta openssl
40
41     //openssl s_client -showcerts -connect localhost:8443 </dev/null | sed -n -e '/-.BEGIN/,/-.END/ p' > certificadoPrueba.pem
42     //genera archivo pem con los certificados del servidor
43     @GET
44     @Path("/generarCertificadoPrueba")

```

Luego debemos ir a nuestro web.xml para cambiar las restricciones de seguridad, donde cambiaremos para autenticar al empleado , únicamente con el rol de empleado.

Quedara de la siguiente manera

```
1 <web-app version="6.0"
2   xmlns="https://jakarta.ee/xml/ns/jakartaee"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xsi:schemaLocation="https://jakarta.ee/xml/ns/jakartaee https://jakarta.ee/xml/ns/jakartaee/web-app_6_0.xsd">
5
6   <!-- restricción de seguridad #1 -->
7   <security-constraint>
8     <web-resource-collection>
9       <web-resource-name>Mensajes autenticados</web-resource-name>
10      <url-pattern>/mensajes/gerente/*</url-pattern>
11    </web-resource-collection>
12    <auth-constraint>
13      <role-name>gerente</role-name> <!-- solo los usuarios con el rol gerente podrán acceder a la url -->
14    </auth-constraint>
15  </security-constraint>
16
17  <!-- restricción de seguridad #2 -->
18  <security-constraint>
19    <web-resource-collection>
20      <web-resource-name>Mensajes autenticados</web-resource-name>
21      <url-pattern>/mensajes/empleado/*</url-pattern>
22    </web-resource-collection>
23    <auth-constraint>
24      <role-name>empleado</role-name> <!-- solo los usuarios con el rol empleado podrán acceder a la url -->
25    </auth-constraint>
26  </security-constraint>
27
28  <!-- restricción de seguridad #3 -->
29  <security-constraint>
30    <web-resource-collection>
31      <web-resource-name>Mensajes autenticados cifrados</web-resource-name>
32      <url-pattern>/mensajes/gerente/enviarMensajeSeguro</url-pattern>
33    </web-resource-collection>
34    <auth-constraint>
35      <role-name>gerente</role-name> <!-- solo los usuarios con el rol gerente podrán acceder a la url -->
36    </auth-constraint>
37
38    <!-- la información viaja cifrada ssl utilizando un certificado generado por el servidor
39    solo no apto para uso en producción -->
40    <user-data-constraint>
41      <transport-guarantee>CONFIDENTIAL</transport-guarantee>
42    </user-data-constraint>
43  </security-constraint>
44
```

Luego podemos probar nuestro Curl, podemos ver que si intentamos con alguno con rol gerente, nos va a arrojar forbidden.

```
root@Artorias: /home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/03a_JakartaEE-Auth...
root@Artorias:/home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04
/03a_JakartaEE-Authorization# curl --cert certificadoPrueba.pem --user usr3:1212 -v http://localhost:8080/03a_Jakarta
EE-Authorization/mensajes/empleado/enviarMensaje?mensaje=hola
* Trying 127.0.0.1:8080...
* Connected to localhost (127.0.0.1) port 8080 (#0)
* Server auth using Basic with user 'usr3'
> GET /03a_JakartaEE-Authorization/mensajes/empleado/enviarMensaje?mensaje=hola HTTP/1.1
> Host: localhost:8080
> Authorization: Basic dXNyMzoxMjEy
> User-Agent: curl/7.81.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 403 Forbidden
< Expires: 0
< Connection: keep-alive
< Cache-Control: no-cache, no-store, must-revalidate
< Pragma: no-cache
< Content-Type: text/html; charset=UTF-8
< Content-Length: 68
< Date: Tue, 23 Apr 2024 16:25:34 GMT
<
* Connection #0 to host localhost left intact
<html><head><title>Error</title></head><body>Forbidden</body></html>root@Artoriasroot@root@root@Arroot@rooro
root@Artorias:/home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/03a_JakartaEE-Authorization#
```

Pero si intentamos con usr1 que es empleado:

```
root@Artorias: /home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/03a_JakartaEE-Auth...
root@Artorias: /home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/03a_JakartaEE-Auth...# curl --cert certificadoPrueba.pem --user usr1:1 -v http://localhost:8080/03a_JakartaEE-Auth.../mensajes/empleado/enviarMensaje?mensaje=hola
* Trying 127.0.0.1:8080...
* Connected to localhost (127.0.0.1) port 8080 (#0)
* Server auth using Basic with user 'usr1'
> GET /03a_JakartaEE-Auth.../mensajes/empleado/enviarMensaje?mensaje=hola HTTP/1.1
> Host: localhost:8080
> Authorization: Basic dXNyMT0x
> User-Agent: curl/7.81.0
> Accept: */*
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< Expires: 0
< Connection: keep-alive
< Cache-Control: no-cache, no-store, must-revalidate
< Pragma: no-cache
< Content-Type: application/json
< Content-Length: 64
< Date: Tue, 23 Apr 2024 16:34:32 GMT
<
Mensaje enviado por empleado: El contenido del mensaje es: hola
* Connection #0 to host localhost left intact
root@Artorias: /home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/03a_JakartaEE-Auth...
```

## Parte C)

Para la ultima parte, una vez parados en la carpeta raiz de nuestra api, podemos correr el comando `mvn clean package wildfly:dev`. Y eso hara deploy de nuestro servidor.

```
root@Artorias: /home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/primera-api
$ java:JmsXA to alias java:jboss/DefaultJMSConnectionFactory
5:26:29,120 INFO [org.jboss.as.server] (Controller Boot Thread) WFLYSRV0212: Resuming server
5:26:29,123 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0060: Http management interface listening on http://127.0.0.1:9990/management
5:26:29,123 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0051: Admin console listening on http://127.0.0.1:9990
5:26:29,125 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0025: WildFly Full 30.0.1.Final (WildFly Core 22.0.2.Final) started in 2800ms - Started 318 of 560 services (332 services are lazy, passive or on-demand) - Server configuration file in use: standalone-full.xml
[WARNING] The server may be in an unexpected state for further interaction. The current state is failed
5:26:29,396 INFO [org.jboss.as.repository] (management-handler-thread - 1) WFLYDR0001: Content added at location /home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/primera-api/target/server/standalone/data/content/de/6a75d147f006e2a8f8840f30aaa325113fff/content
5:26:29,409 INFO [org.jboss.as.server.deployment] (MSC service thread 1-1) WFLYSRV0027: Starting deployment of "EmpleadosAPI.war" (runtime-name: "EmpleadosAPI.war")
5:26:30,900 INFO [org.jboss.weld.deployer] (MSC service thread 1-1) WFLYWELD0003: Processing weld deployment EmpleadosAPI.war
5:26:30,989 INFO [org.hibernate.validator.internal.util.Version] (MSC service thread 1-1) HV000001: Hibernate Validator 8.1.1.Final
5:26:31,187 WARN [org.jboss.as.jaxrs] (MSC service thread 1-6) WFLYRS0018: Explicit usage of Jackson annotation in a Jakarta RESTful Web Services deployment; the system will disable Jakarta JSON Binding processing for the current deployment. Consider setting the 'resteasy.preferJacksonOverJsonB' property to 'false' to restore Jakarta JSON Binding.
5:26:31,207 INFO [org.jboss.weld.Version] (MSC service thread 1-6) WELD-000900: 5.1.2 (Final)
5:26:31,986 INFO [org.wildfly.extension.undertow] (ServerService Thread Pool -- 83) WFLYUT0021: Registered web context: '/EmpleadosAPI' for server 'default-server'
5:26:32,024 INFO [org.jboss.as.server] (management-handler-thread - 1) WFLYSRV0010: Deployed "EmpleadosAPI.war" (runtime-name: "EmpleadosAPI.war")
5:26:50,694 WARN [org.jboss.as.weld] (default task-1) WFLYWELD0052: Using deployment classloader to load proxy classes for module com.fasterxml.jackson.jakarta.jackson-jakarta-json-provider. Package-private access will not work. To fix this the module should declare dependencies on [org.jboss.weld.core, org.jboss.weld.spi, org.jboss.weld.api]
5:26:50,850 INFO [stdout] (default task-1) Retornando todos los empleados
5:26:50,851 INFO [stdout] (default task-1) Invocando PostConstruct
```

Parecido a lo de los primeros ejercicios, necesitamos ir al `web.xml` para generar las restricciones posibles para los roles, por ejemplo, gerente y recursos humanos

```

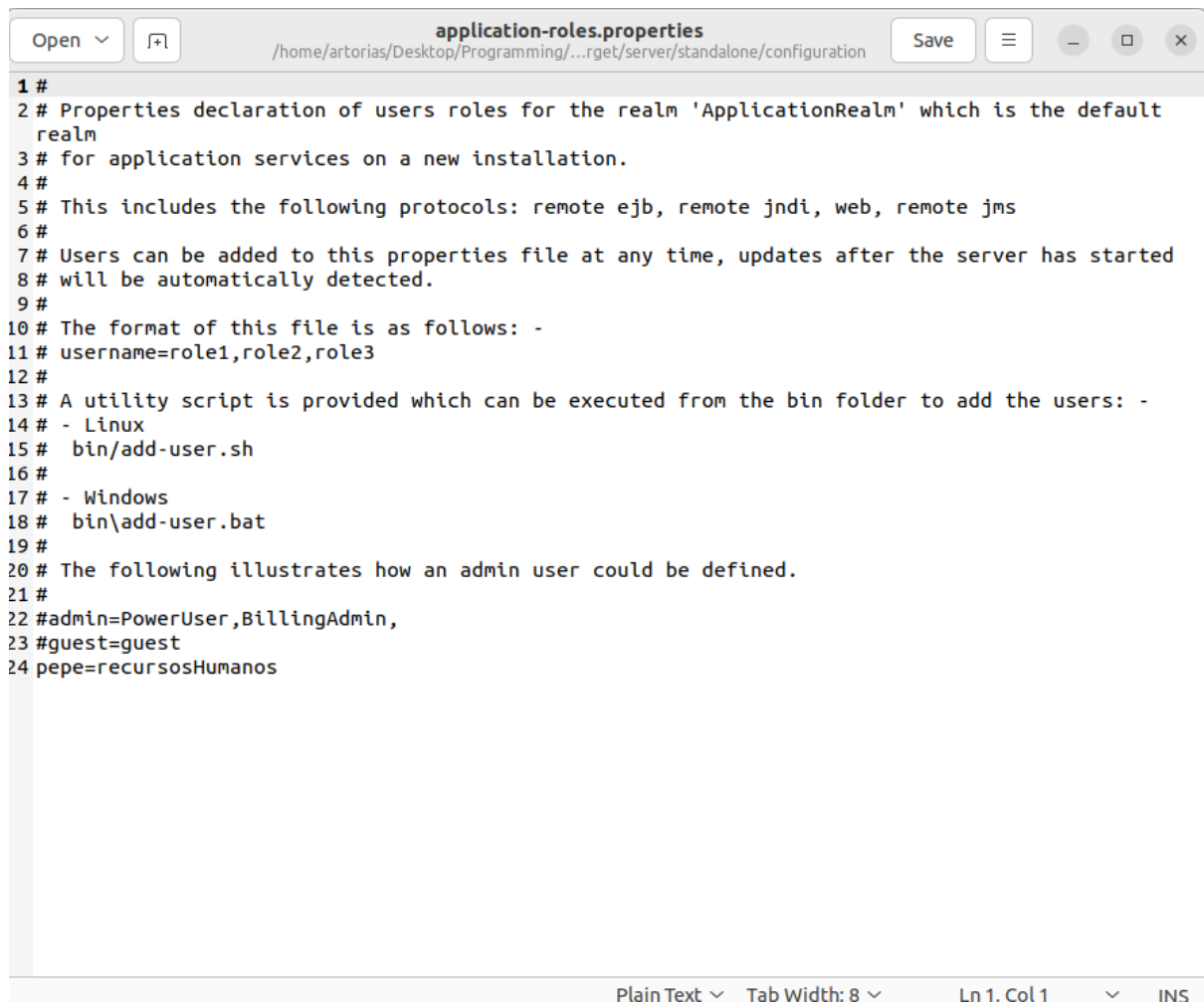
    <servlet-name>jakarta.ws.rs.core.Application</servlet-name>
  </servlet>

  <!-- restricción de seguridad #1 -->
  <security-constraint>
    <web-resource-collection>
      <web-resource-name>Mensajes autenticados</web-resource-name>
      <url-pattern>/api/empleados/gerente/*</url-pattern>
    </web-resource-collection>
    <auth-constraint>
      <role-name>gerente</role-name> <!-- solo los usuarios con el r
    </auth-constraint>
    <user-data-constraint>
      <transport-guarantee>CONFIDENTIAL</transport-guarantee>
    </user-data-constraint>
  </security-constraint>

  <!-- restricción de seguridad #2 -->
  <security-constraint>
    <web-resource-collection>
      <web-resource-name>Mensajes autenticados</web-resource-name>
      <url-pattern>/api/empleados/recursosHumanos/*</url-pattern>
    </web-resource-collection>
    <auth-constraint>
      <role-name>recursosHumanos</role-name> <!-- solo los usuarios
    </auth-constraint>
    <user-data-constraint>
      <transport-guarantee>CONFIDENTIAL</transport-guarantee>
    </user-data-constraint>
  </security-constraint>

```

Visto esto, también, de la misma manera que la vez pasada creamos un rol, lo haremos aquí, en este caso, cree un usuario que se llama 'pepe' con el rol recursosHumanos.



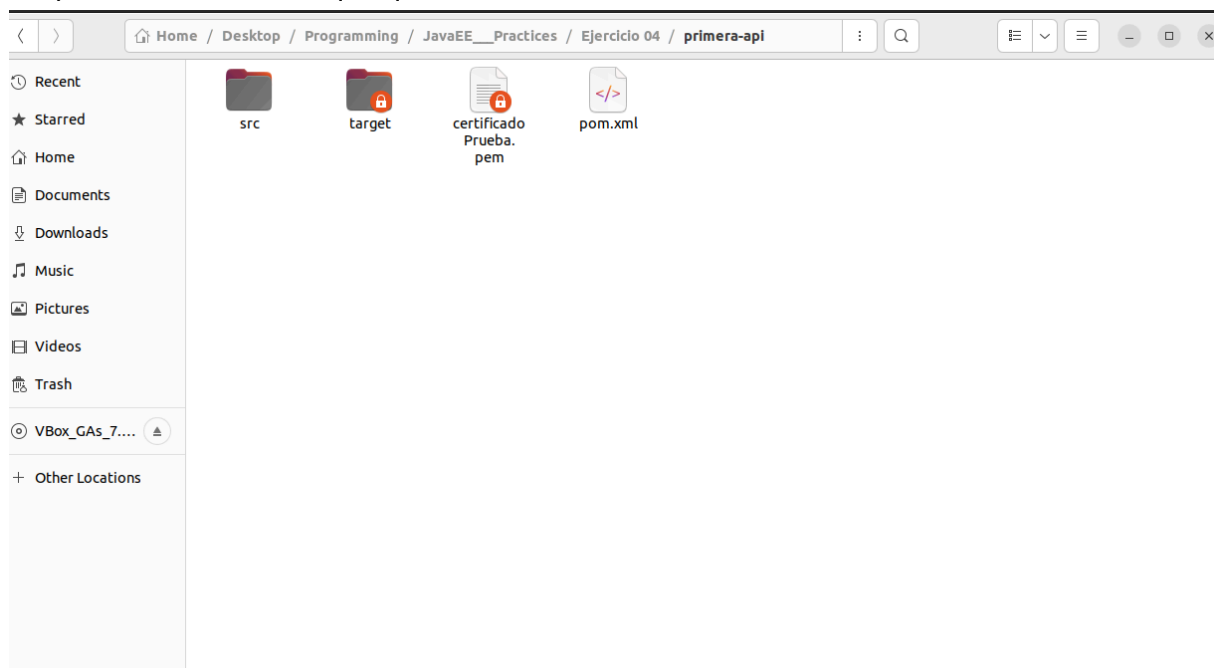
```
1 #
2 # Properties declaration of users roles for the realm 'ApplicationRealm' which is the default
  realm
3 # for application services on a new installation.
4 #
5 # This includes the following protocols: remote ejb, remote jndi, web, remote jms
6 #
7 # Users can be added to this properties file at any time, updates after the server has started
8 # will be automatically detected.
9 #
10 # The format of this file is as follows: -
11 # username=role1,role2,role3
12 #
13 # A utility script is provided which can be executed from the bin folder to add the users: -
14 # - Linux
15 #   bin/add-user.sh
16 #
17 # - Windows
18 #   bin\add-user.bat
19 #
20 # The following illustrates how an admin user could be defined.
21 #
22 #admin=PowerUser,BillingAdmin,
23 #guest=guest
24 pepe=recursosHumanos
```

Plain Text ▾ Tab Width: 8 ▾ Ln 1. Col 1 ▾ INS

Luego debemos generar el certificado para nuestra api.  
Corremos el comando anterior.

```
root@Artorias: /home/artorias/Desktop/Programming/JavaE... Practices/Ejercicio 04
/primer-api# openssl s_client -showcerts -connect localhost:8443 </dev/null | s
ed -n -e '/-BEGIN/,/-.END/ p' > certificadoPrueba.pem
```

Ahi podemos corroborar que quedo creado



Luego procedemos a hacer las pruebas en consola con curl.

Ejecutamos el siguiente comando a modo de ejemplo para probar una solicitud no autorizada.

```
curl --cacert certificadoPrueba.pem --user gerente:1234 -v -X POST -H
"Content-Type: application/json" -d
'{"id":23,"nombre":"Messi","cedula":"123456789"}'
https://localhost:8443/EmpleadosAPI/api/empleados/recursosHumanos/
```



```

root@Artorias: /home/artorias/Desktop/Programming/JavaEE___Practices/Ejercicio 04/primer-api
TLSv1.2 (IN), TLS header, Certificate Status (22):
TLSv1.2 (IN), TLS handshake, Finished (20):
SSL connection using TLSv1.2 / ECDHE-RSA-AES256-GCM-SHA384
ALPN, server accepted to use h2
Server certificate:
  subject: CN=localhost
  start date: Apr 23 19:23:28 2024 GMT
  expire date: Apr 21 19:23:28 2034 GMT
  common name: localhost (matched)
  issuer: CN=localhost
  SSL certificate verify ok.
Using HTTP2, server supports multiplexing
Connection state changed (HTTP/2 confirmed)
Copying HTTP/2 data in stream buffer to connection buffer after upgrade: len=0
TLSv1.2 (OUT), TLS header, Supplemental data (23):
TLSv1.2 (OUT), TLS header, Supplemental data (23):
TLSv1.2 (OUT), TLS header, Supplemental data (23):
Server auth using Basic with user 'usr3'
Using Stream ID: 1 (easy handle 0x6440d82e7eb0)
TLSv1.2 (OUT), TLS header, Supplemental data (23):
POST /EmpleadosAPI/api/empleados/recursosHumanos/ HTTP/2
Host: localhost:8443
authorization: Basic dXNyMzox
user-agent: curl/7.81.0
accept: */*
content-type: application/json
content-length: 53

TLSv1.2 (OUT), TLS header, Supplemental data (23):
We are completely uploaded and fine
TLSv1.2 (IN), TLS header, Supplemental data (23):
Connection state changed (MAX_CONCURRENT_STREAMS == 4294967295)!
TLSv1.2 (OUT), TLS header, Supplemental data (23):
TLSv1.2 (IN), TLS header, Supplemental data (23):
TLSv1.2 (IN), TLS header, Supplemental data (23):
HTTP/2 403
expires: 0
cache-control: no-cache, no-store, must-revalidate
pragma: no-cache
content-type: text/html; charset=UTF-8
content-length: 68
date: Tue, 23 Apr 2024 20:45:36 GMT

Connection #0 to host localhost left intact
<html><head><title>Error</title></head><body>Forbidden</body></html>root@Artoriasroot@Arroot@Artorroot@Arootrooroot@Artorias: /home/artorias/Desktop/Programming/JavaEE___Practices/Ejercicio 04/primer-api# ^C

```

Vemos que como resultado nos arroja 403, que es acceso no autorizado.

Probemos ahora con un user de recursosHumanos (pepe).

```

root@Artorias: /home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/primer-api
* TLSv1.2 (IN), TLS header, Finished (20):
* TLSv1.2 (IN), TLS header, Certificate Status (22):
* TLSv1.2 (IN), TLS handshake, Finished (20):
* SSL connection using TLSv1.2 / ECDHE-RSA-AES256-GCM-SHA384
* ALPN, server accepted to use h2
* Server certificate:
*  subject: CN=localhost
*  start date: Apr 23 19:23:28 2024 GMT
*  expire date: Apr 21 19:23:28 2034 GMT
*  common name: localhost (matched)
*  issuer: CN=localhost
*  SSL certificate verify ok.
* Using HTTP2, server supports multiplexing
* Connection state changed (HTTP/2 confirmed)
* Copying HTTP/2 data in stream buffer to connection buffer after upgrade: len=0
* TLSv1.2 (OUT), TLS header, Supplemental data (23):
* TLSv1.2 (OUT), TLS header, Supplemental data (23):
* TLSv1.2 (OUT), TLS header, Supplemental data (23):
* Server auth using Basic with user 'pepe'
* Using Stream ID: 1 (easy handle 0x556f4e4e2eb0)
* TLSv1.2 (OUT), TLS header, Supplemental data (23):
* POST /EmpleadosAPI/api/empleados/recursosHumanos/ HTTP/2
* Host: localhost:8443
* authorization: Basic cGVwZTox
* user-agent: curl/7.81.0
* accept: */*
* content-type: application/json
* content-length: 53
* TLSv1.2 (OUT), TLS header, Supplemental data (23):
* We are completely uploaded and fine
* TLSv1.2 (IN), TLS header, Supplemental data (23):
* Connection state changed (MAX_CONCURRENT_STREAMS == 4294967295)!
* TLSv1.2 (OUT), TLS header, Supplemental data (23):
* TLSv1.2 (IN), TLS header, Supplemental data (23):
* TLSv1.2 (IN), TLS header, Supplemental data (23):
* HTTP/2 200
* expires: 0
* cache-control: no-cache, no-store, must-revalidate
* pragma: no-cache
* content-type: application/json
* content-length: 75
* date: Tue, 23 Apr 2024 20:48:15 GMT
* Connection #0 to host localhost left intact
* {"nombre":"Harri Poler","id":3,"cedula":"1212121212","tareEmpleado":null}root@Artorias: /home/artorias/Desktop/Prog

```

Como podemos ver, con un user de recursos humanos, la petición es exitosa, arrojando código 200.

Siguiente probamos con baja de empleado, misma metodología, ejecutamos el siguiente comando.

```
curl --cacert certificadoPrueba.pem --user usr1:1 -v -X DELETE
https://localhost:8443/EmpleadosAPI/api/empleados/recursosHumanos/1
```

Primero intentamos con el usr1

```

root@Artorias: /home/artorias/Desktop/Programming/JavaEE__Practices/Ejercicio 04/primera-api
TLSv1.2 (OUT), TLS header, Finished (20):
TLSv1.2 (OUT), TLS change cipher, Change cipher spec (1):
TLSv1.2 (OUT), TLS header, Certificate Status (22):
TLSv1.2 (OUT), TLS handshake, Finished (20):
TLSv1.2 (IN), TLS header, Finished (20):
TLSv1.2 (IN), TLS header, Certificate Status (22):
TLSv1.2 (IN), TLS handshake, Finished (20):
SSL connection using TLSv1.2 / ECDHE-RSA-AES256-GCM-SHA384
ALPN, server accepted to use h2
Server certificate:
  subject: CN=localhost
  start date: Apr 23 19:23:28 2024 GMT
  expire date: Apr 21 19:23:28 2034 GMT
  common name: localhost (matched)
  issuer: CN=localhost
SSL certificate verify ok.
Using HTTP2, server supports multiplexing
Connection state changed (HTTP/2 confirmed)
Copying HTTP/2 data in stream buffer to connection buffer after upgrade: len=0
TLSv1.2 (OUT), TLS header, Supplemental data (23):
TLSv1.2 (OUT), TLS header, Supplemental data (23):
TLSv1.2 (OUT), TLS header, Supplemental data (23):
Server auth using Basic with user 'usr1'
Using Stream ID: 1 (easy handle 0x5d0f1171deb0)
TLSv1.2 (OUT), TLS header, Supplemental data (23):
DELETE /EmpleadosAPI/api/empleados/recursosHumanos/1 HTTP/2
Host: localhost:8443
authorization: Basic dXNyMT0x
user-agent: curl/7.81.0
accept: */*

TLSv1.2 (IN), TLS header, Supplemental data (23):
Connection state changed (MAX_CONCURRENT_STREAMS == 4294967295)!
TLSv1.2 (OUT), TLS header, Supplemental data (23):
TLSv1.2 (IN), TLS header, Supplemental data (23):
TLSv1.2 (IN), TLS header, Supplemental data (23):
HTTP/2 403
expires: 0
cache-control: no-cache, no-store, must-revalidate
pragma: no-cache
content-type: text/html; charset=UTF-8
content-length: 68
date: Tue, 23 Apr 2024 20:52:11 GMT

Connection #0 to host localhost left intact
html><head><title>Error</title></head><body>Forbidden</body></html>root@Artorias: /home/artorias/Desktop/Programming/
JavaEE__Practices/Ejercicio 04/primera-api#

```

403 no autorizado.

Luego intentamos cambiando el usr1 por pepe.

```
root@Artorias: /home/artorias/Desktop/Programming/JavaEE___Practices/Ejercicio 04/primera-api
* TLSv1.2 (OUT), TLS header, Finished (20):
* TLSv1.2 (OUT), TLS change cipher, Change cipher spec (1):
* TLSv1.2 (OUT), TLS header, Certificate Status (22):
* TLSv1.2 (OUT), TLS handshake, Finished (20):
* TLSv1.2 (IN), TLS header, Finished (20):
* TLSv1.2 (IN), TLS header, Certificate Status (22):
* TLSv1.2 (IN), TLS handshake, Finished (20):
* SSL connection using TLSv1.2 / ECDHE-RSA-AES256-GCM-SHA384
* ALPN, server accepted to use h2
* Server certificate:
*  subject: CN=localhost
*  start date: Apr 23 19:23:28 2024 GMT
*  expire date: Apr 21 19:23:28 2034 GMT
*  common name: localhost (matched)
*  issuer: CN=localhost
*  SSL certificate verify ok.
* Using HTTP2, server supports multiplexing
* Connection state changed (HTTP/2 confirmed)
* Copying HTTP/2 data in stream buffer to connection buffer after upgrade: len=0
* TLSv1.2 (OUT), TLS header, Supplemental data (23):
* TLSv1.2 (OUT), TLS header, Supplemental data (23):
* TLSv1.2 (OUT), TLS header, Supplemental data (23):
* Server auth using Basic with user 'pepe'
* Using Stream ID: 1 (easy handle 0x5aba96163eb0)
* TLSv1.2 (OUT), TLS header, Supplemental data (23):
> DELETE /EmpleadosAPI/api/empleados/recursosHumanos/1 HTTP/2
> Host: localhost:8443
> authorization: Basic cGVwZTox
> user-agent: curl/7.81.0
> accept: */*
>
* TLSv1.2 (IN), TLS header, Supplemental data (23):
* Connection state changed (MAX_CONCURRENT_STREAMS == 4294967295)!
* TLSv1.2 (OUT), TLS header, Supplemental data (23):
* TLSv1.2 (IN), TLS header, Supplemental data (23):
* TLSv1.2 (IN), TLS header, Supplemental data (23):
< HTTP/2 201
< expires: 0
< cache-control: no-cache, no-store, must-revalidate
< pragma: no-cache
< content-type: application/octet-stream
< content-length: 52
< date: Tue, 23 Apr 2024 20:54:46 GMT
<
* Connection #0 to host localhost left intact
El empleado con el id: 1 fue eliminado correctamenteroot@Artorias: /home/artorias/Desktop/Programming/JavaEE___Practices
```

Podemos ver que la baja fue exitosa.

Todo el código visto está en el repositorio:

[https://github.com/JuanmaPilon/JavaEE\\_\\_\\_Practices](https://github.com/JuanmaPilon/JavaEE___Practices)