qwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnm

|  |
| --- |
| Despliegue Proyecto Web Java  Actividad 3. Grupal.  19/02/2021  Raúl Montero Peregrina Javier Pérez Vivar Patricia Villanueva Polo Jaime Juárez |

ÍNDICE

[1.- Enunciado 3](#_Toc64644031)

[2.- Introducción 5](#_Toc64644032)

[3.-RAÚL 8](#_Toc64644033)

[1.-CONFIGURACIÓN TOMCAT PARA CONEXIÓN A BBDD 9](#_Toc64644034)

[2.-CONFIGURACIÓN APACHE CDN PARA OBTENER RECURSOS DESDE TOMCAT 14](#_Toc64644035)

[3.-CONFIGURACIÓN SERVIDOR FTP 15](#_Toc64644036)

[4.- HTTPS/CERTIFICADO DIGITAL APACHE 16](#_Toc64644037)

[4.- JAVIER 17](#_Toc64644038)

[5.- PATRICIA 17](#_Toc64644039)

[6.- JAIME 17](#_Toc64644040)

# 1.- Enunciado

**Proyecto Web Java**

Este proyecto consiste en una aplicación web Java EE con servlets para acceder a

* Base de Datos
* FTP
* CDN (Apache)

El proyecto se encuentra en GitHub: https://github.com/ocenteno/sample-jndi

Configuración

**1. Base de Datos**

Se debe crear un recurso ***"jdbc/poolBaseDatos"*** que contendrá los parámetros de conexión. La Base de Datos deberá popularse con la siguiente estructura:

**CREATE TABLE PERSONAS (**

**DNI VARCHAR(8) DEFAULT '00000000' NOT NULL,**

**NOMBRE VARCHAR(20),**

**APELLIDOS VARCHAR(20),**

**PRIMARY KEY (dni)**

**);**

**2. Apache CDN**

CDN es una forma habitual de separar el contenido estático de una aplicación web Consiste en ubicarlo en un servidor virtual que se puede replicar para que cada usuario descargue ese contenido del servidor más cercano a su ubicación La aplicación espera que exista un recurso ***"jndi/CDN"*** que apunte a la dirección web donde se han subido los ficheros estáticos se deberá crear ese servidor virtual y el recurso correspondiente en el Servidor de Aplicaciones

**3. FTP**

Se deberá crear un recurso ***"jndi/FTP"*** que contendrá la URL del servidor FTP con el que trabajar El servidor utiliza 2 usuarios con sus respectivas contraseñas:

* registrado:registrado para poder leer y obtener ficheros del servidor, debe tener permisos de lectura
* administrador:administrador para poder subir ficheros al servidor, debe tener permisos de escritura La aplicación trabaja únicamente con el directorio raíz del servidor, no se implementará la descarga o la subida en otra carpeta

**4. Opcional: DNS**

Opcionalmente se podrá registrar nuestro servidor en un DNS (local o público) para poder hacer uso de URLs más representativas para cada una de las piezas del sistema: Apache, FTP y Tomcat

**5. Librerías y ficheros**

1. Base de Datos

Para la conexión a Base de Datos se requieren las librerías Java correspondientes a la Base de Datos en cuestión.

Para MySQL y MariaDB se deberá utilizar: https://mvnrepository.com/artifact/mysql/mysql-connector-java

Para Oracle se deberá utilizar: https://mvnrepository.com/artifact/oracle/oracle-jdbc

Para otras Bases de Datos se deberá utilizar el JDBC correspondiente

2. Contenido

Éste fichero "contenido.zip" contiene

Una imagen que deberá desplegarse en el servidor Apache

Un video que deberá desplegarse en el servidor Apache

El propio fichero contenido.zip deberá desplegarse en el servidor Apache

3. Aplicación

Éste fichero "proyecto-java-ee.war" contiene la aplicación Java que deberá desplegarse en el servidor de aplicaciones

4. FTP

Deberá hacerse accesible en el servidor FTP el PDF que explica la resolución de la práctica y un fichero ZIP que contenga los ficheros de configuración que se han creado. Además, se deberán añadir estos ficheros de configuración al repositorio GitHub que se ha utilizado para las actividades anteriores

5. Despliegue

Se deberán publicar las librerías necesarias en la carpeta /lib del Servidor de Aplicaciones

Se deberá configurar el recurso "jdbc/poolBaseDatos" en el Contexto del Servidor de Aplicaciones

Se deberán configurar las variables de entorno "jndi/FTP" y "jndi/CDN" en el Contexto del Servidor de Aplicaciones

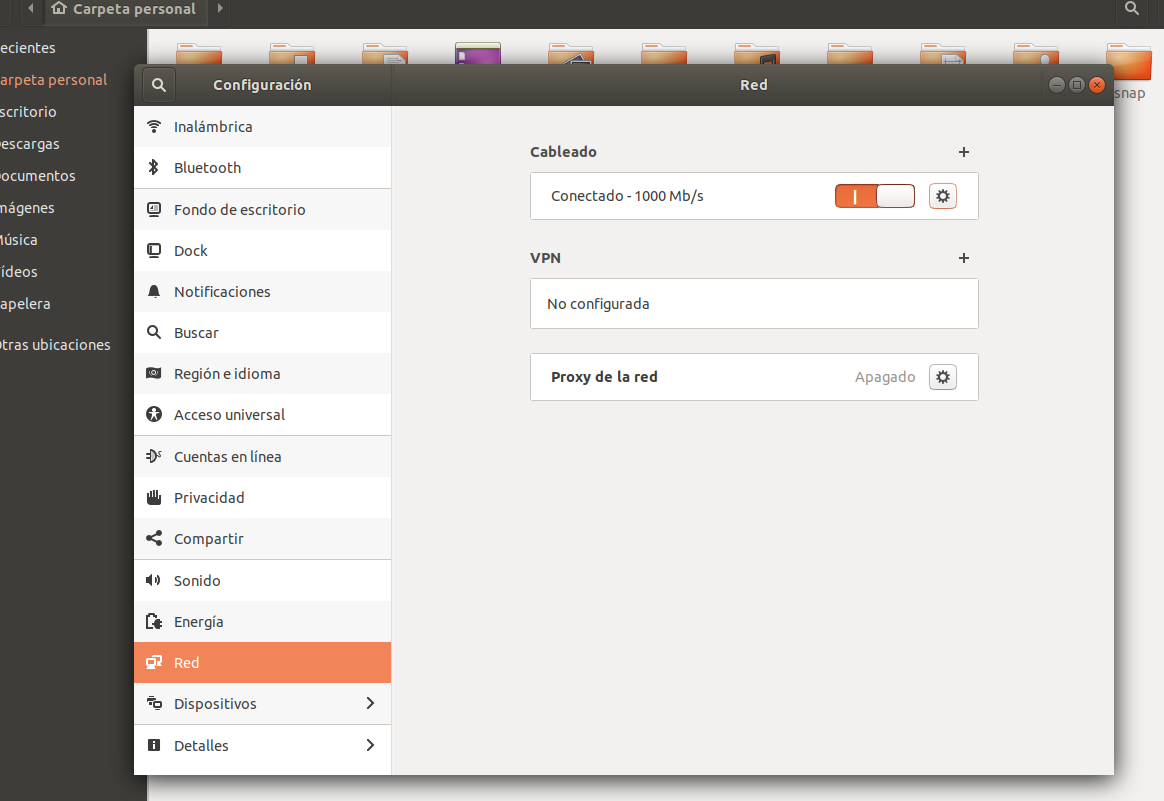
Se deberá desplegar el WAR del proyecto en el Servidor de Aplicaciones

Se deberán crear los usuarios "registrado" y "administrador" en el Servidor FTP

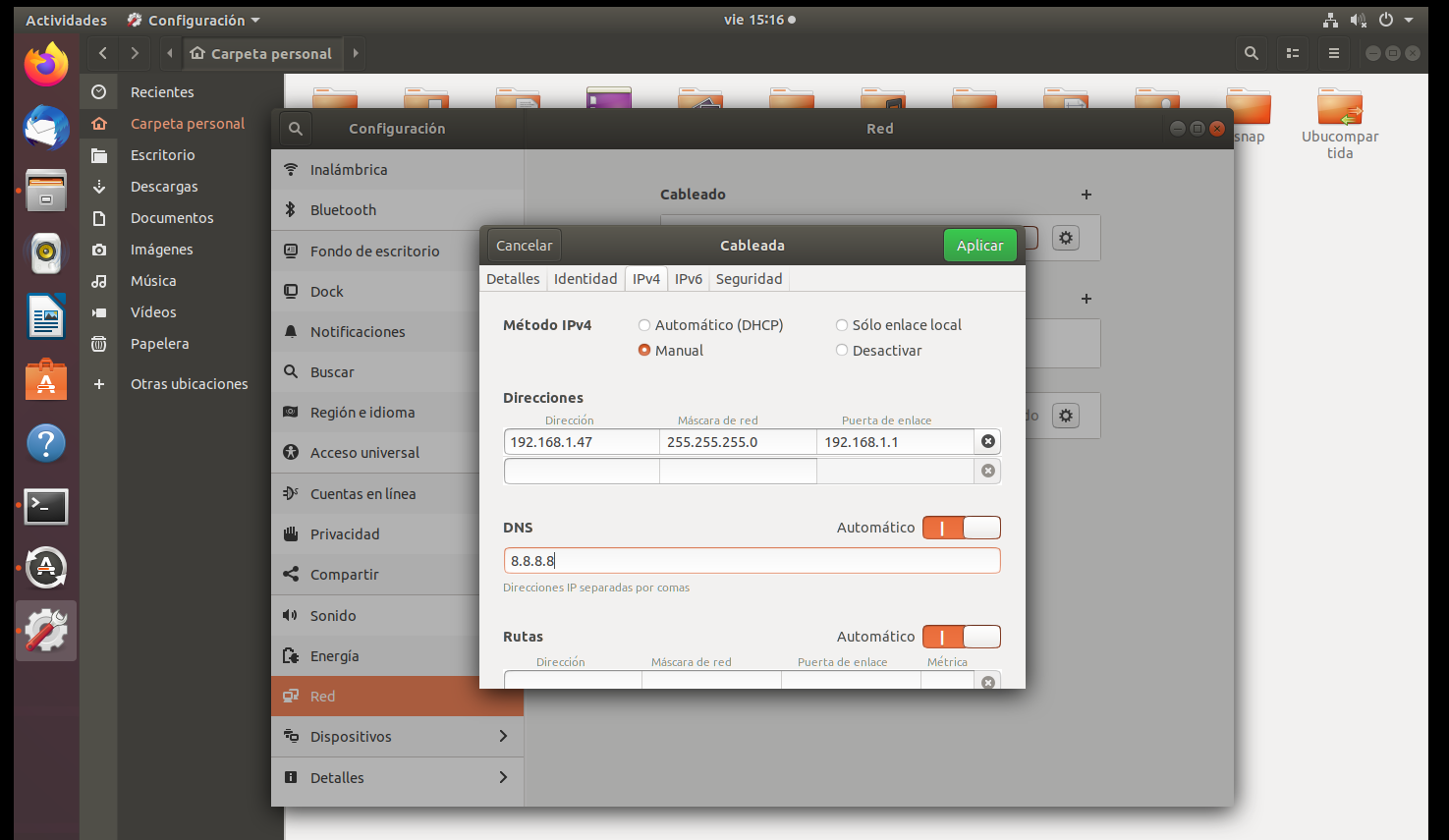
Se deberá configurar el Servidor Apache para poder acceder al directorio CDN y desplegar ahí los ficheros de prueba.

# 2.- Introducción

La idea es que los 4 componentes del equipo trabajen juntos desplegando una App Web dada sobre una misma configuración en una máquina virtual de Ubuntu. Para ello cada uno de los 4 componentes hará **fija su ip: 192.168.1.47** y así desde ahí poder arrancar nuestro Apache. Para ello accedemos a configuración-->red y en la conexión cableada pulsamos en configuración:



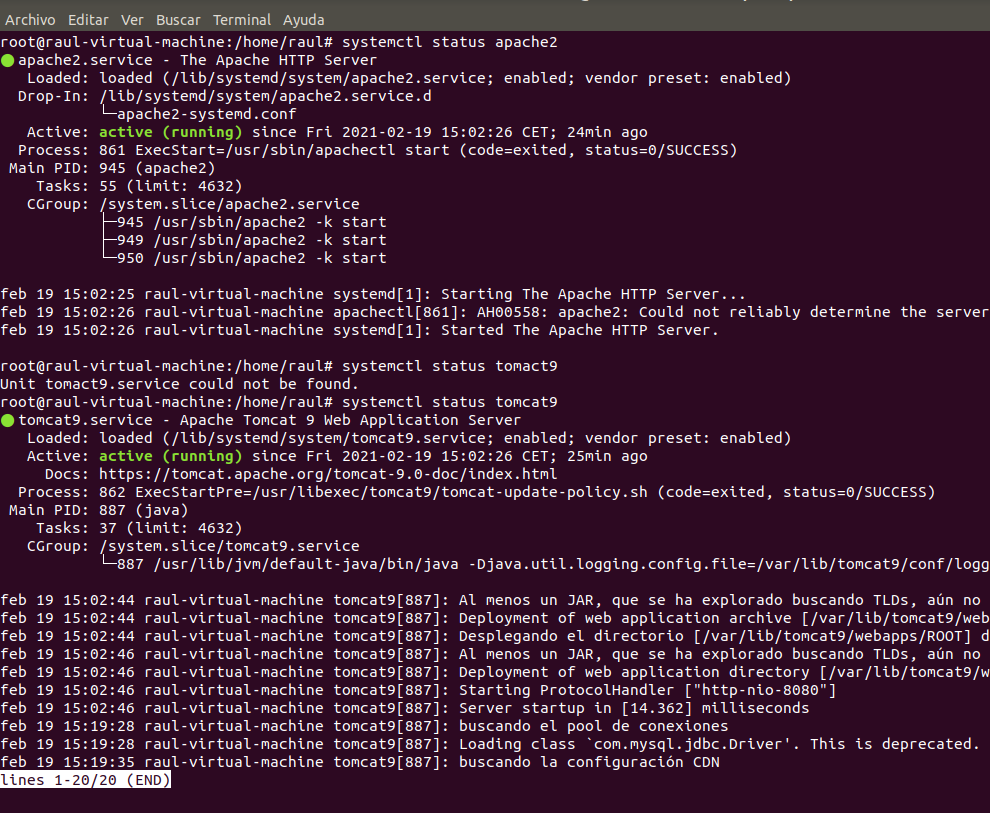
Pulsamos en la rueda de configuración y añadimos y elegimos método manual y pasamos a indicar la ip que queremos fijar, la máscara de subred y la puerta de enlace así como las DNS`S que elegiremos las de Google:



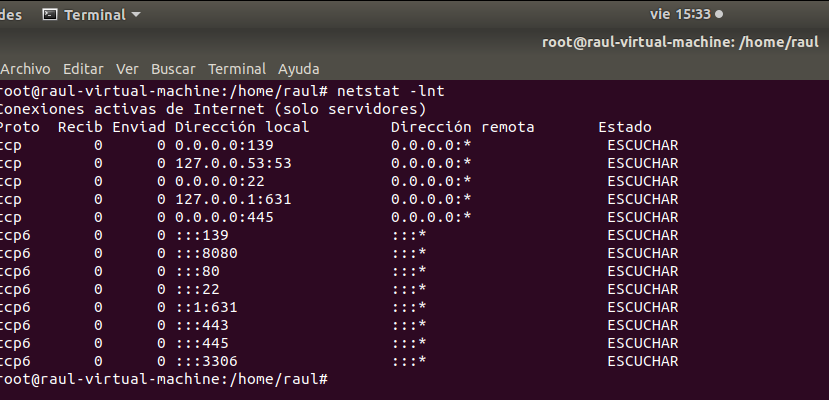
<https://elrespondon.com/como-configurar-direccion-ip-fija-linux-ubuntu/#:~:text=Para%20configurar%20una%20direcci%C3%B3n%20IP,pulsar%20el%20bot%C3%B3n%20de%20A%C3%B1adir>.

Antes de comenzar probaremos que los **servidores están activos** y que los puertos están escuchando y corrrectamente configurados en el firewall:

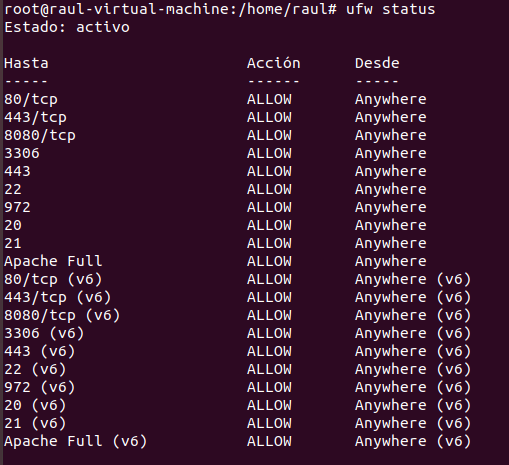
**systemctl status tomcat9 y systemctl status apache2** para ver el estado de los servidores:



Ahora **visualizamos** si nuestros **puertos** 80,8080(servidores apache y tomcat), 3306 del servidor de BBDD, 443 para conexiones https seguras y 22 para ssh **están escuchando**. Para ellos usaremos el comando **netsat –ltn** :

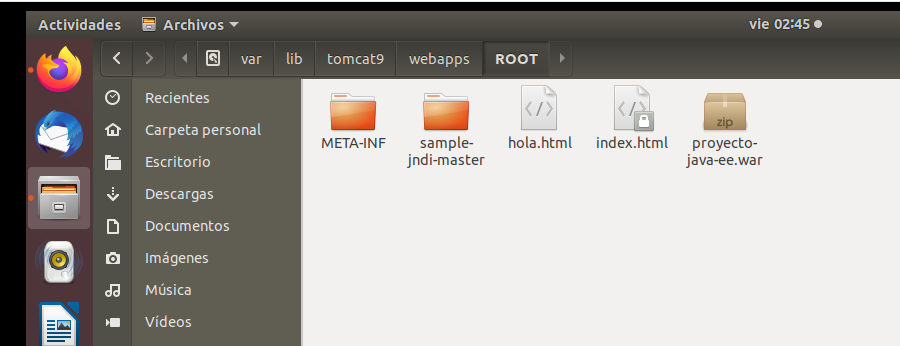


Por último comprobamos si el **firewall** permite tráfico en ellos con el **comando ufw status**:

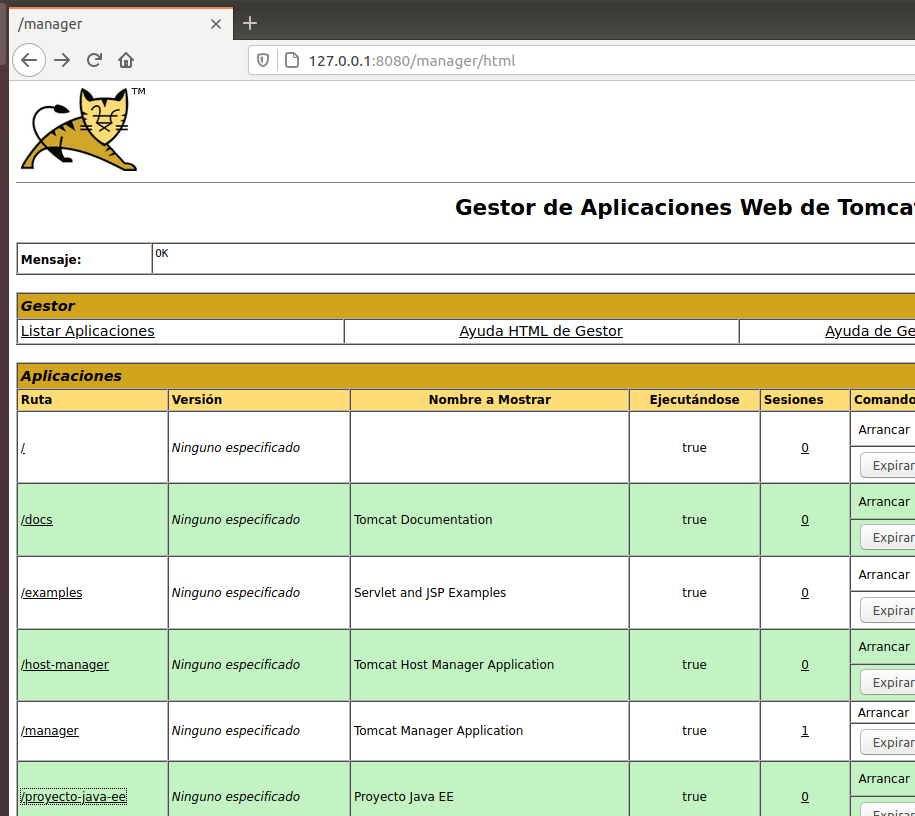


# 3.-RAÚL

Comenzaremos desplegando la aplicación .war en el directorio de Tomcat donde se han de desplegar las aplicaciones: var—lib—tomcat9—webapps--ROOT



Para comprobar que está correctamente desplegada y que podemos acceder a dicha app accedemos al manager de nuestro Tomcat:



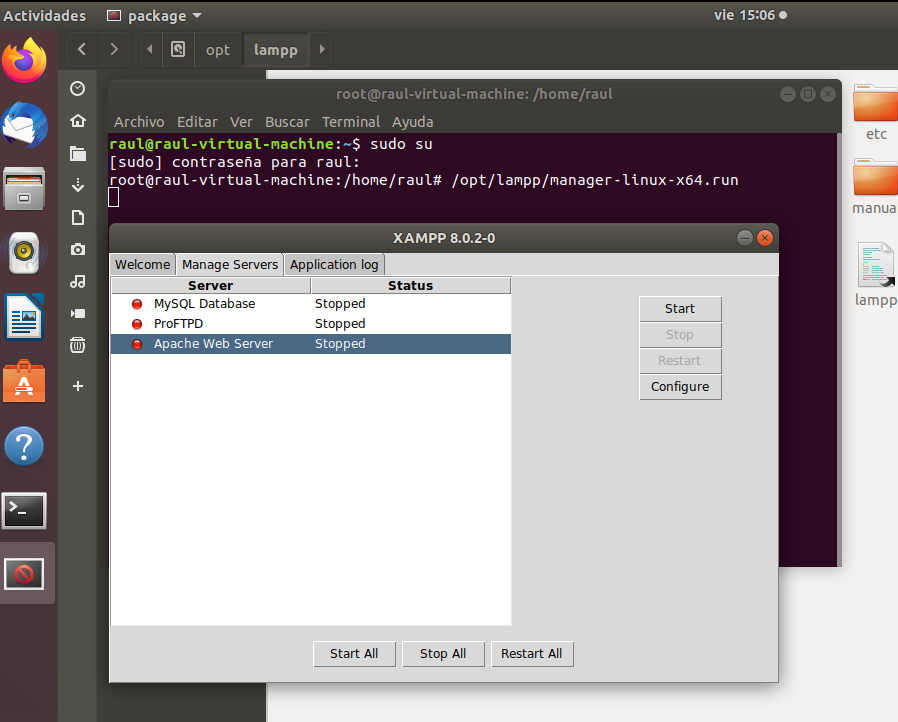
Se puede observar como aparece desplegada la aplicación, y pinchamos sobre ella para acceder:

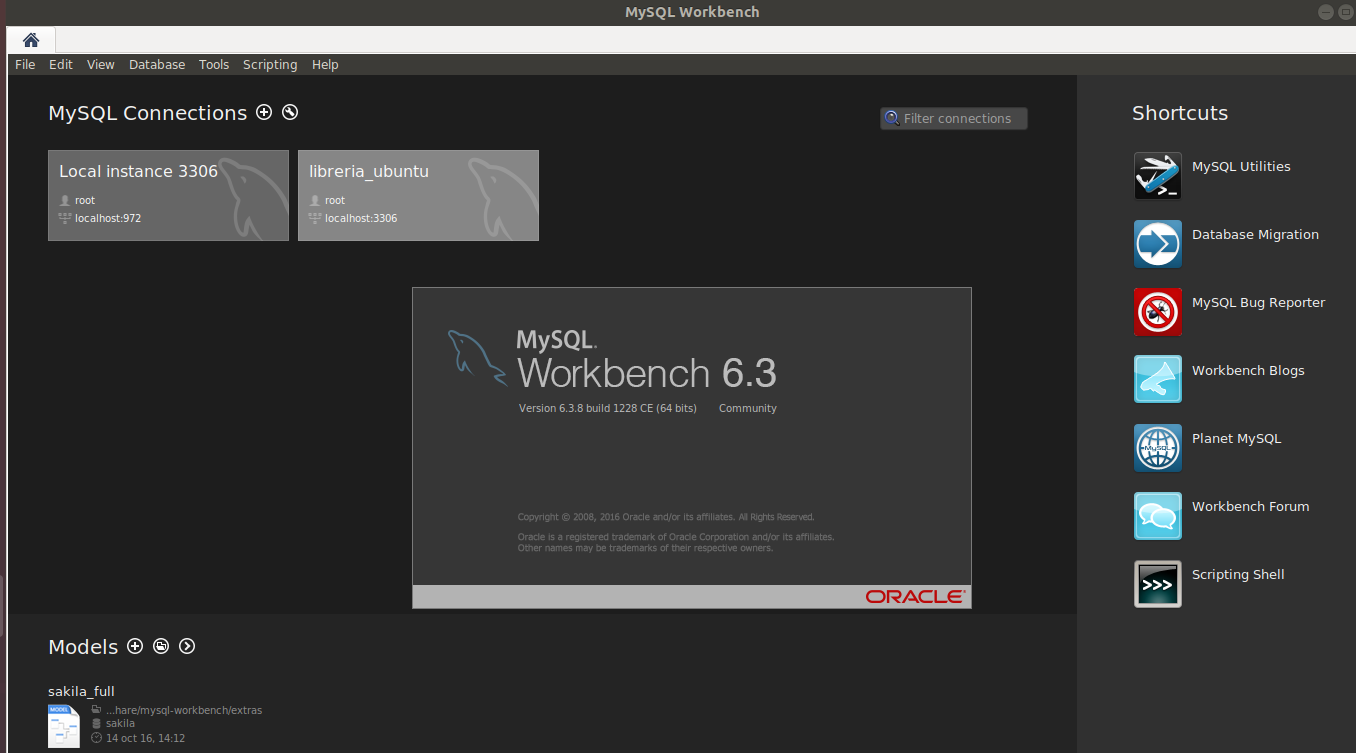


Para poder empezar a utilizar nuestra aplicación y tal y como se pide en la actividad, pasaremos a realizar unas configuraciones previas:

## 1.-CONFIGURACIÓN TOMCAT PARA CONEXIÓN A BBDD

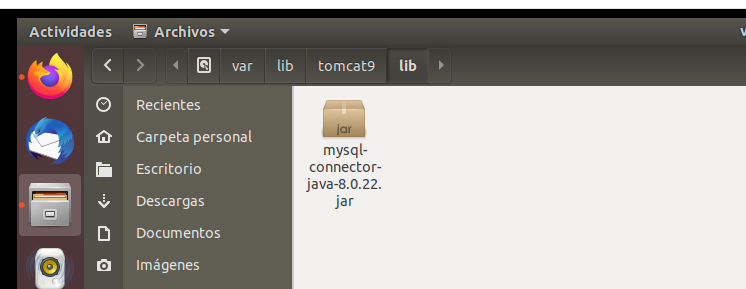
Instalamos **workbench** como gestor de BBDD y **LAMPP** para activar desde él el servidor de MYSQL. Con el comando**/opt/lampp/manager-linux-x64.run** arrancamos la consola de LAMPP:





Posteriormente **descargamos el driver jdbc** de la bbdd de la dirección propuesta en la actividad desde maven: https://mvnrepository.com/artifact/mysql/mysql-connector-java

Una vez descargado lo añadimos a la carpeta lib de nuestro servidor apache:



A continuación **popularemo**s una base de datos llamada proyecto y una tabla personas tal y como se indica en el enunciado:

**CREATE TABLE PERSONAS (**

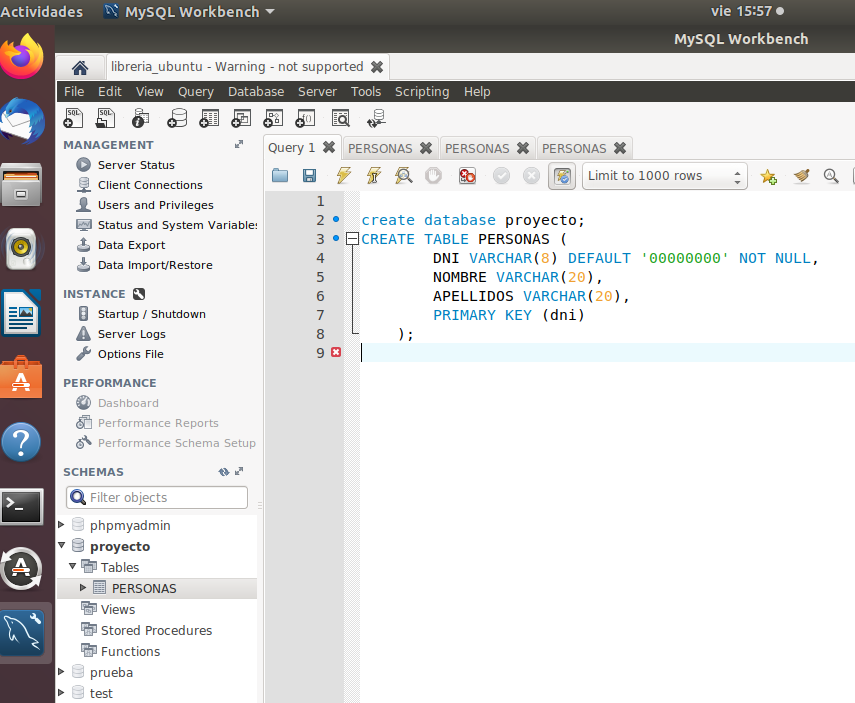
**DNI VARCHAR(8) DEFAULT '00000000' NOT NULL,**

**NOMBRE VARCHAR(20),**

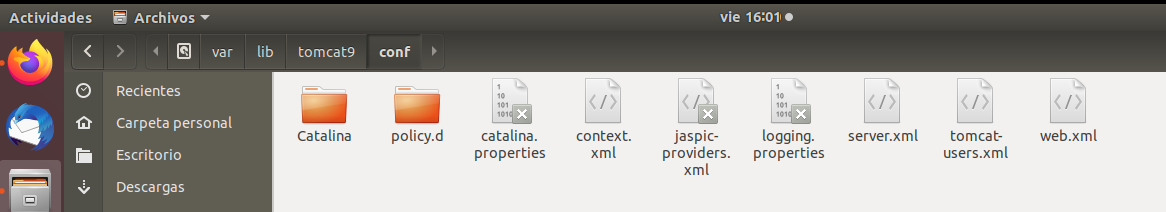
**APELLIDOS VARCHAR(20),**

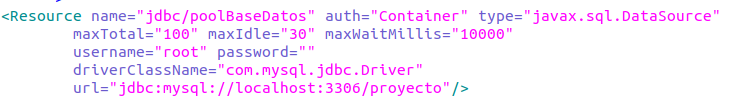
**PRIMARY KEY (dni)**

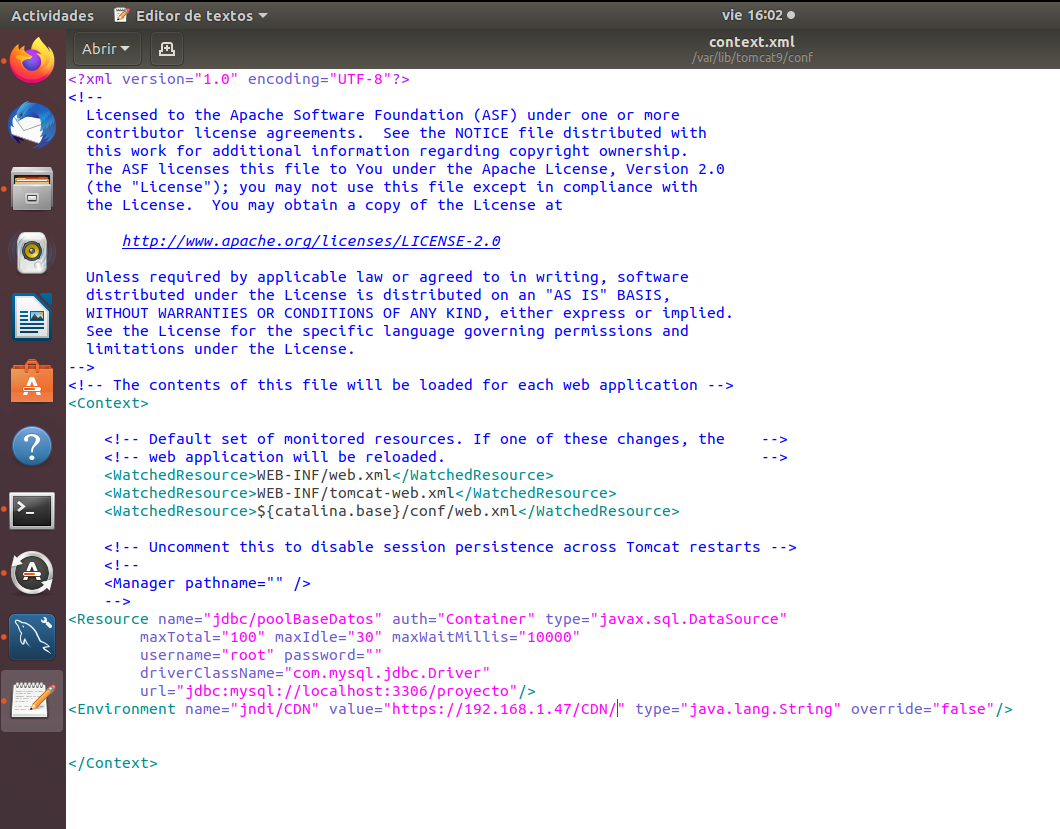
**);**



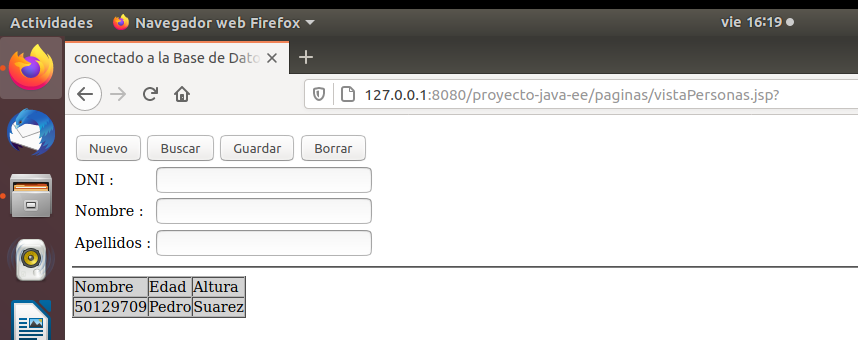
Por último crearemos un recurso ***"jdbc/poolBaseDatos"*** que contendrá los parámetros de conexión. Este recurso lo añadimos al fichero context.xml de nuestro servidor Tomcat:





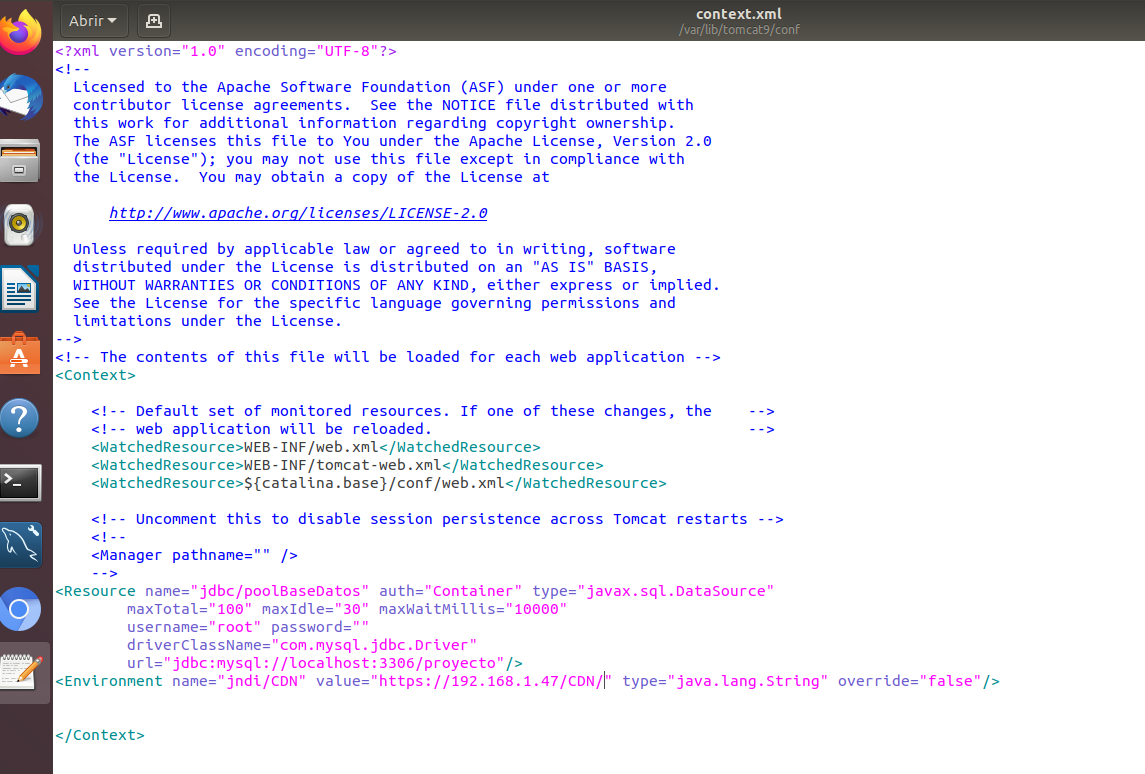


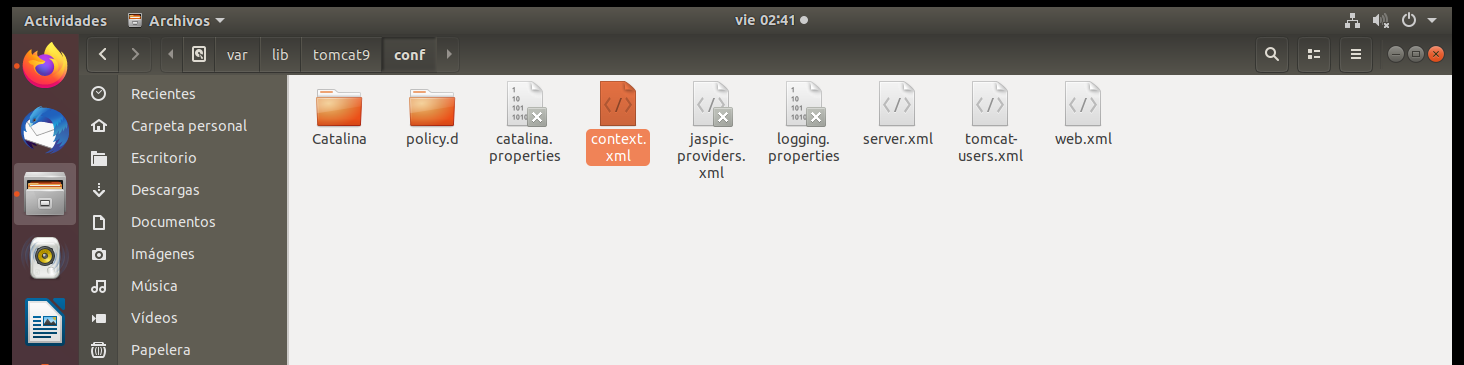
Probamos como nuestra app interactua con la bbdd:



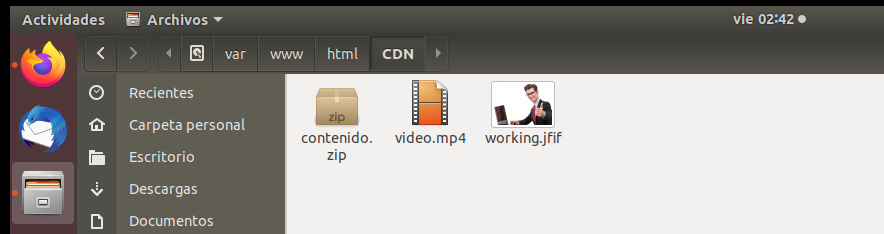
## 2.-CONFIGURACIÓN APACHE CDN PARA OBTENER RECURSOS DESDE TOMCAT

Creamos recurso tipo variable de entorno en Tomcat jndi/CDN que apunta a la url del servidor Apache donde se encuentran los recursos que queremos traernos:

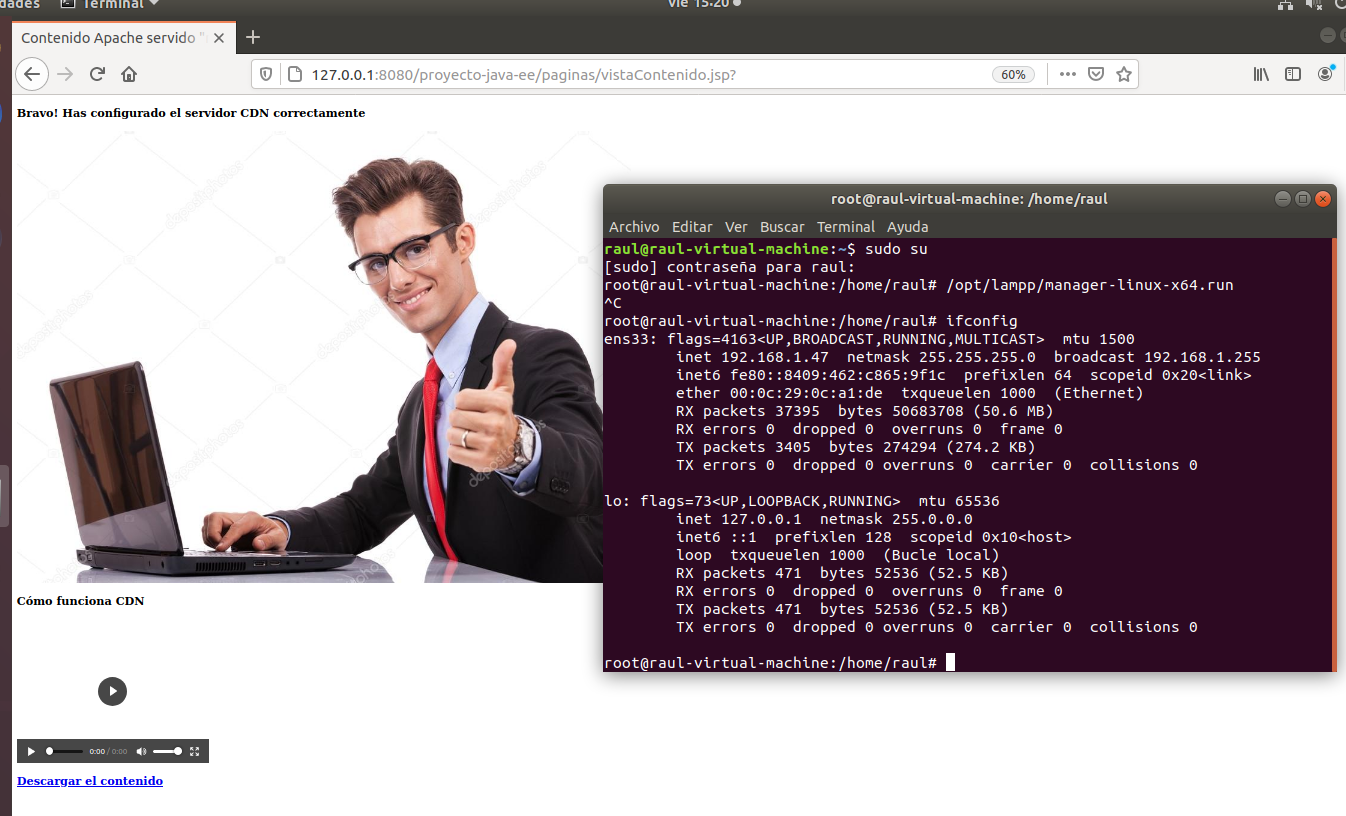




Desplegamos en directorio de Apache los 3 archivos estáticos a compartir



Probamos si está todo ok desde la app

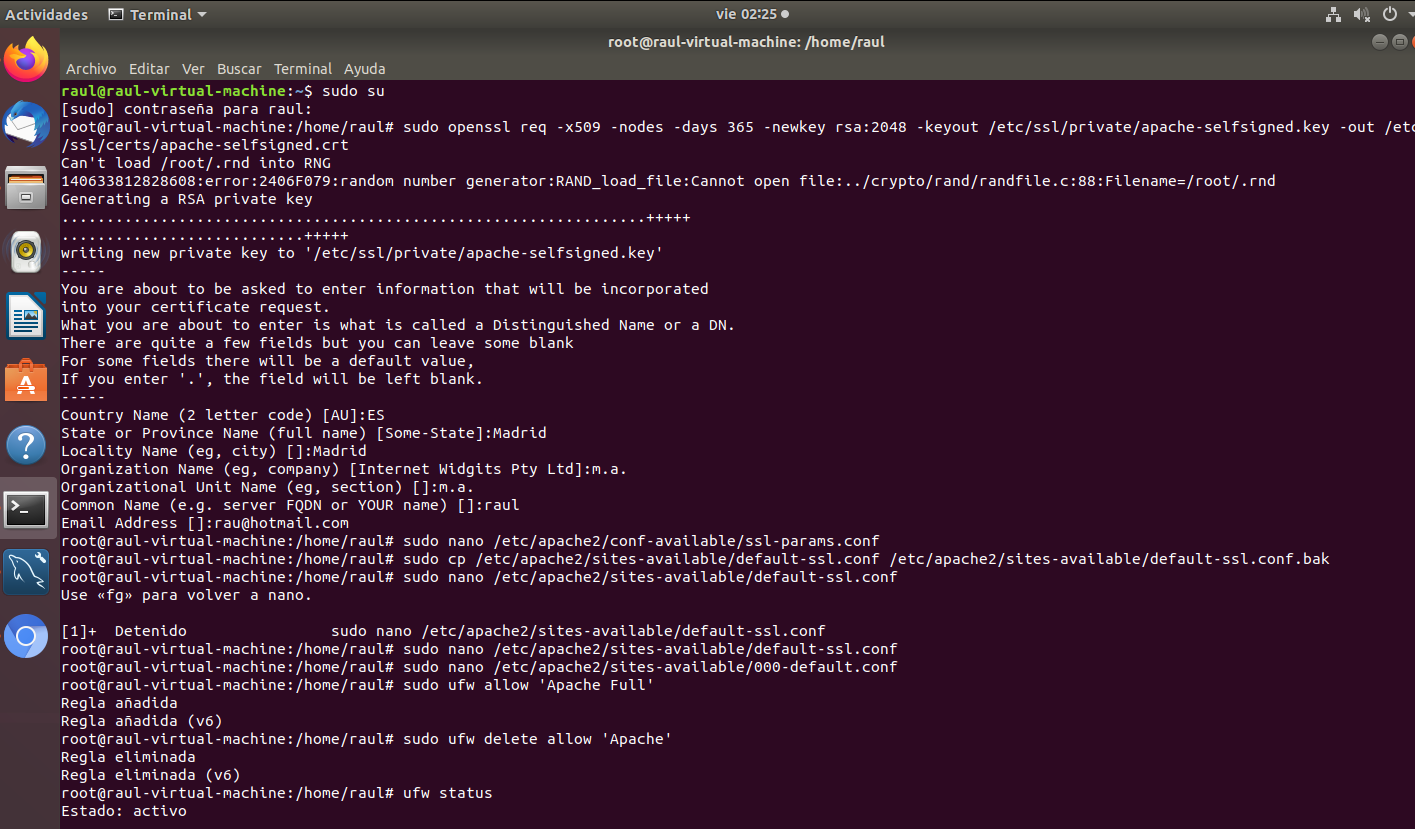


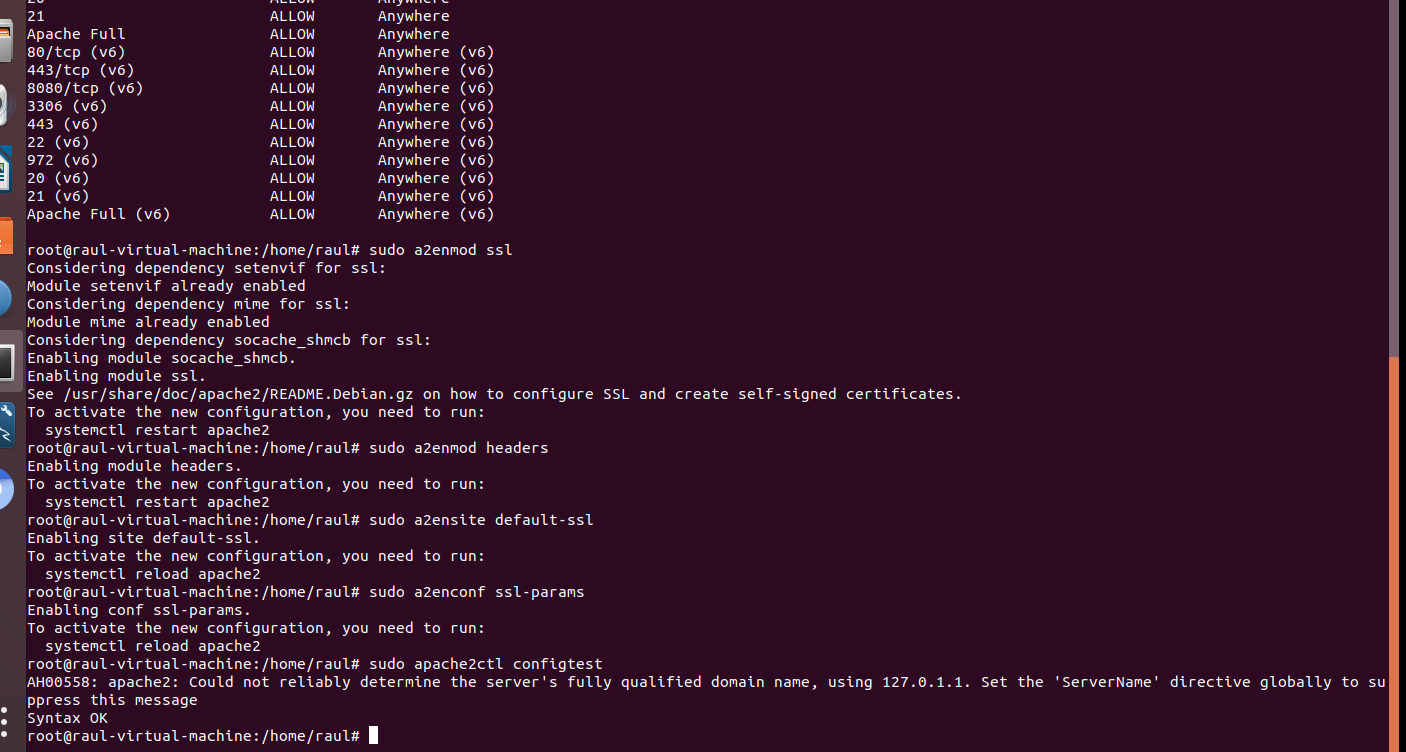
## 3.-CONFIGURACIÓN SERVIDOR FTP

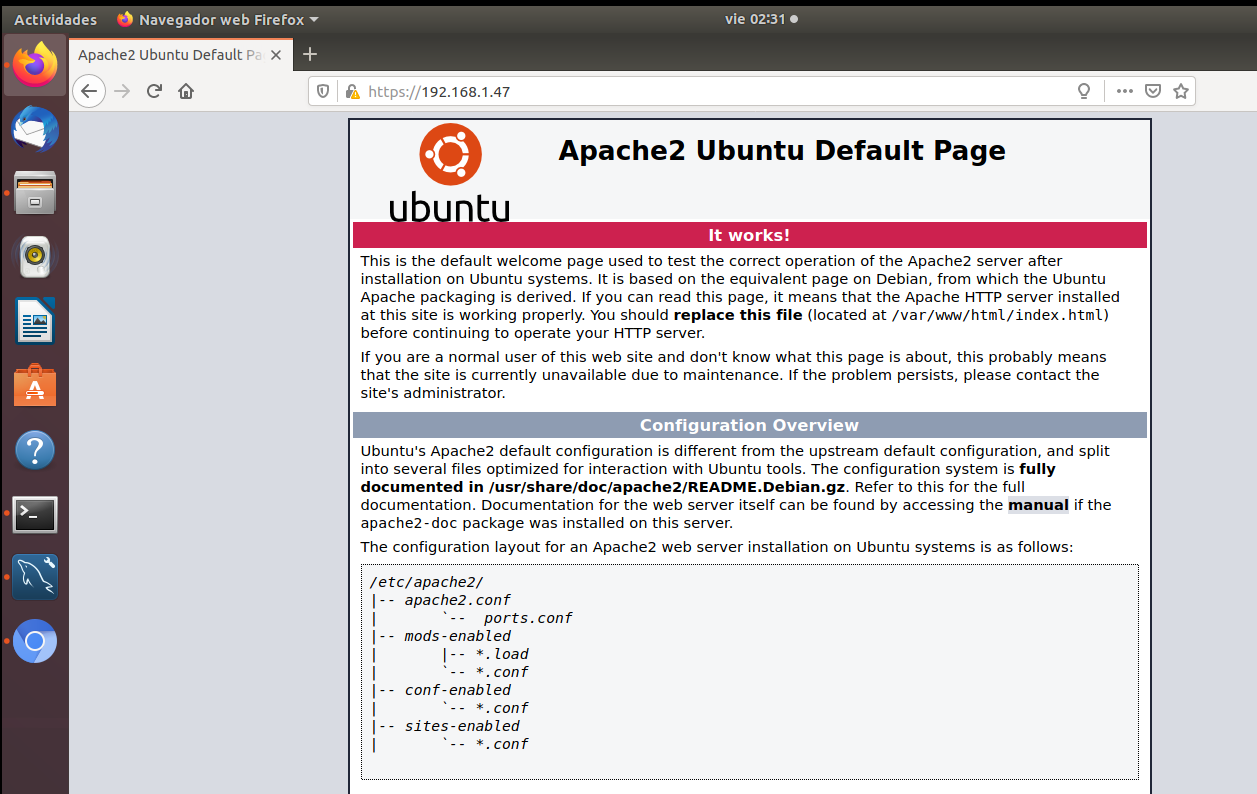
## 4.- HTTPS/CERTIFICADO DIGITAL APACHE

CERTIFICADO DIGITAL

https://www.digitalocean.com/community/tutorials/how-to-create-a-self-signed-ssl-certificate-for-apache-in-ubuntu-18-04-es







# 4.- JAVIER

# 5.- PATRICIA

# 6.- JAIME