PRÁCTICO

Proyecto

Título: Pizarra de reformas

Autor: Francisco José Sanz Belmonte

Tutor/a: Elena Ruiz Larrocha

Curso académico: 2023 – 2024

AGRADECIMIENTOS

En primer lugar quería agradecer de manera muy especial el incesante ánimo, respaldo y esfuerzo realizado por mi mujer y mis hijos para que pudiera dar la dedicación necesaria en la realización del módulo.

En segundo lugar, y no menos importante, quiero trasladar mi agradecimiento a todo el equipo humano que nos ha acompañado durante estos años, nos ha guiado, dado soporte y ánimo cuando lo hemos necesitado, tanto a compañeros, como profesores y tutores.

Índice de contenido

Índice de figuras

Índice de tablas

1. Resumen/Summary

Se trata de la puesta en marcha de una intranet la cual facilitará parte de la operativa diaria de una pequeña empresa de reformas, permitiendo también una mejor comunicación con los trabajadores, como puede ser la organización de fechas, lugares y asignación de trabajadores en las reformas, información de última hora que pueden dejar los empleados, la publicación de comunicados internos y otros documentos relevantes.

La principal idea es proveer a la empresa de una herramienta de gestión de equipo funcional de cara a los trabajos que tienen previstos realizar a corto o medio plazo, y paralelamente facilitar servicios de correo electrónico y servidor de ficheros que son indispensables hoy en día con costes accesibles.

Esta intranet está basada principalmente en la creación de contenedores ligeros como es Docker, donde se asentará una infraestructura basada en LAMP que nos permitirá alojar nuestra página web y controlar nuestra base de datos por medio de Mysql.

Para los servidores de correo y de ficheros utilizamos una máquina virtual, que a su vez funcionará como servidor de dominio. Esta combinación de tecnologías nos dará más versatilidad para adaptarse mejor según el aspecto que nos interese.

Esta solución no pretende competir con soluciones de intranet comerciales, capaces de llevar las redes sociales al entorno laboral, y servir de medio de colaboración o de comunicación muy avanzados como es el caso de *Workplace* o *Interact,* que incluso incluye herramientas de ofimática, colabora con *Zoom* y *Dropbox*, pero donde el desembolso llega a ser muy importante, llegando a cobrar cuotas por persona y mes.

Como adelantábamos, ofreceremos estos servicios (a excepción del registro del dominio) de manera muy económica, pudiendo montar nuestros servidores en máquinas modestas pero con la posibilidad de tener "escalabilidad vertical" a medida que vayan necesitando ampliar la capacidad de sus recursos.

This is about the launch of an intranet which will ease part of the operations of a small refurbishment company in daily basis, also allowing a better communication among workers, such as scheduling dates, location details and assignment of workers in every renovation, last minute information that workers can leave, publishing internal communications and relevant documents.

The main idea is to provide the company with a useful team management tool for the work they plan to carry out, and additionally offer email and file server services that are essential nowadays at affordable costs.

This intranet is mainly based on the creation of lightweight containers such as Docker, where an infrastructure based on LAMP will be installed, which will allow us to host our website and control our database through MySQL system.

We will use a virtual machine for the mail and file servers, that will work as a domain server at the same time. This technology combination will give us more versatility to better adapt to the function we are interested in.

This application doesn't intend to compete with commercial intranet solutions that are able to bring social networks to the work environment, and be used as an advance means of communication as Workplace or Interact do, that even include office tools, collaborate with Zoom and Dropbox, although the disbursement becomes very important, setting a monthly fee per person.

As we have mentioned previously, we will offer these services (but the domain registration) in a very affordable way, being able to set up our servers in modest machines but with the possibility of having vertical scalability as they need to extend the capacity of their resources

2. Antecedentes/Introducción

Normalmente dentro del sector de la construcción, existen muchas pequeñas empresas y autónomos que no cuentan con presupuesto para tener alguna solución informática que les pueda facilitar ciertos aspectos del día a día, limitándose a tener tan solo una página web estática con información muy básica,

El objetivo de este proyecto es el de ofrecer a cualquier pequeño empresario del sector, una herramienta de organización y planificación que pueda ayudar tanto al responsable en la gestión como facilitar información a los empleados, siempre intentando minimizar el coste de la implantación mediante software de código abierto principalmente.

La intención de la intranet tipo "pizarra" es la de facilitar una buena comunicación entre responsables de equipo/s y los trabajadores. Para ello la empresa utilizará este canal para reflejar las asignaciones de su personal a las obras más inmediatas, es decir, será una herramienta complementaria para la gestión de su equipo.

Por un lado tendremos un apartado específico para el administrador, a quién permitirá un acceso con permisos para dar de alta las notificaciones, detalles del trabajo, fechas y asignaciones. Además permitirá completar aquellos que ya se terminaron para pasar a un registro y conservar un histórico de trabajos realizados (con posibilidad de sacar reportes).

Por el otro, tendremos la intranet a la que el resto de empleados tendrán acceso para consultar toda esa información que detallábamos anteriormente.

Además, en esta pizarra publicará distintos documentos como el convenio de los trabajadores de este sector en concreto, actualizaciones, convocatorias a reuniones, reglamento en seguridad laboral y novedades, comunicados internos, etc. lo que lo convertirá en un punto de encuentro y ayuda para todos.

3. Objetivos generales y específicos y alcance del proyecto

3.1 Objetivos generales

Se pretende crear una intranet ágil, visual y sencilla dónde los trabajadores pueden recurrir para acceder a información de última hora y como punto de encuentro. Contará con un apartado específico para el administrador que funcionará como planificador y organizador, aunque también podrá ser utilizado como herramienta para reportes.

Así mismo, queremos cubrir ciertas necesidades básicas digitales como proveer de correo profesional con el nombre de la misma empresa, y establecer una serie de recursos compartidos en una red local que facilitarán también la comunicación entre distintos ordenadores.

Esto conllevará montar servidores en contenedores ligeros, los cuales nos permitirán portabilidad y ahorro de recursos de nuestra máquina, y la implantación de un servidor para compartir ficheros, servidor de DNS y de correo electrónico.

3.2 Objetivos específicos

En esta web los trabajadores podrán ver la dirección de la obra, datos del contacto, fechas de inicio de obra y fecha final de obra aproximada. Además encontrarán información adicional para esa obra como podría ser herramientas y material específicos necesarios, notas, recordatorios a tener en cuenta, etc. Igualmente podrán encontrar distintos documentos de interés (convenio, reglamento en prevención laboral, turnos, etc.) por lo que necesitaremos de los siguientes objetivos:

Crear e instalar máquina virtual Ubuntu 22.04 en VirtualBox,

Instalación de Docker

Despliegue de servidores con **LAMP** en Docker mediante **docker-compose**

Instalación y configuración de servidor de archivos

Implantación de servidor DNS y de correo electrónico

Planificación, creación y gestión de base de datos

Diseño web para intranet trabajadores mediante HTML, CSS y PHP

Web específica para administrador.

3.3 Alcance del proyecto

Como ya adelantábamos, esta web estaría orientada al sector de la construcción, pero más en concreto a pequeñas empresas o autónomos dedicados a las reformas o rehabilitaciones cubriendo una necesidad básica de información legal e interna, una vía rápida de comunicación, y como resultado de todo esto se convertiría en una intranet de la que pocas empresas de este tipo suelen tener.

No pretende ser un programa de gestión de facturación, bases de datos o **ERP,** de los que ya existen bastantes soluciones muy completas y especializadas.

No obstante, esta intranet podría ser fácilmente adaptada a otras empresas y sectores como podría ser el sector servicios (ej: asignación de camareros a distintos catering, bodas o restaurantes, informando de los sitios, horarios y fechas, junto con observaciones a tener en cuenta en cada lugar como podría ser uniformes especiales para cada caso, persona responsable en cada servicio, etc).

A su vez contaríamos con una estructura fácilmente portable a distintos equipos o incluso a un **Paas,** que sería a lo que más se están decantando las distintas compañías en estos momentos.

Este proyecto no tendría sentido si perdemos de vista el tema del coste dado el perfil de empresa al que nos estamos orientando. Gracias al software que utilizamos permitirá que estemos hablando de un presupuesto contenido, incluso en lo que a requerimientos del hardware se refiere.

4. Definiciones

Ca-certificates: es un paquete *deb* que contiene certificados proporcionados por las Autoridades de Certificación (para verificar la identidad de terceros).

CIFS: Common Internet File System o sistema de archivos de internet común, usado para conectar varias plataformas permitiendo compartir archivos y servicios de impresión.

Curl: herramienta de línea de comandos cuyo fin es transferir datos a través de una URL soportando distintos protocolos y certificados.

Docker: es un software de código abierto que se utiliza para desplegar y automatizar aplicaciones dentro de contenedores portables y ligeros en cualquier sistema operativo.

Docker Hub: se trata de un servicio proporcionado por Docker de registro de repositorios que nos permite descargar y publicar imágenes de contenedores.

ERP: Enterprise Resource Planning, es un software de gestión de distintas áreas de la empresa, que facilita el flujo de datos entres ellas. Ayuda en la planificación, aprovisionamiento, predicción y notificación de resultados.

Gnupg: herramienta que sirve para cifrar datos (protegiendo la privacidad del usuario) y realizar firmas digitales.

LAMP: conjunto de aplicaciones de software de código abierto que facilita la infraestructura de un servidor de aplicaciones y páginas web dinámicas (acrónimo de Linux, Apache, MySQL, PHP).

MDA: mail delivery agent, es un agente de reparto el cual almacena el correo entregado por el MTA, mientras espera a que el usuario lo acepte. Ejemplo: Dovecot, Fetchmail, Getmail.

MUA: mail user agent (cliente de correo electrónico, ej: Thunderbird), programa cliente que permite la recuperación de correos almacenados en el **MDA.**

MTA: mail transfer agent, Servidor de correo remitente a destinatarios (entre servidores mediante protocolo SMTP), ejemplo Postfix, Sendmail, Qmail, etc.

Paas (Platform as a service): Se trata de tipo de plataforma en la nube que proporciona de manera muy flexible una infraestructura completa de hardware y

software para nuestras aplicaciones, sin las limitaciones, costes y complejidad que conllevaría el hacerlo en local.

Samba: implementación libre del protocolo SMB cliente-servidor que gestiona el acceso a archivos y directorios. Se encarga de intercambiar información con posibilidad de interconectar redes Microsoft con Linux o mac OS.

systemd: es un gestor y administrador central del sistema que se ha convertido en el nuevo estándar para las distintas distribuciones Debian y otros sistemas Linux, que interactuando con el núcleo del SO, permite iniciar servicios y demonios bajo demanda. Se desarrolló para reemplazar el sistema de inicio init (primer proceso en ejecución tras la carga del núcleo).

Servidor open relay: se trata de un servidor de correo (MTA) que permite el reenvío de correo sin verificar de dónde viene y quién lo envía, pudiendo emplearse para envío de SPAM).

Tee: es un comando Linux que lee la entrada estándar, muestra la salida, y la guarda en uno o más archivos al mismo tiempo, es decir, la copia.

.YML: es un formato estandarizado de archivo de texto (legible por humanos) orientado al intercambio de datos (misma función que los archivos XML).

5. Notaciones y símbolos

Para facilitar la comprensión, describimos algunas notaciones, abreviaciones y símbolos como las detalladas a continuación:

Emulación de terminal en Ubuntu:

```
sudo mkfs -t  ntfs /dev/sdb1
```

Edición de archivos de configuración del sistema:

```
#PARTICIÓN DE 25 GB COMPARTIDO POR SAMBA
UUID=630C503035A02972 /home/adminubuntu/DATOS ntfs    defaults    0       0
```

Emulación en edición de archivos Docker y docker-compose:

```
www:
```

```
image: php:8.1.1-apache
```

lineas o partes de lineas comentadas dentro de estos ficheros

```
#command: --default-authentication-plugin=mysql_native_password
```

Formato de archivos HTML, CSS y PHP

```
<!DOCTYPE html>
```

```
<html>
<head>
<title></title>
```

SMB: Server Message Block, protocolo para control de acceso a archivos, directorios y otros recursos de red con estructura cliente-servidor

LTS: Long term support . En el contexto de las versiones del sistema operativo Ubuntu, significa soporte a largo plazo. En este caso son 5 años de soporte de actualizaciones y seguridad en el momento de la redacción de este proyecto.

IDE: Integrated Development Environment, entorno de desarrollo integrado

ICANN: Internet Corporation for Assigned Names and Numbers, Corporación de Internet para la asignación de nombres y números.

LSSICE: Ley de Servicios de la sociedad de la información y de Comercio Electrónico.

LOPD: Ley Orgánica de Protección de Datos.

6. Desarrollo del trabajo final

En esta sección se va a explicar la organización general del proyecto y la linea de tiempo en que ha sido desarrollado.

Comenzamos por los preparativos base para la "virtualización" para poder ofrecer los distintos servicios mediante una máquina virtual (VirtualBox) y contenedores ligeros **Docker,** lo que permitirá una fácil y rápida portabilidad de nuestra infraestructura. Dentro de este mismo contenedor nos va a permitir desarrollar y ejecutar todos los servicios necesarios para servir nuestra página web, con lo que ello implica, aglutinado todo en nuestra solución LAMP, desde el servidor web - php hasta el gestor de bases de datos MySql.

En nuestra máquina virtual, continuaremos con la preparación de un servidor para compartir archivos utilizando **Samba** por su gran flexibilidad y facilidad de uso entre distintos sistemas operativos, donde habrá que gestionar los permisos necesarios para nuestros grupos y usuarios. Acto seguido se procede a implantar un servidor DNS el cual será la base para crear y configurar un servidor de correo electrónico. Completando esta fase, utilizaremos un cliente de correo en nuestros dispositivos, detallando la configuración para su uso.

Finalmente, en este tercer y último bloque está centrado en la aplicación web con todo lo que esto engloba, desde el diseño, la creación y gestión de la base de datos con la ayuda de la herramienta gráfica phpMyAdmin para MySQL, hasta la creación de código necesario en HTML, CSS y PHP. En esta fase de la creación del código fundamentalmente utilizo Microsoft Visual Studio, un entorno de desarrollo integrado multiplataforma que nos facilita la creación y edición de código.

En la figura 1 se muestra las herramientas utilizadas en el presente proyecto.

Figura 1. Herramientas utilizadas

(fuente: www.mauroalfieri.it)

6.1. Instalación y configuración de VirtualBox

Para empezar a trabajar vamos a preparar una máquina virtual con el citado software VirtualBox perteneciente y actualmente soportado por Oracle (como open source y gratuito), donde instalaremos la última versión **LTS** de Ubuntu. En el momento de la publicación de este proyecto se corresponde con la versión 22.04.

En nuestro caso la máquina contará con características que van a facilitar cierta solvencia, ya que el host nos lo permite (aún siendo un pc que tiene más de una década, ejemplo que no requiere equipos de última generación o gran potencia), pero podría ser igualmente funcional con la mitad de memoria RAM y procesadores. En cuanto al almacenamiento nos reservamos un disco específico para el sistema operativo con margen y capacidad más que suficiente en caso de actualizaciones, nuevas instalaciones, etc.

Sin embargo, emplearemos un disco virtual independiente que nos servirá únicamente como almacenamiento de datos para la empresa. El hacer esta diferenciación de discos nos ofrecerá cierta seguridad en caso de fallo del disco donde se aloja el sistema operativo (por culpa de entrada de virus, actualizaciones del S.O. que puedan producir fallos, etc.)

En la Tabla 6.1 indicamos los detalles de los recursos definidos en nuestra máquina

SS.OO	UBUNTU 22.04 (64-bit)
MEMORIA RAM BASE	8192 MB
PROCESADORES	4
ALMACENAMIENTO SATA 0	35 GB (SISTEMA OPERATIVO)
ALMACENAMIENTO SATA 1	25 GB (DISCO DATOS)
ADAPTADOR RED 1	wlp3s0 - MODO BRIDGE
ADAPTADOR RED 2	intnet1 - RED INTERNA

Tabla 1: Características máquina virtual.

(fuente: propia)

Al empezar con la instalación se han encontrado algunos inconvenientes que han sido solventados en cada momento.

En este caso hemos utilizado un equipo viejo donde hemos tenido que habilitar la opción de virtualización en la BIOS (del equipo físico), ya que no estaba configurado así por defecto. Saltaba mensajes de error avisando sobre el controlador del Kernel de Linux informando que no estaba cargado o configurado correctamente y solicitaba revisión de la citada BIOS.

Tras reiniciar, es muy importante que incluyamos nuestro usuario (mrx) en el grupo "vboxusers" en el equipo físico (host) para obtener los permisos necesarios para operar con este programa, ya que de otra manera nos encontraremos impedimentos para continuar.

```
sudo  usermod  -a -G vboxusers mrx
```

Por otro lado, se ha detectado que tras la instalación de Ubuntu, nuestro usuario no pertenece al grupo de administradores. El sistema deshabilita por defecto el usuario *root* por motivos de seguridad, y a su vez incluye el nuestro en el grupo de administradores *root* para realizar acciones administrativas puntuales que requieren ciertos permisos para después volver a tener los permisos de una cuenta normal. Sin embargo, no se produjo este cambio, y hubo que realizarlo de manera manual montando el sistema en lectura y escritura (en modo "live") para después incluir al usuario en el citado grupo de administradores.

Para ello se recurrió al "recovery mode" de las opciones avanzadas del Grub. Una vez allí nos vamos al intérprete de comandos raíz o consola de superusuario, y montamos el sistema de archivos en modo lectura/escritura (mount -o rw, remount /), en el que ya nos permite añadir nuestro usuario al grupo "sudo".

Realizado ese cambio y arrancada nuestra nueva MV Ubuntu22, empezamos a instalar los paquetes necesarios para compilar con el siguiente comando:

```
sudo apt-get install build-essential gcc make perl dkms
```

Procedemos a instalar los *guest-additions* de VirtualBox en el equipo virtual Ubuntu para un mejor y más completo funcionamiento.

Para ello, nos aseguramos de insertar la imagen como si de un CD se tratara. Tas esto, tan solo nos dirigimos al archivo VBox_GAs_7.0.8 en el lugar del punto de montaje, asegurándonos que tiene los permisos necesarios para ejecutarlo en modo gráfico, o si lo preferimos por la terminal en el archivo *VBoxLinuxAdditions.run* que encontraremos dentro, pero tendremos que darle permiso de ejecución si no lo tiene ya asignado.

Podemos poner la máquina en modo de pantalla completa para comprobar su funcionamiento (es conveniente reiniciar el equipo previamente).

Remarcar que su instalación es optativa, aunque se recomienda ya que mejora el rendimiento optimizando la usabilidad de nuestras nuevas máquinas virtuales.

Entre las características más importantes que nos facilita está la integración del cursor del ratón, carpetas compartidas entre el host y nuestra máquina virtual, mejor soporte de vídeo, sincronización horaria, portapapeles compartido y la total integración de la ventana en nuestra máquina.

Dentro de las configuraciones que queremos definir, estarían los interfaces de red. Por un lado creamos un adaptador de red en modo Bridge 192.168.1.150 que tendrá conexión con el exterior dentro de la misma red que nuestro host, y de manera adicional contaremos con otro adaptador, pero en esta ocasión para nuestra red interna (172.18.0.100/16).

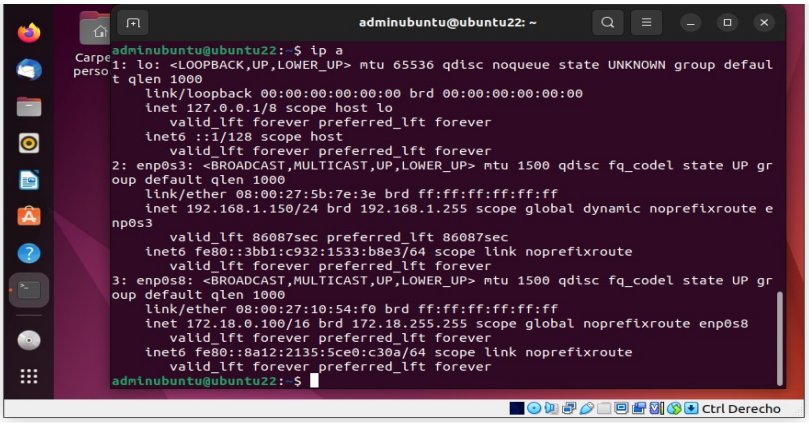

Figura 2. Detalle de interfaces de red máquina virtual

(fuente: propia)

Siguiendo con los preparativos, creamos un segundo disco en una partición distinta destinado a salvaguardar los datos y archivos importantes que considere la empresa, asignándole una capacidad de 25 GB. De esta manera podrán permanecer a salvo en caso de fallos en el sistema operativo, dejando a este en una partición distinta con suficiente espacio para tener nuestros programas y las actualizaciones que vayamos descargando e instalando con un total de 35 GB.

En lo que a características se refiere, será un disco duro conectable en caliente por puerto SATA 1, almacenamiento reservado dinámico.

Para **crear la partición** nos hemos decantado por la utilización de *fdisk* y para formatearla *mkfs,* en ambos casos desde la terminal, pero la mejor alternativa para realizar estas operaciones intuitivamente es la herramienta de gestión de discos Gparted (en modo gráfico y mucho más rápido).

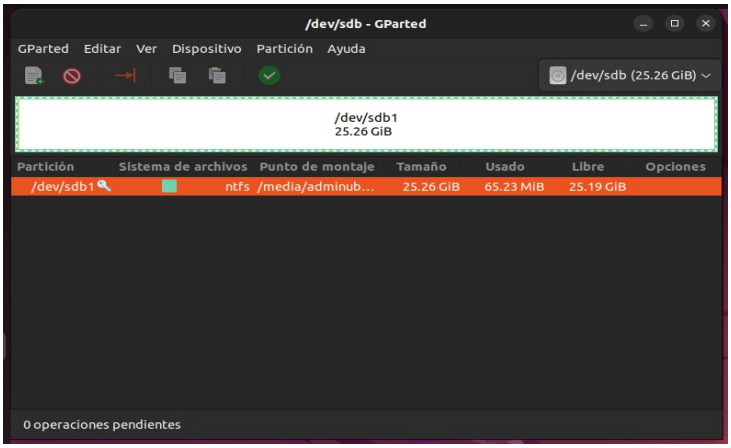

Figura 3. Gparted (fuente: propia)

Para que nuestra partición recientemente creada se pueda montar al iniciar el sistema y nos aparezca como una carpeta más, debemos de configurar nuestro fichero */etc/fstab* (Tabla de sistemas de ficheros). En él encontraremos varias líneas de configuración que provocarán que los cambios que realicemos sean persistentes entre reinicios.

Editamos y añadimos la siguiente línea al mencionado archivo, donde tenemos que informar del disco a montar, el punto de montaje, sistema de archivos a utilizar y resto de opciones por defecto (respaldo y chequeo del sistema).

```
#PARTICIÓN DE 25 GB PARA  COMPARTIR POR SAMBA

UUID=630C503035A02972 /home/DATOS ntfs     defaults              0       0
```

Cuando reiniciamos, se monta como una carpeta más del sistema en el directorio que le indicamos. Además ya aparece en la barra de herramientas del sistema. Nos evitará tener que realizar los montajes manualmente en cada sesión.

Con esto terminamos de preparar nuestra máquina virtual con VirtualBox en el que alojaremos nuestros servidores DNS, de correo y servidor de ficheros.

Ahora damos paso a construir nuestra base de contenedores Docker de manera paralela y alternativa ligera al software explicado en este capítulo, para dar alojamiento al resto de servidores y servicios para la web.

6.2. Instalación de Docker y docker-compose

El siguiente capítulo está dedicado a la instalación y preparación de Docker, una herramienta alternativa o complementaria a la definida creada en el capitulo anterior.

Docker es una herramienta para ejecutar programas o servicios dentro de contenedores. Éstos empaquetan todo el código y dependencias que requieren nuestras aplicaciones pero reunidas en un único archivo, el cuál se ejecutará de la misma manera en cualquier máquina.

En resumen, en concepto, Docker es similar a una máquina virtual, pero mucho más ligera. Puede ejecutar varios contenedores a la vez en una misma máquina sin necesidad de cargar completas y pesadas máquinas virtuales, ya que solo cargará lo indispensable del SO de interés (el kernel y la parte específica de software adicional que se vaya a necesitar).

Antes de empezar con el trabajo en sí, debemos tener en cuenta algunos puntos importantes. Comenzamos por asegurarnos de no tener versiones anteriores, ya que esto podría acabar dando errores, por lo que para evitarlos tendríamos que desinstalar todo. Previo a este paso, para que quede completamente limpio de imágenes, redes, contenedores y volúmenes estos deben ser eliminados en los directorios *docker* y *containerd* en */var/lib* si los hubiese.

Ahora ya empezamos la instalación del mencionado software de virtualización de contenedores ligeros, donde desplegaremos posteriormente nuestros servidores para nuestra intranet "Pizarraweb".

Para ello comprobamos si está cargado el módulo KVM de virtualización mediante el siguiente comando:

```
kvm-ok
```

Creamos el repositorio Apt de Docker para después poder instalar y actualizar las herramientas necesarias. Con el motivo de poder hacer uso de estos repositorios por medio de https, tendremos que instalar estos paquetes que usaremos después: **ca-certificates**, **curl** y **gnupg**.

Hemos de añadir y configurar la clave GPG para el repositorio Docker quedando localizado en el directorio */etc/apt/keyrings/*

Nos ayudaremos con el comando curl, el cual se encarga de descargar la clave GPG de Docker haciendo que dirija la salida al comando gpg . Éste, con la opción adecuada se encarga de desempaquetar el archivo previamente blindado, enviando a su vez el resultado al archivo */etc/apt/keyrings/docker.gpg*

Una vez terminado ese paso, para evitar problemas de acceso, damos permisos de lectura al archivo a todos los usuarios.

Finalmente añadimos el repositorio Docker para nuestro sistema y actualizamos (apt-get update). Llegados a este punto, ahora podemos proceder con la instalación de Docker Engine, el cliente, Docker Compose y otros paquetes necesarios,

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

Para terminar con la instalación se recomienda realizar una comprobación de la versión y ver si están habilitados. Será recomendable crear el grupo docker (si no existiera ya), y después añadir los usuarios que queramos a este grupo.

```
sudo systemctl status containerd
```

```
sudo systemctl status docker
```

A partir de estos momentos ya tenemos preparado nuestro sistema para crear nuestro entorno LAMP basado en contenedores.

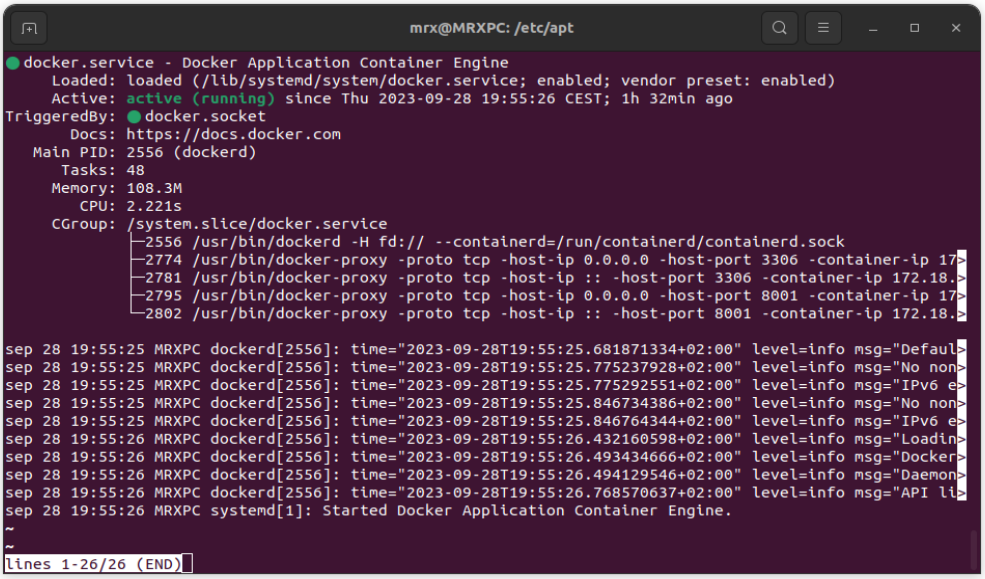

Figura 4. Servicio Docker activo (fuente: propia)

6.3. Creación y configuración entorno LAMP con Docker-Compose

En primer lugar es conveniente presentar a Docker Compose, una herramienta complementaria de Docker que nos ayudará a definir, configurar y ejecutar aplicaciones Docker que requieren la ejecución de varios contenedores.

Para este cometido utiliza un archivo de configuración con extensión YAML, donde dejará detallados los distintos servicios (contenedores) necesitados por nuestra aplicación, realizando una conexión entre ellos por medio de una red Docker que también podremos definir.

Una vez instaladas las herramientas como hemos visto en el punto anterior, vamos a preparar nuestro entorno mediante la configuración del archivo docker-compose.yml, el cual nos facilitará todo el trabajo, ya que incluiríamos en él las distintas órdenes y comandos para la creación de nuestros servidores Apache, MySQL y PHP que necesitamos.

Para ello nos hemos servido de unos de los mejores **IDE** del momento como es Microsoft Visual Studio, y que además cuenta con un plugin específico de Docker (muy sencillo de instalar desde extensiones), que nos permitirá editar código, visualizar los contenedores que se están ejecutando, iniciar y parar, ver el estado de cada uno. De manear adicional nos dará acceso también una consola interactiva, subida de datos a repositorio GitHub, etc.

Resaltar que nos permite ver en la misma pantalla los contenedores parados y en ejecución, todas las imágenes que hemos obtenido de Docker junto con las que hemos creado, las entradas de registros que hemos subido a Docker-Hub con las que podemos interactuar previa identificación. También podemos encontrar en un mismo bloque las redes disponibles, creadas y en uso. En la sección final localizamos todos los volúmenes que hemos creado para nuestro trabajo.

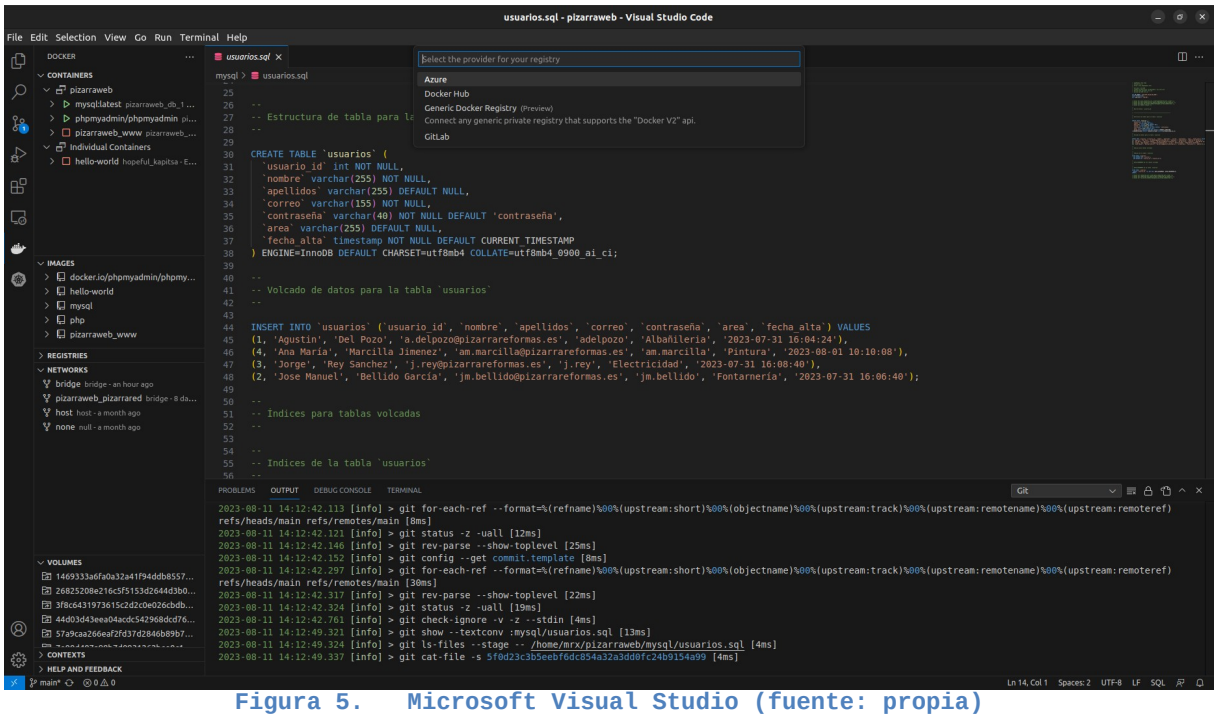

Figura 5. Microsoft Visual Studio (fuente: propia)

Preparamos la estructura de las carpetas necesarias con las que vamos a trabajar.

En nuestra carpeta personal hemos creado la que contendrá el proyecto, "/home/usuario/pizarraweb", donde a su vez colgarán otras tres carpetas:

Dockerfile (contendrá el archivo del mismo nombre), mysql (contendrá copias de seguridad de la base de datos) y www (con los archivos html, css, php y otros que usaremos para nuestra página web).

En esta misma carpeta creamos el archivo docker-compose.yml que definirá nuestros servicios. En él hemos diferenciado tres grandes bloques.

El primero donde informamos de la versión utilizada y el servicio db (mysql), un segundo bloque con la definición del servidor web (www) y el tercer bloque con la configuración para el servidor de PhpMyAdmin como podemos ver en esta imagen.

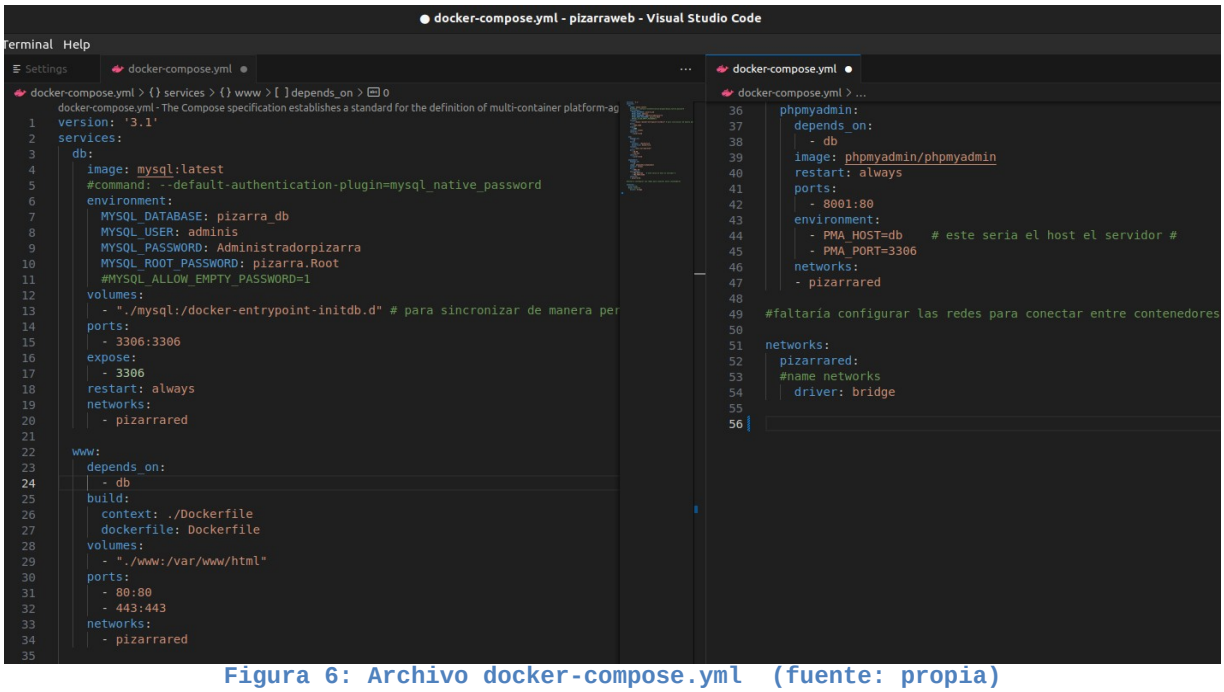

Figura 6: Archivo docker-compose.yml (fuente: propia)

Repasamos brevemente el archivo de configuración partiendo de la versión utilizada y los servicios donde se encuentran esos bloques.

Comienza por "db" que hace referencia al servidor y sistema de gestión de base de datos, donde indicamos la imagen y versión que va a utilizar, junto con las variables de entorno que utilizará, como son el nombre de la base de datos, el usuario, la clave de ese usuario y del usuario root.

Además necesitamos incluir un volumen donde grabar de manera persistente los cambios que vayamos realizando. De esta manera reutiliza los contenedores que ya existen con sus respectivas configuraciones y ajustes como vamos a necesitar en nuestro caso.

Luego informamos de los puertos que expondremos de nuestra máquina y el puerto vinculado al contenedor. Con la opción *expose* le decimos el puerto que estará únicamente disponible para ese servicio vinculado, no para la máquina host.

Para terminar le indicamos por un lado la opción de reiniciar en caso de fallo , y la red interna que utilizarán los contenedores para que se puedan ver entre si.

En el segundo servicio "www" definimos el servidor web con algunas diferencias.

Le hacemos saber que este servicio depende de el anterior "db" para que pueda funcionar de manera correcta. Es importante remarcar que debemos respetar el orden de aparición de estos servicios, ya que, sobre todo en este caso tenemos esa dependencia del anterior y de no ser así acabaría en error.

Tras esta sección, indicamos como se va a realizar la instalación de la imagen recurriendo al archivo Dockerfile que apuntará a la imagen de Php adaptada a Apache disponible en Docker Hub, la cual requerirá al mismo tiempo la instalación de la extensión de msqli (para permitir la comunicación con el sistema gestor de la base de datos). Retomando el archivo *docker-compose.yml,* también definiremos un volumen para no perder la información que vayamos actualizando, los puertos que se expondrán y la red común que utilizaremos (pizarrared) entre los distintos contenedores.

Siguiendo estos servicios definidos, llegamos al bloque Phpmyadmin, donde de nuevo indicamos que dependemos del servicio anterior "db", plasmando la imagen que vamos a utilizar junto con opciones similares a los otros servicios, pero donde cabe destacar las variables de entorno. Una de ellas sería el host del servidor y el puerto que utilizará.

Existe una última sección definida fuera de los servicios detallados. La red que vamos a utilizar para conectar los citados contenedores, informando de su nombre y el interface o driver que hemos utilizado (brigde).

Una vez grabado los cambios del archivo docker-compose.yml, creamos el archivo Dockerfile al que hacíamos referencia anteriormente, en nuestro caso ubicado en /home/usuario/pizarraweb/Dockerfile, pero con respecto a nuestro archivo docker-compose estaría dentro de la carpeta Dockerfile en el mismo directorio, es decir, quedaría en *./Dockerfile.*

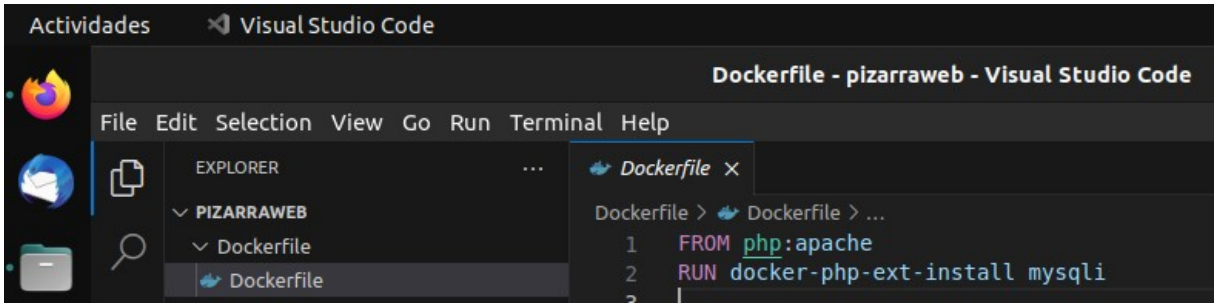

Figura 7. Dockerfile (fuente: propia)

En la imagen anterior vemos como creará la imagen a partir de la versión "php:apache," y además ya incluiremos las librerías de "msqli" necesarias para las conexiones entre Php y Mysql que adelantábamos previamente.

Ahora nos dirigimos a la consola y ejecutamos:

docker-compose down // # docker-compose up

Con el primero indicamos que detenga todos los servicios asociados, pero además elimina los contenedores y las redes internas asociadas con los servicios (aunque no

los volúmenes), a diferencia de la orden "stop" que tan solo los detiene sin eliminar nada.

Después ya podemos levantarlo para implementar los servicios de aplicaciones creando nuevos contenedores a partir de las imágenes indicadas junto con volúmenes, redes, etc.

Realizamos las siguientes pruebas que detallamos a continuación:

En la carpeta de nuestro host www creamos un archivo index.php para realizar una primera comprobación en nuestro navegador en la dirección "http://127.0.0.1:80/". En la segunda comprobación nos conectamos al servidor (contenedor) mysql dejándonos en un *bash* donde podremos ejecutar los siguientes comandos, por un lado la orden de docker, y por otro la orden para acceder a MySQL introduciendo la clave indicada anteriormente.

`docker exec -it 3b34be /bin/bash` // `mysql -u root -p`

Como vemos en la siguiente imagen nos permite entrar con éxito dejándonos en el prompt de MySQL para la ejecución de sus comandos.

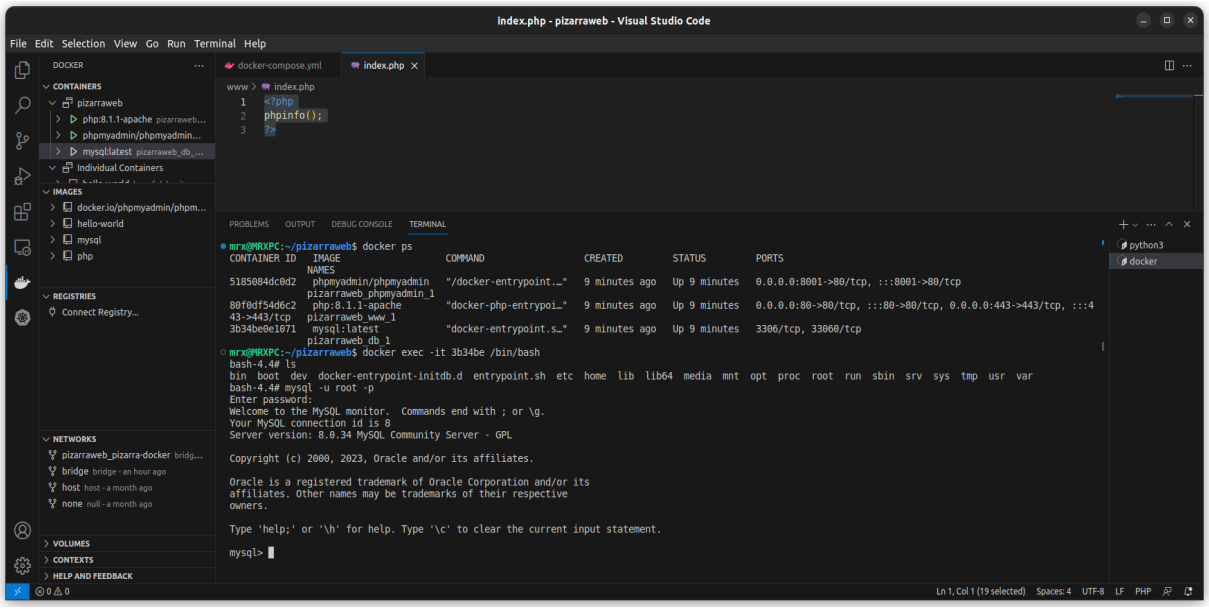

Figura 8. Acceso a consola MySQL (fuente: propia)

También podemos entrar en phpmyadmin por "localhost:8001" con el usuario definido "adminis", con el que creamos la base de datos pizarra_db y una de sus tablas, en particular la de usuarios, que reutilizaremos posteriormente. Por ahora lo utilizamos para realizar pruebas de conexión entre contenedores.

Por el momento editamos un nuevo archivo *index.php* para realizar las mencionadas pruebas entre servicios, donde declaramos una variable *connect* llamando a la

función *mysqli_connect* y sus variables, un query a la BBDD usuarios por medio de un SELECT, y terminando con una llamada a función para realizar un bucle para tomar los datos con mysqli_fetch_array dando el siguiente resultado:

Figura 9. Conexión Php y MySQL (fuente: propia)

De acuerdo a la configuración anterior, como se puede ver en la figura anterior, acabó con éxito la conexión entre el servidor PHP y la base de datos.

Sin embargo, la realidad al principio fue distinta, ya que todos los intentos acababan con un error de conexión o de función mysqli indefinida. Esto llevó a probar la conexión mediante el objeto PDO (PHP Data Object), un método de conexión muy versátil que permite las conexiones a MySQL y a otros sistemas gestores de bases de datos como puede ser PostgreSQL, pero lamentablemente acabó del mismo modo, con el citado error.

Como solución alternativa intentamos realizar de nuevo la conexión mediante las funciones *"mysqli"*, pero persistía el *"Fatal error"* como se puede ver en la imagen que vemos abajo:

Figura 10: Error de conexión de PHP con la base de datos (fuente: propia)

En un principio, en el archivo *docker-compose.yml,* establecimos la creación de la versión de la imagen de php:8.1.1

También se probó a poner en marcha el contenedor, entrar en modo interactivo con un entorno *bash* instalando las extensiones directamente para seguir haciendo pruebas, pero de igual manera persistía el problema.

Después de investigar en distintas webs, documentación oficial, foros, etc. concluían que esta imagen daba bastantes errores por temas de compatibilidad en diferentes versiones de Docker.

Finalmente la solución pasó por cambiar la versión de la imagen, en concreto se utilizó *"php:apache" la cual fue construida* dentro del archivo Dockerfile, junto con la instalación de la extensión *docker-php-ext-install mysqli* en este mismo archivo, que permitía la comunicación entre nuestros contenedores web y de base de datos. Este a su vez quedaba declarado dentro de la sección del servicio "*www"* del archivo docker-compose.yml

Con esto queda en pleno funcionamiento la pila de tecnologías que conforman la plataforma LAMP del lado del servidor, que nos permitirá crear sitios web y aplicaciones entre otras soluciones.

6.4. Instalación y configuración de servidor DNS

Ahora entramos en detalle para ver los servicios y configuración necesarios para poner en funcionamiento el servidor de correo para el que utilizaremos una máquina virtual Ubuntu (configurada al inicio de este proyecto), a diferencia de los contenedores creados en el capítulo anterior para la plataforma LAMP.

En resumidas cuentas, vamos a trabajar en el servidor de correo electrónico de la empresa. Para ello será necesario previamente tener preparado un servidor DNS en el que crearemos el dominio de la empresa (pizarraweb.com), ya que las cuentas de correo no pueden funcionar con direcciones IP.

A fin de conseguir este propósito, en primer lugar estableceremos una IP fija a nuestro equipo, para después comenzar con la instalación y configuración de nuestro servidor *Bind.*

Con este paso terminado, podremos proceder a la instalación del servidor de correo electrónico Postfix (MTA), con miras a hacer lo propio con Dovecot (MDA), nuestro servidor IMAP y POP3 de código abierto para Linux. Con el objeto de complementar estas aplicaciones, finalizaremos con nuestro cliente gráfico Thunderbird (MUA).

6.4.1 Configuración de IP fija de nuestro servidor e instalación del servidor Bind.

En primer lugar necesitamos configurar la IP fija de nuestro servidor y al mismo tiempo vamos a indicar el nuevo servidor de nombres.

Tenemos dos maneras de realizar los cambios, mediante el entorno gráfico de escritorio que en nuestra máquina virtual se trata de Gnome, o editando los ficheros de configuración mediante la terminal de comandos. Este último modo es el que vamos a explicar, aunque es algo más complejo y tendremos que ser más cuidadosos, pero nos ayudará a entender mejor algunas características del sistema operativo.

Hasta la versión 18.04 de Ubuntu, el archivo de configuración principal de red era */etc/network/interfaces.* En él podíamos indicar la IP fija, máscara de red, la red en la que nos encontramos, etc. Sin embargo, ya en esta versión de Ubuntu, la configuración se realiza por medio del archivo *01-network-manager-all.yaml* ubicado en */etc/netplan/.*

Como podemos observar en la siguiente imagen, éste contiene las distintas opciones de configuración (conservando el patrón de datos legible que caracteriza a este tipo de archivos).

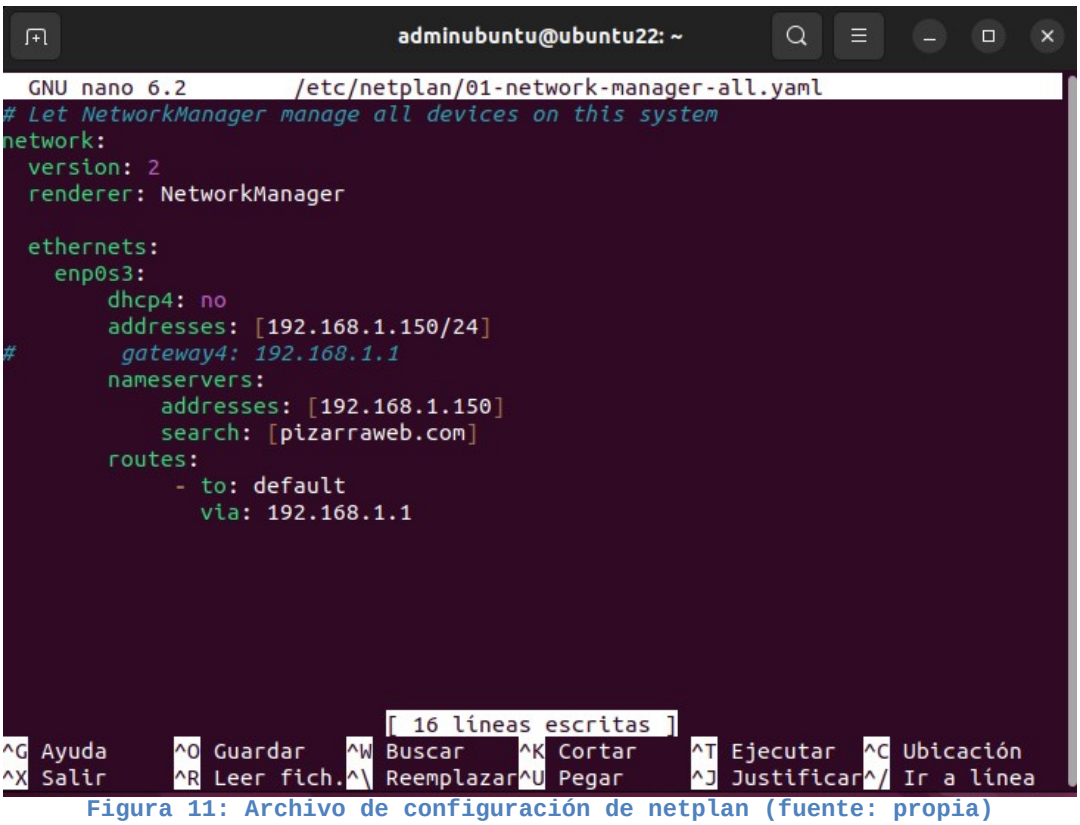

Figura 11: Archivo de configuración de netplan (fuente: propia)

Para empezar, tendremos que desactivar la opción dhcp4 dentro de nuestra interface de red, e informamos sobre nuestra nueva dirección IP fija. Es importante señalar que el servidor de nombres que vamos a utilizar es la dirección de la propia máquina, es decir, 192.168.1.150, donde a continuación debemos incluir nuestro dominio pizarraweb.com en el nombre de servidores, pues lo añadirá cuando hagamos consultas con nombres relativos.

En un principio nos encontramos problemas al configurar estas opciones, ya que al crear las lineas estaba escribiendo objetos con un formato que ya no era válido:

```
ethernets:
  enp0s3:
     dhcp4: no
     addresses: [192.168.1.150/24]
     gateway4: 192.168.1.1
     nameservers:
        addresses: [192.168.1.150,8.8.8.8]
        dns-search: pizarraweb.com
```

Finalmente las opciones fueron corregidas permaneciendo el archivo como muestra la figura 11.

Además, al consultar el archivo */etc/resolv.conf* nos indica que ya no podemos realizar cambios en él de manera persistente como resultado de la incorporación de **systemd** a estos sistemas Linux, **(**tienen su propia implementación de resolución de nombres), por lo que realizamos el enlace simbólico para que tenga efecto:

```
sudo ln -sf /run/systemd/resolve/resolv.conf /etc/resolv.conf
```

Después aplicamos cambios con "sudo netplan try" y "netplan apply"

Ahora vamos a instalar el servidor DNS Bind en Ubuntu con la siguiente orden (tras actualizar los repositorios):

```
sudo apt install bind
```

Necesitaremos editar el archivo */etc/nsswitch.conf* para obligar a que busque primero en el propio dns y luego en host y no al revés. Para ello nos vamos a la siguiente línea y hacemos los siguientes cambios:

```
hosts:          files mdns4_minimal [NOTFOUND=return] dns
```

por esto

```
hosts:          dns files mdns4_minimal [NOTFOUND=return]
```

Acto seguido reiniciamos interfaces de red y dns, y comprobamos

```
sudo systemctl restart systemd-networkd
```

```
sudo systemctl restart systemd-resolved
```

```
sudo resolvectl status
```

6.4.2 Configuración del servidor DNS Bind

Tras la instalación de Bind, tendremos que modificar y crear (en algunos casos) los archivos de configuración para que el servicio pueda funcionar correctamente.

Comenzamos con la modificación del archivo *etc/bind/named.conf.local* configurando el uso de forwarders para realizar consultas externas, pasando a ser nuestro servidor DNS autoritativo para nuestra red:

```
//Busqueda directa
zone "pizarraweb.com" in {
  type master;
  file "/etc/bind/zones/db.pizarraweb.com"                    #dirección y archivo;
};
/Busqueda inversa
zone "1.168.192.in-addr.arpa" in {
  type master;
  file "/etc/bind/zones/db.1.168.192";
};
```

Para una mejor organización creamos el directorio *zones* dentro de /etc/bind donde ubicaremos nuestros archivos de configuración de zona.

Creamos el fichero de nuestra zona a partir de db.local para facilitar el trabajo

```
cp db.local db.pizarraweb.com
```

A continuación editamos db.pizarraweb.com y realizamos comprobación.

```
; BIND data file for pizarraweb.com;
$TTL    604800
@       IN      SOA     pizarraweb.com. root.pizarraweb.com  (
                              2       ; Serial
                         604800             ; Refresh
                          86400             ; Retry
                        2419200             ; Expire
                         604800 )           ; Negative Cache TTL
;
;name servers - NS registros
@                             IN                    NS      servidor.pizarraweb.com.
;name servers - registros tipo A
servidor                      IN                     A       192.168.1.150
pizarraweb.com.               IN                     A       192.168.1.150

;name servers - registros tipo CNAME (alias)
server                        IN                     CNAME   servidor
@                             IN                    A       127.0.0.1
@                             IN                    AAAA    ::1
;tipos:
;
```

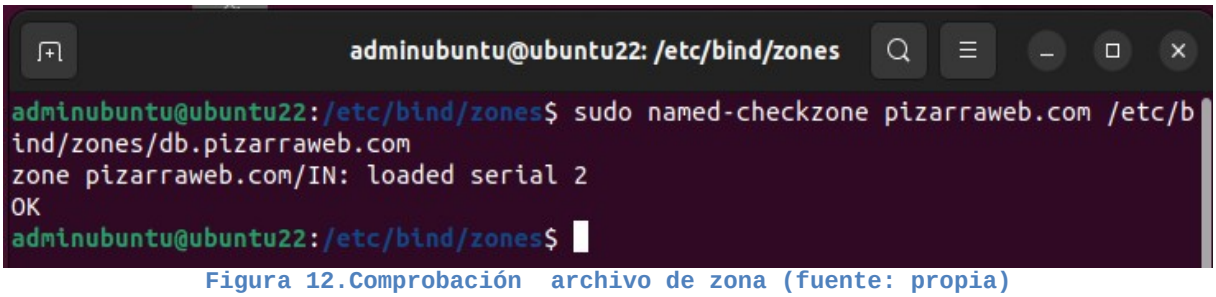

Figura 12.Comprobación archivo de zona (fuente: propia)

Ahora creamos el archivo de zona inversa también indicado en la configuración anterior en el directorio /etc/bind/zones.

Copiamos el archivo *db.127* en el que será nuestro archivo *db.1.168.192* para editarlo:

```
; BIND reverse data file for local loopback interface;
$TTL    604800
@       IN      SOA     localhost. root.localhost. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL;
; name servers
@       IN      NS      pizarraweb.
;PTR Registros
150     IN      PTR     pizarraweb.pizarraweb.com
;1.0.0   IN      PTR     localhost.
```

Al igual que en el anterior archivo realizamos comprobación con el mismo resultado positivo. Reiniciamos el servicio y añadimos una regla para que funcione a través de nuestro firewall si lo tenemos activo:

```
sudo named-checkzone 1.168.192.in.addr-arpa /etc/bind/zones/db.1.168.192
```

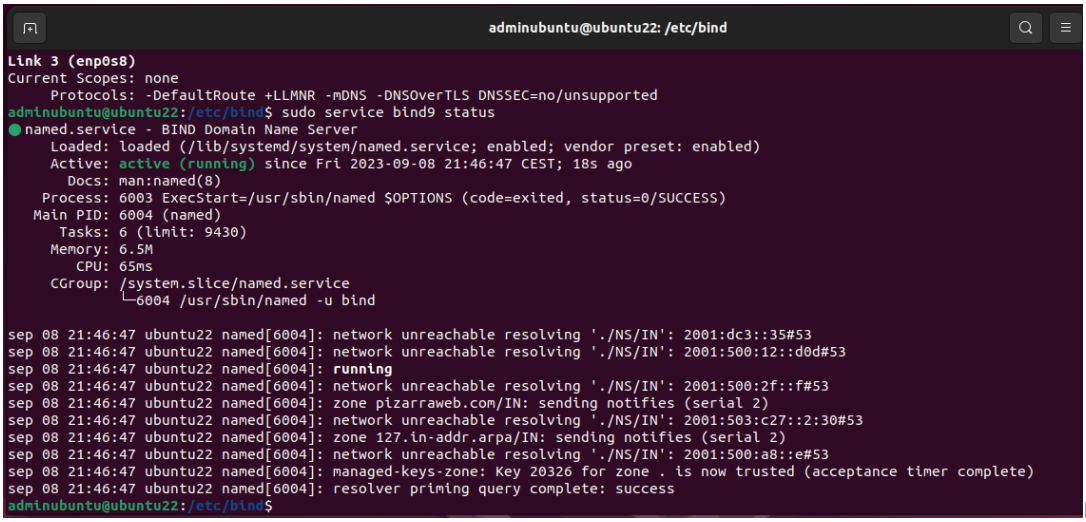

Figura 13.Estado del servidor DNS Bind (fuente: propia)

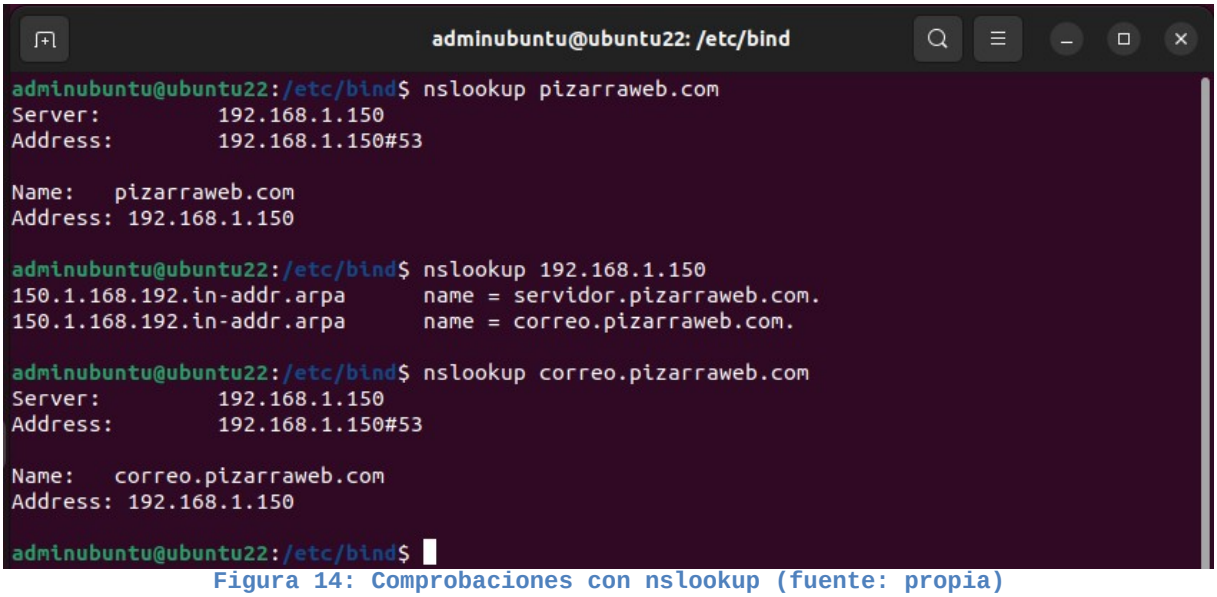

Figura 14: Comprobaciones con nslookup (fuente: propia)

En las dos anteriores imágenes podemos comprobar cómo el servicio Bind9 de nuestro servidor DNS está activo y funcionando de manera correcta.

En esta última figura que vemos más arriba realizamos las comprobaciones mediante el comando *nslookup* con la resolución de nuestro nombre de dominio y mediante la dirección IP para ver el resultado de la resolución inversa.

Una vez preparado y configurado el servidor DNS en nuestra máquina virtual, podemos empezar a preparar la instalación de nuestro servidor de correo Postfix, el cual será completado con Dovecot, que será nuestro agente de entrega de correo.

6.5. Instalación y configuración del servidor de correo electrónico.

Una vez tenemos preparado el servidor DNS, podemos comenzar a trabajar con el servidor de correo, donde tendremos en cuenta los siguientes puntos:

- ✓ El servidor DNS es servidor.pizarraweb.com
- ✓ Nuestro servidor de correo estará en la misma máquina virtual que el servidor DNS
- ✓ El nombre de dominio del servidor de correo será correo.pizarraweb.com

Previo a cualquier paso, es esencial entender el proceso que comprende desde que escribimos un correo hasta que el destinatario lo recibe. En la siguiente imagen se resumen los sujetos que participan y los protocolos empleados en cada envío.

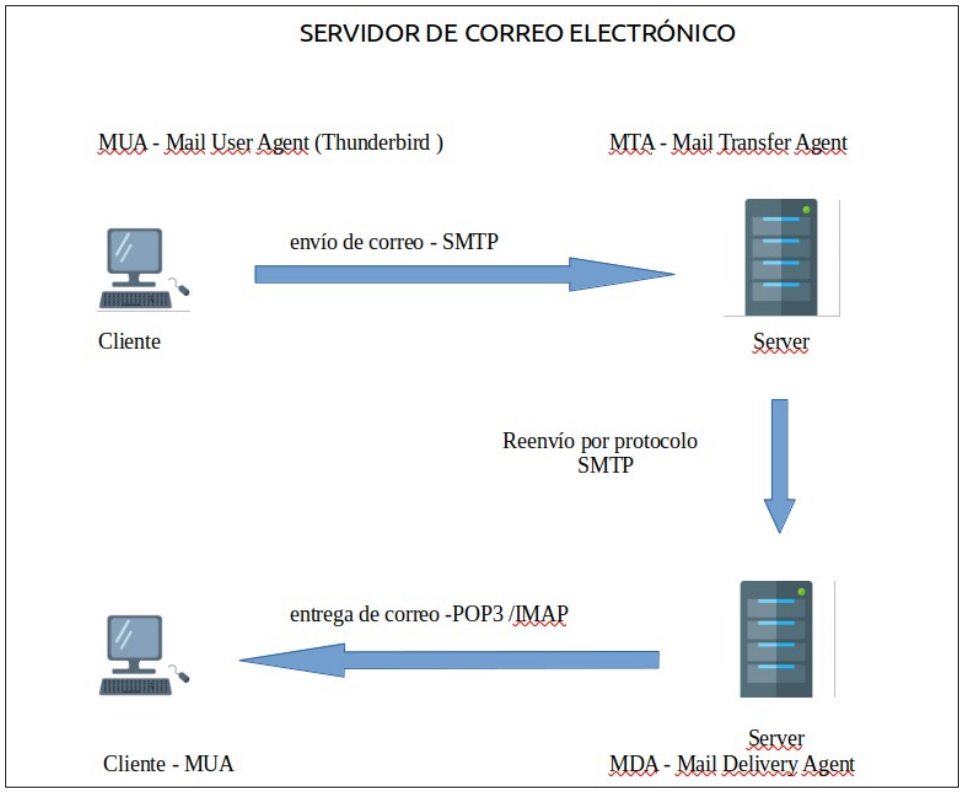

Figura 15: Proceso en el envío de correo electrónico (fuente propia)

Este conjunto de acciones y aplicaciones nos permite almacenar y reenviar los correos. El **MTA** se encarga del proceso de la transferencia por medio del protocolo SMTP entre equipos, mientras que el **MDA** se encarga de depositar esos mensajes en los buzones. Finalmente el **MUA** (cliente) se encarga de obtener esos mensajes almacenados en estos buzones y escribir correos.

Para indicar el nombre del servidor de correo, nos dirigimos a nuestro servidor DNS donde almacenamos los registros de recursos añadiendo el tipo MX, el cual indicará qué equipo actuará como servidor de correo de nuestro dominio. Para ello editaremos el archivo de zona de resolución directa *db.pizarraweb.com* localizado en el directorio */etc/bind/zones/*

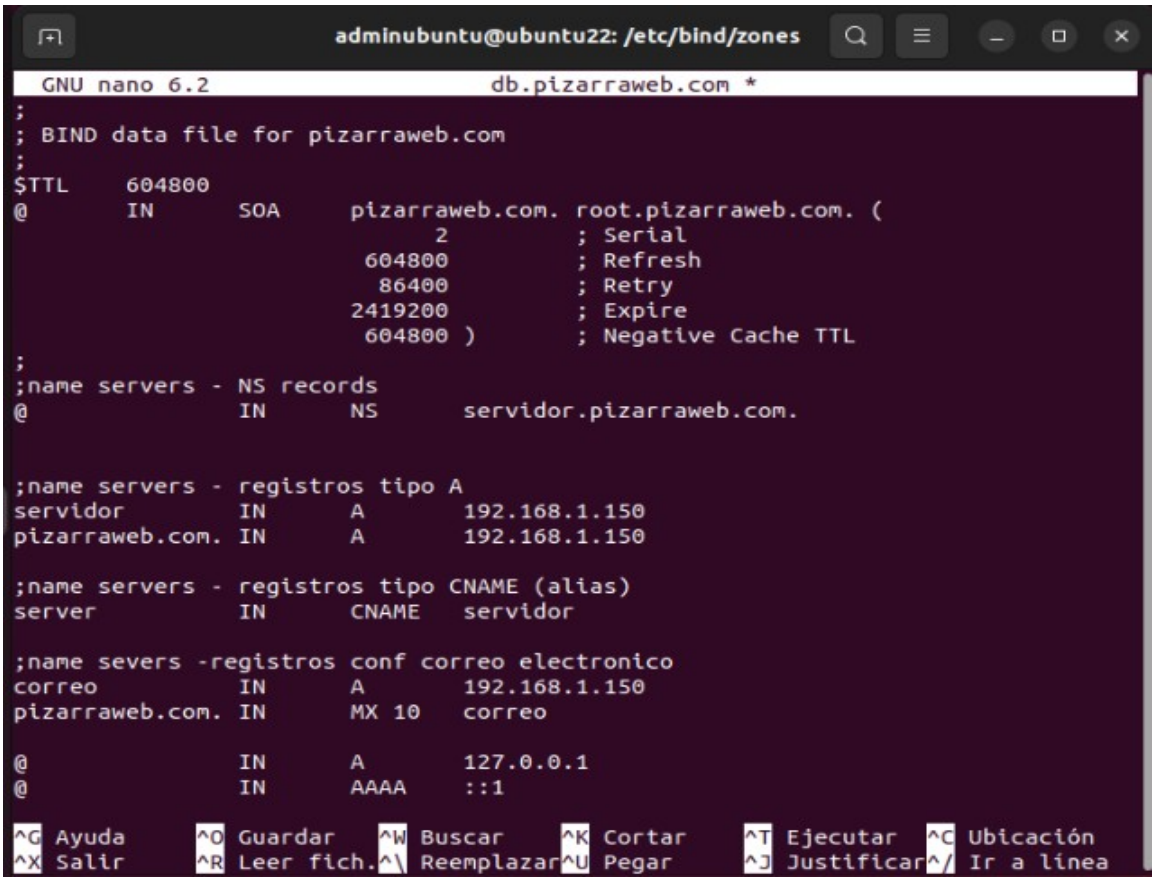

Figura 16: Archivo de zona de resolución directa db.pizarraweb.com (fuente: propia)

También es importante tener configurado el archivo de resolución zona inversa con el fin de que nuestros correos no sean descartados como SPAM por otros MTA's.

Adelantar que para poder crear los usuarios que tendrán correo electrónico en nuestro servidor tendríamos que crearlos previamente. En este caso vamos a empezar con dos usuarios que tendremos en nuestra base de datos antes de configurar nuestro MTA.

```
sudo adduser j.rey --force-badname
```

```
sudo adduser a.delpozo --force-badname
```

6.5.1 Configuración de Postfix

Para ponernos en situación, informar que Postfix es un servidor de correo libre y de código abierto. Es el agente de transporte por excelencia en distribuciones Linux y también incorporado en algunas versiones de Mac OS, el cual utilizaremos en nuestro proyecto.

Como hacemos normalmente, actualizamos repositorios e instalamos después.

```
sudo apt install postfix
```

Durante la instalación nos pedirá ir completando algunos campos necesarios como elegir *para nuestro ejemplo, sitio de Internet. A* continuación indicaremos el dominio "pizarraweb.com", realizamos comprobaciones de funcionamiento y también del estado de nuestros puertos (en concreto nos fijamos en el puerto 25 smtp).

```
postconf mail_version
```

```
sudo service postfix status
```

```
sudo netstat -lnpt
```

A partir de estos momentos ya tenemos funcionando el servidor, pero será necesario realizar algunos cambios en el fichero de configuración.

Una buena práctica a tener en cuenta siempre que modifiquemos este tipo de ficheros es realizar una copia de seguridad para volver al punto inicial en caso de ocasionar errores durante las tareas.

Editamos el fichero */etc/postfix/main.cf* para indicar el nuevo hostname, nuestras redes, los interfaces de red con la que trabajaremos, los dominios desde los que se aceptarán correos entrantes (mydestination).

```
myhostname = correo.pizarraweb.com
```

Por defecto coge el nombre del host y añade el dominio. Cambiamos y ponemos "correo." y seguido nuestro dominio.

Nombre del dominio para el correo saliente. Es decir, la parte de la dirección de correo que queda a la derecha de la @. Está configurado y se redirecciona al archivo "mailname".

```
myorigin = /etc/mailname
```

```
adminubuntu@ubuntu22:/etc/postfix$ cat /etc/mailname
pizarraweb.com
adminubuntu@ubuntu22:/etc/postfix$
```

Figura 17: Contenido de mailname (fuente: propia)

```
mydestination = $myhostname, pizarraweb.com, localhost.localdomain, localhost
```

Son los dominios para los que se considera el MTA como destino final. Se reparten localmente.

```
mynetworks = 192.168.1.0/24 127.0.0.0/8 [::ffff:127.0.0.0]/104 [::1]/128
```

Es la red o redes (IP's) en la que el servidor atenderá el correo que será enviado. Si aceptamos todas las redes se considerará un servidor **Open Relay.**

Recargamos y vemos estado con "sudo service postfix reload" y "sudo service postfix status". Comprobamos funcionamiento de Postfix mediante mailbox (sistema de almacenaje) y sendmail (incorporado por defecto con Postfix).

```
echo "correo de prueba" | sendmail j.rey@pizarraweb.com
```

Nuestro usuario adminubuntu, va a enviar un correo j.rey@pizarraweb.com directamente desde la terminal

El resultado lo podremos ver en otra pestaña en los registros de mail.log:

```
sudo tail -f /var/log/mail.log
```

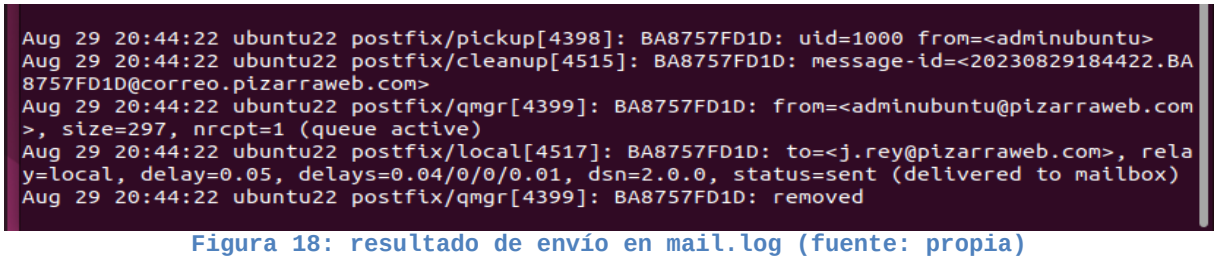

Figura 18: resultado de envío en mail.log (fuente: propia)

En este archivo podemos destacar como refleja el usuario que realiza el envío (from), el número identificador del mensaje, el correo destino a quien va dirigido (to), el estado de esta correo "status=sent" y para terminar indica la entrega a mailbox.

Durante la instalación de esta herramienta se crea el directorio */var/mail , donde* veremos un archivo con el nombre del usuario, el cual almacena los correos recibidos. Señalar que en este único documento se guardan todos los correos recibidos, lo que limita la gestión de las distintas correspondencias.

Una vez comprobado que el envío de correo funciona, procedemos a configurar el sistema de almacenaje de correo Maildir, es decir, cambiamos el buzón de correo Mailbox por este otro nuevo, el cual permite guardar cada correo en un archivo diferente. Editamos de nuevo el archivo */etc/postfix/main.cf* y añadimos la siguiente línea:

```
home_mailbox = Maildir/
```

Una vez hecho guardamos, recargamos el servicio y visualizamos nuestro archivo main.cf

```
service postfix reload
```

```
postconf -n
```

```
adminubuntu@ubuntu22:/etc/bind/zones$ cd /var/mail
adminubuntu@ubuntu22:/var/mail$ ls
j.rey
adminubuntu@ubuntu22:/var/mail$ cat j.rey
cat: j.rey: Permiso denegado
adminubuntu@ubuntu22:/var/mail$ sudo cat j.rey
From adminubuntu@pizarraweb.com  Tue Aug 29 20:43:53 2023
Return-Path: <adminubuntu@pizarraweb.com>
X-Original-To: j.rey@pizarraweb.com
Delivered-To: j.rey@pizarraweb.com
Received: by correo.pizarraweb.com (Postfix, from userid 1000)
        id 0523A7FD1C; Tue, 29 Aug 2023 20:43:52 +0200 (CEST)
Message-Id: <20230829184353.0523A7FD1C@correo.pizarraweb.com>
Date: Tue, 29 Aug 2023 20:43:53 +0200 (CEST)
From: adminubuntu <adminubuntu@pizarraweb.com>

correo de prueba

From adminubuntu@pizarraweb.com  Tue Aug 29 20:44:22 2023
Return-Path: <adminubuntu@pizarraweb.com>
X-Original-To: j.rey@pizarraweb.com
Delivered-To: j.rey@pizarraweb.com
Received: by correo.pizarraweb.com (Postfix, from userid 1000)
        id BA8757FD1D; Tue, 29 Aug 2023 20:44:22 +0200 (CEST)
Message-Id: <20230829184422.BA8757FD1D@correo.pizarraweb.com>
Date: Tue, 29 Aug 2023 20:44:22 +0200 (CEST)
From: adminubuntu <adminubuntu@pizarraweb.com>

correo de prueba

adminubuntu@ubuntu22:/var/mail$
```

Partimos de la instalación de *mailutils,* el cual se define así mismo como una "navaja suiza" por el amplio abanico de utilidades y demonios que ofrece para procesar un email. Básicamente nos permitirá gestionar nuestro correo en distintos formatos de buzón, en concreto el que nos interesa es Maildir.

Para ver las diferencias de gestión realizamos un envío de correo electrónico:

```
echo "Subject:Correo de prueba maildir"| sendmail j.rey@pizarraweb.com
```

Se puede ver que ahora en el directorio */home* de j.rey ha creado 3 subdirectorios (cur, new, tmp), y a su vez crea un archivo dentro de new con el correo que hemos enviado. Creará un archivo nuevo por cada correo nuevo recibido. Los leídos pasan al directorio *cur,* mientras que en *tmp* se guardan los mensajes que están en proceso de ser enviados.

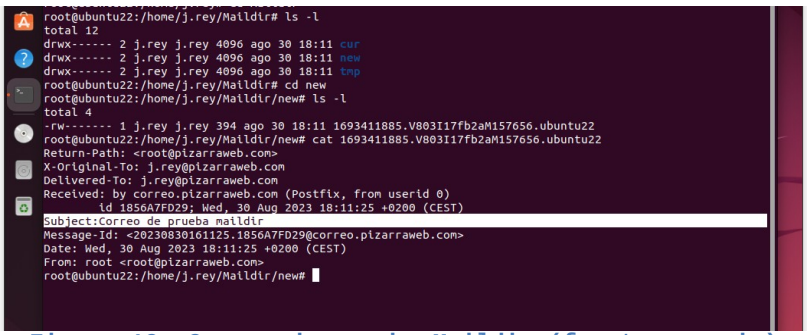

Figura 19: Correo de prueba Maildir (fuente: propia)

En la imagen anterior, podemos apreciar los mencionados directorios que además contienen los ficheros creados, que serán uno por cada correo recibido.

6.5.2 Instalación y configuración de Dovecot

Una vez que tenemos en funcionamiento el MTA Postfix, necesitaremos la función de un MDA que tiene como objetivo almacenar el correo y servirlo mediante los protocolos IMAP y POP3 al programa cliente (MUA).

En este caso vamos a utilizar Dovecot como MDA, por ser una herramienta de código abierto, ligera, rápida y segura.

Empleamos el siguiente comando para la instalación de ambos protocolos (aunque vamos a trabajar con IMAP en este caso).

```
apt install dovecot-core dovecot-imapd dovecot-pop3d
```

Una vez lo tengamos, nos vamos a su archivo de configuración previa copia de seguridad */etc/dovecot/conf.d/10-ssl.conf* y realizamos las siguientes modificaciones para habilitar SSL.

```
ssl = yes
ssl_cert = </etc/dovecot/private/dovecot.pem
ssl_key = </etc/dovecot/private/dovecot.key
```

*En las últimas versiones de Dovecot viene con un certificado propio "autofirmado". Habilitamos la autenticación en texto plano en */etc/dovecot/conf.d/10-auth.conf* .

```
disable_plaintext_auth = no
```

Ahora cambiamos el buzón de almacenamiento a maildir (por defecto estaba en mailbox) editando */etc/dovecot/conf.d/10-mail.conf*

Quitamos la almohadilla de la siguiente linea para que funcione correctamente el buzón maildir:

```
  mail_location = maildir:~/Maildir
```

Comentamos esta otra linea de mailbox para dejarla inutilizada y no cree conflictos:

```
#mail_location = mbox:~/mail:INBOX=/var/mail/%u
```

Para finalizar reiniciamos el servicio y comprobamos:

```
service dovecot reload
```

```
service dovecot status
```

En breves palabras, Postfix deja los mensajes en Maildir, y posteriormente es Dovecot quién los recoge en este mismo buzón.

6.5.3 Configuración de Thunderbird

Por último nos encontramos con el cliente de correo Thunderbird (MUA), de código abierto y multiplataforma. Éste ya viene instalado por defecto en muchas distribuciones Linux, como es el caso de Ubuntu.

Estos programas además de gestionar los buzones, nos permiten la creación, el envío, la consulta y la eliminación de los correos electrónicos. De manera adicional cuentan con ciertas extensiones que nos ayudan en la organización y seguridad como son las etiquetas, el calendario o incorporación de antivirus entre otros.

En nuestro trabajo utilizaremos el protocolo IMAP que utilizará el puerto 143 para acceder y administrar los correos en el buzón del servidor, y SMTP para el envío de correos, es decir, la transferencia por el puerto 25.

Empezamos por indicar el nombre de usuario, dirección de correo electrónico y su contraseña. Más abajo en configuración avanzada, informamos del nombre de servidor de correo, protocolo y puerto utilizado.

En este mismo apartado nos da la posibilidad de poder cifrar y aplicar una capa de seguridad mediante STARTTLS (transforma la comunicación insegura en segura por medio de SSL/TLS). Nos servimos de un certificado digital "auto-firmado" como nos muestra la figura 22.

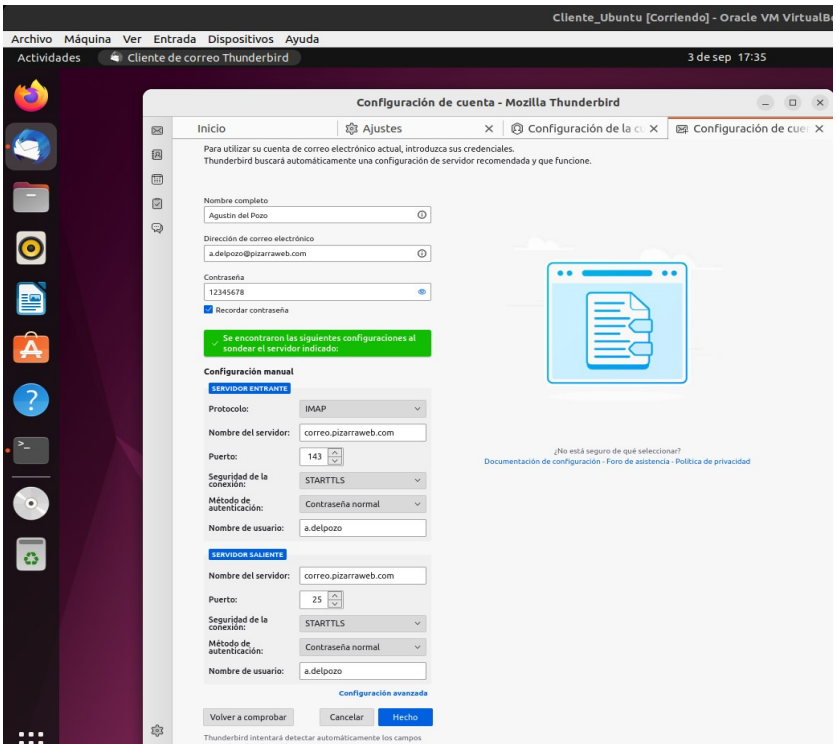

Figura 20: Configuración de cuenta Thunderbird (fuente: propia)

StartTLS es uno de los mecanismos de encriptación de correo electrónico más usado, ya que permite el uso de varios dominios y certificados en un mismo servidor. Sin embargo, a pesar de ofrecer esa capa de seguridad, algunos datos relevantes viajan sin cifrar como es el número de IP de nuestros equipos al establecer la conexión.

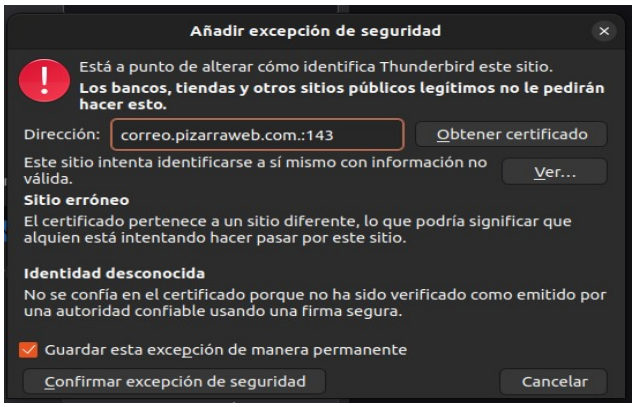

Figura 21: Excepción de seguridad a certificado auto-firmado (fuente: propia)

Con los dos usuarios creados probamos a realizar el envío de uno al otro. (Nos aparece advertencia de seguridad por tener un certificado auto-firmado), pero nos permite confirmar la excepción de seguridad.

Iniciamos varios envíos desde las distintas cuentas de usuario con resultado satisfactorio.

A continuación vemos el envío realizado desde el correo *j.rey@pizarraweb.com* hacia el correo de *a.delpozo@pizarraweb.com*

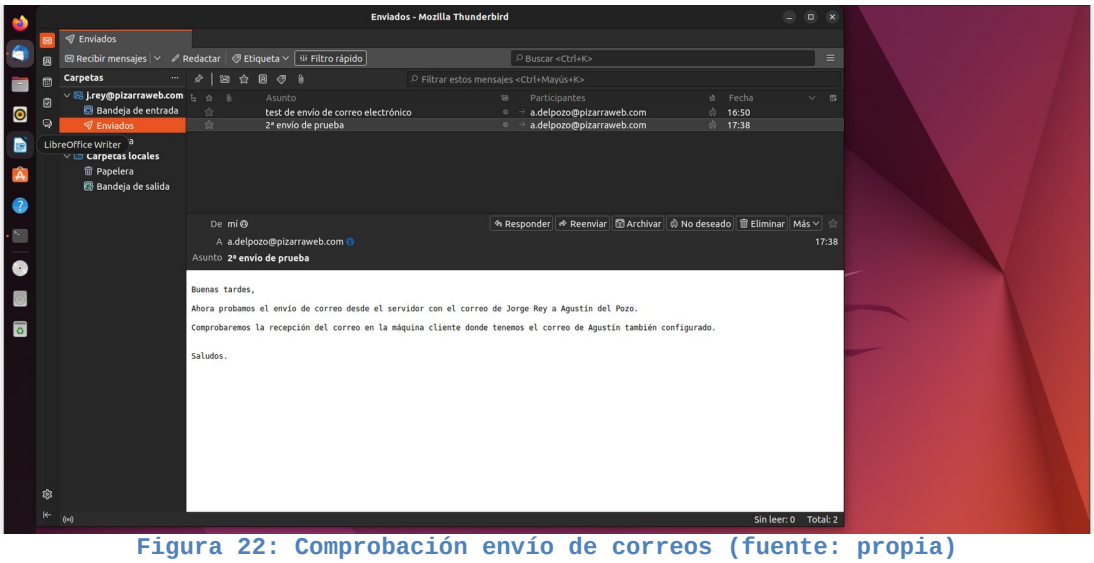

Figura 22: Comprobación envío de correos (fuente: propia)

Con la configuración y pruebas del cliente de correo terminamos esta sección, donde además de crear nuestro servidor de correo, hemos partido de la puesta en marcha de un servidor DNS.

Una parte esencial que tenemos que tener en consideración antes de dar salida a nuestros servidores hacia internet, es el registro de nuestro dominio por medio de las empresas (agentes) registradoras asignadas para ello que están coordinadas por la **ICANN.** En este punto, necesitaremos elegir un dominio disponible, elegir el tiempo que el dominio quedará registrado e indicar la IP del servidor al que apuntaremos mediante el DNS.

6.6 Instalación y configuración servidor SAMBA

Una característica muy útil para las empresas, es el poder compartir archivos dentro de una red privada, de forma que, cualquier usuario con los permisos necesarios pueda tener acceso a listar archivos, modificar o incluso crear nuevos para compartir con el resto.

Para nuestra empresa vamos a montar un servidor Samba. Este protocolo basado inicialmente en SMB (actualmente **CIFS**) nos permitirá comunicar y compartir archivos e impresoras en en una misma red independientemente del sistema operativo de los ordenadores (se podrá integrar sistemas Windows con otros Linux/Unix).

Si bien es cierto que aunque vamos a requerir usuarios con su correspondiente contraseña, debemos de asegurarnos que utilizamos versiones posteriores a la 3.0, ya que las anteriores toda la información viajaba sin cifrar (incluidas las contraseñas). No obstante, para este servicio únicamente lo vamos a tener disponible en nuestra LAN que es nuestro objetivo, sin posibilidad de acceder desde WAN.

Aprovecharemos la misma máquina virtual Ubuntu 22 que hemos utilizado para montar nuestro servidor DNS y servidor de correo electrónico. De esta manera nos permitirá compartir y economizar recursos sin exigir una gran potencia y tráfico excesivo.

Para ello empezamos con la instalación del servicio Samba e incluimos la excepción en nuestro firewall para no tener problemas de acceso.

Creación y configuración de usuarios en Samba. Debemos remarcar que los usuarios que creemos en samba, requieren estar previamente creados en nuestro sistema.

A continuación añadimos nuestros usuarios al grupo "sambashare" y "pizarraweb.com".

Sorprendentemente llego a un punto donde encuentro un problema con el usuario creado en la máquina virtual cliente durante su instalación. El usuario que ya teníamos dado de alta en el servidor , *a.delpozo*, aparece en máquina cliente como *adelpozo* (se ha creado sin el punto en la instalación gráfica, no lo ha cogido el sistema correctamente) al igual que en los ficheros */etc/passwd* , */etc/group* y su directorio */home.*

Tengo que realizar el cambio de nombre de usuario, grupo y home.

Seguidamente procedo a dar permisos de root al otro usuario que tengo en la máquina para poder realizar los cambios sin problemas.
Entramos al sistema con *adelpozo* para dar permisos a *j.rey,* modificando */etc/sudoers*

```
sudo visudo                    # visudo nos permite editar el archivo /etc/sudoers
```

Añadimos en el apartado "User privilege specification" la siguiente linea:

```
j.rey    ALL=(ALL:ALL)        ALL    #permitirá a  este usuario tener privilegios ilimitados
```

Ahora podemos incluir al usuario j.rey al grupo root.

```
sudo usermod -aG root j.rey
```

Después debemos de salir del sistema con el usuario *adelpozo* ya que si no tendría procesos activos y no nos permitiría realizar cambios. Después cambiamos el nombre de usuario y el nombre del grupo principal al que pertenece a *adelpozo* por su nuevo nombre *a.delpozo.*

```
sudo usermod -l a.delpozo adelpozo          #cambia el nombre de inicio de sesion
groupmod -n a.delpozo adelpozo
```

Finalmente también cambiamos el nombre del directorio

```
sudo usermod -d /home/a.delpozo -m adelpozo   #mueve el contenido al nuevo dir
```

Una vez solucionado los cambios del nombre de usuario, añadimos la carpeta que vamos a compartir dentro del disco duro creado para estos datos, y que realiza el montaje en la dirección */home/DATOS/samba*

Modifiqué el archivo /etc/samba/smb.conf añadiendo nuestra ruta al directorio que queremos compartir, le dí permisos para que fuera navegable e informé que no estuviera en solo lectura.

Sin embargo, tras reiniciar el servicio me encontraba con un error al intentar acceder desde otra máquina.

Figura 23: Fallo de autenticación en Samba (fuente: propia)

Tras varios intentos el resultado era el mismo. Intenté compartir la carpeta en modo gráfico con los distintos usuarios por si el problema fuera un tema de permisos, pero igualmente arrojaba un nuevo error.

A partir de ese momento, realicé varios cambios para adecuar el acceso y además no permitir acceso de invitado. Creé el grupo *pizarraweb.com,* incluí a estos usuarios dentro de este grupo, y modifiqué algunas líneas del fichero *smb.conf.*
Al principio en el apartado global "workgroup" indiqué este nuevo grupo y al final configuré nuestro apartado de la siguiente manera:

```
[pizarraweb.com]
comment = Samba Pizarraweb
path = /home/DATOS/
read only = no
writable = yes
create mask = 0770
directory mask = 0770
browsable = yes
valid users= @pizarraweb.com
guest ok = no
```

Una vez reiniciado el servicio conectamos con el servidor en esta dirección smb://ubuntu22/pizarraweb.com

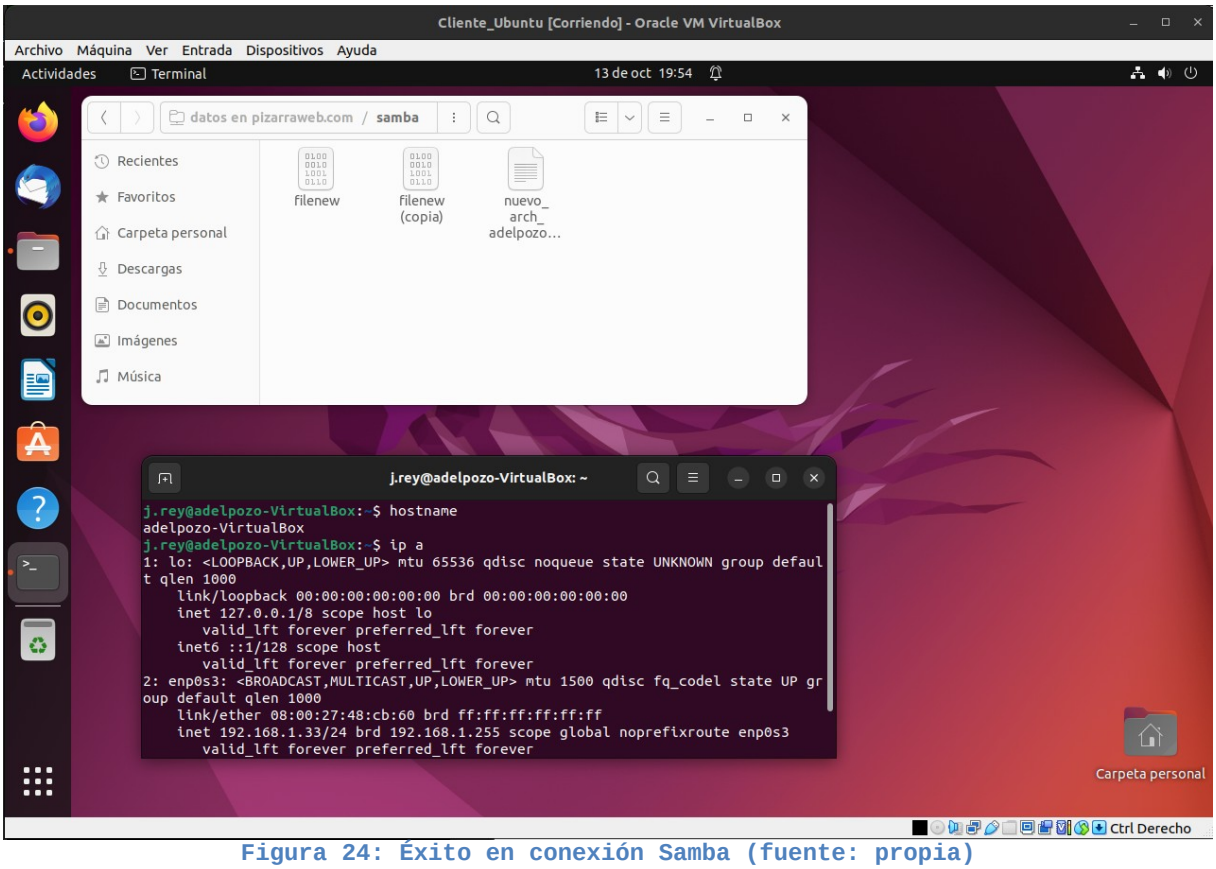

Figura 24: Éxito en conexión Samba (fuente: propia)

En la imagen anterior vemos como hemos podido acceder al recurso compartido de red desde la maquina *Cliente_ubuntu* a la máquina servidor *ubuntu22.*
A partir de este momento los usuarios pertenecientes a este grupo, conectados en la misma red local tendrán acceso a los recursos compartidos, pudiendo listar, leer, crear, modificar y borrar archivos o/y directorios.

Con esto hemos terminado de ver la instalación y configuración de este servidor de ficheros, trabajando a su vez en la gestión de grupos y usuarios de acuerdo a los inconvenientes encontrados.

En estos momentos tenemos todos los servicios necesarios preparados para implantar en nuestra empresa, junto con la infraestructura requerida para poder crear nuestra web o intranet , la cual vamos a desarrollar en la siguiente sección.

6.7 Creación de nuestra página web

Llegamos a este apartado final del proyecto en el que vamos a diseñar la base de datos que vamos a utilizar en nuestra intranet. Será necesaria para mostrar la información que el administrador o responsable irá actualizando de acuerdo a las comunicaciones que vayan surgiendo, nuevas reformas con presupuesto aceptado y publicaciones que vayan realizando también los empleados entre otros datos.

Una vez definido este paso, comenzaremos a trabajar en el código de nuestra página web, desarrollando una estructura en HTML para el acceso de cualquier trabajador dado de alta, pero además facilitando una entrada a una página específica para el administrador. Este acceso concreto servirá para que nuestro responsable publique la nueva información por medio de los conectores de PHP a nuestra BBDD. Todo ello lo maquetaremos en un apartado visual diseñado en CSS que pretenderá ser claro, agradable y organizado.

6.7.1 Diseñando la base de datos

Como en cualquier página web que requiere registros, empezamos por diseñar nuestra base de datos con las tablas que serán necesarias en las distintas secciones.

Nuestra tabla principal será la tabla "usuarios" que representará y dará acceso a nuestros trabajadores, pero será el administrador quién únicamente registrará el alta y las modificaciones pertinentes. En esta nos será suficiente con tener datos para identificar al usuario (nombre, apellidos, correo electrónico y contraseña). No obstante, nos interesará incluir el área o profesión que desempeña, reflejando la fecha de alta y generando un número de empleado que será único.

De igual manera tendremos nuestra tabla "cliente", la cual contendrá información personal como su DNI, nombre, apellidos, teléfono, dirección, la ciudad y el código postal (suele ser de cierta utilidad tener estos campos por separado para realizar consultas estadísticas por zonas). Como en el caso anterior, al cliente le generaremos un número de identificación para poder identificarle de manera inequívoca.

Otra de las tablas sería la de "reformas", que utilizaremos para recopilar y mostrar las fechas entre las que está prevista su ejecución, la información que facilitaremos a nuestros empleados, relacionando al cliente de la anterior tabla e incluyendo al empleado encargado del transporte durante la duración de esa obra (referenciado de la tabla principal). Toda reforma contará con un número de presupuesto identificativo e único. También indicaremos el estado de la reforma, pudiendo elegir entre tres opciones: pendiente, en progreso, completado (nos permitirá realizar filtros para consultas y reportes).

A continuación vemos el diagrama entidad - relación con las tablas utilizadas.

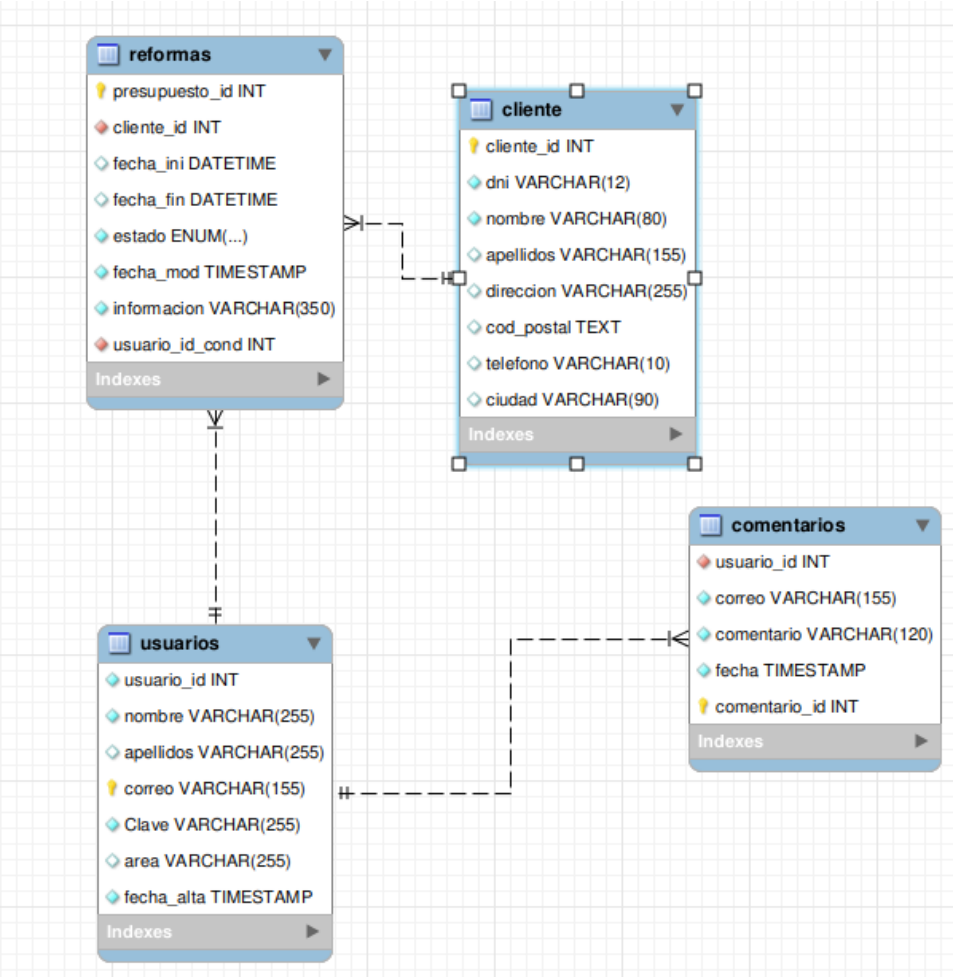

Figura 26: Diagrama entidad - relación en MySQL Workbench

(fuente: propia)

En el supuesto de realizar modificaciones en la estructura podemos utilizar "Reverse Engineer" en nuestro Workbench de MySQL para que actualice nuestro diagrama.

Para la gestión de la base de datos, nos ayudamos de phpMyAdmin instalado en uno de nuestros contenedores, y utilizamos el usuario root o el creado anteriormente "adminis" (recomendable no utilizar el usuario root). Esta herramienta nos permite una fácil creación de nuestras tablas y registros.

Sin embargo, a veces al realizar cambios que terminan en error, tendremos que recurrir a la línea de comandos para corregir algunas órdenes de forma manual.

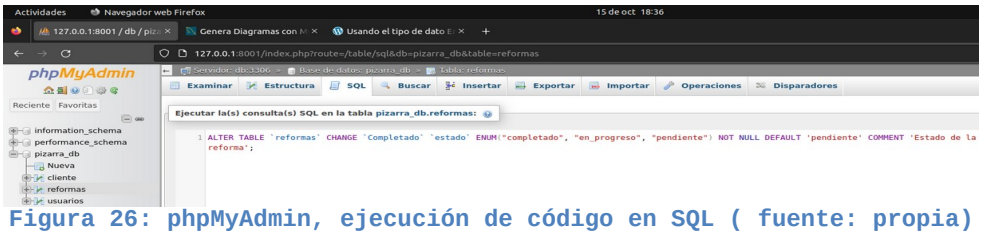

Figura 26: phpMyAdmin, ejecución de código en SQL (fuente: propia)

6.7.2 Pizarraweb Intranet

En este último tramo vamos a ver la estructura, detalles y funcionamiento de nuestra web.
Para esta creación hemos utilizado únicamente lenguaje de marcas HTML- CSS, PHP y como sistema gestor de bases de datos MySQL (ayudado por PhpMyadmin), creado todo el código desde cero, es decir, sin recurrir a plataformas de gestión de contenidos.

Como comentábamos anteriormente, se trata de una Intranet de una empresa, donde los usuarios (en este caso empleados) pueden entrar a visitar y beneficiarse de la información variada que el responsable les provee. Éste se encarga de realizar el alta de los usuarios, por lo que ellos directamente no pueden gestionar su propio registro y alta.
Esta diferenciación estructura nuestra web. Por un lado tenemos la web para el usuario (el trabajador), y el resto de estructura web queda enmarcado dentro de la gestión del responsable (en nuestro caso es el usuario Agustín del Pozo, a.delpozo@pizarraweb.com que tenemos declarado en una variable dentro del archivo de configuración config.php).
Una vez adelantada esta parte, dejamos la representación gráfica del flujo de llamadas a las distintas páginas, donde cada una realiza su propia función.

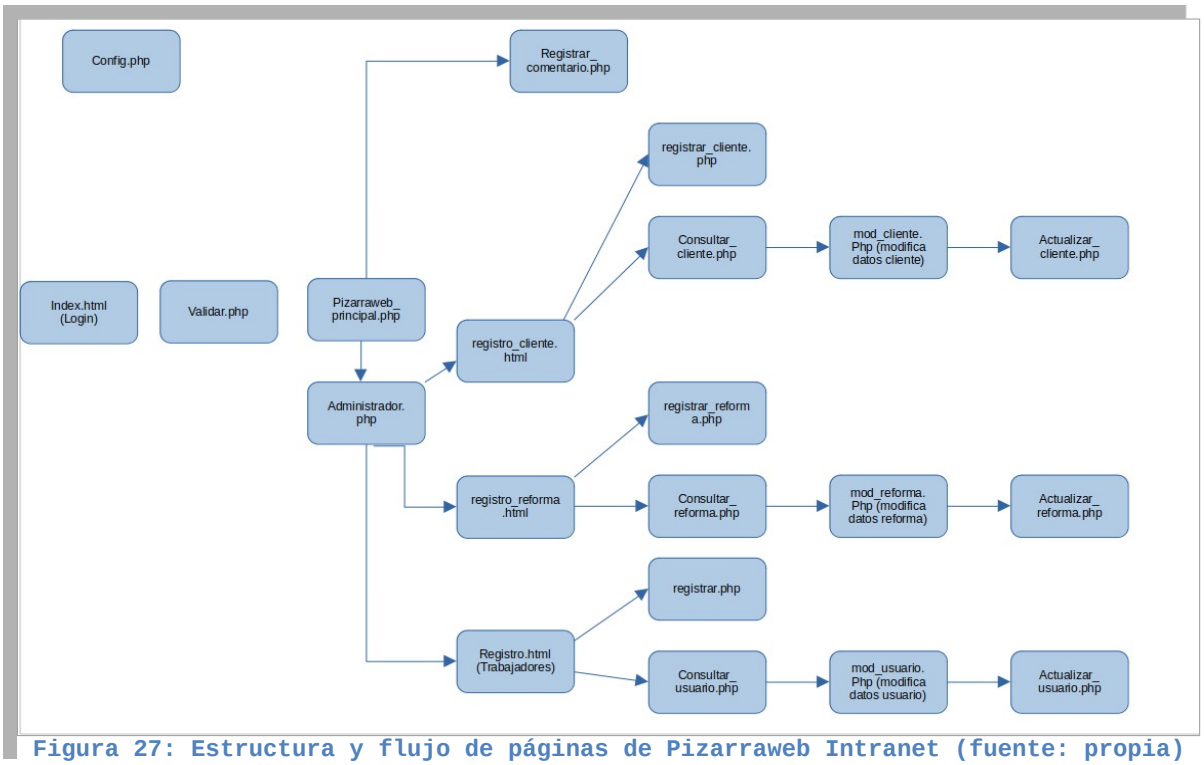

Figura 27: Estructura y flujo de páginas de Pizarraweb Intranet (fuente: propia)

Como se puede observar en el esquema anterior, la entrada la hacemos desde el archivo *index.html* (para acceder 127.0.0.1/index.html).
En la parte superior izquierda se indica el archivo de configuración para el acceso a las bases de datos, donde tenemos el servidor, usuario, password, base de datos, puerto, y la variable ***$responsable*** que almacena el usuario que tendrá acceso a la administración de la base de datos e intranet.
Una vez autenticado, el usuario entra en nuestra "pizarra-virtual", la página principal que hemos querido darle un aspecto de "tarjetones" flotantes conseguidos por medio de archivos "css".

Figura 28: Login Pizarraweb Reformas - Intranet (fuente: propia)

En ella encontramos varios bloques definidos:
Una cabecera con la bienvenida al usuario en cuestión, seguido de un primer bloque izquierdo dedicado a todos los usuarios que quieran dejar o compartir información de última hora.
La columna central da cabida a nuestra pizarra en sí, donde el administrador facilita la información más relevante para las obrás más próximas en el tiempo (esto interesa que se facilite a corto plazo, quizás de una semana para otra o poco más). Quedan reflejados el número ID de la obra, la fecha de inicio (ordenadas de manera descendente), datos del cliente, información detallada y conductor principal asignado.
Le sigue una tercera columna que hemos seccionado en cuatro pequeñas cajas, donde podrán encontrar el calendario laboral del año en curso, un botón para ver o descargar el convenio (en este caso del sector Construcción), agenda de contactos telefónicos para urgencias, anuncio de persona de guardia y empleado del mes. Cerramos con una última sección donde alojamos una foto de referencia de la reforma más importante del mes.
En la parte final, tras dar la posibilidad de salir, terminamos con un pie de página con los datos de la empresa y links que acabarían dando información de avisos legales, política de cookies y de privacidad.

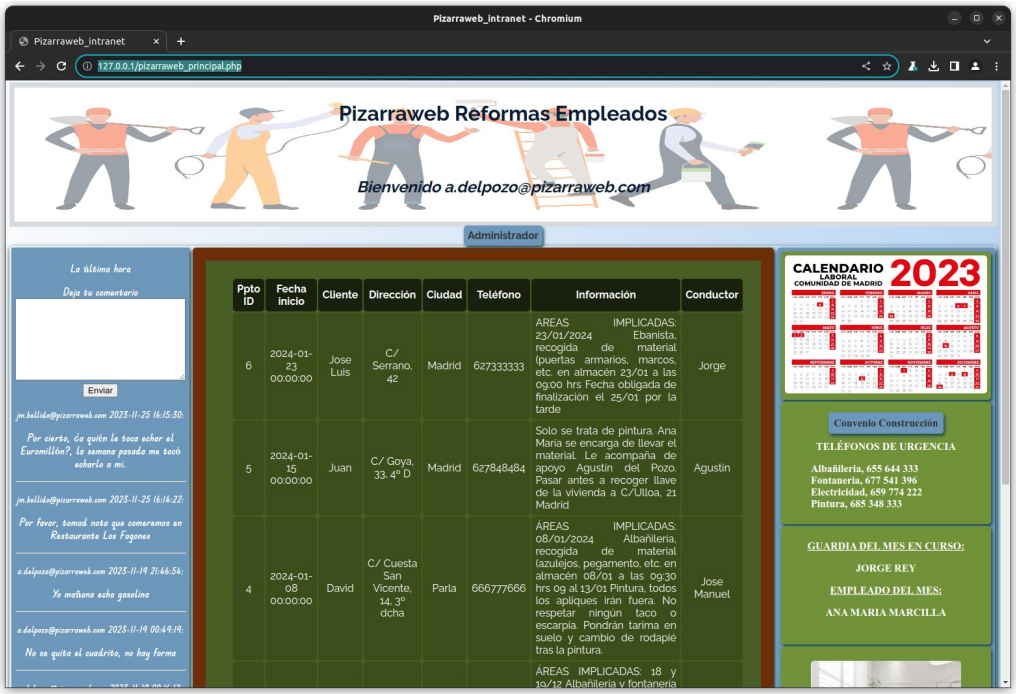

Figura 29: Pizarraweb - Página principal intranet (fuente: propia)

Hasta este punto tendrían acceso los usuarios, pero en caso de registrarse el administrador adicionalmente encontrará la sección (botón) que desemboca en el

panel de gestión de esta web, el cual dará la posibilidad de administrar los tres principales grupos implicados: clientes, reformas y trabajadores.

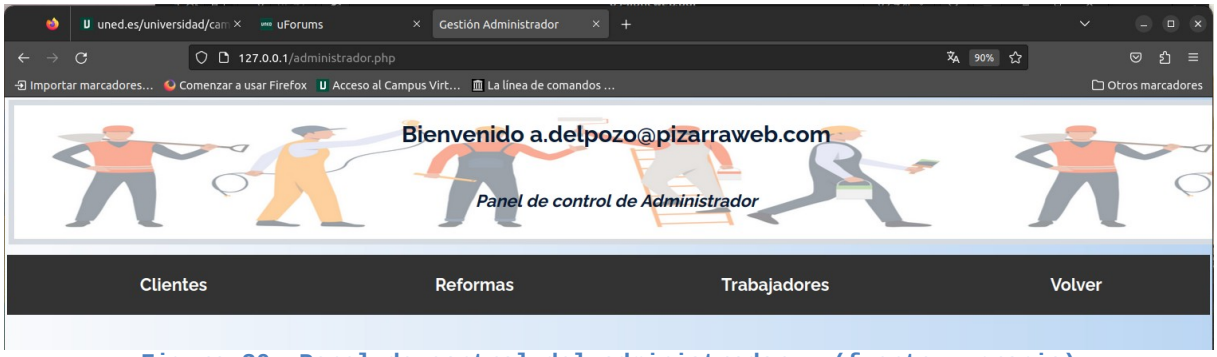

Figura 30: Panel de control del administrador (fuente: propia)

En cada uno de los apartados entraremos a una pantalla de altas y modificaciones para poder ir actualizando los datos de nuestra "pizarraweb". Además encontramos aquí mismo una opción de consultas que nos permitirá buscar y filtrar por ciertos criterios para dar con uno o varios registros, convirtiéndose a su vez en un apartado de reportes que podrían ayudar a analizar ciertos temas estadísticos de la empresa.

Figura 31: Altas y consultas (fuente: propia)

En el caso de una consulta sin filtros podremos acceder al listado total donde nos dará la opción de modificar un registro en caso de necesitarlo.

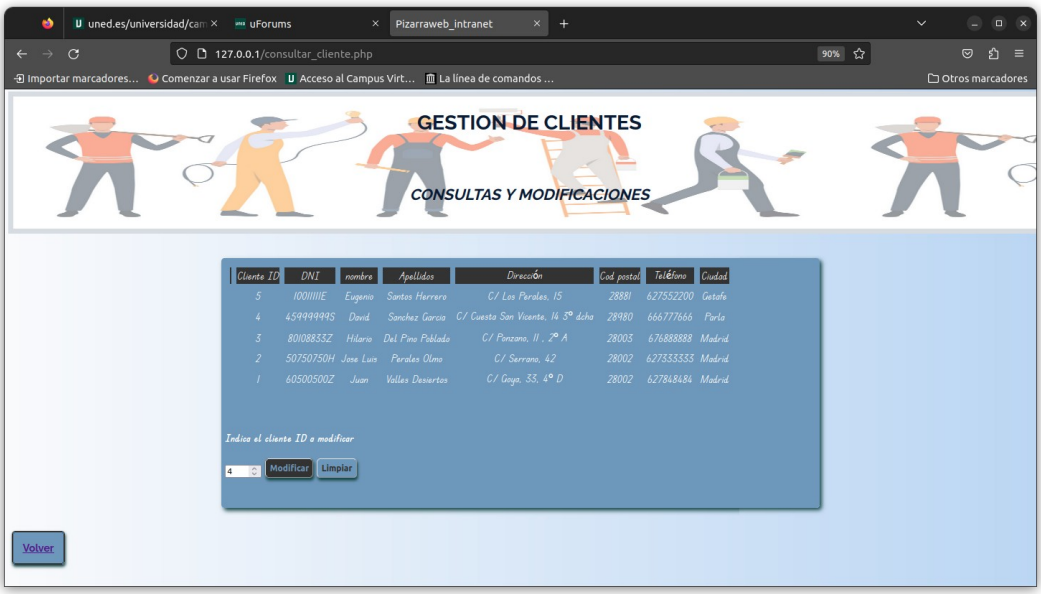

Figura 32: Resultado de consulta (fuente: propia)

En el caso de seleccionar tal registro accederemos a su ficha donde podremos editar y actualizar todos los campos habilitados para ello a excepción de los campos ID que son "autoincrementales" y tipo "timestamp" que lo facilita el propio sistema.

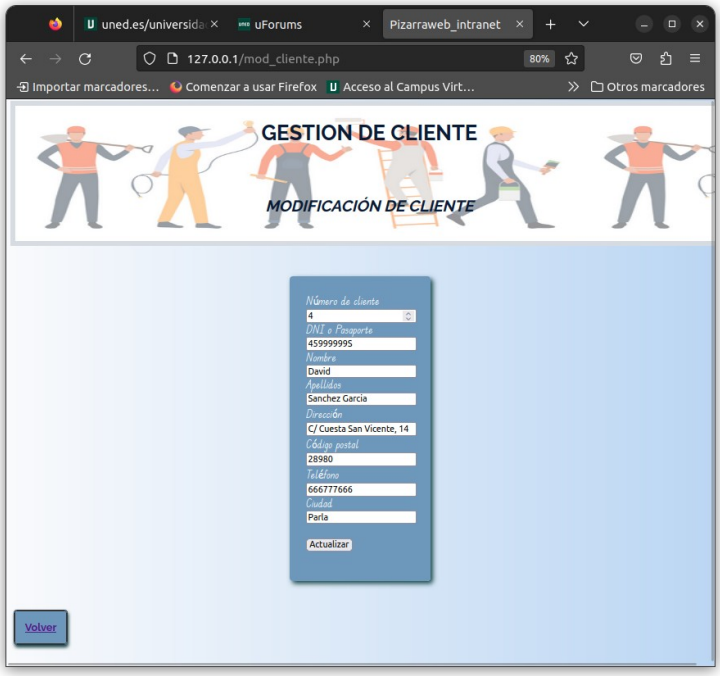

Figura 33: Modificación de cliente (fuente: propia)

Estas pantallas de gestión de clientes, se han replicado de un modo similar a los otros dos objetos de interés, las reformas y los usuarios (trabajadores), adaptando

las características en cada caso, por lo que ofrece una gestión de datos muy flexible y potente para nuestra intranet como resultado final.
Terminamos con una breve mención a los enlaces de avisos legales (en el caso de ser una empresa y no un autónomo), política de *cookies* y de privacidad que debemos incluir en nuestras web.
Aunque realmente según la **LSSICE** los avisos legales serían obligatorios en caso de que se perciba un ingreso económico a través de un medio como una web. En nuestro ejemplo, se trata de una intranet, por lo que no tiene ese objetivo.
Lo que sí debemos tener en cuenta es que los empleados tendrían que firmar un documento para el tratamiento de datos de carácter personal según **LOPD** que también dejamos reflejado en el enlace de cookies de la página principal.
En cuanto a la política de cookies, estamos obligados legalmente a informar sobre su almacenamiento en los equipos de los usuarios, que solo puede ser realizado con su consentimiento y aceptación previa. En este apartado se les informa de qué tipo de cookies se tratan y el motivo. Por un lado, las que son técnicamente necesarias (datos de inicio de sesión, idioma, etc.) y por otro las que no lo son (de seguimiento, de segmentación, análisis y redes sociales).

Por otro lado nos falta hablar del aspecto visual. Aquí seguimos las recomendaciones, por lo que principalmente damos formato a nuestras páginas con el uso de archivos externos. Este tipo de archivo los hemos querido dejar recogidos en una carpeta con todos los "css" creados en nuestro trabajo. En un primer momento se quería reutilizar el mismo archivo para varias páginas, pero lamentablemente cada una tenía una serie particularidades distintas a la otra, convirtiéndose en un archivo muy grande y poco intuitivo como para ser retomado por otra persona o equipo. No obstante nos decantamos por reutilizar la misma estructura y formato cuando se trataba de algún resultado de página similar.
Continuando con la utilización de este lenguaje, como queríamos dar un aspecto algo particular a nuestra web, necesitábamos la elección de unas fuentes de letra que llamasen la atención, por lo que recurrimos a "Google fonts" donde podemos conseguir un gran abanico de posibilidades dentro de su biblioteca, las cuales además son fuentes libres, de código abierto y gratis.

```
@import url('https://fonts.googleapis.com/css?family=Raleway:400,700');

@import url('https://fonts.googleapis.com/css2?
family=Edu+TAS+Beginner:wght@400;600&display=swap');
```

En cuanto al diseño hemos pretendido conservar un aspecto lineal a toda nuestra web, sin grandes cambios o sorpresas entre pantalla y pantalla, utilizando mismos colores y estilos.

Para ello nos servimos de la caracterización de los elementos principales que se repiten como *body, h1, h2, header* (con una imagen de fondo) y *footer* allí donde los hemos puesto.
En la pantalla principal nos decantamos por el uso del modo de *display* en "grid" con el fin de dar estructura a nuestra página. Se trata de un modelo de diseño en formato "parrilla" de gran uso en las nuevas webs, ya que nos ofrece una gran flexibilidad y simplicidad de uso (tan solo debemos indicar el número de columnas y de filas). En nuestro caso marcamos 3 columnas, fijando el ancho en "fr" (fracciones) para repartir el ancho de acuerdo a la pantalla o ventana disponible, dejando la parte central con la mayor porción. Además podemos crear dentro de una de estas secciones otros subbloques añadiendo un nuevo "grid" dividido únicamente en filas que es lo que conseguimos en la tercera columna donde incorporamos los documentos de interés compartidos, la agenda, y resto de información.

En el resto de páginas dedicadas a la gestión de la intranet ya no utilizamos este sistema de "css", simplemente realizamos un efecto de tarjetas flotantes con posiciones absolutas, definiendo un color de fondo de ese elemento con bordes redondeados y sombra en los "div" señalados para tal caso.
El último punto de los más importantes a señalar dentro del diseño es la personalización de los botones, donde intercambiamos los colores cuando pasamos el ratón por encima con el objetivo de dejar bien identificada la opción que vamos a elegir.

Como resultado global de todo lo abordado anteriormente obtendríamos nuestra preciada intranet, algo fuera de lo común, pero cumple con los objetivos y pretensiones que teníamos marcadas.

7. Conclusiones y recomendaciones

La consecución del proyecto la he podido completar como inicialmente la había planificado, aunque si es cierto que ha sufrido variaciones en cuanto a excesos de tiempos consumidos en algunas secciones del trabajo, pero que finalmente han podido ser compensados y absorbidos en puntos posteriores.

En cuanto a las tecnologías aplicadas, puedo concluir que, aunque los contenedores Docker han entrado en juego hace relativamente poco, su uso creciente es una realidad gracias a su portabilidad, ligereza y facilidad de uso una vez implantado (juega y jugará un papel muy importante de cara a *microservicios,* parece que los negocios irán utilizándolos cada vez más). Como contrapartida, señalar que requiere una curva de aprendizaje mayor a lo que esperaba en un principio si queremos entender y dominar bien la herramienta.

También puedo afirmar, que la tecnología anterior no es un completo sustituto de las máquinas virtuales, ya que en determinados momentos interesará dar alojamiento a algunos sistemas operativos completos para dar algunos servicios. Como conclusión se podría decir que ambas soluciones se complementan, y pueden convivir ofreciendo cada una las mejores características de sí mismas.

En relación a los servicios, se ha conseguido montar el servidor de compartición de ficheros y recursos SAMBA, el servidor DNS para dar posibilidad de funcionamiento al servidor de correo, aunque si sería clave aplicar mejoras de seguridad por medio de certificados.

Tanto en estos casos como en las herramientas utilizadas para servir la página web (Apache, MySQL, PHP, PhpMyadmin) se ha utilizado software libre y gratuito como habíamos ideado originalmente, siendo fieles al tema de costes del proyecto.

En el proceso de creación de la web, apuntar que durante ciertos tramos de las distintas páginas me he encontrado con bastantes incidencias y errores, que he podido ir subsanando por medio de la utilización de "traces" , que como bien nos enseñaron, han sido de gran ayuda para la detección del punto de ejecución donde estaba la incidencia (cuando la información del error emitido por el sistema no era suficiente).

Tan solo remarcar un gran momento de atasco en la creación de páginas para realizar consultas, donde a nuestro parecer era crucial que fueran flexibles, y de esta manera poder realizarlas marcando uno o varios filtros de búsqueda, o incluso ninguno para que devolviera todos los registros.

Aunque la instrucción de MySQL no tenía complicación, si lo tenía el poder trasladarlo al código antes de esa llamada, resultando muy difícil el poderlo encajar. Tras realizar distintas búsquedas, acababa siempre en soluciones que requerían la inclusión de scripts de otros lenguajes, pero pretendíamos evitar a no ser que fuera inevitable.

Finalmente en una de esas búsquedas, encontramos unas indicaciones de algo que no tenía mucho que ver, pero me dio la idea para solventar este punto.
La solución pasaba por meter en una variable la instrucción SQL. Ésta iría incrementando y encadenando la instrucción pasando por distintos tramos condicionales donde preguntaba por cada uno de los campos de búsqueda, y en caso de cumplirse, añade al final el nuevo requisito a cumplir.

Por último, en lo que a la intranet en general se refiere, cumple con el cometido que habíamos propuesto, cubriendo una necesidad que no suele existir en pequeñas empresas y que puede dar cierto juego al empresario. Creo que tiene una buena base para potenciarlo aún más, con ciertas mejoras de ampliación de información, nuevos campos de búsqueda, automatizar envío de correos con notificaciones a empleados, etc. Un punto relevante sería relacionar y cruzar estas bases de datos con otras existentes de facturación y clientes (lo podrían convertir en un pequeño y potente CRM). Esta mejora sería extrapolable a la parte de empleados.

Técnicamente, se podría ampliar funciones en los distintos servicios:

Empezaríamos por el servicio SAMBA donde podríamos instalar una impresora compartida en la LAN con nuestro grupo de trabajo.

Otro punto que hubiera sido interesante incluir es la instalación de un certificado SSL autofirmado creado con OpenSSL para ver su funcionamiento, aunque en realidad, para que nuestro certificado sea válido tendríamos que gestionarlo por alguna Autoridad Certificadora (Verisign, Digicert, Geotrust, etc.). Este sería utilizado por nuestro servidor de correo y también lo configuraríamos para que fuera utilizado en nuestra web.

Una buena opción posterior sería el migrar nuestras máquinas y contenedores a la nube, para tener alta disponibilidad, permitiéndonos además ganar en ciertos aspectos de seguridad.

Por medio de empresas de hosting nos dan ciertas facilidades como la posibilidad de instalar nuestros certificados desde la sección de seguridad de un "CPANEL" o panel de control. Desde aquí podremos generar un CSR (requerimiento para certificar) completando algunos datos. Una vez generado nos permitirá crear nuestra clave privada facilitando información del dominio y datos personales.

Una vez completados estos pasos nos iríamos a la sección del certificado en sí donde encontraríamos varios caminos para crearlo (correo electrónico, por medio de un archivo basado en https, etc. En el primer caso nos llegaría un correo con la clave que tendríamos que validar en la web de la autoridad de certificación, quién a su vez nos enviaría el certificado que copiaríamos para pegarlo en nuestro servidor en la sección de CRT para instalarlo finalmente.

En cuanto al código empleado en nuestra web, nos faltaría por añadir algunos métodos de seguridad para evitar ataques de bots (por medio de la inclusión de captchas en formularios, inteligencia artificial, plasmar información en la web que solo un bot podría ver, etc.). Lo que sí se ha aplicado ha sido la función mysqli *real_escape_string* en el archivo de validación de nuestro usuario y password para evitar ataques de inyección SQL.

En el apartado de CSS, adelantar que nuestra web está diseñada para un entorno de escritorio. Sería casi imprescindible hoy en día el adaptarla a otros tipos de dispositivos como serían tablets y teléfonos móviles, obteniendo lo que comúnmente se llama "web responsive", que optimiza de manera automática la visualización del contenido en función de estos. Se conseguiría indicando de manera adicional los comportamientos y tamaños a aplicar según unas medidas mínimas de ancho que indiquemos.

Con todo lo comentado conseguiríamos una solución muy seria en cuanto a fiabilidad y seguridad sin perder de vista el entorno amigable que se encontrará el usuario independientemente del dispositivo que utilice.

A continuación les facilito un enlace en la nube con acceso a los videos de prueba de este proyecto y la carpeta contenedora de los archivos de nuestra web "www":

https://drive.google.com/drive/folders/0Bwob-GmjKgG5dEc0NkpyVDVUdE0?resourcekey=0-1TFDk_hCQ7gQBNN0x0tdtw&usp=sharing

8. Referencias

[1] "VirtualBox End-user documentation". Disponible en https://www.virtualbox.org/wiki/End-user_documentation

[2] “Docker Desktop", "Docker Engine", "Docker Build", "Docker Compose", "Docker Hub". Disponible en https://docs.docker.com/desktop/

[3] "Docker Compose". Disponible en https://docs.docker.com/compose/

[4] "Docker tutoriales Docker" , https://www.hostinger.es/tutoriales/como-instalar-y-usar-docker-en-ubuntu

[5] "MDN Web Docs- Mozilla", https://developer.mozilla.org/es/docs/Web/CSS

[6] "Stack overflow", https://stackoverflow.com/questions/

[7] "Redes Plus", https://www.youtube.com/@RedesPlus

[8] "Código Binario - Docker", https://www.youtube.com/@CodigoBinarioo

[9] "CSS, Nicolas Schurmann", https://www.youtube.com/watch?v=wZniZEbPAzk&t=5434s

[10] "La aplicación de la ley de cookies europea en España", https://www.ionos.es/digitalguide/paginas-web/derecho-digital/la-ley-de-cookies-y-su-aplicacion-en-espana/

9. Bibliografía

[1] Implantación de aplicaciones web, Sánchez Asenjo, Jorge, grupo editorial Garceta, 2015.

[2] Servicios de Red e Internet, García Sánchez, Álvaro, Enamorado Sarmiento, Luis, González Sotillo, Álvaro , Sanz Rodríguez, Javier, grupo editorial Garceta, 2015.

[3] Gestión de Bases de Datos, López Montalban, I. , de Castro Vázquez M., grupo editorial Garceta, 2015.

[4] Fundamentos programación Linux, Matthew ,Neil , Stones, Richard, Anaya 2008.

[5] Seguridad y Alta Disponibilidad, Costas Santos, Jesús, RA-MA, S.A. 2014.

10. Anexos

10.1 Preparación de una partición desde terminal

Desde la terminal y con permisos de root, en primer lugar tenemos que crear la partición, pero antes comprobamos con qué disco vamos a trabajar

```
$ sudo fdisk -l
```

```
sudo fdisk /dev/sdb
```

```
n                #(nueva partición)
```

```
p                #(partición primaria)
```

```
w                #grabamos los cambios
```

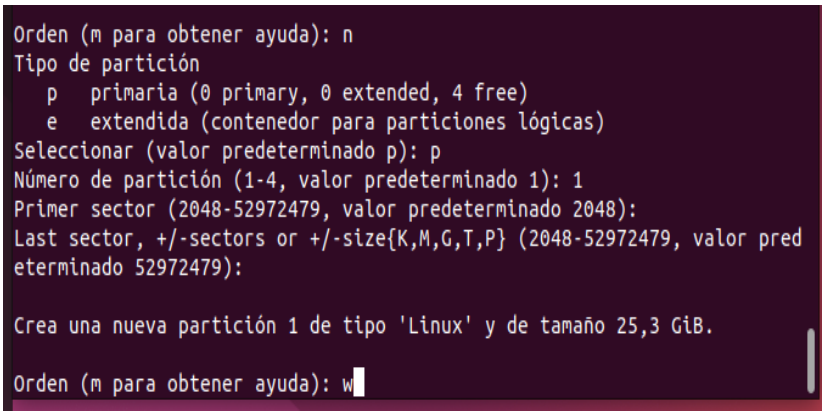

Figura 34. Creación de la partición con fdisk

Seleccionamos en este caso la "P" de partición primaria, y por defecto dejamos el valor 1 como número de partición. Terminamos con "W" para grabar los cambios.

Una vez creado, comprobamos que aparece y formateamos la partición con los siguientes comandos:

```
$ sudo lsblk -l
```

```
sudo mkfs -t  ntfs /dev/sdb1
```

10.2 Añadir repositorio de Docker a nuestro sistema

Previa a la instalación necesitamos instalar algunas librerías y paquetes para poder incorporar la clave GPG para el repositorio oficial de Docker.

```
sudo apt-get install ca-certificates curl gnupg
```

Hemos de añadir y configurar la clave GPG para el repositorio Docker quedando guardado en el directorio */etc/apt/keyrings/* (el archivo se llama docker.gpg)

```
sudo install -m 0755 -d /etc/apt/keyrings
```

-m, (chmod mode) le indicamos que aplique los permisos indicados al instalar.

-d, le estamos diciendo que el argumento que le sigue es un directorio (en él está el archivo docker.gpg)

Con curl descargamos la clave GPG, y añadimos (lo hacemos en una sola linea con "pipe")

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
```

-f fail silently, si el servidor falla, da el error 22 pero no da ninguna salida

sS, la primera "s" le indicamos que esté en modo quiet mode de cara a errores, y con la segunda "S" nos devuelve un error si falla.

-L, permite a curl seguir redirección a otras páginas (si se diera el caso).

-gpg --dearmor, desempaqueta archivo previamente blindado con "gpg --enarmor".

-o, le indicamos la dirección y nombre del archivo en nuestro dispositivo.

Damos permisos de lectura al archivo a todos los usuarios.

```
sudo chmod a+r /etc/apt/keyrings/docker.gpg
```

Finalmente añadimos el repositorio Docker para nuestro sistema mediante el siguiente comando (al fichero APT sources.list.d/docker.list):

```
echo \
  "deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
  "$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

Podemos comprobarlo mediante el siguiente comando, con el que veremos si está instalado o tiene un candidato para la instalación.

```
sudo apt-cache policy docker-ce
```

10.3 Creación y configuración de usuarios en Samba

Desde la terminal, actualizamos repositorios e instalamos el programa, para después asegurarnos de su funcionamiento. Posteriormente, procedemos a añadir la excepción en nuestro firewall de Ubuntu, y terminar añadiendo a nuestros usuarios al grupo sambashare. Es imprescindible crear a nuestros usuarios también en Samba.

```
sudo apt update
sudo apt install samba
systemctl status smbd                    #comprobamos el funcionamiento del servicio
sudo ufw allow samba
sudo usermod -aG sambashare $USER
sudo usermod -aG sambashare j.rey
sudo usermod -aG sambashare a.delpozo
sudo smbpasswd -a $USER
sudo smbpasswd -a j.rey
sudo smbpasswd -a a.delpozo
sudo pdbedit -L                          #vemos  los usuarios creados en samba
```