



App Cauldron. Elabora tu aplicación web.

Autor: Francisco Javier Loscos Gil

Tutor: Roberto Simal Martínez

Departamento: Informática y Comunicaciones

Ciclo de Grado Superior Administración de Sistemas Informáticos en Red

Centro: IES Pablo Serrano

Grupo: Distancia

Fecha: 13 junio 2024

APP CAULDRON. ELABORA TU APLICACIÓN WEB. RESUMEN	6
INTRODUCCIÓN	9
1. Objetivo y alcance del proyecto.....	9
2. Descripción de las secciones	9
DISEÑO PREVIO DEL PROYECTO	10
3. Propuesta de implementación	10
SISTEMA OPERATIVO – UBUNTU 22.04	10
4. Motivos para seleccionar la Distribución.....	10
APACHE Y CONFIGURACIÓN DEL HOST VIRTUAL.....	11
5. Resumen del proceso de instalación.....	11
DOCKER Y DOCKER-COMPOSE	11
6. Resumen del proceso de instalación.....	11
MARIADB Y PHP	12
7. Resumen del proceso de instalación.....	12
EL FICHERO YAML Y SERVICIOS SELECCIONADOS	13
8. El fichero yaml.....	13
9. Servicios seleccionados	14
9.1. Base de Datos (MySQL - Imagen Oficial)	14
9.2. Servidor Web (Nginx - Imagen Oficial).....	14
9.3. Servidor FTP (Pure-FTPd - Imagen Andrew Stilliard).....	15
9.4. Monitorización (Netdata - Imagen Oficial)	15
9.5. Gestión de aprendizaje (Moodle - Imagen Bitnami)	16
9.6. Videoconferencia (Jitsi Meet - Imagen Oficial).....	16
9.7. Wordpress (Wordpress - Imagen Oficial)	17
9.8. Office (WPS Office - Imagen linuxserver.io).....	17
APP CAULDRON – FORMULARIO Y SITIO WEB	18
10. Sitio web /var/www/html/appcauldron.com.....	18
10.1. index.html	19
10.2. cauldron.html	19

10.3.	cauldron2.php	20
10.4.	spells.php	21
10.5.	grimoire.php.....	22
10.6.	commands.php	23
10.7.	info.html	24
	CONCLUSIONES.....	25
	BIBLIOGRAFÍA	26
	ANEXO I – INSTALACIÓN MAQUINA VIRTUAL UBUNTU	27
	ANEXO II – INSTALACIÓN DE APACHE Y VIRTUAL HOST	28
	ANEXO III – INSTALACIÓN DOCKER Y DOCKER-COMPOSE.....	33
	ANEXO IV – INSTALACION MARIADB Y PHP.....	36
	ANEXO V – APPCAULDRON.COM - /var/www/html/appcauldron.com/	39
10.8.	index.html	39
10.9.	cauldron.html	40
10.10.	cauldron2.php.....	42
10.11.	spells.php	44
10.12.	grimoire.php.....	47
10.13.	commands.php	48
10.14.	info.html	52
	ANEXO VI – PLANTILLAS - /var/www/html/appcauldron.com/templates/	53
11.	Base de Datos (MySQL - Imagen Oficial)	53
11.1.	database	53
11.2.	database.volumes	53
11.3.	database.info	53
12.	Servidor Web (Nginx - Imagen Oficial).....	54
12.1.	nginx	54
12.2.	nginx.info	54
13.	Servidor FTP (Pure-FTPD - Imagen Andrew Stilliard)	55
13.1.	ftp	55

13.2.	ftp.info	55
14.	Monitorización (Netdata - Imagen Oficial)	56
14.1.	netdata.....	56
14.2.	netdata.volumes	56
14.3.	netdata.info	56
15.	Gestión de aprendizaje (Moodle - Imagen Bitnami)	57
15.1.	lms	57
15.2.	lms.volumes.....	58
15.3.	lms.info	58
16.	Videoconferencia (Jitsi Meet - Imagen Oficial)	59
16.1.	jitsi.....	60
16.2.	jitsi.info.....	66
17.	Wordpress (Wordpress - Imagen Oficial)	67
17.1.	wordpress	68
17.2.	wordpress.volumes.....	68
17.3.	wordpress.info	69
18.	Office (WPS Office - Imagen linuxserver.io).....	70
18.1.	office	70
18.2.	office.info	70
	ANEXO VII – SCRIPTS - /var/www/html/appcauldron.com/scripts/.....	71
19.	Ejecución en /var/www/html/appcauldron/cauldron2.php	71
19.1.	magic_cauldron.sh	71
20.	Botones /var/www/html/appcauldron.com/spells.php	72
20.1.	cast_spell.sh (Crear hechizo)	72
20.2.	stop_spell.sh (Parar hechizo)	72
20.3.	resume_spell.sh (Continuar hechizo)	72
20.4.	undo_spell.sh (Deshacer hechizo)	72
21.	Ejecución en /var/www/html/appcauldron/commands.php	73
21.1.	docker_ps.sh	73
21.2.	docker_container_ls.sh.....	73

21.3.	docker-compose_ls.sh.....	73
21.4.	docker_network_ls.sh.....	73
21.5.	docker_volume_ls.sh.....	73
21.6.	docker_images.sh	73
22.	Botones /var/www/html/appcauldron.com/commands.php	74
22.1.	delete_stopped_containers.sh (Borrar contenedores parados)	74
22.2.	delete_containers.sh (Borrar TODOS los contenedores)	74
22.3.	delete_active_containers.sh (Borrar contenedores activos)	74
22.4.	delete_anon_networks.sh (Borrar redes no referenciadas)	74
22.5.	delete_anon_vols.sh (Borrar volumenes no referenciados).....	74
22.6.	delete_volumes.sh (Borrar TODOS los volumenes)	74
	ANEXO VIII – LOGOS - /var/www/html/appcauldron.com/images/	75
23.	Herramientas y proceso para la creación del logo	75
23.1.	Half banner - 234x60	76
23.2.	Full banner - 468x60.....	76
23.3.	Leaderboard - 728x90	77
23.4.	3:1 rectangle - 300x100.....	77
	ANEXO IX - CSS - /var/www/html/appcauldron.com/css/	78
24.	appcauldron-style.css	78
	ANEXO X – OBJETIVOS Y CALENDARIO DE TRABAJO	80
25.	Propuesta de objetivos	80
26.	Propuesta calendario de trabajo	81

APP CAULDRON. ELABORA TU APLICACIÓN WEB. RESUMEN

AppCauldron es una solución web basada en apache que nos permite seleccionar una serie de servicios a través de un formulario, que hemos denominado el caldero “The Cauldron” (Ilustración A).

Al usuario apache www-data se le ha dado permiso para lanzar scripts de una carpeta concreta como usuario del sistema.

The screenshot shows a web application titled "APP CAULDRON". At the top, there is a navigation bar with links: Home, The Cauldron, Hechizos, Grimoario, Info, and a button labeled "Contemplar el Universo". Below the navigation bar, there is a sidebar on the left with the text "Selecciona los ingredientes de tu aplicación web". To the right of the sidebar is a list of service options, each preceded by a checkbox:

- Base de Datos (MySQL - Imagen Oficial)
- Servidor Web (Nginx - Imagen Oficial)
- Servidor FTP (Pure-FTPD - Imagen Andrew Stilliard)
- Monitorización (Netdata - Imagen Oficial)
- Gestión de aprendizaje (Moodle - Imagen Bitnami)
- Videoconferencia (Jitsi Meet - Imagen Oficial)
- Wordpress (Wordpress - Imagen Oficial)
- Office (WPS Office - Imagen linuxserver.io)

At the bottom of the list is a button labeled "Arrojar al caldero".

Ilustración A

A continuación, se lanza un script (`magic_cauldron.sh`) que recibe las variables correspondientes a la selección a través del formulario. Conforme a ellas añade código a través de una plantilla preparada para cada servicio con la configuración adecuada. Esto da forma a un archivo, `spell.yaml`, que hemos denominado hechizo, y que nos permitirá con el comando `docker-compose` iniciar los contenedores que contengan las imágenes de los servicios deseados.

Adicionalmente generamos un archivo `grimoire.txt` con las instrucciones de uso.

Por la magnitud y complejidad a la que se puede llegar, se ha decidido configurar un total de 8 servicios basados en imágenes Docker disponibles online. Algunos requieren servicios adicionales para su correcto funcionamiento, llegando a desplegar hasta 14 contenedores actualmente.

Se ha realizado un trabajo de configuración y revisión de puertos para que puedan convivir todas a la vez en una misma red de Docker.

Para llevar más allá a AppCauldron, desde el menú “Hechizos” (Ilustración B) se han habilitado botones para lanzar el comando docker-compose con nuestro archivo generado, iniciando así los servicios en contenedores.

Adicionalmente, podemos parar los servicios, reanudarlos, eliminar los contenedores o eliminar el hechizo.

```
services:
  wps-office:
    image: lscr.io/linuxserver/wps-office:latest
    container_name: wps-office
    security_opt:
      - seccomp:unconfined #optional
    environment:
      - PUID=1000
      - PGID=1000
      - TZ=Europe/Madrid
    volumes:
      - ~/Documentos:/config/Documents
      - ~/Escritorio:/config/Desktop
      - ~/Configuracion:/config
    ports:
      - '3000:3000'
      - '3001:3001'
    shm_size: "1gb"
    restart: unless-stopped
  networks:
    network:
```

Ilustración B

Desde el menú Grimorio (Ilustración C), podemos consultar la información (el archivo grimoire.txt generado previamente) para revisar que el servicio o servicios seleccionados están funcionando. Se puede acceder a la información de puertos, usuarios, contraseñas y comandos de terminal para acceder directamente a los servicios, o enlaces a las imágenes y la documentación oficiales.

```
Servidor Web (Nginx - Imagen Oficial)
=====
TEST: Comprobar visualización página web nginx alojada en nuestro host en: /var/www/html/nginx a través del puerto 9090.
Está preparado para abrir el documento index.html situado en la ruta actualmente o lo que hayamos incluido.
http://localhost:8090
Nginx - Imagen Oficial:
https://hub.docker.com/\_/nginx

Videoconferencia (Jitsi Meet - Imagen Oficial)
=====
TEST: Acceder a Jitsi Meet por el puerto 8445 (HTTPS).
Se puede entrar por el puerto 8200 (HTTP) pero no funcionaría el micrófono y la cámara.
Comprobar que se puede iniciar una reunión y funciona correctamente.
El archivo de configuración .env se encuentra en:
/var/www/html/appcauldron.com/generatedfiles/.env
En el se indican los puertos que se configuran entre otros ajustes.

Enlaces:
http://localhost:8200
https://localhost:8445
Jitsi Meet - Imagen Oficial:
https://github.com/jitsi/docker-jitsi-meet
```

Ilustración C

Finalmente, desde el menú Contemplar el Universo (Ilustración D), se puede consultar la salida de los principales comandos que se utilizan en Docker y que se lanzan a través de scripts por el usuario www-data.

The screenshot shows the 'APP CAULDRON' interface with a navigation bar at the top. The main area displays command history and Docker status. The command history includes:

- docker ps**: Shows one container named 'appcauldron-netdata-1' with ID 'a9054d5f04da'. It has an image 'netdata/netdata', command '/usr/bin/run.sh', was created about a minute ago, is up, and is healthy. It has port 0.0.0.0:19999 mapped to 19999/tcp. The name is 'appcauldron-netdata-1'.
- docker container ls -a**: Shows the same container 'appcauldron-netdata-1'.
- docker-compose ls**: Shows one service named 'appcauldron' which is running (1). The configuration file is '/var/www/html/appcauldron.com/generatedfiles/spell.yaml'.
- docker network ls**: Shows one network named 'appcauldron_network' with ID '920c29b3e03e'. It is a bridge network with local scope.
- docker volume ls**: Shows five volumes:

DRIVER	VOLUME NAME
local	appcauldron_db_moodle-volume
local	appcauldron_db_mysql-volume
local	appcauldron_db_wordpress-volume
local	appcauldron_moodle-volume
local	appcauldron_wordpress-volume

Buttons for 'Borrar contenedores parados' and 'Borrar TODOS los contenedores' are visible under the 'docker container ls -a' section. Buttons for 'Borrar contenedores activos' and 'Borrar redes no referenciadas' are visible under the 'docker compose ls' and 'docker network ls' sections respectively. A 'Borrar volúmenes no referenciados' button is visible under the 'docker volume ls' section.

Ilustración D

Es posible eliminar parte o toda la información de los contenedores, volúmenes, redes e imágenes a través de botones, convirtiendo a AppCauldron en una herramienta ágil para trabajar, y comprender de manera muy visual el funcionamiento de Docker y docker-compose.

Se ha trabajado en una forma modular que permite escalar fácilmente la cantidad de servicios, su puesta en marcha y su documentación a través de la carpeta *templates* del proyecto, como se puede observar en la Ilustración E.

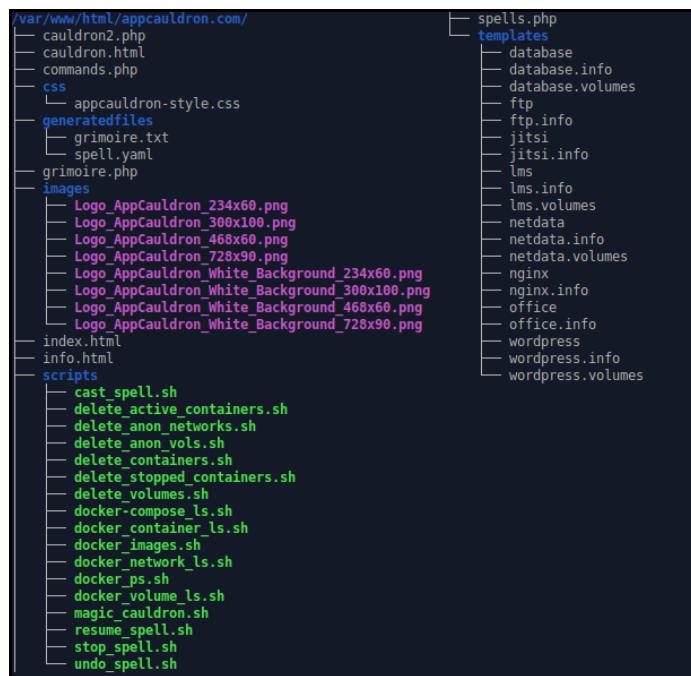


Ilustración E

INTRODUCCIÓN

1. Objetivo y alcance del proyecto

El objetivo del proyecto es desarrollar una aplicación web, con la perspectiva de un equipo profesional, que pueda iniciar los servicios escogidos a través de un formulario web utilizando imágenes Docker. No es un proyecto real que lleve idea de publicarse, pero se busca llegar a un punto en que se vea una utilidad, complejidad y seriedad de un desarrollo real.

El conocimiento personal sobre Docker y los contenedores es bastante básico a la hora de comenzar a abordar el mismo, con lo que espero que sirva de aprendizaje sobre un tema interesante y con potencial, y con una buena aplicación en la vida real.

2. Descripción de las secciones

Comenzamos con un DISEÑO PREVIO DEL PROYECTO, donde planteamos un concepto inicial para abordarlo, un desglose de tareas y un tiempo estimado para realizarlas.

Continuamos con la SISTEMA OPERATIVO – UBUNTU 22.04 y su instalación, que consideramos idóneo para las características de nuestro proyecto.

En la siguiente sección, APACHE Y CONFIGURACIÓN DEL HOST VIRTUAL, procedemos con la instalación de apache y la configuración de un host virtual para alojar <http://appcauldron>.

Damos permiso al usuario apache www-data para ejecutar scripts como usuario appcauldron de la carpeta /var/www/html/appcauldron.com/scripts/

Instalamos DOCKER Y DOCKER-COMPOSE. Docker es una plataforma de software que nos permite crear, probar e implementar aplicaciones rápidamente, empaquetando todo el software necesario para su ejecución en contenedores. Docker-compose nos permite definir una serie de servicios, redes y volúmenes en un único archivo YAML y ponerlo en marcha con un comando. Este archivo es el que formaremos a través de nuestro formulario web.

A continuación MARIADB Y PHP. Mariadb es necesario para probar conexiones a las bases de datos que vamos a crear con Docker, y PHP para crear el formulario y nuestra web.

En EL FICHERO YAML Y SERVICIOS SELECCIONADOS explicamos las partes principales del fichero YAML que crearemos, y revisaremos los 8 servicios escogidos para el formulario.

En APP CAULDRON – FORMULARIO Y SITIO WEB damos cierre al proyecto con la creación del formulario web para crear el archivo YAML y del sitio web App Cauldron elaborado para alojar al mismo. Conseguimos una mayor funcionalidad como plataforma de elaboración de aplicaciones web personalizada.

Acabaremos con las CONCLUSIONES del proyecto y el trabajo realizado. Posteriormente se encuentran los distintos ANEXOS correspondientes a las instalaciones realizadas y los archivos creados.

DISEÑO PREVIO DEL PROYECTO

3. Propuesta de implementación

Se ha considerado crear una única máquina virtual en Ubuntu.

En la máquina se instalará apache y Docker.

Se creará un formulario web que permita seleccionar los servicios que se deseen incluir en la arquitectura de la aplicación web.

Se valorará la creación de contenedores con los servicios personalizados o los disponibles en recursos online a incluir en la selección del formulario.

Una vez seleccionados se generará un archivo yaml o Dockerfile (aún por concretar) que nos permitirá desplegar el contenedor con los servicios seleccionados a través del formulario utilizando la herramienta Docker.

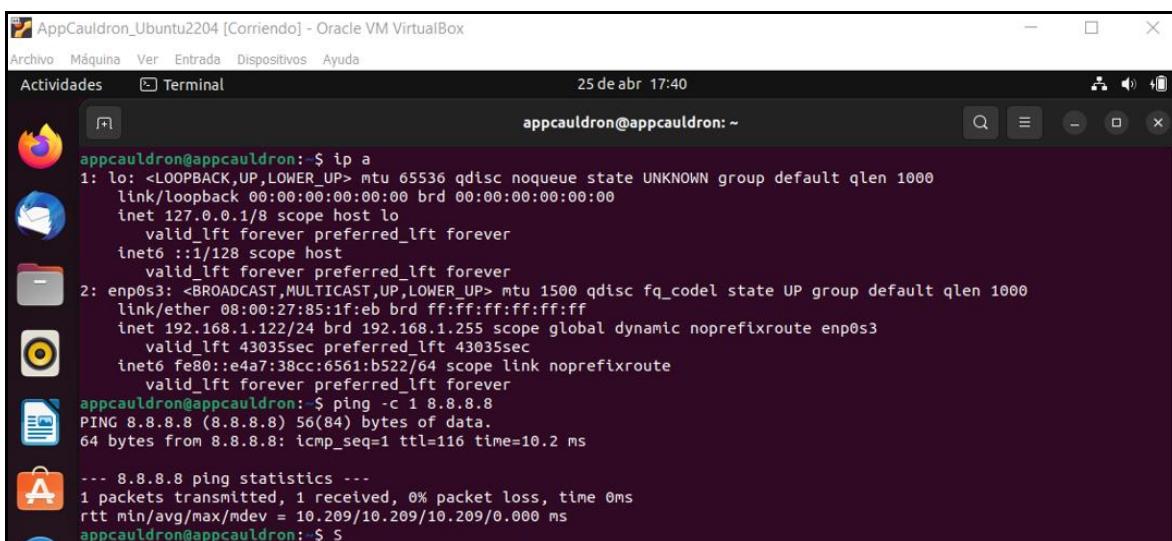
Se comprobará el correcto despliegue y funcionamiento de los servicios seleccionados.

SISTEMA OPERATIVO – UBUNTU 22.04

4. Motivos para seleccionar la Distribución

Como se puede seguir en el ANEXO I – INSTALACIÓN MAQUINA VIRTUAL UBUNTU, se procede a instalar una máquina con sistema operativo Ubuntu 22.04 en VirtualBox.

Se ha trabajado durante el grado con esta instalación y reúne los requisitos para cubrir nuestras necesidades. Instalar apache para alojar nuestra web y host virtual. Instalar Docker y docker-compose para gestionar los contenedores. Instalar Mariadb para conectar a las bases de datos que levantemos y PHP para utilizarlo en la web. Nos apoyaremos de scripts de bash. Todo esto lo convierte en un sistema ideal para nuestro proyecto.



The screenshot shows a terminal window titled "AppCauldron_Ubuntu2204 [Corriendo] - Oracle VM VirtualBox". The terminal displays the following command-line session:

```
appcauldron@appcauldron:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:85:1f:eb brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.122/24 brd 192.168.1.255 scope global dynamic noprefixroute enp0s3
            valid_lft 43035sec preferred_lft 43035sec
        inet6 fe80::e4a7:38cc:6561:b522/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
appcauldron@appcauldron:~$ ping -c 1 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=116 time=10.2 ms
--- 8.8.8.8 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 10.209/10.209/10.209/0.000 ms
appcauldron@appcauldron:~$
```

Ilustración F

APACHE Y CONFIGURACIÓN DEL HOST VIRTUAL

5. Resumen del proceso de instalación

Procedemos con la instalación de apache y el host virtual para el sitio web AppCauldron.

Creamos una página de prueba para verificar que funciona <http://appcauldron>

Configuramos el firewall ufw y abrimos los puertos 80 y 443.

Damos permiso al usuario www-data para ejecutar scripts como usuario appcauldron de la carpeta /var/www/html/appcauldron.com/scripts/.

Se puede consultar el paso a paso de la instalación en el ANEXO II – INSTALACIÓN DE APACHE Y VIRTUAL HOST.



Ilustración G

DOCKER Y DOCKER-COMPOSE

6. Resumen del proceso de instalación

Instalamos el repositorio y la clave gpg de Docker y procedemos con su instalación.

Comprobamos su funcionamiento. Añadimos el usuario appcauldron al grupo docker para evitar utilizar sudo.

Instalamos docker-compose y comprobamos que se ha instalado correctamente.

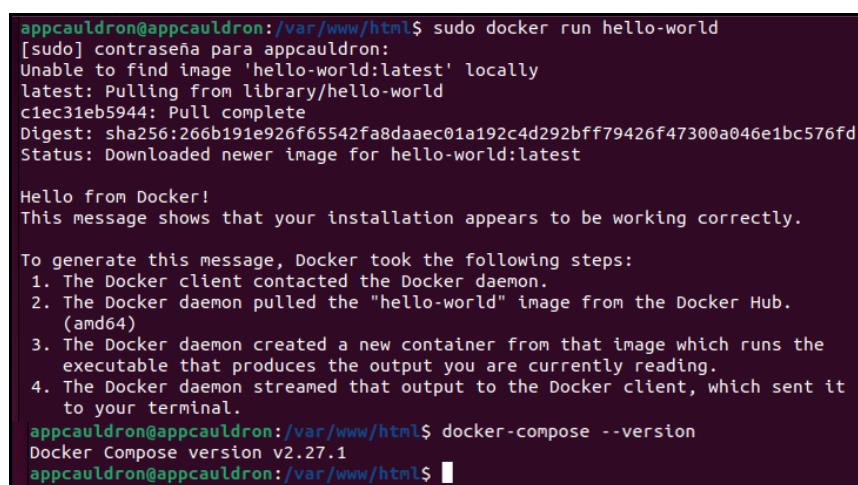


Ilustración H

Se puede consultar el paso a paso de la instalación en el ANEXO III – INSTALACIÓN DOCKER Y DOCKER-COMPOSE.

MARIADB Y PHP

7. Resumen del proceso de instalación

Instalamos mariadb-server y ejecutamos el script de seguridad.

Instalamos php, el módulo apache y los paquetes opcionales de scripts, MySQL y PostgreSQL.

Comprobamos creando y accediendo a una página de prueba: <http://appcauldron/testphp.php>

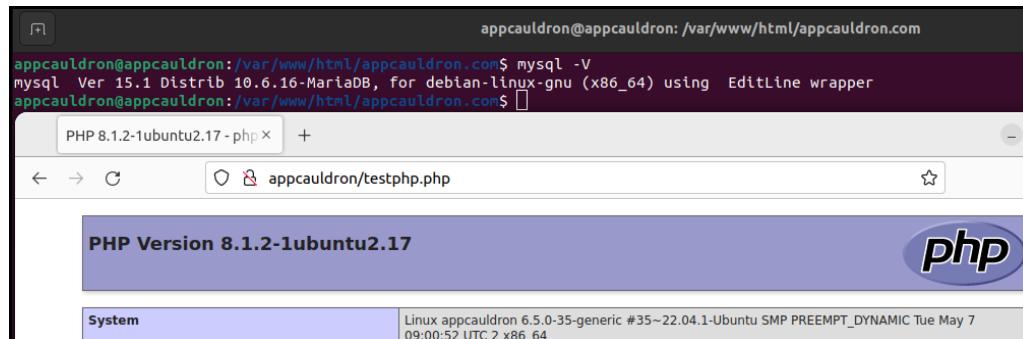


Ilustración I

Se puede consultar el paso a paso de la instalación en el ANEXO IV – INSTALACION MARIADB Y PHP.

EL FICHERO YAML Y SERVICIOS SELECCIONADOS

8. El fichero yaml

Tras revisar el funcionamiento de Docker y docker-compose, vamos a generar un fichero en formato yaml con la configuración de los servicios elegidos para lanzar con docker-compose.

Seleccionamos una serie de servicios de los disponibles en repositorios Docker online, principalmente en <https://hub.docker.com/> o en <https://github.com/>, y preparamos una configuración adecuada para cada uno de ellos.

El formato del archivo docker-compose a utilizar es idóneo para preparar plantillas. Junto al script magic_cauldron.sh, que se ejecuta al enviar el formulario, se genera el archivo spell.yaml. Con el comando docker-compose -f spell.yaml up levantamos los servicios.

Un formato básico de fichero comienza con el apartado services, y se enumeran los servicios a incluir con su configuración. Puertos y rutas asociadas entre host y contenedor, variables de entorno o volúmenes asociados a rutas del contenedor para garantizar persistencia de datos. Posteriormente un apartado volumes y un apartado network con la red o redes que utilizarán.

services:

nombre_servicio1:

image: imagen:versión

ports:

- puerto_host:puerto_contenedor

environment:

- VARIABLE_ENTORNO1=

- VARIABLE_ENTORNO2=

volumes:

- nombre_volumen_servicio1:/ruta

- /ruta/host:/ruta/contenedor

nombre_servicio2:

image: imagen:versión

environment:

volumes:

volumes:

nombre_volumen_servicio1:

nombre_volumen_servicio2:

networks:

nombre_red:

9. Servicios seleccionados

9.1. Base de Datos (MySQL - Imagen Oficial)

Servicio de base de datos MySQL que nos permite configurar una base de datos y usuario de acceso.

Puede consultarse la configuración y las plantillas en el apartado 11. Base de Datos (MySQL - Imagen Oficial).

Enlace a la imagen utilizada y su documentación: https://hub.docker.com/_/mysql

Probamos a levantar el servicio preconfigurado, comprobarlo y descargar la imagen:

```

appcauldron@appcauldron: /var/www/html/appcauldron.com/generatedfiles$ docker-compose -f testdatabase.yaml up
db_mysql Pulled
✓ 07bc8e18c4a Pull complete
✓ 1a9c166bf49 Pull complete
✓ 1e21ddafeecf Pull complete
✓ fbd1b56ac1 Pull complete
✓ 4e0ab39b8a5b Pull complete
✓ 165e33d37ca Pull complete
✓ zeitb
✓ dba
✓ ed9t
✓ e641
appcauldron@appcauldron: $ mysql -P 3310 -u root --password=aslr123 -e 'SHOW DATABASES; SELECT user,host FROM mysql.user;' 
+-----+-----+
| Database | 
+-----+-----+
| information_schema | 
| performance_schema | 
| sys | 
+-----+
1 row in set (0.00 sec)

+-----+-----+
| user | host |
+-----+-----+
| % | % |
| appcauldron | localhost |
| mysql.infoschema | localhost |
| mysql.session | localhost |
| mysql.sys | localhost |
| root | localhost |
+-----+
5 rows in set (0.00 sec)

::: portappcauldron@appcauldron: $ mysql -P 3310 -u appcauldron --password=aslr123 -e 'SHOW DATABASES;' 
+-----+-----+
| Database | 
+-----+-----+
| appcauldron-db_mysql | 
| information_schema | 
| performance_schema | 
+-----+
3 rows in set (0.00 sec)

```

Ilustración J

9.2. Servidor Web (Nginx - Imagen Oficial)

Servidor Web Nginx que nos permite levantar una página web en la ruta indicada de nuestro host.

Puede consultarse la configuración y las plantillas en el apartado 12. Servidor Web (Nginx - Imagen Oficial).

Enlace a la imagen utilizada y su documentación: https://hub.docker.com/_/nginx

Creamos una página de prueba en /var/www/html/nginx.

Probamos a levantar el servicio preconfigurado, comprobarlo y descargar la imagen:



Ilustración K

9.3. Servidor FTP (Pure-FTPd - Imagen Andrew Stilliard)

Servidor FTP Pure-FTPd que nos permite configurar una ruta de nuestro host como FTP.

Puede consultarse la configuración y las plantillas en el apartado 13. Servidor FTP (Pure-FTPd - Imagen Andrew Stilliard).

Enlace a la imagen utilizada y su documentación: <https://github.com/stilliard/docker-pure-ftpd>

Probamos a levantar el servicio preconfigurado, comprobarlo y descargar la imagen:

```
appcauldron@appcauldron: /var/www/html/appcauldron.com/generatedfiles$ docker-compose -f testftplib.yaml up
✓ Puled
  ✓ b4d181a07f88 Pull complete
  ✓ 6425163fae62 Pull complete
  ✓ 64008175ac6c Pull complete
  ✓ 592937979479 Pull complete
  ✓ 400c2d0aeca50 Pu [ ] appcauldron@appcauldron: $ ftp ftp://appcauldron:aslr123@localhost
  ✓ e6abdd7d7bf4f Pu [ ] appcauldron@appcauldron: $ Connected to localhost.
  ✓ 45dica929304 Pu [ ] 220----- Welcome to Pure-FTPd [privsep] [TLS] -----
  ✓ 9cd473bc0d69 Pu [ ] 220----- Welcome to Pure-FTPd [privsep] [TLS] -----
  ✓ fe93c726f579 Pu [ ] 220-User are user number 1 of 5 allowed.
  ✓ 792bc5f53975 Pu [ ] 220-Local time is now 21:21. Server port: 21.
  ✓ 220-This is a private system. No anonymous login
  ✓ 220-User accounts are passworded after 15 minutes of inactivity.
Attached to ftp-1
ftp-1: ~
```

Ilustración L

9.4. Monitorización (Netdata - Imagen Oficial)

Servicio de Monitorización de nuestro host por medio de paneles configurables.

Puede consultarse la configuración y las plantillas en el apartado 14. Monitorización (Netdata - Imagen Oficial).

Enlace a la imagen utilizada y su documentación: <https://hub.docker.com/r/netdata/netdata>

Probamos a levantar el servicio preconfigurado, comprobarlo y descargar la imagen:

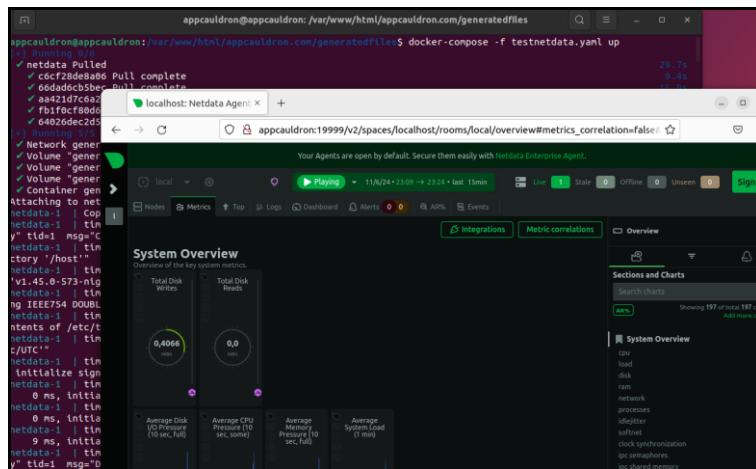


Ilustración M

9.5. Gestión de aprendizaje (Moodle - Imagen Bitnami)

Servicio que nos permite crear una plataforma de aprendizaje.

Puede consultarse la configuración y las plantillas en el apartado 15. Gestión de aprendizaje (Moodle - Imagen Bitnami).

Enlace a la imagen utilizada y su documentación:

<https://github.com/bitnami/containers/tree/main/bitnami/moodle>

Probamos a levantar el servicio preconfigurado, comprobarlo y descargar la imagen:

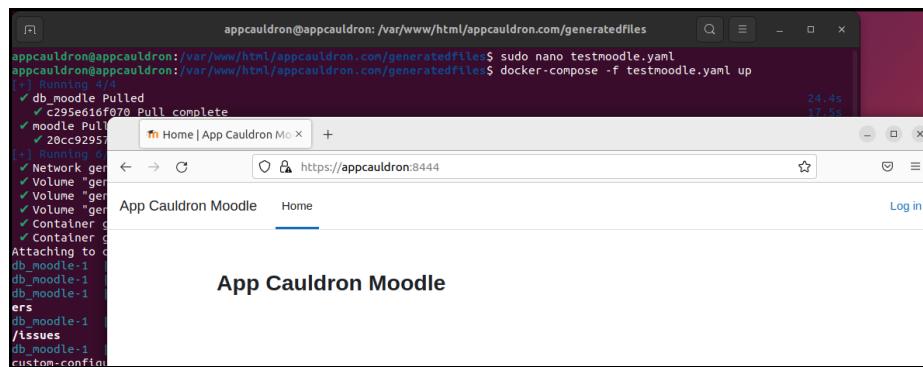


Ilustración N

9.6. Videoconferencia (Jitsi Meet - Imagen Oficial)

Servicio que nos permite levantar una plataforma de videoconferencia y crear salas de reuniones.

Puede consultarse la configuración y las plantillas en el apartado 16. Videoconferencia (Jitsi Meet - Imagen Oficial).

Enlace a la imagen utilizada y su documentación: <https://github.com/jitsi/docker-jitsi-meet>

Probamos a levantar el servicio preconfigurado, comprobarlo y descargar la imagen:

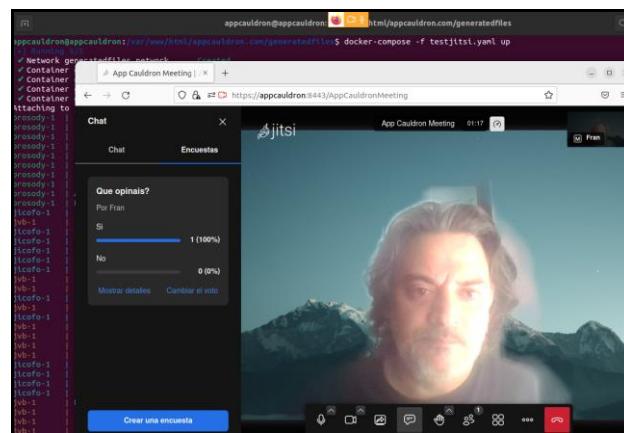


Ilustración O

9.7. Wordpress (Wordpress - Imagen Oficial)

Servicio que nos permite levantar y empezar a crear un Wordpress.

Puede consultarse la configuración y las plantillas en el apartado 17. Wordpress (Wordpress - Imagen Oficial).

Enlace a la imagen utilizada y su documentación: https://hub.docker.com/_/wordpress

Probamos a levantar el servicio preconfigurado, comprobarlo y descargar la imagen:

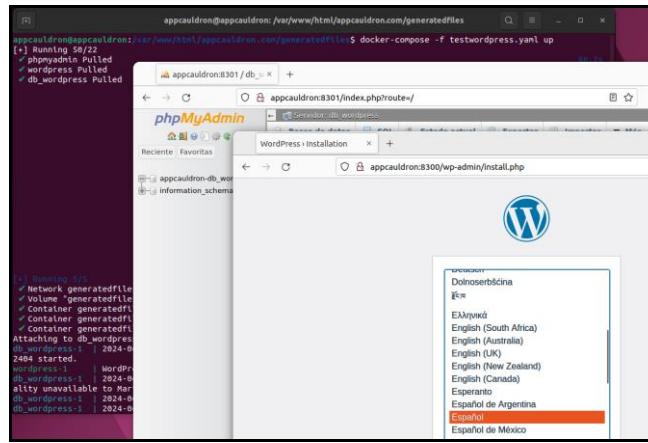


Ilustración P

9.8. Office (WPS Office - Imagen linuxserver.io)

Servicio que nos permite levantar un explorador con capacidad de crear, editar y consultar archivos Office.

Puede consultarse la configuración y las plantillas en el apartado 18. Office (WPS Office - Imagen linuxserver.io).

Enlace a la imagen utilizada y su documentación:

<https://hub.docker.com/r/linuxserver/wps-office>

Probamos a levantar el servicio preconfigurado, comprobarlo y descargar la imagen:

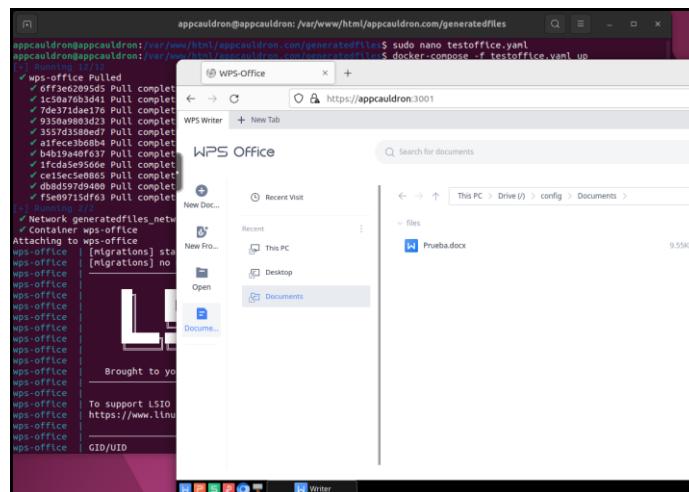


Ilustración Q

APP CAULDRON – FORMULARIO Y SITIO WEB

10. Sitio web /var/www/html/appcauldron.com

Para crear el código html base del sitio hemos utilizado como referencia el borrador de diseño disponible en:

https://www.w3schools.com/howto/howto_make_a_website.asp

Hemos generado una cabecera (Header) para incluir el logo, debajo una barra de navegación (navigation bar), y 2 paneles, uno lateral (side) y otro principal (main), sin pie de página (footer) como se puede ver en la Ilustración R.

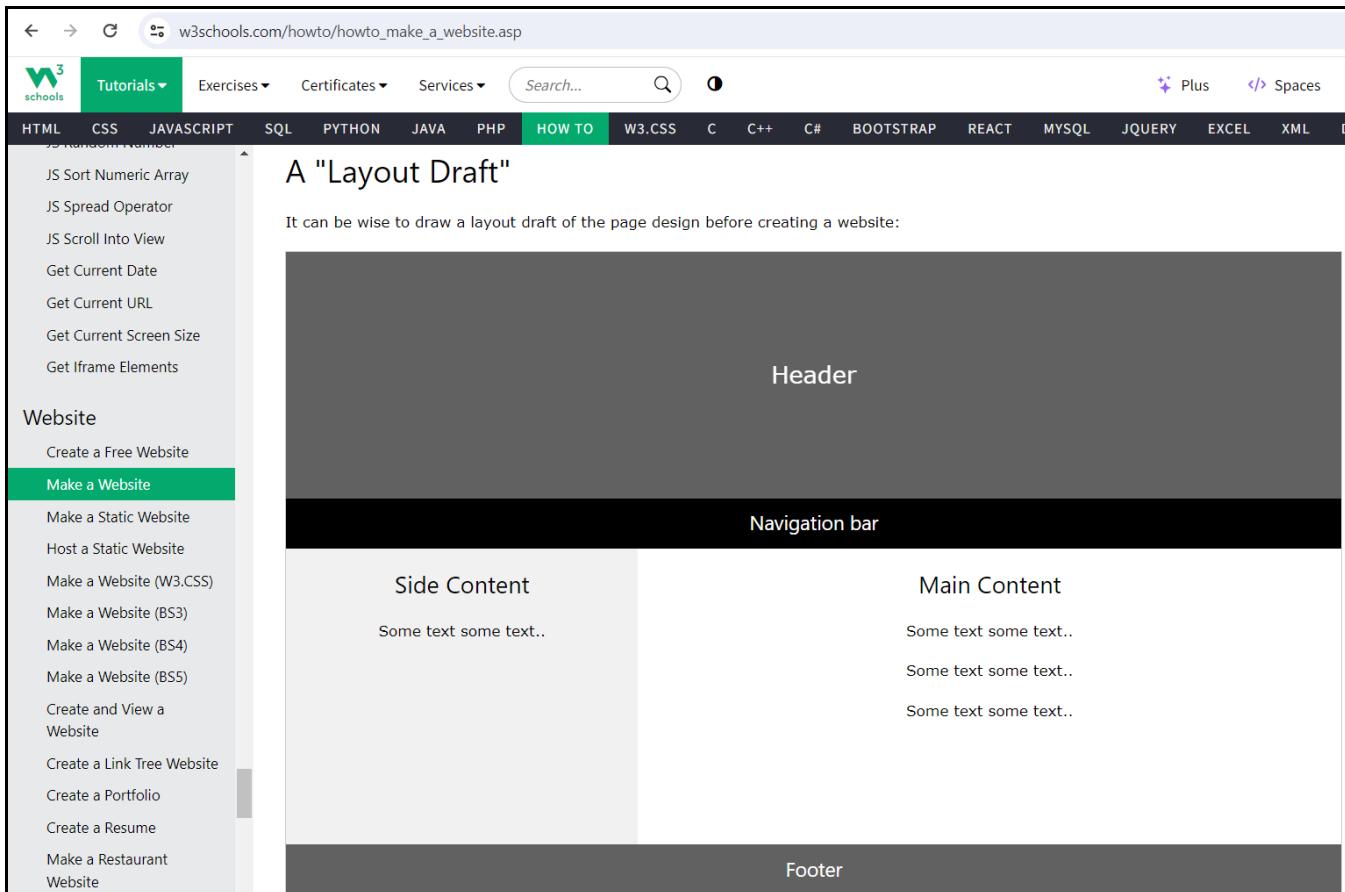


Ilustración R

Para terminar de darle forma, se ha creado una hoja de estilos appcauldron-style.css en la ruta /var/www/html/appcauldron.com/css/

Se puede consultar en el ANEXO IX - CSS - /var/www/html/appcauldron.com/css/

10.1. index.html

Página de inicio, menú “Home” del sitio web. Se da la bienvenida al sitio y se explica brevemente el funcionamiento principal de los menús y la web.

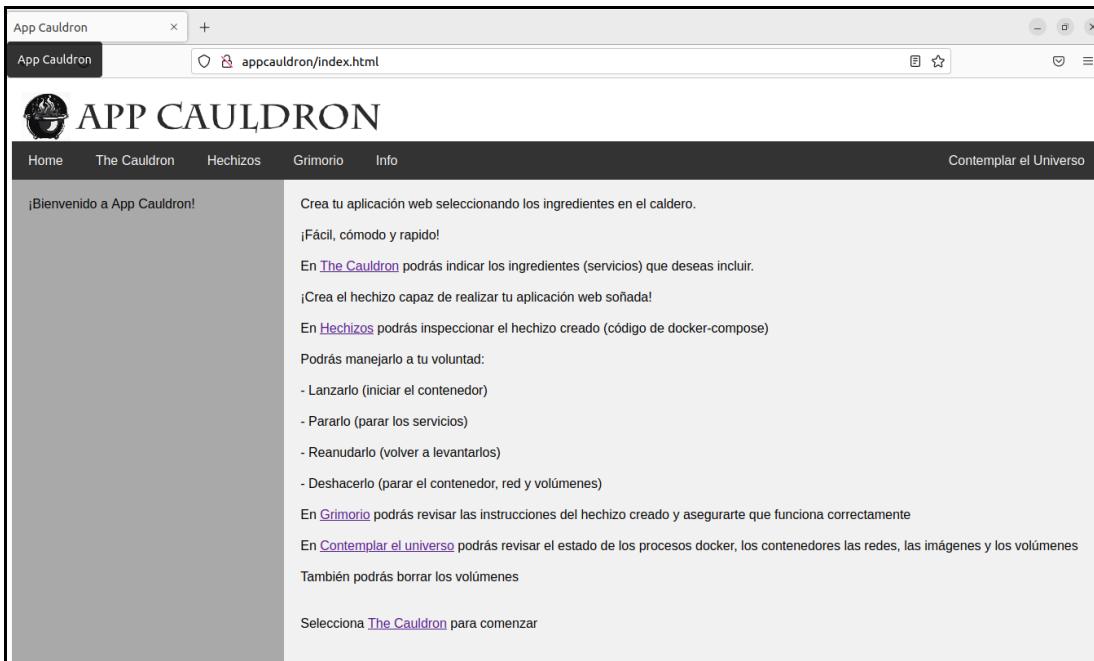


Ilustración S

Se puede consultar el código en el apartado 10.8. index.html.

10.2. cauldron.html

El menú “The Cauldron” aloja el formulario principal tipo checkbox para selección de servicios. A través de POST enviamos las palabras correspondientes a la selección a cauldron2.php.

Creamos una función validateForm donde se comprueban los checkbox que existen y si están marcados. Como se puede comprobar en la Ilustración T, si no hay ninguno salta una alerta indicando que se debe seleccionar como mínimo una opción. Posteriormente se crea un evento para llamar a la función si se envía el formulario con el botón “Arrojar al caldero”.

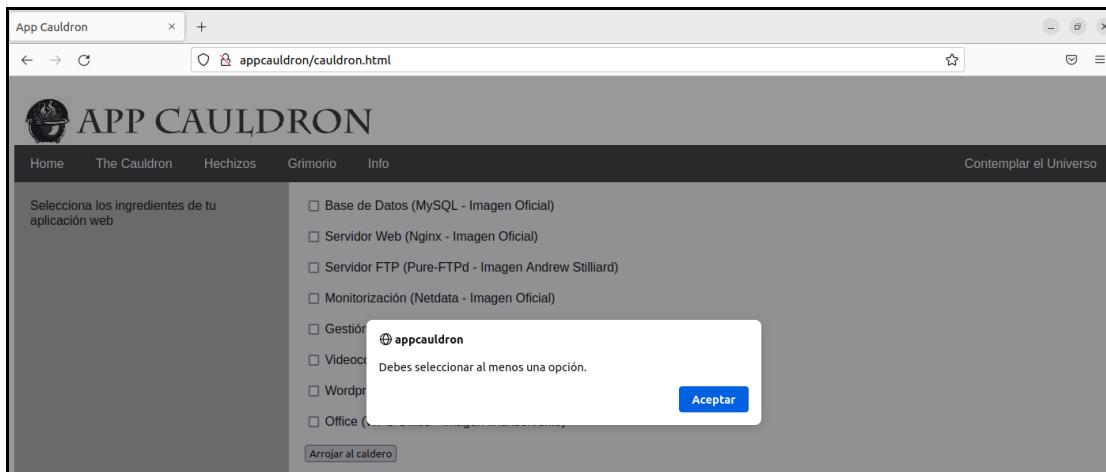


Ilustración T

Se puede consultar el código en el apartado 10.9. cauldron.html.

10.3. cauldron2.php

Se crea un script para mostrar una frase aleatoria de calderos en la parte izquierda.

Se verifica que haya llegado el submit del formulario de la página cauldron.html por método POST, y si es el caso, se verifica si ha llegado por método POST cada palabra asignada a cada casilla. Si es el caso se asigna a una variable con el mismo nombre.

Se crean 2 ficheros en la ruta generatedfiles/ con nombre ingredientes.txt y servicios.txt con la función fopen de PHP. Se comprueba si cada variable existe y no está vacía y si es el caso, con la función fwrite de PHP se añade a cada fichero. Después utilizamos fclose de PHP.

Ejecutamos el script de bash magic_cauldron.sh con la función shell_exec.

En esta función comenzamos escribiendo el archivo spell.yaml con la palabra services:

Posteriormente recorremos el fichero servicios.txt generado anteriormente y si contiene las palabras correspondientes a cada servicio, añade su <plantilla> de la ruta templates/ con la configuración del servicio. También añadimos a un nuevo fichero grimoire.txt la <plantilla>.info con la información del servicio que podremos consultar en el menú Grimorio de la web. Si cualquier servicio utiliza volúmenes marcamos la variable apartadovolumes a 1.

Con esto hemos terminado el apartado services: del archivo spell.yaml. Si algún servicio marca apartadovolumes a 1, añadimos a spell.yaml la palabra volumes: y así comenzar con esta sección del fichero.

Volvemos a recorrer el fichero servicios.txt y si hay algún servicio con volúmenes añade su <plantilla>.volumes. Con esto terminamos con el apartado volumes: del archivo spell.yaml

Añadimos el apartado correspondiente a la red y borramos los ficheros temporales utilizados:

networks:

network:

Finalmente, de vuelta en cauldron2.php comprobamos si existe spell.yaml y lo mostramos, si no mostramos un mensaje de error.

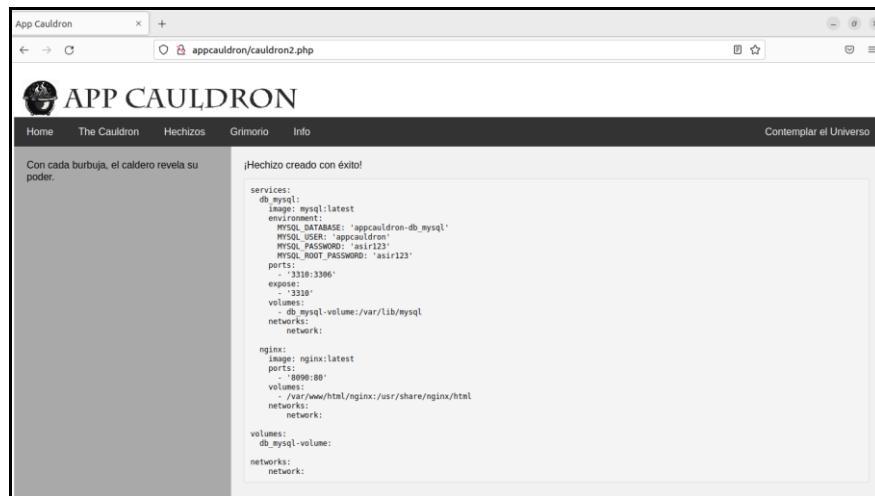


Ilustración U

Se puede consultar el código en el apartado 10.10. cauldron2.php.

10.4. spells.php

En el menú “Hechizos”, si no existe el fichero spell.yaml mostramos un enlace al caldero para generarlo.

Si existe, lo mostramos y además creamos 5 botones que mostramos en la parte izquierda.

4 botones envían un formulario POST a la misma página para que se refresque.

Estos botones ejecutan un script con el comando PHP como usuario appcauldron:

```
shell_exec('sudo -u appcauldron scripts/<script>.sh')
```

Lanzan el comando docker-compose -f spell.yaml con las opciones:

- up (“Crear hechizo” - inicia los servicios que hemos configurado en spell.yaml).
- stop (“Parar hechizo” – para los servicios, pero no elimina los contenedores).
- start (“Continuar hechizo” – relanza los contenedores parados con stop).
- down (“Deshacer hechizo” – para los contenedores en ejecución y los elimina).

Con down también se eliminan las redes. Si hubiéramos utilizado la opción -v en este comando, eliminaríamos también los volúmenes asociados.

Como los volúmenes se pueden asignar a otros servicios / contenedores y el objetivo es poder añadir funcionalidades a AppCauldron, se ha preferido mantenerlos por si algún usuario prefiere utilizar esta opción.

Para eliminarlos, se ha dado la opción de gestionar los volúmenes en el menú “Contemplar el universo” (commands.php) a través de botones.

Creamos un último botón (“Eliminar hechizo”) que utiliza la función unlink de PHP para eliminar el fichero spell.yaml.

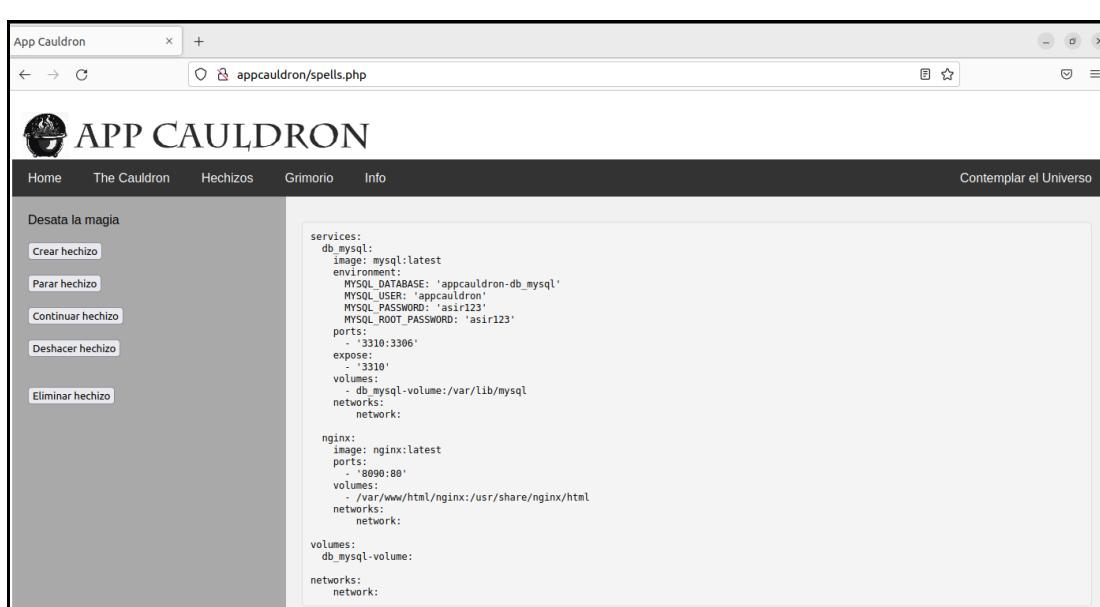


Ilustración V

Se puede consultar el código en el apartado 10.11. spells.php.

10.5. grimoire.php

En el menú “Grimorio”, si no existe el fichero generatedfiles/grimoire.txt, en la parte izquierda mostramos el texto “No hay entradas para tu hechizo” y en la parte derecha mostramos un enlace al caldero.

Si existe, en la parte izquierda mostramos el texto “Sigue bien las instrucciones” y en la parte derecha mostramos el contenido generado en el apartado 10.3. cauldron2.php y el script magic_cauldron.sh, es decir, el contenido de templates/<servicio>.info de cada servicio seleccionado.

El texto lo mostramos con la función htmlspecialchars y un estilo configurado en la etiqueta <pre> de nuestra hoja de estilos. Como con este formato no vamos a poder mostrar hipervínculos y nos interesa para facilitar las comprobaciones al usuario, se revisa si algo comienza con [enlace:httpXXX] o [enlace:httpsXXX], y en ese caso escapamos y creamos un hipervínculo en una pestaña nueva.

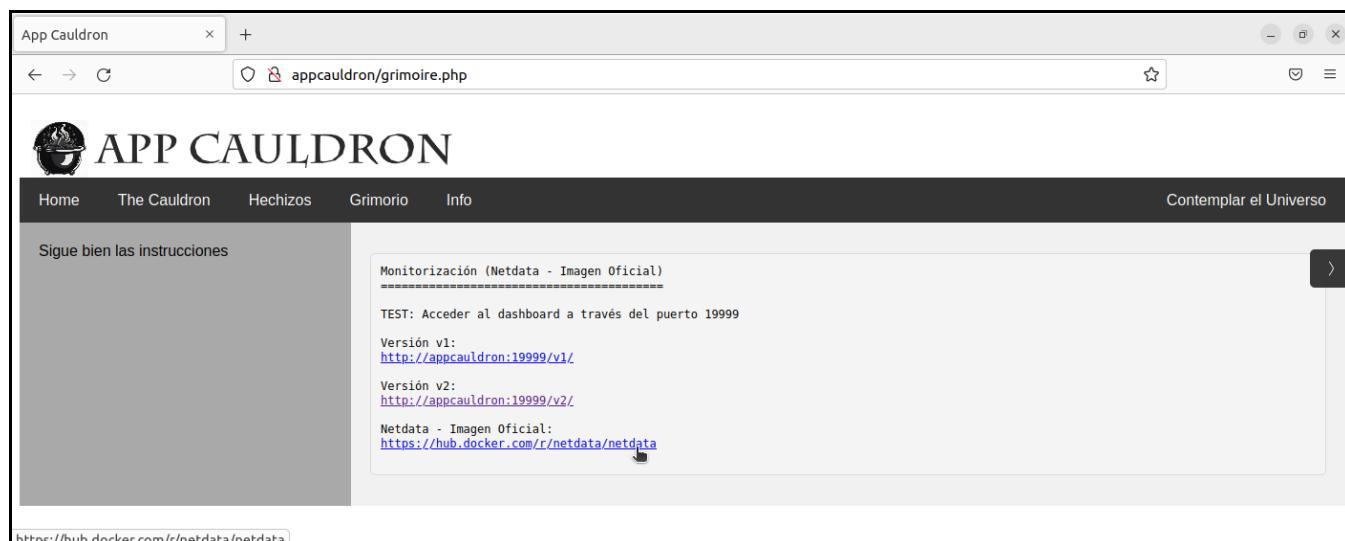


Ilustración W

Se puede consultar el código en el apartado 10.12. grimoire.php.

10.6. commands.php

Como hemos configurado en el apartado del ANEXO II – INSTALACIÓN DE APACHE Y VIRTUAL HOST, el usuario www-data de apache tiene permisos para ejecutar scripts de la carpeta /var/www/html/appcauldron.com/scripts/

En el menú “Contemplar el Universo” se puede revisar la salida de distintos comandos útiles de Docker que se ejecutan al entrar en la página a través de scripts.

En la parte izquierda se indica el comando lanzado y en la derecha la salida correspondiente.

Pulsando el botón de refresco o Contemplando el Universo nuevamente, se puede ir revisando la salida.

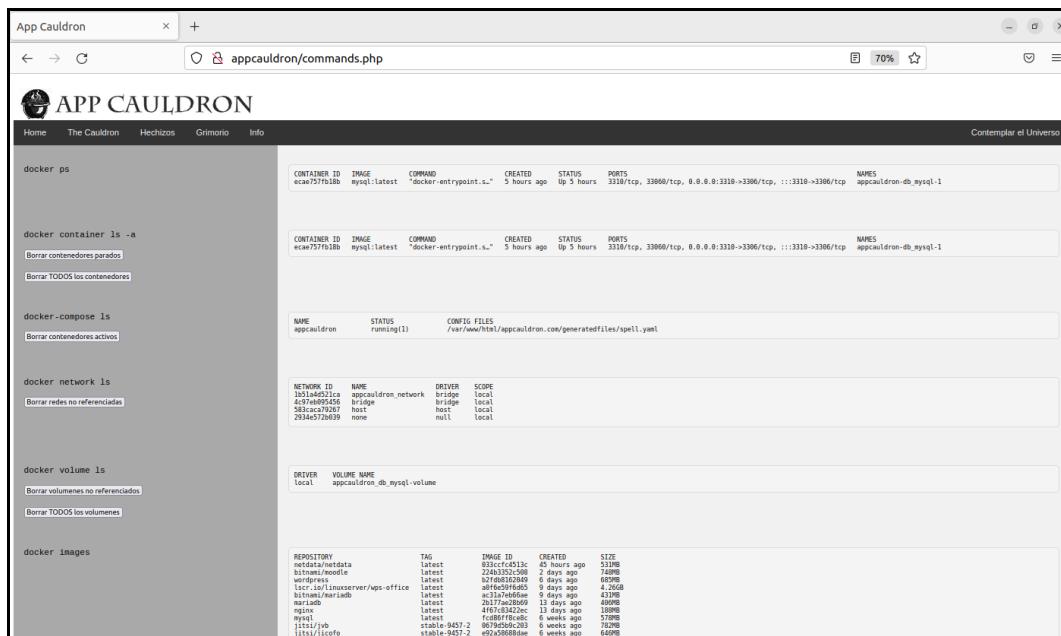
- docker ps: Podemos revisar los contenedores en ejecución.
- docker container ls -a: Igual que docker ps. Con -a podemos ver todos los contenedores.
- docker container ls: Lista los proyectos docker-compose en ejecución, con los contenedores o servicios que contiene y el archivo de configuración utilizado.
- docker network ls: Lista todas las redes docker.
- docker volume ls: Muestra todos los volúmenes que conoce Docker.
- docker images: Muestra todas las imágenes Docker descargadas y presentes en nuestro sistema.

Adicionalmente se han habilitado 6 botones:

- “Borrar contenedores parados”.
- “Borrar TODOS los contenedores”.
- “Borrar contenedores activos”.
- “Borrar redes no referenciadas”.
- “Borrar volúmenes no referenciados”: Volúmenes a los que no apunta ningún contenedor. Tienden a ser anónimos (es decir, no tienen un nombre a diferencia de los que creamos nosotros. Los utiliza Docker para almacenar información si lo necesita).
- “Borrar todos los volúmenes”.

Se puede consultar el código de los botones en:

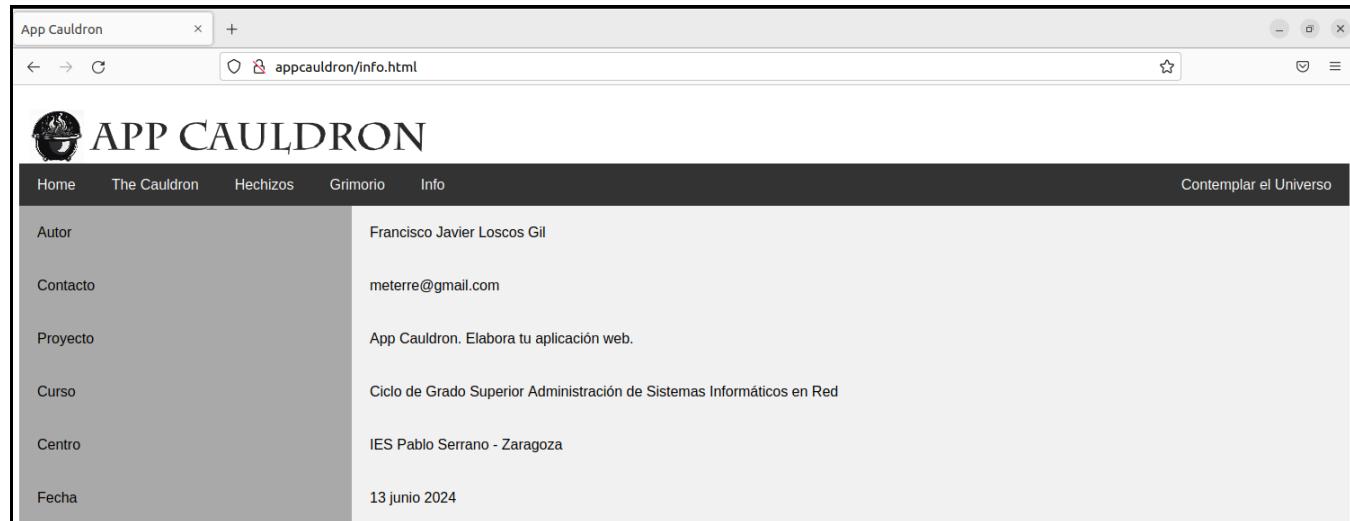
22. Botones /var/www/html/appcauldron.com/commands.php.

**Ilustración X**

Se puede consultar el código en el apartado 10.13 commands.php.

10.7. info.html

Se muestra información del autor, correo de contacto, nombre de este proyecto, nombre del grado, centro en el que se estudia y la fecha de entrega del proyecto.

**Ilustración Y**

Se puede consultar el código en el apartado 10.14. info.html.

CONCLUSIONES

Trabajar en este proyecto ha sido muy interesante y me ha permitido aprender mucho sobre Docker y el funcionamiento de los contenedores en general.

A medida que he ido adquiriendo conocimientos he podido pasar del planteamiento inicial, crear un formulario web que genere un archivo y con él levantar un par de servicios como sea utilizando algún comando de Docker, a plantearme crear casi una plataforma que se puede utilizar tanto para montar tu propia aplicación web como reza el título del proyecto, como una plataforma de aprendizaje. El concepto de ‘caldero’ me ha ayudado mucho a abordarlo y darle una forma y un sentido más profesional.

Aunque quizás no sea la forma más adecuada, incluir el menú “Contemplar el Universo” para poder estar revisando los procesos y lo que iba ocurriendo “en tiempo real” ha sido una mejora y algo necesario personalmente para ir comprendiendo y realizando pruebas de lo que sucedía. A medida que iba encontrando un comando útil para ver el estado de las cosas lo he intentado integrar. Quizás he echado de menos una manera de consultar el log de los contenedores sin tener que ir al terminal y es algo que me gustaría barajar incluir de cara a futuro.

Otras opciones que me gustaría haber estudiado o integrado, aparte de darle una mejora visual ya que en un principio se ha buscado la claridad y funcionalidad, serían:

- Generar una imagen de Docker con Dockerfile y subirlo a Docker Hub o a GitHub.
- Trabajar en una documentación o sistema para que cualquiera añada aplicaciones al formulario de una manera fácil.
- Poder generar varios hechizos o ficheros de configuración.
- “Hechizos avanzados”, poder elegir varios aspectos en el formulario de creación, como el nombre de proyecto, nombres de usuarios a crear o contraseñas, selección de puertos, etc.
- Crear un apartado de enlaces a tutoriales y documentación.

En las dificultades encontradas, aparte de problemas puntuales de código, quizás los mayores han sido comprender el funcionamiento o conceptos de utilizar volúmenes frente a asignar rutas de host al contenedor, o algunas asignaciones de puertos.

Revisar imágenes y configuraciones de servicios a incluir tenía su complejidad, porque en algunas me faltaba documentación, o se daba por hecho que manejas bien el servicio o aplicación X hasta el punto de saber configurar todo correctamente. He tenido que descartar varias después de invertir muchísimas horas intentando aclararme.

He intentado buscar servicios que funcionaran bien, tuvieran utilidad y algún elemento diferenciador para que su configuración no fuera siempre igual. Ya fuera tener que levantar servicios y relacionarlos, como una base de datos y una aplicación, poner dependencias entre servicios o tener que instalar algo en local y utilizar un archivo de variables .env.

El proyecto me ha servido para aclararme y aprender muchísimo a medida que avanzaba.

Estoy muy contento del trabajo realizado y de haber participado en el mismo.

BIBLIOGRAFÍA

- Tutoriales Docker en Youtube:

TechWorld with Nana. (2020, 21 de octubre). *Docker Tutorial for Beginners [FULL COURSE in 3 Hours]* [Video]. Youtube. <https://youtu.be/3c-iBn73dDE?si=7mlzf58zXKQ8fR3i>

TechWorld with Nana. (2023, 15 de febrero). *Docker Crash Course for Absolute Beginners [NEW]* [Video]. Youtube. <https://youtu.be/pg19Z8LL06w?si=exjhUi19oUckTUp>

TechWorld with Nana. (2024, 11 de enero). *Ultimate Docker Compose Tutorial* [Video]. Youtube. <https://youtu.be/SXwC9fSwct8?si=blyvQXOCCj9fn6EG>

- Instalaciones:

Leonardo, Juan. (2024, 9 de enero). Cómo instalar Apache en Ubuntu 22.04. *EXTASSIS NETwork Tutoriales*. <https://extassisnetwork.com/tutoriales/como-instalar-apache-en-ubuntu/>

Klein, Tomer. (2023, 22 de octubre). Step-by-Step Tutorial: Installing Docker and Docker Compose on Ubuntu. *Medium*. <https://medium.com/@tomer.klein/step-by-step-tutorial-installing-docker-and-docker-compose-on-ubuntu-a98a1b7aaed0>

Boucheron, Brian, Drake, Mark, Tran, Tony (2022, 8 de junio 2022). How To Install MariaDB on Ubuntu 22.04. *DigitalOcean*. <https://www.digitalocean.com/community/tutorials/how-to-install-mariadb-on-ubuntu-22-04>

Ubuntu. (2024, 31 de julio). *Ubuntu Server - for scale out with Ubuntu Server*. <https://ubuntu.com/server/docs/how-to-install-and-configure-php>

- HTML:

W3schools. (s.f.). *How to Host a Static Website*. https://www.w3schools.com/howto/howto_make_a_website.asp

- Generación de imágenes por IA:

Logo de AppCauldron creado con Leonardo.Ai (<https://app.leonardo.ai/>)

- Dudas puntuales sobre código:

OpenAI. (2023). ChatGPT (versión GPT-4) [Modelo de lenguaje grande]. <https://www.openai.com/>

- Imágenes Docker y documentación:

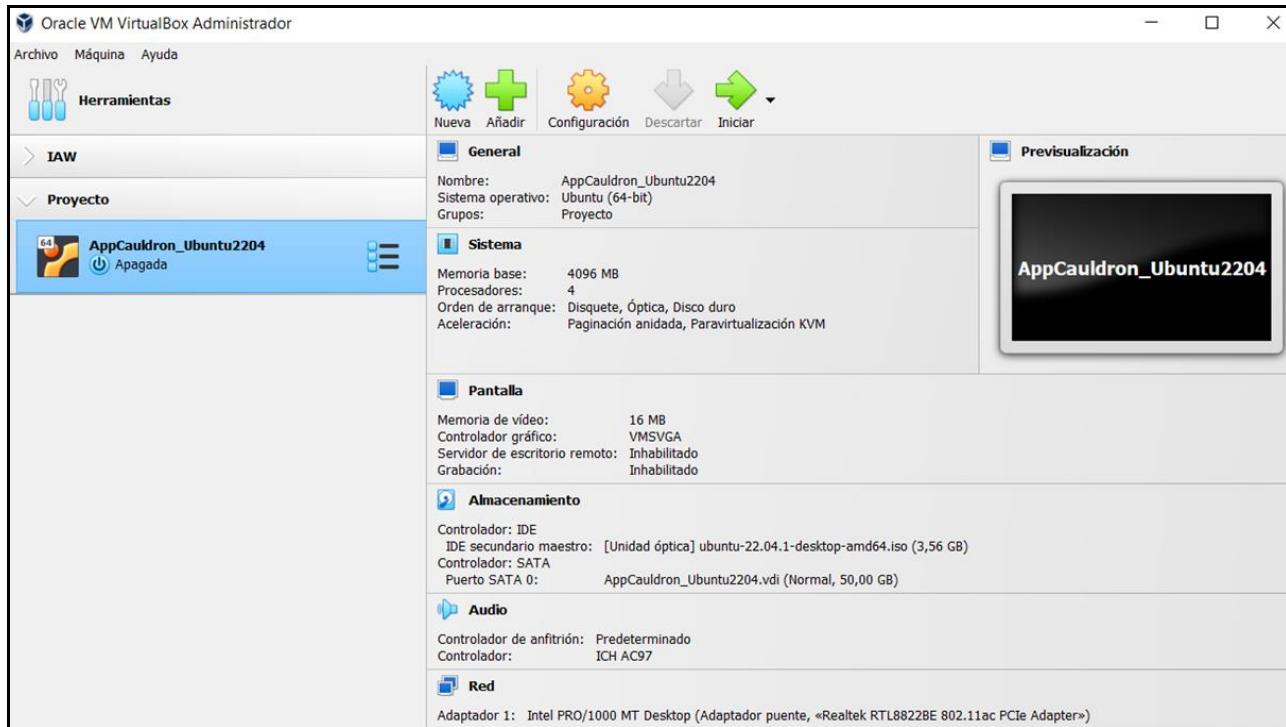
Revisión de imágenes Docker y documentación en Docker Hub (<https://hub.docker.com/>)

Revisión de imágenes Docker y documentación en GitHub (<https://github.com/>)

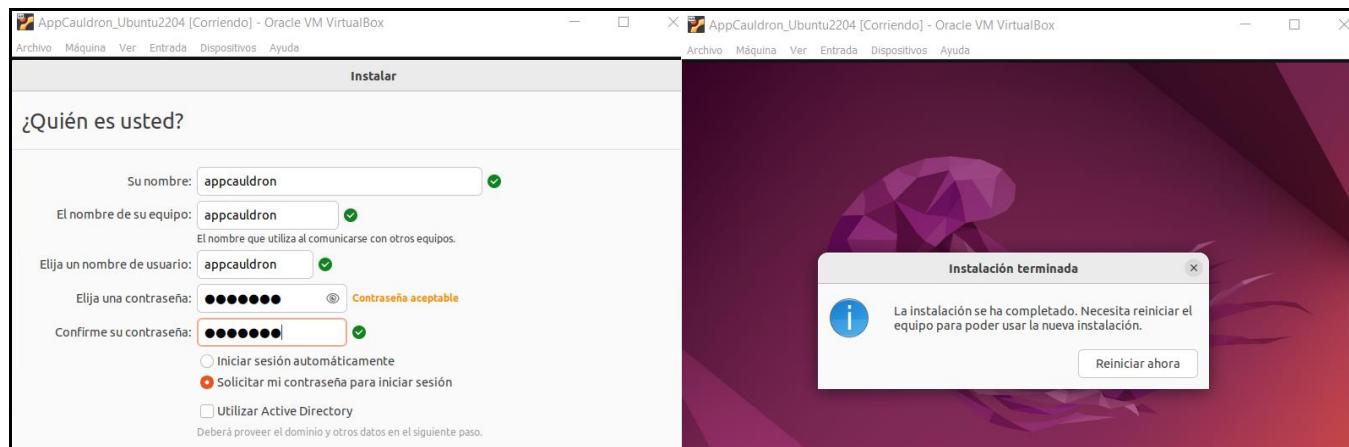
ANEXO I – INSTALACIÓN MAQUINA VIRTUAL UBUNTU

En Oracle VM VirtualBox creamos la máquina virtual AppCauldron_Ubuntu2204 utilizando una imagen Ubuntu 22.04.1-desktop.

Añadimos un disco de 50,00GB y Adaptador puente para que utilice la red de casa:



Siguiendo el proceso de instalación y tras elegir el idioma, seleccionamos el nombre de usuario y equipo como appcauldron. Finalizamos el proceso:



ANEXO II – INSTALACIÓN DE APACHE Y VIRTUAL HOST

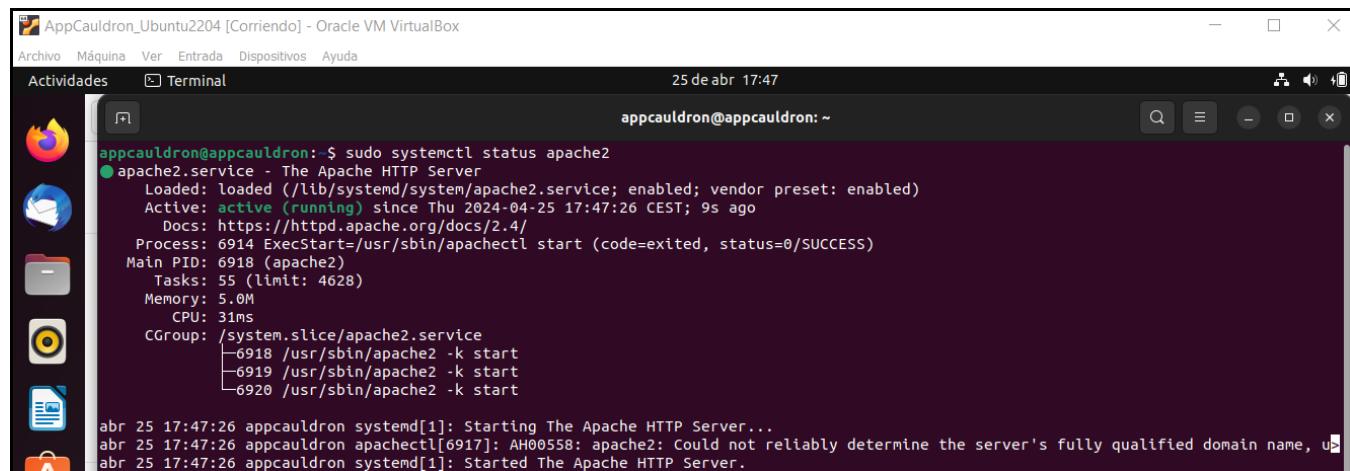
Actualizamos el sistema, instalamos apache y comprobamos su estado.

Ejecutamos los comandos:

```
sudo apt update
```

```
sudo apt install apache2
```

```
sudo systemctl status apache2
```



```
appcauldron@appcauldron:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2024-04-25 17:47:26 CEST; 9s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 6914 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
 Main PID: 6918 (apache2)
    Tasks: 55 (limit: 4628)
   Memory: 5.0M
      CPU: 31ms
     CGroup: /system.slice/apache2.service
             ├─6918 /usr/sbin/apache2 -k start
             ├─6919 /usr/sbin/apache2 -k start
             ├─6920 /usr/sbin/apache2 -k start

abr 25 17:47:26 appcauldron systemd[1]: Starting The Apache HTTP Server...
abr 25 17:47:26 appcauldron apachectl[6917]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.0.1 for port 80
abr 25 17:47:26 appcauldron systemd[1]: Started The Apache HTTP Server.
```

Para corregir el mensaje AH00558, editamos el archivo apache2.conf y añadimos al final ServerName 127.0.0.1.

```
sudo nano /etc/apache2/apache2.conf
```

```
# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
ServerName 127.0.0.1
```

Comprobamos que la configuración es correcta y reiniciamos el servicio apache:

```
sudo apachectl configtest
```

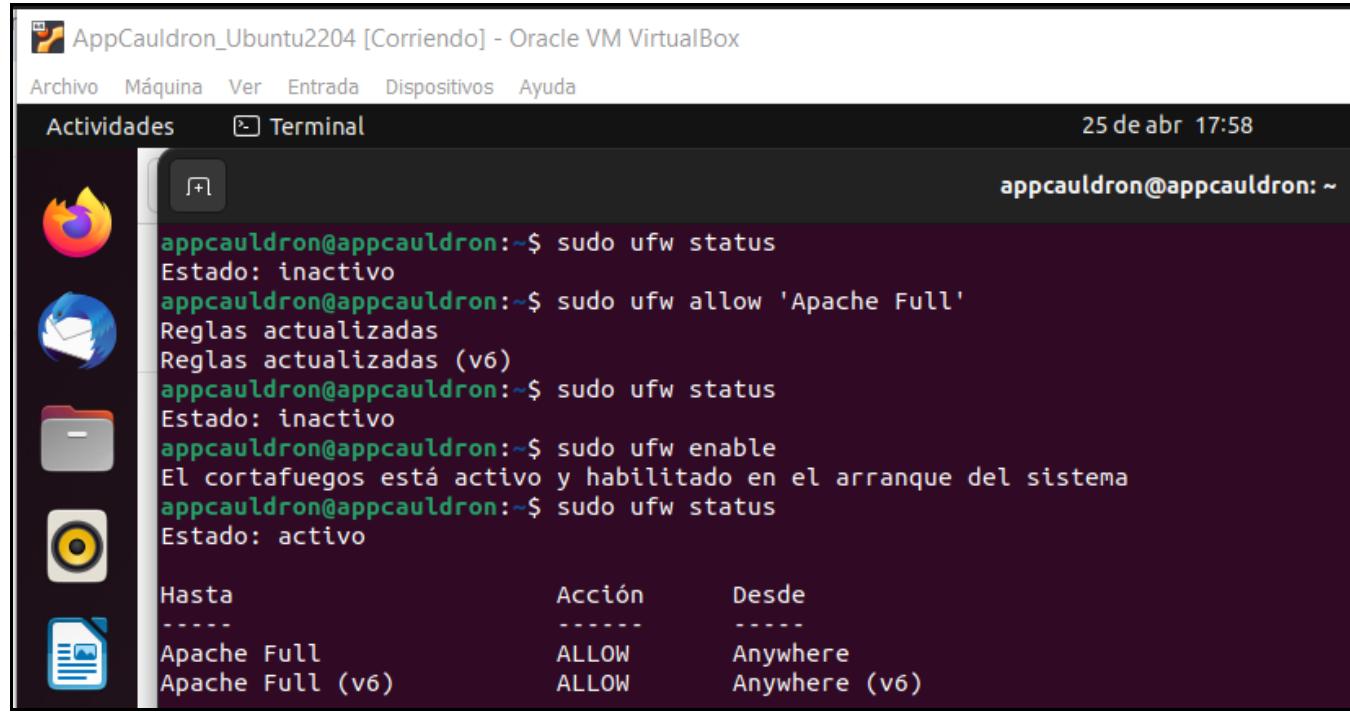
```
sudo systemctl reload apache2.service
```

Activamos el firewall ufw y lo configuramos para abrir los puertos 80 y 443, habilitando el perfil 'Apache Full', que incluye reglas para ambos puertos:

```
sudo ufw allow 'Apache Full'
```

```
sudo ufw enable
```

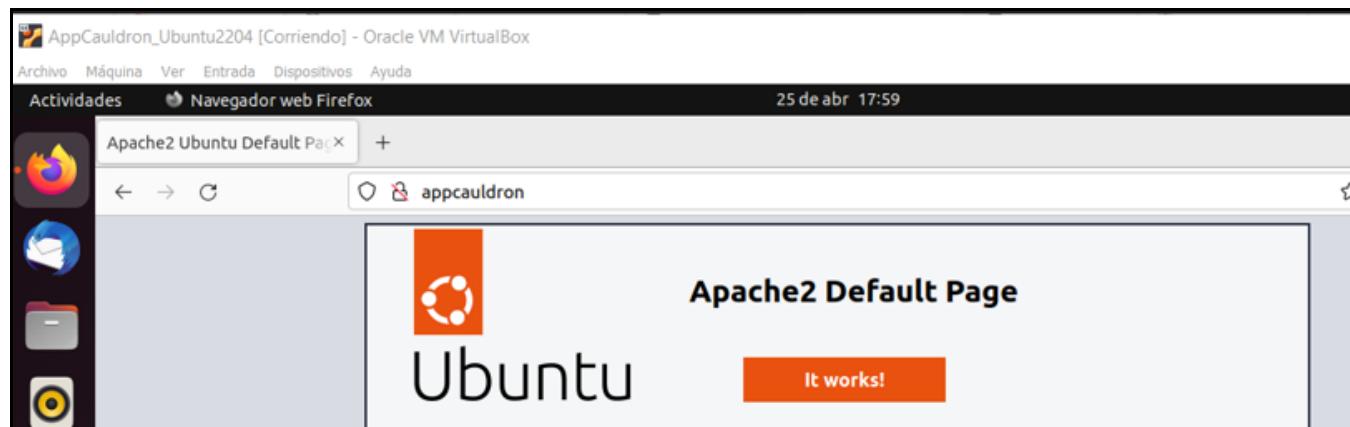
```
sudo ufw status
```



```
appcauldron@appcauldron:~$ sudo ufw status
Estado: inactivo
appcauldron@appcauldron:~$ sudo ufw allow 'Apache Full'
Reglas actualizadas
Reglas actualizadas (v6)
appcauldron@appcauldron:~$ sudo ufw status
Estado: inactivo
appcauldron@appcauldron:~$ sudo ufw enable
El cortafuegos está activo y habilitado en el arranque del sistema
appcauldron@appcauldron:~$ sudo ufw status
Estado: activo

Hasta          Acción      Desde
-----          -----      -----
Apache Full    ALLOW      Anywhere
Apache Full (v6) ALLOW      Anywhere (v6)
```

Comprobamos que funciona la instalación de apache accediendo a <http://appcauldron>:

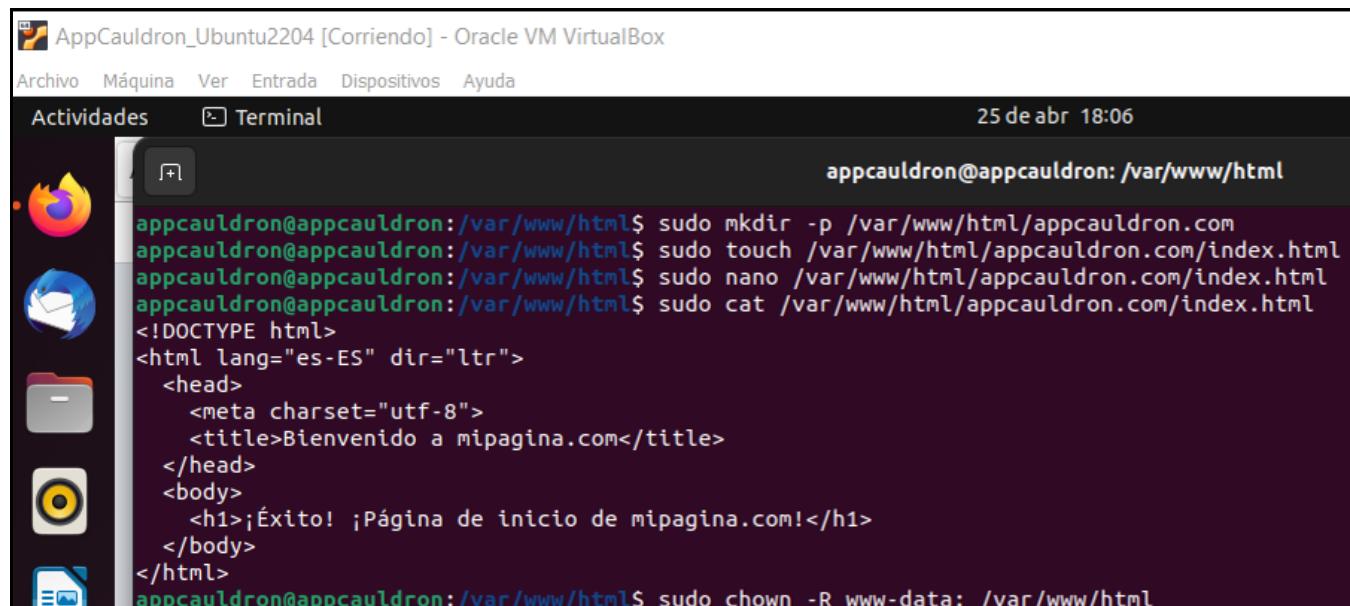


Configuramos el servidor virtual. Creamos el directorio /var/www/html/appcauldron.com para contenerlo. Posteriormente creamos una página básica para comprobar su funcionamiento. Cambiamos el propietario del directorio /var/www/html al usuario usuario de Apache (www-data).

```
sudo mkdir -p /var/www/html/appcauldron.com
sudo touch /var/www/html/appcauldron.com/index.html
sudo nano /var/www/html/appcauldron.com/index.html
<!DOCTYPE html>
<html lang="es-ES" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Bienvenido a mipagina.com</title>
  </head>
  <body>
    <h1>¡Éxito! ¡Página de inicio de mipagina.com!</h1>
  </body>
</html>
```



```
sudo cat /var/www/html/appcauldron.com/index.html
sudo chown -R www-data: /var/www/html
```



Creamos la configuración de nuestro servidor virtual. Activamos la configuración, comprobamos que es correcta y reiniciamos el servicio apache.

```
sudo nano /etc/apache2/sites-disponibles/appcauldron.com.conf
```

```
<VirtualHost *:80>
```

```
    ServerAdmin appcauldron@appcauldron.com
```

```
    DocumentRoot /var/www/html/appcauldron.com/
```

```
    ServerName appcauldron
```

```
    ServerAlias www.appcauldron
```

```
<Directory /var/www/html/appcauldron.com/>
```

```
    Options Indexes FollowSymLinks MultiViews
```

```
    AllowOverride All
```

```
    Order allow,deny
```

```
    allow from all
```

```
</Directory>
```

```
ErrorLog ${APACHE_LOG_DIR}/appcauldron.com-error.log
```

```
CustomLog ${APACHE_LOG_DIR}/appcauldron.com-access.log combined
```

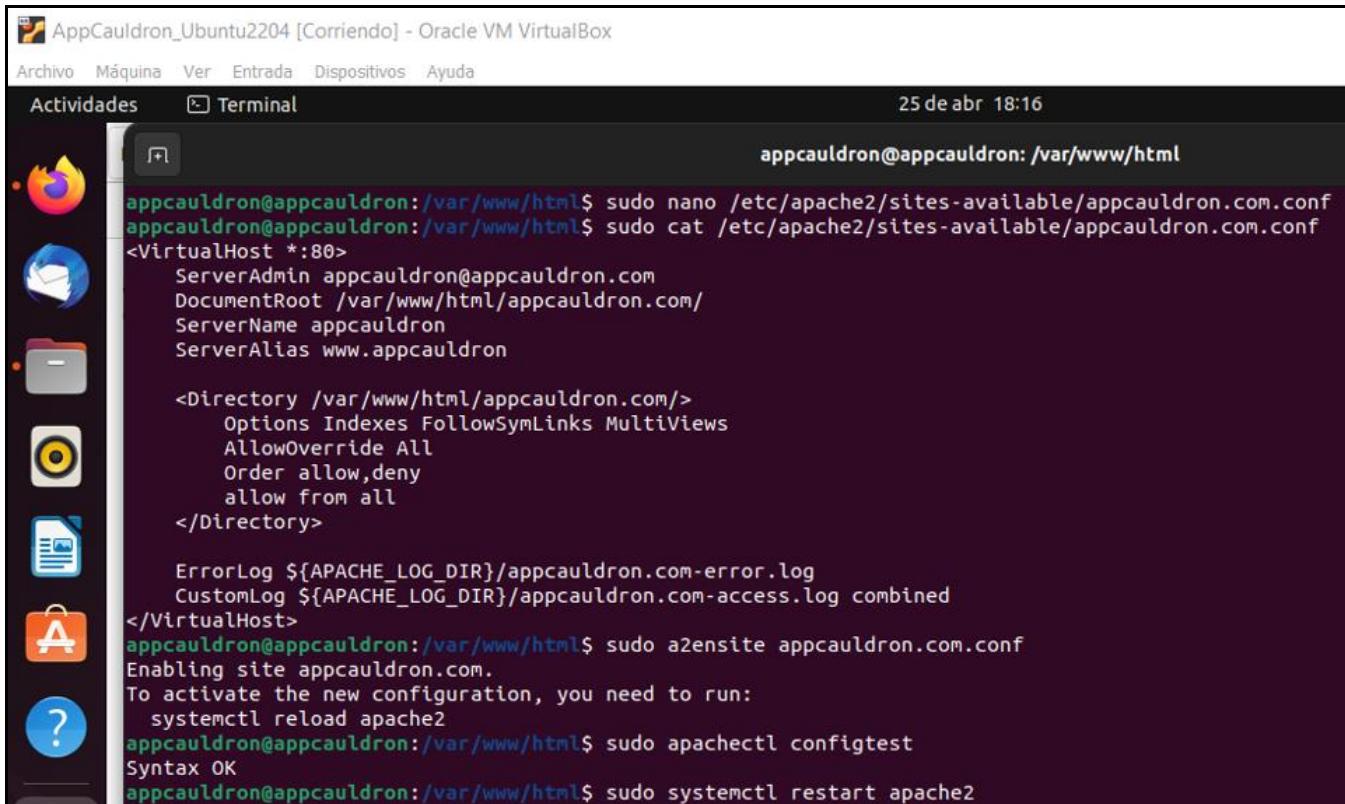
```
</VirtualHost>
```

```
sudo a2ensite appcauldron.com
```

```
sudo apachectl configtest
```

```
Syntax OK
```

```
sudo systemctl restart apache2
```



AppCauldron_Ubuntu2204 [Corriendo] - Oracle VM VirtualBox

Archivo Máquina Ver Entrada Dispositivos Ayuda

Actividades Terminal 25 de abr 18:16

```
appcauldron@appcauldron:/var/www/html$ sudo nano /etc/apache2/sites-available/appcauldron.com.conf
appcauldron@appcauldron:/var/www/html$ sudo cat /etc/apache2/sites-available/appcauldron.com.conf
<VirtualHost *:80>
    ServerAdmin appcauldron@appcauldron.com
    DocumentRoot /var/www/html/appcauldron.com/
    ServerName appcauldron
    ServerAlias www.appcauldron

    <Directory /var/www/html/appcauldron.com/>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride All
        Order allow,deny
        allow from all
    </Directory>

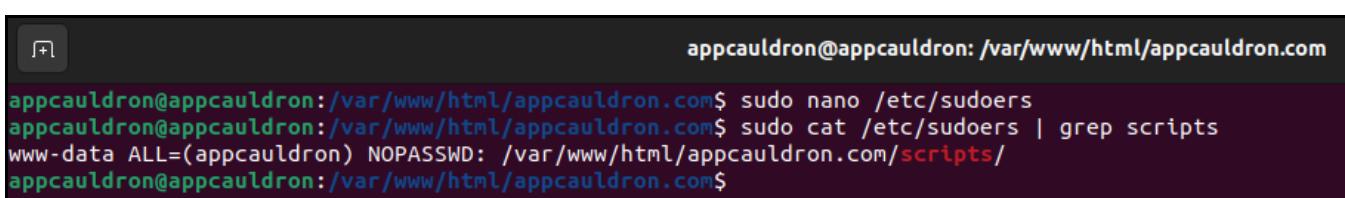
    ErrorLog ${APACHE_LOG_DIR}/appcauldron.com-error.log
    CustomLog ${APACHE_LOG_DIR}/appcauldron.com-access.log combined
</VirtualHost>
appcauldron@appcauldron:/var/www/html$ sudo a2ensite appcauldron.com.conf
Enabling site appcauldron.com.
To activate the new configuration, you need to run:
    systemctl reload apache2
appcauldron@appcauldron:/var/www/html$ sudo apachectl configtest
Syntax OK
appcauldron@appcauldron:/var/www/html$ sudo systemctl restart apache2
```

Comprobamos el correcto funcionamiento accediendo a <http://appcauldron>:



Damos permiso al usuario apache www-data para ejecutar scripts de la carpeta /var/www/html/appcauldron.com/scripts como usuario appcauldron modificando /etc/sudoers:

www-data ALL=(appcauldron) NOPASSWD: /var/www/html/appcauldron.com/scripts/

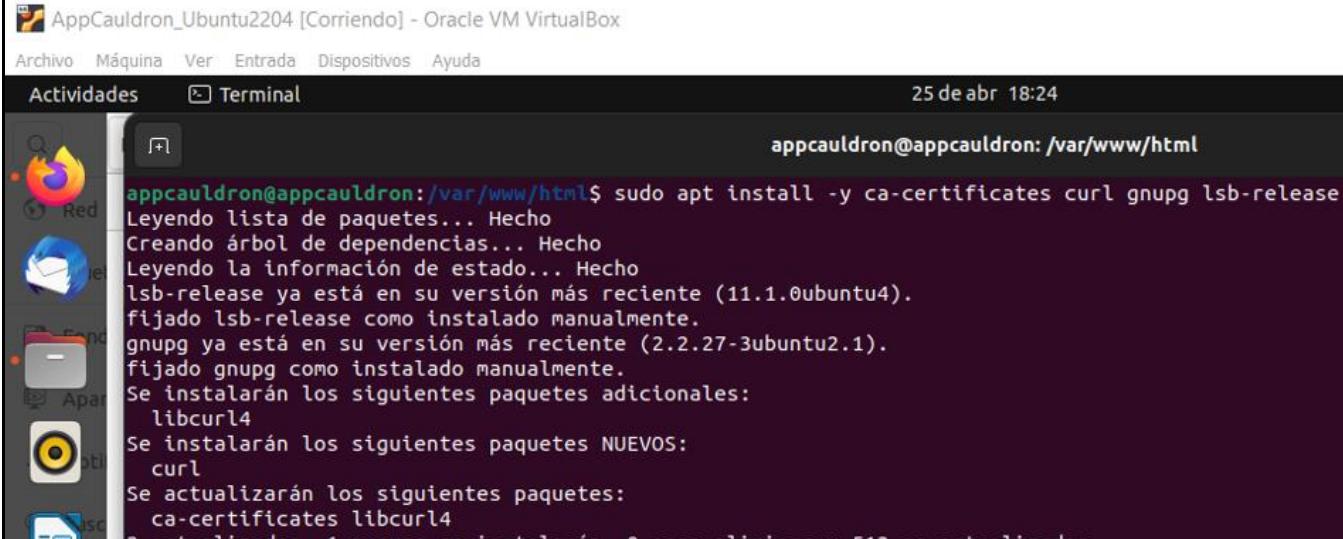


```
appcauldron@appcauldron:/var/www/html/appcauldron.com$ sudo nano /etc/sudoers
appcauldron@appcauldron:/var/www/html/appcauldron.com$ sudo cat /etc/sudoers | grep scripts
www-data ALL=(appcauldron) NOPASSWD: /var/www/html/appcauldron.com/scripts/
appcauldron@appcauldron:/var/www/html/appcauldron.com$
```

ANEXO III – INSTALACIÓN DOCKER Y DOCKER-COMPOSE

Instalamos los paquetes necesarios para instalar el repositorio de docker:

```
sudo apt install -y ca-certificates curl gnupg lsb-release
```



The screenshot shows a terminal window titled "AppCauldron_Ubuntu2204 [Corriendo] - Oracle VM VirtualBox". The terminal window has tabs for "Actividades" and "Terminal". The date and time at the top right are "25 de abr 18:24". The user is "appcauldron" and the current directory is "/var/www/html". The terminal output shows the command "sudo apt install -y ca-certificates curl gnupg lsb-release" being run, followed by its execution and completion message. It also lists packages being installed (curl, libcurl4) and updated (ca-certificates). The message concludes with "2 actualizados, 1 nuevos se instalarán, 0 para eliminar y 512 no actualizados".

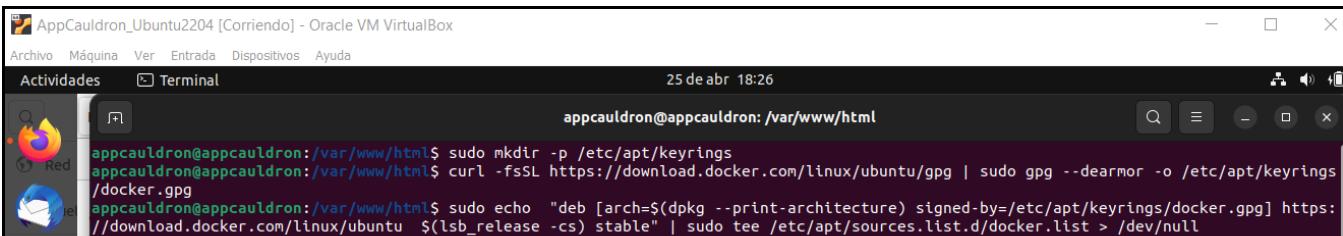
Añadimos la clave gpg de Docker en el sistema. Posteriormente instalamos el repositorio docker:

```
sudo mkdir -p /etc/apt/keyrings
```

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

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

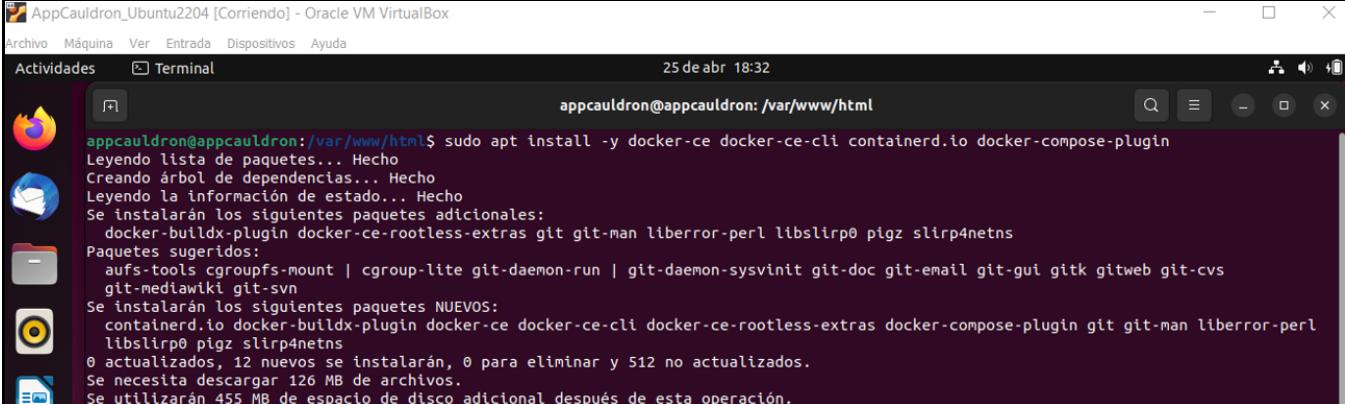
```
sudo apt update
```



The screenshot shows a terminal window titled "AppCauldron_Ubuntu2204 [Corriendo] - Oracle VM VirtualBox". The terminal window has tabs for "Actividades" and "Terminal". The date and time at the top right are "25 de abr 18:26". The user is "appcauldron" and the current directory is "/var/www/html". The terminal output shows the commands to create the keyring directory, download the GPG key, and add it to the keyring. It then adds a Docker repository entry to the sources list and runs an apt update command. The output ends with "0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 512 no actualizados".

Instalamos Docker:

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



```
appcauldron@appcauldron:~$ sudo apt install -y docker-ce docker-ce-cli containerd.io docker-compose-plugin
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  docker-buildx-plugin docker-ce-rootless-extras git git-man liberror-perl libslirp0 pigz slirp4netns
Paquetes sugeridos:
  aufs-tools cgroupfs-mount | cgroup-lite git-daemon-run | git-daemon-sysvinit git-doc git-email git-gui gitk gitweb git-cvs
  git-mediawiki git-svn
Se instalarán los siguientes paquetes NUEVOS:
  containerd.io docker-buildx-plugin docker-ce docker-ce-rootless-extras docker-compose-plugin git git-man liberror-perl
  libslirp0 pigz slirp4netns
0 actualizados, 12 nuevos se instalarán, 0 para eliminar y 512 no actualizados.
Se necesita descargar 126 MB de archivos.
Se utilizarán 455 MB de espacio de disco adicional después de esta operación.
```

Comprobamos que funciona la instalación:

```
sudo docker run hello-world
```

```
appcauldron@appcauldron:~$ sudo docker run hello-world
[sudo] contraseña para appcauldron:
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:266b191e926f65542fa8daaec01a192c4d292bff79426f47300a046e1bc576fd
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

Creamos un grupo docker para evitar tener que usar sudo con el usuario appcauldron:

```
sudo groupadd docker
```

```
sudo usermod -aG docker $USER
```

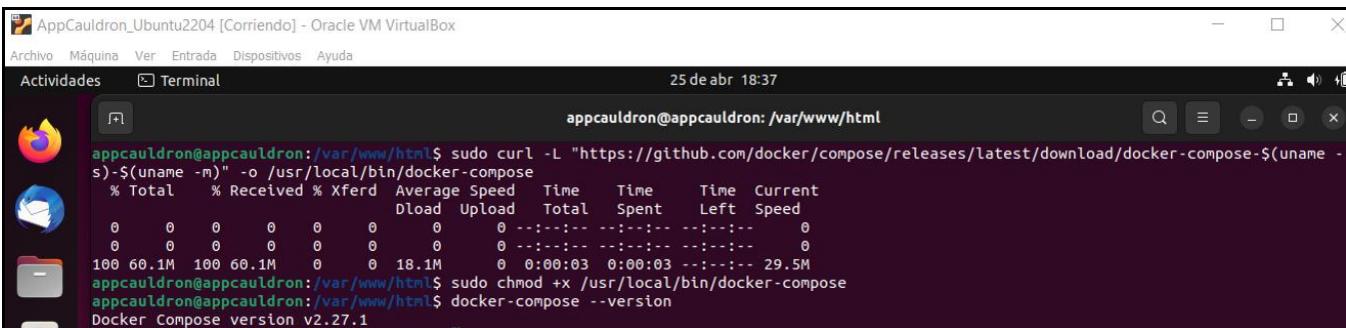
```
appcauldron@appcauldron:/var/www/html$ sudo groupadd docker
groupadd: el grupo «docker» ya existe
appcauldron@appcauldron:/var/www/html$ sudo usermod -aG docker $USER
appcauldron@appcauldron:/var/www/html$ █
```

Instalamos docker-compose y comprobamos que se ha instalado correctamente:

```
sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

```
sudo chmod +x /usr/local/bin/docker-compose
```

```
docker-compose --version
```



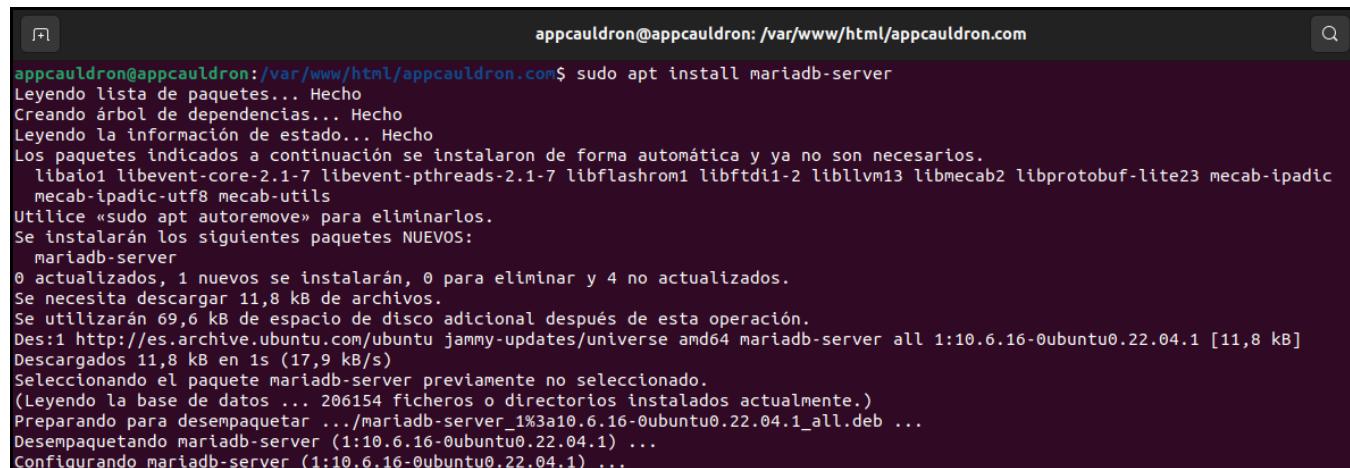
The screenshot shows a terminal window titled 'AppCauldron_Ubuntu2204 [Corriendo] - Oracle VM VirtualBox'. The window has a dark theme with white text. It displays the command 'curl' being used to download Docker Compose from GitHub. The output shows the progress of the download, which completes at 100% with a size of 29.5M. After the download, the command 'chmod +x' is run to make the executable file executable. Finally, the command 'docker-compose --version' is run to verify the installation, showing the version 'Docker Compose version v2.27.1'.

```
appcauldron@appcauldron:/var/www/html$ sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
  % Total    % Received % Xferd  Average Speed   Time   Time     Time Current
               Dload  Upload Total Spent   Spent    Left  Speed
0     0    0     0    0     0      0 --:--:-- --:--:-- --:--:-- 0
0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:-- 0
100  60.1M 100  60.1M  0     0  18.1M      0  0:00:03  0:00:03  --:--:-- 29.5M
appcauldron@appcauldron:/var/www/html$ sudo chmod +x /usr/local/bin/docker-compose
appcauldron@appcauldron:/var/www/html$ docker-compose --version
Docker Compose version v2.27.1
```

ANEXO IV – INSTALACION MARIADB Y PHP

Instalamos mariadb-server con el comando:

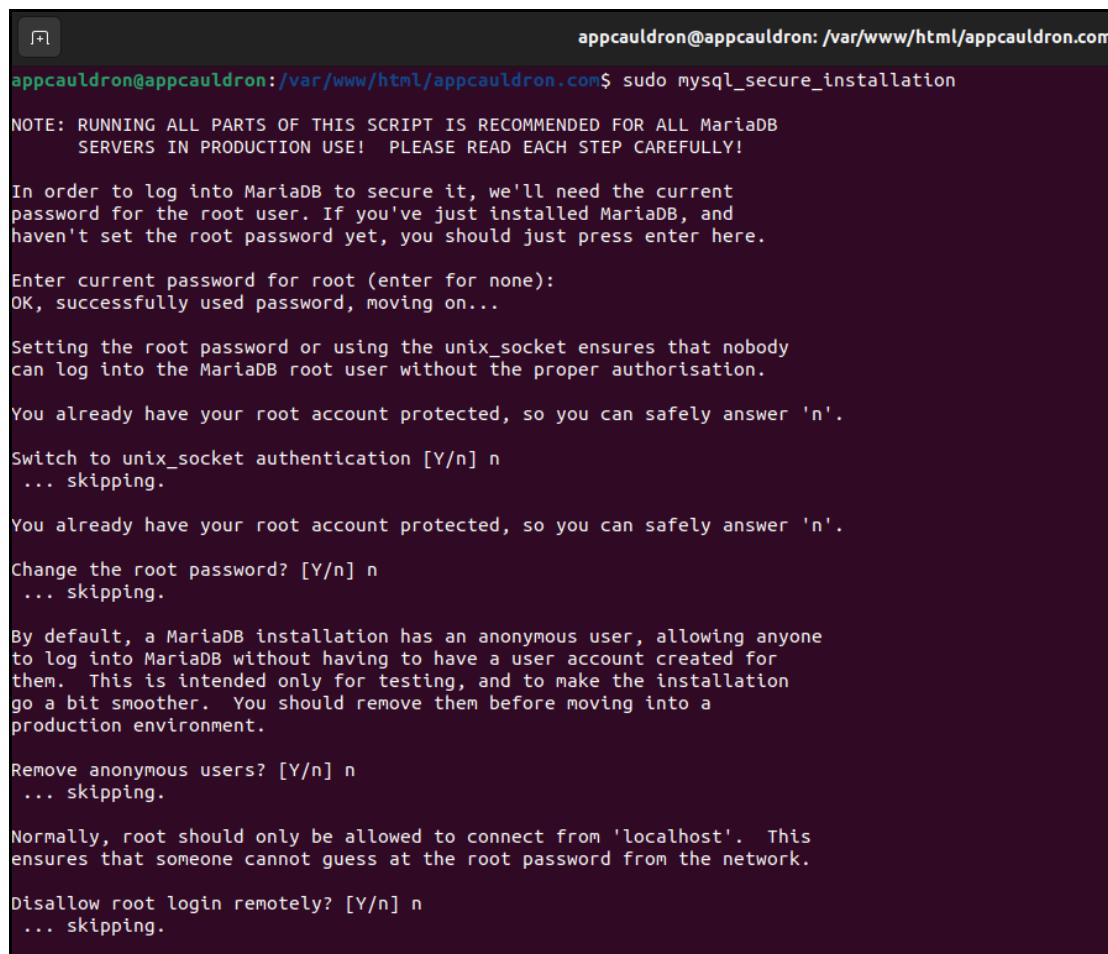
```
sudo apt install mariadb-server
```



```
appcauldron@appcauldron:/var/www/html/appcauldron.com$ sudo apt install mariadb-server
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  libai01 libevent-core-2.1-7 libevent-pthreads-2.1-7 libflashrom1 libftdi1-2 liblvm13 libmecab2 libprotobuf-lite23 mecab-ipadic
  mecab-ipadic-utf8 mecab-utils
Utilice «sudo apt autoremove» para eliminarlos.
Se instalarán los siguientes paquetes NUEVOS:
  mariadb-server
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 4 no actualizados.
Se necesita descargar 11,8 kB de archivos.
Se utilizarán 69,6 kB de espacio de disco adicional después de esta operación.
Des:1 http://es.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 mariadb-server all 1:10.6.16-0ubuntu0.22.04.1 [11,8 kB]
Descargados 11,8 kB en 1s (17,9 kB/s)
Seleccionando el paquete mariadb-server previamente no seleccionado.
(Leyendo la base de datos ... 206154 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../mariadb-server_1%3a10.6.16-0ubuntu0.22.04.1_all.deb ...
Desempaquetando mariadb-server (1:10.6.16-0ubuntu0.22.04.1) ...
Configurando mariadb-server (1:10.6.16-0ubuntu0.22.04.1) ...
```

Ejecutamos el script de seguridad:

```
sudo mysql_secure_installation
```



```
appcauldron@appcauldron:/var/www/html/appcauldron.com$ sudo mysql_secure_installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user. If you've just installed MariaDB, and
haven't set the root password yet, you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password or using the unix_socket ensures that nobody
can log into the MariaDB root user without the proper authorisation.

You already have your root account protected, so you can safely answer 'n'.

Switch to unix_socket authentication [Y/n] n
  ... skipping.

You already have your root account protected, so you can safely answer 'n'.

Change the root password? [Y/n] n
  ... skipping.

By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them. This is intended only for testing, and to make the installation
go a bit smoother. You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] n
  ... skipping.

Normally, root should only be allowed to connect from 'localhost'. This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] n
  ... skipping.
```

Instalamos php y el módulo apache con el comando:

```
sudo apt install php libapache2-mod-php
```

```
appcauldron@appcauldron:/var/www/html/appcauldron.com$ sudo apt install php libapache2-mod-php
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  libaio1 libevent-core-2.1-7 libevent-pthreads-2.1-7 libflashrom1 libftdi1-2 liblvm13 libmecab2 libprotobuf-lite23 mecab-ipadic
  mecab-ipadic-utf8 mecab-utils
Utilice «sudo apt autoremove» para eliminarlos.
Se instalarán los siguientes paquetes adicionales:
  libapache2-mod-php8.1 php-common php8.1 php8.1-cli php8.1-common php8.1-opcache php8.1-readline
Paquetes sugeridos:
  php-pear
Se instalarán los siguientes paquetes NUEVOS:
  libapache2-mod-php libapache2-mod-php8.1 php php-common php8.1 php8.1-cli php8.1-common php8.1-opcache php8.1-readline
0 actualizados, 9 nuevos se instalarán, 0 para eliminar y 4 no actualizados.
Se necesita descargar 5.135 kB de archivos.
Se utilizarán 21,3 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
```

Instalamos los paquetesopcionales:

```
sudo apt install php-cgi
```

```
appcauldron@appcauldron:/var/www/html/appcauldron.com$ sudo apt install php-cgi
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  libaio1 libevent-core-2.1-7 libevent-pthreads-2.1-7 libflashrom1 libftdi1-2 liblvm13 libmecab2 libprotobuf-lite23 mecab-ipadic
  mecab-ipadic-utf8 mecab-utils
Utilice «sudo apt autoremove» para eliminarlos.
Se instalarán los siguientes paquetes adicionales:
  php8.1-cgi
Paquetes sugeridos:
  php-pear
Se instalarán los siguientes paquetes NUEVOS:
  php-cgi php8.1-cgi
0 actualizados, 2 nuevos se instalarán, 0 para eliminar y 4 no actualizados.
Se necesita descargar 1.790 kB de archivos.
Se utilizarán 10,9 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
```

```
sudo apt install php-mysql
```

```
appcauldron@appcauldron:/var/www/html/appcauldron.com$ sudo apt install php-mysql
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  libaio1 libevent-core-2.1-7 libevent-pthreads-2.1-7 libflashrom1 libftdi1-2 liblvm13 libmecab2 libprotobuf-lite23 mecab-ipadic
  mecab-ipadic-utf8 mecab-utils
Utilice «sudo apt autoremove» para eliminarlos.
Se instalarán los siguientes paquetes adicionales:
  php8.1-mysql
Se instalarán los siguientes paquetes NUEVOS:
  php-mysql php8.1-mysql
0 actualizados, 2 nuevos se instalarán, 0 para eliminar y 4 no actualizados.
Se necesita descargar 132 kB de archivos.
Se utilizarán 476 kB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
```

```
sudo apt install php-pgsql
```

```
appcauldron@appcauldron: /var/www/html/appcauldron.com$ sudo apt install php-pgsql
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  libaio1 libevent-core-2.1-7 libevent-pthreads-2.1-7 libflashrom1 libftdi1-2 liblomm13 libmecab2 libprotobuf-lite23 mecab-ipadic
  mecab-ipadic-utf8 mecab-utils
Utilice «sudo apt autoremove» para eliminarlos.
Se instalarán los siguientes paquetes adicionales:
  libpq5 php8.1-pgsql
Se instalarán los siguientes paquetes NUEVOS:
  libpq5 php-pgsql php8.1-pgsql
0 actualizados, 3 nuevos se instalarán, 0 para eliminar y 4 no actualizados.
Se necesita descargar 214 kB de archivos.
Se utilizarán 674 kB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
```

A continuación reiniciamos apache y probamos un código de prueba:

```
sudo systemctl restart apache2.service
```

```
sudo nano testphp.php
```

```
<?php
```

```
phpinfo();
```

```
?>
```

The terminal session shows the following commands and output:

```
appcauldron@appcauldron: /var/www/html/appcauldron.com$ sudo apt install php-pgsql
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  libaio1 libevent-core-2.1-7 libevent-pthreads-2.1-7 libflashrom1 libftdi1-2 liblomm13 libmecab2 libprotobuf-lite23 mecab-ipadic
  mecab-ipadic-utf8 mecab-utils
Utilice «sudo apt autoremove» para eliminarlos.
Se instalarán los siguientes paquetes adicionales:
  libpq5 php8.1-pgsql
Se instalarán los siguientes paquetes NUEVOS:
  libpq5 php-pgsql php8.1-pgsql
0 actualizados, 3 nuevos se instalarán, 0 para eliminar y 4 no actualizados.
Se necesita descargar 214 kB de archivos.
Se utilizarán 674 kB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s

appcauldron@appcauldron: /var/www/html/appcauldron.com$ sudo systemctl restart apache2.service
appcauldron@appcauldron: /var/www/html/appcauldron.com$ sudo nano testphp.php
appcauldron@appcauldron: /var/www/html/appcauldron.com$ cat testphp.php
<?php
    phpinfo();
?>

appcauldron@appcauldron: /var/www/html/appcauldron.com$ sudo chown www-data:www-data -R /var/www/html
appcauldron@appcauldron: /var/www/html/appcauldron.com$ 
```

The browser window shows the PHP info page with the following details:

PHP Version 8.1.2-1ubuntu2.17

System	Linux appcauldron 6.5.0-35-generic #35~22.04.1-Ubuntu SMP PREEMPT_DYNAMIC Tue May 7 09:00:52 UTC 2024 x86_64
Build Date	May 1 2024 10:10:07
Build System	Linux
Server API	Apache 2.0 Handler

ANEXO V – APPCAULDRON.COM - /var/www/html/appcauldron.com/

10.8. index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>App Cauldron</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="css/appcauldron-style.css">
</head>
<body>
<div class="header">
    <h1></h1>
</div>
<div class="navbar">
    <a href="index.html">Home</a>
    <a href="cauldron.html">The Cauldron</a>
    <a href="spells.php">Hechizos</a>
    <a href="grimoire.php">Grimorio</a>
    <a href="info.html">Info</a>
    <a href="commands.php" class="right">Contemplar el Universo</a>
</div>
<div class="row">
    <div class="side">
        ¡Bienvenido a App Cauldron!
    </div>
    <div class="main">
        Crea tu aplicación web seleccionando los ingredientes en el caldero.<br>
        <br>¡Fácil, cómodo y rápido!<br>
        <br>En <a href="cauldron.html">The Cauldron</a> podrás indicar los ingredientes (servicios) que deseas incluir.<br>
        <br>¡Crea el hechizo capaz de realizar tu aplicación web soñada!<br>
        <br>En <a href="spells.php">Hechizos</a> podrás inspeccionar el hechizo creado (código de docker-compose)<br>
        <br>Podrás manejarlo a tu voluntad:<br>
        <br>- Lanzarlo (iniciar el contenedor)<br>
        <br>- Pararlo (parar los servicios)<br>
        <br>- Reanudarlo (volver a levantarlos)<br>
        <br>- Deshacerlo (parar el contenedor, red y volúmenes)<br>
        <br>En <a href="grimoire.php">Grimorio</a> podrás revisar las instrucciones del hechizo creado y asegurarte que funciona correctamente<br>
        <br>En <a href="commands.php">Contemplar el universo</a> podrás revisar el estado de los procesos docker, los contenedores las redes, las imágenes y los volúmenes<br>
        <br>También podrás borrar los volúmenes<br>
        <br>
        <br>Selecciona <a href="cauldron.html">The Cauldron</a> para comenzar<br>
        <br>
    </div>
</div>
</body>
</html>
```

10.9. cauldron.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>App Cauldron</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="css/appcauldron-style.css">
<script>
function validateForm(event) {
    var checkboxes = document.querySelectorAll('input[type="checkbox"]');
    var isChecked = false;
    for (var i = 0; i < checkboxes.length; i++) {
        if (checkboxes[i].checked) {
            isChecked = true;
            break;
        }
    }
    if (!isChecked) {
        event.preventDefault(); // Evita el envío del formulario
        alert('Debes seleccionar al menos una opción.');
    }
}
document.addEventListener('DOMContentLoaded', function() {
    var form = document.getElementById('ingredientes');
    form.addEventListener('submit', validateForm);
});
</script>
</head>
<body>
<div class="header">
    <h1></h1>
</div>
<div class="navbar">
    <a href="index.html">Home</a>
    <a href="cauldron.html">The Cauldron</a>
    <a href="spells.php">Hechizos</a>
    <a href="grimoire.php">Grimorio</a>
    <a href="info.html">Info</a>
    <a href="commands.php" class="right">Contemplar el Universo</a>
</div>
<div class="row">
    <div class="side">
        Selecciona los ingredientes de tu aplicación web
        <br>
    </div>
    <div class="main">
        <form id="ingredientes" action="cauldron2.php" method="post">
            <input type="checkbox" id="ch1" name="bdd" value="bdd">
            <label for="ch1">Base de Datos (MySQL - Imagen Oficial)</label>
            <br>
            <br>
            <input type="checkbox" id="ch2" name="nginx" value="nginx">
            <label for="ch2">Servidor Web (Nginx - Imagen Oficial)</label>
            <br>
            <br>
            <input type="checkbox" id="ch3" name="ftp" value="ftp">
            <label for="ch3">Servidor FTP (Pure-FTPD - Imagen Andrew Stilliard)</label>
            <br>
            <br>
            <input type="checkbox" id="ch4" name="netdata" value="netdata">
            <label for="ch4">Monitorización (Netdata - Imagen Oficial)</label>
            <br>
            <br>
            <input type="checkbox" id="ch5" name="lms" value="lms">
            <label for="ch5">Gestión de aprendizaje (Moodle - Imagen Bitnami)</label>
            <br>
            <br>
            <input type="checkbox" id="ch6" name="video" value="video">
```

```
<label for="ch8">Videoconferencia (Jitsi Meet - Imagen Oficial)</label>
<br>
<br>
<input type="checkbox" id="ch7" name="wordpress" value="wordpress">
<label for="ch9">Wordpress (Wordpress - Imagen Oficial)</label>
<br>
<br>
<input type="checkbox" id="ch8" name="office" value="office">
<label for="ch10">Office (WPS Office - Imagen linuxserver.io)</label>
<br>
<br>
<input type="submit" name="submit" value="Arrojar al caldero">
</div>
</div>
</body>
</html>
```

10.10. cauldron2.php

```
$servicio = $nginx . "\n";
fwrite($ficheroreceta, $ingrediente);
fwrite($ficheroservicios, $servicio);
}
if (isset($ftp) && !empty($ftp)) {
$ingrediente = $ftp . "\n\n";
$servicio = $ftp . "\n";
fwrite($ficheroreceta, $ingrediente);
fwrite($ficheroservicios, $servicio);
}
if (isset($netdata) && !empty($netdata)) {
$ingrediente = $netdata . "\n\n";
$servicio = $netdata . "\n";
fwrite($ficheroreceta, $ingrediente);
fwrite($ficheroservicios, $servicio);
}
if (isset($lms) && !empty($lms)) {
$ingrediente = $lms . "\n\n";
$servicio = $lms . "\n";
fwrite($ficheroreceta, $ingrediente);
fwrite($ficheroservicios, $servicio);
}
if (isset($video) && !empty($video)) {
$ingrediente = $video . "\n\n";
$servicio = $video . "\n";
fwrite($ficheroreceta, $ingrediente);
fwrite($ficheroservicios, $servicio);
}
if (isset($wordpress) && !empty($wordpress)) {
$ingrediente = $wordpress . "\n\n";
$servicio = $wordpress . "\n";
fwrite($ficheroreceta, $ingrediente);
fwrite($ficheroservicios, $servicio);
}
if (isset($office) && !empty($office)) {
$ingrediente = $office . "\n\n";
$servicio = $office . "\n";
fwrite($ficheroreceta, $ingrediente);
fwrite($ficheroservicios, $servicio);
}
fclose($ficheroreceta);
fclose($ficheroservicios);
shell_exec('/var/www/html/appcauldron.com/scripts/magic_cauldron.sh');
$spell = 'generatedfiles/spell.yaml';
if (file_exists($spell)) {
$hechizo = file_get_contents($spell);
echo '<pre>' . htmlspecialchars($hechizo) . '</pre>';
}
else {
echo '<pre>Algo ha fallado en la elaboración..</pre>';
}
?>
</div>
</body>
</html>
```

10.11. spells.php

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>App Cauldron</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="css/appcauldron-style.css">
</head>
<body>
<div class="header">
    <h1></h1>
</div>
<div class="navbar">
    <a href="index.html">Home</a>
    <a href="cauldron.html">The Cauldron</a>
    <a href="spells.php">Hechizos</a>
    <a href="grimoire.php">Grimorio</a>
    <a href="info.html">Info</a>
    <a href="commands.php" class="right">Contemplar el Universo</a>
</div>
<script>
    function crearhechizo() {
        // Crear un formulario invisible y enviarlo
        var form = document.createElement('form');
        form.method = 'POST';
        form.action = ''; // Enviar la solicitud al mismo archivo PHP
        var input = document.createElement('input');
        input.type = 'hidden';
        input.name = 'crear_hechizo';
        input.value = '1';
        form.appendChild(input);
        document.body.appendChild(form);
        form.submit();
    }
    function pararhechizo() {
        // Crear un formulario invisible y enviarlo
        var form = document.createElement('form');
        form.method = 'POST';
        form.action = ''; // Enviar la solicitud al mismo archivo PHP
        var input = document.createElement('input');
        input.type = 'hidden';
        input.name = 'parar_hechizo';
        input.value = '1';
        form.appendChild(input);
        document.body.appendChild(form);
        form.submit();
    }
    function continuarhechizo() {
        // Crear un formulario invisible y enviarlo
        var form = document.createElement('form');
        form.method = 'POST';
        form.action = ''; // Enviar la solicitud al mismo archivo PHP
        var input = document.createElement('input');
        input.type = 'hidden';
        input.name = 'continuar_hechizo';
        input.value = '1';
        form.appendChild(input);
        document.body.appendChild(form);
        form.submit();
    }
    function deshacerhechizo() {
        // Crear un formulario invisible y enviarlo
        var form = document.createElement('form');
        form.method = 'POST';
        form.action = ''; // Enviar la solicitud al mismo archivo PHP
        var input = document.createElement('input');
        input.type = 'hidden';
        input.name = 'deshacer_hechizo';
    }
</script>
```

```
        input.value = '1';
        form.appendChild(input);
        document.body.appendChild(form);
        form.submit();
    }
    function eliminarhechizo() {
        // Crear un formulario invisible y enviarlo
        var form = document.createElement('form');
        form.method = 'POST';
        form.action = ''; // Enviar la solicitud al mismo archivo PHP
        var input = document.createElement('input');
        input.type = 'hidden';
        input.name = 'eliminar_hechizo';
        input.value = '1';
        form.appendChild(input);
        document.body.appendChild(form);
        form.submit();
    }
</script>


<div class="side">
        <?php
            $spell = 'generatedfiles/spell.yaml';
            if (file_exists($spell)) {
                echo "Desata la magia<br>";
            } else {
                echo "No hay hechizos";
            }
        ?>
        <?php
            if (file_exists($spell)) {
        ?>
        <br>
        <button onclick="crearhechizo()">Crear hechizo</button>
        <div id="cast">
            <?php
                if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['crear_hechizo'])) {
                    shell_exec('sudo -u appcauldron scripts/cast_spell.sh');
                }
            ?>
        </div>
        <?php
    }
    ?>
        <?php
            if (file_exists($spell)) {
        ?>
        <br>
        <button onclick="pararhechizo()">Parar hechizo</button>
        <div id="cast">
            <?php
                if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['parar_hechizo'])) {
                    shell_exec('sudo -u appcauldron scripts/stop_spell.sh');
                }
            ?>
        </div>
        <?php
    }
    ?>
        <?php
            if (file_exists($spell)) {
        ?>
        <br>
        <button onclick="continuarhechizo()">Continuar hechizo</button>
        <div id="cast">
            <?php
                if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['continuar_hechizo'])) {
                    shell_exec('sudo -u appcauldron scripts/resume_spell.sh');
                }
            ?>


```

```
?>
</div>
<?php
}
?>
<?php
if (file_exists($spell)) {
?>
<br>
<button onclick="deshacerhechizo()">Deshacer hechizo</button>
<div id="dismiss">
<?php
if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['deshacer_hechizo'])) {
    shell_exec('sudo -u appcauldron scripts/undo_spell.sh');
}
?>
</div>
<?php
}
?>
<br>
<?php
if (file_exists($spell)) {
?>
<br>
<button onclick="eliminarhechizo()">Eliminar hechizo</button>
<div id="exterminate">
<?php
if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['eliminar_hechizo'])) {
    unlink($spell);
    $grimoario = 'generatedfiles/grimoire.txt';
    if (file_exists($grimoario)) {
        unlink($grimoario);
    }
    header("Location: http://$_SERVER[HTTP_HOST]$_SERVER[REQUEST_URI]");
}
?>
</div>
<?php
}
?>
</div>
<div class="main">
<?php
$spell = 'generatedfiles/spell.yaml';
if (file_exists($spell)) {
$hechizo = file_get_contents($spell);
echo '<pre>' . htmlspecialchars($hechizo) . '</pre>';
}
else {
echo '<pre>Utiliza el <a href="cauldron.html">caldero</a>!</pre>';
}
?>
</div>
</div>
</body>
</html>
```

10.12. grimoire.php

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>App Cauldron</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="css/appcauldron-style.css">
</head>
<body>
<div class="header">
    <h1></h1>
</div>
<div class="navbar">
    <a href="index.html">Home</a>
    <a href="cauldron.html">The Cauldron</a>
    <a href="spells.php">Hechizos</a>
    <a href="grimoire.php">Grimorio</a>
    <a href="info.html">Info</a>
    <a href="commands.php" class="right">Contemplar el Universo</a>
</div>
<div class="row">
    <div class="side">
        <?php
        $grimoire = 'generatedfiles/grimoire.txt';
        if (file_exists($grimoire) && !empty(trim(file_get_contents($grimoire)))) {
            echo "Sigue bien las instrucciones<br>";
        } else {
            echo "No hay entradas para tu hechizo";
        }
        ?>
    </div>
    <div class="main">
        <?php
        if (file_exists($grimoire) && !empty(trim(file_get_contents($grimoire)))) {
            $entrada = file_get_contents($grimoire);
            // Escapar el contenido completo
            $entrada_escaped = htmlspecialchars($entrada);
            // Permitir enlaces reemplazando las partes específicas
            // Aquí asumimos que el enlace está en el formato [enlace:URL]
            // Por ejemplo: [enlace:http://example.com]
            $entrada_formateada = preg_replace_callback(
                '/\[enlace:(https?:\/\/\S+)\]/',
                function ($matches) {
                    // Decodificar cualquier HTML especial en la URL
                    $url = htmlspecialchars_decode($matches[1]);
                    // Crear el enlace HTML permitido con target="_blank" para abrir en una nueva pestaña
                    return '<a href="' . htmlspecialchars($url) . '" target="_blank">' . htmlspecialchars($url) . '</a>';
                },
                $entrada_escaped
            );
            echo '<pre>' . $entrada_formateada . '</pre>';
        } else {
            echo '<pre>Utiliza el <a href="cauldron.html">caldero</a>!</pre>';
        }
        ?>
    </div>
</div>
</body>
</html>
```

10.13. commands.php

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>App Cauldron</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="css/appcauldron-style.css">
</head>
<body>
<div class="header">
    <h1></h1>
</div>
<div class="navbar">
    <a href="index.html">Home</a>
    <a href="cauldron.html">The Cauldron</a>
    <a href="spells.php">Hechizos</a>
    <a href="grimoire.php">Grimorio</a>
    <a href="info.html">Info</a>
    <a href="commands.php" class="right">Contemplar el Universo</a>
</div>
<script>
    function borrarcontenedoresparados() {
        // Crear un formulario invisible y enviarlo
        var form = document.createElement('form');
        form.method = 'POST';
        form.action = ''; // Enviar la solicitud al mismo archivo PHP
        var input = document.createElement('input');
        input.type = 'hidden';
        input.name = 'borrar_contenedores_parados';
        input.value = '1';
        form.appendChild(input);
        document.body.appendChild(form);
        form.submit();
    }
    function borrarcontenedores() {
        // Crear un formulario invisible y enviarlo
        var form = document.createElement('form');
        form.method = 'POST';
        form.action = ''; // Enviar la solicitud al mismo archivo PHP
        var input = document.createElement('input');
        input.type = 'hidden';
        input.name = 'borrar_contenedores';
        input.value = '1';
        form.appendChild(input);
        document.body.appendChild(form);
        form.submit();
    }
    function borrarcontenedoresactivos() {
        // Crear un formulario invisible y enviarlo
        var form = document.createElement('form');
        form.method = 'POST';
        form.action = ''; // Enviar la solicitud al mismo archivo PHP
        var input = document.createElement('input');
        input.type = 'hidden';
        input.name = 'borrar_contenedores_activos';
        input.value = '1';
        form.appendChild(input);
        document.body.appendChild(form);
        form.submit();
    }
    function borrarredesanonimas() {
        // Crear un formulario invisible y enviarlo
        var form = document.createElement('form');
        form.method = 'POST';
        form.action = ''; // Enviar la solicitud al mismo archivo PHP
        var input = document.createElement('input');
        input.type = 'hidden';
        input.name = 'borrar_redes_anonimas';
```

```
        input.value = '1';
        form.appendChild(input);
        document.body.appendChild(form);
        form.submit();
    }
    function borrarvolanomimos() {
        // Crear un formulario invisible y enviarlo
        var form = document.createElement('form');
        form.method = 'POST';
        form.action = ''; // Enviar la solicitud al mismo archivo PHP
        var input = document.createElement('input');
        input.type = 'hidden';
        input.name = 'borrar_vol_anonimos';
        input.value = '1';
        form.appendChild(input);
        document.body.appendChild(form);
        form.submit();
    }
    function borrarvolumenes() {
        // Crear un formulario invisible y enviarlo
        var form = document.createElement('form');
        form.method = 'POST';
        form.action = ''; // Enviar la solicitud al mismo archivo PHP
        var input = document.createElement('input');
        input.type = 'hidden';
        input.name = 'borrar_volumenes';
        input.value = '1';
        form.appendChild(input);
        document.body.appendChild(form);
        form.submit();
    }
</script>
<div class="row">
    <div class="side">
        <p class="p1">docker ps</p>
    </div>
    <div class="main">
        <p class="p1">
            <?php
                $result = shell_exec('sudo -u appcauldron scripts/docker_ps.sh');
                echo '<pre>' . htmlspecialchars($result) . '</pre>';
            ?>
        </p>
    </div>
    <div class="side">
        <p class="p1">docker container ls -a</p>
        <button onclick="borrarcontenedoresparados()">Borrar contenedores parados</button>
        <div id="deletecontainers">
            <?php
                if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['borrar_contenedores_parados'])) {
                    shell_exec("sudo -u appcauldron scripts/delete_stopped_containers.sh");
                    header("Location: http://$_SERVER[HTTP_HOST]$_SERVER[REQUEST_URI]");
                }
            ?>
        </div>
        <br>
        <button onclick="borrarcontenedores()">Borrar TODOS los contenedores</button>
        <div id="deletecontainers">
            <?php
                if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['borrar_contenedores'])) {
                    shell_exec("sudo -u appcauldron scripts/delete_containers.sh");
                    header("Location: http://$_SERVER[HTTP_HOST]$_SERVER[REQUEST_URI]");
                }
            ?>
        </div>
    </div>
    <div class="main">
        <p class="p1">
            <?php
                $result = shell_exec("sudo -u appcauldron scripts/docker_container_ls.sh");
            ?>
        </p>
    </div>
</div>
```

```
echo '<pre>' . htmlspecialchars($result) . '</pre>';
?>
</p>
</div>
<div class="side">
<p class="p1">docker-compose ls</p>
<button onclick="borrarcontenedoresactivos()">Borrar contenedores activos</button>
<div id="deleteactivecontainers">
<?php
if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['borrar_contenedores_activos'])) {
    shell_exec("sudo -u appcauldron scripts/delete_active_containers.sh");
    header("Location: http://$_SERVER[HTTP_HOST]$_SERVER[REQUEST_URI]");
}
?>
</div>
</div>
<div class="main">
<p class="p1">
<?php
$result = shell_exec("sudo -u appcauldron scripts/docker-compose_ls.sh");
echo '<pre>' . htmlspecialchars($result) . '</pre>';
?>
</p>
</div>
<div class="side">
<p class="p1">docker network ls</p>
<button onclick="borrarredesanonimas()">Borrar redes no referenciadas</button>
<div id="deleteanonnetworks">
<?php
if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['borrar_redes_anonimas'])) {
    shell_exec("sudo -u appcauldron scripts/delete_anon_networks.sh");
    header("Location: http://$_SERVER[HTTP_HOST]$_SERVER[REQUEST_URI]");
}
?>
</div>
</div>
<div class="main">
<p class="p1">
<?php
$result = shell_exec("sudo -u appcauldron scripts/docker_network_ls.sh");
echo '<pre>' . htmlspecialchars($result) . '</pre>';
?>
</p>
</div>
<div class="side">
<p class="p1">docker volume ls</p>
<button onclick="borrarvolanonomos()">Borrar volumenes no referenciados</button>
<div id="deleteanonvols">
<?php
if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['borrar_vol_anonimos'])) {
    shell_exec("sudo -u appcauldron scripts/delete_anon_vols.sh");
    header("Location: http://$_SERVER[HTTP_HOST]$_SERVER[REQUEST_URI]");
}
?>
</div>
<br>
<button onclick="borrarvolumenes()">Borrar TODOS los volumenes</button>
<div id="deletevolumes">
<?php
if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['borrar_volumenes'])) {
    shell_exec("sudo -u appcauldron scripts/delete_volumes.sh");
    header("Location: http://$_SERVER[HTTP_HOST]$_SERVER[REQUEST_URI]");
}
?>
</div>
</div>
<div class="main">
<p class="p1">
<?php
$result = shell_exec("sudo -u appcauldron scripts/docker_volume_ls.sh");
```

```
echo '<pre>' . htmlspecialchars($result) . '</pre>';
?>
</p>
</div>
<div class="side">
<p class="p1">docker images</p>
</div>
<div class="main">
<p class="p1">
<?php
$result = shell_exec("sudo -u appcauldron scripts/docker_images.sh");
echo '<pre>' . htmlspecialchars($result) . '</pre>';
?>
</p>
</div>
</div>
</body>
</html>
```

10.14. info.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>App Cauldron</title>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="css/appcauldron-style.css">
</head>
<body>
<div class="header">
    <h1></h1>
</div>
<div class="navbar">
    <a href="index.html">Home</a>
    <a href="cauldron.html">The Cauldron</a>
    <a href="spells.php">Hechizos</a>
    <a href="grimoire.php">Grimorio</a>
    <a href="info.html">Info</a>
    <a href="commands.php" class="right">Contemplar el Universo</a>
</div>
<div class="row">
    <div class="side">
        Autor
    </div>
    <div class="main">
        Francisco Javier Loscos Gil
    </div>
</div>
<div class="row">
    <div class="side">
        Contacto
    </div>
    <div class="main">
        meterre@gmail.com
    </div>
</div>
<div class="row">
    <div class="side">
        Proyecto
    </div>
    <div class="main">
        App Cauldron. Elabora tu aplicación web.
    </div>
</div>
<div class="row">
    <div class="side">
        Curso
    </div>
    <div class="main">
        Ciclo de Grado Superior Administración de Sistemas Informáticos en Red
    </div>
</div>
<div class="row">
    <div class="side">
        Centro
    </div>
    <div class="main">
        IES Pablo Serrano - Zaragoza
    </div>
</div>
<div class="row">
    <div class="side">
        Fecha
    </div>
    <div class="main">
        13 junio 2024
    </div>
</div>
</body>
</html>
```

ANEXO VI – PLANTILLAS - /var/www/html/appcauldron.com/templates/

11. Base de Datos (MySQL - Imagen Oficial)

Levanta el servicio db_mysql que utiliza la última imagen de mysql disponible.

Crearemos una base de datos llamada appcauldron-db_mysql, un usuario appcauldron, como contraseña asir123 y la misma como contraseña de root.

Asocia el puerto 3310 de nuestro host con el 3306 del contenedor.

Creará el volumen db_mysql-volume que contendrá /var/lib/mysql del contenedor.

11.1. database

```
db_mysql:  
  image: mysql:latest  
  environment:  
    MYSQL_DATABASE: 'appcauldron-db_mysql'  
    MYSQL_USER: 'appcauldron'  
    MYSQL_PASSWORD: 'asir123'  
    MYSQL_ROOT_PASSWORD: 'asir123'  
  ports:  
    - '3310:3306'  
  expose:  
    - '3310'  
  volumes:  
    - db_mysql-volume:/var/lib/mysql  
  networks:  
    network:
```

11.2. database.volumes

```
db_mysql-volume:
```

11.3. database.info

Base de Datos (MySQL - Imagen Oficial)

=====

TEST: Acceder a base de datos mysql por el puerto 3310 con usuario y contraseña proporcionados.

Usuario: root

Contraseña: asir123

Usuario: appcauldron

Contraseña: asir123

Tiene que haber creado la base de datos de prueba:

appcauldron-db_mysql

En un terminal:

Con usuario root:

Prueba de conexión. Mostramos las bases de datos y los usuarios.

```
mysql -P 3310 -u root --password=asir123 -e 'SHOW DATABASES; SELECT user,host FROM mysql.user;'
```

Abrir una sesión:

```
mysql -P 3310 -u root --password=asir123
```

Con usuario appcauldron:

Prueba de conexión. Mostramos las bases de datos.

```
mysql -P 3310 -u appcauldron --password=asir123 -e 'SHOW DATABASES;'
```

Abrir una sesión:

```
mysql -P 3310 -u appcauldron --password=asir123
```

MySQL - Imagen Oficial:

[enlace:https://hub.docker.com/_/mysql]

12. Servidor Web (Nginx - Imagen Oficial)

Levanta el servicio nginx que utiliza la última imagen de nginx disponible.

Asocia el puerto 8090 de nuestro host con el 80 del contenedor.

Asociará /var/www/html/nginx de host con /usr/share/nginx/html del contenedor.

Podremos crear una página dentro de la ruta de nuestro host para poder visualizarla a través del contenedor.

12.1. nginx

```
nginx:  
  image: nginx:latest  
  ports:  
    - '8090:80'  
  volumes:  
    - /var/www/html/nginx:/usr/share/nginx/html  
  networks:  
    network:
```

12.2. nginx.info

Servidor Web (Nginx - Imagen Oficial)

=====

TEST: Comprobar visualización página web nginx alojada en nuestro host en: /var/www/html/nginx a través del puerto 9090.

Está preparado para abrir el documento index.html situado en la ruta actualmente o lo que hayamos incluido.

[enlace:http://appcauldron:8090]

Nginx - Imagen Oficial:

[enlace:https://hub.docker.com/_/nginx]

13. Servidor FTP (Pure-FTPd - Imagen Andrew Stilliard)

Levanta el servicio ftp que utiliza una imagen de pure-ftpd creada por Andrew Stilliard y que tiene una mayor seguridad (hardened).

Asocia los puertos 20, 21 y 30000-30009 de host y contenedor.

Declaramos en las variables como servidor público 'localhost', y creamos un usuario ftp appcauldron, con contraseña asir123 y home de ftp /var/www/html/pure-ftpd.

Asociamos /var/www/html/pure-ftpd/data de host con /var/www/html/pure-ftpd del contenedor.

De esta manera utilizamos la ruta de nuestra host mencionada como ruta de ftp.

Creamos la ruta /var/www/html/pure-ftpd/data en nuestra máquina y un archivo pruebaftp.txt.

13.1. ftp

```
ftp:  
  image: stilliard/pure-ftpd:hardened  
  ports:  
    - '21:21'  
    - '20:20'  
    - '30000-30009:30000-30009'  
  environment:  
    PUBLICHOST: 'localhost'  
    FTP_USER_NAME: 'appcauldron'  
    FTP_USER_PASS: 'asir123'  
    FTP_USER_HOME: '/var/www/html/pure-ftpd'  
  volumes:  
    - /var/www/html/pure-ftpd/data:/var/www/html/pure-ftpd
```

13.2. ftp.info

Servidor FTP (Pure-FTPd - Imagen Andrew Stilliard)
=====

TEST: Acceder al ftp por el puerto 21 con el usuario y contraseña proporcionados.

Usuario: appcauldron

Contraseña: asir123

El servidor FTP está configurado para almacenar los archivos utilizando la ruta local:

/var/www/html/pure-ftpd/data

Puedes comprobar el funcionamiento desde un terminal con:

ftp ftp://appcauldron:asir123@appcauldron

Pure-FTPd - Imagen Andrew Stilliard:

[enlace:<https://github.com/stilliard/docker-pure-ftpd>]

14. Monitorización (Netdata - Imagen Oficial)

Levanta el servicio netdata que utiliza una imagen de netdata oficial.

Asocia el puerto 19999 de host y contenedor.

Se crean 3 volúmenes netdataconfig-volume, netdatalib-volume y netdatacache-volume asociadas a las rutas del contenedor /etc/netdata, /var/lib/netdata, /var/cache/netdata.

Asocia varias rutas de host con las del contenedor como sólo lectura para la monitorización.

Descubre aplicaciones/servicios locales. Mapea puertos abiertos (en escucha) a aplicaciones/servicios en ejecución con SYS_PTRACE.

14.1. netdata

```
netdata:  
  image: netdata/netdata  
  hostname: localhost  
  ports:  
    - 19999:19999  
  restart: unless-stopped  
  cap_add:  
    - SYS_PTRACE  
  security_opt:  
    - apparmor:unconfined  
  volumes:  
    - netdataconfig-volume:/etc/netdata  
    - netdatalib-volume:/var/lib/netdata  
    - netdatacache-volume:/var/cache/netdata  
    - /etc/passwd:/host/etc/passwd:ro  
    - /etc/group:/host/etc/group:ro  
    - /proc:/host/proc:ro  
    - /sys:/host/sys:ro  
    - /etc/os-release:/host/etc/os-release:ro  
  networks:  
    network:
```

14.2. netdata.volumes

```
netdataconfig-volume:  
netdatalib-volume:  
netdatacache-volume:
```

14.3. netdata.info

```
Monitorización (Netdata - Imagen Oficial)  
=====  
TEST: Acceder al dashboard a través del puerto 19999  
Versión v1:  
[enlace:http://appcauldron:19999/v1/]  
Versión v2:  
[enlace:http://appcauldron:19999/v2/]  
Netdata - Imagen Oficial:  
[enlace:https://hub.docker.com/r/netdata/netdata]
```

15. Gestión de aprendizaje (Moodle - Imagen Bitnami)

Levanta el servicio db_moodle que utiliza la última imagen de mariadb creada por Bitnami.

Asocia el puerto 3311 de host con el 3306 del contenedor.

Se crea un usuario de la base de datos appcauldron_db, una base de datos appcauldron-db_moodle y se permiten contraseñas vacías.

Se crea el volumen db_moodle-volume asociado a la ruta del contenedor /bitnami/mariadb.

Levanta el servicio moodle que utiliza la última imagen de moodle creada por Bitnami.

Asocia los puertos 8100 y 8444 de host con el 8080 y 8443 del contenedor.

Se indica que como base de datos utiliza la levantada en el contenedor db_moodle por el puerto 3306 y que depende de la misma. Se indica que como usuario de la base de datos y base de datos utiliza las creadas en db_moodle. Se crea un usuario de moodle appcauldron, con contraseña asir123 y como nombre del sitio Moodle lo denominamos App Cauldron Moodle. Permitimos contraseñas vacías.

Se crean dos volúmenes más, moodle-volumen y moodledata-volume a las rutas del contenedor /bitnami/moodle y /bitnami/moodledata.

15.1. Ims

```
db_moodle:
  image: docker.io/bitnami/mariadb:latest
  restart: unless-stopped
  ports:
    - '3311:3306'
  environment:
    # ALLOW_EMPTY_PASSWORD is recommended only for development.
    - ALLOW_EMPTY_PASSWORD=yes
    - MARIADB_USER=appcauldron_db
    - MARIADB_DATABASE=appcauldron-db_moodle
    - MARIADB_CHARACTER_SET=utf8mb4
    - MARIADB_COLLATE=utf8mb4_unicode_ci
  volumes:
    - db_moodle-volume:/bitnami/mariadb

moodle:
  image: docker.io/bitnami/moodle:latest
  ports:
    - '8100:8080'
    - '8444:8443'
  environment:
    - MOODLE_DATABASE_HOST=db_moodle
    - MOODLE_DATABASE_PORT_NUMBER=3306
    - MOODLE_DATABASE_USER=appcauldron_db
    - MOODLE_DATABASE_NAME=appcauldron-db_moodle
    - MOODLE_USERNAME=appcauldron
    - MOODLE_PASSWORD=asir123
    - MOODLE_SITE_NAME=App Cauldron Moodle
    # ALLOW_EMPTY_PASSWORD is recommended only for development.
    - ALLOW_EMPTY_PASSWORD=yes
  volumes:
    - moodle-volume:/bitnami/moodle
    - moodledata-volume:/bitnami/moodledata
  depends_on:
    - db_moodle
```

15.2. lms.volumes

```
db_moodle-volume:  
moodle-volume:  
moodledata-volume:
```

15.3. lms.info

Gestión de aprendizaje (Moodle - Imagen Bitnami)

=====

TEST:

- Acceder a base de datos mariadb por el puerto 3311 con usuario proporcionado:

Usuario: root

Usuario: appcauldron_db

Tiene que haber creado la base de datos para moodle:

appcauldron-db_moodle

Con usuario root:

Prueba de conexión. Mostramos las bases de datos y los usuarios.

```
mysql -P 3311 -u root -e 'SHOW DATABASES; SELECT user,host FROM mysql.user;'
```

Abrir una sesión:

```
mysql -P 3311 u root
```

Con usuario appcauldron_db:

Prueba de conexión. Mostramos las bases de datos.

```
mysql -P 3311 -u appcauldron_db -e 'SHOW DATABASES;'
```

Abrir una sesión:

```
mysql -P 3311 -u appcauldron_db
```

- Conectar con Moodle por el puerto 8100 (HTTP) o 8444 (HTTPS) con las credenciales proporcionadas:

Usuario: appcauldron

Contraseña: asir123

ADVERTENCIA: La primera vez que se ejecuta o se han borrado los volúmenes, tarda unos minutos en instalar y estar disponible el Moodle.

Moodle:

```
[enlace:http://appcauldron:8100/]
```

```
[enlace:https://appcauldron:8444/]
```

Moodle login:

```
[enlace:http://appcauldron:8100/login/index.php]
```

```
[enlace:https://appcauldron:8444/login/index.php]
```

Moodle - Imagen Bitnami:

```
[enlace:https://github.com/bitnami/containers/tree/main/bitnami/moodle]
```

16. Videoconferencia (Jitsi Meet - Imagen Oficial)

La instalación manual requiere descargar la última versión.

En la carpeta Descargas del usuario appcauldron creamos la carpeta jitsi y procedemos a la descarga con:

```
wget $(curl -s https://api.github.com/repos/jitsi/docker-jitsi-meet/releases/latest | grep 'zip' | cut -d\" -f4)
```

Descomprimimos con:

```
unzip stable-9457-2
```

Copiamos el archive de variables de entorno de ejemplo env.example a .env:

```
appcauldron@appcauldron:~/Descargas/jitsi/jitsi-docker-jitsi-meet-2423033$ cp env.example .env
```

Generamos contraseñas seguras para las opciones del archivo .env ejecutando el script:

```
./gen-passwords.sh
```

Por último creamos las carpetas CONFIG de configuración:

```
mkdir -p ~/.jitsi-meet-cfg/{web,transcripts,prosody/config,prosody/prosody-plugins-custom,jicofo,jvb,jigasi,jibri}
```

Vamos a modificar el archivo .env para asociar el puerto de host 8443 con el 443 del contenedor, y el puerto 8000 de host con el 80. De todas maneras, entraremos por el puerto seguro 8443 porque si no, no funcionará el micrófono y la cámara.

Este archivo .env lo debemos copiar a la ruta donde se va a ejecutar el comando docker-compose con el archivo spell.yaml, así que lo copiamos a:

```
/var/www/html/appcauldron.com/generatedfiles/
```

Levanta 4 servicios, el web que es por el que accederemos por el puerto 8443, el servidor XMPP prosody, un componente jicofo y un puente de video jvb. Establece bastantes variables y asocia las carpetas CONFIG creadas. Todas levantan las imágenes stable-9457-2 de jitsi.

16.1. jitsi

```
# Frontend
web:
  image: jitsi/web:${JITSI_IMAGE_VERSION:-stable-9457-2}
  restart: ${RESTART_POLICY:-unless-stopped}
  ports:
    - '${HTTP_PORT}:80'
    - '${HTTPS_PORT}:443'
  volumes:
    - ${CONFIG}/web:/config:Z
    - ${CONFIG}/web/crontabs:/var/spool/cron/crontabs:Z
    - ${CONFIG}/transcripts:/usr/share/jitsi-meet/transcripts:Z
  environment:
    - AMPLITUDE_ID
    - ANALYTICS_SCRIPT_URLS
    - ANALYTICS_WHITELISTED_EVENTS
    - AUDIO_QUALITY_OPUS_BITRATE
    - AUTO_CAPTION_ON_RECORD
    - BRANDING_DATA_URL
    - BOSH_RELATIVE
    - CHROME_EXTENSION_BANNER_JSON
    - COLIBRI_WEBSOCKET_PORT
    - COLIBRI_WEBSOCKET_JVB_LOOKUP_NAME
    - COLIBRI_WEBSOCKET_REGEX
    - CONFCODE_URL
    - CORS_HEADER_ACCESS_CONTROL_ALLOW_ORIGIN
    - DEFAULT_LANGUAGE
    - DEPLOYMENTINFO_ENVIRONMENT
    - DEPLOYMENTINFO_ENVIRONMENT_TYPE
    - DEPLOYMENTINFO_REGION
    - DEPLOYMENTINFO_SHARD
    - DEPLOYMENTINFO_USERREGION
    - DESKTOP_SHARING_FRAMERATE_AUTO
    - DESKTOP_SHARING_FRAMERATE_MIN
    - DESKTOP_SHARING_FRAMERATE_MAX
    - DIALIN_NUMBERS_URL
    - DIALOUT_AUTH_URL
    - DIALOUT_CODES_URL
    - DISABLE_AUDIO_LEVELS
    - DISABLE_COLIBRI_WEBSOCKET_JVB_LOOKUP
    - DISABLE_DEEP_LINKING
    - DISABLE_GRANT_MODERATOR
    - DISABLE_HTTPS
    - DISABLE_KICKOUT
    - DISABLE_LOCAL_RECORDING
    - DISABLE_POLLS
    - DISABLE_PRIVATE_CHAT
    - DISABLE_PROFILE
    - DISABLE_REACTIONS
    - DISABLE_REMOTE_VIDEO_MENU
    - DISABLE_START_FOR_ALL
    - DROPBOX_APPKEY
    - DROPBOX_REDIRECT_URI
    - DYNAMIC_BRANDING_URL
    - ENABLE_AUDIO_PROCESSING
    - ENABLE_AUTH
    - ENABLE_AUTH_DOMAIN
    - ENABLE_BREAKOUT_ROOMS
    - ENABLE_CALENDAR
    - ENABLE_COLIBRI_WEBSOCKET
    - ENABLE_COLIBRI_WEBSOCKET_UNSAFE_REGEX
    - ENABLE_E2EPING
    - ENABLE_FILE_RECORDING_SHARING
    - ENABLE_GUESTS
    - ENABLE_HSTS
    - ENABLE_HTTP_REDIRECT
    - ENABLE_IPV6
    - ENABLE_LETSENCRYPT
    - ENABLE_NO_AUDIO_DETECTION
```

- ENABLE_NOISY_MIC_DETECTION
- ENABLE_OCTO
- ENABLE_OPUS_RED
- ENABLE_PREJOIN_PAGE
- ENABLE_P2P
- ENABLE_WELCOME_PAGE
- ENABLE_CLOSE_PAGE
- ENABLE_LIVESTREAMING
- ENABLE_LIVESTREAMING_DATA_PRIVACY_LINK
- ENABLE_LIVESTREAMING_HELP_LINK
- ENABLE_LIVESTREAMING_TERMS_LINK
- ENABLE_LIVESTREAMING_VALIDATOR_REGEXP_STRING
- ENABLE_LOAD_TEST_CLIENT
- ENABLE_LOCAL_RECORDING_NOTIFY_ALL_PARTICIPANT
- ENABLE_LOCAL_RECORDING_SELF_START
- ENABLE_RECORDING
- ENABLE_REMB
- ENABLE_REQUIRE_DISPLAY_NAME
- ENABLE_SERVICE_RECORDING
- ENABLE_SIMULCAST
- ENABLE_STATS_ID
- ENABLE_STEREO
- ENABLE_SUBDOMAINS
- ENABLE_TALK_WHILE_MUTED
- ENABLE_TCC
- ENABLE_TRANSCRIPTIONS
- ENABLE_XMPP_WEBSOCKET
- ENABLE_JAAS_COMPONENTS
- ETHERPAD_PUBLIC_URL
- ETHERPAD_URL_BASE
- E2EPING_NUM_REQUESTS
- E2EPING_MAX_CONFERENCE_SIZE
- E2EPING_MAX_MESSAGE_PER_SECOND
- GOOGLE_ANALYTICS_ID
- GOOGLE_API_APP_CLIENT_ID
- HIDE_PREMEETING_BUTTONS
- HIDE_PREJOIN_DISPLAY_NAME
- HIDE_PREJOIN_EXTRA_BUTTONS
- INVITE_SERVICE_URL
- JVB_PREFER_SCTP
- LETSENCRYPT_DOMAIN
- LETSENCRYPT_EMAIL
- LETSENCRYPT_USE_STAGING
- MATOMO_ENDPOINT
- MATOMO_SITE_ID
- MICROSOFT_API_APP_CLIENT_ID
- NGINX_RESOLVER
- NGINX_WORKER_PROCESSES
- NGINX_WORKER_CONNECTIONS
- PEOPLE_SEARCH_URL
- PREFERRED_LANGUAGE
- PUBLIC_URL
- P2P_PREFERRED_CODEC
- RESOLUTION
- RESOLUTION_MIN
- RESOLUTION_WIDTH
- RESOLUTION_WIDTH_MIN
- START_AUDIO_MUTED
- START_AUDIO_ONLY
- START_SILENT
- START_WITH_AUDIO_MUTED
- START_VIDEO_MUTED
- START_WITH_VIDEO_MUTED
- TESTING_AV1_SUPPORT
- TOKEN_AUTH_URL
- TOOLBAR_BUTTONS
- TRANSLATION_LANGUAGES
- TRANSLATION_LANGUAGES_HEAD
- TZ
- USE_APP_LANGUAGE

```
- VIDEOQUALITY_BITRATE_H264_LOW
- VIDEOQUALITY_BITRATE_H264_STANDARD
- VIDEOQUALITY_BITRATE_H264_HIGH
- VIDEOQUALITY_BITRATE_H264_FULL
- VIDEOQUALITY_BITRATE_H264_ULTRA
- VIDEOQUALITY_BITRATE_H264_SS_HIGH
- VIDEOQUALITY_BITRATE_VP8_LOW
- VIDEOQUALITY_BITRATE_VP8_STANDARD
- VIDEOQUALITY_BITRATE_VP8_HIGH
- VIDEOQUALITY_BITRATE_VP8_FULL
- VIDEOQUALITY_BITRATE_VP8_ULTRA
- VIDEOQUALITY_BITRATE_VP8_SS_HIGH
- VIDEOQUALITY_BITRATE_VP9_LOW
- VIDEOQUALITY_BITRATE_VP9_STANDARD
- VIDEOQUALITY_BITRATE_VP9_HIGH
- VIDEOQUALITY_BITRATE_VP9_FULL
- VIDEOQUALITY_BITRATE_VP9_ULTRA
- VIDEOQUALITY_BITRATE_VP9_SS_HIGH
- VIDEOQUALITY_BITRATE_AV1_LOW
- VIDEOQUALITY_BITRATE_AV1_STANDARD
- VIDEOQUALITY_BITRATE_AV1_HIGH
- VIDEOQUALITY_BITRATE_AV1_FULL
- VIDEOQUALITY_BITRATE_AV1_ULTRA
- VIDEOQUALITY_BITRATE_AV1_SS_HIGH
- VIDEOQUALITY_PREFERRED_CODEC
- XMPP_AUTH_DOMAIN
- XMPP_BOSH_URL_BASE
- XMPP_DOMAIN
- XMPP_GUEST_DOMAIN
- XMPP_MUC_DOMAIN
- XMPP_RECORDER_DOMAIN
- XMPP_PORT
- WHITEBOARD_ENABLED
- WHITEBOARD_COLLAB_SERVER_PUBLIC_URL

networks:
  network:
depends_on:
  - jvb

# XMPP server
prosody:
  image: jitsi/prosody:${JITSI_IMAGE_VERSION:-stable-9457-2}
  restart: ${RESTART_POLICY:-unless-stopped}
  expose:
    - '${XMPP_PORT:-5222}'
    - '${PROSODY_S2S_PORT:-5269}'
    - '5347'
    - '${PROSODY_HTTP_PORT:-5280}'
  volumes:
    - ${CONFIG}/prosody/config:/config:Z
    - ${CONFIG}/prosody/prosody-plugins-custom:/prosody-plugins-custom:Z
  environment:
    - AUTH_TYPE
    - DISABLE_POLLS
    - ENABLE_AUTH
    - ENABLE_AV_MODERATION
    - ENABLE_BREAKOUT_ROOMS
    - ENABLE_END_CONFERENCE
    - ENABLE_GUESTS
    - ENABLE_IPV6
    - ENABLE_LOBBY
    - ENABLE_RECORDING
    - ENABLE_S2S
    - ENABLE_VISITORS
    - ENABLE_XMPP_WEBSOCKET
    - ENABLE_JAAS_COMPONENTS
    - GC_TYPE
    - GC_INC_TH
    - GC_INC_SPEED
    - GC_INC_STEP_SIZE
```

- GC_GEN_MIN_TH
- GC_GEN_MAX_TH
- GLOBAL_CONFIG
- GLOBAL_MODULES
- JIBRI_RECORDER_USER
- JIBRI_RECORDER_PASSWORD
- JIBRI_SIP_BREWERY_MUC
- JIBRI_XMPP_USER
- JIBRI_XMPP_PASSWORD
- JICOFO_AUTH_PASSWORD
- JICOFO_COMPONENT_SECRET
- JIGASI_XMPP_USER
- JIGASI_XMPP_PASSWORD
- JVB_AUTH_USER
- JVB_AUTH_PASSWORD
- JWT_APP_ID
- JWT_APP_SECRET
- JWT_ACCEPTED_ISSUERS
- JWT_ACCEPTED_AUDIENCES
- JWT_ASAP_KEYSERVER
- JWT_ALLOW_EMPTY
- JWT_AUTH_TYPE
- JWT_ENABLE_DOMAIN_VERIFICATION
- JWT_SIGN_TYPE
- JWT_TOKEN_AUTH_MODULE
- MATRIX_UVS_URL
- MATRIX_UVS_ISSUER
- MATRIX_UVS_AUTH_TOKEN
- MATRIX_UVS_SYNC_POWER_LEVELS
- MATRIX_LOBBY_BYPASS
- LOG_LEVEL
- LDAP_AUTH_METHOD
- LDAP_BASE
- LDAP_BINDDN
- LDAP_BINDPW
- LDAP_FILTER
- LDAP_VERSION
- LDAP_TLS_CIPHERS
- LDAP_TLS_CHECK_PEER
- LDAP_TLS_CACERT_FILE
- LDAP_TLS_CACERT_DIR
- LDAP_START_TLS
- LDAP_URL
- LDAP_USE_TLS
- MAX_PARTICIPANTS
- PROSODY_HELPERS
- PROSODY_AUTH_TYPE
- PROSODY_C2S_LIMIT
- PROSODY_C2S_REQUIRE_ENCRYPTION
- PROSODY_RESERVATION_ENABLED
- PROSODY_RESERVATION_REST_BASE_URL
- PROSODY_ENABLE_RATE_LIMITS
- PROSODY_ENABLE_S2S
- PROSODY_GUEST_AUTH_TYPE
- PROSODY_HTTP_PORT
- PROSODY_LOG_CONFIG
- PROSODY_MODE
- PROSODY_RATE_LIMIT_LOGIN_RATE
- PROSODY_RATE_LIMIT_SESSION_RATE
- PROSODY_RATE_LIMIT_TIMEOUT
- PROSODY_RATE_LIMIT_ALLOW_RANGES
- PROSODY_RATE_LIMIT_CACHE_SIZE
- PROSODY_S2S_LIMIT
- PROSODY_S2S_PORT
- PROSODY_TRUSTED_PROXYIES
- PROSODY_VISITOR_INDEX
- PROSODY_VISITORS_MUC_PREFIX
- PUBLIC_URL
- STUN_HOST
- STUN_PORT

```
- TURN_CREDENTIALS
- TURN_HOST
- TURNS_HOST
- TURN_PORT
- TURNS_PORT
- TURN_TRANSPORT
- TZ
- VISITORS_MAX_VISITORS_PER_NODE
- VISITORS_XMPP_DOMAIN
- VISITORS_XMPP_SERVER
- VISITORS_XMPP_PORT
- XMPP_BREAKOUT_MUC_MODULES
- XMPP_CONFIGURATION
- XMPP_DOMAIN
- XMPP_AUTH_DOMAIN
- XMPP_GUEST_DOMAIN
- XMPP_MUC_DOMAIN
- XMPP_INTERNAL_MUC_DOMAIN
- XMPP_LOBBY_MUC_MODULES
- XMPP_MODULES
- XMPP_MUC_MODULES
- XMPP_MUC_CONFIGURATION
- XMPP_INTERNAL_MUC_MODULES
- XMPP_RECORDER_DOMAIN
- XMPP_PORT
- XMPP_SERVER_S2S_PORT
- XMPP_SPEAKERSTATS_MODULES

networks:
  network:
    aliases:
      - ${XMPP_SERVER:-xmpp.meet.jitsi}

# Focus component
jicofo:
  image: jitsi/jicofo:${JITSI_IMAGE_VERSION:-stable-9457-2}
  restart: ${RESTART_POLICY:-unless-stopped}
  ports:
    - '127.0.0.1:${JICOFO_REST_PORT:-8888}:8888'
  volumes:
    - ${CONFIG}/jicofo:/config:Z
  environment:
    - AUTH_TYPE
    - BRIDGE_AVG_PARTICIPANT_STRESS
    - BRIDGE_STRESS_THRESHOLD
    - ENABLE_AUTH
    - ENABLE_AUTO_OWNER
    - ENABLE_CODEC_VP8
    - ENABLE_CODEC_VP9
    - ENABLE_CODEC_AV1
    - ENABLE_CODEC_H264
    - ENABLE_CODEC_OPUS_RED
    - ENABLE_JVB_XMPP_SERVER
    - ENABLE_OCTO
    - ENABLE_OCTO_SCTP
    - ENABLE_RECORDING
    - ENABLE_SCTP
    - ENABLE_VISITORS
    - ENABLE_AUTO_LOGIN
    - JICOFO_AUTH_LIFETIME
    - JICOFO_AUTH_PASSWORD
    - JICOFO_AUTH_TYPE
    - JICOFO_BRIDGE_REGION_GROUPS
    - JICOFO_ENABLE_AUTH
    - JICOFO_ENABLE_BRIDGE_HEALTH_CHECKS
    - JICOFO_CONF_INITIAL_PARTICIPANT_WAIT_TIMEOUT
    - JICOFO_CONF_SINGLE_PARTICIPANT_TIMEOUT
    - JICOFO_CONF_SOURCE_SIGNALING_DELAYS
    - JICOFO_CONF_MAX_AUDIO_SENDERS
    - JICOFO_CONF_MAX_VIDEO_SENDERS
    - JICOFO_CONF_STRIP_SIMULCAST
```

```

- JICOFO_CONF_SSRC_REWRITING
- JICOFO_ENABLE_HEALTH_CHECKS
- JICOFO_ENABLE_REST
- JICOFO_HEALTH_CHECKS_USE_PRESENCE
- JICOFO_MAX_MEMORY
- JICOFO_MULTI_STREAM_BACKWARD_COMPAT
- JICOFO_OCTO_REGION
- JICOFO_TRUSTED_DOMAINS
- JIBRI_BREWERY_MUC
- JIBRI_REQUEST_RETRIES
- JIBRI_PENDING_TIMEOUT
- JIGASI_BREWERY_MUC
- JIGASI_SIP_URI
- JIGASI_TRUSTED_DOMAINS
- JVB_BREWERY_MUC
- JVB_XMPP_AUTH_DOMAIN
- JVB_XMPP_INTERNAL_MUC_DOMAIN
- JVB_XMPP_PORT
- JVB_XMPP_SERVER
- MAX_BRIDGE_PARTICIPANTS
- OCTO_BRIDGE_SELECTION_STRATEGY
- PROSODY_VISITORS_MUC_PREFIX
- SENTRY_DSN="${JICOFO_SENTRY_DSN:-0}"
- SENTRY_ENVIRONMENT
- SENTRY_RELEASE
- TZ
- VISITORS_MAX_PARTICIPANTS
- VISITORS_MAX_VISITORS_PER_NODE
- VISITORS_XMPP_AUTH_DOMAIN
- VISITORS_XMPP_SERVER
- VISITORS_XMPP_DOMAIN
- XMPP_DOMAIN
- XMPP_AUTH_DOMAIN
- XMPP_INTERNAL_MUC_DOMAIN
- XMPP_MUC_DOMAIN
- XMPP_RECORDER_DOMAIN
- XMPP_SERVER
- XMPP_PORT
- MAX_SSRCs_PER_USER
- MAX_SSRC_GROUPS_PER_USER
depends_on:
- prosody
networks:
  network:

# Video bridge
jvb:
  image: jitsi/jvb:${JITSI_IMAGE_VERSION:-stable-9457-2}
  restart: ${RESTART_POLICY:-unless-stopped}
  ports:
    - '${JVB_PORT:-10000}:${JVB_PORT:-10000}/udp'
    - '127.0.0.1:${JVB_COLIBRI_PORT:-8080}:8080'
  volumes:
    - ${CONFIG}/jvb:/config:Z
  environment:
    - AUTOSCALER_SIDECAr_KEY_FILE
    - AUTOSCALER_SIDECAr_KEY_ID
    - AUTOSCALER_SIDECAr_GROUP_NAME
    - AUTOSCALER_SIDECAr_HOST_ID
    - AUTOSCALER_SIDECAr_INSTANCE_ID
    - AUTOSCALER_SIDECAr_PORT
    - AUTOSCALER_SIDECAr_REGION
    - AUTOSCALER_SIDECAr_SHUTDOWN_POLLING_INTERVAL
    - AUTOSCALER_SIDECAr_STATS_POLLING_INTERVAL
    - DOCKER_HOST_ADDRESS
    - ENABLE_COLIBRI_WEBSOCKET
    - ENABLE_JVB_XMPP_SERVER
    - ENABLE_OCTO
    - JVB_ADVERTISE_IPS
    - JVB_ADVERTISE_PRIVATE_CANDIDATES

```

```
- JVB_AUTH_USER
- JVB_AUTH_PASSWORD
- JVB_BREWERY_MUC
- JVB_DISABLE_STUN
- JVB_INSTANCE_ID
- JVB_PORT
- JVB_MUC_NICKNAME
- JVB_STUN_SERVERS
- JVB_LOG_FILE
- JVB_OCTO_BIND_ADDRESS
- JVB_OCTO_REGION
- JVB_OCTO_RELAY_ID
- JVB_REQUIRE_VALID_ADDRESS
- JVB_WS_DOMAIN
- JVB_WS_SERVER_ID
- JVB_XMPP_AUTH_DOMAIN
- JVB_XMPP_INTERNAL_MUC_DOMAIN
- JVB_XMPP_PORT
- JVB_XMPP_SERVER
- PUBLIC_URL
- SENTRY_DSN="${JVB_SENTRY_DSN:-0}"
- SENTRY_ENVIRONMENT
- SENTRY_RELEASE
- COLIBRI_REST_ENABLED
- SHUTDOWN_REST_ENABLED
- TZ
- VIDEOBRIDGE_MAX_MEMORY
- XMPP_AUTH_DOMAIN
- XMPP_INTERNAL_MUC_DOMAIN
- XMPP_SERVER
- XMPP_PORT
depends_on:
- prosody
networks:
  network:
```

16.2. jitsi.info

Videoconferencia (Jitsi Meet - Imagen Oficial)

=====

TEST: Acceder a Jitsi Meet por el puerto 8443 (HTTPS).

Se puede entrar por el puerto 8000 (HTTP) pero no funcionaría el micrófono y la cámara.

Comprobar que se puede iniciar una reunión y funciona correctamente.

El archivo de configuración .env se encuentra en:

/var/www/html/appcauldron.com/generatedfiles/.env

En el se indican los puertos que se configuran entre otros ajustes.

Enlaces:

[enlace:<http://appcauldron:8000>]

[enlace:<https://appcauldron:8443>]

Jitsi Meet - Imagen Oficial:

[enlace:<https://github.com/jitsi/docker-jitsi-meet>]

17. Wordpress (Wordpress - Imagen Oficial)

Levanta el servicio db_wordpress que utiliza la última imagen mariadb disponible.

Asocia el puerto 3312 de host con el 3306 del contenedor.

Se establece la password de root asir123, se crea la base de datos appcauldron-db_wordpress, el usuario appcauldron y contraseña asir123.

Se crea el volumen db_wordpress-volume y se asocia a /var/lib/mysql del contenedor.

Levanta el servicio phpmyadmin, que depende de db_wordpress. Utiliza la imagen phpmyadmin.

Asocia el puerto 8301 de host con el 80 del contenedor.

Asocia como contraseña de root asir123 y como host a db_wordpress.

Levanta el servicio wordpress, que depende de db_wordpress, y utiliza la última imagen de wordpress disponible.

Asocia el puerto 8300 de host con el 80 del contenedor.

Establece como host de la base de datos a db_wordpress por el puerto 3306 y como base de datos la creada en mariadb, appcauldron-db_wordpress, y como usuario de la base de datos appcauldron y contraseña asir123.

Asocia /var/www/html/wordpress/wp-content/ de host con /var/www/html/wp-content del contenedor.

Esto nos permite tener en host en la ruta indicada la carpeta wp-content para poder modificarlo, con lo que crearemos la carpeta para que se copie el contenido al levantar el servicio.

17.1. wordpress

```
db_wordpress:
  image: mariadb:latest
  restart: unless-stopped
  ports:
    - '3312:3306'
  environment:
    MYSQL_ROOT_PASSWORD: 'asir123'
    MYSQL_DATABASE: 'appcauldron-db_wordpress'
    MYSQL_USER: 'appcauldron'
    MYSQL_PASSWORD: 'asir123'
  volumes:
    - db_wordpress-volume:/var/lib/mysql
  networks:
    network:

phpmyadmin:
  depends_on:
    - db_wordpress
  image: phpmyadmin/phpmyadmin
  restart: unless-stopped
  ports:
    - '8301:80'
  environment:
    PMA_HOST: 'db_wordpress'
    MYSQL_ROOT_PASSWORD: 'asir123'
  networks:
    network:

wordpress:
  depends_on:
    - db_wordpress
  image: wordpress:latest
  restart: unless-stopped
  ports:
    - '8300:80'
  environment:
    WORDPRESS_DB_HOST: 'db_wordpress:3306'
    WORDPRESS_DB_NAME: 'appcauldron-db_wordpress'
    WORDPRESS_DB_USER: 'appcauldron'
    WORDPRESS_DB_PASSWORD: 'asir123'
  volumes:
    - /var/www/html/wordpress/wp-content/:/var/www/html/wp-content
  networks:
    network:
```

17.2. wordpress.volumes

```
db_wordpress-volume:
```

17.3. wordpress.info

Wordpress (Wordpress - Imagen Oficial)

=====

TEST:

- Acceder a base de datos mysql por el puerto 3312 con usuario y contraseña proporcionados.

Usuario: root

Contraseña: asir123

Usuario: appcauldron

Contraseña: asir123

Tiene que haber creado la base de datos para wordpress:

appcauldron-db_wordpress

Con usuario root:

Prueba de conexión. Mostramos las bases de datos y los usuarios.

mysql -P 3312 -u root --password=asir123 -e 'SHOW DATABASES; SELECT user,host FROM mysql.user;'

Abrir una sesión:

mysql -P 3312 -u root --password=asir123

Con usuario appcauldron:

Prueba de conexión. Mostramos las bases de datos.

mysql -P 3312 -u appcauldron --password=asir123 -e 'SHOW DATABASES;'

Abrir una sesión:

mysql -P 3312 -u appcauldron --password=asir123

- Acceder a phpMyAdmin por el puerto 8301 con usuario y contraseña proporcionados.

Usuario: root

Contraseña: asir123

Usuario: appcauldron

Contraseña: asir123

Tiene que haber creado la base de datos para wordpress:

appcauldron-db_wordpress

[enlace:<http://appcauldron:8301>]

- Acceder a Wordpress por el puerto 8300:

La primera vez nos abrirá:

[enlace:<http://appcauldron:8300/wp-admin/install.php>]

Seleccionamos idioma.

Seleccionamos nombre del sitio, usuario, contraseña (marcamos confirmación si es débil) e ingresamos un correo.

Con el usuario podremos iniciar sesión en:

[enlace:<http://appcauldron:8300/wp-admin/>]

Nuestro Wordpress estará disponible en:

[enlace:<http://appcauldron:8300>]

Se ha mapeado la ruta de wp-content para que pueda modificarse desde local:

/var/www/html/wordpress/wp-content/

Wordpress - Imagen Oficial:

[enlace:https://hub.docker.com/_/wordpress]

18. Office (WPS Office - Imagen linuxserver.io)

Levanta el servicio wps-office que utiliza la última versión de wps-office creada por linuxserver.

Asocia los puertos 3000 y 3001 de host y contenedor.

Asocia la carpeta de Documentos, Escritorio y Configuración (que creamos manualmente) del home del usuario appcauldron con las carpetas Documents, Desktop y config del contenedor.

18.1. office

```
wps-office:  
  image: lscr.io/linuxserver/wps-office:latest  
  container_name: wps-office  
  security_opt:  
    - seccomp:unconfined #optional  
  environment:  
    - PUID=1000  
    - PGID=1000  
    - TZ=Europe/Madrid  
  volumes:  
    - ~/Documentos:/config/Documents  
    - ~/Escritorio:/config/Desktop  
    - ~/Configuracion:/config  
  ports:  
    - '3000:3000'  
    - '3001:3001'  
  shm_size: "1gb"  
  restart: unless-stopped  
  networks:  
    network:
```

18.2. office.info

```
Office (WPS Office - Imagen linuxserver.io)  
=====  
TEST: Acceder a WPS Office por el puerto 3000 (HTTP) o el puerto 3001 (HTTPS).  
Se han mapeado las siguientes carpetas del home del usuario:  
~/Documentos:/config/Documents  
~/Escritorio:/config/Desktop  
~/Configuracion:/config  
Enlaces:  
[enlace:http://appcauldron:3000/]  
[enlace:https://appcauldron:3001/]  
WPS Office - Imagen linuxserver.io:  
[enlace:https://hub.docker.com/r/linuxserver/wps-office]
```

ANEXO VII – SCRIPTS - /var/www/html/appcauldron.com/scripts/

Ejecutados por www-data como usuario appcauldron con el comando de php:

```
shell_exec('sudo -u appcauldron scripts/<script.sh>');
```

19. Ejecución en /var/www/html/appcauldron/cauldron2.php

19.1. magic_cauldron.sh

```
#!/bin/bash
generatedfolder=/var/www/html/appcauldron.com/generatedfiles
templatesfolder=/var/www/html/appcauldron.com/templates
ingredientes=$generatedfolder/ingredientes.txt
servicios=$generatedfolder/servicios.txt
spell=$generatedfolder/spell.yaml
grimoire=$generatedfolder/grimoire.txt
echo "services:" > $spell
rm $grimoire
touch $grimoire
while IFS= read -r ingrediente; do
    if [ "$ingrediente" == "bbdd" ] ; then
        apartadovolumes=1;
        cat $templatesfolder/database >> $spell
        echo "" >> $spell
        cat $templatesfolder/database.info >> $grimoire
        echo "" >> $grimoire
    fi
    if [ "$ingrediente" == "nginx" ] ; then
        cat $templatesfolder/nginx >> $spell
        echo "" >> $spell
        cat $templatesfolder/nginx.info >> $grimoire
        echo "" >> $grimoire
    fi
    if [ "$ingrediente" == "ftp" ] ; then
        cat $templatesfolder/ftp >> $spell
        echo "" >> $spell
        cat $templatesfolder/ftp.info >> $grimoire
        echo "" >> $grimoire
    fi
    if [ "$ingrediente" == "netdata" ] ; then
        apartadovolumes=1;
        cat $templatesfolder/netdata >> $spell
        echo "" >> $spell
        cat $templatesfolder/netdata.info >> $grimoire
        echo "" >> $grimoire
    fi
    if [ "$ingrediente" == "lms" ] ; then
        apartadovolumes=1;
        cat $templatesfolder/lms >> $spell
        echo "" >> $spell
        cat $templatesfolder/lms.info >> $grimoire
        echo "" >> $grimoire
    fi
    if [ "$ingrediente" == "video" ] ; then
        cat $templatesfolder/jitsi >> $spell
        echo "" >> $spell
        cat $templatesfolder/jitsi.info >> $grimoire
        echo "" >> $grimoire
    fi
    if [ "$ingrediente" == "wordpress" ] ; then
        apartadovolumes=1;
        cat $templatesfolder/wordpress >> $spell
        echo "" >> $spell
        cat $templatesfolder/wordpress.info >> $grimoire
```

```

echo "" >> $grimoire
fi
if [ "$ingrediente" == "office" ] ; then
cat $templatesfolder/office >> $spell
echo "" >> $spell
cat $templatesfolder/office.info >> $grimoire
echo "" >> $grimoire
fi
done < $servicios
if [ "$apartadovolumes" == "1" ] ; then
    echo "volumes:" >> $spell
fi
while IFS= read -r ingrediente; do
    if [ "$ingrediente" == "bbdd" ] ; then
        cat $templatesfolder/database.volumes >> $spell
        echo "" >> $spell
    fi
    if [ "$ingrediente" == "netdata" ] ; then
        cat $templatesfolder/netdata.volumes >> $spell
        echo "" >> $spell
    fi
    if [ "$ingrediente" == "lms" ] ; then
        cat $templatesfolder/lms.volumes >> $spell
        echo "" >> $spell
    fi
    if [ "$ingrediente" == "wordpress" ] ; then
        cat $templatesfolder/wordpress.volumes >> $spell
        echo "" >> $spell
    fi
done < $servicios
echo "networks:" >> $spell
echo "    network:" >> $spell
rm $ingredientes
rm $servicios

```

20. Botones /var/www/html/appcauldron.com/spells.php

20.1. cast_spell.sh (Crear hechizo)

```

#!/bin/bash
docker-compose --project-name appcauldron -f /var/www/html/appcauldron.com/generatedfiles/spell.yaml up

```

20.2. stop_spell.sh (Parar hechizo)

```

#!/bin/bash
docker-compose --project-name appcauldron -f /var/www/html/appcauldron.com/generatedfiles/spell.yaml stop

```

20.3. resume_spell.sh (Continuar hechizo)

```

#!/bin/bash
docker-compose --project-name appcauldron -f /var/www/html/appcauldron.com/generatedfiles/spell.yaml start

```

20.4. undo_spell.sh (Deshacer hechizo)

```

#!/bin/bash
docker-compose --project-name appcauldron -f /var/www/html/appcauldron.com/generatedfiles/spell.yaml down

```

21. Ejecución en /var/www/html/appcauldron/commands.php

21.1. docker_ps.sh

```
#!/bin/bash
docker ps
```

21.2. docker_container_ls.sh

```
#!/bin/bash
docker container ls -a
```

21.3. docker-compose_ls.sh

```
#!/bin/bash
docker-compose ls
```

21.4. docker_network_ls.sh

```
#!/bin/bash
docker network ls
```

21.5. docker_volume_ls.sh

```
#!/bin/bash
docker volume ls
```

21.6. docker_images.sh

```
#!/bin/bash
docker images
```

22. Botones /var/www/html/appcauldron.com/commands.php

22.1. delete_stopped_containers.sh (Borrar contenedores parados)

```
#!/bin/bash
docker container ls -f status=exited -aq | xargs docker rm -f
```

22.2. delete_containers.sh (Borrar TODOS los contenedores)

```
#!/bin/bash
docker ps -aq | xargs docker rm -f
```

22.3. delete_active_containers.sh (Borrar contenedores activos)

```
#!/bin/bash
docker ps -q | xargs docker rm -f
```

22.4. delete_anon_networks.sh (Borrar redes no referenciadas)

```
#!/bin/bash
docker network prune -f
```

22.5. delete_anon_vols.sh (Borrar volumenes no referenciados)

```
#!/bin/bash
docker volume prune -f
```

22.6. delete_volumes.sh (Borrar TODOS los volumenes)

```
#!/bin/bash
docker volume ls -q | xargs docker volume rm -f
```

ANEXO VIII – LOGOS - /var/www/html/appcauldron.com/images/

23. Herramientas y proceso para la creación del logo

Como se puede ver en la Ilustración Z, se ha utilizado <https://app.leonardo.ai/> para la creación del logo con el prompt:

Logo. Black and White colors. Cauldron with some ingredients being introduced on it. Main ingredient is an icon with the text "APP" inside. Black and white.

Y el modelo: Leonardo Lightning XL.

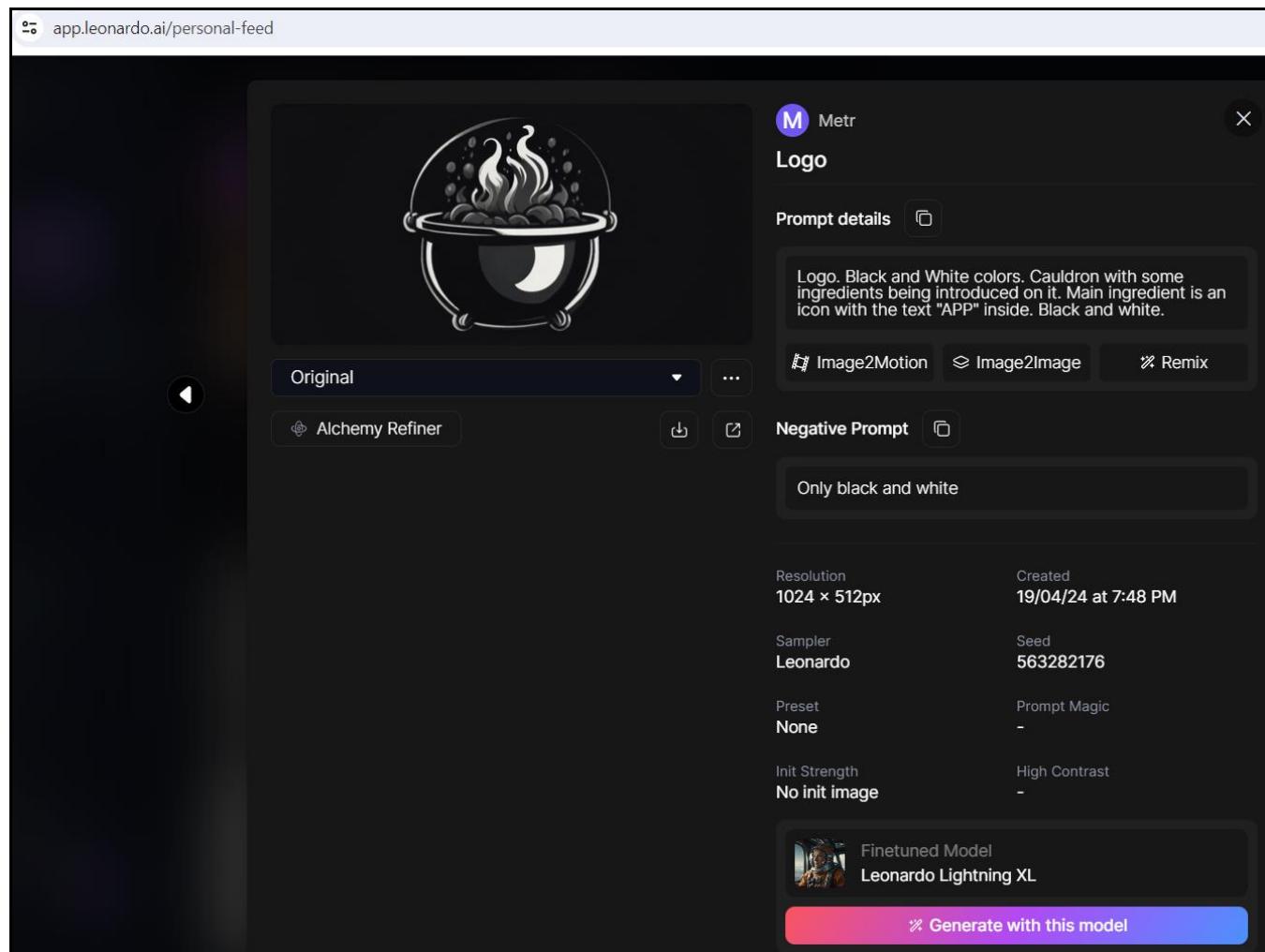


Ilustración Z

Posteriormente con Paint 3D de Windows se ha añadido APP CAULDRON y se han creado los formatos basados en:

https://en.m.wikipedia.org/wiki/File:Standard_web_banner_ad_sizes.svg

23.1. Half banner - 234x60

Transparente:



Ilustración AA

Fondo blanco:



Ilustración BB

23.2. Full banner - 468x60

Transparente:



Ilustración CC

Fondo blanco:



Ilustración DD

23.3. Leaderboard - 728x90

Transparente:



APP CAULDRON

Ilustración EE

Fondo blanco:



APP CAULDRON

Ilustración FF

23.4. 3:1 rectangle - 300x100

Transparente:



APP CAULDRON

Ilustración GG

Fondo blanco:



APP CAULDRON

Ilustración HH

ANEXO IX - CSS - /var/www/html/appcauldron.com/css/

24. appcauldron-style.css

```
body {  
    font-family: Arial, Helvetica, sans-serif;  
}  
.header {  
    padding: 0px; /* some padding */  
    text-align: center; /* center the text */  
    background: #ffffff; /* background color */  
    color: black; /* white text color */  
    background-image: url('../images/Logo_AppCauldron_468x60.png');  
    background-size: contain; /* Adjust the image size to cover the header */  
    width: 468px;  
    height: 60px;  
    background-position: center; /* Center the image */  
    /* Additional properties for background if needed */  
}  
/* Increase the font size of the <h1> element */  
.header h1 {  
    font-size: 40px;  
}  
.p1 {  
    font-family: "Lucida Console", "Courier New", monospace;  
}  
pre {  
    background-color: #f4f4f4;  
    padding: 10px;  
    border: 1px solid #ddd;  
    border-radius: 5px;  
    overflow-x: auto;  
}  
/* Style the top navigation bar */  
.navbar {  
    overflow: hidden; /* Hide overflow */  
    background-color: #333; /* Dark background color */  
}  
/* Style the navigation bar links */  
.navbar a {  
    float: left; /* Make sure that the links stay side-by-side */  
    display: block; /* Change the display to block, for responsive reasons (see below) */  
    color: white; /* White text color */  
    text-align: center; /* Center the text */  
    padding: 14px 20px; /* Add some padding */  
    text-decoration: none; /* Remove underline */  
}  
/* Right-aligned link */  
.navbar a.right {  
    float: right; /* Float a link to the right */  
}  
/* Change color on hover/mouse-over */  
.navbar a:hover {  
    background-color: #ddd; /* Grey background color */  
    color: black; /* Black text color */  
}  
/* Ensure proper sizing */  
* {  
    box-sizing: border-box;  
}  
/* Column container */  
.row {  

```

```
flex: 25%; /* Set the width of the sidebar */
background-color: #a9a9a9; /* Grey background color */
padding: 20px; /* Some padding */
}
/* Main column */
.main {
  flex: 75%; /* Set the width of the main content */
  background-color: #f1f1f1; /* White background color */
  padding: 20px; /* Some padding */
}
/* Responsive layout - when the screen is less than 700px wide, make the two columns stack on top of each other instead of next to each other */
@media screen and (max-width: 700px) {
  .row {
    flex-direction: column;
  }
}
/* Responsive layout - when the screen is less than 400px wide, make the navigation links stack on top of each other instead of next to each other */
@media screen and (max-width: 400px) {
  .navbar a {
    float: none;
    width:100%;
  }
}
.footer {
  padding: 20px; /* Some padding */
  text-align: center; /* Center text*/
  background: #ddd; /* Grey background */
}
```

ANEXO X – OBJETIVOS Y CALENDARIO DE TRABAJO

25. Propuesta de objetivos

Se han elaborado los siguientes grupos de objetivos y calculado su tiempo estimado:

- Entrega de propuesta de trabajo revisada.
- Elaboración de borrador de memoria de proyecto.

Tiempo estimado: 10 horas.

- Generación de máquina virtual con sistema operativo Xubuntu.
- Instalación de apache, docker y docker-compose.

Tiempo estimado: 10 horas.

- Pruebas de funcionamiento servicio apache y Docker.

Tiempo estimado: 20 horas.

- Elaboración de contenedores manuales para cada servicio (servidor web, base de datos, etc.) y/o revisión y selección de los disponibles online.

Tiempo estimado: 20 horas.

- Confección de formulario web para selección de servicios.

Tiempo estimado: 15 horas.

- Generar archivo modular basado en la selección del formulario para docker (fichero yaml o Dockerfile).

- Revisión funcional.

Tiempo estimado: 15 horas.

- Integración total del proyecto.

- Pruebas funcionales.

Tiempo estimado: 20 horas.

- Revisión.

- Estudio e implementación de mejoras.

Tiempo estimado: 20 horas.

- Creación de presentación en PowerPoint.

- Entrega de proyecto.

Tiempo estimado: 10 horas.

26. Propuesta calendario de trabajo

Conforme a los grupos de objetivos indicados en el apartado Propuesta de objetivos, se ha elaborado el siguiente calendario de trabajo ilustrado en la Tabla A:

Tabla A