

Tarea 1: Plataformas de Programación Web en el Entorno Servidor. Aplicaciones LAMP

Francisco Javier Sueza Rodríguez

27 de noviembre de 2023

Centro:	IES Aguadulce
Ciclo Formativo:	Desarrollo Aplicaciones Web (Distancia)
Asignatura:	Desarrollo Web en Entorno Servidor
Tema:	Tema 1 - Plataformas de Programación Web en el Entorno Servidor.

1. Caso Práctico

La asociación **Respira** para la atención de menores en situación de riesgo de exclusión social ha recibido una pequeña subvención para el desarrollo de estrategias digitales en el ámbito de su actividad. La asociación tiene su sede central en un barrio conflictivo donde reciben multitud de casos, principalmente de menores y familias en situaciones muy complejas, a los que intentan ayudar con los pocos medios que tienen.

Juan y **María** han recibido el proyecto con los brazos abiertos y están dispuestos a dar el todo por el todo para que esta asociación ayude a un mayor número de personas. **María** está especialmente motivada, dado que es el barrio donde ella se crío cuando era pequeña.

2. Ejercicios

2.1. Ejercicio 1

¿Qué tipo de proyectos podría hacer la asociación **Respira** con el dinero recibido? Actualmente no tiene web, ¿se te ocurre como podrían mejorar su actividad y su proyección?

2.1.1. Solución

Para mejorar su actividad y proyección podrían emplear la subvención recibida para varios proyectos. El principal sería la **creación de una página web**, que va a ser en el que nos centremos en este ejercicio, pero también podría incluirse la **creación de puestos informáticos en el centro**, para permitir a los jóvenes que no tengan recursos tener acceso a internet, incluida a la propia página web.

Respecto a la creación de la **página web**, se podría contratar a un desarrollador web para que creara la aplicación web necesaria. La página, además de la **información relativa** a todos los **servicios** que ofrece la asociación, podría tener diversos **canales de comunicación**.

Por ejemplo, se podría implementar un **chatbot** para responder a las preguntas frecuentes de forma más inmediata. Además, sería interesante la creación de un **foro** donde los usuarios pudieran exponer sus dudas sobre cualquier tema relacionado con la asociación, para que estas se pudieran responder y debatir. Siendo además un buen canal para recibir feedback de los usuarios y mejorar, en la medida de lo posible, el funcionamiento de la asociación.

También debería incluir varias **secciones con información** sobre la asociación así como las diferentes actividades que se realicen, siendo interesante la inclusión de un **calendario** donde se pueda acceder a las actividades previstas de forma sencilla y rápida.

Esta web proporcionaría **más visibilidad**, **mejorando la proyección** de la asociación y permitiendo que los usuarios estén más implicados en esta.

Además de esta web, sería interesante añadir una serie de **puestos informáticos**, si el presupuesto lo permite, donde además de acceder a la propia web, permitiera a los usuarios con menos recursos acceder a internet para realizar diferentes gestiones, incluso podría valorarse la creación de algún curso online de inicialización a las nuevas tecnologías, uso de editores de texto, creación de CV, etc...

2.2. Ejercicio 2

Fíjate en el siguiente ejemplo que describe la interacción entre navegador (cliente web) y servidor web donde se detalla lo que ocurre en cada una de las partes. Intenta pensar en otro “Caso de uso” donde

un potencial usuario (o miembro de la organización) utilice la página web de la asociación Respira para una acción concreta y detalla lo que ocurriría paso por paso.

Caso de Uso: Un voluntario rellena un formulario

1. La persona voluntaria está visualizando un formulario en la página web de la asociación y ha terminado de rellenarlo, con lo que hace clic en “enviar”.
2. El navegador realiza una petición tipo HTTP POST al servidor web.
3. El servidor web recibe la petición con los datos del formulario, y, siguiendo su configuración, detecta que esa petición corresponde a un script PHP y que debe ser redireccionada al motor de ejecución PHP.
4. El motor PHP (PHP engine) se arranca e inicia la ejecución del script el cual contiene:
 - a) Código para comprobar si los datos recibidos son correctos.
 - b) Código para almacenar los datos recibidos en la base de datos.
 - c) Código para generar HTML con la respuesta al candidato a la candidata.
5. Una vez que el motor PHP ha terminado de ejecutar el script, el HTML generado es enviado al servidor web como resultado de la ejecución.
6. El servidor web recoge el resultado generado por el motor PHP y lo envía al navegador con una respuesta HTTP.
7. El navegador HTTP recibe la respuesta que incluye el HTML y lo renderiza para que la persona candidata lo vea.

2.2.1. Solución

El caso de uso que vamos a proponer es el de un usuario que **realiza una búsqueda en el foro** para que se muestren los mensajes o hilos con una determinada palabra.

Caso de Uso: Búsqueda de mensajes en el foro

1. El usuario que esta en la página relativa a los foros, introduce un termino en la caja de búsqueda y pulsa en **Enviar**.
2. El navegador realiza una **petición tipo HTTP POST o GET**, dependiendo de como se haya programado la página, al servidor.
3. El **servidor web** recibe la petición con los datos de la búsqueda, y pasa la petición al **motor PHP** para que se ejecute el script adecuado.
4. El **motor PHP** se inicializa e inicia la ejecución del script el cual contiene:
 - a) Código para realizar una **llamada a la base de datos** y obtener todos los mensajes que contengan el termino de búsqueda.
 - b) Código para **filtrar y ordenar el resultado**, para que se muestre de una forma más adecuada al usuario.
 - c) Código para **generar HTML** y que el resultado se muestre de forma apropiada en la web.
5. Una vez que se a **procesado el script**, el motor PHP pasa el resultado al servidor web, para que este pueda servirlo al cliente.
6. El **servidor web** procesa el resultado del motor PHP y lo manda al cliente web.
7. El **cliente** recibe el resultado y lo renderiza.

2.3. Ejercicio 3

Rellena la siguiente tabla indicando casos concretos en los que podría ser necesario emplear las siguientes tecnologías en el desarrollo de la web de la asociación **Respira**:

Tecnología	¿Dónde y como se usaría?
HTTP	
HTTPS	
HTML	
PHP	
CSS	
SQL	
JavaScript	

2.3.1. Solución

A continuación se muestra la tabla rellena con la información solicitada.

Tecnología	¿Dónde y Como se usaría?
HTTP	Se usaría en todas las peticiones que se hicieran al servidor, usando métodos POST o GET. Estas peticiones se realizarían desde un script PHP o desde el cliente con Javascript.
HTTPS	Este protocolo se debería emplear para que la comunicación cliente/servidor fuera más segura. Se debe configurar el servidor para que se use este protocolo en vez de HTTP.
HTML	Se usaría para estructurar la página web, usando código HTML en los archivos de las diferentes páginas que la formen o generándolo en scripts PHP o Javascript.
PHP	Es lenguaje se usaría en diferentes script para manejar la lógica de la aplicación web en la parte del servidor. Se incluirían diferentes script para manejar la peticiones desde el cliente y pasarlas al servidor.
CSS	CSS se usaría para dar estilo a las diferentes páginas web. Se podría incluir junto con el código HTML pero es más recomendable incluir archivos independientes CSS que se puedan aplicar a varias páginas.
SQL	Con SQL realizaríamos las diferentes peticiones de consulta, escritura y borrado a la base de datos. Se usaría tanto en peticiones desde script PHP como en el gestos de bases de datos.
Javascript	Este lenguaje de script se usaría en la parte del cliente para proporcionar interactividad a la página web y gestionar y procesar las peticiones de usuarios para pasarlas a la parte del servidor. Así mismo, se encargaría de gestionar las respuesta del servidor al cliente y generar el HTML deseado.

Realiza la instalación de XAMPP 8.2.4 en tu ordenador (preferiblemente en C:/XAMPP) y realiza lo siguiente:

- NOTA:** hay un error en la numeración de los ejercicios en la página de la tarea, pasando de ejercicio 3 al 5, por lo que en este documento se van a numerar omitiendo ese error, este sería el ejercicio 5 de la descripción de la tarea.

En este ejercicio se debía realizar la instalación de XAMPP, pero a riesgo de que el ejercicio puntúe como no realizado, **no he instalado** dicha versión de AMP, ya que durante el proceso de estudio de la unidad 1 **se instaló LAMP** en la distribución Linux que uso, una Kubuntu en concreto.

Así, en lugar de mostrar capturas del proceso de instalación de XAMPP, se mostrarán de los paquetes ya instalados y de los comandos usados para arrancar y detener el servidor web, que se realizan usando `systemctl`.

```
~: bash — Konsole

{cosueza in Sulaco in ~}
> apt show apache2
Package: apache2
Version: 2.4.52-1ubuntu4.7
Priority: optional
Section: web
Origin: Ubuntu
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Original-Maintainer: Debian Apache Maintainers <debian-apache@lists.debian.org>
Bugs: https://bugs.launchpad.net/ubuntu/+filebug
Installed-Size: 546 kB
Provides: httpd, httpd-cgi
Pre-Depends: init-system-helpers (≥ 1.54~)
Depends: apache2-bin (= 2.4.52-1ubuntu4.7), apache2-data (= 2.4.52-1ubuntu4.7), lib-base, mime-support, perl, perl-modules-5.38
Recommends: ssl-cert
Suggests: apache2-doc, apache2-suexec-pristine | apache2-suexec-custom, www-browser, ufw
Conflicts: apache2.2-bin, apache2.2-common
Replaces: apache2.2-bin, apache2.2-common
Homepage: https://httpd.apache.org/
Task: lamp-server
Download-Size: 97,8 kB
APT-Manual-Installed: no
APT-Sources: http://es.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages
Description: servidor HTTP Apache
El objetivo del proyecto servidor HTTP Apache es construir un servidor
HTTP seguro, eficiente y extensible como software de código abierto
compatible con los estándares. El resultado ha sido durante mucho tiempo
el servidor web número uno en Internet.

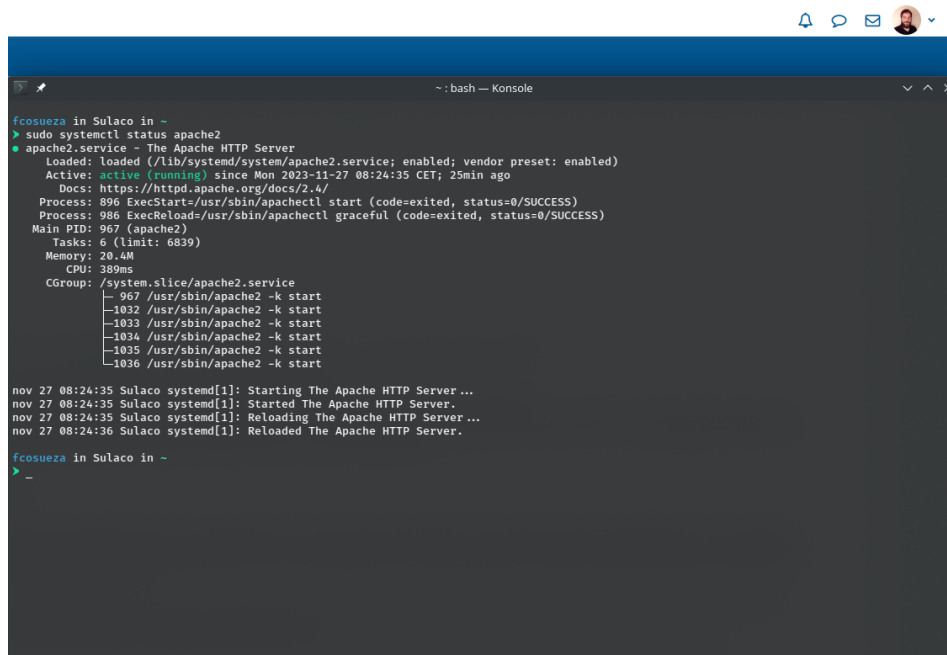
El resultado de la instalación de este paquete es una instalación
completa, incluyendo los archivos de configuración, los scripts de inicio
y los scripts de asistencia.

B: Hay 1 registro adicional. Utilice la opción -a para verlo.

{cosueza in Sulaco in ~}
>
```

5

Para **arrancar el servidor Web**, o bien reiniciarlo, se usan el comando **systemctl**. En la siguiente captura se muestra la salida del estado del servidor con este comando.



```

fcosueza in Sulaco in ~
> 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 Mon 2023-11-27 08:24:35 CET; 25min ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 896 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Process: 986 ExecReload=/usr/sbin/apachectl graceful (code=exited, status=0/SUCCESS)
   Main PID: 967 (apache2)
      Tasks: 6 (limit: 6839)
     Memory: 20.4M
        CPU: 389ms
   CGroup: /system.slice/apache2.service
           └─ 967 /usr/sbin/apache2 -k start
             1032 /usr/sbin/apache2 -k start
             1033 /usr/sbin/apache2 -k start
             1034 /usr/sbin/apache2 -k start
             1035 /usr/sbin/apache2 -k start
             1036 /usr/sbin/apache2 -k start

nov 27 08:24:35 Sulaco systemd[1]: Starting The Apache HTTP Server ...
nov 27 08:24:35 Sulaco systemd[1]: Started The Apache HTTP Server.
nov 27 08:24:35 Sulaco systemd[1]: Reloading The Apache HTTP Server ...
nov 27 08:24:36 Sulaco systemd[1]: Reloaded The Apache HTTP Server.

fcosueza in Sulaco in ~
>

```

Figura 2.2: Uso de systemctl para gestión del servidor Web

La carpeta por defecto de nuestro sitio web en Linux se encuentra en **/var/www/html**. Se ha añadido el script pedido y en la siguiente captura podemos ver su carga en el navegador web Firefox.

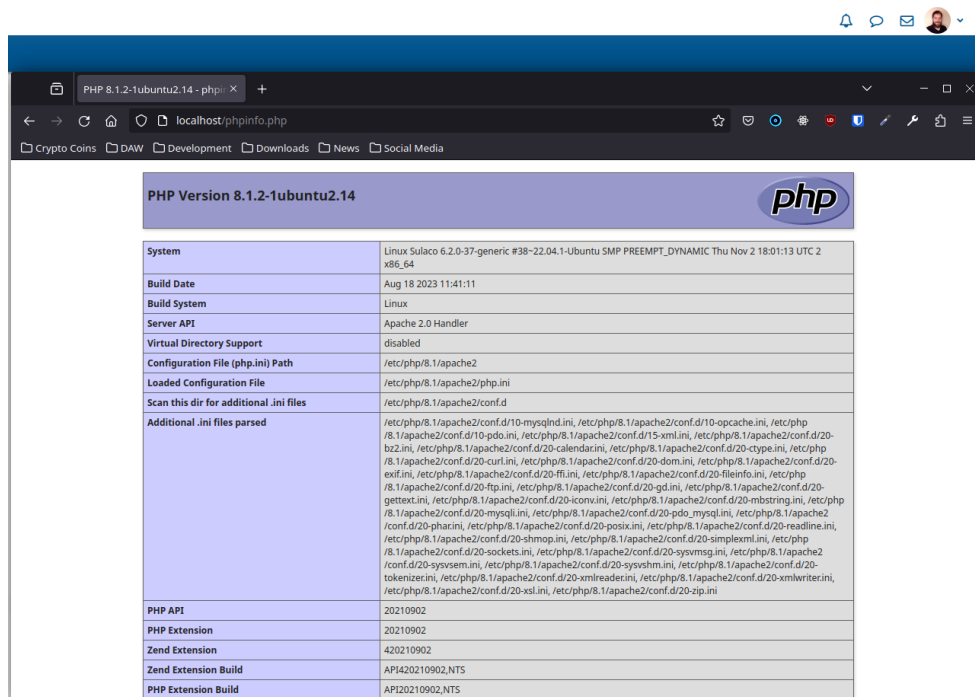


Figura 2.3: Carga del script phpinfo.php

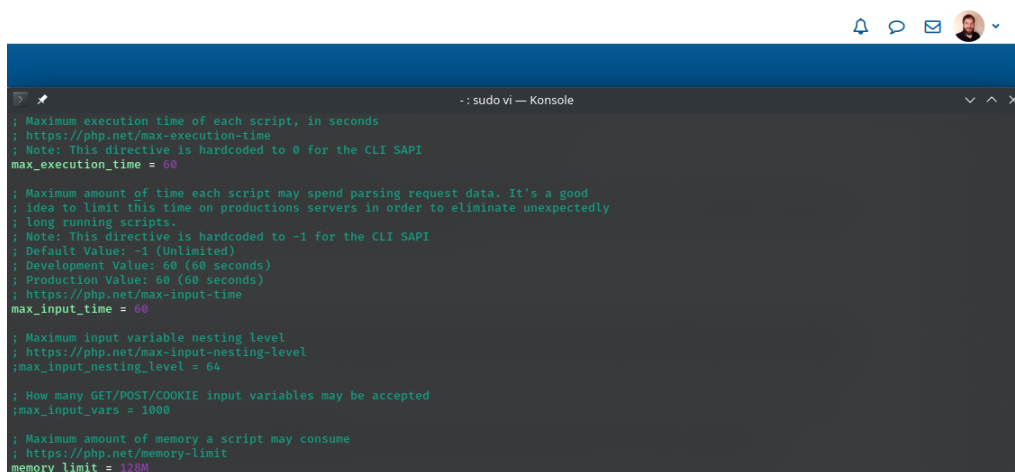
2.5. Ejercicio 5

Localiza el archivo `php.ini` (generalmente en `c:/xampp/php/php.ini`). Este archivo contiene un montón de directivas que determinan como trabaja el motor PHP cuando es invocado por el servidor Apache. La configuración de PHP incluida por defecto viene establecida para un escenario de desarrollo (cuando la aplicación pasa a producción se pone una configuración más segura y fiable). El objetivo ahora es modificar dos directivas que afectarán al funcionamiento de PHP:

- Modifica la directiva `max_execution_time` que determina el tiempo máximo que un script PHP puede ejecutarse, cambiando el valor a 60 segundos. Realiza una captura de pantalla de la modificación.
- Modifica la directiva modificar los errores reportados por el usuario de forma que se quede así: `error_reporting=E_ALL & E_STRICT`.

2.5.1. Solución

En este caso, el archivo **php.ini** se encuentra en la ubicación `/etc/php/8.1/apache2/php.ini`. Se ha abierto el archivo y se han cambiado las dos directivas que se piden, como podemos ver en las siguientes capturas de pantalla.



```
--: sudo vi -- Konsole
; Maximum execution time of each script, in seconds
; https://php.net/max-execution-time
; Note: This directive is hardcoded to 0 for the CLI SAPI
max_execution_time = 60

; Maximum amount of time each script may spend parsing request data. It's a good
; idea to limit this time on production servers in order to eliminate unexpectdly
; long running scripts.
; Note: This directive is hardcoded to -1 for the CLI SAPI
; Default Value: -1 (Unlimited)
; Development Value: 60 (60 seconds)
; Production Value: 60 (60 seconds)
; https://php.net/max-input-time
max_input_time = 60

; Maximum input variable nesting level
; https://php.net/max-input-nesting-level
;max_input_nesting_level = 64

; How many GET/POST/COOKIE input variables may be accepted
;max_input_vars = 1000

; Maximum amount of memory a script may consume
; https://php.net/memory-limit
memory_limit = 128M
```

Figura 2.4: Cambio de la directiva `max_execution_time`



```
--: sudo vi -- Konsole
; Default Value: On
; Development Value: On
; Production Value: Off

error_reporting = E_ALL & ~E_DEPRECATED & ~E_STRICT

; log_errors
; Default Value: Off
; Development Value: On
; Production Value: On

; max_input_time
; Default Value: -1 (Unlimited)
; Development Value: 60 (60 seconds)
; Production Value: 60 (60 seconds)

; output_buffering
; Default Value: Off
; Development Value: 4096
; Production Value: 4096
```

Figura 2.5: Cambio de la directiva `error_reporting`