	UNIVERSIDAD DON BOSCO **FACULTAD DE INGENIERÍA** **ESCUELA DE COMPUTACIÓN**
CICLO: 01/2020	*GUIA DE LABORATORIO #7* **Nombre de la Practica:** **Gestión de archivos y directorios con PHP** **Lugar de Ejecución:** **Centro de Cómputo** **Tiempo Estimado:** **2 horas con 30 minutos** **MATERIA:** **Desarrollo de Aplicaciones Web con Software Interpretado en el Servidor** **DOCENTES:** **Ing. Ricardo Elías / Lic. Felipe Acosta / Ing. Saul Díaz**

I. OBJETIVOS

En esta guía el alumno conocerá y aprenderá a:

1. Comprender la importancia de la utilización de los archivos en aplicaciones web del lado del servidor.
2. Dominar las distintas funciones proporcionadas por el lenguaje PHP para la gestión de directorios y archivos.
3. Adquirir habilidad en el manejo de archivos csv para gestionar información de forma organizada.
4. Lograr la habilidad necesaria para subir archivos al servidor desde el cliente realizando verificaciones de seguridad antes de procesar los datos.

II. INTRODUCCION TEORICA

Definición de archivo

Los **archivos** son una **estructura de datos que se almacena en memoria secundaria** y en la cual es posible **guardar información** de **forma permanente**. Dentro del sistema de archivos del servidor los **archivos** son **organizados** en **directorios o carpetas**. Los directorios son un tipo especial de archivo creado para almacenar otros archivos. También pueden crearse jerárquicamente dentro de otros directorios, comenzando por el **directorio raíz**, conocido como **nivel superior**.

En los archivos se puede almacenar cualquier tipo de datos y también se puede almacenar mucha más información que la que puede soportar la memoria RAM. PHP facilita el trabajo con el sistema de archivos del servidor al incluir funciones que le permitan obtener información sobre los archivos, así como abrirlos, leerlos y escribir en ellos.

Trabajar con archivos en PHP

Al igual que muchos otros lenguajes de programación PHP proporciona una amplia gama de funciones para acceder al sistema de archivos del servidor con las cuales es posible desarrollar las operaciones típicas que se pueden realizar con archivos, como la **lectura** y la **escritura**, así como el **manejo de permisos**. Vamos a examinar las operaciones con archivos desde las más básicas hasta las más avanzadas, siendo estas:

a) Abrir un archivo.
b) Cerrar un archivo.
c) Leer un archivo.
d) Recorrer un archivo.
e) Escribir en un archivo.

Abrir un archivo

Prácticamente la totalidad de las funciones del sistema de archivos carecen de operatividad sin la función fopen(), que es la función específica de PHP para abrir un archivo y que permite posteriormente, operaciones típicas como la lectura y la escritura sobre éste.

La sintaxis de la función **fopen()** es la siguiente:

```
resource fopen(string $filename, string $mode[, bool $use_include_path=false[, resource $context]]);
```

Donde, `$filename` es el nombre del archivo en disco que se desea abrir; `$mode`, representa el modo en el que la función lo abrirá (vea el detalle de los modos de apertura en la tabla que se muestra abajo); `$use_include_path`, puede ser usado si desea que PHP busque el archivo en la ruta definida en la directiva `include_path` del archivo de configuración php.ini. Hay que apuntar que el primer argumento, `$filename`, puede ser un nombre de archivo, una ruta local o una URL remota hacia el archivo. En el caso de tratarse del último caso, debe considerar que la directiva `allow_url_fopen` del php.ini debe estar habilitada para que PHP pueda abrir el archivo.

Con respecto al segundo argumento `$mode`, cabe mencionar que en PHP los archivos pueden ser abiertos en diversos modos. La siguiente tabla resume todos ellos:

Modo de apertura	Descripción
r	Abre el archivo en modo sólo lectura.
r+	Abre el archivo en modo lectura y escritura.
w	Abre el archivo en modo sólo escritura. Si el arhivo no existe, se crea, y si ya existe, se sobrescribe.
w+	Abre el archivo en modo lectura y escritura. Si el archivo no existe, se crea, y si ya existe, se sobrescribe.
a	Abre el archivo en modo escritura. Si el archivo no existe, se crea, y si existe, le añade la nueva información.
a+	Abre el archivo en modo lectura y escritura. Si el archivo no existe, se crea, y si ya existe, la añade la nueva información.
x	
b	Abre el archivo en modo binario para lectura y escritura. Su aplicación es exclusiva para sistemas Windows, en donde se hace distinción entre archivos de texto y archivo binarios.

NOTA: A todos estos modos de apertura de archivos se les puede añadir las letras b o t, dependiendo de si el archivo es binario o de texto. Esto puede ser útil en sistemas en los que se distingue entre archivos binarios y de texto, como en Windows.

La función fopen() devuelve un recurso en caso de tener éxito, este recurso es un **descriptor de archivo** o un **puntero** o **referencia** a éste. Es recomendable asignar este valor devuelto a una variable, que a partir de ese momento se convertirá en la **referencia al archivo abierto** y con la cual deberá realizar las operaciones pertinentes de lectura y escritura. En caso de producirse algún error al intentar abrir el archivo devolverá False. Si la apertura del archivo falla, se producirá un error de nivel E_WARNING, que puede ser ignorado haciendo uso del operador @.

Ejemplo:

```
//Abrir archivo en modo solo lectura
$fr = fopen("myfile.txt", 'r');

//Abrir archive en modo binario para lectura y escritura
$fb = fopen("myfile.txt", 'ba+');

//Abrir archivo en modo lectura y escritura, incluyendo la ruta definida en include_path
$fw = fopen("myfile.txt", 'w+', true);
```

Cerrar un archivo
Siempre que se haya terminado de utilizar un archivo es recomendable cerrar las referencias que se hayan hecho sobre este a través de la función fopen(). Para tal efecto, debe utilizar la función fclose(), cuya sintaxis se muestra a continuación:

```
bool fclose(resource $handle);
```

La función fclose() recibe un descriptor de archivo como argumento y devuelve un valor booleano que será True, en caso de tener éxito o False en caso de producirse algún error.
Ejemplo:

```
$fr = fopen("listado.txt", 'w');
//Cerrar el archivo
fclose($fr);
```

Leer archivos
PHP proporciona muchas formas diferentes de leer un archivo y para cada una de estas formas se dispone de una o varias funciones que facilitan la tarea.

Leer un archivo carácter por carácter
Para realizar una lectura de archivo carácter a carácter se utiliza la función fgetc() que lee y devuelve un carácter a la vez de un archivo abierto. Devuelve falso en caso de alcanzar el final del archivo.
La sintaxis de la función es la siguiente:

```
string fgetc(resource $handle);
```

La función fgetc() recibe un descriptor de archivo o un apuntador a éste y devuelve un solo carácter del archivo en forma de cadena de texto. Es preciso, si se va a leer todo el contenido de un archivo, utilizar esta función en conjunto con un ciclo repetitivo para acceder a todos los caracteres del archivo.

La sintaxis de la función fgetc() es la siguiente:

```
$filename = $_SERVER[DOCUMENT_ROOT] . "/examples/datos.dat";
$fh = fopen($filename, 'r');
while(!feof($fh){
   $car = fgetc($fh);
   if($car == '\n'){
      $car = "<br />";
   }
   print $car;
}
fclose($fh);
```

Leer archivos línea por línea
Existen dos funciones de PHP que permiten realizar la lectura de un archivo línea a línea, estas funciones son: fgets() y fgetss(). La diferencia que existe entre una y otra función es que la primera está diseñada para leer texto y la segunda para leer archivos html, de los que desea excluirse las etiquetas HTML.
La sintaxis de la función fgets() es la siguiente:

```
string fgets(resource $handle[, int $length]);
```

Donde, `$handle` es el manejador o descriptor de archive del que se va a leer líneas y `$length` es un argumento opcional que se puede utilizar para indicar un número específico de bytes que se van a leer del archivo. La lectura puede terminar por una de tres razones:

a) Se han leído ya $length bytes menos 1 ($length-1), en caso de que se haya proporcionado el segundo argumento,
b) Se ha leído un carácter de fin de línea ('\n'), o
c) Se ha llegado al final del archivo (EOF: End Of File).

Lo que suceda primero.

Ejemplo:
```
$filename = $_SERVER[DOCUMENT_ROOT] . "/examples/datos.dat";
$fh = fopen($filename);
while(!feof($fh)){
   $line = fgets($fh);
   print $line;
}
fclose($fh);
```

La sintaxis de la función fgetss() es la siguiente:
```
string fgetss(resource $handle[, int $length[, string $allowable_tags]]);
```

Esta función es en la práctica, igual que la función fgets(), la diferencia está en que de forma predeterminada elimina las etiquetes HTML en la operación de lectura del archivo. Si se desean conservar etiquetas HTML en la lectura, debe proporcionar en el tercer argumento la lista de etiquetas que no deberán eliminarse en la cadena devuelta por la función.

Ejemplo:
```
$filename = "pagina.html";
$fh = fopen($filename, 'r');
while(!feof($fh)){
   $content = fgetss($fh);
   echo $content;
}
fclose($fh);
```

Leer archivos una cantidad específica de bytes

Para leer de un archivo una cantidad específica de bytes puede hacer uso de la función fread(). Esta función trata a cualquier archivo que reciba como argumento como una secuencia de bytes, sin tener en cuenta finales de línea o cualquier otro tipo de carácter especial. En sistemas operativos de servidor en los que se diferencie entre archivos binarios y de texto, como Windows, el archivo debería ser abierto en modo binario ('b') al usar la función fopen().

Si se va a leer el archivo completo, es aconsejable utilizar la función filesize() para obtener el número de bytes (o caracteres) del archivo y proporcionar el valor devuelto por esta función como argumento en la función fread(). La sintaxis de la función fread() es la siguiente:
```
string fread(resource $handle, int $length);
```

Donde, el primer argument `$handle` es el descriptor o manejador de archive y el segundo argumento, `$length`, opcional, es el número de bytes que se van a leer del archivo. La función fread() devuelve los bytes leídos o el valor False en caso de que se haya producido algún error.

La lectura puede terminar por la primera de las tres condiciones siguientes que se llegue a producir:

a) Se han leídos los $length bytes,
b) Se ha alcanzado el final del archivo (EOF), o
c) Un paquete se encuentra disponible o el tiempo límite del socket se agota (esto aplica para flujos de red).

Ejemplo:
```
$filename = "c:/wamp/www/examples/data.dat"; //Funcionará solo en Windows
$fr = fopen($filename, 'rt');
$contents = fread($fr, filesize($filename));
print "<pre>$contents</pre>";
fclose($fr);
```

En este otro ejemplo se ilustra cómo puede utilizarse la función fread() para leer un archivo de imagen:
```
$filename = "c:\\wamp\\www\\examples\\images\\tulips.jpg";
# $filename = "c:/wamp/www/examples/images/tulips.jpg";
```

```
$fh = fopen($filename, "rb");
$contents = fread($fh, filesize($filename));
header("Content-type: image/jpeg"); //Para indicar al navegador que lo que se va a mostrar es una imagen
echo $content;
fclose($fh);
```

Acceso aleatorio a los datos del archivo

Hasta ahora, la forma en que se ha accedido a los archivo, ha sido de forma secuencial, esto es leer los datos del archivo en el orden en que están dispuestos. Sin embargo, en la práctica puede encontrarse con la necesidad de acceder a un dato específico que no necesariamente esté al inicio del archivo. Existe la posibilidad de mover el indicador de posición en el archivo de modo que se pueda comenzar a leer o a escribir en cualquier punto del archivo. A esto se le llama también recorrer un archivo.

Las funciones de PHP que se utilizan para este propósito son las siguientes:

a) fseek(): Desplaza el indicador de posición a un punto específico en el archivo.
b) rewind(): Mueve el indicador de posición al inicio del archivo.
c) ftell(): Devuelve la posición actual del indicador de posición del archivo.

La sintaxis de la función fseek() es la siguiente:

```
int fseek(resource $handle, int $offset[, int $whence=SEEK_SET]);
```

En donde, `$handle` es el manejador de archive, `$offset` establece el desplazamiento a partir de la posición indicada por `$whence`, cuando se proporcione, si no se proporciona, el desplazamiento dentro del archivo se hará a partir del inicio del archivo. El valor del tercer argumento, `$whence`, puede ser:

a) SEEK_SET: Establece el indicador de posición al principio del archivo, más el desplazamiento indicado en el segundo argumento de la función fseek(), `$offset`.
b) SEEK_CUR: Establece el indicador de posición en la posición actual dentro del archivo abierto más el desplazamiento especificado en `$offset`.
c) SEEK_END: Establece el indicador de posición al final del archivo, más el desplazamiento establecido en el argumento `$offset`. Se utiliza con desplazamientos negativos para recorrer el archivo en forma inversa.

Los valores devueltos por fseek() serán 0 si el indicador se ha posicionado correctamente, o -1 se ocurrió algún problema.

La función rewind() se utiliza para regresar el indicador de posición dentro del archivo al inicio. Su sintaxis es la siguiente:

```
bool rewind(resource $handle);
```

Donde, `$handle` es el descriptor o manejador de archivo, y el valor devuelto por la función es un valor booleano que será `True`, si se hace la operación con éxito, o `False`, si se produce algún error.

La función ftell() se puede utilizar para una de tres cosas, obtener la posición en bytes actual del indicador de posición en un archivo sobre el que se ha realizado una operación de lectura, el número de bytes desde el principio del archivo en un determinado momento o la posición, en bytes, desde dónde comenzará la siguiente operación de lectura. La sintaxis de la función ftell() es la siguiente:

```
int ftell(resource $handle);
```

La función recibe como argumento el descriptor o manejador de archivo y devuelve un valor entero con la posición del puntero al archivo referenciado. Si se produce algún error, devolverá False.

Trabajar con directorios

También es posible trabajar con directorios en PHP de la misma forma que con archivos, para lo cual se proporcionan una gran variedad de funciones equivalentes. Algunas de estas funciones utilizan un indicador de directorio, mientras que otras utilizan una cadena que contiene el nombre del directorio con el que se va a trabajar.
Un indicador de directorio es una variable especial que apunta a un directorio. Para obtener el indicador de directorio se puede utilizar la función opendir().

```
$dh = opendir("/home/files");  //Funcionará en sistemas Unix/Linux
```

Al igual que con las operaciones con archivos, también existe una función para cerrar un directorio. Esa función es closedir(), que recibe como argumento el indicador de directorio.

```
closedir($dh);
```

Además de estas funciones está la función readdir() que se utiliza para obtener el nombre del siguiente archivo leído desde el directorio abierto con opendir(). Los nombres de archivo son devueltos en el orden en que son almacenados por el sistema de archivos. Cada directorio contiene una lista de entradas para cada uno de los archivos y subdirectorios que posee, así como dos entradas especiales que son: ., que representa el directorio y .., que representa el padre del directorio. PHP mantiene un puntero interno que hace referencia a la siguiente entrada de la lista, del mismo modo que un puntero de archivo apunta a la posición en un archivo donde debería ocurrir la siguiente operación de archivo.

Ejemplo:

```
$dirpath = "/home/files/images";
if(!$handle = opendir($dirpath)) die("No se puede abrir el directorio");
echo $dirpath . " contiene los siguientes directorios y archivos: <br />";
echo "<ul>";
while($file = readdir($handle)){
   if($file != "." && $file != "..") echo "<li>$file</li>";
}
echo "</ul>";
closedir($handle);
```

Subir archivos al servidor web

Esta es una tarea muy usual en los formularios web. Existen muchos sitios web que permiten a los usuarios subir sus archivos al servidor desde el navegador haciendo uso de una interfaz de usuario que forzosamente deberá incluir un campo de formulario del tipo file (`<input type="file" … />`). Ejemplos prácticos de esto los puede encontrar en sitios web que le permiten subir fotos, su hoja de vida (curriculum vitae), música, etc. También lo puede encontrar en los sitios de correo electrónico web como Hotmail, Yahoo y Gmail, entre otros.
La tarea de crear una aplicación que le permita subir archivos al servidor debe dividirse en dos partes:

1. La página web que contendrá el formulario con el campo de formulario que le permitirá al usuario seleccionar el archivo que desea subir al servidor.
2. La programación PHP necesaria para procesar el archivo que ya ha sido transferido al servidor. En este punto habrá que acceder a dicho archivo para realizar las tareas que se consideren oportunas.

Creación del formulario para subir un archivo al servidor

En la página HTML necesita básicamente, un formulario que incluya una etiqueta input de tipo file. Esto se consigue con un código tan simple como el siguiente:

```
<form action="procesararchivo.php" method="POST" enctype="multipart/form-data">
 <!-- El valor establecido para el atributo value del input type="hidden" no debe ser superior al
 indicado en la    directiva upload_max_filesize del php.ini -->
   <input type="hidden" name="max_file_size" value="250000" />
   <label for="adjunto">Archivo a adjuntar:</label>
   <input type="file" name="adjunto" id="adjunto" /><br />
   <input type="submit" name="enviar" value="Subir" />
</form>
```

Note que en el elemento form que crea el formulario se ha utilizado el atributo enctype con el valor especial: multipart/form-data. Esto es requerido para que funcione correctamente la operación de subir el archivo al servidor web. Opcionalmente, se puede utilizar un campo oculto con el que se indicará el tamaño máximo del archivo a ser enviado a través del formulario. En este campo oculto se sugiere utilizar como nombre del campo el nombre max_file_size. Si no se hace esto se tomará de referencia el nombre de la directiva upload_max_filesize definida en el archivo php.ini.
Al pulsar el botón enviar del formulario el archivo seleccionado por el usuario será enviado al servidor y, como es de esperar, el archivo no es enviado a un directorio o carpeta al azar, ni tampoco al directorio donde está el script con el formulario. El archivo enviado es almacenado en un directorio indicado por la directiva upload_tmp_dir establecida en el archivo de configuración php.ini.
Para realizar las tareas de procesamiento en el servidor, una vez que el archivo está almacenado en el servidor, PHP proporciona la matriz asociativa $_FILES, que consta de dos dimensiones. En la primera columna de la matriz se utilizan como claves los identificadores (name) de los elementos input definidos en el formulario. En la segunda columna se utilizan claves especiales y predefinidas que se corresponden con las características de cada archivo. La siguiente tabla muestra los nombres de estas claves:

Clave	Descripción
name	**Nombre original del archivo transmitido.**
tmp_name	**Nombre temporal del archivo que se establece en el momento que se termina de transferir el archivo al servidor. Si el archivo no se hubiera podido transmitir, por ejemplo, porque es demasiado grande en comparación con la directiva upload_max_filesize del php.ini, entonces obtendremos el valor none.**
size	**Tamaño en caracteres (bytes) del archivo.**
type	**El tipo MIME del archivo, indicado en el cliente.**
error	**Código de error devuelto que indica qué ha sucedido en la transferencia.**

La última clave de la tabla anterior, puede ser consultada en caso de que se produzca algún error al intentar cargar el archivo al servidor. Los códigos de error que pueden devolverse son los siguientes:

Valor	Constante	Significado
0	**UPLOAD_ERR_OK**	**No se han producido errores.**
1	**UPLOAD_ERR_INI_SIZE**	**El tamaño del archivo es superior al indicado en la directiva upload_max_filesize del archivo de configuración php.ini.**
2	**UPLOAD_ERR_FORM_SIZE**	**El archivo sobrepasa el tamaño indicado en el campo oculto max_file_size.**
3	**UPLOAD_ERR_PARTIAL**	**El archivo ha sido parcialmente transferido.**
4	**UPLOAD_ERR_NO_FILE**	**No se ha transferido el archivo.**

También se proporcionan algunas funciones útiles para trabajar con archivos transferidos desde el cliente al servidor. La primera de ellas, denominada is_uploaded_file($_FILES['adjunto']['tmp_name']) nos indica si realmente el archivo ha sido transferido. La segunda, denominada move_uploaded_file(nombrearchivo, rutadestino) sirve para renombrar y/o mover el archivo temporal a una ubicación permanente.
El siguiente código de ejemplo procesa el archivo una vez que llega al servidor:

```
<?php
   foreach($_FILES['adjunto'] as $clave => $valor)
   echo "\$_FILES[$clave] : $valor <br />";
   if(!is_uploaded_file($_FILES['adjunto']['tmp_name'])){
      $error = $_FILES['adjunto']['error'];
      die("<h3>*** ERROR: Archivo no transferido: $error ***</h3>");
   }
   if($_FILES['adjunto']['type'] != "application/x-zip-compressed"){
      echo "<h3>** ERROR: El archivo enviado no es un archive comprimido **</h3>";
```

```
    }
?>
```

III. MATERIALES Y EQUIPO

Para la realización de la guía de práctica se requerirá lo siguiente:

No.	Requerimiento	Cantidad
1	Guía de práctica #7: Archivos y directorios en PHP	1
2	Computadora con WampServer y PHP Designer 2007 instalado	1
3	Memoria USB	1

IV. PROCEDIMIENTO

Realice ordenadamente cada uno de los siguientes ejercicios. Algunos incluyen más de una script PHP junto con otros tipos de archivos.

Ejercicio #1: El siguiente ejemplo muestra cómo crear un archivo en formato csv (archivos de texto en los que cada registro de la información es separado por comas) para llevar registrado valores que se guardan y que luego pueden ser mostrados en una tabla HTML estándar.

Archivo 1: nuevocsv.html

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="utf-8" />
    <title>Nuevo enlace</title>
    <link rel="stylesheet" href="css/metallicform.css" />
</head>
<body>
<section>
<article id="metallic-form">
    <h1>Ingreso de enlaces</h1>
    <form method="POST" action="infoenlaces.php">
        <input type="text" name="dominio" id="dominio" maxlength="60" placeholder="(Nombre
del dominio)" />
        <input type="text" name="enlace" id="enlace" maxlength="150"
placeholder="http://www.dominio.com" />
        <input type="text" name="nivel" id="nivel" maxlength="150" placeholder="Nivel" />
        <input type="submit" name="registrar" value="Registrar" />
        <input type="reset" name="restablecer" value="Restablecer" />
    </form>
</article>
</section>
</body>
<script src="js/modernizr.custom.lis.js"></script>
</html>
```

Archivo 2: infoenlaces.php

```
<!DOCTYPE html>
<html lang="es">
<head>
   <meta charset="utf-8" />
   <title>Listado de enlaces</title>
   <link rel="stylesheet" href="css/niko.css" />
</head>
<body>
<header id="vintage">
    <h1>Enlaces registrados</h1>
</header>
```

```
<section>
   <article>
      <table>
         <caption>Información de los enlaces</caption>
            <thead>
               <tr>
                  <th>Texto del enlace</th>
                  <th>Enlace</th>
                  <th>Nivel</th>
               </tr>
            </thead>
         <tbody>
<?php
if($_SERVER['REQUEST_METHOD'] == "POST"){
   include_once("ponercampo.php");
   if(!file_exists("files")):
      if(!mkdir("files", 0700)):
         die("ERROR: No se puede crear el directorio.");
      endif;
   endif;
   define("ENLACES", "files/enlacescsv.txt");
   $archivo = fopen(ENLACES, 'a') or die("Error al intentar abrir archivo" . ENLACES);
   //Se genera un array con una posición por cada campo
   $valores = array($_POST["dominio"], $_POST["enlace"], $_POST["nivel"]);
   $escritos = fputcsv($archivo, $valores, ",");
   //Se cierra el archivo para que los datos queden guardados
   fclose($archivo);
   //Se abre el archivo para leer los datos de los productos
   $archivo = fopen(ENLACES, 'r');
   //Inicializar variable para llevar el control del número de fila
   $fila = 0;
   //Se recorre el archivo para mostrar el nuevo contenido
   while(!feof($archivo)){
      $buffer = fgetcsv($archivo, 4096, ",");
      if($buffer[0] != null){
         if($fila%2 != 0):
             echo "<tr>\n";
         else:
             echo "<tr class=\"odd\">\n";
         endif;
         ponercampo($buffer[0]);
         ponercampo($buffer[1]);
         ponercampo($buffer[2]);
         echo "</tr>\n";
         $fila++;
      }
   }
}
else{
   echo "\t\t\t<tr>\n";
   echo "\t\t\t\t<td colspan=\"3\">Error. No se puede acceder de forma directa a este
programa.</td>";
   echo "\t\t\t</tr>\n";
}
?>
         </tbody>
      </table>
      <a class="a-btn" title="Regresar" href="nuevocsv.html">
         <span class="a-btn-symbol">#</span>
         <span class="a-btn-text">Ingresar</span>
         <span class="a-btn-slide-text">nuevo enlace</span>
      </a>
   </article>
```

```
</section>
</body>
<script src="js/modernizr.custom.lis.js"></script>
</html>
```

Archivo 3: ponercampo.php

```
<?php
   function ponercampo($valor){
      //Si el campo está vacío, se escribe un carácter HTML en blanco
      if($valor == ""){
         $valor = " ";
      }
      echo "<td>\n" . $valor . "\n</td>\n";
   }
?>
```

En el navegador:

Cuando ya haya ingresado varios registros:

Ejercicio #2: El siguiente ejemplo muestra cómo implementar con cuatro scripts PHP un administrador de archivos básico con interfaz web que permitirá: operaciones como navegación por el árbol de directorios (el directorio actual), borrado, renombrado y copiado de archivos y carpetas, entre otras operaciones típicas con directorios y archivos:

Archivo 1: administrador.php

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1"/>
    <title>Administrador de archivos</title>
    <link rel="stylesheet" href="css/filemanager.css" />
    <!--[if IE]>
      <script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script>
    <![endif]-->
</head>
<body>
<header>
    <h1>Administrador de archivos</h1>
</header>
<section>
<article id="manager">
<?php
    if(!isset($_GET['directorio']) || $_GET['directorio'] == "."):
        $directorio = ".";
        $nombre_directorio = "/";
    else:
        if(isset($_GET['directorio']) && $_GET['directorio'] !== null):
            $directorio = $_GET['directorio'];
            $nombre_directorio = "\t" . basename($directorio);
        endif;
    endif;

    //Cambiar al directorio que se ha recibido como parámetro en la URL
    if(!chdir($directorio)):
        die("<h3>ERROR: No se puede acceder a este directorio</h3>");
    endif;
    $tabla  = "<table>\n";
    $tabla .= "\t<caption>Elementos del directorio: $nombre_directorio</caption>\n";

    //Abrir el manejador del directorio
    $manejador = opendir(".");

    //Procesar todos los elementos del directorio que también pueden ser directorios
    while($elemento = readdir($manejador)):
        if(is_dir($elemento) && ($elemento != "." && !($directorio == "." && $elemento ==
".."))):
             if($elemento == ".."):
                $ruta = dirname($directorio);
                $item = "<span>Directorio anterior</span>";
            else:
                $ruta = $directorio . "/" . $elemento;
                $item = "<span>$elemento</span>";
             endif;
             $tabla .= "\t<tr>\n\t\t<td>\n";
             $tabla .= "\t\t\t<a href=\"administrador.php?directorio=";
            $tabla .= rawurlencode($ruta) . "\">\n";
             $tabla .= "\t\t\t\t<img src=\"img/openfolder.png\" alt=\"Cambiar a $elemento\"
/>\n";
             $tabla .= "\t\t\t</a>\n";
            $tabla .= "\t\t</td>\n";
            $tabla .= "\t<td>\n";
            $tabla .= $elemento;
            $tabla .= "\t\t\t</td>\n\t\t</tr>\n";
            //echo $tabla;
        endif;
```

```
    endwhile;

    //Rebobinar el manejador de directorio
    rewinddir($manejador);

    //Procesar todos los elementos del directorio que son archivos
    while($elemento = readdir($manejador)):
        if(!is_dir($elemento)):
            $tabla .= "\t<tr>\n";
            $tabla .= "\t\t<td>\n";
            $tabla .= "\t\t\t<a href=\"" . $directorio . "/" . $elemento . "\">\n";
            $tabla .= "\t\t\t\t<img src=\"img/file.jpg\" alt=\"Mostrar $elemento\" />\n";
            $tabla .= "\t\t\t</a>\n";
            $tabla .= "\t\t</td>\n";
            $tabla .= "\t\t<td>$elemento</td>\n";
            $tabla .= "\t\t<td>\n";
            $tabla .= "\t\t\t<a href=\"operacionesarchivo.php?directorio=";
            $tabla .= rawurlencode($directorio . "/" . $elemento) . "\">\n";
            $tabla  .=  "\t\t\t\t<img  src=\"img/toolsicon.png\"  alt=\"Operaciones  con
$elemento\" />\n";
            $tabla .= "\t\t\t</a>\n";
            $tabla .= "\t\t</td>\n";
            $tabla .= "\t</tr>\n";
        endif;
    endwhile;
    //Cerrar el manejador de directorio
    closedir($manejador);
    $tabla .= "</table>\n";
    echo $tabla;
?>
</article>
</section>
</body>
<script src="js/modernizr.custom.lis.js"></script>
</html>
```

Archivo 2: operacionesdirectorio.php

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="utf-8" />
    <title>Operaciones con directorios</title>
    <link rel="stylesheet" href="css/filemanager.css" />
</head>
<body>
<header>
       <h1>Operaciones con el directorio
       <?php
         error_reporting(E_ALL & ~E_NOTICE);
         if($_GET['directorio'] == "."):
             $nombre_directorio = " Raíz";
         else:
             $nombre_directorio = " " . basename($_GET['directorio']);
         endif;
         echo basename($nombre_directorio);
         $valor_directorio = rawurlencode($_GET['directorio']);
       ?>
       </h1>
</header>
<section>
    <article>
    <?php
        $form  = "\n\t\t<form method=\"POST\"
```

```
action=\"operacion.php?operacion=0&directorio=";
        $form .= $valor_directorio . "\">\n\t\t\t";
        echo $form;
    ?>
        <input type="image" name="crear" id="crear" src="img/newfolder.png" alt="Crear
directorio" />
        <label for="crear">Crear</label>
        <label for="nombre_directorio">Nombre directorio</label>
        <input      type="text"      name="nombre_directorio"      id="nombre_directorio"
placeholder="Nombre del directorio" /><br />
    </form>
    <?php
        $form  = "\n\t\t<form method=\"POST\" action=";
        $form .= "\"operacion.php?operacion=1&directorio=$valor_directorio\">\n\t\t\t";
        echo $form;
    ?>
        <input  type="image"  name="mostrar"  id="mostrar"  src="img/folderopenfiles.jpg"
alt="Mostrar directorio completo" />
        <label for="mostrar">Mostrar</label>
    </form>
    <?php
        $form  = "<form method=\"POST\" action=\"operacion.php?operacion=2&directorio=";
        $form .= "$valor_directorio\">\n\t\t\t";
        echo $form;
    ?>
        <input type="image" name="borrar" id="borrar" src="img/deletefolder.png"
alt="Borrar directorio" />
        <label for="borrar">Borrar</label>
    </form>
    <?php
        $form  = "<form method=\"POST\" action=\"";
        $form .= "administrador.php?directorio=$valor_directorio\">\n\t\t\t";
        echo $form;
    ?>
        <input type="submit" name="volver" value="Volver al directorio" />
    </form>
    </article>
</section>
</body>
<script src="js/modernizr.custom.lis.js"></script>
</html>
```

Archivo 3: operacionesarchivo.php

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="utf-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1"/>
    <title>Operaciones con archivos</title>
    <link rel="stylesheet" href="css/filemanager.css" />
    <link rel="stylesheet" href="css/demo.css" />
    <link rel="stylesheet" href="css/component.css" />
    <!--[if IE]>
      <script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script>
    <![endif]-->
</head>
<body>
<header>
    <h1>Operaciones con el archivo
    <?php
        error_reporting(E_ALL & ~E_NOTICE);
        if(!empty($_GET['directorio']) && $_GET['directorio'] !== null):
```

```
            echo basename($_GET['directorio']);
            $valor_directorio = rawurlencode($_GET['directorio']);
        endif;
    ?>
    </h1>
</header>
<section>
    <article>
    <?php
        $form  = "<form method=\"POST\" action=\"";
        $form .= "operacion.php?operacion=3&directorio=" . $valor_directorio . "\"
class=\"ccform\">\n\t\t\t";
        echo $form;
    ?>
        <div class="ccfield-prepend">
            <input type="image" name="borrar" id="borrar" src="img/delete-file.gif"
alt="Borrar archivo" class="ccimagefield" />
            <label for="borrar">Borrar</label>
        </div>
    </form>
    <?php
        $form  = "<form method=\"POST\" action=\"";
        $form .= "operacion.php?operacion=4&directorio=" . $valor_directorio . "\"
class=\"ccform\">\n\t\t\t";
        echo $form;
    ?>
        <div class="ccfield-prepend">
            <span class="ccform-addon">
                <i class="fa fa-user fa-2x fa-spin"></i>
            </span>
            <input type="text" name="destino" id="destino" placeholder="Nombre de la
copia" required class="ccformfield" />
            <input type="image" name="copiar" id="copiar" src="img/copyfile.jpg"
alt="Copiar archivo" class="ccimagefield" />
            <label for="copiar">Copiar</label>
        </div>
    </form>
    <?php
        $form  = "<form method=\"POST\" action=\"";
        $form .= "operacion.php?operacion=5&directorio=" . $valor_directorio . "\"
class=\"ccform\">\n\t\t\t";
        echo $form;
    ?>
        <div class="ccfield-prepend">
            <span class="ccform-addon">
                <i class="fa fa-envelope fa-2x"></i>
            </span>
            <input type="text" name="nuevo_nombre" id="nuevo_nombre" placeholder="Nuevo
nombre" required class="ccformfield" />
            <input type="image" name="renombrar" id="renombrar" src="img/renamefile.png"
alt="Renombrar archivo" class="ccimagefield" />
            <label for="renombrar">Renombrar</label>
        </div>
    </form>
    <?php
        $form  = "<form method=\"POST\" action=\"";
        $form .= "administrador.php?directorio=" .
rawurlencode(dirname($_GET['directorio'])) . "\">\n\t\t\t";
        echo $form;
    ?>
        <div class="ccfield-prepend">
            <input type="submit" name="volver" value="Volver al directorio" class="ccbtn"
/>
```

```
        </div>
    </form>
    </article>
</section>
</body>
<script src="js/modernizr.custom.lis.js"></script>
</html>
```

Archivo 4: operacion.php

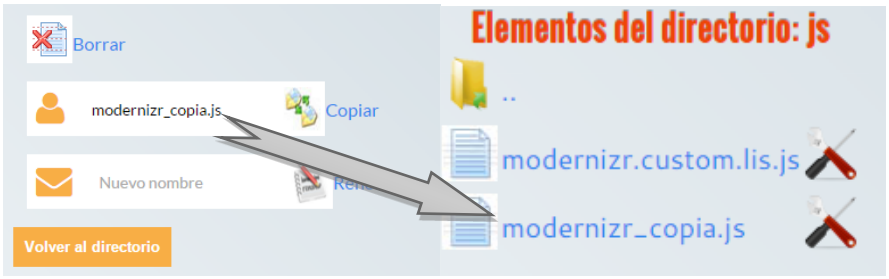
```
<?php
    //Función que muestra todos los elementos descendientes directos
    //o indirectos del directorio que se pasa como parámetro
    function mostrar_arbol($raiz){
        $nivel = "     ";
        echo $nivel . "<img src=\"directorio.gif\" alt=\"$raiz\" />";
        echo "<span>$raiz</span><br />";
        $manejador = opendir(".");
        while($elemento = readdir()):
            if(!is_dir($elemento)):
                echo $nivel . $nivel . "<img src=\"archivo.gif\" alt=\"$elemento\"
/>\n\t\t\t";
                echo $elemento . "<br />\n\t\t\t";
            endif;
        endwhile;
        rewinddir($manejador);
        while($elemento = readdir($manejador)):
             if(is_dir($elemento) && $elemento != "." && $elemento != ".."):
                chdir($elemento);
                mostrar_arbol("$raiz/$elemento");
             endif;
        endwhile;
        closedir($manejador);
        chdir("..");
    }

    //Función que determina si ya existe en el directorio
    //el nombre que se proporciona como parámetro
    function existe_en_directorio($nombre){
        return file_exists($nombre);
    }

    //Función que determina si el directorio actual se encuentra vacío
    function esta_vacio_directorio(){
        $manejador = opendir(".");
        $contador = 0;
        while($elemento = readdir($manejador)):
            $contador++;
        endwhile;
        closedir($manejador);
        return ($contador == 2);
    }

    //Función que escribe un botón que al ser pulsado vuelve
    //a mostrar el directorio indicado como parámetro
    function escribir_boton_volver($directorio){
        return "<form method=\"POST\" action=\"administrador.php?directorio=" .
               rawurlencode("$directorio") . "\">\n\t\t\t" .
               "<input type=\"submit\" name=\"volver\" value=\"Volver\" />\n\t\t" .
               "</form>\n\t";
    }

    //Función que muestra un mensaje de error y termina la ejecución del script
    function error($numero, $directorio){
```

```
      $htmlstr  = "<!DOCTYPE html>\n";
      $htmlstr .= "<html lang=\"es\">\n";
      $htmlstr .= "<head>\n\t";
      $htmlstr .= "<title>Página de error</title>\n";
      $htmlstr .= "</head>\n";
      $htmlstr .= "<body>\n\t<header>\n\t\t";
      $htmlstr .= "<h1>ERROR: No se puede ";
      switch($numero):
            case 0:
                  $htmlstr .= "acceder a este directorio</h1>\n\t";
                  break;
            case 1:
                  $htmlstr .= "crear este directorio</h1>\n\t";
                  break;
            case 2:
                  $htmlstr .= "borrar este directorio</h1>\n\t";
                  break;
            case 3:
                  $htmlstr .= "borrar este archivo</h1>\n\t";
                  break;
            case 4:
                  $htmlstr .= "copiar este archivo</h1>\n\t";
                  break;
            case 5:
                  $htmlstr .= "renombrar este archivo</h1>\n\t";
                  break;
      endswitch;
      $htmlstr .= escribir_boton_volver($directorio) . "</header>\n</boby>\n</html>";
      die($htmlstr);
    }

    if(isset($_GET['directorio']) && ($_GET['directorio'] != null || $_GET['directorio']
!= "")):
         $elemento = basename($_GET['directorio']);
         $ruta = dirname($_GET['directorio']);
    endif;

    //Cambio de directorio
    if(isset($_GET['operacion'])):
    if(($_GET['operacion'] >= 0 && $_GET['operacion'] <= 2 && !chdir($_GET['directorio']))
|| ($_GET['operacion'] >= 3 && $_GET['operacion'] <= 5 && !chdir($ruta))):
       error(0, ".");
    endif;

    //Ejecutar la operación requerida
    switch($_GET['operacion']):
        case 0:
             if(empty($_POST['nombre_directorio']) ||
             existe_en_directorio($_POST['nombre_directorio']) ||
             !mkdir($_POST, 0777)):
             error(1, $_GET['directorio']);
             endif;
             $ruta = $_GET['directorio'];
             break;
        case 1:
             $htmlstr  = "<!DOCTYPE html>\n";
             $htmlstr .= "<html lang=\"es\">\n";
             $htmlstr .= "<head>\n\t";
             $htmlstr .= "<title>Página de error</title>\n";
             $htmlstr .= "</head>\n";
             $htmlstr .= "<body>\n\t<header>\n\t\t";
             $htmlstr .= "<h1>Árbol completo de $elemento</h1>\n\t";
             $htmlstr .= "</header>\n";
```

```
            echo $htmlstr;
            break;
        case 2:
            if(!esta_vacio_directorio() || !rmdir("."):
                error(2, $_GET['directorio']);
            endif;
            break;
        case 3:
            if(empty($elemento) || !existe_en_directorio($elemento) ||
            !unlink($elemento)):
             error(3, $ruta);
            endif;
            break;
        case 4:
            if(empty($_POST['destino']) || existe_en_directorio($_POST['destino']) ||
            !copy($elemento, $_POST['destino'])):
             error(4, $ruta);
            endif;
            break;
        case 5:
            if(empty($_POST['nuevo_nombre'])                                    ||
existe_en_directorio($_POST['nuevo_nombre']) ||
            !rename($elemento, $_POST['nuevo_nombre'])):
                error(5, $ruta);
            endif;
            break;
    endswitch;
    endif;

    if(isset($_GET['operacion']) && $_GET['operacion'] != 1):
        header("Location: administrador.php?directorio=" . rawurlencode($ruta));
    endif;
?>
```

En el navegador:

Navegando en carpetas:



Ejercicio #3: El siguiente ejemplo muestra la implementación de un libro de visitas. Para que el ejercicio funcione sin inconvenientes deberá crear una carpeta en la carpeta wamp, fuera de la carpeta www y en el mismo nivel de dicha carpeta. El nombre que debe asignar a esta carpeta es guest. Si cambia este nombre deberá cambiar también la línea de código donde se hace referencia a la ubicación donde se almacenará el archivo con datos.

Archivo 1: guestBook.class.php

```
<?php
   class guestBook {
      //Propiedades
      protected $name;
      protected $address;
      protected $phone;
      protected $birthday;
      private $file;
      //Métodos para escritura de las propiedades
      function setName($name){
         $this->name = $name;
      }
      function setAddress($address){
         $this->address = $address;
      }
      function setPhone($phone){
         $this->phone = $phone;
      }
      function setBirthday($birthday){
         $this->birthday = $birthday;
      }
      function setFile($file){
         $this->file = $file;
      }
      //Métodos para lectura de las propiedades
      function getName($name){
         return $this->name;
```

```
      }
      function getAddress($address){
         return $this->address;
      }
      function getPhone($phone){
         return $this->phone;
      }
      function getBirthday($birthday){
         return $this->birthday;
      }
      function getFile($file){
         return $this->file;
      }
      function showGuest(){
         echo "<div id=\"showGuest\">";
         echo "$this->name.<br>";
         echo "$this->address.<br>";
         echo "$this->phone.<br>";
         echo "$this->birthday.<br>";
         echo "</div>";
      }
      function saveGuest(){
         $outputstring  = $this->name . " : " . $this->address . " : ";
         $outputstring .= $this->phone . " : " . $this->birthday . "\n";
         //Se requerirá que cree esta carpeta "/guest" en el directorio raíz del servidor
         $path = "$_SERVER[DOCUMENT_ROOT]/../guest/$this->file";
         @$fh = fopen($path, "ab");
         if(!$fh){
            $fh = fopen($path, "wb");
         }
         fwrite($fh, $outputstring, strlen($outputstring));
         fclose($fh);
         echo "<h3>Datos salvados en la ruta indicada $path</h3>";
      }
   }
?>
```

Archivo 2: recordvisits.php

```
<!DOCTYPE html>
<html lang="es">
<head>
    <title>Registrar usuarios en libro de visitas</title>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="stylesheet" href="css/style.css" />
    <link rel="stylesheet" href="css/responsive.css" />
    <style type="text/css">
        @import url(jscalendar/calendar-blue2.css);
    </style>
    <script src="js/actions.js"></script>
    <script src="jscalendar/calendar.js"></script>
    <script src="jscalendar/lang/calendar-es.js"></script>
    <script src="jscalendar/calendar-setup.js"></script>
</head>
<body>
<?php
//Implementación de la clase classAutoLoader
if(!function_exists('classAutoLoader')){
   function classAutoLoader($classname){
       $classname = strtolower($classname);
       $classFile = 'class/' . $classname . '.class.php';
       if(is_file($classFile) && !class_exists($classname)) include $classFile;
   }
```

```
 }

//Registrando la clase classAutoLoader
spl_autoload_register('classAutoLoader');

if($_SERVER['REQUEST_METHOD'] == "POST"){
    //Obtener los datos del formulario sin procesamiento
    //algo que sólo haremos por motivos de demostración
    extract($_POST);
    $entry = new guestBook();
    $entry->setName($yourname);
    $entry->setAddress($youraddress);
    $entry->setPhone($yourphone);
    $entry->setBirthday($yourbirthday);
    $entry->setFile($yourfile);
    $entry->showGuest();
    $entry->saveGuest();
}
else{
?>
<section id="container">
    <!-- <span class="chyron"><em><a href="http://www.hongkiat.com/blog/">&laquo; back to
the site</a></em></span> -->
    <h2>Ingreso de datos</h2>
    <form name="hongkiat" id="hongkiat-form" action="<?php echo $_SERVER['PHP_SELF'] ?>"
method="POST">
    <div id="wrapping" class="clearfix">
        <section id="aligned">
            <input type="text" name="yourname" id="yourname" placeholder="(Tu nombre)"
autocomplete="off" maxlength="40" tabindex="1" class="txtinput" />
            <input type="text" name="youraddress" id="youraddress" placeholder="(Tu
dirección)" autocomplete="off" maxlength="60" tabindex="2" class="txtinput" />
            <input type="tel" name="yourphone" id="yourphone" placeholder="(Tu número de
teléfono)" tabindex="4" maxlength="14" class="txtinput" />
            <input type="hidden" name="yourfile" id="yourfile" value="guestbook.txt" />
        </section>

        <section id="aside" class="clearfix">
            <section id="recipientcase">
                <h3>Fecha de nacimiento:</h3>
                <input type="date" name="yourbirthday" id="yourbirthday" placeholder="(Tu
fecha de nacimiento)" class="txtdate" />
                <img src="jscalendar/image.png" id="imgcalendar" style="cursor: pointer;
border: 1px solid red; display: inline-block;" title="Ingrese la fecha" /><br />
                <script>
                    Calendar.setup(
                        {
                            inputField  : "yourbirthday", // ID of the input field
                            ifFormat    : "%Y-%m-%d", // the date format
                            button      : "imgcalendar", // ID of the button
                            singleClick : true
                        }
                    );
                </script>
            </section>
        </section>
    </div>
    <section id="buttons">
        <input type="reset" name="reset" id="resetbtn" class="resetbtn" value="Restaurar"
/>
        <input type="submit" name="submit" id="submitbtn" class="submitbtn" tabindex="7"
value="Guardar" />
        <br style="clear:both;">
```

```
    </section>
    </form>
</section>
<?php
   }
?>
</body>
</html>
```

En el navegador:

Al guardar los datos:

Ejercicio #4: La siguiente aplicación ilustra cómo crear un editor de texto en PHP, con la particularidad que permite mediante el uso de formularios crear archivos, listar los archivos creados y editarlos en una misma aplicación. Para la edición de los archivos existentes se hace uso de un campo de formulario del tipo textarea. Debe crear una carpeta llamada archivos para que funcione correctamente este ejercicio. En este ejemplo también utilizará la hoja de estilos enlaces.css.

Archivo 1: editortexto.php

```
<?php
  require_once("funcioneseditor.php");
  if($_SERVER['REQUEST_METHOD'] == "POST" && isset($_POST['savefile'])){
      savefile();
  }
  elseif($_SERVER['REQUEST_METHOD'] == "GET" && isset($_GET['filename'])){
      displayeditform();
  }
  elseif($_SERVER['REQUEST_METHOD'] == "POST" && isset($_POST['createfile'])){
      createfile();
  }
  else{
      displayfilelist();
  }
?>
```

Archivo 2: funcioneseditor.php

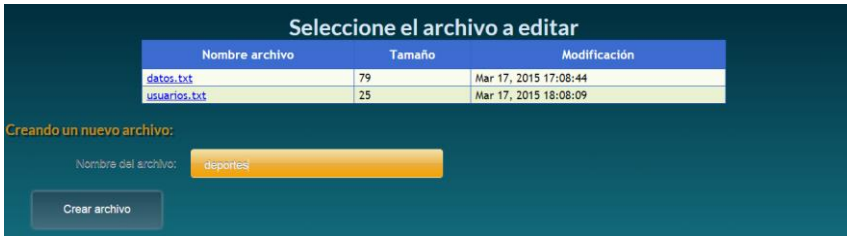
```
<?php
define("PATH", "archivos");
```

```
function displayfilelist($message = ""){
   displaypageheader();
   //Verificar si existe el directorio. Si no existe crearlo
   if(!file_exists(PATH)){
      //Crear el directorio y asignar los permisos al mismo
      if(!mkdir(PATH, 0777, true)) {
         die('No se ha podido crear el directorio');
      }
   }
   if(!($dir = dir(PATH))) die("No se puede abrir el directorio");
   if($message){
      echo "<p class=\"error\">$message</p>";
   }
   //Imprimiendo encabezados de la tabla
   $table  = "<table id=\"file-data\">\n";
   $table .= "<caption>Seleccione el archivo a editar</caption>\n";
   $table .= "<thead>\n<tr>\n<th>\nNombre archivo\n</th>\n";
   $table .= "<th>\nTamaño\n</th>\n";
   $table .= "<th>\nModificación</th>\n</tr>\n</thead>\n";
   //Imprimiendo los archivos contenidos en el directorio y
   //creando los enlaces para su edición
   $table .= "<tbody>\n";
   $numfilas = 0;
   //Usando pseudo-clase para leer
   while($filename = $dir->read()){
      $filepath = PATH . "/$filename";
      if($filename != "." && $filename != ".." && !is_dir($filepath) && strrchr($filename,
".") == ".txt"){
         $clase = ($numfilas%2 != 0) ? "odd" : "even";
         $table .= "<tr class=\"$clase\">\n<td>\n";
         $table  .=  "<a  href=\"editortexto.php?filename="  .  urlencode((binary)$filename)  .
"\">";
         $table .= $filename . "</a>\n</td>\n";
         $table .= "<td>\n" . filesize($filepath) . "\n</td>\n";
         $table .= "<td>\n" . date("M j, Y H:i:s", filemtime($filepath)) . "\n</td>\n</tr>\n";
      }
      $numfilas++;
   }
   $dir->close();
   $table .= "</tbody>\n</table>\n";
   echo $table;
   $form  = "<form name=\"nuevo\" action=\"editortexto.php\" method=\"POST\">\n";
   $form .= "<fieldset>\n<legend><span>Creando un nuevo archivo:</span></legend>\n";
   $form .= "<ul>\n<li>\n";
   $form .= "<div id=\"campo\">\n";
   $form .= "<label for=\"filename\">Nombre del archivo: </label>";
   $form .= "<input type=\"text\" name=\"filename\" id=\"filename\" placeholder=\"(Ingrese el
nombre del archivo)\" maxlength=\"100\" />\n";
   $form .= "</div>\n</li>\n";
   $form .= "<li>\n<div id=\"boton\">\n";
   $form .= "<input type=\"submit\" name=\"createfile\" value=\"Crear archivo\" />";
   $form .= "</div>\n</li>\n</ul>\n";
   $form .= "</fieldset>\n</form>\n";
   echo $form;
   displaypagefooter();
}

function displayeditform($filename = ""){
   $archivo = isset($_GET['filename']) ? $_GET['filename'] : "";
   echo "<div id=\"info-file\">\n";
   //echo $archivo . "<br />\n";
   if(!$filename) $filename = basename($archivo);
```

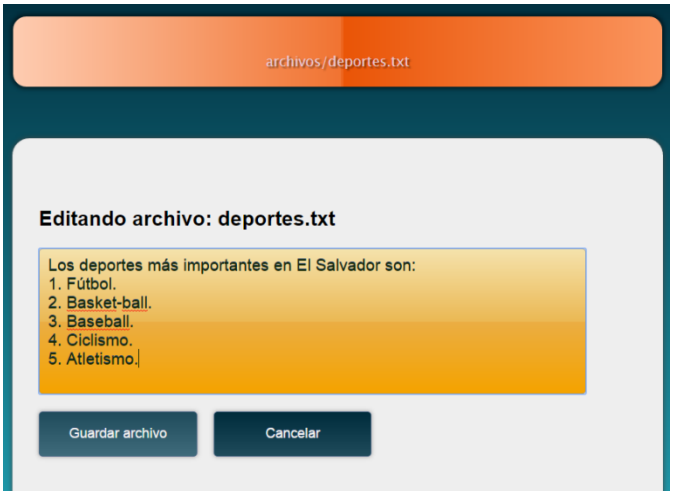
```
   if(!$filename) die("Nombre de archivo inválido");
   $filepath = PATH . "/" . $filename;
   echo $filepath . "\n";
   echo "</div>\n";
   if(!file_exists($filepath)) die("Archivo no encontrado");
   displaypageheader();
   $editform  = "<section id=\"formulario\">\n";
   $editform .= "\t<h2>Editando archivo: $filename</h2>\n";
   $editform .= "\t<form name=\"creararchivo\" action=\"editortexto.php\" method=\"POST\">\n";
   $editform .= "\t\t<div style=\"width:40em\">\n";
   $editform .= "\t\t\t<input type=\"hidden\" name=\"filename\" value=\"$filename\" />\n";
   $editform  .=  "\t\t\t<textarea  name=\"filecontents\"  id=\"filecontents\"  cols=\"80\"
rows=\"20\">\n";
   $editform .= file_get_contents($filepath) . "\n";
   $editform .= "</textarea>\n";
   $editform .= "\t\t\t<div style=\"clear:both\">\n";
   $editform .= "\t\t\t\t<input type=\"submit\" name=\"savefile\" value=\"Guardar archivo\"
/>\n";
   $editform .= "\t\t\t\t<input type=\"submit\" name=\"cancel\" value=\"Cancelar\" />\n";
   $editform .= "\t\t\t</div>\n";
   $editform .= "\t\t</div>\n\t</form>\n";
   $editform .= "</section>\n";
   echo $editform;
   displaypagefooter();
}

function savefile(){
   $archivo = isset($_POST['filename']) ? $_POST['filename'] : "";
   $filename = basename($archivo);
   $filepath = PATH . "/$filename";
   if(file_exists($filepath)){
      $filecontents = isset($_POST['filecontents']) ? $_POST['filecontents'] : "";
      if(file_put_contents($filepath, $_POST['filecontents']) === false)
         die("<p class=\"error\">No se ha podido guardar el archivo</p>\n");
      displayfilelist();
   }
   else{
      die("Archivo no encontrado...");
   }
}

function createfile(){
   $filename = basename($_POST['filename']);
   echo $filename . "<br />\n";
   $filename = preg_replace("/[^A-Za-z0-9_\- ]/", "", $filename);
   if(!$filename){
      displayfilelist("<p  class=\"error\">Nombre  de  archivo  no  válido.  Pruebe  con  otro
nombre.</p>");
      return;
   }
   $filename .= ".txt";
   $filepath = PATH . "/$filename";
   if(file_exists($filepath)){
      displayfilelist("El archivo $filename ya existe.");
   }
   else{
      if(file_put_contents($filepath, "") === false)
         die("No se ha podido crear el archivo");
      chmod($filepath, 0777);
      displayeditform($filename);
   }
}
```

```
function displaypageheader(){
   echo "<!DOCTYPE html>\n";
   echo "<html lang=\"es\">\n";
   echo "<head>\n";
   echo "<title>Editor de texto basado en web</title>\n";
   echo "<meta charset=\"utf-8\" />\n";
   echo "<link rel=\"stylesheet\" href=\"css/page.css\" />\n";
   echo "</head>\n";
   echo "<body>\n";
}

function displaypagefooter(){
   echo "</body>\n";
   echo "</html>\n";
}
?>
```

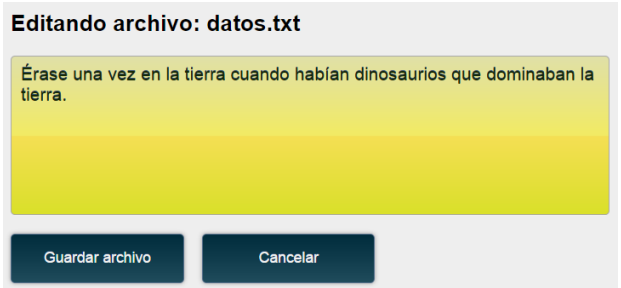
En el navegador:



Creando un nuevo archivo con el editor:



Cargando un archivo previamente creado:



Ejercicio #5: Ejemplo de cómo implementar la carga de archivos al servidor desde un formulario web en el cliente. La aplicación incluye un script JavaScript que se utiliza para poder subir más de un archivo a la vez, creando dinámicamente un nuevo juego de controles de formulario del tipo file. Debe crear una carpeta llamada archivos dentro de la carpeta donde está guardando los scripts PHP de esta guía de práctica.

Archivo 1: uploadfile.html

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="utf-8" />
    <title>Subir múltiples archivos</title>
    <link                                                                    rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css" />
    <link rel="stylesheet" href="css/upload.css" />
    <script src="js/upload.js"></script>
</head>
<body>
<section>
   <article>
      <form      name="formu"      id="formu"      action="upload.php"      method="POST"
enctype="multipart/form-data">
         <dl>
            <dt>
               <h2>Archivos a Subir:</h2>
            </dt>
            <!-- Este div contendrá todos los campos file que creemos -->
            <dd>
            <div id="conteGeneral">
               <div class="contenedor">
                  <input  type="text"  id="uploadFile0"  placeholder="Seleccionar  archivo"
disabled="disabled" />
                  <div id="adjuntos" class="file-upload btn ">
                     <!-- Hay que prestar atención a esto, el nombre
                     de este campo siempre debe terminar en [] como
                     un vector, y además debe coincidir con el nombre
                     que se da a los campos nuevos en el script js -->
                        <span class="btn btn-primary span" id="spanadj">Adjunto</span>
                        <input type="file" name="archivos[]" id="uploadBtn0" class="upload"
/><br/>
                  </div>
               </div>
            </div>
            </dd>
            <dt>
               <a href="javascript:void(0);" id="addfieldlink">Subir otro archivo</a>
            </dt>
            <dd>
               <input type="submit" value="Enviar" id="envia" name="envia" class="btn btn-
default" />
            </dd>
         </dl>
      </form>
   </article>
</section>
</body>
</html>
```

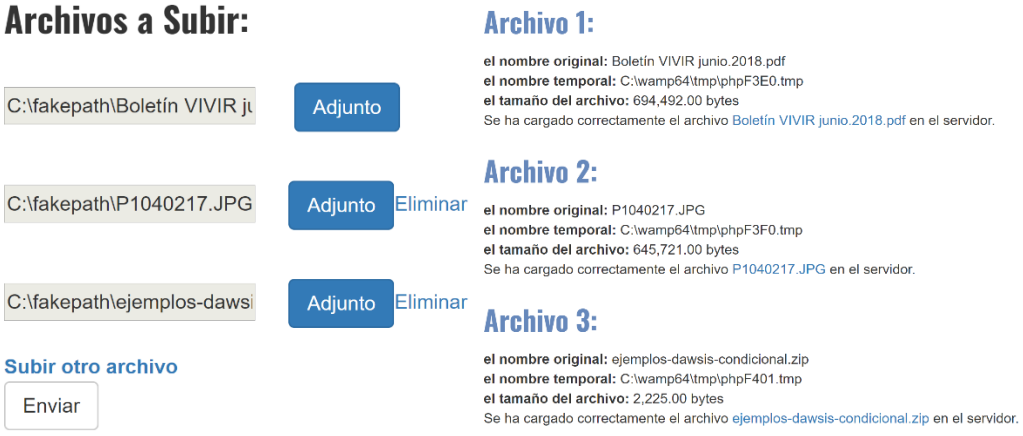
Archivo 2: upload.php

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="utf-8" />
    <title>Subir múltiples archivos</title>
```

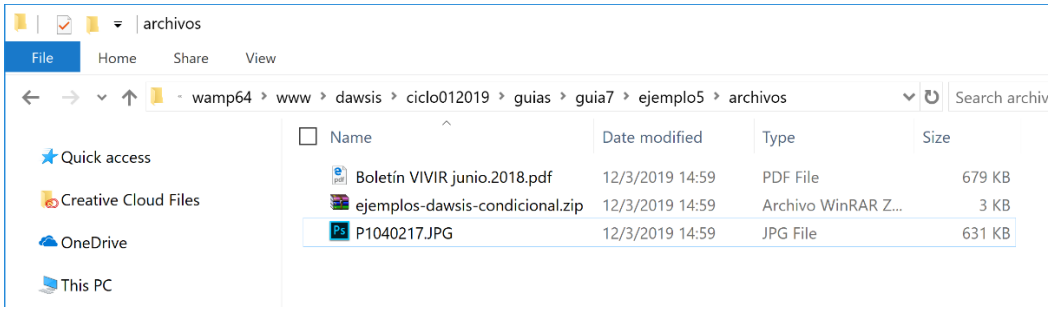
```
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css" />
    <link rel="stylesheet" href="css/upload.css" />
</head>
<body>
<?php
  define("PATH", "archivos");
  //Verificar que la matriz asociativa $_FILES["archivos"] haya sido definida
  if ($_SERVER['REQUEST_METHOD'] == "POST" && isset($_FILES["archivos"])){
      //De ser así se procesa cada uno de los archivos.
      //Para poder hacerlo es conveniente obtener la cantidad
          //de elementos que tiene matriz $_FILES["archivos"]
      $total = count($_FILES["archivos"]["name"]);
      //este for recorre la matriz $_FILES
      for($i = 0; $i < $total; $i++){
          //Las propiedades definidas para cada archivo son:
          //1. 'tmp_dir': directorio temporal en el servidor donde se aloja el archivo
          //2. 'name': nombre original del archivo seleccionado por el usuario
          //3. 'size': tamaño en bytes del archivo
          //Para recorrer uno a uno los archivos que se hayan decidido subir al servidor
          //se utilizará el contador $i. En caso de que solo se suba un archivo el ciclo
          //se ejecutará una sola vez.
          $tmp_name = $_FILES["archivos"]["tmp_name"][$i];
          $name = $_FILES["archivos"]["name"][$i];
          $size = $_FILES["archivos"]["size"][$i];
          //echo "<h3>$size bytes</h3>";
          if($size > 2621440){
              echo "<h3>El tamaño del archivo es superior al admitido por el
servidor</h3><br>";
              echo "<a href=\"uploadfile.html\">Intentar de nuevo</a>";
          }
          echo "<h3 class=\"title\">Archivo " . ($i+1) . ":</h3>";
          echo "<b>el nombre original:</b> ";
          echo $name;
          echo "<br />";
          echo "<b>el nombre temporal:</b> \n";
          echo $tmp_name;
          echo "<br />";
          echo "<b>el tamaño del archivo:</b> \n";
          echo number_format($size, 2);
          echo " bytes<br />";
          //Verificar la carpeta en el servidor donde se alojarán los archivos
          //que se desean subir. Si no existe esta carpeta se creará y si no
          //es posible crearla se lanzará un error y se terminará el script
          if(!file_exists(PATH)){
              //Crear el directorio y asignar los permisos al mismo
              if(!mkdir(PATH, 0777, true)) {
                  die('No se ha podido crear el directorio');
              }
           }
           //Una vez que es procesado cada archivo correctamente, se moverá
           //a una carpeta específica en el servidor, en este caso se usará
           //la carpeta files/.
           if(move_uploaded_file($tmp_name, PATH . "/" . utf8_decode($name))){
              echo "Se ha cargado correctamente el archivo " .
                   "<a href=\"archivos/" . urldecode($name) . "\" target=\"_blank\">" .
$name . "</a>\n" . " en el servidor.\n<br />\n";
           }
           else{
              switch($_FILES['archivos']['error'][$i]){
                 //No hay error, pero puede ser un ataque
                 case UPLOAD_ERR_OK:
                      echo "<p>Se ha producido un problema con la carga del archivo.</p>\n";
```

```
                    break;
                //El tamaño del archivo es mayor que upload_max_filesize
                case UPLOAD_ERR_INI_SIZE:
                    echo "<p>El archivo es demasiado grande, no se puede cargar.</p>\n";
                    break;
                //El tamaño del archivo es mayor que MAX_FILE_SIZE
                case UPLOAD_ERR_FORM_SIZE:
                    echo "<p>El archivo es demasiado grande, no se pudo cargar.</p>\n";
                    break;
                //Solo se ha cargado parte del archivo
                case UPLOAD_ERR_PARTIAL:
                    echo "<p>Sólo se ha cargado una parte del archivo.</p>\n";
                    break;
                //No se ha seleccionado ningún archivo para subir
                case UPLOAD_ERR_NO_FILE:
                    echo "<p>Debe elegir un archivo para cargar.</p>\n";
                    break;
                //No hay directorio temporal
                case UPLOAD_ERR_NO_TMP_DIR:
                    echo "<p>Problema con el directorio temporal. Parece que no
existe</p>\n";
                    break;
                default:
                    echo "<p>Se ha producido un problema al intentar mover el archivo "
. $name . "</p>\n";
                    break;
            }
        }
    }
    else{
        echo "<h3>No se han seleccionado archivos.</h3>";
    }
?>
</body>
</html>
```

El resultado al visualizarlo en el navegador de su preferencia sería:



Carpeta en donde han sido alojados los archivos:



Ejercicio #6: En este ejemplo se busca dentro de un archivo el registro indicado por el primer nombre ingresado en la caja de texto para búsqueda. Abra el archivo datebook.txt y seleccione el primer nombre de un registro y digítelo en la caja de texto (Nombre a buscar). El script PHP debería mostrar todos los registros del archivo que coincidan con ese primer nombre.

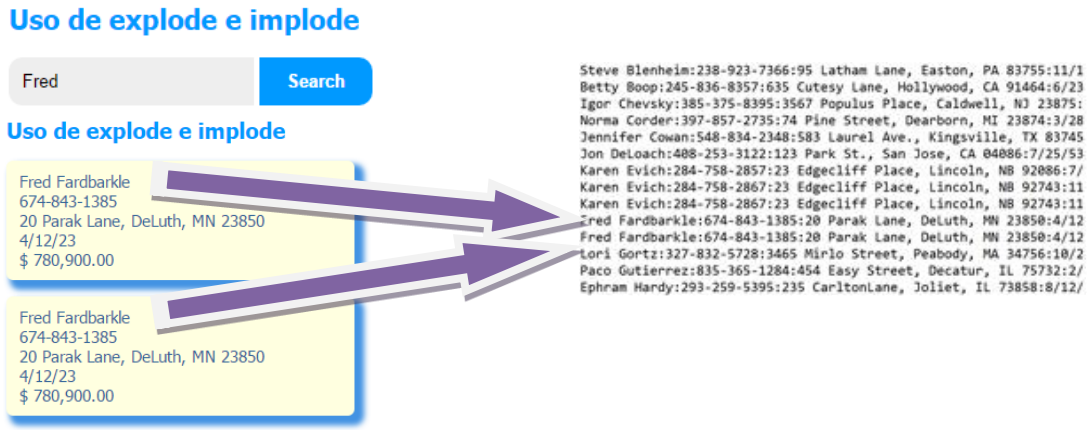
Archivo 1: functions.lib.php

```
<?php
function showform(){
?>
<form method="POST" action="<?php echo $_SERVER['PHP_SELF']; ?>" id="searchthis">
    <input type="text" name="firstname" id="namanyay-search-box" placeholder="(Nombre a
buscar e.g. Karen, Evelyn, Arthur)"/>
    <input name="submit" id="namanyay-search-btn" value="Search" type="submit" />
</form>
<?php
}

function processfile(){
       $filename = "files/datebook.txt";
       //Obteniendo las líneas del archivo para asignarlas en una matriz
       $rows = file($filename);
       $firstname = trim($_POST['firstname']);

       //Recorriendo una por una las líneas extraídas del archivo a través de la matriz
       $count = 0;
       foreach($rows as $register):
              $fields = explode(":", $register);
           $fullname = explode(" ", $fields[0]);
           $phone = $fields[1];
           $address = $fields[2];
           $birthday = $fields[3];
           $salary = $fields[4];
           if(strcasecmp($fullname[0], $firstname) == 0):
              $birthday = explode("/", $birthday);
                $newstring = implode("<br />",
                                     array($fields[0],
                                     $phone,
                                     $address,
                                     implode("/", $birthday),
                                     '$ ' . number_format(floatval($salary), 2, '.', ',')));
                echo "<p>" . $newstring . "</p>";
                $count++;
           endif;
       endforeach;
       if($count == 0):
              echo "<p>El nombre " . $firstname . " no está en el archivo</p>";
       endif;
}
?>
```

Archivo 2: implodefile.php

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="utf-8" />
    <title>Accediendo a las líneas de un archivo</title>
    <link rel="stylesheet" href="css/form.css" />
</head>
<body>
<header>
    <h2>Uso de explode e implode</h2>
</header>
<section>
<article>
<?php
    include_once("functions.lib.php");
    if(!isset($_POST['submit'])):
        showform();
    elseif($_SERVER['REQUEST_METHOD'] == "POST"):
        processfile();
    endif;
?>
</article>
</section>
</body>
</html>
```

En el navegador de su preferencia podrá ver:



Ejercicio #7: Este ejemplo muestra una aplicación que permite subir imágenes al servidor web en una carpeta y mostrarlos de forma dinámica, una vez estos archivos ya han sido procesados. El programa valida que las imágenes sean de los formatos: JPG, PNG o GIF para aceptarlos. Además, existe una opción de eliminar archivos de imagen del servidor de manera rápida y fácil.

Archivo 1: index.php

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Subida de archivos al servidor con PHP</title>
    <!-- Bootstrap core CSS -->
    <link rel="stylesheet" href="boostrap/css/bootstrap.min.css">
</head>
```

```
<body>
<!-- Static navbar -->
<div class="navbar navbar-default navbar-static-top">
  <div class="container">
    <div class="navbar-header">
      <a class="navbar-brand" href="index.php">Cargador de archivos al servidor</a>
    </div>
  </div>
</div>
<div class="container">
       <div class="row">
       <?php
             //Escaneando la carpeta uploads para verificar
        //si existen archivos subidos y mostrarlos.
       $folder = "uploads";
       $results = scandir('uploads');
       foreach ($results as $result) {
             if ($result === '.' or $result === '..') continue;

             if (is_file($folder . '/' . $result)) {
                   echo '
                   <div class="col-md-3">
            <!-- Dentro de este elemento DIV se muestran todas
                 las imágenes que han sido subidas previamente
                 a la carpeta uploads -->
                            <div class="thumbnail">
                                  <!-- Se utiliza utf8_encode() para garantizar
                    que se muestren correctamente los nombres
                    de los archivos si incluyeran caracteres
                    especiales del idioma español -->
              <img src="' . $folder . '/' . utf8_encode($result) . '" alt="...">
                                  <div class="caption">
                 <!-- Se utiliza basename para extraer
                      únicamente el nombre del archivo sin la
                      ruta de la variable $result -->
                                        <p>' . basename(utf8_encode($result)) . '</p>
                                          <p><a  href="remove.php?name='  .  $result  .  '"
class="btn btn-danger btn-xs" role="button">Eliminar</a></p>
                                  </div>
                            </div>
                     </div>';
             }
       }
       ?>
      </div>
      <div class="row">
             <div class="col-lg-12">
              <form class="well" action="upload.php" method="POST" enctype="multipart/form-
data">
                    <div class="form-group">
                         <input type="hidden" name="MAX_FILE_SIZE" value="4096000" />
                      <!--<label for="file">Seleccione el archivo a cargar</label>-->
                      <div class="input-group">
            <div class="input-group-prepend">
              <span class="input-group-text" id="inputGroupFileAddon01">Upload</span>
            </div>
            <div class="custom-file">
              <input  type="file"  accept="image/gif,  image/jpeg,  image/png"  name="file"
id="inputGroupFile01" class="custom-file-input" aria-describedby="inputGroupFileAddon01">
              <label class="custom-file-label" for="inputGroupFile01">Seleccione el archivo
a cargar</label>
            </div>
          </div>
```

```
                        <p class="help-block">Sólo se permiten imágenes de formato jpg,
jpeg, png y gif con un máximo de 4 MB.</p>
                      </div>
                      <input type="submit" class="btn btn-lg btn-primary" value="Subir">
                    </form>
              </div>
       </div>
</div> <!-- Fin del div class="container" -->
</body>
</html>
```

Archivo 2: upload.php

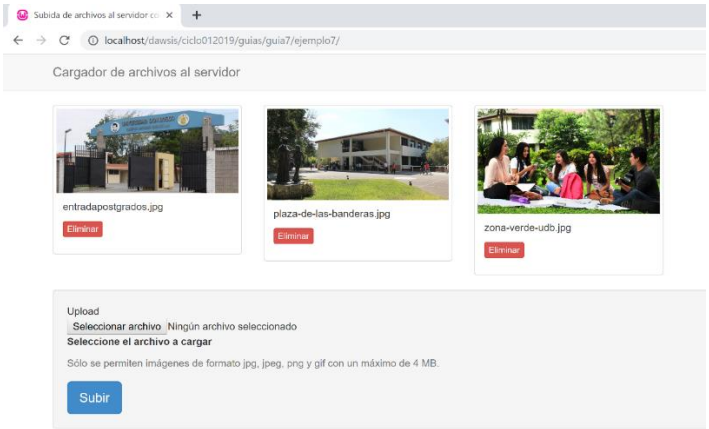
```
<?php
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
       $name     = $_FILES['file']['name'];
       $tmpName  = $_FILES['file']['tmp_name'];
       $error    = $_FILES['file']['error'];
       $size     = $_FILES['file']['size'];
    $ext       = strtolower(pathinfo($name, PATHINFO_EXTENSION));

       switch ($error) {
              //Verificando que el archivo ha sido subido con éxito
              //utilizando las constantes de error devueltas por
              //la matriz superglobal $_FILES['archivo_usuario']['error']
              case UPLOAD_ERR_OK:
                     $valid = true;
                     //Validando que la extensión del archivo subido
                     //al servidor sea de las extensiones permitidas
                     if (!in_array($ext, array('jpg','jpeg','png','gif'))) {
                            $valid = false;
                            $response  =  '<div  class="col-md-3"><div  class="caption"><p>La
extensión del archivo no es válida.</p></div></div>';
                     }
                     //Validando que el archivo pese 4 MB o menos
                     if ($size > 4096000) {
                            $valid = false;
                            $response  =  '<div  class="col-md-3"><div  class="caption"><p>El
tamaño del archivo excede el máximo permitido de 4 MB.</p></div></div>';
                     }
                     //Si se pasaron todas las verificaciones mover
                     //el archivo a la carpeta destino que es uploads
                     if ($valid) {
                            $targetPath  =   dirname(  __FILE__  )  .  DIRECTORY_SEPARATOR  .
'uploads' . DIRECTORY_SEPARATOR . $name;
                            move_uploaded_file($tmpName, utf8_decode($targetPath));
                            //Redirigir al usuario a la página de inicio index.php
                            header('Location: index.php') ;
                            exit;
                     }
                     break;
              //El archivo subido excede la directiva
              //upload_max_filesize del php.ini
              case UPLOAD_ERR_INI_SIZE:
                     $response = '<div class="col-md-3"><div class="caption"><p>El tamaño del
archivo   excede   el   establecido   en   la   directiva   upload_max_filesize   del
php.ini.</p></div></div>';
                     break;
         //El archivo subido excede la directiva MAX_FILE_SIZE
         //especificada en el formulario HTML
              case UPLOAD_ERR_FORM_SIZE:
             $response  =  '<div  class="col-md-3"><div  class="caption"><p>El  archivo  subido
excede la directiva MAX_FILE_SIZE especificada en el formulario HTML.</p></div></div>';
             break;
```

```
            //El archivo sólo se subió parcialmente.
            case UPLOAD_ERR_PARTIAL:
                  $response = '<div class="col-md-3"><div class="caption"><p>El archivo
se subió sólo parcialmente.</p></div></div>';
                  break;
            //No se subió archivo alguno.
            case UPLOAD_ERR_NO_FILE:
                  $response = '<div class="col-md-3"><div class="caption"><p>No se ha
subido ningún archivo.</p></div></div>';
                  break;
            //No existe la carpeta temporal requerida en el servidor
            //para implementar la carga de archivos.
            case UPLOAD_ERR_NO_TMP_DIR:
                  $response = '<div class="col-md-3"><div class="caption"><p>No existe la
carpeta temporal en el servidor para subida de archivos.</p></div></div>';
                  break;
            //No se ha podido escribir el archivo en disco,
            //seguramente por problemas de permisos y porque
            //falta la carpeta o directorio destino en el servidor.
            case UPLOAD_ERR_CANT_WRITE:
                  $response = '<div class="col-md-3"><div class="caption"><p>Se ha
producido un error de escritura en disco cuando se intentó subir el archivo al
servidor.</p></div></div>';
                  break;
            default:
                  $response = 'Error desconocido';
            break;
      }
      echo $response;
}
?>
```

Archivo 3: remove.php

```
<?php
// Obtener la ruta correcta
$fileName = $_GET['name'];
$filePath = 'uploads/'.$fileName;
// Eliminar el archivo, si se encontró
if (file_exists($filePath)) {
      unlink($filePath);
      header('Location:index.php');
}
?>
```

En el navegador:



V. DISCUSION DE RESULTADOS

1. Tome el ejemplo sobre cuentas bancarias de la guía de práctica #6 (guía anterior) de esta materia y realice las modificaciones necesarias sobre los archivos o secuencias de comando PHP que contiene esa aplicación para que ahora, haciendo uso de archivos, de preferencia csv, pueda realizar operaciones de retiro y depósito sobre las cuentas bancarias conservando los valores si la cuenta bancaria existe. En caso de que se quiera crear una cuenta con el mismo nombre de un propietario lanzar un mensaje de error. En el archivo debe almacenar el número de la cuenta, el propietario y el saldo que presenta la misma.
2. Realice una aplicación PHP en la que se pueda llevar un log de visitas indicando en primer lugar la IP del visitante (utilice alguna de las matrices superglobales de servidor de PHP, $_SERVER, para obtener esta IP), el nombre del script que ejecutó y la fecha y hora al que se produjo esta visita. Los datos deben ser almacenados en un archivo de texto y luego un administrador debe poder abrir este archivo para verificar los datos de los visitantes que han ingresado al sitio mostrando los datos antes mencionados en alguna tabla HTML.

VI. INVESTIGACIÓN COMPLEMENTARIA

- Investigue sobre la utilización de la función de las funciones file_get_contents() y file_put_contents() indicando con detalle su sintaxis, argumentos que reciben y valores devueltos. Si es necesario explicar en detalle uno o varios de los argumentos de la función puede utilizar tablas u otro medio para realizar la explicación. Además de la sintaxis muestre un ejemplo detallado sobre cómo utilizar estas dos funciones, desde que abre el archivo, hasta cerrarlo.
- Investigue sobre la pseudo-clase dir para el trabajo con directorios y los métodos que posee para realizar tareas como leer rebobinar y cerrar, así como las propiedades de consulta que posee. Además de la descripción de la clase, muestre ejemplos cortos sobre su utilización.

VII. BIBLIOGRAFIA

- Cabezas Granado, Luis Miguel. PHP 6 Manual Imprescindible. 1ra. Edición. Editorial Anaya Multimedia. Madrid, España. 2010.
- Doyle, Matt. Fundamentos de PHP Práctico. Editorial Anaya Multimedia. 1ª. Edición. Madrid, España. 2010.
- Gutierrez, Abraham / Bravo, Ginés. PHP 5 a través de ejemplos. 1ra Edición. Editorial Alfaomega. Junio 2005. México.
- Welling, Luke / Thomson, Laura. Desarrollo web con PHP y MySQL. Traducción de la 3ra Edición en inglés. Editorial Anaya Multimedia. 2005. Madrid, España.
- Gil Rubio / Francisco Javier, Villaverde / Santiago Alonso. Creación de sitios web con PHP5. 1a. edición en español. Editorial McGraw Hill. Madrid, España. 2006.
- John Coggeshall. LA BIBLIA DE PHP 5. 1ra. Edición. Editorial Anaya Multimedia. Madrid, España 2005.