Práctica de Validación de Formularios en PHP

Descripción de la Práctica:

Se realiza la validación de formulario en PHP, aprovechando el uso de variables, funciones y arrays. La temática elegida fue la creación de un formulario que consta de varios tipos de campos, como texto, radio, checkbox, lista de selección múltiple (select) y botones de enviar, borrar y confirmar. Se definen reglas de validación con funciones.

El objetivo es que al cargar la página, aparezca el formulario, con los tres botones, pero el de confirmar está deshabilitado, se rellenan los datos o se dejan campos vacíos y al darle a enviar aparezca encima de cada campo correspondiente, los errores si los hubiera. Solo se puede dar a enviar cuando no haya ningún error, y cuando eso pasa, entonces aparecen los datos enviados debajo del formulario, y cada campo recordará los datos, y además el botón de confirmar aparece habilitado.

Y al darle a confirmar, el formulario desaparece y en su lugar se ve una lista con los datos enviados y un link de volver para volver al formulario.

Pasos Realizados:

1. **Apertura HTML y línea de estilo:**

```
<!DOCTYPE html>
<html>

<head>
    <title>Ejemplo de Formulario</title>
    <!-- línea de estilo color rojo, que luego aplico con span en el lugar correspondiente -->
    <style> .error{color: red;} </style>
</head>

<body>

```

2. **Abro PHP, genero tabla con los datos servidor y cabeceras:**

   Los datos de servidor se obtienen con variable $_SERVER y añadiendo el tipo de dato que quiero obtener. En este caso muestro el nombre de servidor, la ip remota, el protocolo y el servidor.

   Y los datos de cabeceras de solicitud HTTP se obtiene con apache_request_headers() y como se considera un array, lo recorro con foreach y voy añadiendo cada campo en una celda y cada contenido en otra. Y muestro las cabeceras que contengan user- y accept:

```php
<?php
//creo tabla para poner los datos de servidor, uno por cada fila:
echo "<table border='1'>";
    echo "<tr><th>Datos servidor: </th></tr>";
    // Mostrar datos del servidor:
    echo "<tr><td>Nombre Servidor: </td><td>" . $_SERVER['SERVER_NAME'] . "</td></tr>";
    echo "<tr><td>IP remota: </td><td>" . $_SERVER['REMOTE_ADDR'] . "</td></tr>";
    echo "<tr><td>Protocolo: </td><td>" . $_SERVER['SERVER_PROTOCOL'] . "</td></tr>";
    echo "<tr><td>Servidor / Software: </td><td>" . $_SERVER['SERVER_SOFTWARE'] . "</td></tr>";
echo "</table>";
    echo "<br>";
// creo tabla para poner las cabeceras:
echo "<table border='1'>";
    echo "<TR><TH>Cabeceras: </TH></TR>";
    // Mostrar cabeceras que contienen "User-" o "Accept"
    $cabecera = apache_request_headers();
    foreach ($cabecera as $campo => $contenido) {
        if (substr_count($campo, "User-") != 0) {
            echo "<TR><TD>$campo</TD><TD>$contenido</TD></TR>";
        }//if
        if (substr_count($campo, "Accept") != 0) {
            echo "<TR><TD>$campo</TD><TD>$contenido</TD></TR>";
        }//if
    }//foreach
echo "</table>";
```

3. **Creación variables:**

Se crean variables de estado, como $enviado y $confirmado, para determinar si el formulario se ha enviado o confirmado, y les asigno el valor que solicito al mismo formulario de la siguiente manera:

```php
    //creo variables que usaré si se ha dado al botón de enviar o confirmar:
    $enviado = false;
    $confirmado = false;

    // Comprobar si se envió el formulario y cambio variable.
    if ($_SERVER["REQUEST_METHOD"] == "POST"){
        $enviado = true;}

    // Comprobar si se ha confirmado y cambio variable.
    if (isset($_REQUEST['confirmar'])){
        $confirmado = true;}
```

Genero la función si_existe para obtener valores de variables de solicitud con un valor por defecto si no existen. Esta función recibe dos parámetros, uno como clave que será un campo, y otro un valor por defecto.

Creo variables, que corresponderán a cada campo del formulario, y les asigno el valor llamando a la función si_existe, pasando por parámetro el value de cada campo como valor, y un texto o array vacío como valor por defecto:

```php
// Función para obtener un valor de una variable, con un valor por defecto si no existe.
function si_existe($clave, $valor_defecto) {
    //si recibe dato, se asigna ese dato y si no el valor por defecto.
    return isset($_REQUEST[$clave]) ? $_REQUEST[$clave] : $valor_defecto;
}

//Creo variables con los valores obtenidos de los campos del formulario (llamando a la función).
$nombre = si_existe("nombre", "");
$edad = si_existe("edad", 0);
$email = si_existe("email", "");
$comida = si_existe("comida", "");
$hobbies = si_existe("hobbies", []);
$ciudades = si_existe("ciudades", []);
?>
```

4. **Visualización de Datos y creación Formulario:**

Antes de crear formulario, como queremos que cuando se dé al botón de confirmar no se visualice, primero ponemos la condición de: si está confirmado, y mostramos solo los datos, llamando a cada variable de cada campo y mostrando un link de volver:

```php
<h2>Formulario </h2>
<!-- Si se ha dado a confirmar, se muestran los datos del formulario
y un link para volver al propio formulario. -->
<?php if($confirmado){ ?>
    <h3>Datos Confirmados:</h3>
        <p>Nombre: <?php echo $nombre; ?></p>
        <p>Edad: <?php echo $edad; ?></p>
        <p>Email: <?php echo $email; ?></p>
        <p>Comida: <?php echo $comida; ?></p>
        <p>Hobbies: <?php echo implode(', ',$hobbies); ?></p>
        <p>Ciudades: <?php echo implode(', ',$ciudades); ?></p>
        <a href="p2-ud2-version2.php">[Volver]</a>
<?php }
```

Y ahora, si no se ha confirmado, por lo tanto al iniciar la página, se crea el formulario.

Se creó el formulario HTML con campos de texto para nombre, edad e email, radio para comida preferida, checkbox para hobbies y una lista de selección múltiple para ciudades visitadas.

Importante: encima de cada campo se llama a la función validar que **explicamos más adelante,** para que salga un mensaje de error correspondiente en el caso de que no valide, además a este mensaje le aplicamos el estilo de error para que salga de color rojo:

```php
// Y si no se ha dado a confirmar, se muestra el formulario, es decir al iniciar la página:
if (!$confirmado){ ?>
<!-- Creo formulario con action al propio formulario -->
    <form action="<?php echo $_SERVER['PHP_SELF'] ?>" method="post">
        <!-- Antes de cada campo, aplico estilo error con etiqueta span si el formulario se ha enviado
        y hay mensajes de error. -->

        <!-- Creo tres campos de tipo texto -->
        <!-- como valor, aplico lógica de si recibe datos, tenga ese dato y si no campo vacío,
        para que recuerden los datos. -->
        <?php if ($enviado){ echo "<span class='error'>" . validarNombre($nombre) . "</span><br>"; } ?>
```

En el código se explica cómo pongo el valor de cada tipo de campo y cómo hacer para que el formulario recuerde los datos. ***Muestro capturas con comentarios:***

- Campos de texto:

```php
<!-- como valor, aplico lógica de si recibe datos, tenga ese dato y si no campo vacío,
para que recuerden los datos. -->
<?php if ($enviado){ echo "<span class='error'>" . validarNombre($nombre) . "</span><br>"; } ?>
<label for="nombre">Nombre: </label>
<input type="text" name="nombre" value="<?php echo isset($_POST['nombre']) ? $_POST['nombre'] : '' ; ?> ">
<br><br>
<?php if ($enviado){ echo "<span class='error'>" . validarEdad($edad) . "</span><br>"; } ?>
<label for="edad">Edad: </label>
<input type="text" name="edad" value="<?php echo isset($_POST['edad']) ? $_POST['edad'] : ''; ?> ">
</br></br>
<?php if ($enviado){ echo "<span class='error'>" . validarEmail($email) . "</span><br>"; } ?>
<label for="email">Email: </label>
<input type="text" name="email" value="<?php echo isset($_POST['email']) ? $_POST['email'] : ''; ?> ">
</br></br>
```

- Radio:

```php
<!-- Creo 4 input de tipo radio, solo se selecciona 1 opción -->
<!-- en cada uno, si recibe dato y si su dato corresponde con su value, el campo será checked. -->
<?php if ($enviado){ echo "<span class='error'>" . validarComida($comida) . "</span><br>"; } ?>
<label>¿Qué prefieres?:</label>
<input type="radio" name="comida" value="pizza" <?php if (isset($_POST['comida']) && $_POST['comida'] == 'pizza') echo 'checked'; ?>>
<label for="pizza">Pizza</label>
<input type="radio" name="comida" value="hamburguesa" <?php if (isset($_POST['comida']) && $_POST['comida'] == 'hamburguesa') echo 'checked'; ?>>
<label for="hamburguesa">Hamburguesa</label>
<input type="radio" name="comida" value="croquetas" <?php if (isset($_POST['comida']) && $_POST['comida'] == 'croquetas') echo 'checked'; ?>>
<label for="croquetas">Croquetas</label>
<input type="radio" name="comida" value="coliflor" <?php if (isset($_POST['comida']) && $_POST['comida'] == 'coliflor') echo 'checked'; ?>>
<label for="coliflor">Coliflor</label>
</br></br>
```

- Checkbox:

```php
<!-- creo 5 campos checkbox, que serán array y multiselección -->
<!-- en cada uno, si en el array está su value, entonces será checked (para recordar el dato) -->
<?php if ($enviado){ echo "<span class='error'>" . validarHobbies($hobbies) . "</span><br>"; } ?>
<label>Hobbies:</label>
    <input type="checkbox" name="hobbies[]" value="Deportes" <?php if (in_array('Deportes', $hobbies)) echo 'checked'; ?>>
    <label for="hobby1">Deportes</label>
    <input type="checkbox" name="hobbies[]" value="Lectura" <?php if (in_array('Lectura', $hobbies)) echo 'checked'; ?>>
    <label for="hobby2">Lectura</label>
    <input type="checkbox" name="hobbies[]" value="Viajes" <?php if (in_array('Viajes', $hobbies)) echo 'checked'; ?>>
    <label for="hobby3">Viajes</label>
    <input type="checkbox" name="hobbies[]" value="Música" <?php if (in_array('Música', $hobbies)) echo 'checked'; ?>>
    <label for="hobby4">Música</label>
    <input type="checkbox" name="hobbies[]" value="Cocina" <?php if (in_array('Cocina', $hobbies)) echo 'checked'; ?>>
    <label for="hobby5">Cocina</label>
</br></br>
```

- select-opction:

```php
<!-- creo campos select con 5 opctions, serán multiseleccion y tipo array -->
<!-- en cada uno, si en el array está su value, entonces será selected (para recordar el dato) -->
<?php if ($enviado){ echo "<span class='error'>" . validarCiudades($ciudades) . "</span><br>"; } ?>
<label for="ciudades">Ciudades que has visitado:</label>
<select name="ciudades[]" multiple>
    <option value="Nueva York" <?php if (in_array('Nueva York', $ciudades)) echo 'selected'; ?>>Nueva York</option>
    <option value="Boston" <?php if (in_array('Boston', $ciudades)) echo 'selected'; ?>>Boston</option>
    <option value="Londres" <?php if (in_array('Londres', $ciudades)) echo 'selected'; ?>>Londres</option>
    <option value="Roma" <?php if (in_array('Roma', $ciudades)) echo 'selected'; ?>>Roma</option>
    <option value="AtenasV" <?php if (in_array('AtenasV', $ciudades)) echo 'selected'; ?>>Atenas en Verano</option>
    <option value="AtenasI" <?php if (in_array('AtenasI', $ciudades)) echo 'selected'; ?>>Atenas en Invierno</option>
</select>
</br></br>
```

También se incluyeron botones para enviar y borrar el formulario.

```
<!-- y creo boton sumbit de enviar y otro de reset para borrar datos. -->
<input type="submit" value="Enviar">
<input type="reset" value="Borrar Formulario">
```

5. **Validación de formulario con restricciones:**

Para la validación, he utilizado funciones, una por cada campo, que reciben por parámetro la variable de cada campo. Devuelven un mensaje de error en caso de que lo sea y cadena vacía si no hay error.

Indico restricciones aplicadas y funciones usadas y , **muestro captura del código con más explicaciones:**

- campos de texto nombre: para decir que el campo es obligatorio uso **empty**, y para la longitud de caracteres uso **strlen**, función de string que indica la longitud de la cadena:

```
//validar nombre: obligatorio, min 3 caracteres y max 10 caracteres..
function validarNombre($nombre){
    //antes de leer el campo, elimino espacios al principio y al final con la función trim.
    $nombre = trim($nombre);
    if (empty($nombre)){
        return "Es obligatorio";
    }elseif (strlen($nombre) > 10 || strlen($nombre) < 3){
        return "Tienen que tener entre 3 y 10 carácteres.";
    }else{
        return "";
    }
}//valNombre
```

- campo de texto edad: uso **is_numeric** para indicar que sea un número y operadores de mayor y menos para el rango:

```
//validar edad: campo numérico y entre 18 y 70.
function validarEdad($edad){
    $edad = trim($edad);
    if (!is_numeric($edad)) {
        return "Pon un número";
    } elseif ($edad < 18 || $edad > 70) {
        return "La edad deber estar entre 18 y 70.";
    } else {
        return "";
    }
}//valEdad
```

- campo texto email: uso función de string **substr_count** para buscar la cadena que quiero y así obligar que contenga @ y .com.

```php
//validar email: debe tener @ y .com.
function validarEmail($email){
    $email = trim($email);
    if (substr_count($email, "@") == 0 || substr_count($email, ".com") == 0) {
        return "Pon un email válido.";
    } else {
        return "";
    }
}//valEmail
```

- radio comida: también uso **substre_count** para indicar si el valor es coliflor:

```php
//validar comida: coliflor campo prohibido.
function validarComida($comida){
    if (substr_count($comida, "coliflor") != 0){
        return "No puedes seleccionar coliflor.";
    }else{
        return "";
    }
}//valComida
```

- checkbox hobbies: al ser array, uso count para indicar que hobbies tenga más de dos valores:

```php
//validar hobbies: mínimo 2.
function validarHobbies($hobbies){
    if (count($hobbies) <2){
        return "Selecciona al menos 2.";
    } else{
        return "";
    }
}//valHobbies
```

- **opción multiselect ciudades:** uso **in_array** para comparar dos valores y así que sean opciones incompatibles:

```php
//validar ciudades: atenasV y atenasI no son compatibles.
function validarCiudades($ciudades){
    if(in_array('AtenasV', $ciudades) && in_array('AtenasI',$ciudades)){
        return "Atenas en Verano y Atenas en Invierno son incompatibles.";
    } else{
        return "";
    }
}//valCiudades
```

6. **Mensajes de Error y Visualización:**

   Ahora creo variable de tipo array para almacenar los errores que obtengo llamando a cada función validar pasando por parámetro la variable de cada campo, y elimino cadenas vacías con **array_filter** ya que las funciones de validación devuelven tanto texto de error como texto vacío y de esta manera el array solo contendrá los mensajes de error.

```php
// Crear un array de errores con las funciones de validar y eliminar cadenas vacías:
$errores = array();
$errores[] = validarNombre($nombre);
$errores[] = validarEdad($edad);
$errores[] = validarEmail($email);
$errores[] = validarComida($comida);
$errores[] = validarHobbies($hobbies);
$errores[] = validarCiudades($ciudades);
// elimina las cadenas vacías:
$errores = array_filter($errores);
```

   Y estos errores se visualizan encima de cada campo del formulario, **se puede ver en las capturas anteriores.**

7. **Validación correcta:**

   Cuando la validación es correcta, es decir, cuando se ha dado a enviar y no hay errores, se muestra un resumen de los datos procesados, incluyendo Nombre, Edad, Email, Comida, Hobbies y Ciudades visitadas. En el caso de hobbies y ciudades, al ser array, para mostrar todos los valores seleccionados hay que poner implode, con un primer parámetro de separación entre valores, y un segundo parámetro con el nombre del array.

   Creo que esto se puede considerar cálculo de texto.

```php
// Muestro los datos si se ha dado a enviar y si no hay errores (si el array errores está vacío.)
if ($enviado = true && (empty($errores))){
    echo "<br><h3>Has enviado los siguientes datos: </h3>
    Nombre: $nombre<br>
    Edad: $edad<br>
    Email: $email<br>
    Comida: $comida<br>
    Hobbies: " . implode(', ', $hobbies) . "<br>
    Ciudades: " . implode(', ', $ciudades) . "<br><br><br>";
}//if
```

8. **Botón confirmar:**

   Por último he añadido el botón de confirmar, cuya función es que se deje de mostrar el formulario y aparezca en su lugar los datos confirmados/enviados y un link para volver al formulario.

Este botón está deshabilitado a menos que se envíe el formulario y no haya errores. Esto se especifica indicando según el caso, disabled:

```
<!-- creo botón de confirmar y cierro formaulario -->
<!-- indico que si no se ha enviado y no hay errores, el botón está inactivo-->
<input type="submit" name="confirmar" value="Confirmar" <?php echo $enviado=true && (empty($errores)) ? '' : 'disabled'; ?>>
orm>
```

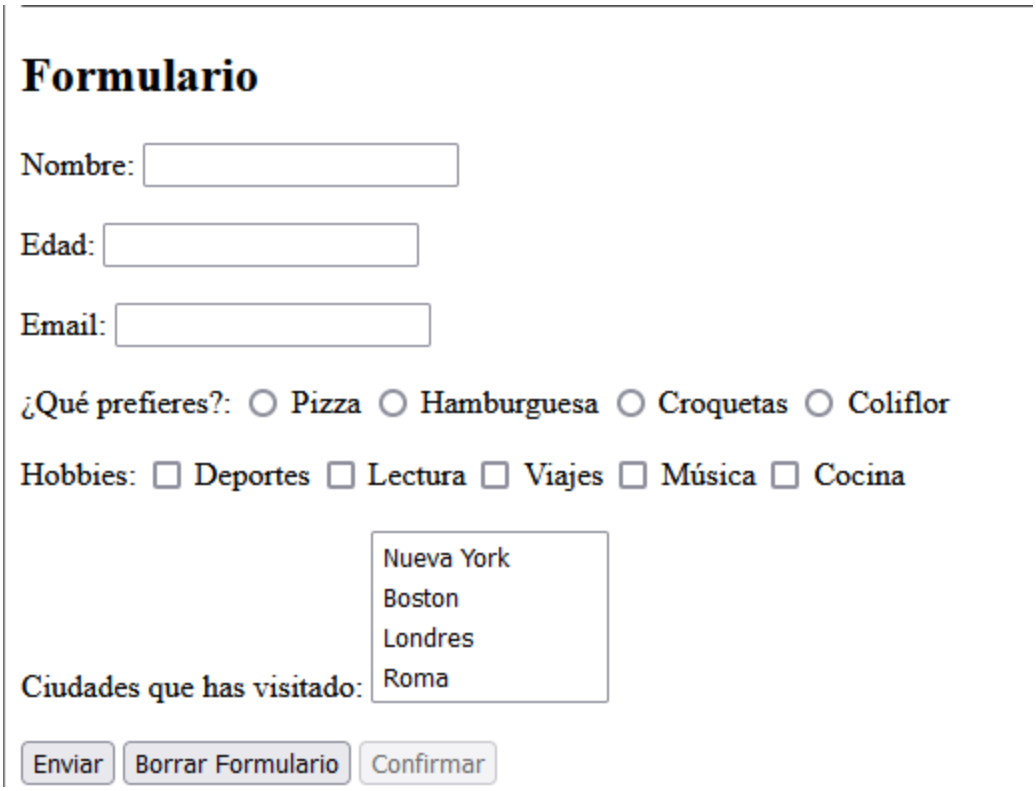
PRUEBAS DE EJECUCIÓN

Al iniciar la página se ve encima cabeceras y datos de servidor:

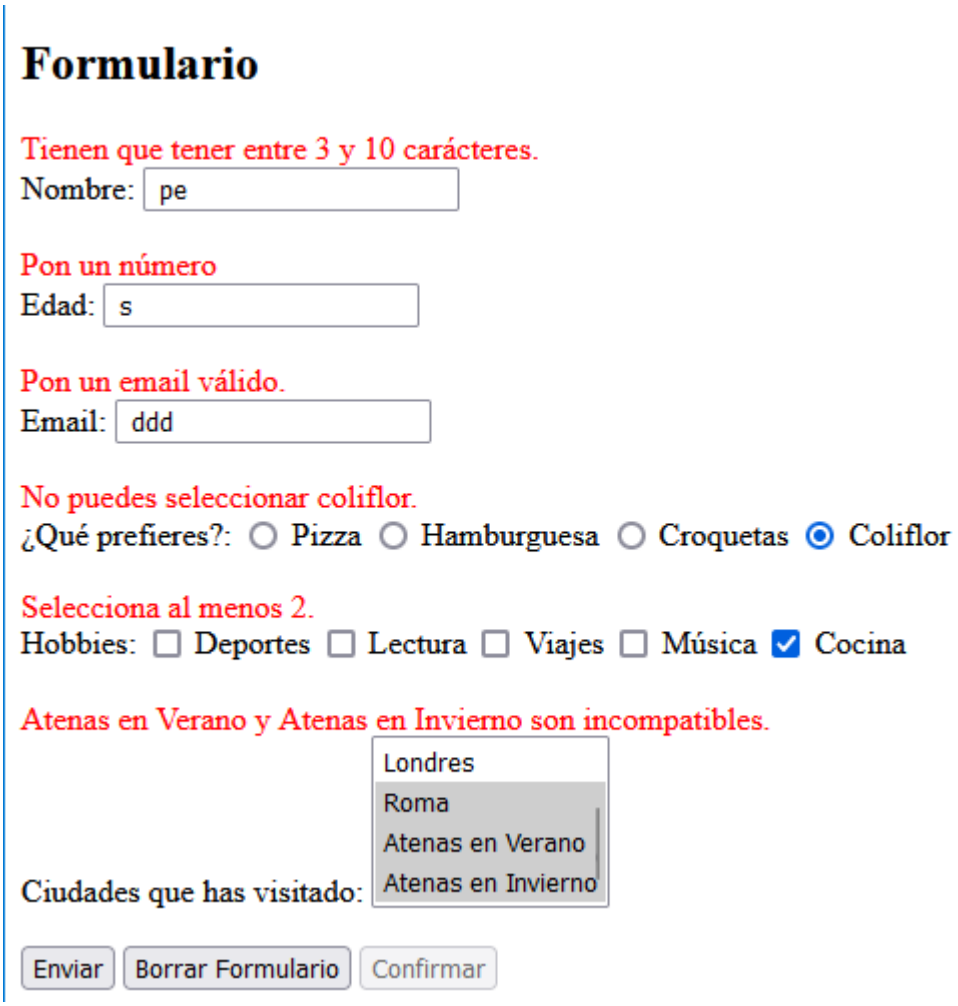
Datos servidor:	
Nombre Servidor:	localhost
IP remota:	127.0.0.1
Protocolo:	HTTP/1.1
Servidor / Software:	Apache/2.4.56 (Win64) OpenSSL/1.1.1t PHP/8.2.4

Cabeceras:	
User-Agent	Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/118.0
Accept	text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language	es-ES,es;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding	gzip, deflate, br

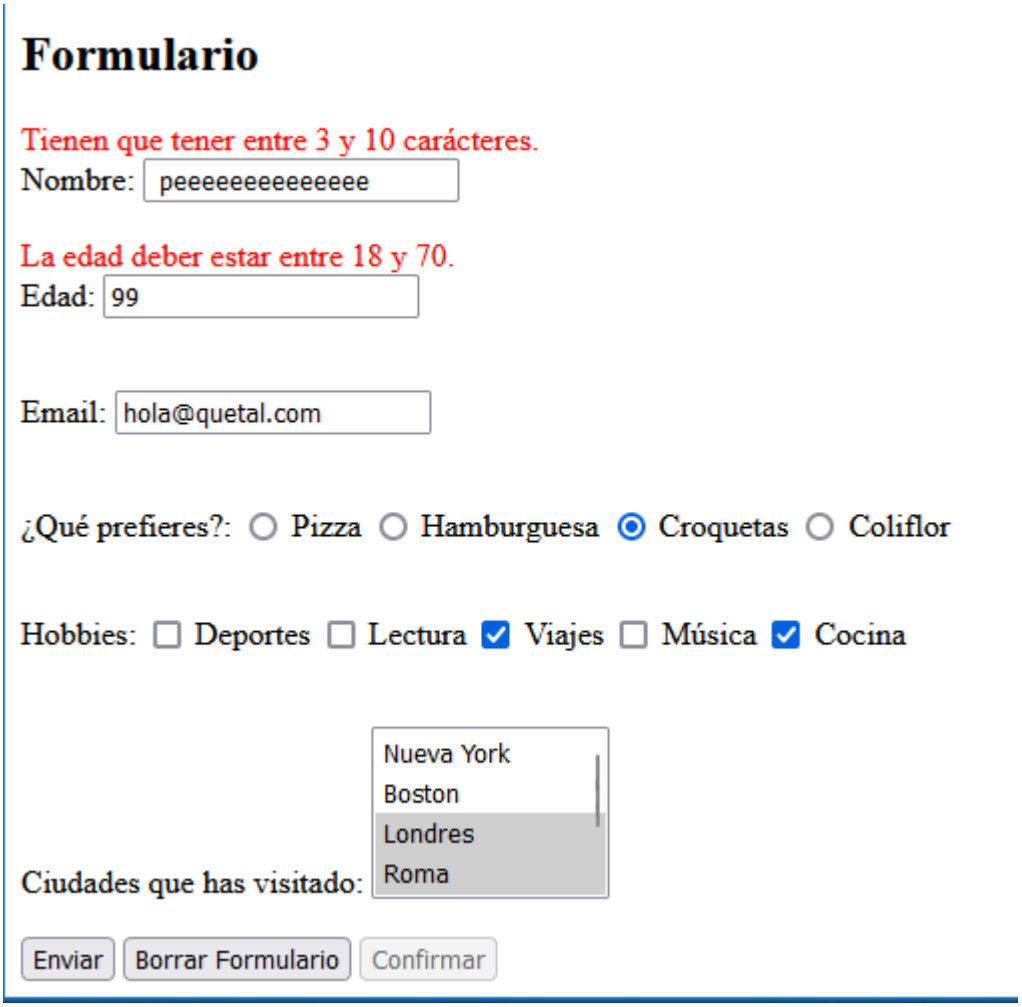
Y luego el formulario con el botón confirmar deshabilitado:



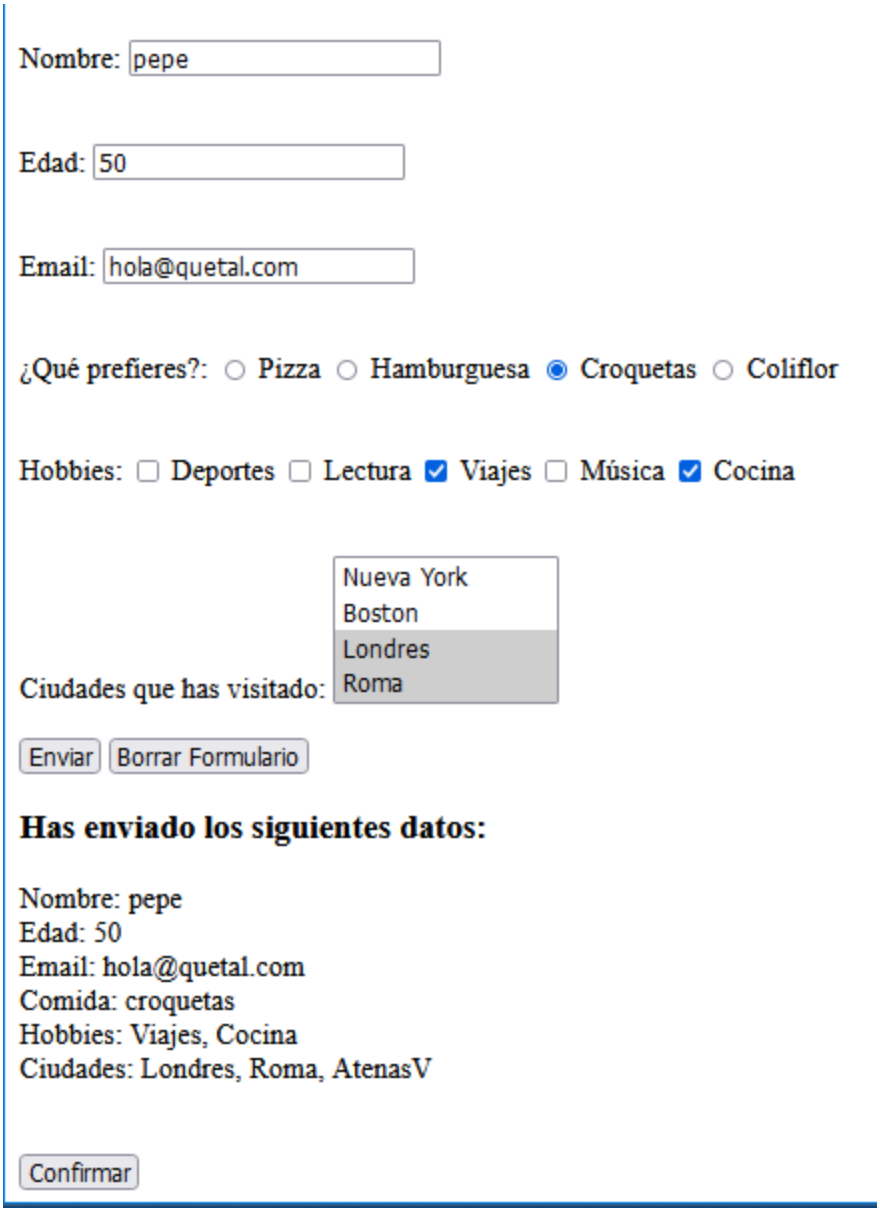
Probamos añadiendo datos erróneos y vemos los mensajes de error en rojo y encima de cada campo, y el botón de confirmar sigue deshabilitado:



Probamos con otros errores:



Y ahora con valores correctos y vemos que no salen errores y que al final sale un resumen de los datos enviados, que el formulario recuerda los valores y que el botón de confirmar ya está habilitado:



Y al darle a confirmar salen los datos que hemos confirmado y el enlace para volver al formulario y volver a rellenar datos:

Accept	text/html,application/xhtml+xml,application/xml;q=0.9,ima
Accept-Language	es-ES,es;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding	gzip, deflate, br

Formulario

Datos Confirmados:

Nombre: pepe

Edad: 50

Email: hola@quetal.com

Comida: croquetas

Hobbies: Viajes, Cocina

Ciudades: Londres, Roma, AtenasV

[Volver]

Has enviado los siguientes datos:

Nombre: pepe

Dificultad y Problemas Encontrados:

- Uno de los desafíos más importantes fue aplicar reglas de validación para cada campo del formulario, que hice con funciones según la recomendación del enunciado, una vez que lo comprendí como aplicarla, ahora me resulta más fácil, pues ahora entiendo fácilmente que al llamar a la función de validación, devuelve mensaje de error si no se valida.
- De lo que más me costó fue como validar los datos en el campo radio, hasta que al final di con la función substr_count, que seguro que hay otra más sencilla. De este campo radio también me costó el cómo recordar el check en el formulario, sabía que tenía que usar el checked pero no cómo, hasta que di con la solución, ahora entiendo la lógica de “si recibe dato y siese dato es su valor, entonces cheked”.
- También tuve problemas con errores de sintaxis, que me resultaron difíciles de encontrar pero fáciles de solucionar. Es lo malo de que en php es difícil encontrar los errores.
- Otros problemas encontrados fue sobre el orden del código, al querer realizar algunas funciones y no aplicarse por estar debajo de un código. Reconozco que lo he solucionado probando en un sitio y en otro sin terminar de entenderlo bien.

Conclusiones:

Lo más importante a decir, fuera del ejercicio en sí, es que creo que ha sido un ejercicio muy exigente para conseguir buena nota, requería de mucha dificultad, pues es muy técnico, y sobretodo muchas horas de dedicación. Además de las 5 o 6 horas de clase, he estado dos tardes enteras en casa para poder hacer un ejercicio, creo que muy completo y bastante bueno, pero aún así me faltan los ejercicios del envío y descarga de archivos para logar el excelente según los criterios de evaluación.

Aunque me falten estas dos últimas cosas, estoy contenta con el ejercicio, pues he conseguido el resto de puntos:
- mensajes de error en rojo encima de cada campo.

- varios campos adicionales y restricciones varias.

- muestra de las cabeceras y datos de servidor.

- uso de funciones de validación.

- uso de función si_existe con clave valor.

- que el formulario recuerde los datos introducidos.

-validaciones correctas y resumen de los datos.

- botón de confirmar datos con su función correcta de no mostrar formulario y si mostrar datos y enlace de volver.

- y creo que cálculo sobre texto que se pide en el punto 6.3.

A parte de esto, al haber dedicado tantas horas a este ejercicio, creo que he aprendido bastante sobre funciones y código en general, también recordar el uso de formularios y de los distintos campos. Me ha gustado aplicar las restricciones y ver que funcionaban. Quedo muy satisfecha con el trabajo realizado y con la correcta aplicación del código.