2º Desarrollo de Aplicaciones Web
Curso 2014/2015

Desarrollo Web en Entorno Servidor

Práctica 4
Carta de Restaurante

Salvador Camacho Soto

1.- Enlaces web.

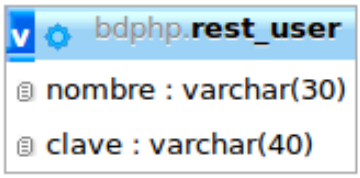
Enlace a la página web: http://salvacam.x10.mx/
Enlace a la práctica, front-end: http://salvacam.x10.mx/practica4/
Enlace a la práctica, back-end: http://salvacam.x10.mx/practica4/administrador
Enlace a GitHub: https://github.com/salvacam/Practica-4

2.- Requerimientos.

Crear una aplicación web para la gestión de los platos de la carta de un restaurante. Con un front-end desde donde se pueden ver todos los platos, con su imagen principal y una página para cada plato donde se ven todos los detalles del plato y todas las fotos. En el back-end podemos crear nuevos platos, editar y borrar los existentes, todo el back-end mediante AJAX.

3.- Implementación.

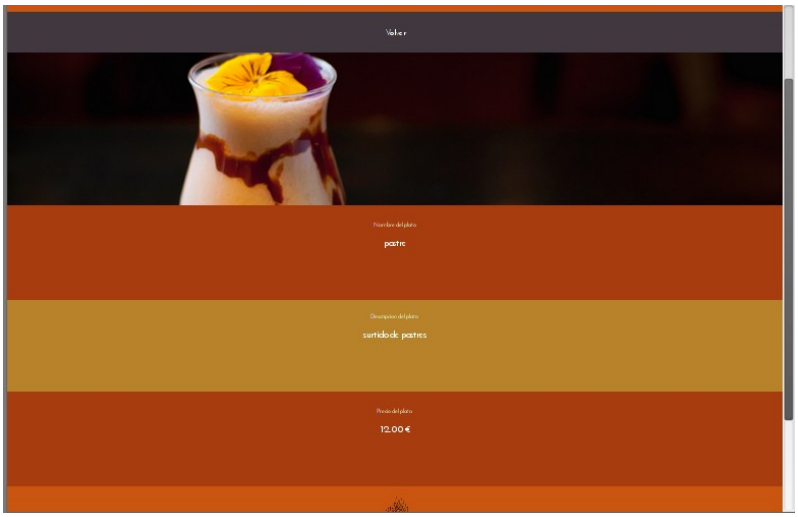
En la base de datos usare dos tablas.



La tabla rest_*user* contiene los datos de los usuarios que pueden acceder al back-end y la tabla rest_*plato* contiene los datos de cada plato.

Para el *front-end* tengo una página web *index.php* en la raíz del proyecto, donde se ven todos los platos.

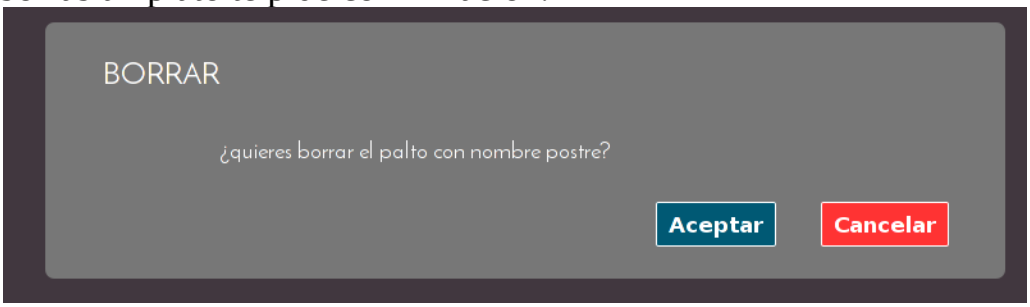
Desde la página index.php puedo acceder a otra página donde me muestra toda la información y fotos del plato al cual le haya hecho clic.



Para el *back-end* tengo una página, donde tengo todo los elementos del html y los voy mostrando y ocultando según necesidades. Para acceder tengo que creado un usuario (root) con contraseña (toor).

Una vez introducido un usuario y su contraseña, muestro una *tabla* con los datos de los platos, un botón para *desconectarse* y otro para *insertar* un nuevo plato.

Cuando borras un plato te pide confirmación.



Cuando insertas un nuevo plato se muestra un formulario con los datos necesarios para crear un nuevo plato.

El formulario de editar es el mismo con los campos y fotos del plato.

4.- Código

Como en las prácticas anteriores para cada tabla uso dos clases, una con el nombre de la tabla y otra cuyo nombre es ModeloNombre de la tabla, con lo tenga 4 clases para las tablas de la base de datos: User, ModeloUser, Plato y ModeloPlato.

En la página principal del proyecto muestro cada plato con su imagen principal, si estuviera definido en la tabla o no existiera la imagen le asigno una por defecto.

```
$ruta = "./fotos/" . $objeto->getId() . "/" . $objeto->getFoto();
        if ($objeto->getFoto() != NULL && file_exists($ruta)) {
            $ruta = "./fotos/" . $objeto->getId() . "/" . $objeto->getFoto();
        } else {
            $ruta = "./fotos/defecto.jpg";
        }
```

Para el campo precio filtro las teclas para que solo escriba números y un punto.

```
  function filtroNumero(e) {
     var codigo = e.charCode || e.keyCode;
     if (codigo < 32 || codigo == 37 || codigo == 38 || codigo == 39 || codigo == 40)
        return;
     var texto = String.fromCharCode(codigo);
     var permitidos = "0123456789";
     if (permitidos.indexOf(texto) === -1 && codigo != 46) {
        if (e.preventDefault)
           e.preventDefault();
        return false;
     } else if (e.target.value.indexOf('.') >= 0 && codigo == 46) {
        if (e.preventDefault)
           e.preventDefault();
        return false;
     }
  }
```

El archivo *back.js* es el que contiene todas las llamadas ajax. Al acceder a la página del *back-end* lo primero que se hace es comprobar si ya esta conectado. Si estuviera conectado muestra la tabla con los platos y no el formulario de login.

```
  function conexion() {
     var ajax = new Ajax();
     ajax.setUrl("ajaxConectado.php");
     ajax.setRespuesta(conectar);
     ajax.doPeticion();
  }
```

El archivo *ajaxConectado.php* devuelve *'r': 1* si esta conectado ó *'r':0* si no lo esta.

```
header('Content-Type: application/json');
if($sesion->getUsuario() !== null)
   echo '{"r": 1}';
else
   echo '{"r": 0}';
```

En el archivo *back.js* tengo el método *logearDatos*. Comprueba si el usuario es correcto, si es correcto muestra la tabla con los platos y oculta el formulario de login.

```
function logearDatos() {
      var nombreUser = document.getElementById("nombreUser").value;
      var clave = document.getElementById("clave").value;
      if (nombreUser !== "" && clave !== "") {
         var ajax = new Ajax();
         ajax.setUrl("ajaxGet.php?login=" + nombreUser + "&clave=" + clave);
         ajax.setRespuesta(loguear);
         ajax.setRespuestaError(mostrarMensaje("No se puede intentar conectar"));
         ajax.doPeticion();
      }
      document.getElementById("nombreUser").value = "";
      document.getElementById("clave").value = "";
   }
```

El archivo *ajaxGet.php* devuelve *'r': 1* si existe el usuario y ejecuta el método *setUsuario* de sesión o *'r':0* si no existe el usuario.

```
$aux = $modelo->get($login,$clave);
if ($aux->getNombre() === null) {
   echo '{"r": 0}';
} else {
   $sesion->setUsuario($aux);
   echo '{"r": 1}';    }
```

Cuando se hace click en el botón desconectar se hace la llamada ajax a ajaxLogout.php, que cierra la sesión y ejecuta la función desloguear, que muestra un mensaje, oculta la tabla y vuelve a mostrar el formulario de login.

```
   function desloguear() {
         desconectarUsuario();
         mostrarMensaje("Usuario desconectado");
}
```

La funcion *cargarPagina* del archivo *back.js*, hace una llamada ajax al archivo *ajaxSelect.php*, que devuelve (si estas conectado) el resultado del select en json y el array de paginación, y se le pasa a la función *mostrarTabla*.

```
header('Content-Type: application/json');
if ($sesion->getUsuario() !== null) {
   $bd = new BaseDatos();
   $modelo = new ModeloPlato($bd);
   $pagina = 0;
   if (Leer::get("pagina") != null)
      $pagina = Leer::get("pagina");
   $total = $modelo->count();
   $enlaces = Paginacion::getEnlacesPaginacionJSON($pagina, $total[0], 2);
   echo '{"paginas":' . json_encode($enlaces) . ',"platos":' . $modelo->getListJSON($pagina, 2,
"1=1", array(), "id DESC") . '}';
  $bd->closeConexion();
}
```

Todos los elementos de la clase *borrar* se los paso a la función *agregarEventoBorrar*, la cual coge el *id* y el *nombre* del plato y se lo pasa a la función *mostrarConfirm*.

```
var enlacesBorrar = document.getElementsByClassName("borrar");
      for (var i = 0; i < enlacesBorrar.length; i++) {
         agregarEventoBorrar(enlacesBorrar[i]);
      }

function agregarEventoBorrar(elemento) {
      var id = elemento.getAttribute("data-id");
      var name = elemento.getAttribute("data-nombre");
      elemento.addEventListener("click", function (e) {
         mostrarConfirm(id, name);
      });
   }
```

La función *mostrarConfirm* oculta la tabla y muestra un *div* con los botones para aceptar o cancelar el borrado.

```
   function mostrarConfirm(id, name) {
         MDtitle.textContent = "Borrar";
         MDcontent.textContent =
               "¿quieres borrar el palto con nombre " + name + "?";
         var cancelado = "Borrado cancelado";

      modaldialog.setAttribute("class", "flex");
      login.setAttribute("class", "oculto");
      tabla.setAttribute("class", "oculto");
      form.setAttribute("class", "oculto");
      btMYes.onclick = function () {
         borrar(id);
         ocultarConfirm();
      };
      btMNo.onclick = function () {
         mostrarMensaje("Borrado cancelado");
         ocultarConfirm();
      };
   }
```

Si acepta el borrar el plato se le pasa la *id* a la funcion *borrar*, que hace la llamada ajax a *ajaxDelete.php*.

```
   function borrar(id) {
      var ajax = new Ajax();
      ajax.setUrl("ajaxDelete.php?id=" + id + "&pagina=" + paginaActual);
      ajax.setRespuesta(borrarResul);
      ajax.doPeticion();
   }
```

Ajaxdelete.php intenta borrar el plato en la base de datos y si lo borra elimina la carpeta y su contenido del plato

```
$r = $modelo->deletePorId($id);
   if ($r == 1) {
      $total = $modelo->count();
      $paginas = ceil($total[0] / 2) - 1;
      if ($pagina > $paginas) {
         $pagina = $paginas;
      }
      //borrar carpeta
      $ruta = "../fotos/" . $id;
      if(file_exists($ruta)){
         foreach (scandir($ruta) as $valor) {//readdir  scandir
            if ($valor == '.' || $valor == '..')
               continue;
            unlink($ruta . DIRECTORY_SEPARATOR . $valor);
         }
         rmdir($ruta);
      }
      $enlaces = Paginacion::getEnlacesPaginacionJSON($pagina, $total[0], 2);
      echo '{"r": 1,"paginas":' . json_encode($enlaces) . ',"platos":' . $modelo->getListJSON($pagina,
2) . '}';
      $bd->closeConexion();
      exit();
   }
   $bd->closeConexion();
   echo '{"r":0}';
```

Al insertar un nuevo plato o editar uno existente compruebo que el nombre no exista. Al cambiar el valor del campo *nombre*, si esta vació le muestro un mensaje al lado del input, si no hace la llamada ajax a *ajaxComprobarNombre.php*

```
      nombre.addEventListener("blur", function () {
         if (nombre.value == "") {
            nombreValido.textContent = "Necesario un nombre";
         } else {
            var ajax = new Ajax();
            ajax.setUrl("ajaxComprobarNombre.php?nombre=" + nombre.value);
            ajax.setRespuesta(comprobarNombre);
            ajax.doPeticion();
         }
      });
```

Si el nombre existe muestro un mensaje al lado del input, si no no muestro nada.

```
   var comprobarNombre = function comprobarNombre(text) {
      var ojson = JSON.parse(text);
      if (ojson.r == 1)
         nombreValido.textContent = "Nombre ya en uso";
      else
         nombreValido.textContent = "";
   };
```

Cuando esta validado el formulario para insertar un plato, se realiza la llamada ajax al archivo *ajaxInsertar.php* y muestra un mensaje según el resultado.

```
ajax = new XMLHttpRequest();
                if (ajax.upload) {
                   ajax.open("POST", "ajaxInsertar.php", true);
                   ajax.onreadystatechange = function () {
                      if (ajax.readyState == 4) {
                         if (ajax.status == 200) {
                            var ojson = JSON.parse(ajax.responseText);
                            if (ojson.r == 0) {
                               mostrarMensaje("Insertado correctamente");
                               ocultarForm();
                               limpiarForm();
                               cargarPagina(0);
                            } else if (ojson.r == -1) {
                               mostrarMensaje("Datos no validos");
                            } else if (ojson.r == -3) {
                               mostrarMensaje("No se ha podido insertado");
                            } else if (ojson.r == -2) {
                               mostrarMensaje("Insertado, con error al subir archivos");
                               ocultarForm();
                               limpiarForm();
                               cargarPagina(0);
                            }
                         } else {
                            mostrarMensaje("Error al intentar insertar");
                         }
                      }
                   };
                   ajax.send(parametros);
                }
```

El archivo *ajaxInsertar.php* intenta crear en la base de datos el plato, si lo hace crea la carpeta donde se guardan las imágenes y las sube.

```
  $r = $modelo->add(new Plato(null, $nombre, $descripcion, $precio));
  if ($r != -1) {
     $ruta = "../fotos/" . $r;
     if (!file_exists($ruta)) {
        mkdir($ruta, Configuracion::PERMISOS);
     }
     $principal = Leer::post("principal");

     if (isset($_FILES["archivo"])) {
        $errores = 0;
        if ($_FILES["archivo"]["error"] > 0) {
           foreach ($_FILES["archivo"]["error"] as $indice => $valor) {
              if ($valor == UPLOAD_ERR_OK) {
                 $tipo = explode("/", $_FILES["archivo"]["type"][$indice]);
                 $tamano = $_FILES["archivo"]["size"][$indice];
                 if ($tipo[0] == "image" && $tamano < 2097152) {   //0.5 megas "2097152"
                    $tmp = $_FILES["archivo"]["tmp_name"][$indice];
```

```
                    $name = $_FILES["archivo"]["name"][$indice];
                    $pos = strrpos($name, ".");
                    $ext = substr($name, $pos, 4);
                    move_uploaded_file($tmp, "../fotos/" . $r . "/foto" . ($indice + 1) . $ext);
                    if ($principal == $indice) {
                        $fotoPpal = "foto" . ($indice + 1) . $ext;
                        $objeto = $modelo->get($r);
                        $objeto->setFoto($fotoPpal);
                        $modelo->edit($objeto);
                    }
                } else
                    $errores ++;
            } else
                $errores ++;
        }
    }
    if ($errores == 0) {
        echo '{"r": 0}'; // se ha insertado y subido bien
        exit();
    } else {
        echo '{"r": -2}' . $errores; // se ha insertado pero  no se han subido bien los archivos
        exit();
    }
} else {
    echo '{"r": -3}'; // no se ha insertado
    exit();
}
}
```

Al editar un plato lo primero que se hace es una llamada ajax al archivo *ajaxGetObjeto.php*, que devuelve los datos del plato y los nombres de las imágenes

```
$ruta = "../fotos/" . $id;
$imagenes = array();
if (file_exists($ruta)) {
    $directorio = opendir($ruta);
    while ($archivo = readdir($directorio)) {
        if ($archivo != ".." && $archivo != ".")
            $imagenes[] = $ruta . "/" . $archivo;
    }
    sort($imagenes, SORT_STRING);
}
$resul = substr($modelo->getJSON($id), 0, -1);
$resul .= ',"fotos": {';
for ($i = 0; $i < sizeof($imagenes); $i++) {
    if ($i === 0)
        $resul .= '"foto' . ($i + 1) . '": "' . $imagenes[$i] . '"';
    else
        $resul .= ',"foto' . ($i + 1) . '": "' . $imagenes[$i] . '"';
}
echo $resul . '}}';
```

Después asigno los valores a los campos y muestro las fotos

```
 nombre.value = ojson.nombre;
 descripcion.value = ojson.descripcion;
 precio.value = ojson.precio;

for (i in ojson.fotos) {
                if (ojson.fotos[i] !== "") {
                   console.log(i);
                   num = i.substring(i.length - 1);
                   imgInput[num - 1].setAttribute("hidden", "");
                   var imgNew = document.createElement("img");
                   imgNew.setAttribute("src", ojson.fotos[i]);
                   imgNew.setAttribute("class", "miniatura");
                   borrarImg[num - 1].parentNode.insertBefore(imgNew, borrarImg[num - 1]);
                   ppal[num - 1].disabled = false;
                   reset[num - 1].value = "Borrar";
                }
             }
```

Si borro una foto se hace es una llamada ajax al archivo *ajaxBorrarFoto.php*.

```
 ajax = new XMLHttpRequest();
                    ajax.open("GET", "ajaxBorrarFoto.php?foto=" + previo.getAttribute("src"), true);
                    ajax.onreadystatechange = function () {
                       if (ajax.readyState == 4) {
                          if (ajax.status == 200) {
                             console.log(ajax.responseText);
                             var ojsonBorrar = JSON.parse(ajax.responseText);
                             if (ojsonBorrar.r == 1) {
                                e.target.value = "Limpiar";

e.target.previousElementSibling.parentNode.removeChild(e.target.previousElementSibling);
                                var radio = e.target.parentNode.nextElementSibling.lastElementChild;
                                radio.disabled = true;
                                radio.checked = false;

e.target.parentNode.firstElementChild.nextElementSibling.removeAttribute("hidden");
                             }
                          }
                       }
                    };
                    ajax.send();
```

Cuando esta validado el formulario para editar un plato, se realiza la llamada ajax al archivo *ajaxEditar.php* y muestra un mensaje según el resultado.

```
ajax.open("POST", "ajaxEditar.php", true);
                          ajax.onreadystatechange = function () {
                             if (ajax.readyState == 4) {
                                if (ajax.status == 200) {
                                   var ojson = JSON.parse(ajax.responseText);
                                   if (ojson.r == 0) {
```

```
                        mostrarMensaje("Editado correctamente");
                        ocultarForm();
                        limpiarForm();
                        cargarPagina(paginaActual);
                    } else if (ojson.r == -1) {
                        mostrarMensaje("Datos no validos");
                    } else if (ojson.r == -2) {
                        mostrarMensaje("No se ha podido editar");
                        ocultarForm();
                        limpiarForm();
                        cargarPagina(paginaActual);
                    }
                } else {
                    mostrarMensaje("Error al intentar insertar");
                }
            }
        };
        ajax.send(parametros);
```

5.- Recursos

Apuntes Desarrollo web en entorno servidor.
Apuntes Desarrollo web en entorno cliente.
Manual PHP: http://php.net/manual/es/index.php
Efecto Parallax:
http://code.tutsplus.com/tutorials/a-simple-parallax-scrolling-technique--net-27641
Limitar el tipo de archivo de un input file:
http://www.iteramos.com/pregunta/10447/formato-de-archivo-de-limite-al-utilizar-input-typefile-
Borrar el valor de un campo input file:
http://susitioweb.co/jquery/40-borrar-el-valor-de-un-campo-input-file-con-jquery