



UNIVERSIDAD  
DE GRANADA

*Este documento está protegido por la Ley de Propiedad Intelectual ([Real Decreto Ley 1/1996 de 12 de abril](#)).  
Queda expresamente prohibido su uso o distribución sin autorización del autor.*

# Tecnologías Web

## 3º Grado en Ingeniería Informática



### Proyecto Web Recetario

1. Objetivos del proyecto.....	2
2. La aplicación web.....	3
3. Información que debe gestionar el sistema.....	4
4. Descripción del sistema en función del tipo de usuario.....	6
5. Log de la aplicación.....	14
6. Listados de recetas.....	15
7. Otras utilidades.....	16
8. Instalación de la aplicación.....	17
9. Cuestiones genéricas sobre diseño e implementación.....	17
10. Aspectos principales de la evaluación.....	20
11. Entrega de la práctica.....	21

© Prof. Javier Martínez Baena  
Dpto. Ciencias de la Computación e I. A.  
Universidad de Granada

Imagen de portada obtenida de Freepik



Departamento de  
Ciencias de la Computación  
e Inteligencia Artificial

## 1. Objetivos del proyecto

Con este proyecto se pretende que el alumno aplique de forma práctica gran parte de los conocimientos adquiridos durante el desarrollo de la asignatura. Consistirá en el desarrollo de una aplicación web sencilla pero suficientemente completa de manera que el resultado final sea totalmente funcional y comparable a cualquier aplicación web real (sin hacer demasiado énfasis en aspectos estéticos). Dicha aplicación, en líneas generales, dispondrá de:

- Una interfaz totalmente funcional que cubra las necesidades básicas expuestas en este guión.
- Una base de datos que almacene información relacionada con la aplicación y que permita que el contenido del sitio sea dinámico.
- Una gestión básica de usuarios. El sitio web mostrará diferentes vistas dependiendo el tipo de usuario que lo visite en cada momento.

Algunos requisitos técnicos básicos que deberá incorporar:

- Gestión correcta de sesiones.
- Gestión de la base de datos mediante formularios web correctamente diseñados e implementados.
- Operaciones CRUD para mantener actualizada la base de datos, es decir, formularios de creación, lectura, actualización y borrado de registros que permitan la gestión completa del sistema desde la aplicación web.
  - Uso de formularios "sticky".
  - Peticiones de confirmación de operaciones CRUD al usuario.
  - Validación de datos en el servidor y, de forma secundaria, en el cliente.
- Uso de cookies.
- La implementación deberá cumplir con unos mínimos de calidad.

Como puede ver, para el éxito en el desarrollo de este proyecto va a necesitar, principalmente:

- HTML+CSS para el desarrollo del frontend.
- PHP para el desarrollo del backend.

Se planteará también el uso de JavaScript para el frontend, aunque su uso será menos relevante. El motivo no es otro que la temporización del temario de la asignatura. JavaScript se estudia al final del mismo y, por tanto, no queda demasiado tiempo para incorporarlo a este proyecto.

En internet podrá encontrar muchos sitios web con aplicaciones similares a la planteada. Puede usarlos como referencia para el diseño e incluso para la implementación.

Todas las capturas de pantalla, menús, etc de este guión son orientativos y el proyecto no tiene porqué parecerse en absoluto a ellos. Únicamente se pretende ilustrar la funcionalidad de la aplicación.

Se recomienda leer este documento por completo antes de iniciar la práctica para tener una visión global de lo que se pide y, así, se pueda abordar el diseño con suficiente previsión. Esencialmente el documento contiene una descripción del sistema como podría hacerla un potencial cliente e incorpora algunos detalles técnicos o sugerencias de implementación. Al final se indican algunas reglas para la elaboración, entrega y evaluación del proyecto.

Algunos requisitos están marcados como "*Opcional 0*". Si no implementa ninguno de ellos, la nota máxima de la práctica estará acotada por un valor MAX. En caso de realizar todos los requisitos opcionales puede aspirar a una nota máxima de 10. El profesor dará indicaciones sobre cual es esa nota MAX en clase.

## 2. La aplicación web

La aplicación web que debe implementar tiene como finalidad la gestión de un libro de recetas de cocina. Cualquier visitante podrá verlas e incluso hacer comentarios sobre las mismas.



# Comida sana para todos los días

[Listado de recetas](#) [Añadir nueva receta](#) [Ver mis recetas](#)

## Risotto de calabaza y champiñones



Arroz, Cena, Queso, Fácil

Autor: el cocinillas

El risotto es una técnica culinaria italiana que tiene su origen en el noroeste del país, concretamente en el Piamonte, donde tradicionalmente había abundancia de arroz. Cuando se cocina el risotto, el arroz cuece poco a poco con el resto de ingredientes del plato, no por separado. Verás como en este risotto de calabaza y champiñones uno de los distintivos es el queso parmesano, fundamental en cualquier variedad de risotto.



1.- Si no tienes caldo de verduras, puedes preparar uno mientras elaborados el resto de la receta. Para ello, pon a hervir unas verduras en abundante agua. Puedes incluir cebolla, puerro, apio y zanahoria. Deja hervir durante media hora y pon un poco de sal.

2.- Mientras se hace el caldo de verduras, hamos un sofrito con la cebolleta picada y un poco de aceite de oliva. Dejamos que se cocine durante 4 minutos.

3.- Agregamos la calabaza pelada y cortada en cuadraditos. Cuanto más pequeña la cortes antes se cocinará. Ponemos un poco de sal y pimienta negra molida y dejamos cocinar hasta que comience a ablandarse un poco, unos 20 minutos.

4.- Es el momento de incorporar los champiñones fileteados y limpios a este risotto de calabaza. Dejamos cocinar durante 2-3 minutos.

5.- Echamos el arroz, rehogamos mezclando bien con el resto de ingredientes y cubrimos con el caldo de verduras. Vamos moviendo el risotto de calabaza y champiñones poco a poco y dejamos que reduzca el agua.

6.- Lo importante del risotto es ir incorporando el caldo poco a poco y dejar que el arroz suelte el almidón y se cocine a fuego lento, pero siempre con líquido, sin que quede seco. El tiempo de cocción es de unos 20-22 minutos, dependiendo del tipo de arroz.

7.- Cuando tengamos el arroz casi listo, ponemos un poco de parmesano rallado y movemos para que se integre su sabor.

8.- Servimos el risotto de calabaza y champiñones con un poco más de parmesano rallado por encima.

1 kilogramo de calabaza  
1 Cebolleta  
120 gramos de champiñón  
60 gramos de Parmesano  
1 chorro de Aceite de oliva  
150 gramos de arroz blanco  
1 pizca de Sal  
1 pizca de Pimienta negra  
700 mililitros de caldo de verduras



10/07/2020. Juanita Pérez

Hmmmm ... ¡ qué buena pinta tiene !

12/07/2020. Anónimo

Sí, mañana lo voy a probar y ya os contaré

1

2

3

4

5



### Login

Usuario

Clave

Login

### + valoradas

- 1 Risotto de calabaza y champiñones
- 2 Pollo al salmorejo
- 3 Ensalada de espinacas y mango

### nº recetas

El sitio contiene 1452 recetas diferentes

En la figura anterior puede ver un ejemplo del aspecto que podría tener la aplicación. Las partes más relevantes son:

- Encabezado con algún logotipo y título.
- Barra de menús.
- Barra lateral con información accesoria.
- Pie de página.
- Zona principal de contenidos (en este caso mostrando una receta<sup>1</sup>).

El sitio web permitirá a cualquier visitante consultar y buscar las recetas de cocina almacenadas. Además, los usuarios podrán identificarse en el sitio de manera que solo si están identificados podrán añadir nuevas recetas.

Debe existir también un tipo de usuario administrador que pueda realizar tareas de gestión del sitio como, por ejemplo, añadir o borrar usuarios del sistema.

### 3. Información que debe gestionar el sistema

Debe gestionar información de diversa índole. La más importante es la que tiene que ver con las recetas de manera que, para cada una, debe almacenar (al menos):

- Autor [Un usuario registrado]. Solo los usuarios registrados pueden incluir recetas.
- Nombre de la receta [Cadena]
- Descripción [Cadena larga]
- Ingredientes [Cadena larga]
- Preparación [Cadena larga]
- Fotografías del proceso: 0 o más [Imágenes en formato JPEG]
- Categoría [Cadena]. Se refiere al tipo de receta (pescado, carne, sopa, ...)
- Comentarios: será una lista de comentarios de texto (cada uno contiene la fecha y hora, el comentario y el usuario que lo ha escrito -o "anónimo" si lo ha hecho un usuario sin identificar-)

Las categorías hacen referencia al tipo de comida: carne, pescado, crema, cena, desayuno, postre, etc y puede optar por varias posibilidades:

- Permitir solo una categoría por receta. En este caso se puede almacenar como una cadena en la misma tabla de la receta. Esto es lo mínimo que se requiere en la aplicación.
- **Permitir más de una categoría por receta.** Esto tiene sentido porque puede haber distintas clasificaciones de recetas (carne/pescado/vegano/...) (desayuno/cena/postre/...) (ligera/pesada/...) (congelable/no congelable) ... y así tantas como se nos puedan ocurrir. La forma que, en principio, puede parecer más simple para poder tener varias categorías es almacenarlas todas en una cadena separadas por comas. Así la base de datos no requiere cambios y la aplicación solo pequeños ajustes.

Tenga en cuenta que cada receta puede incorporar un número indeterminado de fotografías (ninguna, una, dos, cien, ...). Estas pueden almacenarse en la base de datos (campo de tipo BLOB) o en ficheros.

Igualmente el número de comentarios asociados a una receta será indeterminado.

☆ *Opcional 1: Si queremos disponer de múltiples categorías por receta, la solución indicada antes no es la mejor. Podemos pensar en tener una tabla en la que se guardan todas las posibles categorías. Esta tabla estará relacionada con la de recetas. Cuando el usuario añada o edite una receta le aparecerán checkboxes para marcar las categorías a las que pertenece. Esta lógica implica que debe disponer de formularios para editar esa tabla de categorías (añadir, editar, borrar). Esa tabla solo será modificable por usuarios administradores.*

1 Obtenida de [www.recetasgratis.net](http://www.recetasgratis.net)

☆ **Opcional 2:** Puede añadir a la aplicación la posibilidad de que los usuarios puntúen las recetas con un valor entre 1 y 5. En este caso, cada usuario registrado puede valorar solo una vez una misma receta. Por tanto, deberán almacenarse todas las valoraciones individuales aunque finalmente lo que se muestre a los usuarios de la página web será la media de dichas valoraciones. Los visitantes sin identificar no podrán valorar las recetas para evitar que puedan puntuar muchas veces una misma receta falseando los datos.

Otra información importante que debe gestionar es la relativa a los usuarios del sistema. Para cada uno de ellos almacenará (al menos):

- Nombre [Cadena]
- Email [Cadena]
- Fotografía [una imagen JPEG]
- Password [Cadena cifrada]
- Tipo de usuario [colaborador / administrador]

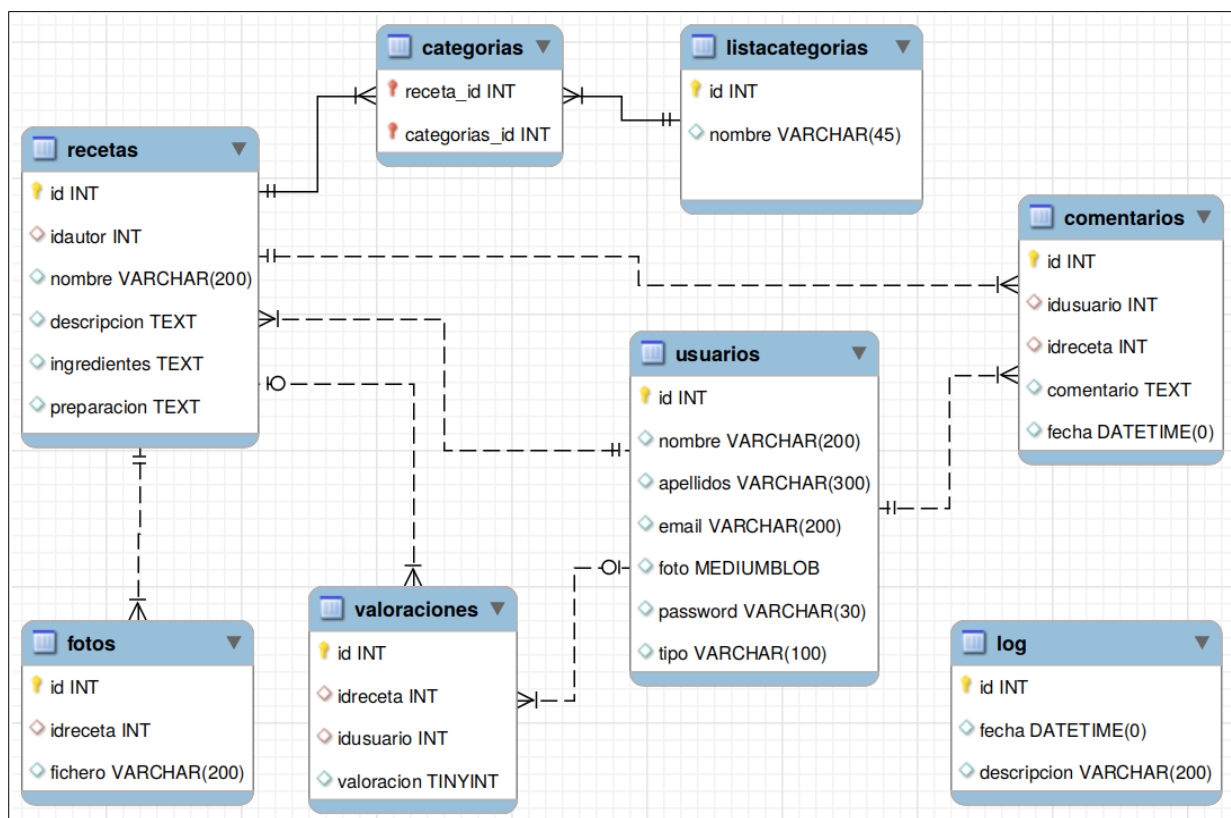
Observe que puede haber varios usuarios de tipo administrador, ya que esta propiedad está almacenada en la BBDD y, por tanto, puede tenerla cualquier usuario del sistema.

También deberá disponer de un cuaderno de bitácora (log) en el que vaya registrando los eventos más relevantes que se vayan produciendo durante la ejecución. Para ello, cada vez que ocurra algo relevante anotará:

- Fecha [Fecha]
- Descripción [Cadena]

### 3.1. Diseño de la base de datos

Debe incluir en la documentación un diseño de la base de datos. A continuación tiene una sugerencia que incluye algunas de las cuestiones consideradas como optativas en este guión pero puede optar por un diseño distinto si lo estima oportuno.



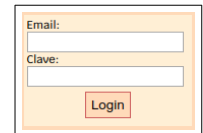


## 4. Descripción del sistema en función del tipo de usuario

Los usuarios de la página pueden ser de tres tipos distintos:

1. **Visitantes.** Son los usuarios que navegan por el sitio sin ningún tipo de autenticación, es decir, cualquier persona que acceda a nuestro sitio sin estar registrado en el mismo.
2. **Colaboradores.** Son usuarios que están registrados en nuestro sitio y que, además de ver la información pública del sitio, **pueden añadir nuevas recetas a la web.**
3. **Administradores.** Son usuarios que, además de hacer lo que hacen los visitantes y los colaboradores, **pueden gestionar a otros usuarios** (editar, borrar, etc), **moderar comentarios** (borrarlos), **editar la lista de categorías** (si existe), etc.

Cuando el visitante de la web no está autenticado en el sistema, el sitio debe mostrar en alguna parte un formulario de login o un enlace para acceder a dicho formulario e identificarse. Una vez identificado, en lugar del formulario o del enlace de login, deberá mostrarse la identificación del usuario y un botón o enlace para cerrar la sesión y dejar de estar identificado en el sitio.



Formulario de login con campos para Email y Clave, y un botón Login.

Una vez identificado un usuario, y dependiendo del tipo del mismo (colaborador o administrador), el sistema mostrará nuevas opciones en el menú de navegación o en los distintos items visualizados en las páginas del sitio. Por ejemplo, si el usuario se identificase como administrador, debería disponer de opciones para hacer listados de los usuarios registrados, para añadir nuevos usuarios, para borrar algún usuario, etc. Al menos, los menús deberían incluir:



Perfil de usuario de Javier Martínez, colaborador, con botones Editar y Logout.

- **Menú para visitantes de la página:**
  - Ver listado. Muestra un **listado de recetas**. Puede tener submenús con categorías. Debe incluir un **formulario de búsqueda y ordenación** que permita listar solo algunas recetas según algún criterio.
- **Menú para usuarios identificados:**
  - Ver **listado**.
  - **Añadir nueva receta**.
  - **Ver mis recetas**. Muestra solo las recetas del usuario identificado.
- **Menú para administradores:**
  - Las mismas que para usuarios identificados.
  - **Gestión de usuarios**. Permite añadir usuarios, editar los existentes y borrarlos.
  - **Ver log**. Para visualizar el cuaderno de bitácora.
  - **Gestión de la BBDD**. Para realizar operaciones de mantenimiento de la BBDD (backup).

### 4.1. Visitantes de la página

Un visitante de nuestro sitio web sin autenticar podrá:

- Ver listados de recetas. Estos listados pueden filtrarse para adecuarse a determinados criterios de búsqueda u ordenación (ver más adelante).
- Ver recetas.
- Añadir comentarios a las recetas.

**Cuando un usuario visita por primera vez la página le mostrará una receta al azar.** A partir de ese momento, cada vez que visite de nuevo el sitio verá la **última receta que vió en la visita previa**. Esto lo debe gestionar mediante el uso de cookies: cada vez que visualice una receta se enviará una **cookie** al cliente para que recuerde esa información en próximas visitas.

Aunque puede optar por un diseño diferente, una posibilidad es que al visualizar las recetas se añadan botones o enlaces que permitan realizar acciones sobre ellas. En el ejemplo de la sección 2 puede ver al final de la receta una botonera. Uno de los botones permite añadir un comentario.

## 4.2. Usuarios colaboradores

Estos usuarios son los que van a nutrir de contenido al sitio web. Serán usuarios registrados en el sistema y tienen la posibilidad de añadir nuevas recetas.

Las tareas que puede llevar a cabo este usuario son las siguientes:

- Las mismas que un visitante. Cuando añaden comentarios, ya no son anónimos.
- **Modificar sus datos personales.**
- Añadir nuevas recetas al sistema.
- Ver sus propias recetas y modificarlas.

A modo de ejemplo, a continuación tiene un formulario para editar o añadir nuevos usuarios al sistema<sup>2</sup>:



Edición de usuario

Foto:   Ningún archivo seleccionado

Nombre:

Apellidos:

Email:

Clave:

Dirección:

Teléfono:

Rol:

Estado:

El formulario incluirá un botón para modificar los datos. Al pulsarlo, el sistema debería pedir confirmación de la acción. Para ello se puede mostrar el mismo formulario (o parecido) pero sin posibilidad de edición. En la figura siguiente se ilustra:



Edición de usuario

Foto:   Ningún archivo seleccionado

Nombre:

Apellidos:

Email:

Dirección:

Teléfono:

Rol:

Estado:

Al pulsar sobre el botón de confirmación debería mostrarse un mensaje informando al usuario del éxito (o no) de la operación:

Se han modificado los datos del usuario

Observe que el proceso ha requerido de 3 páginas:

2 Los campos pueden variar dependiendo del diseño de su BBDD.

1. Edición
2. Confirmación
3. Información

Esta filosofía debe aplicarla siempre que tenga que realizar operaciones que afecten a la información almacenada en la BBDD.

Nota: por motivos de seguridad las claves no se almacenan en texto plano en la BBDD. Antes de hacer la inserción debe cifrarlas.

☆ **Opcional 3:** se puede implementar un módulo que permita que un **usuario visitante se dé de alta en el sistema de forma autónoma** sin intervención del administrador. Para ello, accederán al sitio como visitantes y rellenarán un formulario con los datos indicados anteriormente. A continuación, la aplicación web les enviará un **correo que incluirá un mensaje de bienvenida y un enlace con una URL única que permitirá verificar que el email es válido**. Una vez validado el usuario ya podrá acceder de forma identificada al sistema pasando a ser un usuario de tipo colaborador. En el formulario, la clave debe introducirse dos veces por motivos de seguridad.

Puede contemplar la opción de moderación de este registro de usuarios por parte del administrador. Es decir, en lugar de que el usuario quede registrado, este quedará almacenado pero sin posibilidad de autenticarse hasta que el administrador le dé permiso.

#### 4.2.1. Añadir nuevas recetas

Los usuarios identificados tendrán acceso a un formulario para crear nuevas recetas. Aquí, de nuevo, tiene múltiples posibilidades. Por simplificar el mecanismo para añadir fotografías, se recomienda crear varios formularios. Un primer formulario permite crear una receta sin fotografías:

Datos de la receta

Título

Descripción

Ingredientes

Preparación

☐ Carnes

☐ Pescados

☐ Arroz

☐ Sopa

☐ Fácil

☐ Difícil

☐ Ligero

☐ Pesado

Añadir receta

Observe que algunos datos no se están introduciendo aquí:

- El autor de la receta lo añade automáticamente nuestra aplicación.



- Los comentarios y las valoraciones los añadirán los usuarios a posteriori.
- Las fotografías se podrán añadir más adelante.

Además, según haya optado por la forma de considerar las categorías, en lugar de la lista de checkbox podría tener un único campo de texto.

Al igual que con el caso de la edición de usuarios, tras añadir la receta se debería mostrar la información y pedir confirmación. Finalmente se mostrará información sobre el éxito o no de la operación de inserción en la BBDD.

#### 4.2.2. ¿Cómo se añaden fotografías a una incidencia?

Una receta puede contener cero o más fotografías pero no se sabe a priori cuántas puede haber. El inconveniente es que, al desconocer este dato, no es posible crear un formulario con un determinado número de controles de tipo input file.

Una solución sencilla es la siguiente:

1. primero creamos la receta solo con la información textual.
2. después la editamos y le añadimos fotografías.

Veamos un ejemplo: comenzamos introduciendo los datos de tipo textual de la receta en un formulario:

### Datos de la receta

Título	<input type="text" value="Risotto de calabaza y champiñones"/>
Descripción	<input type="text" value="El risotto es una técnica que bla bla bla"/>
Ingredientes	<input type="text" value="1 kilogramo de calabaza"/> <input type="text" value="1 cebolleta"/> <input type="text" value="..."/>
Preparación	<input type="text" value="1.- Si no tienes caldo de verduras bla bla bla"/> <input type="text" value="2.- Mientras se hace el caldo bla bla bla"/> <input type="text" value="etc"/>
Categorías	<input type="checkbox"/> Carnes <input type="checkbox"/> Pescados <input checked="" type="checkbox"/> Arroz <input type="checkbox"/> Sopa <input checked="" type="checkbox"/> Fácil <input type="checkbox"/> Difícil <input type="checkbox"/> Ligero <input type="checkbox"/> Pesado
<input type="button" value="Añadir receta"/>	

Al pulsar el botón "Añadir receta", el sistema pedirá confirmación antes de hacer la inserción en la base de datos. Puesto que esencialmente se trata de mostrar la misma información del formulario anterior, se puede resolver de forma simple reutilizándolo pero desactivando la entrada de datos a dicho formulario:

### Datos de la receta

Título

Risotto de calabaza y champiñones

Descripción

El risotto es una técnica que bla bla bla

Ingredientes

1 kilogramo de calabaza  
 1 cebolleta  
 ...

Preparación

1.- Si no tienes caldo de verduras bla bla bla  
 2.- Mientras se hace el caldo bla bla bla  
 etc

Categorías

☐ Carnes

☐ Pescados

☒ Arroz

☐ Sopa

☒ Fácil

☐ Difícil

☐ Ligero

☐ Pesado

Confirmar

Si todo es correcto, pulsaremos el botón "Confirmar". En ese momento se hará la modificación de la base de datos y se debería mostrar algún mensaje informativo sobre el resultado del proceso:

### Datos de la receta

Receta añadida con éxito

Hasta aquí ya tenemos introducida la receta básica: solo datos de tipo textual.

Para añadir fotografías necesitamos un formulario que permita añadirlas de una en una, es decir, un formulario que muestre información de la receta sobre la que estamos trabajando y que permita añadir una única fotografía. Cada vez que añadimos una nueva, se mostrará de nuevo el mismo formulario permitiendo que añadamos otra.

Podemos aprovechar que también necesitamos un formulario para editar los datos de la receta y crear una página con esta estructura:

- Un primer formulario para editar los datos textuales de la receta (igual que el que se ha utilizado para crear la receta).
- Un segundo formulario que muestre cada una de las fotografías ya insertadas con un botón al lado de cada una que permita eliminarlas. Además, se incluirá un botón para añadir una nueva fotografía.

Con esta solución, el usuario puede añadir tantas fotografías como desee o eliminar las ya existentes de una forma muy simple. En este proyecto puede optar por alguna solución diferente siempre y cuando permita añadir un número indeterminado de fotografías a una receta.

Veamos el aspecto que podría tener la página citada:

### Datos de la receta

Título

Descripción

Ingredientes

Preparación

Categorías
☐ Carnes
☐ Pescados
☒ Arroz
☐ Sopa
☒ Fácil
  
☐ Difícil
☐ Ligero
☐ Pesado

### Fotografías adjuntas

La primera fotografía se usará en la cabecera de la receta y el resto al final de la misma

Ningún archivo seleccionado

Cada vez que se añade una fotografía, se carga de nuevo la página añadiendo nuevos campos. Veamos un ejemplo en el que se ilustra el resultado de añadir dos fotografías:

### Fotografías adjuntas

La primera fotografía se usará en la cabecera de la receta y el resto al final de la misma



Ningún archivo seleccionado

### Fotografías adjuntas

La primera fotografía se usará en la cabecera de la receta y el resto al final de la misma





Ningún archivo seleccionado

En ambos casos se muestra solo la parte final de la página. La primera parte sería la misma que en la del formulario inicial (datos de texto de la receta).

Al subir fotografías al servidor debería comprobarse que, efectivamente, son archivos que contienen imágenes (y no otra cosa). También se podría comprobar el formato o permitir solo algunos formatos (jpg, gif, png, etc).

Respecto a la forma de almacenamiento de fotografías en el servidor tiene tres opciones:

- Almacenar las fotografías en ficheros y registrar en la base de datos únicamente el nombre de dichos ficheros. En este caso debe tener cuidado con los nombres de los ficheros para evitar caracteres extraños, accesos a otras carpetas del sistema o nombres duplicados. Los ficheros se almacenarían en alguna carpeta accesible desde internet que, en principio, no tendría porqué estar protegida frente a accesos indebidos.
- Almacenar las fotografías en la base de datos (campo de tipo BLOB). En este caso se garantiza la privacidad de las mismas (por si fuese importante). Evitamos así problemas con nombres de ficheros, duplicidades, etc. Cuando se recupera esta información de la BBDD, tenemos varias alternativas para mostrarla en la web.
  - La primera de ellas es incrustar la imagen dentro del tag img de HTML evitando así el acceso a ficheros adicionales. El inconveniente es que el documento HTML es mucho más pesado, se impide la recuperación en paralelo de varias fotografías con HTTP y se impide también el uso de la caché del navegador para fotografías que ya se cargaron con anterioridad.
  - La segunda forma sería que, una vez recuperada la imagen de la BBDD, el servidor la almacenaría en un fichero temporal en disco y en el documento HTML se colocaría la URL de dicho archivo. Una vez descargado el archivo de imagen este no debería seguir siendo accesible desde internet. Esta opción complica un poco la lógica de programación pero aceleraría la carga de la página.
  - Otra forma sería implementar un script PHP en el servidor al que se le solicita una imagen y este devuelve el fichero con dicha imagen. De esta forma evitaríamos crear ficheros de acceso temporal en el servidor.
- Una tercera alternativa sería almacenar las fotografías en ficheros aunque en carpetas no accesibles desde internet. Cuando se accede a una de esas fotografías, el sistema puede copiar el fichero en una carpeta accesible desde internet y enviar su URL. De esta forma mantenemos las ventajas de ambos métodos, aunque se complica también la lógica de la aplicación.

En este proyecto se valorarán todas las opciones por igual y no se tendrán en cuenta los factores de eficiencia o privacidad comentados.

En el caso de las fotografías de los usuarios registrados también debe tomar esta decisión, aunque en ese caso sí puede ser relevante preservar la privacidad impidiendo que usuarios ajenos al sistema vean las fotografías de los usuarios (si estos no lo desean o si el sistema no está diseñado para ello).

*☆ Opcional 4: El inconveniente de la solución propuesta para subir fotografías es que cada vez que se añade una nueva, esta se sube al servidor y este genera un nuevo formulario de edición que permitirá añadir la siguiente. Esto genera un tráfico alto de datos (cada vez que se presenta el formulario se incluyen las fotografías ya almacenadas por si hay que borrarlas) y no es la forma más cómoda para el usuario final. Este es un punto en el que el uso de AJAX y JavaScript mejorarían sustancialmente la usabilidad del sistema. Opcionalmente puede optar por implementar una solución basada en tecnologías de actualización dinámica de la web.*

*☆ Opcional 5: En situaciones como esta en la que en una página se muestra una colección de fotografías, hay que cuidar el consumo de ancho de banda. Cuando un usuario crea una receta, lo normal será que añada fotografías más o menos grandes (es decir, "pesadas" en términos de espacio). Sin embargo, la visualización de la receta las muestra en un formato pequeño. Esto se puede conseguir con el atributo width del tag img. El inconveniente es que estamos transfiriendo varias fotografías pesadas para verlas en pequeño o, en otras palabras, estamos consumiendo innecesariamente ancho de banda. Opcionalmente puede implementar alguna*

*solución para este problema. Se sugieren varias:*

- Cuando se sube una imagen al servidor se almacena la imagen y una versión pequeña que se ajuste al tamaño con la que la vamos a visualizar.
- Cuando se visualiza la receta, se crean bajo demanda las imágenes pequeñas y se envían esas en lugar de las grandes. El inconveniente es que cargamos más el servidor.

☆ *Opcional 6: En cualquier caso, lo importante es que al mostrar la receta se ven las versiones en miniatura y, cuando pulsamos sobre alguna de las imágenes podremos ver la versión grande. Para esto podemos optar por:*

- La imagen pequeña es un enlace a la imagen grande. El inconveniente es que la imagen grande se verá sin ningún tipo de integración con el sitio web.
- Al pulsar la imagen pequeña se genera una página que incluye la imagen grande. Esa página mantiene el esquema del sitio web.
- Al pulsar la imagen pequeña usamos AJAX para recuperar y visualizar la imagen grande.

### 4.3. Usuarios administradores

Estos son los gestores de la aplicación web. Por motivos de seguridad no pueden registrarse de forma autónoma si es que ha decidido implementar esa opción.

Además de las tareas que hacen los colaboradores, los administradores podrán:

- Gestionar los usuarios. Pueden dar de alta nuevos usuarios, editarlos e incluso borrarlos. También pueden asignarles privilegios (colaborador o administrador). Como un administrador puede borrar usuarios, ha de tener cuidado con borrar todos los que tengan privilegios de administración porque dejaría el sistema inusable. Una solución simple es impedir que un usuario pueda borrarse a sí mismo.
- Consultar el log del sistema.
- Tareas administrativas sobre la BBDD (básicamente para crear y restaurar backups).

Para gestionar a los usuarios deberá poder acceder a un listado de los mismos al estilo del siguiente:

Gestión de usuarios		
Indique la acción a realizar		
<a href="#">Listado</a> <a href="#">Añadir nuevo</a>		
	Usuario: Zeus de Grecia Email: <a href="mailto:admin@admin.admin">admin@admin.admin</a> Dirección: Teléfono: Rol: administrador Estado: activo	 
	Usuario: Ana Martín Martín Email: <a href="mailto:ana@noexiste.noexiste">ana@noexiste.noexiste</a> Dirección: Urbanización Limpita, nº 10 Teléfono: Rol: administrador Estado: activo	 
	Usuario: Antonio García Email: <a href="mailto:antonio@noexiste.noexiste">antonio@noexiste.noexiste</a> Dirección: C/ de un sitio Teléfono: 555 121212 Rol: colaborador Estado: activo	 
	Usuario: María López Pérez Email: <a href="mailto:maria@noexiste.noexiste">maria@noexiste.noexiste</a> Dirección: C/ de por aquí, nº 3 Teléfono: 555 112233 Rol: colaborador Estado: activo	 
	Usuario: Javier Martínez Email: <a href="mailto:jbaena@ugr.es">jbaena@ugr.es</a> Dirección: He modificado mi dirección, ahora vivo por acá Teléfono: 555 212121 Rol: colaborador Estado: activo	 
	Usuario: Juan Sinnombre Email: <a href="mailto:juan@noexiste.noexiste">juan@noexiste.noexiste</a> Dirección: Teléfono: Rol: colaborador Estado: activo	 

A la derecha de cada usuario se ven botones para editar y borrar.

Los administradores también podrán:

- Modificar y borrar recetas de otros usuarios.
- Borrar comentarios de usuarios.

Para controlar esto basta con añadir botones cuando proceda. Por ejemplo, si visualiza una receta (y tiene privilegios de administrador) se añadirá un botón para editar y otro para borrar. Algo parecido se puede hacer con los comentarios: añadir botones para borrarlos en caso de que se esté autenticado como administrador.

### 4.3.1. Gestión de la base de datos

En relación con este tema, la aplicación deberá incluir para los usuarios administradores la opción de crear una copia de seguridad de la BBDD. Consistirá en devolver un fichero con la secuencia de cláusulas SQL que permitirían restaurar la BBDD completa en caso de necesidad.

*☆ Opcional 7: con el fichero anterior se podría, en caso de fallo, restaurar la información del sistema. Para ello, la aplicación debe proveer un formulario que permita subir dicho fichero al sistema y ejecutar las cláusulas SQL que contiene, restaurando de esa forma la BBDD completa. Debe tener cuidado con estas operaciones ya que la restauración de la BBDD implica que se debe borrar antes lo que pudiese haber almacenado en la BBDD. Ese borrado y restauración podrían dejar sin usuarios de tipo administrador al sistema.*

*Además, al ser una operación de borrado y creación de tablas así como de inserción masiva de datos, habría que cuidar de deshabilitar las comprobaciones de integridad de la BBDD durante el proceso (por ejemplo, deshabilitar la comprobación de claves externas).*

*También puede haber detalles importantes relacionados con la configuración del SGBD ya que cabe la posibilidad de que tras borrar y crear una tabla, si esta información no queda asentada en la BBDD, las operaciones de inserción o consulta posteriores fallen. Para ello deberá gestionar adecuadamente transacciones que garanticen que el proceso es correcto.*

*☆ Opcional 8: otra opción a incorporar es el borrado completo de la BBDD. Tenga cuidado porque esta operación podría dejar sin usuarios de tipo administrador al sistema impidiendo el acceso a él.*

## 5. Log de la aplicación

Se debe mantener un registro (log) con los eventos principales que se van produciendo en la aplicación:

- Cada vez que se identifica un usuario.
- Cada vez que se registra un nuevo usuario en el sistema.
- Cada vez que un usuario hace alguna acción que modifique la BBDD (inserciones, editados o borrados de datos).
- Cada vez que cierra la sesión un usuario identificado.
- Intentos de identificación erróneos.
- ...

Este registro puede mantenerse en un fichero o en la BBDD y la información que debe almacenarse es:

- Hora y fecha del evento.
- Breve descripción (una cadena de texto).

Cualquier usuario administrador podrá ver este registro. En el listado aparecerán primero las entradas más recientes.



### Log del sistema

2019-01-22 19:19:29	El usuario admin@admin accede al sistema
2019-01-22 19:19:29	El usuario admin@admin sale del sistema
2019-01-22 19:19:29	El usuario pepito@sincorreio.com ha añadido una receta
2019-01-22 19:19:29	El usuario pepito@sincorreio.com ha modificado una receta
2019-01-22 19:19:29	El usuario admin@sincorreio.com ha borrado un comentario

## 6. Listados de recetas

Al mostrar listado de recetas deben poder aplicarse algunos criterios básicos de ordenación y búsqueda:

- Se podrán filtrar recetas que incluyan algún texto en el título.
- Se podrán filtrar recetas que incluyan algún texto en cualquier campo (título, descripción, preparación, ingredientes).
- Se podrán filtrar recetas por categorías.

Además, el orden de las recetas también deberá poder elegirse de entre estas opciones:

- Orden alfabético.
- De más a menos comentarios.
- De más a menos puntuación.

Se ilustra con un ejemplo:

### Datos de la receta

Buscar en título

Buscar en receta

Categorías

☐

Carnes

☐

Pescados

☐

Arroz

☐

Sopa

☐

Fácil

☐

Difícil

☐

Ligero

☐

Pesado

### Ordenar por

☐

Alfabético

☐

De más a menos comentadas

☐

De más a menos puntuación

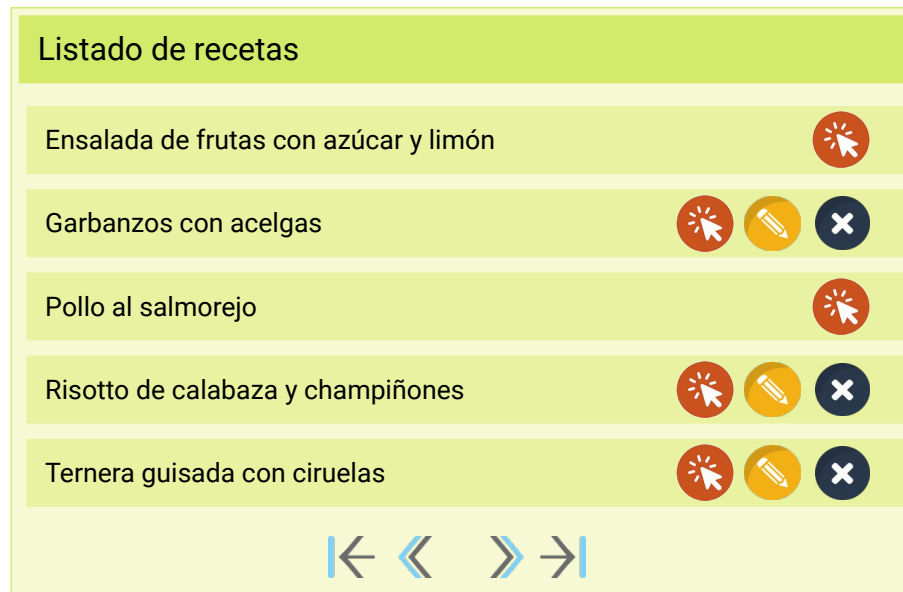
Recetas por página

5 recetas ▼

Aplicar criterios de ordenación y búsqueda

En caso de que se indiquen varios criterios la búsqueda deberá cumplirlos todos.

Una vez aplicados los criterios, veremos un listado de recetas:



Desde el listado será posible realizar acciones sobre cada una de las recetas: verlas, editarlas, borrarlas, ... aunque las acciones posibles dependerán del tipo de usuario que está usando la web:

- Un visitante únicamente podrá verlas (el resto de enlaces o botones no se mostrarían).
- Un usuario identificado, además, podrá editarlas o borrarlas solo si es el autor.
- Un administrador puede editar y borrar siempre.

Por simplicidad, el formulario de ordenación y búsqueda y el listado de recetas pueden mostrarse siempre de forma conjunta, es decir, en una única página.

☆ *Opcional 9: En aplicaciones de este tipo es muy frecuente que los listados sean demasiado largos como para mostrarlos en una única página. Por ello se propone que se incluya la posibilidad de limitar el número de resultados por página (se puede ver en el formulario de ejemplo) y añadir un control de paginación para ir avanzando o retrocediendo en las páginas.*



## 7. Otras utilidades

En las aplicaciones web es frecuente que en algunos lugares de la página se muestre publicidad u otra información relevante al margen del contenido principal. Se propone que en una barra lateral se muestren algunas estadísticas relevantes globales del sistema como, por ejemplo:

- Ranking de las 3 recetas más valoradas.
- Ranking de las 3 recetas más comentadas.
- Ranking de los 3 usuarios que más opinan.
- Número total de recetas.
- ...

Estos "widgets" se pueden añadir en una barra lateral y se muestran en todas las páginas del sitio.

Deberá incluir dos elementos de este tipo en la aplicación web.

**+ valoradas**

- 1 Risotto de calabaza y champiñones
- 2 Pollo al salmorejo
- 3 Ensalada de espinacas y mango

**nº recetas**

El sitio contiene 1452 recetas diferentes

## 8. Instalación de la aplicación

Uno de los problemas a resolver al crear una aplicación web consiste en facilitar el proceso de instalación o despliegue en un servidor nuevo. Para que la aplicación funcione correctamente deben existir las tablas en la BBDD y, posiblemente, deban rellenarse algunos datos iniciales. En nuestro caso, deberíamos incluir en la tabla de usuarios uno con privilegios de administración para poder empezar a trabajar.

Este proceso puede ser manual, es decir, al instalar la aplicación en el servidor, se ejecutan scripts de creación de las tablas (o se crean manualmente) y se insertan datos en la BBDD también de forma manual. Una vez hecho ya podemos acceder a la aplicación web desde un navegador y comenzar a funcionar. Esta metodología supone más trabajo de instalación y es más propenso a cometer errores. Además, presupone conocimientos de BBDD a quien instala la aplicación.

Una forma sencilla de evitar este proceso es hacer que la aplicación detecte si es la primera vez que se ejecuta (la primera vez que alguien accede a ella). En caso afirmativo ejecuta código (PHP) para crear las tablas de la BBDD. Para crear un primer usuario administrador puede optar por: hacer una inserción de una tupla en la tabla de usuarios con un usuario predeterminado o bien mostrar un formulario para pedir los datos de ese primer administrador al usuario que está ejecutando la aplicación.

¿Cómo se puede detectar que es la primera ejecución? También hay varias formas:

- Comprobar si existen las tablas en la BBDD.
- Comprobar si existe o no un fichero concreto. Si existe entonces se trata de la primera ejecución: lo borramos y procedemos al proceso de creación de tablas y primer administrador.
- ...

☆ *Opcional 10: Implemente algún mecanismo que permita la autoinstalación de la aplicación.*

## 9. Cuestiones genéricas sobre diseño e implementación

### 9.1. Diseño del sitio

El diseño de sitios web es un tema complejo que no se aborda en la asignatura. En esta práctica, tanto el diseño gráfico como el maquetado del sitio son libres. Se pueden entender las capturas de pantalla mostradas solo como sugerencias y se han incluido con el único propósito de ilustrar la funcionalidad requerida. La valoración tendrá en cuenta, sobre todo, la parte técnica y funcional del sitio (uso de las tecnologías estudiadas, usabilidad, etc).

La parte estética o artística pasa a un segundo plano salvo que sea tan mala que impida un uso razonable de la web o demuestre un bajo conocimiento de las tecnologías empleadas, en cuyo caso será valorada de forma negativa.

Deberá considerar si es más adecuado usar un diseño de ancho fijo o fluido. En cualquier caso, la página deberá visualizarse en pantallas de ancho mayor o igual a 800px y en diferentes navegadores.

La práctica podría corregirse sobre cualquier navegador por lo que se recomienda que el alumno la pruebe en varios de ellos para asegurarse de que todo es correcto. Al menos debería probarla con Firefox y Chrome, que serán los utilizados en la corrección de la misma.

El uso de un diseño adaptable se valorará positivamente. No es necesario que sea muy sofisticado pero, al menos, debe permitir una visualización correcta en dispositivos con ancho inferior a 800px.

### 9.2. Formularios del sitio

Todos los formularios del sitio deben cumplir las siguientes normas:

- Deben ser de tipo sticky cuando tenga sentido. Por ejemplo en todas las operaciones de

edición.

- La validación de datos del formulario debe hacerse en el servidor (PHP) en cualquier caso.
- Cada dato deberá ser validado de acuerdo a su tipo. Por ejemplo, la validación de un email será distinta que la de un apellido.
- Deberá incluir validación en el cliente (JavaScript). Desde un punto de vista técnico, esta es secundaria y se hace solo para mejorar la usabilidad del sitio.
- En la medida de lo posible, los datos de un proceso deben recogerse en un único formulario. No se admitirán soluciones que obliguen a un usuario a pasar por varios formularios de forma innecesaria (para realizar un proceso que podría haberse hecho con un único formulario).
  - Ejemplo 1: Para crear una receta se piden todos los datos en un único formulario en lugar de pedir el título, en un segundo formulario la descripción, en un tercer formulario los ingredientes, etc. (Se exceptúan las fotos por los motivos ya explicados).
  - Ejemplo 2: Para editar la información personal de un usuario debe hacerse en un único formulario que incluya todos los datos modificables. No se admitirá una solución que obligue al usuario a pasar por varios formularios para hacer alguna modificación.

El caso de la inserción de fotografías sería una excepción a esta norma.

- No deberá mostrar o solicitar información que sea interna del sistema (por ejemplo códigos, claves primarias de la BBDD, hash de passwords, etc).
- La modificación o inserción de datos, como norma general, serán procesos que pidan confirmación, es decir, que necesitarán de:
  - Una primera página que muestra el formulario para editar o añadir información nueva.
  - Una segunda página que pedirá confirmación de inserción o edición. En este paso la información no debe poder modificarse, simplemente se muestra para que el usuario sepa lo que va a modificar o insertar.
  - Una tercera página que informe del éxito o no de la operación
- El borrado de datos, como norma general, también pedirá confirmación por lo que se necesita:
  - Un primer formulario que presenta la información a borrar y pide confirmación. En este caso, el formulario podría tener como único input el botón de submit, el resto de datos podrían ser inputs no editables o incluso tags HTML que no sean controles del formulario.
  - Una segunda página que informa del resultado de la operación.

En los procesos de formulario que necesitan varios pasos debe tener en cuenta que:

- La información que se rellena en el primer formulario se envía normalmente con el método POST.
- Esa información llega al segundo formulario (o página) para mostrarla pero sin posibilidad de edición. En este caso lo habitual será poner los controles de tipo input con el atributo disabled o readonly activado (se pueden ver pero no modificar). Debe tener en cuenta que al enviar el formulario es posible que estos controles no sean enviados por lo que deberá estudiar la forma de que lleguen, si fuese necesario, al tercer formulario que es el que realiza la operación de inserción o modificación en la BBDD. En estos casos es frecuente duplicar la información en inputs de tipo hidden para pasarlos de un formulario a otro mediante POST (o GET).
- En particular, no es posible rellenar el atributo value de un input de tipo file desde PHP por lo que no es posible reenviar ficheros de un formulario a otro. Una solución es que tras enviar el fichero desde el primer formulario, se utilicen variables de sesión para recuperar la información en los siguientes formularios. Esta solución es válida también para cualquier tipo de input y minimiza el trasiego de datos entre formularios: una vez enviados los

datos desde el primer formulario estos se almacenan en el servidor en variables de sesión y en los siguientes formularios sólo se necesita pasar de uno a otro una pequeña porción de información.

### 9.3. Diseño y organización del código

La organización interna del sitio es libre. Se puede optar por distintos enfoques y/o paradigmas de programación:

- Paradigma procedural u orientado a objetos.
- Tener un único fichero (index.php) para gestionar el sitio completo.
  - Usando parámetros GET para ver los distintos contenidos.
  - Usando algún tipo de enrutador para analizar la URL en lugar de usar parámetros GET.
- Un fichero PHP diferente para cada página del sitio.
- Un diseño de tipo MVC.
- ...

En cualquier caso **debe evitar la técnica del "copia/pega"**, que será evaluada de forma negativa (e incluso podría implicar obtener una nota insuficiente para superar el proyecto). Dependiendo del diseño o enfoque utilizado, se admite un copia/pega básico con el esqueleto de una página PHP siempre y cuando los elementos que incluya sean mínimos y correctamente modularizados. Por ejemplo para incluir código HTML o PHP o para hacer algunas llamadas a funciones PHP que maqueten la página o comprueben la autenticación de usuarios. Si tiene dudas consulte con el profesor.

En cuanto a la organización de los ficheros también se deja libertad pero procure que su organización en carpetas sea razonable. Procure también proteger aquellos ficheros que contengan información sensible (datos de acceso a MySQL, log, copias de seguridad, etc) en lugares no accesibles desde internet.

### 9.4. Uso de JavaScript

Debe utilizar JavaScript para realizar algunas mejoras del sistema que, aunque no afectan a la funcionalidad del mismo, sí mejorarían la experiencia de usuario. Algunas sugerencias:

- Se pueden ocultar o mostrar formularios o determinados items para facilitar la legibilidad de la página. Por ejemplo:
  - Al mostrar una receta, es posible que no nos interese toda la información. Podríamos ver inicialmente únicamente el esqueleto de la receta con los distintos apartados (descripción, ingredientes, elaboración, fotografías, comentarios) y, al pulsar sobre alguno de ellos mostrar su contenido.
  - ...

Observe que la funcionalidad de la aplicación no varía en caso de que, por algún motivo, el navegador no pueda ejecutar el código JavaScript, simplemente vería toda la información.

- Se puede usar JavaScript para hacer la validación de datos en el cliente. Esto no nos exime de hacer la validación en el servidor, que es obligatoria en cualquier caso.

Si el navegador no puede ejecutar el código JavaScript, simplemente afectaría a la usabilidad pero no sería un problema para la aplicación.

- ...

Se recomienda dejar esta tecnología para el final del proyecto. Una vez finalizado se pueden añadir estas pequeñas mejoras con poco esfuerzo y, de paso, evitarán que hagamos nuestra aplicación dependiente de JavaScript.

### 9.5. Uso de AJAX

☆ *Opcional 11: El uso de tecnología AJAX no se considera obligatorio en este proyecto aunque su*

*uso en alguna parte se valorará de forma positiva.*

Algunos lugares en los que sería apropiado su uso:

- Paginación de listados de resultados. Al pulsar sobre los botones del paginador la página se actualizaría de forma dinámica.
- Ampliación de información de items. Por ejemplo, al hacer un listado de recetas podría mostrarse únicamente el título de las mismas y un botón para ampliar información. Al pulsar el botón se contactaría con el servidor para obtener los datos de la receta y se mostrarían en la misma página.
- Se puede mejorar la usabilidad en el formulario de edición/creación de recetas permitiendo que, cada vez que el usuario selecciona un fichero con una fotografía, esta se muestre de forma dinámica (sin necesidad de pulsar el botón de enviar) y generando nuevos controles en el formulario que permitan borrarla o añadir nuevas fotografías.
- En el formulario de edición/creación de usuarios se puede usar también para visualizar la fotografía antes de enviar el formulario.

## 10. Aspectos principales de la evaluación

Se evaluará la práctica en todas sus vertientes aunque, como ya se ha dicho, el diseño visual pasará a un segundo plano siempre y cuando sea funcional y usable. Se enumeran a continuación algunos detalles a tener en cuenta de cara a la evaluación de la práctica:

- Se deberá prestar especial atención al cumplimiento de todas las normas explicadas en este documento. El no cumplimiento de las mismas podrá implicar una calificación de "cero".
- La aplicación debe funcionar correctamente en el servidor void.ugr.es. En caso de que falle la práctica, podrá tener una calificación de "cero".
- La base de datos debe contener datos de prueba que permitan comprobar el correcto funcionamiento de todas las páginas del sitio. Así mismo, debe suministrarse el login y clave de todos los usuarios registrados en el sistema así como el rol de cada uno de ellos. El no cumplimiento de este punto puede implicar una calificación de "cero".
- Se valorará el cumplimiento de los requisitos especificados en este documento.
- Se deberán utilizar elementos de HTML y CSS estándares y con suficiente implantación en los navegadores actuales. La práctica se podrá probar indistintamente (al menos) en Firefox y Chrome.

### Diseño de la web:

- La interfaz de la aplicación debe ser homogénea en todas las páginas. Se valorará muy negativamente el cambio de diseño en diferentes páginas del sitio.
- La interfaz debe ser funcional: tamaño proporcionado del texto y de otros elementos, colores "razonables", elementos bien posicionados, ...
- Se valorará que el estilo sea fluido y adaptable frente a estilos de ancho fijo. Aunque si la página tiene un elevado componente gráfico/artístico sí se valorará igualmente el maquetado fijo.
- Se valorará el uso de diseño adaptable para móviles o tabletas.
- Los formularios serán de tipo "sticky" y deben facilitar la inserción y edición de datos.
- Tras procesar datos provenientes de un formulario se debe informar al usuario sobre el éxito o no de la operación realizada.
- Aunque como ya se ha dicho la valoración principal de la página se hará en base a criterios técnicos, también se valorarán positivamente los diseños que tengan un aspecto más profesional.

### Base de datos:

Aunque no se pide mucho detalle en el diseño de la base de datos, sí se requiere que en la documentación del proyecto se incluya, al menos, y de forma breve:

- Un modelo E-R.
- El modelo relacional que se obtiene a partir del modelo E-R.



- Breve descripción de las tablas obtenidas.

El modelo relacional explicado en la documentación debe corresponderse con el implementado en la aplicación. El no cumplimiento de este punto puede implicar una calificación de "cero".

En la entrega de la práctica debe incluir un fichero de restauración de la BBDD que contenga todas las cláusulas que permiten borrar (DROP) y crear (CREATE) las tablas de la aplicación así como las inserciones (INSERT) de los datos de prueba. Debe indicar, en la documentación, el nombre del fichero (incluyendo la ruta si no está en el directorio raíz).

### Implementación:

- Se comprobarán algunas cuestiones de seguridad sencillas como, por ejemplo:
  - Saneado de datos en formularios.
  - Consultas seguras a bases de datos.
  - Validación de formularios adecuada en el servidor.
  - Acceso de usuarios no identificados a las páginas destinadas a usuarios identificados.
- Se valorará por igual el paradigma de programación usado con PHP: programación procedural u orientada a objetos.
- Se valorará la calidad del código desarrollado:
  - Muchas de las páginas del sitio hacen tareas comunes o muy similares: no debe usar la técnica del "copia/pega" sino modularizar adecuadamente su código. Por ejemplo: se deberían reutilizar los formularios para añadir nuevos registros y para editarlos o confirmar su actualización; bastaría con crear un formulario tipo y activar o desactivar campos o botones según la tarea.
  - Calidad técnica.
  - Estilo.
  - Documentación interna / comentarios.
  - Organización del código en ficheros.
- Se valorarán todos los items opcionales.

En la asignatura no se instruye sobre el uso de ningún tipo de framework. Previa consulta al profesor, se permite el uso de algún framework de CSS siempre y cuando quede acreditado que el alumno tiene los conocimientos impartidos en la asignatura además de los conocimientos adecuados sobre el framework en cuestión.

**En ningún caso se permite usar frameworks de PHP ni de JavaScript.**

## 11. Entrega de la práctica

Al comienzo del curso al alumno se le ha facilitado un usuario y clave para acceder al servidor void.ugr.es. En dicho servidor hay una carpeta public\_html en la que el alumno puede subir páginas web, siendo estas accesibles a través de cualquier navegador. Dichas páginas se encuentran protegidas también por el usuario y clave del alumno. El profesor puede acceder en todo momento a toda la información subida al servidor.

Dentro de la carpeta public\_html, deberá crear una subcarpeta llamada **proyecto** en la que subirá la práctica desarrollada de manera que sea funcional. Debe existir en dicha carpeta un fichero index.html o index.php tal que, al cargarlo, se visualice la aplicación desarrollada.

Por ejemplo, si el alumno tuviese como nombre de usuario "joseperez1920" la práctica debería visualizarse al cargar la siguiente URL:

`https://void.ugr.es/~joseperez1920/proyecto`

Cualquier documentación adicional que necesite esta práctica deberá estar contenida en un fichero llamado `documentacion.pdf` alojado también en la carpeta del proyecto.

- No cumplir con alguno de los requisitos de entrega invalidará la entrega completa.
- No se admitirán entregas pasado el plazo establecido para ello bajo.
- No se admitirán entregas por ningún otro medio.

Se considera obligatorio que la aplicación web entregada incluya información que tenga sentido e ilustre correctamente su funcionamiento. Quizá lo más apropiado sea usar algún ejemplo real para evitar invenciones fuera de lugar. En caso de que se opte por usar datos ficticios no se permitirá

usar textos sin sentido o muy genéricos (es decir, no se deben usar nombres o textos como "aheic4mw2xsko" o "Texto de prueba"). Asimismo, se habrán incluido datos de prueba para todas las partes de la aplicación (usuarios de varios tipos, textos, fotografías, etc.).

### 11.1. Documentación de la aplicación

Debe incluir en la entrega un único fichero en formato pdf que incluya, al menos, los siguientes elementos:

- Identificación de el/los alumno/s que ha/n realizado la práctica.
- Nombre de usuario y clave de los usuarios registrados en la aplicación web para poder probarla (indicando para cada uno el usuario, la clave y el rol).
- Nombre del fichero de restauración de la BBDD con datos de prueba.
- Diseño previo de la aplicación (mockups, wireframe, etc)
- Diseño de la BBDD (modelo E-R y modelo relacional).
- Explicaciones técnicas que considere relevantes para la evaluación de la práctica o que quiera poner en valor por algún motivo.
- Listado de los items opcionales que haya incluido en su proyecto.

Además, el código desarrollado (HTML, CSS, PHP, JavaScript) debe estar convenientemente comentado.

**En el pie de página del sitio web incluirá obligatoriamente un enlace al fichero pdf con la documentación del proyecto.**

### 11.2. Prácticas en equipo

Esta práctica se puede desarrollar de forma individual o en pareja. La autoría debe quedar reflejada la documentación entregada así como en los ficheros fuente del proyecto. En caso de que se vaya a realizar en parejas se le debe comunicar al profesor durante las dos semanas siguientes a la propuesta de esta práctica (a la publicación de este documento). **No comunicar este dato en tiempo y forma supondrá que la práctica deberá entregarse de forma individual.**

En caso de que se haga en pareja, podrá solicitar al profesor un usuario/clave para alojarla que sea compartido por los integrantes del equipo de forma que se mantenga la privacidad en el resto de prácticas.

**En el pie de página del sitio web incluirá obligatoriamente el nombre de el/los autor/es del proyecto.**

### 11.3. Publicidad de la práctica

Se recomienda no publicar la práctica (ni esta ni otras) en repositorios públicos para velar por su privacidad **hasta la finalización del curso académico**. Según la normativa de evaluación y de calificación de los estudiantes de la Universidad de Granada:

*Artículo 13. Desarrollo de las pruebas de evaluación*

*7. Los estudiantes están obligados a actuar en las pruebas de evaluación de acuerdo con los principios de mérito individual y autenticidad del ejercicio. Cualquier actuación contraria en este sentido, aunque sea detectada en el proceso de evaluación de la prueba, que quede acreditada por parte del profesorado, dará lugar a la calificación numérica de cero, la cual no tendrá carácter de sanción, con independencia de las responsabilidades disciplinarias a que haya lugar.*

*Artículo 15. Originalidad de los trabajos y pruebas.*

*2. El plagio, entendido como la presentación de un trabajo u obra hecho por otra persona como propio o la copia de textos sin citar su procedencia y dándolos como de elaboración propia, conllevará automáticamente la calificación numérica de cero en la asignatura en la que se hubiera detectado, independientemente del resto de las calificaciones que el estudiante hubiera obtenido. Esta consecuencia debe entenderse sin perjuicio de las responsabilidades disciplinarias en las que pudieran incurrir los estudiantes que plagien.*