



MEMORIA AGO

(APLICACIÓN GESTORA DE OBRAS)

Aarón Medina
Desarrollo de Aplicaciones Web
01/06/2018

ÍNDICE

- [Introducción y finalidad](#)
- [Herramientas y servicios](#)
- [Posicionamiento SEO](#)
- Páginas
 - [Descripción de las páginas](#)
 - [Relaciones entre páginas](#)
- Base de Datos
 - [Diagrama de entidad relación](#)
 - [Descripción de las tablas](#)
- [Seguridad](#)
- [Código relevante](#)
- [Estructura de ficheros](#)
- [Despliegue de la aplicación en hosting](#)
- [Escalabilidad](#)
- [Acceso](#)
- [Registro de cambios de versión](#)
- [Conclusión](#)

INTRODUCCIÓN Y FINALIDAD

La aplicación a desarrollar se basa en un sistema que facilite la gestión de una pequeña empresa de construcción.

Se crea a medida para cada cliente, incorporando y eliminando distintas funciones, pero siempre trabajando sobre la misma base.

Su principal objetivo es la creación de facturas más fácil y rápidamente.

Se implementan otras funciones como la gestión de los clientes, proveedores, materiales, etc.

El desarrollo se ha basado en una empresa ya creada, aportando así los datos necesarios para completar el programa.

HERRAMIENTAS Y SERVICIOS

Para la creación del programa se han utilizado diversas herramientas, tanto software como hardware, y diversos servicios de hosting, APIs, etc.

-HARDWARE:

La aplicación se ha creado mayoritariamente en un portátil Asus con las siguientes características.

Modelo --> Asus F540L

Microprocesador --> Intel I3 5005U

Memoria RAM --> 4GB

-SOFTWARE:

Programa de diseño --> Sublime Text 3

Programa de servidor local --> Xampp

Programa de diseño --> Photoshop y adobe flash

Navegador web --> Google Chrome y Mozilla Firefox

-SERVICIOS Y APIs:

API localización --> Google maps

Servicio de mensajería --> PHP Mailer

Servicio de host --> <https://www.easynome.es/es>

Servicio gestor de base de datos --> phpMyAdmin

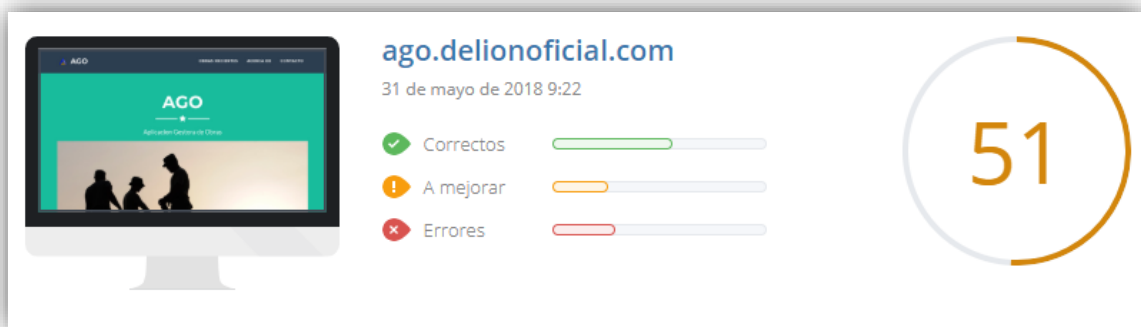
POSICIONAMIENTO SEO

El posicionamiento en los buscadores es un punto importante en las páginas que queremos dar visibilidad.

Por este motivo solamente me he centrado en aumentar el posicionamiento de la Front-page, ya que la parte de Back-End es privada y cuanto más oculta se encuentre más seguridad tendremos.

Para la realización del posicionamiento he utilizado varias herramientas.

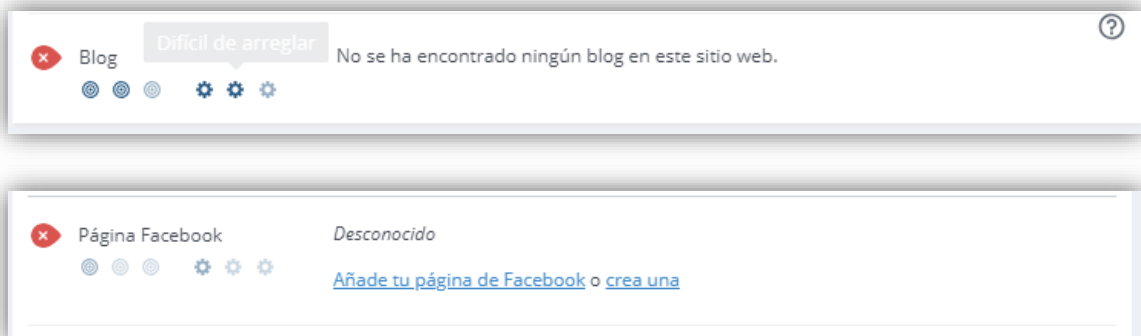
-WOORANK:



Se trata de una página web que comprueba las posibles mejoras en el posicionamiento.

Esta página en concreto tiene una puntuación de 0 a 100, y por otras experiencias aumentar el rango de 60 es extremadamente complicado, por lo que más de la mitad del rango es un posicionamiento muy favorable.

Algún ejemplo de las mejoras que me propone es la incorporación de un blog o enlaces a redes sociales.



Como puntos fuertes de la aplicación tenemos una buena descripción de la página, dentro de los límites establecidos, y encabezados creados.

También nos aporta una visualización de las búsquedas en google.

The screenshot displays a web analytics dashboard with three main sections: Meta Descripción, Vista previa de Google, and Encabezados.

Meta Descripción: Shows a green checkmark, a status of 'Página principal del proyecto AGO para el módulo de Desarrollo de Aplicaciones Web.', and a length of 'Longitud: 83 caracteres'.

Vista previa de Google: Displays a preview of the page as it would appear in Google search results. It includes the title 'Ago Index', the URL 'ago.delionoficial.com/', and the description 'Página principal del proyecto AGO para el módulo de Desarrollo de Aplicaciones Web.'

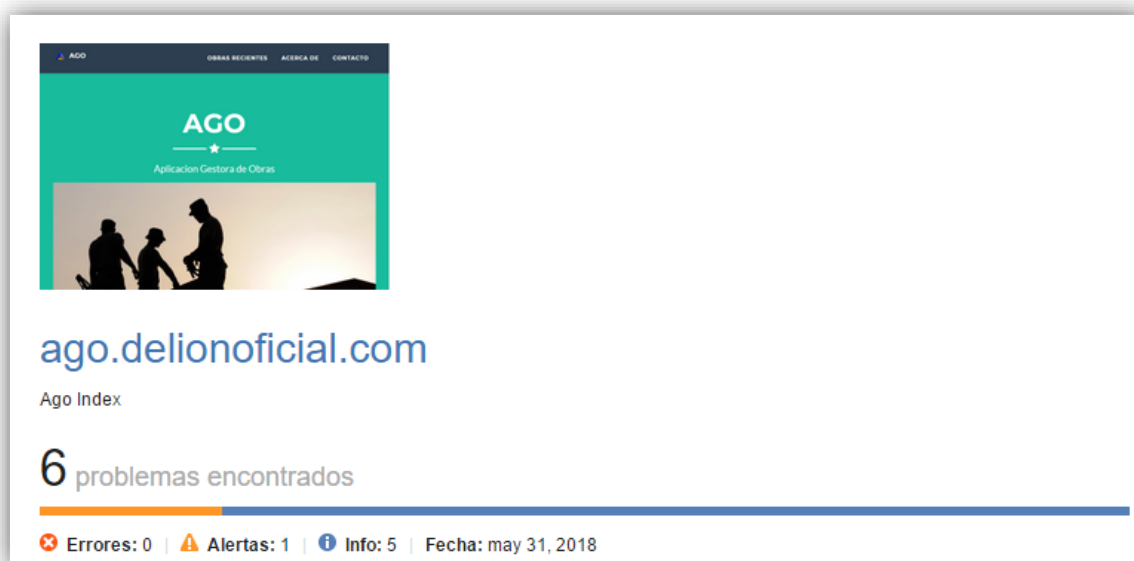
Encabezados: Shows a green checkmark and a table of heading counts. Below the table, specific heading examples are listed.

	<H1>	<H2>	<H3>	<H4>	<H5>
	1	7	0	2	0

<H1>	AGO
<H2>	Aplicacion Gestora de Obras
<H2>	Obras recientes

[Mostrar más](#)

-SEO POWERSUITE:



Es una herramienta que nos analiza más en profundidad nuestro posicionamiento web.

En este caso los errores no son graves, ya que solo tenemos una alerta la cual no afecta para el funcionamiento de la aplicación.

Aquí podemos ver un poco más la información del aviso:



Algunos ejemplos más importantes para el posicionamiento son los siguientes:

✓ **Meta descripción vacía (0 páginas)**

¡Enhorabuena! No hay meta descripciones vacías en tu sitio web.

Acerca de este factor SEO:

A pesar de que las meta descripciones no tienen efecto directo en el posicionamiento, son muy importantes, ya que forman parte del fragmento que la gente ve en los resultados de búsqueda. Por lo tanto, las descripciones deben "vender" la página web para los buscadores y animarles a hacer clic en el enlace.

Si la meta descripción está vacía, será el propio motor de búsqueda el que decidirá qué incluir en el fragmento.

Las descripciones de cada página nos permiten visualizar información en los resultados de búsqueda en los navegadores, por lo que es importante que las páginas contengan esta etiqueta, así como que cumpla los requisitos.

✓ **Optimizada móviles (Si)**

¡Bien hecho! La página principal de tu sitio web es amigable con los móviles.

Acerca de este factor SEO:

De acuerdo con Google, el algoritmo móvil afecta a las búsquedas desde móviles en todos los idiomas del mundo y tiene un impacto significativo en los resultados de búsqueda de Google. Este algoritmo funciona sobre una base de página por página - no se trata de cómo de amigables son tus páginas para móviles, sino simplemente de saber si nos amigables a móviles o no.

El algoritmo se basa en criterios tales como tamaños pequeños de letra, objetivos de pulsación y enlaces, legibilidad de contenido, punto de vista, etc.

Las páginas que son responsive tienen más posibilidades de ser visitadas, ya que podemos acceder a ellas desde cualquier dispositivo sin que sea difícil navegar por la página. En una página responsive los elementos se redimensionan y permiten su correcta visualización.

✓ Archivo robots.txt (Si)

¡Bien hecho! Un archivo robots.txt está disponible en tu sitio web.

Acerca de este factor SEO:

El archivo robots.txt se rastrea automáticamente por robots cuando llegan a tu sitio web. Este archivo debe contener comandos para robots, tales como las páginas que deben o no deben ser indexadas. Si deseas no permitir la indexación de algunos contenidos (por ejemplo, páginas con contenido privado o duplicado), sólo tiene que utilizar una regla apropiada en el archivo robots.txt. Para más información sobre estas normas, echa un vistazo a <http://www.robotstxt.org/robotstxt.html>.

Por favor ten en cuenta que los comandos ubicados en el archivo robots.txt son más como sugerencias en lugar de reglas absolutas para robots. No hay ninguna garantía de que un robot no comprobará el contenido que tengas anulado.

Este fichero nos permite indexar una página, es decir, que los buscadores, como google o bing, puedan encontrar nuestra página y registrarla, haciendo que nuestra página aparezca en los resultados de búsqueda.

También nos permite decidir que páginas queremos que muestre o no, haciendo que la parte privada permanezca oculta.

✓ Sitemap .xml (Si)

¡Bien hecho! Tienes un sitemap .xml presente en tu sitio web. Recuerda reenviarlo a los motores de búsqueda cada vez que realices cambios en él.

Acerca de este factor SEO:

Un sitemap XML debe contener todas las páginas del sitio web que deseas sean indexadas, y debe estar situado en el sitio web, en una estructura de directorios a poca distancia de la página de inicio (por ejemplo <http://www.sitio.com/sitemap.xml>). En general, sirve para ayudar a la indexación. Debes actualizarlo cada vez que agregues nuevas páginas a tu sitio web. Además, el mapa del sitio debe seguir una sintaxis particular.

El mapa del sitio te permite establecer la prioridad de cada página, diciendo a los motores de búsqueda las páginas que se supone deben rastrearse con más frecuencia (es decir, que se actualizan con más frecuencia). Aprende cómo crear un mapa del sitio .xml en <http://www.sitemaps.org/>.

Este fichero genera un mapa de nuestro sitio que complementa al fichero anterior.

En este fichero podemos elegir la periodicidad con la que los buscadores rastrean nuestra web, haciendo que nuestras páginas se indexen con más rapidez.

PANTALLAS ->DESCRIPCIÓN

La aplicación se compone de 4 pantallas principales, sobre las que cargarán las distintas partes de la aplicación.

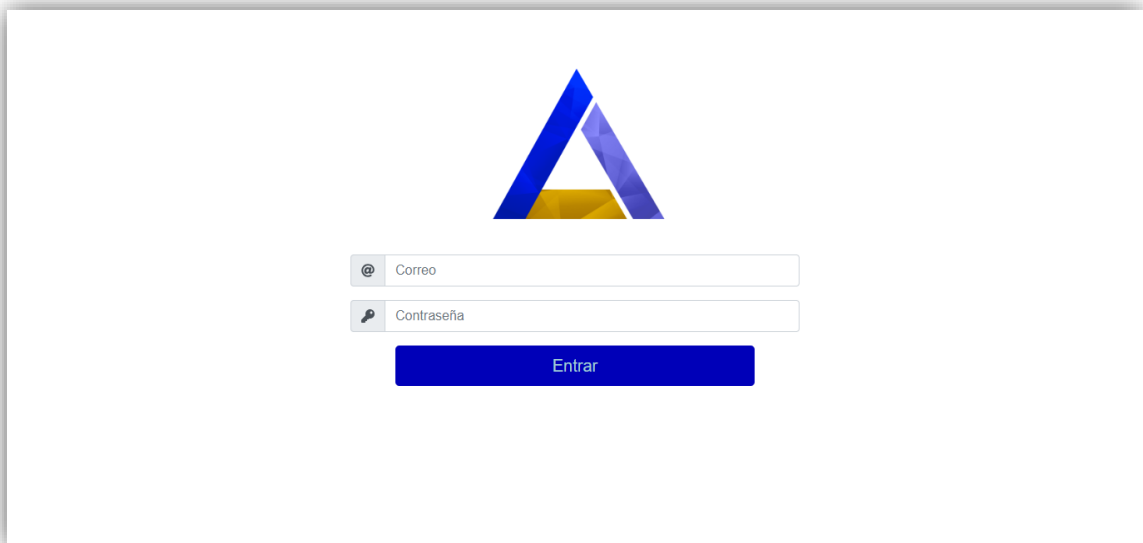
Las pantallas principales son:

-INDEX.HTML:

Esta página es la puerta a la aplicación. Es la pantalla de login. Aquí se realiza la autenticación del usuario y se guardan sus datos en la sesión.

La página mostrará mensajes de error si el correo no existe en la base de datos, la contraseña es errónea, algún campo de acceso está vacío o si el correo introducido no tiene formato de correo.

Si la información enviada es correcta, la página redirige a index.php.



-CREAR/RECUPERAR CONTRASEÑA:

En esta página nos permite tanto crear la contraseña de una cuenta nueva como cambiar la contraseña de una cuenta antigua.

A esta pantalla solo se podrá acceder mediante los correos que envía la aplicación, ya que los correos generan una cadena JSON con el hash del usuario, su correo y el tipo de acceso.

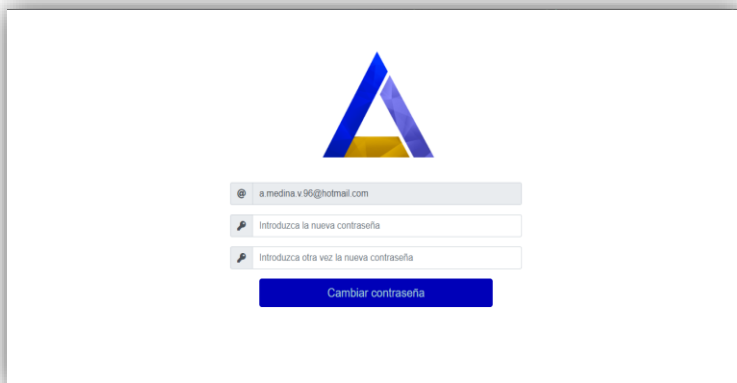
Esta cadena se cifra y se envía mediante GET a esta página.

Si cualquier variable que se envíe a esta página no está registrada en la base de datos el enlace redirige a la página de login de la aplicación, evitando así accesos no deseados.

Una vez editados los datos vuelve a verificar que el usuario y el hash coinciden para que no se pueda modificar la contraseña de otro usuario.



En esta imagen podemos ver la pantalla de recuperación de contraseña.

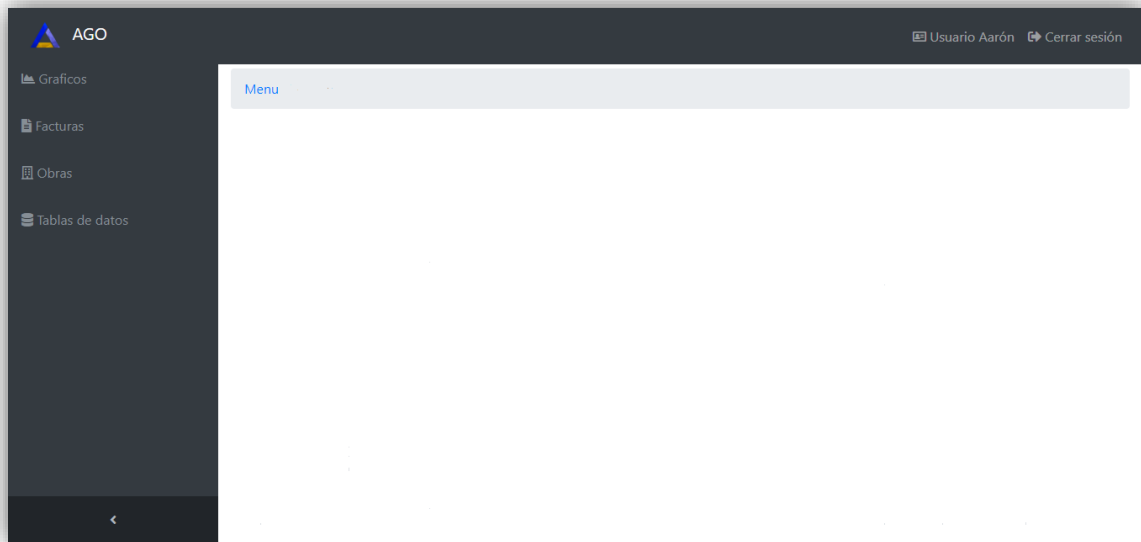


En esta imagen podemos ver la pantalla de creación de contraseña.

-INDEX.PHP:

Esta pantalla es la que aparece al loguearse en la aplicación.

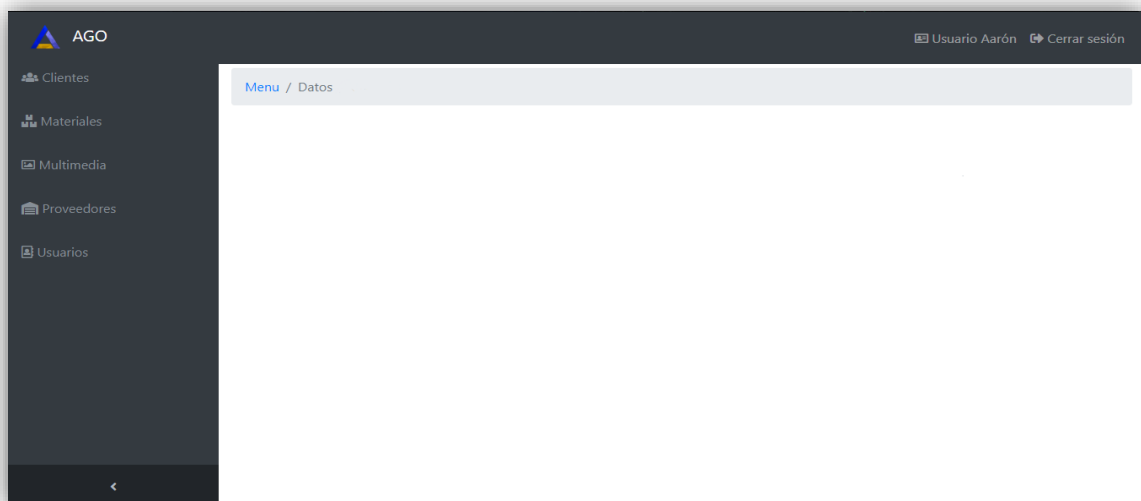
Cuando accedemos a esta página se carga por defecto la página de obras.php.



-DATOS.PHP:

En esta pantalla se cargan las pantallas con los datos de la aplicación (clientes, multimedia, materiales, proveedores y usuarios).

Tanto en la página de index.php como en datos.php se carga la página de datos del usuario.



Las pantallas secundarias se encargan de albergar el contenido de la aplicación.

En estas páginas podemos encontrar permisos, los cuales nos muestran más o menos opciones según sea nuestro nivel de permisos.

Actualmente solo se cuenta con usuarios y administradores.

Las pantallas secundarias son:

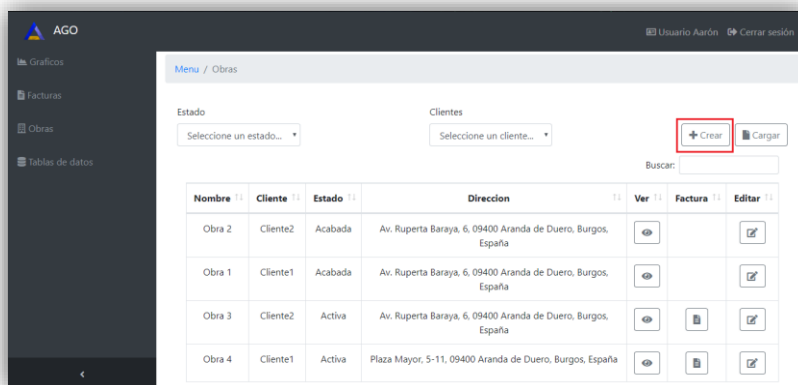
-OBRAS.PHP:

Esta ventana se encarga de la gestión de las obras.

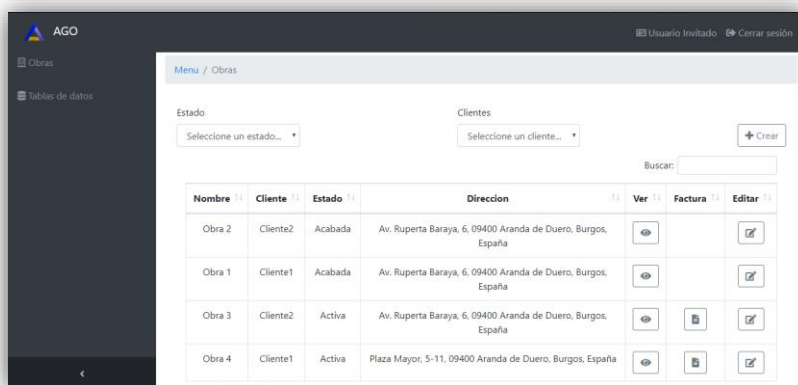
En ella podremos crear, importar, y editar obras, así como generar presupuestos y visualizar el contenido de una obra para añadir elementos.

Está formada por una datatable, en la cual podemos filtrar las obras por estado y cliente, así como ordenar cada columna o buscar un valor determinado en la caja de búsqueda.

Utiliza una conexión Ajax para recoger los datos de la base de datos, por lo cual se actualiza dinámicamente cada vez que se realiza una búsqueda.



En esta imagen podemos ver como los permisos de administrador permiten importar archivos CSV.



En esta imagen podemos ver como los permisos de usuario nos impiden visualizar el botón de importar archivos CSV.

-FACTURAS:

Esta ventana se encarga de mostrarnos las facturas ya creadas. A esta ventana solo podremos acceder si somos usuarios administradores.

Para crear una factura tendremos que acceder a la ventana de obras y con el botón de generar factura nos crea una entrada en la datatable de esta página.

Esta ventana es simplemente informativa, aunque se está trabajando en ella para poder crear las facturas en PDF.

Está formada por una datatable, en la cual podemos buscar un valor determinado en la caja de búsqueda.

Utiliza una conexión Ajax para recoger los datos de la base de datos, por lo cual se actualiza dinámicamente cada vez que se realiza una búsqueda.

Menu / Facturas

Buscar:

Obra	Cliente	Total	Fecha de finalización	Acciones
Obra 3	Cliente2	407	2018-05-28 18:19:01	

Mostrar: 10 registros

Anterior 1 Siguiente

Última actualización: Tuesday 29 de May del 2018, 09:23:50 AM

Copyright © AGO

En esta imagen podemos ver como los permisos de administrador nos permiten acceder a la ventana de facturas.

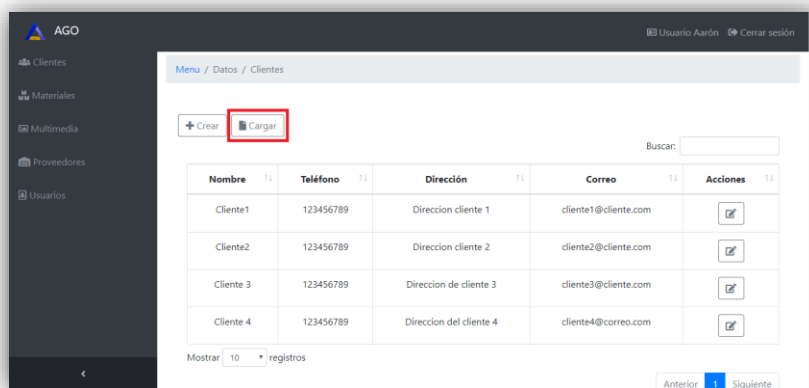
-CLIENTES:

Esta ventana se encarga de gestionar los datos de los clientes. A los clientes que introduzcamos aquí se les podrán asignar obras posteriormente.

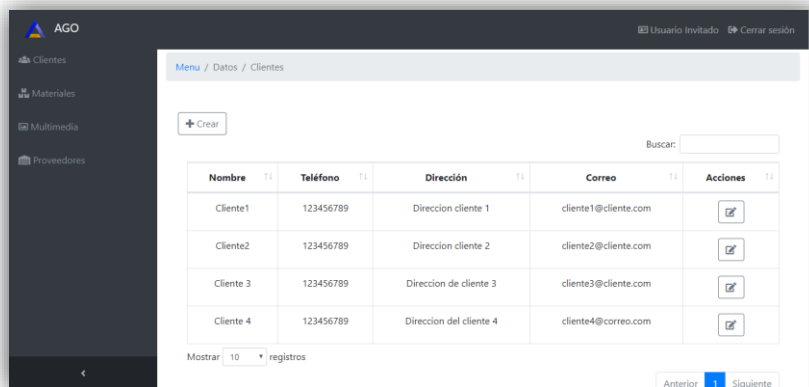
En ella podemos crear, importar y editar clientes.

Está formada por una datatable, en la cual podemos ordenar cada columna o buscar un valor determinado en la caja de búsqueda.

Utiliza una conexión Ajax para recoger los datos de la base de datos, por lo cual se actualiza dinámicamente cada vez que se realiza una búsqueda.



En esta imagen podemos ver como los permisos de administrador permiten importar archivos CSV.



En esta imagen podemos ver como los permisos de usuario nos impiden visualizar el botón de importar archivos CSV.

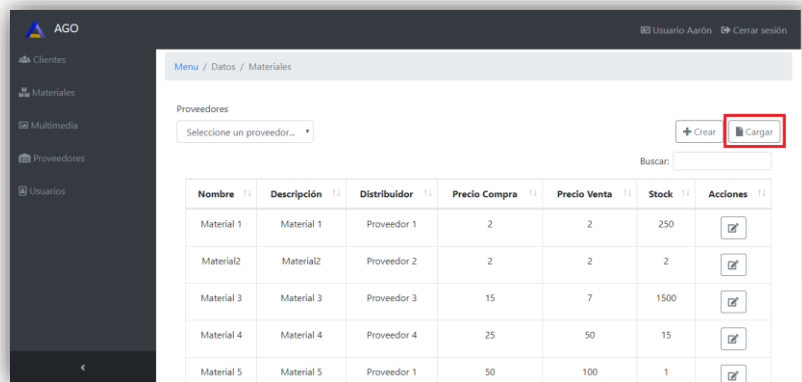
-MATERIALES:

Esta ventana se encarga de gestionar los datos de los materiales. Los materiales que introduzcamos aquí se podrán añadir posteriormente a una obra.

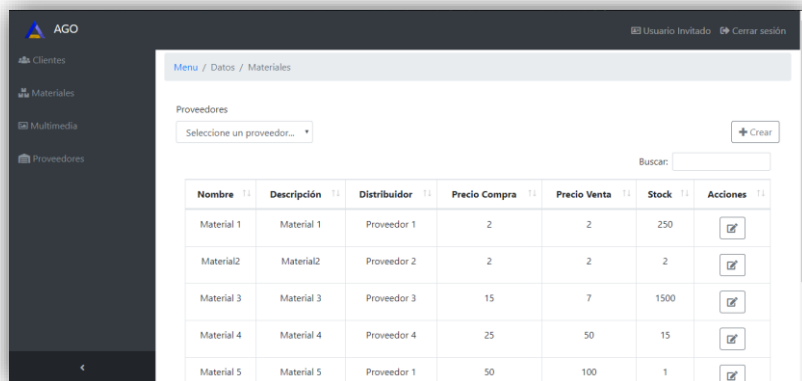
En ella podremos crear, importar, y editar materiales.

Está formada por una datatable, en la cual podemos filtrar los materiales por proveedor, así como ordenar cada columna o buscar un valor determinado en la caja de búsqueda.

Utiliza una conexión Ajax para recoger los datos de la base de datos, por lo cual se actualiza dinámicamente cada vez que se realiza una búsqueda.



En esta imagen podemos ver como los permisos de administrador permiten importar archivos CSV.



En esta imagen podemos ver como los permisos de usuario nos impiden visualizar el botón de importar archivos CSV.

-MULTIMEDIA:

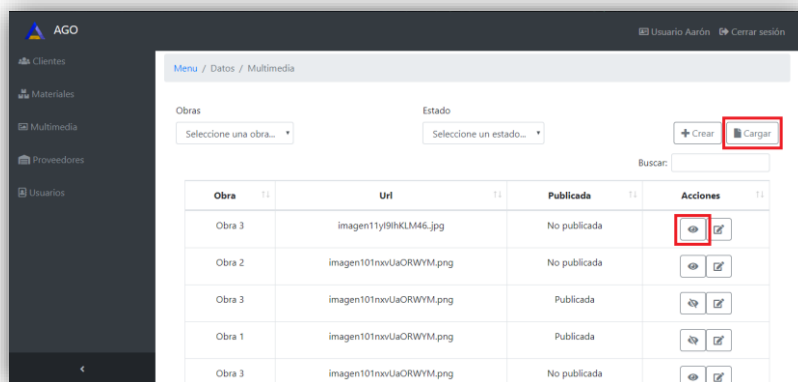
Esta ventana se encarga de gestionar los datos multimedia. Esta tabla está pensada para la actualización de la FrontPage, que actualizará sus imágenes a medida que introduzcamos datos en esta página.

Las imágenes que introduzcamos en esta tabla se guardan en el sistema de archivos del servidor, por lo que en la base de datos solo se guarda el nombre de la imagen.

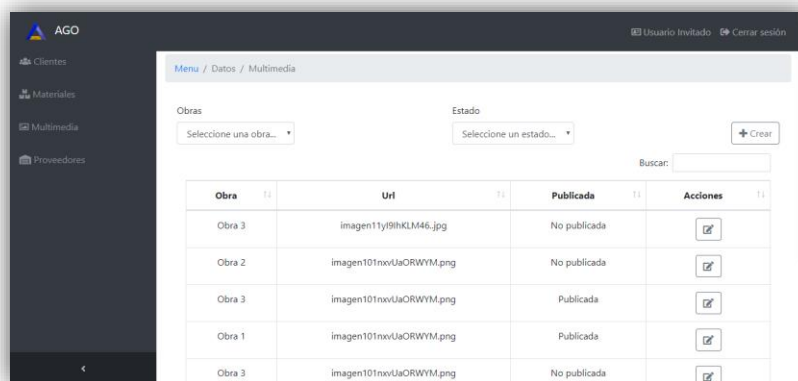
En ella podremos crear, importar, y editar las imágenes de ciertas obras.

Está formada por una datatable, en la cual podemos filtrar las entradas multimedia por obras y estado, ya que se pueden mostrar u ocultar en la FrontPage, así como ordenar cada columna o buscar un valor determinado en la caja de búsqueda.

Utiliza una conexión Ajax para recoger los datos de la base de datos, por lo cual se actualiza dinámicamente cada vez que se realiza una búsqueda.



En esta imagen podemos ver como los permisos de administrador permiten tanto importar archivos CSV como publicar y ocultar imágenes.



En esta imagen podemos ver como los permisos de usuario nos impiden visualizar el botón de importar archivos CSV y el de publicar.

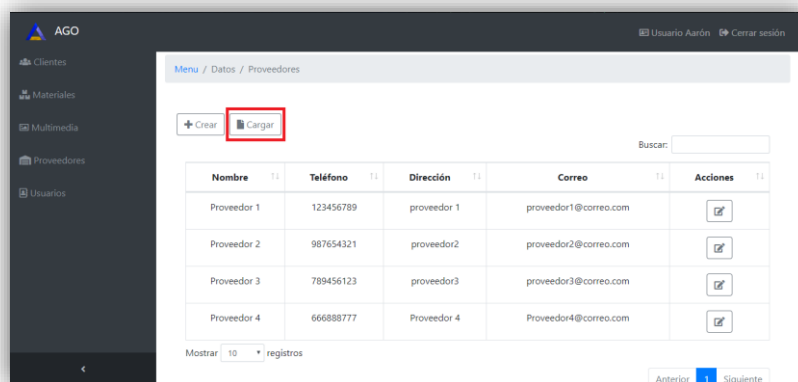
-PROVEEDORES:

Esta ventana se encarga de gestionar los datos de los proveedores. Los proveedores que introduzcamos aquí se podrán asignar a los materiales ya introducidos.

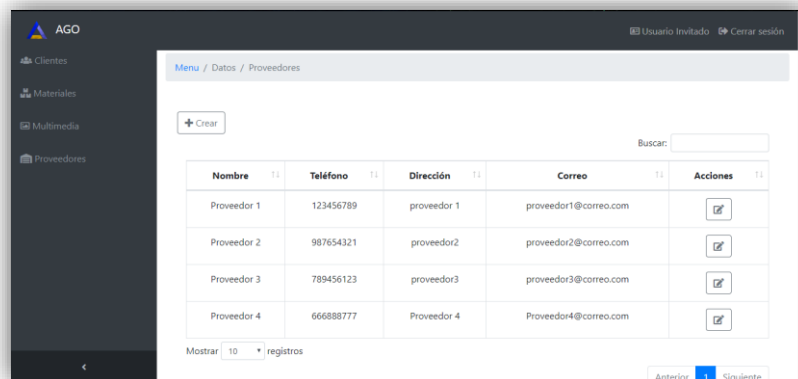
En ella podremos crear, importar, y editar los proveedores.

Está formada por una datatable, en la cual podemos ordenar cada columna o buscar un valor determinado en la caja de búsqueda.

Utiliza una conexión Ajax para recoger los datos de la base de datos, por lo cual se actualiza dinámicamente cada vez que se realiza una búsqueda.



En esta imagen podemos ver como los permisos de administrador permiten importar archivos CSV.



En esta imagen podemos ver como los permisos de usuario nos impiden visualizar el botón de importar archivos CSV.

-USUARIOS:

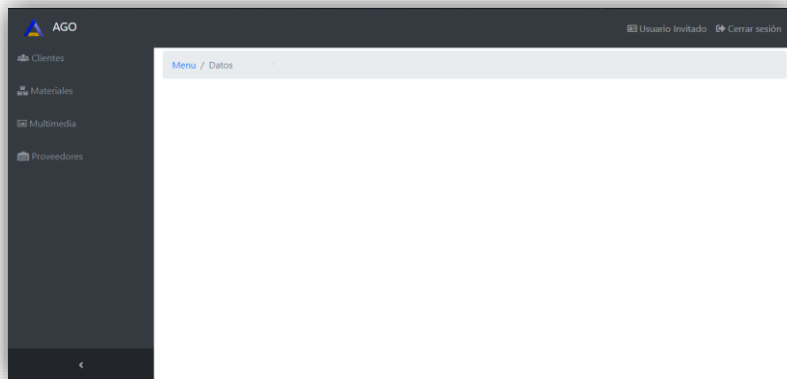
Esta ventana se encarga de añadir cuentas de usuarios a la aplicación. Para ello se ha de introducir el correo del usuario y se enviará un mensaje con la página que permite introducir una contraseña. Una vez introducida la contraseña se actualizará la información del usuario verificando la cuenta.

Además de añadir usuarios también podremos importarlos de un archivo CSV.

A esta página solo podremos acceder si poseemos permisos de administrador, permaneciendo oculta en la ventana de un usuario sin permisos.

Los usuarios que creemos desde aquí no poseerán permisos. Esto se ha realizado de esta manera para evitar vulnerabilidades en las tablas de la base de datos, así como

En esta imagen podemos ver como los permisos de administrador permiten acceder a la página de usuarios.



En esta imagen podemos ver como los permisos de usuario nos impiden acceder a la página de usuarios.

En esta imagen podemos ver la ventana de importar usuarios mediante CSV.

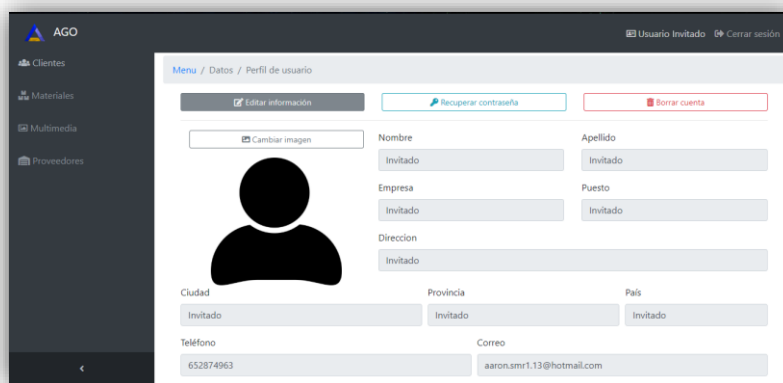
-PERFIL DE USUARIO:

Esta ventana se encarga de gestionar nuestra cuenta de usuario. Podremos editar los campos de información.

También podremos recuperar la contraseña. Para ello se nos enviará al correo un enlace con una página que nos permite actualizar nuestra contraseña. Como medida anti-spam esta opción solo se podrá utilizar cada 15 minutos. Así evitamos que cualquier aplicación externa mande correos automáticamente.

En esta pantalla también podremos eliminar nuestra cuenta. Como medida de seguridad se tendrá que introducir la contraseña de la cuenta para ello.

Por último podremos modificar la imagen del usuario. Esta imagen se guarda en el sistema de archivos del servidor, por lo que en la base de datos solo se graba el nombre de la imagen.

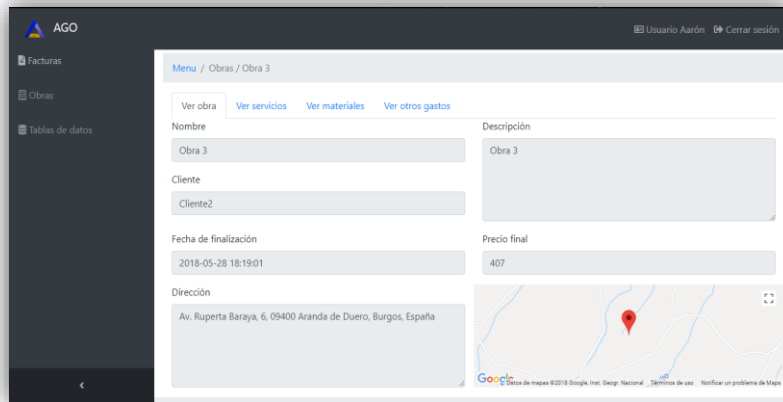
The screenshot shows the 'Perfil de usuario' (User Profile) page in the AGO application. On the left is a dark sidebar with a menu containing 'Clientes', 'Materiales', 'Multimedia', and 'Proveedores'. The main content area has a top bar with 'Menu / Datos / Perfil de usuario' and user controls 'Usuario Invitado' and 'Cerrar sesión'. Below this are three buttons: 'Editar información', 'Recuperar contraseña', and 'Borrar cuenta'. The profile form includes a 'Cambiar imagen' button and a placeholder for a user profile picture. The form fields are organized as follows: 'Nombre' and 'Apellido' (both 'Invitado'), 'Empresa' and 'Puesto' (both 'Invitado'), 'Direccion' ('Invitado'), 'Ciudad' ('Invitado'), 'Provincia' ('Invitado'), 'País' ('Invitado'), 'Teléfono' ('652874963'), and 'Correo' ('aaron.smr1.13@hotmail.com').

En esta pantalla podemos ver los datos de un usuario.

-VER OBRA:

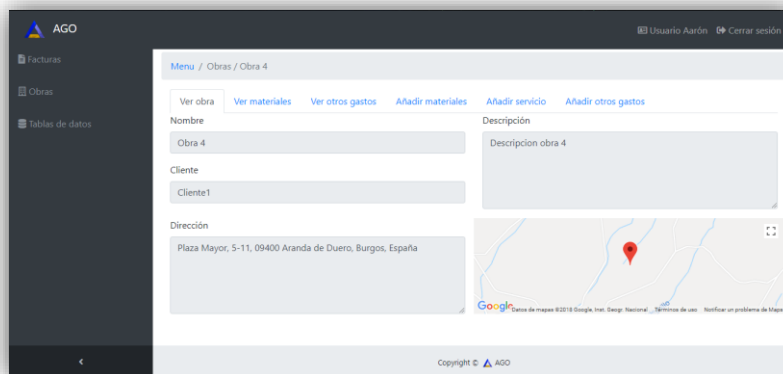
Esta ventana se encarga de gestionar mostrarnos la información de una obra.

Esta ventana se muestra de diferente manera si la obra está acabada o está todavía activa.



En esta pantalla podemos ver los datos de una obra acabada. Al estar acabada solo nos permite ver los registros ya creados. También podemos apreciar que al estar una obra acabada nos muestra la fecha de finalización y el precio

total de la obra.



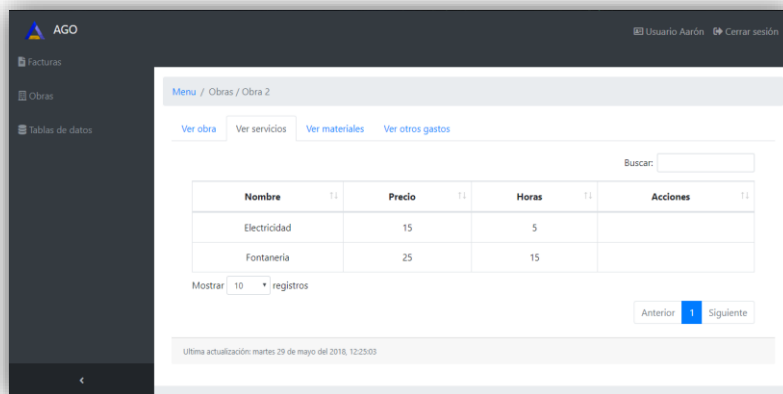
En esta pantalla podemos ver los datos de una obra activa. Al estar activa nos permite acceder a crear nuevos registros.

-VER SERVICIOS:

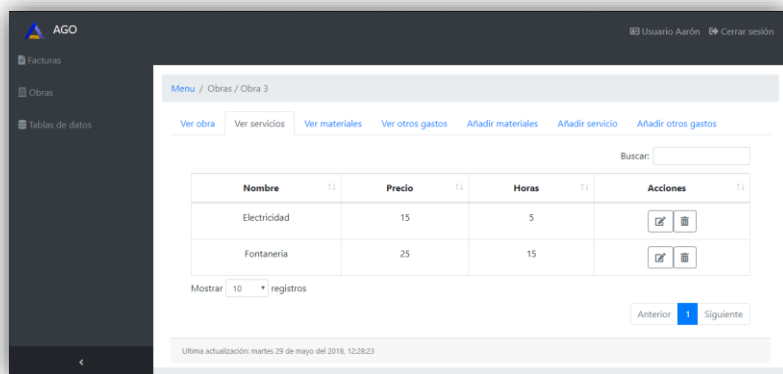
Esta ventana se encarga de mostrarnos los servicios de una obra.

Si la obra está acabada no nos permitirá modificar ni eliminar un registro.

Si la obra está todavía activa podremos editar y eliminar los registros de la datatable.



En esta pantalla podemos ver los servicios de una obra. Al estar acabada la obra los botones de modificar y eliminar desaparecen de la datatable.



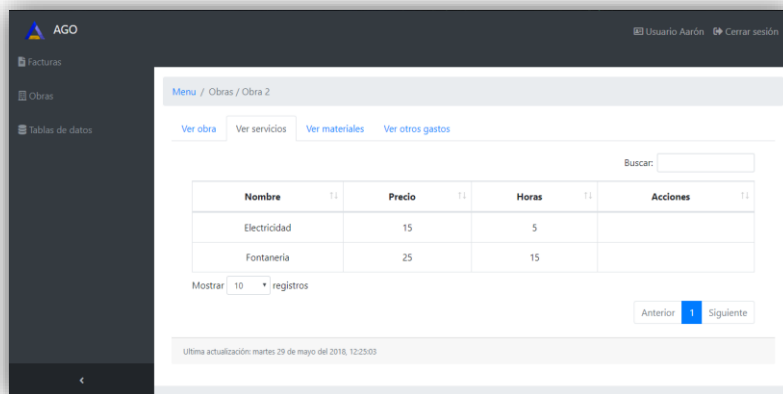
En esta pantalla podemos ver los servicios de una obra.

-VER MATERIALES:

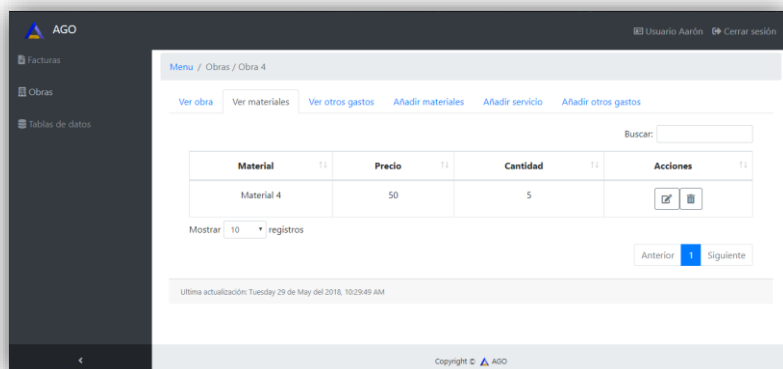
Esta ventana se encarga de mostrarnos los materiales de una obra.

Si la obra está acabada no nos permitirá modificar ni eliminar un registro.

Si la obra está todavía activa podremos editar y eliminar los registros de la datatable.



En esta pantalla podemos ver los materiales de una obra. Al estar acabada la obra los botones de modificar y eliminar desaparecen de la datatable.



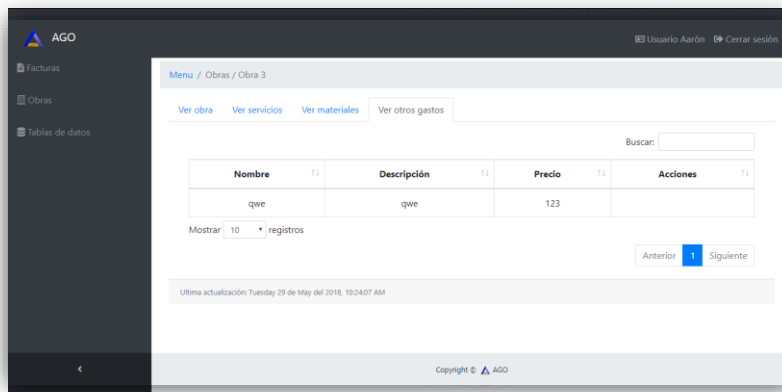
En esta pantalla podemos ver los materiales de una obra.

-VER OTROS GASTOS:

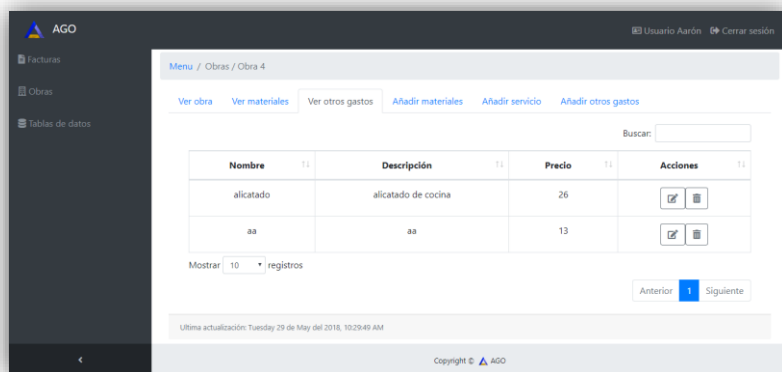
Esta ventana se encarga de mostrarnos otros gastos de una obra.

Si la obra está acabada no nos permitirá modificar ni eliminar un registro.

Si la obra está todavía activa podremos editar y eliminar los registros de la datatable.



En esta pantalla podemos ver otros gastos de una obra. Al estar acabada la obra los botones de modificar y eliminar desaparecen de la datatable.



En esta pantalla podemos ver otros gastos de una obra.

-AÑADIR MATERIAL:

Esta ventana se encarga de mostrarnos un formulario para añadir un material a una obra.

Esta ventana solamente se mostrara si la obra está activa.

El campo de materiales se rellena automáticamente con los materiales introducidos en el sistema, y el campo precio con el valor de precio del material de la base de datos.

The screenshot displays the AGO application interface. On the left is a dark sidebar with the AGO logo and navigation links: Facturas, Obras, and Tablas de datos. The top header shows the user 'Usuario Aarón' and a 'Cerrar sesión' button. The main content area has a breadcrumb 'Menu / Obras / Obra 4' and a tabbed interface with 'Ver obra', 'Ver materiales', 'Ver otros gastos', 'Añadir materiales' (active), 'Añadir servicio', and 'Añadir otros gastos'. Below the tabs, the 'Materiales' section features a dropdown menu labeled 'Seleccione un material...'. The 'Precio' section has a text input field labeled 'Precio' and a blue 'Guardar material' button. The 'Cantidad' section has a text input field labeled 'Cantidad'. The footer includes 'Copyright © AGO'.

-AÑADIR SERVICIO:

Esta ventana se encarga de mostrarnos un formulario para añadir un servicio a una obra.

Esta ventana solamente se mostrara si la obra está activa.

The screenshot displays the AGO application interface. On the left is a dark sidebar with the AGO logo and navigation links: Facturas, Obras, and Tablas de datos. The top header shows the user 'Usuario Aarón' and a 'Cerrar sesión' button. The main content area has a breadcrumb 'Menu / Obras / Obra 4' and a tabbed interface with options: Ver obra, Ver materiales, Ver otros gastos, Añadir materiales, Añadir servicio (selected), and Añadir otros gastos. The 'Añadir servicio' form includes a 'Nombre' field, a 'Precio' field, and a 'Horas' field. A blue 'Guardar servicio' button is at the bottom left of the form. The footer contains a copyright notice 'Copyright © AGO'.

-AÑADIR OTRO GASTO:

Esta ventana se encarga de mostrarnos un formulario para añadir otro gasto a una obra.

Esta ventana solamente se mostrara si la obra está activa.

The screenshot displays the AGO application interface. On the left is a dark sidebar with the AGO logo and menu items: Facturas, Obras, and Tablas de datos. The main content area has a top navigation bar with the breadcrumb 'Menu / Obras / Obra 4' and a sub-menu with options: Ver obra, Ver materiales, Ver otros gastos, Añadir materiales, Añadir servicio, and Añadir otros gastos (which is highlighted). Below the sub-menu is a form titled 'Descripción' with a large text input field. To the right of this field are two input fields labeled 'Nombre' and 'Precio'. A blue button labeled 'Guardar otro gasto' is positioned below the 'Descripción' field. The bottom of the interface shows a footer with 'Copyright © AGO'.

En las pantallas secundarias también cargamos información en forma de ventana modal.

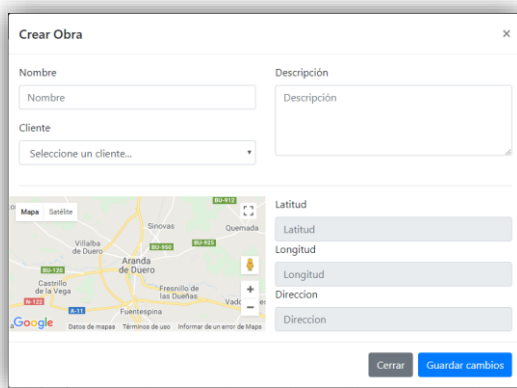
Principalmente las ventanas que mostramos son las de creación, edición e importación, aunque en la ventana de datos del usuario las ventanas se utilizan para eliminar la cuenta, cambiar la imagen y recuperar la contraseña.

Otra ventana que cargamos en una modal es la de logout.

Las ventanas modales que utilizamos son:

-CREACIÓN:

●Crear obra:

The 'Crear Obra' modal form contains several input fields. At the top, there is a 'Nombre' field and a 'Descripción' text area. Below these is a 'Cliente' dropdown menu with the placeholder text 'Seleccione un cliente...'. The lower half of the form features a Google Map on the left and a set of input fields on the right for 'Latitud', 'Longitud', and 'Dirección'. At the bottom right, there are two buttons: 'Cerrar' (grey) and 'Guardar cambios' (blue).

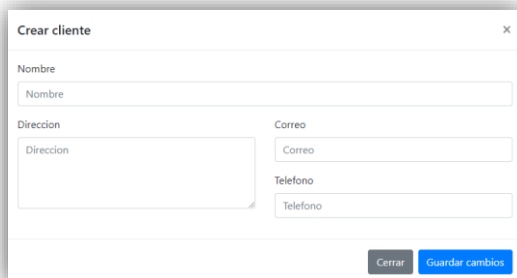
En esta ventana creamos una obra nueva. Aquí podremos introducir un nombre, cliente, descripción y localización de la obra.

La localización se introduce mediante coordenadas en el api de google maps.

Cuando hacemos marcamos una coordenada en el mapa se actualizan los campos de latitud, longitud y dirección, y también se crea un marcador que nos indica el punto del

mapa.

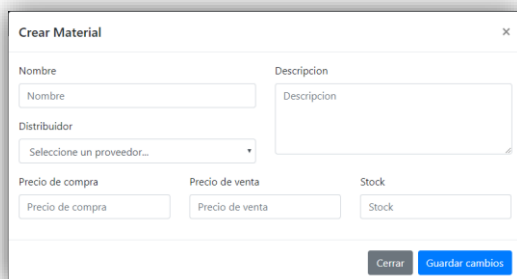
●Crear cliente:

The 'Crear cliente' modal form includes input fields for 'Nombre', 'Dirección', 'Correo', and 'Teléfono'. The 'Dirección' field is a text area, while the others are single-line text boxes. At the bottom right, there are 'Cerrar' and 'Guardar cambios' buttons.

En esta ventana damos de alta a un cliente nuevo.

El sistema no nos permitirá duplicar correos de usuarios, por lo que evitamos clientes duplicados.

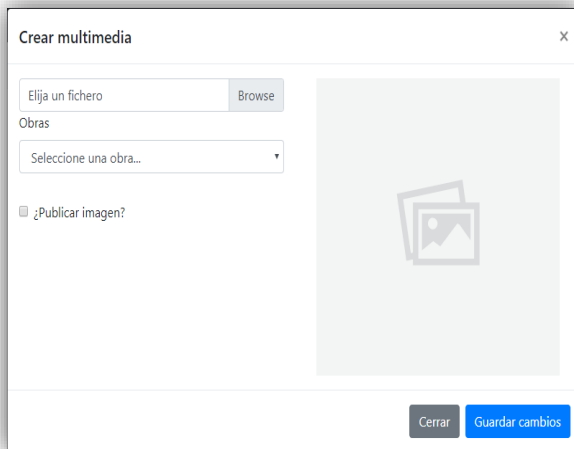
●Crear material:

The 'Crear Material' modal form has input fields for 'Nombre', 'Descripción', 'Distribuidor', 'Precio de compra', 'Precio de venta', and 'Stock'. The 'Distribuidor' field is a dropdown menu. At the bottom right, there are 'Cerrar' and 'Guardar cambios' buttons.

En esta ventana insertaremos un material nuevo.

Aquí introduciremos su nombre, una breve descripción, que distribuidor lo vende, el precio de venta, el precio de compra y el stock de material del que disponemos.

●Crear multimedia:

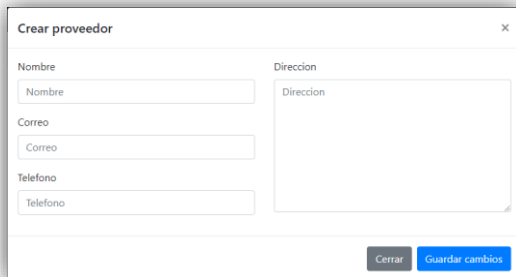
The screenshot shows a web form titled "Crear multimedia". It has a close button (X) in the top right corner. The form contains a file selection area with a text input "Elija un fichero" and a "Browse" button. Below this is a section labeled "Obras" with a dropdown menu "Seleccione una obra...". There is a checkbox labeled "¿Publicar imagen?". On the right side of the form is a large light gray area with a small icon of two overlapping photos, representing a preview. At the bottom right are two buttons: "Cerrar" (gray) and "Guardar cambios" (blue).

En esta ventana insertaremos una imagen en una obra.

Aquí elegiremos la foto que queramos de nuestro ordenador, una obra y podremos elegir si la publicamos o no, aunque esto lo podremos hacer también desde la datatable.

Cuando elijamos una foto nos la mostrará en nuestra parte derecha de la ventana, a modo de pre visualización.

●Crear proveedor:

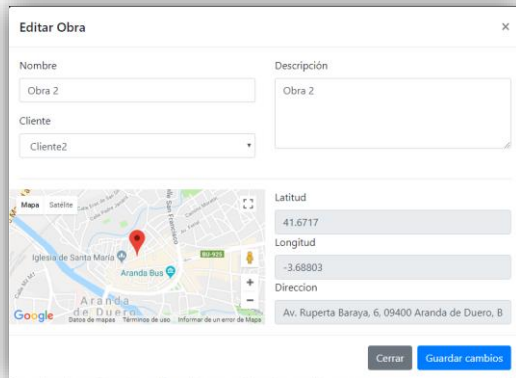
The screenshot shows a web form titled "Crear proveedor". It has a close button (X) in the top right corner. The form is divided into two main sections. The left section contains three text input fields labeled "Nombre", "Correo", and "Telefono". The right section is labeled "Direccion" and contains a larger text area. At the bottom right are two buttons: "Cerrar" (gray) and "Guardar cambios" (blue).

En esta ventana insertaremos un proveedor nuevo.

Aquí introduciremos su nombre, un correo que no se podrá repetir, un teléfono y una dirección.

-MODIFICACIÓN:

●Modificar obra:

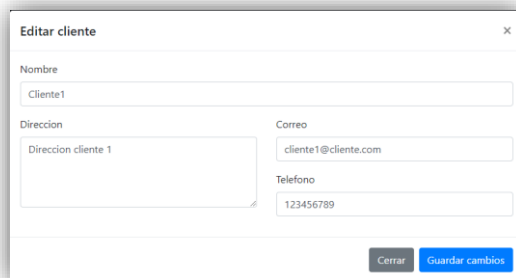


En esta ventana editaremos la información de una obra.

Tendremos los mismos datos que en la ventana de crear obra, solo que los campos estarán rellenos con la información de la obra que vayamos a editar.

El api de google maps indicará la ubicación de la obra en el mapa con el marcador.

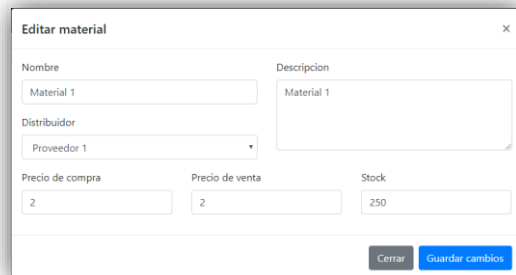
●Modificar cliente:



En esta ventana editaremos la información de un cliente.

Tendremos los mismos datos que en la ventana de crear cliente, solo que los campos estarán rellenos con la información del cliente que vayamos a editar.

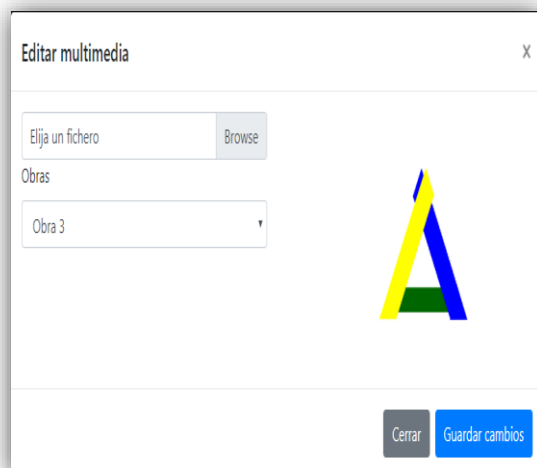
●Modificar material:



En esta ventana editaremos la información de un material.

Tendremos los mismos datos que en la ventana de crear material, solo que los campos estarán rellenos con la información del material que vayamos a editar.

●Modificar multimedia:

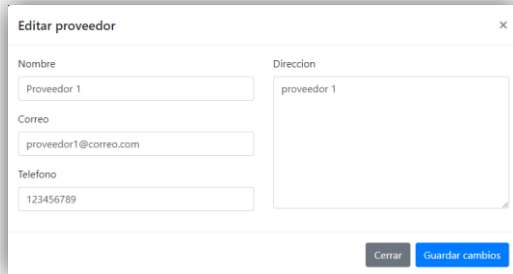


En esta ventana editaremos la información de una entrada multimedia.

Tendremos los mismos datos que en la ventana de crear multimedia, solo que los campos estarán rellenos con la información multimedia que vayamos a editar.

Aquí tendremos la visualización de la imagen que tenemos insertada, y cambiará cuando insertemos una nueva.

●Modificar proveedor:

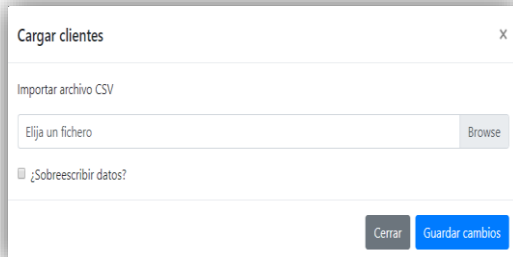


En esta ventana editaremos la información de un proveedor.

Tendremos los mismos datos que en la ventana de crear material, solo que los campos estarán rellenos con la información del proveedor que vayamos a editar.

-IMPORTACIÓN, LOGOUT Y VENTANAS DE LOS DATOS DE USUARIO:

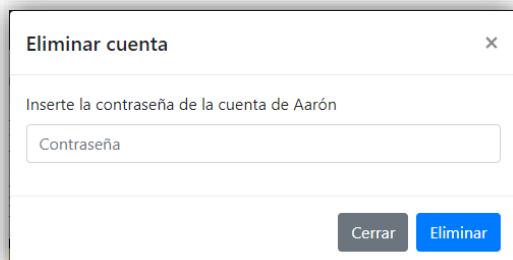
●Importar tablas:



En esta ventana importaremos la información de las distintas tablas. Cada página tiene una, pero todas tienen el mismo aspecto.

Solo podremos añadir archivos con extensión CSV, y esta ventana solo será visible si la cuenta es administradora.

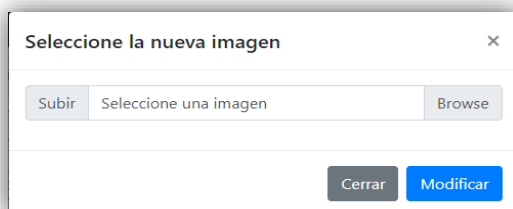
●Borrar usuario:



Esta ventana nos permite eliminar una cuenta de usuario, pero para ello tendremos que introducir la contraseña de este.

Si la contraseña es errónea o vacía no nos permitirá borrar la cuenta.

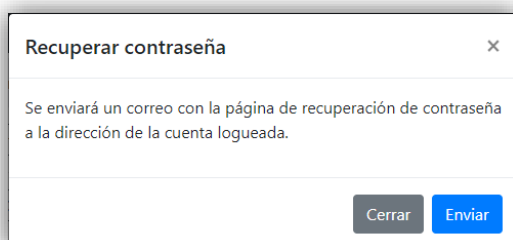
●Cambiar imagen de usuario:



En esta ventana modificaremos la imagen del usuario.

Solo podremos añadir imágenes con extensión jpg y png, el resto las rechazará.

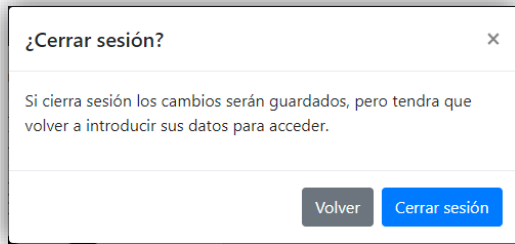
●Importar tablas:



Esta ventana nos permite enviar un correo al email del usuario logueado con el enlace para cambiar la contraseña.

Solo se permitirá enviar un correo cada 15 minutos.

●Logout:



En esta ventana cerraremos la sesión del programa.

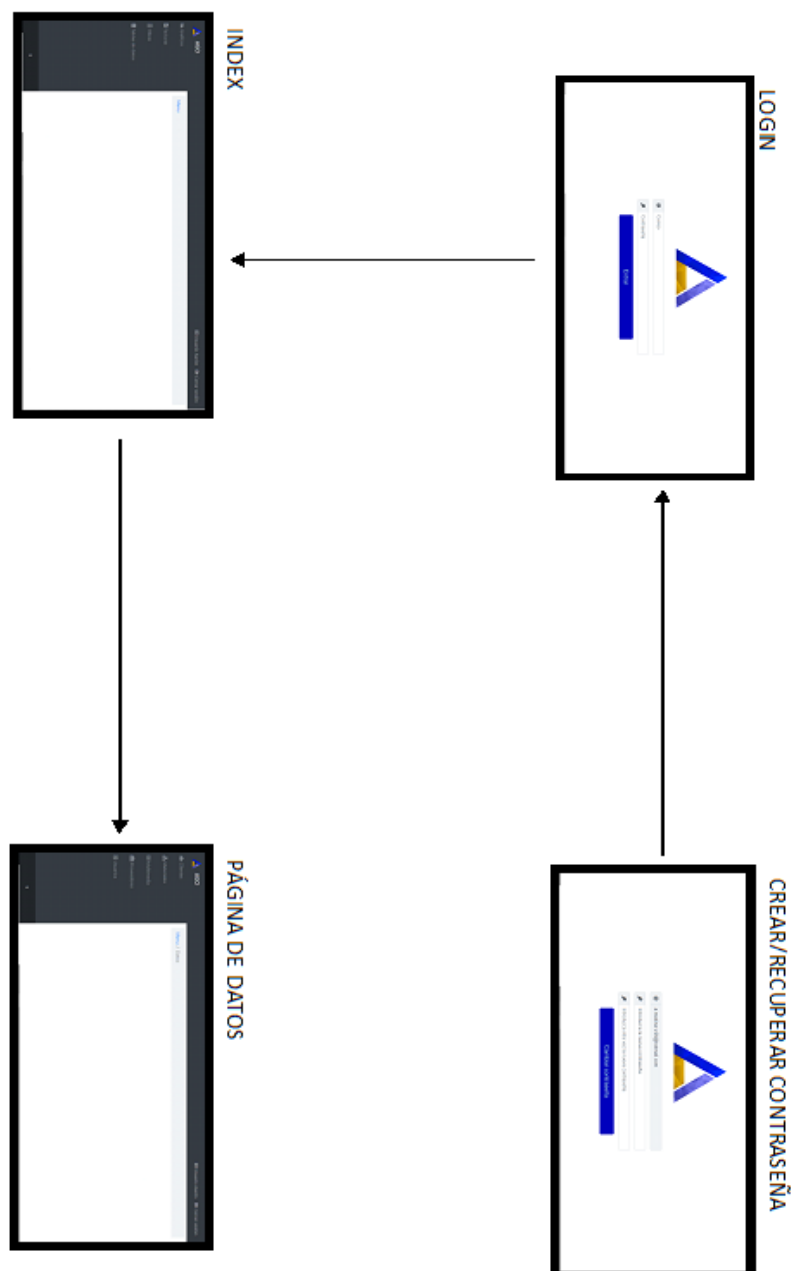
PANTALLAS ->RELACIÓN

En este apartado mostraré las ventanas que son accesibles desde cada ventana del programa.

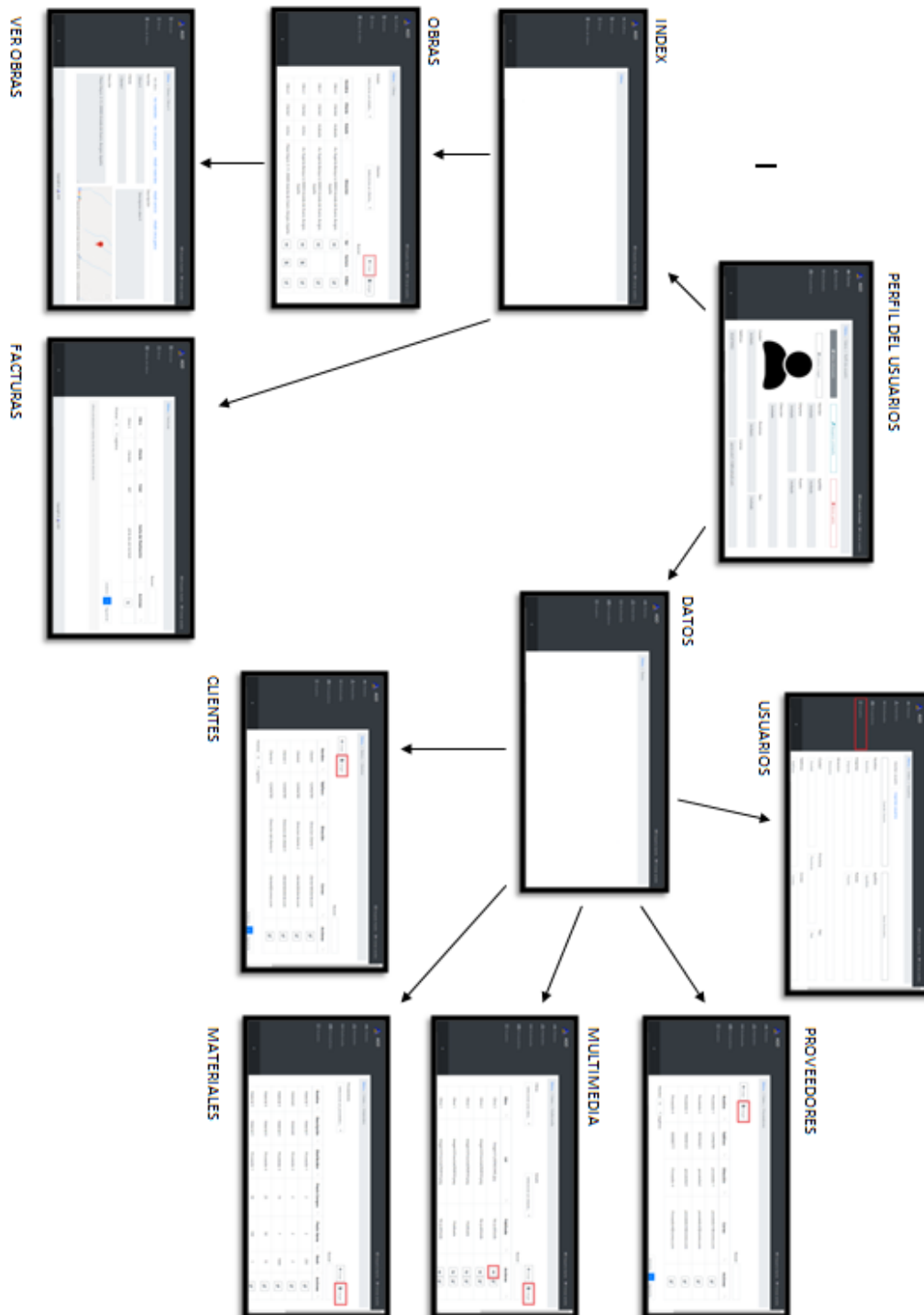
Omitiré las ventanas modales y me centraré en las páginas principales y secundarias.

La ruta comienza desde la ventana de login, ya que es la entrada de la aplicación.

-PANTALLAS PRINCIPALES:

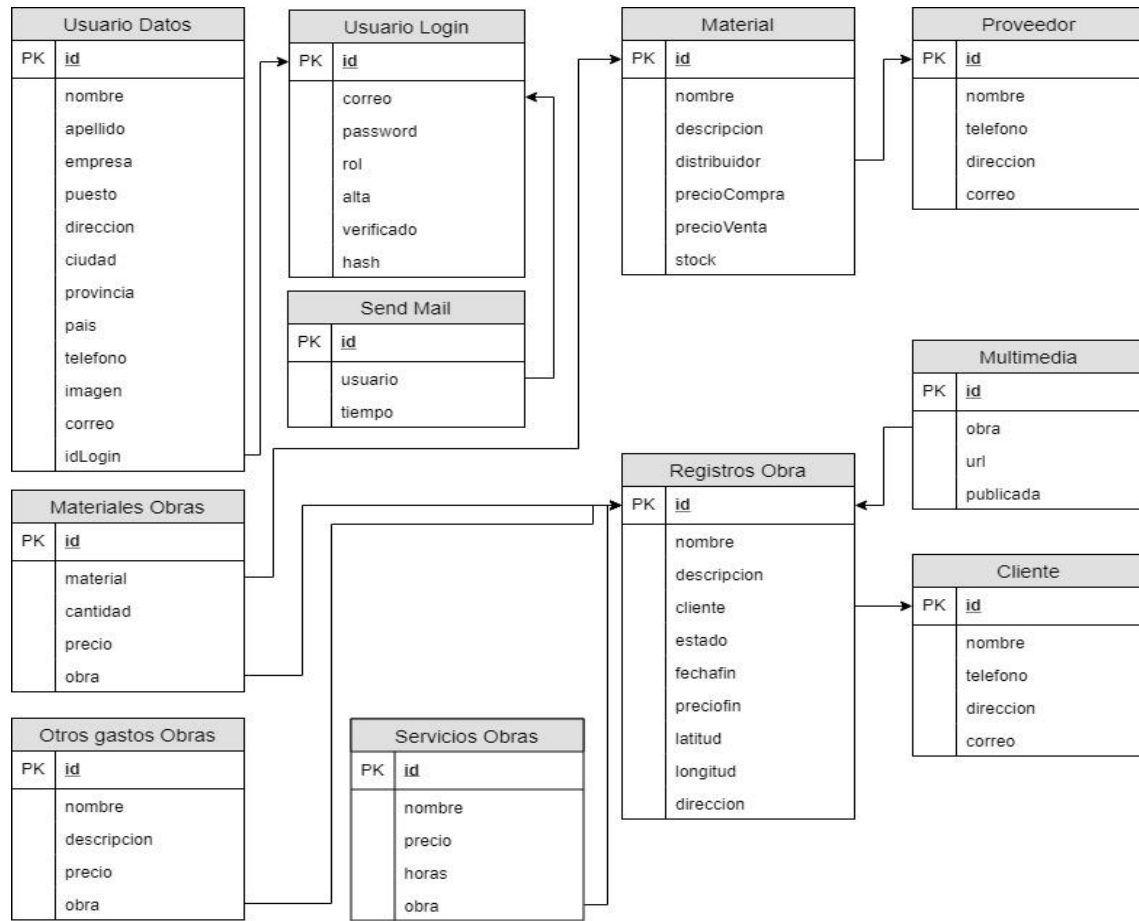


-PANTALLAS SECUNDARIAS:



BASE DE DATOS->DIAGRAMAS

-ENTIDAD RELACIÓN:



BASE DE DATOS->TABLAS

La aplicación se compone de once tablas, las cuales proporcionan la información a las diferentes pantallas.

En la siguiente imagen podemos ver una lista de todas las tablas.

Tabla	Acción	Filas	Tipo	Cotejamiento	Tamaño	Residuo a depurar
<input type="checkbox"/> cliente	★ Examinar Estructura Buscar Insertar Vaciar Eliminar	4	InnoDB	utf8_spanish_ci	64 KB	-
<input type="checkbox"/> facturas	★ Examinar Estructura Buscar Insertar Vaciar Eliminar	1	InnoDB	utf8_spanish_ci	16 KB	-
<input type="checkbox"/> material	★ Examinar Estructura Buscar Insertar Vaciar Eliminar	5	InnoDB	utf8_spanish_ci	64 KB	-
<input type="checkbox"/> materialesobras	★ Examinar Estructura Buscar Insertar Vaciar Eliminar	5	InnoDB	utf8_spanish_ci	16 KB	-
<input type="checkbox"/> multimedia	★ Examinar Estructura Buscar Insertar Vaciar Eliminar	6	InnoDB	utf8_spanish_ci	48 KB	-
<input type="checkbox"/> otros gastosobras	★ Examinar Estructura Buscar Insertar Vaciar Eliminar	4	InnoDB	utf8_spanish_ci	16 KB	-
<input type="checkbox"/> proveedor	★ Examinar Estructura Buscar Insertar Vaciar Eliminar	4	InnoDB	utf8_spanish_ci	48 KB	-
<input type="checkbox"/> registrosobra	★ Examinar Estructura Buscar Insertar Vaciar Eliminar	4	InnoDB	utf8_spanish_ci	48 KB	-
<input type="checkbox"/> sendmail	★ Examinar Estructura Buscar Insertar Vaciar Eliminar	7	InnoDB	utf8_spanish_ci	16 KB	-
<input type="checkbox"/> serviciosobras	★ Examinar Estructura Buscar Insertar Vaciar Eliminar	2	InnoDB	utf8_spanish_ci	16 KB	-
<input type="checkbox"/> usuariodatos	★ Examinar Estructura Buscar Insertar Vaciar Eliminar	4	InnoDB	utf8_spanish_ci	16 KB	-
<input type="checkbox"/> usuariologin	★ Examinar Estructura Buscar Insertar Vaciar Eliminar	4	InnoDB	utf8_spanish_ci	48 KB	-
12 tablas	Número de filas	50	MyISAM	latin1_swedish_ci	416 KB	0 B

La base de datos utiliza una codificación UTF-8 para evitar caracteres extraños en los registros.

La herramienta utilizada para gestionar la base de datos es phpMyAdmin.

En la aplicación he prescindido de la relación de tablas en phpMyAdmin, ya que podría generar errores en el programa con ciertas funcionalidades, pero controlo las relaciones en la aplicación.

A continuación podemos ver cada tabla más detalladamente:

-TABLA DATOS USUARIO:

Esta tabla proporciona información a un usuario. Es simplemente informativa, y se utiliza solamente en la ventana de información de usuario.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	id	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Primaria Más
2	nombre	varchar(250)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más
3	apellido	varchar(250)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más
4	empresa	varchar(250)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más
5	puesto	varchar(250)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más
6	direccion	varchar(250)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más
7	ciudad	varchar(250)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más
8	provincia	varchar(250)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más
9	pais	varchar(250)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más
10	telefono	int(11)			No	Ninguna			Cambiar Eliminar Primaria Más
11	imagen	varchar(2500)	utf8_spanish_ci		Sí	NULL			Cambiar Eliminar Primaria Más
12	correo	varchar(250)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más
13	idLogin	int(11)			No	Ninguna			Cambiar Eliminar Primaria Más

Contiene la siguiente información:

Nombre --> Nombre del usuario.

Apellidos --> Apellidos del usuario.

Empresa --> Empresa para la que trabaja.

Puesto --> El puesto en la empresa.

Dirección --> Dirección donde vive.

Ciudad --> Ciudad donde vive.

Provincia --> Provincia donde vive.

Teléfono --> Teléfono del usuario.

Imagen --> Imagen de perfil del usuario.

Correo --> Correo del usuario.

-TABLA LOGIN USUARIO:

Esta tabla es la que nos proporciona la información de acceso del usuario.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	id	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Primaria Más
2	correo	varchar(250)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más
3	password	varchar(60)	utf8_spanish_ci		Si	NULL			Cambiar Eliminar Primaria Más
4	rol	varchar(250)	utf8_spanish_ci		No	usuario			Cambiar Eliminar Primaria Más
5	alta	timestamp			No	CURRENT_TIMESTAMP			Cambiar Eliminar Primaria Más
6	verificado	tinyint(1)			No	0			Cambiar Eliminar Primaria Más
7	hash	varchar(250)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más

Contiene la siguiente información:

Correo --> Correo de acceso a la aplicación. También es el correo al que se enviarán notificaciones.

Password --> Contraseña encriptada mediante blowfish.

Rol --> Nivel de acceso del usuario. Los dos niveles implementados son: Usuario y Administrador.

Alta --> Fecha de creación del usuario.

Verificado --> Si el usuario no ha confirmado su cuenta y creado una contraseña no tendrá la cuenta verificada.

Hash --> Cadena de caracteres generada automáticamente que proporciona información para autenticar al usuario cuando se crea o recupera la contraseña.

-TABLA CLIENTE:

Esta tabla es la que nos proporciona la información de los clientes.

De esta tabla se extrae la información de un cliente a la hora de crear una obra o una factura.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	id	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Primaria Único Más
2	nombre	varchar(20)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Único Más
3	telefono	int(11)			No	Ninguna			Cambiar Eliminar Primaria Único Más
4	direccion	varchar(30)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Único Más
5	correo	varchar(20)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Único Más

Contiene la siguiente información:

Nombre --> Nombre del cliente.

Teléfono --> Teléfono del cliente.

Dirección --> Dirección del cliente.

Correo --> Correo del cliente.

-TABLA MATERIAL:

Esta tabla es la que nos proporciona la información de los materiales.

De esta tabla se extrae la información de un material a la hora de crear una obra.

Tiene relación con la tabla de proveedores.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	id	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Primaria Más
2	nombre	varchar(20)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más
3	descripcion	varchar(200)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más
4	distribuidor	varchar(250)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más
5	preciocompra	int(11)			No	Ninguna			Cambiar Eliminar Primaria Más
6	precioventa	int(11)			No	Ninguna			Cambiar Eliminar Primaria Más
7	stock	int(11)			No	Ninguna			Cambiar Eliminar Primaria Más

Contiene la siguiente información:

Nombre --> Nombre del material.

Descripción --> Descripción del material.

Distribuidor --> Distribuidor que proporciona el material. Se obtiene de la tabla de distribuidores.

Precio de compra --> Precio al que se compra el material.

Precio de venta --> Precio al que se vende el material.

Stock --> Cantidad de material que tenemos disponible.

-TABLA MATERIAL DE OBRAS:

Esta tabla es la que nos proporciona la información de los materiales que utilizamos en una obra.

Tiene relación con la tabla de obras.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	id	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Primaria Único Más
2	material	varchar(250)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Único Más
3	cantidad	int(11)			No	Ninguna			Cambiar Eliminar Primaria Único Más
4	precio	float			No	Ninguna			Cambiar Eliminar Primaria Único Más
5	obra	int(11)			No	Ninguna			Cambiar Eliminar Primaria Único Más

Contiene la siguiente información:

Material --> Nombre del material que incluiremos en una obra.

Cantidad --> Cantidad del material que hemos empleado en una obra.

Precio --> Precio del material. Este campo nos lo proporciona la tabla de materiales.

Obra --> Obra en la que emplearemos el material.

-TABLA MULTIMEDIA:

Esta tabla es la que utilizamos para guardar fotos de obras que se mostrarán en la FrontPage.

Tiene relación con la tabla de obras.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	id	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Primaria Único Más
2	obra	varchar(250)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Único Más
3	url	varchar(100)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Único Más
4	publicada	tinyint(1)			No	Ninguna			Cambiar Eliminar Primaria Único Más

Contiene la siguiente información:

Obra --> Obra a la que aplicaremos la imagen.

Url --> Nombre de la imagen. Las imágenes están alojadas en una carpeta dentro del servidor.

Publicada --> Estado de un material. Puede publicarse o no publicarse. Si se publica aparecerá en la FrontPage.

-TABLA OTROS GASTOS DE OBRAS:

Esta tabla es la que nos proporciona la información de cualquier gasto que no sea de material o servicios en una obra.

Tiene relación con la tabla de obras.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
<input type="checkbox"/> 1	id	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Primaria ▼ Más
<input type="checkbox"/> 2	nombre	varchar(250) utf8_spanish_ci			No	Ninguna			Cambiar Eliminar Primaria ▼ Más
<input type="checkbox"/> 3	descripcion	varchar(250) utf8_spanish_ci			No	Ninguna			Cambiar Eliminar Primaria ▼ Más
<input type="checkbox"/> 4	precio	int(11)			No	Ninguna			Cambiar Eliminar Primaria ▼ Más
<input type="checkbox"/> 5	obra	int(11)			No	Ninguna			Cambiar Eliminar Primaria ▼ Más

Contiene la siguiente información:

Nombre --> Nombre del gasto que incluiremos en una obra.

Descripción --> Descripción del gasto.

Precio --> Precio que asignaremos al gasto.

Obra --> Obra a la que asignaremos el gasto.

-TABLA PROVEEDORES:

Esta tabla es la que nos proporciona la información sobre los proveedores que nos proporcionan el material.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	id	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Primaria Único Más
2	nombre	varchar(20)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Único Más
3	telefono	int(11)			No	Ninguna			Cambiar Eliminar Primaria Único Más
4	direccion	varchar(30)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Único Más
5	correo	varchar(50)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Único Más

Contiene la siguiente información:

Nombre --> Nombre del proveedor.

Teléfono --> Descripción del proveedor.

Dirección --> Dirección del proveedor.

Correo --> Correo del proveedor.

-TABLA REGISTROS DE OBRAS:

Esta tabla es la que nos proporciona la información sobre las obras que estamos realizando o que hemos realizado.

Tiene relación con la tabla de clientes.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	id	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Primaria Más
2	nombre	varchar(50)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más
3	descripcion	varchar(200)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más
4	cliente	varchar(250)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más
5	estado	enum('Acabada', 'Activa')	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más
6	fechafin	varchar(20)	utf8_spanish_ci		Sí	NULL			Cambiar Eliminar Primaria Más
7	preciofin	int(11)			Sí	NULL			Cambiar Eliminar Primaria Más
8	latitud	float			No	Ninguna			Cambiar Eliminar Primaria Más
9	longitud	float			No	Ninguna			Cambiar Eliminar Primaria Más
10	direccion	varchar(500)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Más

Contiene la siguiente información:

Nombre --> Nombre de la obra.

Descripción --> Descripción de la obra.

Cliente --> Cliente de la obra.

Estado --> Estado de la obra. Puede estar activa o acabada. Si esta activa el precio final y la fecha final estarán vacías hasta que se cambie de estado. Cuando la obra esté acabada, se actualizarán el precio y la fecha y se generará una factura con la información de la obra.

Fecha fin --> Fecha de finalización de la obra.

Precio fin --> Precio final de la obra. Incluye los materiales, los servicios y los otros gastos.

Latitud --> Latitud de la dirección. Solo se utiliza para posicionar el puntero del mapa.

Longitud --> Longitud de la dirección. Solo se utiliza para posicionar el puntero del mapa.

Dirección --> Dirección de la obra. Se obtiene por la latitud y la longitud.

-TABLA SEND MAIL:

Esta tabla es la que nos deshabilita el envío de correos de recuperación de contraseña si hemos utilizado este servicio en los últimos 15 minutos.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	id	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Primaria ▼ Más
2	usuario	varchar(250)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria ▼ Más
3	tiempo	timestamp			No	CURRENT_TIMESTAMP			Cambiar Eliminar Primaria ▼ Más

Contiene la siguiente información:

Usuario --> Usuario que ha enviado el correo.

Tiempo --> Fecha y hora en la que se ha mandado el último correo. Para evitar que se manden correos antes de los 15 minutos el sistema coge la hora actual y la compara con el tiempo 15 minutos más tarde de este campo.

-TABLA SERVICIOS OBRAS:

Esta tabla es la que nos proporciona la información de los servicios contratados para una obra.

Tiene relación con la tabla de obras.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	id	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar Eliminar Primaria Único Más
2	nombre	varchar(250)	utf8_spanish_ci		No	Ninguna			Cambiar Eliminar Primaria Único Más
3	precio	float			No	Ninguna			Cambiar Eliminar Primaria Único Más
4	horas	int(11)			No	Ninguna			Cambiar Eliminar Primaria Único Más
5	obra	int(11)			No	Ninguna			Cambiar Eliminar Primaria Único Más

Contiene la siguiente información:

Nombre --> Nombre del servicio que incluiremos en una obra.

Precio --> Precio por horas del servicio.

Horas --> Tiempo para la realización del servicio.

Obra --> Obra a la que asignaremos el servicio.

-TABLA FACTURAS:

En esta tabla guardamos la información básica para poder crear un PDF de una factura.

Tiene relación con la tabla de obras y clientes.

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Comentarios	Extra	Acción
1	id	int(11)			No	Ninguna		AUTO_INCREMENT	Cambiar Elim
2	obra	int(11)			No	Ninguna			Cambiar Elim
3	cliente	varchar(250)	utf8_spanish_ci		No	Ninguna			Cambiar Elim
4	total	float			No	Ninguna			Cambiar Elim
5	fechaFin	timestamp		on update CURRENT_TIMESTAMP	No	CURRENT_TIMESTAMP		ON UPDATE CURRENT_TIMESTAMP	Cambiar Elim

Contiene la siguiente información:

Obra --> Obra de la cual generamos la factura.

Cliente --> Cliente al que generamos la factura.

Total --> Precio total de la obra. Este campo lo obtenemos al sumar los siguientes campos:

-Precio de servicios = Se obtiene al calcular el total del precio de los servicios por las horas.

-Precio de materiales = Se obtiene al calcular el total del precio de los materiales por la cantidad empleada en la obra.

-Precio de otros gastos = Se obtiene al calcular el total del precio de los otros gastos.

Fecha fin --> Fecha de creación de la factura. Se genera automáticamente cuando cambiamos el estado de una obra a acabada.

SEGURIDAD

Para evitar que cualquiera tenga acceso a los datos de la aplicación se crearon diversos mecanismos de seguridad en la aplicación.

Los principales objetivos de la seguridad son:

-CONTRASEÑAS DE USUARIOS:

Para proteger las contraseñas contra ataques se ha realizado un sistema de cifrado mediante [blowfish](#).

Esto crea una cadena de 60 caracteres ilegible.

```
password
$2y$07$krctYWNmqCszXNXtIWEqv.E/iogkoBb4hgY.jhGARNs...
$2y$07$7IZCeQ90yjRoG15uJVTXeVk.Esrlu0Y.wNHoOZnlpT...
$2y$07$DY7HQ1o26ulXKFTnoFbf0OPyNO8gkNCNhmdGWMDWKgZ..
```

Mediante el cifrado blowfish nos aseguramos de que la contraseña no pueda volver a ser devuelta a los caracteres originales.

Para cifrar las contraseñas utilizamos el siguiente pedazo de código:

```
/**
 * [user_crypt => Encriptación de contraseñas]
 * @param [String] $password [Contraseña a encriptar]
 * @param [integer] $cost [Rondas de encriptacion]
 * @return [String] [Contraseña cifrada]
 */
function userCrypt($password, $cost = 7) {
    $salt = "";
    $salt_chars = array_merge(range('A', 'Z'), range('a', 'z'), range(0, 9));
    for ($i=0; $i<22; $i++) {
        $salt.=$salt_chars[array_rand($salt_chars)];
    }
    return crypt($password, sprintf('$2y$%02d$', $cost) . $salt);
}
```

Con la contraseña sin poder ser descifrada lo único que podemos hacer para verificar es encriptar la contraseña que introducimos al login y compararla con la ya cifrada. Para ello utilizamos el siguiente método:

```
/**
 * [password_check => ¿La contraseña introducida es la misma que la del usuario?]
 * @param [String] $password [Contraseña introducida]
 * @param [String] $encriptado [Contraseña del usuario]
 * @return [Boolean] [1 | 0]
 */
function password_check($password, $encriptado) {
    return (crypt($password, $encriptado) == $encriptado);
}
```

-RESTRICCION DE ENVIO DE CORREOS:

Para evitar que el servidor colapse con envío de correos masivos se ha creado un sistema de seguridad que evita que se puedan enviar correos de recuperación de contraseña si no se cumple determinado espacio de tiempo.

Solo se puede mandar un correo cada 15 minutos.
Último correo mandado a las: 11:49:50



Para crear este sistema comparamos de la base de datos el registro de la última vez que enviamos un correo y la fecha actual.

El código para ello es:

```
/**
 * [compareTimeMail Compara la fecha y hora actual con la de un registro en la tabla sendmail]
 * @param [String] $lastTime [Fecha del registro de un usuario]
 * @return [Boolean] [true || false]
 */
function compareTimeMail($lastTime) {

    $variable = 'false';

    setlocale(LC_TIME, 'spanish');
    $currentDate = utf8_encode( strftime("%Y-%m-%d") );
    $currentHour = utf8_encode( strftime("%r") );
    $databaseTime = explode(" ", $lastTime);

    // Separación del registro de la base de datos
    $hmsDB = explode(":", $databaseTime[1]);

    // Hora, minuto y segundo del registro de la base de datos
    $dbHour = $hmsDB[0];
    $dbMinute = $hmsDB[1];

    // Separación del registro de la fecha actual
    $hmsCurrent = explode(":", $currentHour);

    // Hora, minuto y segundo del registro de la base de datos
    $currentHour = $hmsCurrent[0];
    $currentMinute = $hmsCurrent[1];

    // Si el año, mes y día de la base de datos es mas pequeño que la fecha actual $variable = true
    if ( $databaseTime[0] < $currentDate ) {
        $variable = 'true';
    }

    // Si el año, mes y día de la base de datos es igual que la fecha actual comprobamos la hora
    if ( $databaseTime[0] == $currentDate ) {
        // Si la hora es menor devuelve true, sino comprueba if hayan pasado 15 minutos desde la última petición
        if ( $dbHour < $currentHour ) {
            $variable = 'true';
        } elseif ( $dbHour == $currentHour ) {
            if ( $dbMinute == ( $currentMinute + 15 ) ) {
                $variable = "true";
            }
        }
    }

    return $variable;
}
```

-CONTROL DE SESIÓN DE USUARIO:

Para evitar que cualquier visitante pueda acceder a las páginas privadas se ha creado un sistema que controla que haya un usuario logueado y que exista en la base de datos. Si no se cumple esto se revoca el acceso a la aplicación.

Para crear la sesión del usuario hemos utilizado el siguiente código:

```
session_start();
//DATOS DE LA TABLA DE LOGIN
$_SESSION["idLogin"] = $usuario["idLogin"];
$_SESSION["password"] = $usuario["password"];
$_SESSION["correo"] = $usuario["correo"];
$_SESSION["rol"] = $usuario["rol"];
//DATOS DE LA TABLA DE DATOS
$_SESSION["idDatos"] = $usuario["idDatos"];
$_SESSION["nombre"] = $usuario["nombre"];
$_SESSION["apellido"] = $usuario["apellido"];
$_SESSION["empresa"] = $usuario["empresa"];
$_SESSION["puesto"] = $usuario["puesto"];
$_SESSION["direccion"] = $usuario["direccion"];
$_SESSION["ciudad"] = $usuario["ciudad"];
$_SESSION["provincia"] = $usuario["provincia"];
$_SESSION["pais"] = $usuario["pais"];
$_SESSION["telefono"] = $usuario["telefono"];
$_SESSION["imagen"] = $usuario["imagen"];
```

Antes de poder crear la sesión se comprueba si el usuario realmente existe y si la contraseña que introduce pertenece realmente a esa cuenta.

Una vez logueado el usuario hay que comprobar en cada página que el usuario tiene una sesión. Si no es así se revoca el acceso.

Esto se realiza mediante el siguiente código:

```
k?php
session_start();
include_once('rutas.php');
if ( !isset($_SESSION['correo']) ) {
    header('Location:'.$inicioHtml);
}
?>
```


-CONTROL SOBRE LAS IMÁGENES Y LOS FICHEROS CSV:

Cuando añadimos una imagen a una obra o a un usuario comprobamos que sea realmente una imagen.

En el programa para poder controlar esto reducimos el abanico de extensiones de imágenes a las dos más comunes: PNG y JPG.

Para comprobar que el archivo es realmente una imagen realizamos la siguiente comprobación:

```
if (!empty($_SERVER['HTTP_X_REQUESTED_WITH']) && strtolower($_SERVER['HTTP_X_REQUESTED_WITH']) == 'xmlhttprequest') {  
    if ( $_FILES['addImage']['type'] === 'image/png' ) {  
        $extension = ".png";  
    } elseif ( $_FILES['addImage']['type'] === 'image/jpeg' ) {  
        $extension = ".jpg";  
    } else {  
        $retorno->msg = 'Formato de archivo no válido';  
        echo json_encode($retorno);  
        exit;  
    }  
}
```

De igual manera al importar los datos de una tabla podemos tener problemas con el fichero que se envía.

Para ello el programa controla de igual manera que el archivo que introducimos sea el tipo de archivo que realmente queremos que sea.

Para comprobarlo utilizamos los siguientes códigos:

```
/**  
 * [validarCSV Valida que los archivos para importar sean CSV]  
 * @return {[boolean]} [true || false]  
 */  
function validarCSV() {  
    var archivo = $("#addFile").val();  
    var extensiones = archivo.substring(archivo.lastIndexOf("."));  
  
    if (extensiones != ".csv")  
    {  
        return false;  
    } else {  
        return true;  
    }  
}
```

```
if( $_POST['validate'] === 'true' ) {  
}  
  
elseif ( $_POST['validate'] === 'false' ) {  
    $retorno->msg = 'El fichero tiene que ser un fichero con extensión CSV.';  
}
```

-CONTROL DE DUPLICIDAD DE REGISTROS DE TABLAS:

El programa también controla que ciertas tablas no puedan tener duplicidad de datos.

Las tablas en las que controlamos esto son: clientes, proveedores y usuarios.

Para controlar esta duplicidad comparamos el correo que introducimos con los que ya existen en la base de datos, a no ser que estemos editando el registro que no podemos realizar esta comprobación, ya que también existe el registro que editamos en la base de datos.

Para este último caso la solución que se ha implementado es obviar el registro que estamos editando.

Para realizar la comprobación de usuarios utilizamos el siguiente código:

```
if ( $register->userExist($_POST['correo']) === 'true' ) {
```

```
/**
 * [userExist Comprueba que el correo introducido no se encuentre en la base de datos]
 * @param [String] $correo [correo a introducir]
 * @return [Boolean]      [true || false]
 */
function userExist($correo) {

    $regreso = "true";

    $params = array(':correo' => $correo);

    $sql = "SELECT * FROM `usuariologin` WHERE `correo` = :correo";
    $stmt = $this->pdo->prepare($sql);
    $result = $stmt->execute($params);

    $sql = "SELECT * FROM `usuariodatos` WHERE `correo` = :correo";
    $statement = $this->pdo->prepare($sql);
    $result = $statement->execute($params);

    if( $stmt->rowCount() === 1 ) {
        $regreso = "false";
    } elseif( $statement->rowCount() === 1 ) {
        $regreso = "false";
    }

    return $regreso;
}
```

Para realizar la comprobación de clientes utilizamos el siguiente código:

```
if ( $cruds->buscarCliente($id, $correo) === false ) {
```

```
/**
 * [buscarCliente] Comprueba si existe un cliente en la base de datos]
 * @param [String] $correo [Correo del cliente a buscar]
 * @return [Boolean] [true || false ]
 */
function comprobarCliente($nombre) {
    $params = array(':nombre' => $nombre);
    $sql = "SELECT * FROM `cliente` WHERE `nombre` = :nombre";
    $stmt = $this->pdo->prepare($sql);
    $result = $stmt->execute($params);
    $cuenta = $stmt->rowCount();

    $valor = false;
    if ( $cuenta === 1 ) {
        $valor = true;
    }

    return $valor;
}
```

Para realizar la comprobación de proveedores utilizamos el siguiente código:

```
if( $cruds->buscarProveedor($id, $correo) === false ) {
```

```
/**
 * [buscarProveedor Comprueba si existe un proveedor en la base de datos]
 * @param [String] $correo [Correo del proveedor a buscar]
 * @param [String] $id [id del proveedor a buscar]
 * @return [Boolean] [true || false ]
 */
function buscarProveedor($id, $correo) {
    $valor = false;

    if ( $id === 'null' ) {
        $params = array(':correo' => $correo);
        $sql = "SELECT `id` FROM `proveedor` WHERE `correo` = :correo";
        $stmt = $this->pdo->prepare($sql);
        $result = $stmt->execute($params);
        $cuenta = $stmt->rowCount();

        if ( $cuenta === 1 ) {
            $valor = true;
        }
    } else {
        $params = array(':correo' => $correo);
        $sql = "SELECT `id` FROM `proveedor` WHERE `correo` = :correo";

        $stmt = $this->pdo->prepare($sql);
        $result = $stmt->execute($params);
        $value = $stmt->fetch(PDO::FETCH_ASSOC);
        $ide = htmlspecialchars($value['id'], ENT_QUOTES, 'UTF-8');

        if ( $id === $ide ) {
            $valor = true;
        } else {
            $params = array(':correo' => $correo);
            $sql = "SELECT * FROM `proveedor` WHERE `correo` = :correo";
            $stmt = $this->pdo->prepare($sql);
            $result = $stmt->execute($params);
            $cuenta = $stmt->rowCount();

            if ( $cuenta === 0 ) {
                $valor = true;
            }
        }
    }

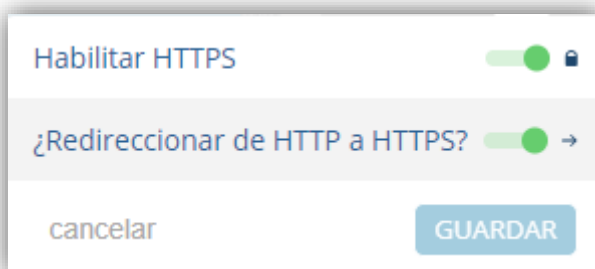
    return $valor;
}
```


-PROTOCOLO SEGURO DE TRANSFERENCIA DE HIPERTEXTO:

El sistema HTTPS utiliza un cifrado basado en SSL/TLS para crear un canal cifrado (cuyo nivel de cifrado depende del servidor remoto y del navegador utilizado por el cliente) más apropiado para el tráfico de información sensible que el protocolo HTTP. De este modo se consigue que la información sensible no pueda ser usada por un atacante que haya conseguido interceptar la transferencia de datos de la conexión, ya que lo único que obtendrá será un flujo de datos cifrados que le resultará imposible de descifrar.

Para poder utilizar el protocolo tendremos que tener un certificado HTTPS.

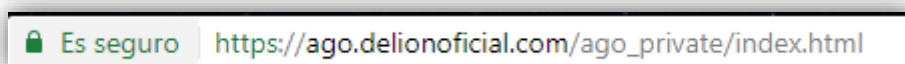
Dentro del servidor instalaremos este certificado para poder activar esta opción.



En el caso del servidor por el que yo he optado la redirección de las páginas de HTTP a HTTPS se realiza automáticamente, pero quise confirmar este suceso añadiendo el siguiente código a las páginas.

```
<link rel="canonical" href="https://ago.delionoficial.com/ago_private/index.html" />
```

Una vez terminado todo el proceso podemos comprobar que efectivamente el cifrado funciona correctamente, protegiendo nuestra página de posibles ataques de *Sniffing*.



El certificado que utilizemos tendremos que renovarlo con el periodo de caducidad que hayamos contratado, ya que si no la url cambiará de verde a rojo y nos mostrará un aviso al acceder a la página.

-PROTECCION CONTRA INYECCION SQL:

Inyección SQL es un método de infiltración de código intruso que se vale de una vulnerabilidad informática presente en una aplicación en el nivel de validación de las entradas para realizar operaciones sobre una base de datos.

Una de las prioridades en la aplicación es la seguridad, así que comenzar por evitar accesos no deseados es primordial.

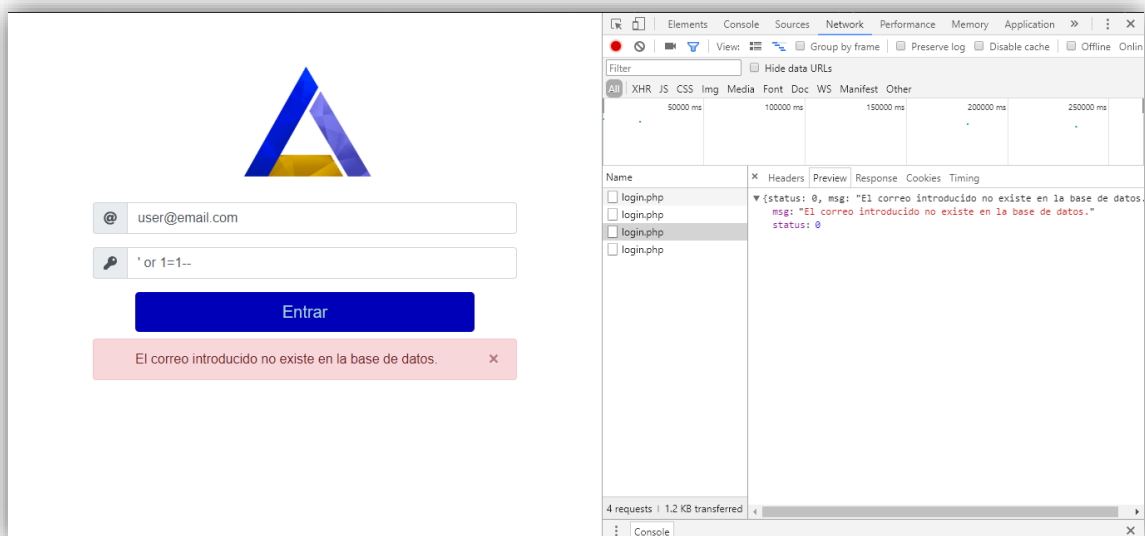
Para comprobar que esta seguridad funciona tendremos que acceder a la página de login e introducir los siguientes datos:

Usuario --> ' or '1'='1

Contraseña --> ' or '1'='1

Si la página no está protegida contra este tipo de ataques o nos permitirá el acceso directamente o nos retornará los datos de acceso.

En el caso de la aplicación este es el resultado:



Para evitar que la página retorne los datos de acceso utilizaremos este código:

```
$correo = htmlspecialchars($value['correo'], ENT_QUOTES, 'UTF-8');  
$password = htmlspecialchars($value['password'], ENT_QUOTES, 'UTF-8');
```

Aquí eliminamos los caracteres extraños que enviamos al servidor, asegurándonos así que no pueden ejecutar llamadas a la base de datos inseguras.

CODIGO RELEVANTE

Aquí varias partes de código que he considerado más importantes, ya que son funcionalidades que no han sido trabajadas y pueden ayudar en el futuro a otra gente.

-ENVIO DE CORREOS:

Para hacer el envío de correos dentro de la aplicación no he utilizado la funcionalidad que nos proporciona PHP, sino que he utilizado un servicio externo.

El servicio concretamente se llama PHPMailer y se pueden bajar los archivos necesarios desde este enlace: [Link](#).

Para utilizar este servicio gratuito solo tenemos que contar con una cuenta de correo electrónico.

Primer importaremos las bibliotecas de los archivos que hemos importado.

```
use PHPMailer\PHPMailer\PHPMailer;
use PHPMailer\PHPMailer\Exception;

require '../libs/PHPMailer-master/src/Exception.php';
require '../libs/PHPMailer-master/src/PHPMailer.php';
require '../libs/PHPMailer-master/src/SMTP.php';
```

Una vez importadas estas bibliotecas crearemos una instancia de la clase del servidor de correo y la modificaremos, utilizando nuestros parámetros:

```
$asunto = 'Recuperación de contraseña';
$cuerpo = 'Este correo es enviado desde la aplicación AGO para la recuperación de la contraseña del correo '.$correo.'. Si usted no es el propietario de la cuenta, por favor ignore este mensaje. Link: '.$paginaCorreo.'?u='.$enlaceCodificado;

$mail = new PHPMailer;           //Se crea una clase mail
$mail->CharSet = "UTF-8";        //Codificación UTF-8 para el correo

// $mail->isSMTP();               // Set mailer to use SMTP
$mail->Host = 'smtp-mail.outlook.com'; // Servidor smtp del cliente del correo
$mail->Username = ;               // Correo que envia el correo
$mail->Password = ;              // Contraseña del correo que envia los correos

$mail->SMTPAuth = true;          // Enable SMTP authentication
$mail->SMTPSecure = 'tls';        // Enable encryption, 'ssl' also accepted
$mail->Port = 587;
$mail->SMTPDebug = 4;

$mail->From = 'helpAgo@hotmail.com'; // Correo que envia los mensajes
$mail->addAddress($correo);        // Esta es la direccion al que se manda el correo

$mail->Subject = $asunto;         // Asunto del correo
$mail->Body = $cuerpo;            // Cuerpo del correo
```

Una vez hecho esto solamente tendremos que llamar al fichero PHP que ejecuta este código y revisar nuestra bandeja de entrada para comprobar que el correo se envía correctamente.

-CLASES DE PHP:

En esta aplicación me ha ayudado mucho segregar las partes, lo que ha sido sencillo creando ficheros de clases. Estos ficheros contienen funciones que llamo desde ficheros externos.

Por lo general las clases las he utilizado para las llamadas a la base de datos, ya que estos ficheros no pueden ser abiertos desde un sitio externo ni se puede visualizar el contenido, lo que aumenta aún más la seguridad de la aplicación.

Para crear una clase lo primero que hay que hacer es crear esta estructura en un fichero PHP:

```
class Busqueda {  
  
    private $pdo = null;  
    private $ruta_app = null;  
    private $ruta_server = null;  
  
    function __construct() {  
        global $ruta_app;  
        global $ruta_server;  
  
        $this->ruta_app = $ruta_app;  
        $this->ruta_server = $ruta_server;  
  
        $fc_inc = true;  
        $ruta = $ruta_server."../inc/connect.php";  
        include $ruta;  
        $fc_inc = false;  
    }  
}
```

De estas variables solo modificaremos \$ruta, que es la variable que nos conecta con la base de datos.

Después podremos añadir nuestras funciones, a las cuales podremos pasar también parámetros y devolverlos mediante un return:

```
/**  
 * [searchUser Busca todos los materiales]  
 * @return [array] [Todos los registros de la tabla material]  
 */  
function searchMateriales($selectProveedor) {  
    $params = array(':defecto' => 1);  
    $sql = "SELECT * FROM `material` WHERE 1=:defecto ";  
  
    if ( !empty($selectProveedor) ) {  
        $params[':distribuidor'] = $selectProveedor;  
        $sql .= "AND `distribuidor` = :distribuidor ";  
    }  
  
    $stmt = $this->pdo->prepare($sql);  
    $result = $stmt->execute($params);  
    $rows = $stmt->fetchAll(PDO::FETCH_ASSOC);  
    $arraymateriales = array();  
  
    foreach ($rows as $key => $value) {  
        $id = htmlspecialchars($value['id'], ENT_QUOTES, 'UTF-8');  
        $nombre = htmlspecialchars($value['nombre'], ENT_QUOTES, 'UTF-8');  
        $descripcion = htmlspecialchars($value['descripcion'], ENT_QUOTES, 'UTF-8');  
        $distribuidor = htmlspecialchars($value['distribuidor'], ENT_QUOTES, 'UTF-8');  
        $preciocompra = htmlspecialchars($value['preciocompra'], ENT_QUOTES, 'UTF-8');  
        $precioventa = htmlspecialchars($value['precioventa'], ENT_QUOTES, 'UTF-8');  
        $stock = htmlspecialchars($value['stock'], ENT_QUOTES, 'UTF-8');  
  
        $p = array('id' => $id, 'nombre' => $nombre, 'descripcion' => $descripcion, 'distribuidor' => $distribuidor, 'preciocompra' => $preciocompra, 'precioventa' => $precioventa, 'stock' => $stock);  
  
        array_push($arraymateriales, $p);  
    }  
  
    return $arraymateriales;  
}
```

Para poder llamar a estas clases desde un fichero PHP primero tendremos que importar el fichero que contiene las clases.

Después tendremos que crear una instancia de la clase para poder llamar a la función que queramos.

Dentro de un fichero de clases podemos crear varias clases.

```
include_once("../inc/session.php");  
include_once("../clases/Login_Register.php");  
include_once("../clases/Busqueda.php");  
  
$cruds = new Cruds;  
$busqueda = new Busqueda;
```

Una vez hecho esto solo tenemos que llamar a las funciones.

Con la variable en la que hemos instanciado una clase realizamos una llamada de la siguiente manera:

```
$cruds->borrarMaterialObras($id);
```

Si queremos pasar valores a la función de la clase tendremos que poner estas variables ordenadas y separadas por comas entre los paréntesis de las llamadas.

Si declaramos esta llamada como valor de una variable y en la función de la clase asignamos un valor return podremos enviar información entre los ficheros PHP mediante POST.

-PDO PARA LA CONEXIÓN A LA BASE DE DATOS:

Para esta aplicación he utilizado PDO.

El PDO es la programación orientada a objetos de PHP.

Aquí dejo un enlace a la página de PHP para conocer esta funcionalidad: [Link](#).

Para poder realizar la conexión a la base de datos mediante PDO hay que crear un fichero de conexión con los siguientes parámetros:

```
<?php
include('rutas.php');
if ($fc_inc) {
    // Determina si el servidor es local o externo
    if ( $server === 'localhost' ) {
        // Parte de local
        $dbuser = "root";
        $dbpass = "";
        $dbdb = "ago";
    } else {
        // Parte de servidor
        $dbuser = "u130711db1"; // Nombre del usuario de la base de datos
        $dbpass = "*****"; // Contraseña de la base de datos
        $dbdb = "u130711db1"; // Nombre de la base de datos
    }
    // PDO
    $this->pdo = new PDO("mysql:host=localhost;dbname=".$dbdb, $dbuser, $dbpass, array(PDO::MYSQL_ATTR_INIT_COMMAND => 'SET NAMES utf8' ));
    $this->pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
}
>>
```

Una vez completado esto creamos la estructura de una clase:

```
class Busqueda {
    private $pdo = null;
    private $ruta_app = null;
    private $ruta_server = null;

    function __construct() {
        global $ruta_app;
        global $ruta_server;

        $this->ruta_app = $ruta_app;
        $this->ruta_server = $ruta_server;

        $fc_inc = true;
        $ruta = $ruta_server."../inc/conect.php";
        include $ruta;
        $fc_inc = false;
    }
}
```

Para poder realizar la llamada a la base de datos utilizamos el siguiente código:

```
$params = array(':nombre' => $nombre, ':telefono' => $telefono, ':direccion' => $direccion, ':correo' => $correo);

$sql = "INSERT INTO `cliente`(`id`, `nombre`, `telefono`, `direccion`, `correo`) VALUES (null, :nombre, :telefono, :direccion, :correo)";
$stmt = $this->pdo->prepare($sql);
$result = $stmt->execute($params);
```

Si no tenemos parámetros tendremos que eliminar la variable \$params incluyendo del execute. En la variable \$sql insertaremos nuestra consulta a la base de datos.

-SUBIDA DE IMÁGENES:

Otra funcionalidad que es bastante útil es la de subir imágenes a una carpeta del servidor.

Esto lo realizaremos mediante una conexión ajax de un formulario con un input de tipo files.

Para realizar la conexión ajax utilizaremos el siguiente código:

```
$( "body" ).on( "click", ".guardarImagen", function( e ) {
    e.preventDefault();
    var btn = $(this);
    var f = btn.parents('.modal-dialog');
    var url = f.find('form').attr('action');

    datos = new FormData($("#formu")[0]);
    datos.append('action', 'guardarImagen');
    datos.append('id', btn.data('id'));

    //hacemos la petición ajax
    $.ajax({
        url: url,
        type: 'POST',
        data: datos,
        cache: false,
        contentType: false,
        processData: false,
        dataType: "json",

        //mientras enviamos el archivo
        beforeSend: function() {
            preloadAlternativo(f);
        },
        //una vez finalizado correctamente
        success: function(data) {

            f.find('.modal-error').empty();
            $('#close').click();
            if (data.status == 1) {
                var href = ruta+"pages/perfilUsuario.php";
                $('#contenido').empty();
                var datos = {};
                var tipo = '#contenido';
                cargar(datos, href, tipo);
                $('#modal-backdrop').remove();
            } else {
                errorModal(data);
            }
        }
    });
});
```

En esta llamada podemos incluir variables para enviar al servidor, así como podremos realizar acciones mientras se realiza la conexión y si la respuesta es satisfactoria o errónea.

En la parte servidor en un fichero PHP utilizaremos el siguiente código:

```
if ( $_POST['action'] == 'guardarMultimedia' ) {
    if ( empty($_FILES) || empty($_POST['obra']) ) {
        $retorno->msg = 'Inserte todos los datos';
    } else {
        if (empty($_SERVER['HTTP_X_REQUESTED_WITH']) && strtolower($_SERVER['HTTP_X_REQUESTED_WITH']) == 'xmlhttprequest' ) {

            if ( $_FILES['addImage']['type'] == 'image/png' ) {
                $extension = ".png";
            } elseif ( $_FILES['addImage']['type'] == 'image/jpeg' ) {
                $extension = ".jpg";
            } else {
                $retorno->msg = 'Formato de archivo no válido';
                echo json_encode($retorno);
                exit;
            }

            if ( $_POST['publicar'] == 'si' ) {
                $publicar = '1';
            } else {
                $publicar = '0';
            }

            $obra = htmlspecialchars($_POST['obra'], ENT_QUOTES, 'UTF-8');
            $opt = $cruds->ingCrypt();

            //obtenemos el archivo a subir
            $file = "Imagen".$opt.$extension;

            //comprobamos si existe un directorio para subir el archivo
            //si no es así, lo creamos
            if (!is_dir("../media/Obras/")) {
                mkdir("../media/Obras/", 0777);
            }

            //comprobamos si el archivo ha subido
            if ($file && move_uploaded_file($_FILES['addImage']['tmp_name'], "../media/Obras/".$file)) {
                //retrasamos la petición 3 segundos
                sleep(3);
            }

            $cruds->saveMultimedia($obra, $file, $publicar);
        }
    }
}
```

Aquí podremos controlar el tipo de fichero que enviamos, el nombre del fichero y si existe en la ruta indicada.

-IMPORTACION DE CSV:

La importación de ficheros CSV se realiza de la misma manera que las imágenes.

La conexión Ajax está separada en dos imágenes, pero el código va seguido:

```
if (t==='ajax') {
    f.prepend( '<div class="progress mb-1" style="height: 4px;"><div clas
    var datos = new FormData(f[0]);
    var validar = validarCSV();
    datos.append("validate", validar);
    f.find( "input[name='addFile']" ).each(function() {
        var ifile = $(this);
        $.each(ifile[0].files, function(i, file) {
            datos.append(f.attr("name"), file);
            datos.append("tipo", "clientes");
            if ( f.find("input[type='checkbox']").is(':checked') ) {
                datos.append("variable", "sobrescribir");
            } else {
                datos.append("variable", "");
            }
        });
    });
};
```

```
$.ajax({
    xhr: function() {
        var xhr = new window.XMLHttpRequest();
        xhr.upload.addEventListener("progress", function(evt) {
            if (evt.lengthComputable) {
                var percentComplete = evt.loaded / evt.total;
                percentComplete = parseInt(percentComplete * 100);
                f.find(".progress-bar").css("width", percentComplete+"%");
            }
        }, false);
        return xhr;
    },
    url: a,
    type: "POST",
    data: datos,
    processData: false,
    contentType: false,
    mimeType: "multipart/form-data",
    dataType: "json",
    success: function(data) {
        if (data.status == 1) {
            window.location.href = window.location;
        } else {
            errorModal(data, 1);
        }
    }
});
```

Para la parte de servidor la conexión se realiza de la siguiente manera:

```
if ($_FILES['addFile']['error'] == UPLOAD_ERR_OK && is_uploaded_file($_FILES['addFile']['tmp_name'])) {
    if( $_POST['validate'] === 'true' ) {
        // GUARDAR CLIENTES
        if ( $_POST['tipo'] === 'clientes' ) {
            if ( $_POST['variable'] === 'sobrescribir' ) {
                $datos->deleteCliente();
            }
            try{
                $handle = fopen($_FILES['addFile']['tmp_name'], "r");
                if ($handle) {
                    while (($buffer = fgets($handle, 4096)) !== false) {
                        $cadena = explode(";", $buffer);
                        for($i = 0; $i < count($cadena); $i++) {
                            $cadena[$i] = trim ($cadena[$i]);
                        }
                        $nombre = utf8_encode($cadena[0]);
                        $telefono = utf8_encode($cadena[1]);
                        $direccion = utf8_encode($cadena[2]);
                        $correo = utf8_encode($cadena[3]);
                        $datos->guardarCliente($nombre, $telefono, $direccion, $correo);
                        $retorno->status = 1;
                        $retorno->msg = 'Fichero importado correctamente.';
                    }
                }
                fclose($handle);
            } catch (Exception $e) {
                $retorno->msg = $e;
            }
        }
    }
}
```


- RUTAS DINAMICAS:

Para poder utilizar rutas que dinámicas en PHP utilizaremos un fichero externo que contiene las rutas que nos proporciona PHP con la variable \$_SERVER.

En este caso he utilizado también rutas dinámicas para el protocolo del servidor.

El fichero se construye de la siguiente manera:

```
k?php
//Servidor
$server = $_SERVER['HTTP_HOST'];

//Url del navegador
$host = $_SERVER['REQUEST_SCHEME']."://".$_SERVER['HTTP_HOST'];

//Directorio para estilos y scripts
$directorio = $_SERVER['DOCUMENT_ROOT']."/ago_private/inc/";

//Carpeta private
$private = $_SERVER['REQUEST_SCHEME']."://".$_SERVER['HTTP_HOST']."/ago_private/";

//Carpeta media
$media = $_SERVER['REQUEST_SCHEME']."://".$_SERVER['HTTP_HOST']."/ago_private/media/";
//Carpeta imagen usuario
$imageUser = $_SERVER['DOCUMENT_ROOT']."/ago_private/media/Usuarios/";

//Ruta datatables
$datatables = $_SERVER['REQUEST_SCHEME']."://".$_SERVER['HTTP_HOST']."/ago_private/action/datatables.php";

//Página de inicio
$inicio = $_SERVER['REQUEST_SCHEME']."://".$_SERVER['HTTP_HOST']."/ago_private/index.php";
$inicioHtml = $_SERVER['REQUEST_SCHEME']."://".$_SERVER['HTTP_HOST']."/ago_private/index.html";

//Propia página
$pagina = $_SERVER['PHP_SELF'];

//Página de contraseña
$paginaCorreo = $_SERVER['REQUEST_SCHEME']."://".$_SERVER['HTTP_HOST']."/ago_private/crearPassword.php";
?>
```

Al igual que hemos hecho con PHP también podemos construir rutas dinámicas en Java Script de la siguiente manera:

```
var protocolo = location.protocol;
var ruta = protocolo+'//'+document.domain+'/ago_private/';
```

-CONTROL DE ACCESO DE USUARIOS:

Una funcionalidad nueva que he añadido al proyecto es la posibilidad de crear permisos de usuarios de manera sencilla.

Para ello simplemente creo en la tabla de usuarios un rol y lo guardo en las variables de sesión para poder acceder a él en todo momento.

Dentro de los ficheros de php simplemente tengo que crear una regla en php que controle lo que quiero que muestre o no.

Un ejemplo de esto es el apartado de facturas:

```
<!-- Facturas -->
<?php if ( $_SESSION['rol'] == 'Administrador' ) :?>
    <li class="nav-item" data-toggle="tooltip" data-placement="right" title="Facturas">
        <a class="nav-link" role="facturas" href="pages/facturas.php">
            <i class="fas fa-file-alt"></i>
            <span class="nav-link-text"> Facturas</span>
        </a>
    </li>
<?php endif; ?>
```

Por supuesto esta función la podemos añadir a cualquier parte de la aplicación, ya sea denegando el permiso a una función o controlando alguna respuesta, por ejemplo.

-MENSAJES DE ERRORES:

Una funcionalidad que he añadido nueva es la opción de visualizar errores con un formato cómodo y totalmente personalizables.

Un claro ejemplo es la página de login:

A screenshot of a login form. It features two input fields: the first for an email address, containing 'a.medina.v.96@hotmail.com', and the second for a password, represented by six dots. Below the password field is a blue button labeled 'Entrar'. At the bottom, a light blue error message box displays 'Contraseña errónea.' with a close icon (X) on the right.

Esto lo controlo en los ficheros PHP de la siguiente manera:

```
$retorno = new stdClass();  
$retorno->status = 0;  
$retorno->msg = 'Error desconocido';
```

```
$retorno->status = 1;  
$retorno->msg = 'Material actualizado correctamente';
```

```
echo json_encode($retorno);
```

El Json que devuelvo simplemente tengo que recogerlos mediante jquery e imprimirlo en una ventana de aviso de bootstrap.

ESTRUCTURA DE FICHEROS

El proyecto se compone de dos partes:

-FRONT-END:

Se compone de la página que se muestra a todo el mundo.

En ella se pueden observar las imágenes que introducimos en la tabla multimedia que estén activas, una descripción y un contacto. También tiene un enlace para acceder a la parte del Back-end.

La estructura de ficheros de esta parte es:

Contiene las siguientes carpetas:

-AGO_PRIVATE --> Es la carpeta del Back-end.

-CSS --> Es la carpeta de los estilos de la Front-Page.

-JS --> Es la carpeta que contiene los scripts de java script y jquery de la Front-page.

-MEDIA --> Es la carpeta que contiene los archivos multimedia de la Front-page.

-PHP --> Contiene los scripts de php que permiten mostrar las imágenes de la tabla multimedia y la parte de contacto.

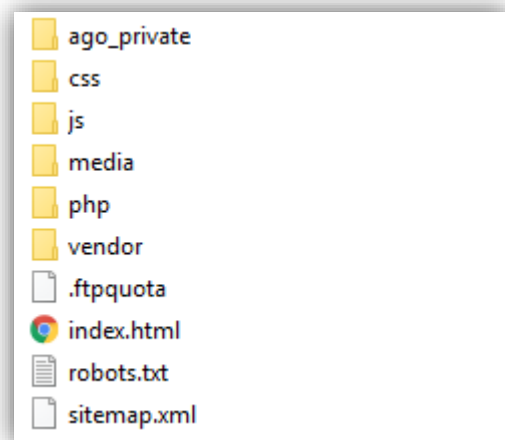
-VENDOR --> Contiene las librerías de la Front-page, como bootstrap o jquery.

Contiene los siguientes archivos:

-INDEX.HTML --> Es la página del Front-page.

-ROBOTS.TXT --> Es el archivo que permite y niega el acceso de los robots a las diferentes partes de la aplicación.

-SITEMAP.XML --> Es el archivo que contiene todos los ficheros de la aplicación.



-BACK-END:

Se compone de las páginas a las que solamente pueden acceder los usuarios registrados.

Solo pueden acceder usuarios registrados ya que está toda la información sensible de la aplicación.

La estructura de ficheros de esta parte es:

Contiene las siguientes carpetas:

-ACTION --> Contiene los ficheros php que realizan acciones.

Por ejemplo guardar, editar, modificar, enviar correos, etc.

-CLASES --> Contiene los ficheros php que tienen acceso a las tablas de la base de datos.

-INC --> Contiene ficheros php que se incluyen en la aplicación. Por ejemplo el footer, la navegación, etc.

-LIBS --> Contiene todas las librerías para que funcione el programa.

-MEDIA --> Contiene imágenes para la interfaz del programa. También alberga las imágenes de usuario y las de multimedia.

-PAGES --> Contiene todas las páginas de la aplicación.

-SCRIPTS --> Contiene los scripts de java script y jquery de la aplicación.

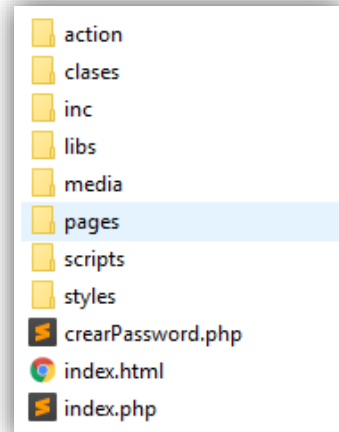
-STYLES --> Contiene los ficheros de estilo de la aplicación.

Contiene los siguientes archivos:

-CREARPASSWORD.PHP --> Es la página para recuperar y crear una contraseña. Solo se puede acceder mediante los enlaces que se envían por correo.

-INDEX.HTML --> Es la página que contiene el formulario de acceso a la parte de Back-end.

-INDEX.PHP --> Es la página principal de la aplicación.



DESPLIEGUE DE LA APLICACIÓN EN HOSTING

Para instalar la aplicación en un servidor lo haremos en varias partes:

-BASE DE DATOS --> La base de datos tiene que tener todas las tablas con los nombres correctos.

-FRONT-END Y BACK-END --> Se pueden instalar por separado o junta.

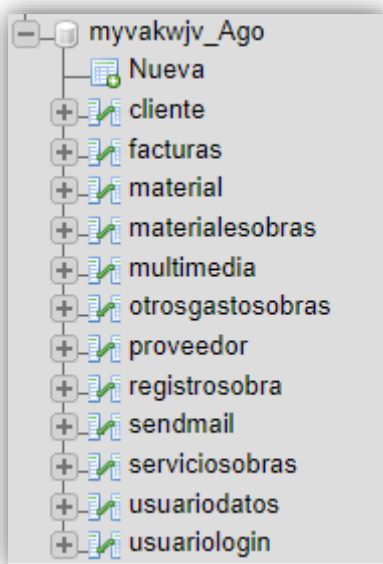
-BASE DE DATOS:

Para instalar la base de datos tenemos que tener el fichero agoDB.sql.

Crearemos en el servidor de phpMyAdmin una base de datos nueva y una vez creado todo importaremos este fichero dentro de la base de datos.

✓ Importación ejecutada exitosamente, 60 consultas ejecutadas. (database.sql)

Las consultas variarán dependiendo de los campos que tengamos introducidos, pero la estructura tiene que quedar así:



Una vez creada esta parte pasaremos a la siguiente, que es añadir los archivos.

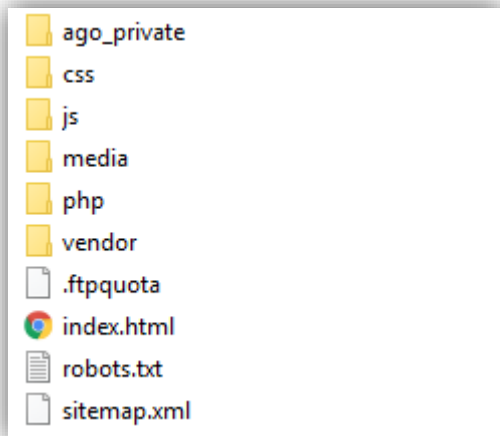
-FRONT-END Y BACK-END:

Para instalar el sistema de ficheros tendremos que disponer mínimo del Back-end.

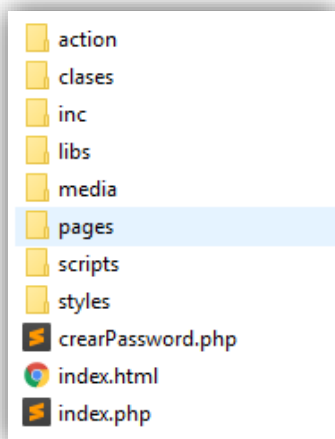
Después tendremos que subir los ficheros a nuestro servidor mediante ftp.

La parte del Back-end siempre tiene que estar dentro de una carpeta que se llame ago_private.

El sistema de ficheros debería de quedar así para el Front-end:



Y así para el Back-end:



Después de instalar el sistema de archivos es hora de editar los ficheros.

-EDICIÓN DE FICHEROS:

La ventaja de esta aplicación es que las rutas están creadas para que varíen dinámicamente con respecto al servidor que lo contenga y si el protocolo de la página es http o https.

Para los archivos PHP el sistema es el siguiente:

```
<?php
//Servidor
$server = $_SERVER['HTTP_HOST'];

//Url del navegador
$host = $_SERVER['REQUEST_SCHEME']."://".$_SERVER['HTTP_HOST'];

//Directorio para estilos y scripts
$directorio = $_SERVER['DOCUMENT_ROOT']."/ago_private/inc/";

//Carpeta private
$private = $_SERVER['REQUEST_SCHEME']."://".$_SERVER['HTTP_HOST']."/ago_private/";

//Carpeta media
$media = $_SERVER['REQUEST_SCHEME']."://".$_SERVER['HTTP_HOST']."/ago_private/media/";
//Carpeta imagen usuario
$imageUser = $_SERVER['DOCUMENT_ROOT']."/ago_private/media/Usuarios/";

//Ruta datatables
$datatables = $_SERVER['REQUEST_SCHEME']."://".$_SERVER['HTTP_HOST']."/ago_private/action/datatables.php";

//Página de inicio
$inicio = $_SERVER['REQUEST_SCHEME']."://".$_SERVER['HTTP_HOST']."/ago_private/index.php";
$inicioHtml = $_SERVER['REQUEST_SCHEME']."://".$_SERVER['HTTP_HOST']."/ago_private/index.html";

//Propia página
$pagina = $_SERVER['PHP_SELF'];

//Página de contraseña
$paginaCorreo = $_SERVER['REQUEST_SCHEME']."://".$_SERVER['HTTP_HOST']."/ago_private/crearPassword.php";
?>
```

Para los ficheros Java Script el sistema es el siguiente:

```
var protocolo = location.protocol;
var ruta = protocolo+'//'+document.domain+'/ago_private/';
```


-EDICIÓN DE ACCESO A BASE DE DATOS:

Para poder vincular la base de datos y el sistema de archivos tenemos que editar solamente un fichero.

Éste se alberga en ago_private/inc/conect.php.

En el fichero hay que editar los parámetros de la parte de servidor.

El código es el siguiente:

```
<?php
include('rutas.php');
if ($fc_inc) {
    // Determina si el servidor es local o externo
    if ( $server == 'localhost' ) {
        // Parte de local
        $dbuser = "root";
        $dbpass = "";
        $dbdb = "ago";
    } else {
        // Parte de servidor
        $dbuser = "u130711db1"; // Nombre del usuario de la base de datos
        $dbpass = "*****"; // Contraseña de la base de datos
        $dbdb = "u130711db1"; // Nombre de la base de datos
    }
    // PDO
    $this->pdo = new PDO("mysql:host=localhost;dbname=".$dbdb, $dbuser, $dbpass, array(PDO::MYSQL_ATTR_INIT_COMMAND => 'SET NAMES utf8') );
    $this->pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
}
?>
```

Una vez terminado esto ya tendremos acceso a la aplicación, y las páginas conectadas a la base de datos.

ESCALABILIDAD

Aumentar la funcionalidad de la aplicación siempre es un problema, pero se ha intentado reducir la dificultad, el coste y el tiempo con la nueva distribución de la aplicación.

Inicialmente se han separado todas las partes de la aplicación, permitiendo así que la incorporación de una nueva funcionalidad sea solamente sobre archivos concretos.

Todos los ficheros tienen la documentación necesaria para identificar la finalidad de las funciones, scripts o secciones de las páginas sin dificultad.

En las tablas de las bases de datos se han eliminado las relaciones para que sea más sencilla la incorporación o eliminación de tablas, siendo estas relaciones controladas dentro del sistema de la aplicación.

Un ejemplo práctico:

Añadir una nueva página con una nueva tabla en la base de datos y funcionalidad tomaría alrededor de un día y medio de trabajo, siempre contando con el diseño propio de la aplicación.

Para modificar el diseño de la aplicación con cambiar el estilo del esqueleto de HTML y CSS bastaría.

También se ha pensado en las siguientes actualizaciones, que pretenderán incorporar funcionalidades importantes que no se han podido añadir en esta versión.

Entre ellas se encuentran:

- Añadir un nivel de seguridad más creando un sistema para evitar la falsificación de sitios cruzados.
- Creación de una auditoría de seguridad, ya que por motivos de tiempo entre el desarrollo y el lanzamiento ha sido imposible crearlo antes.
- Utilización de los datos de la aplicación para la creación de un sistema generador automático de presupuestos, mediante el cual podremos calcular una factura detalladamente antes de que se genere. El sistema creará presupuestos a medida a los clientes, aportando ciertas ventajas a la empresa con respecto a la competencia.

ACCESO

Para acceder a la aplicación he creado un usuario invitado.

Este acceso cuenta con permisos de usuario por lo que no tendrá la funcionalidad al completo.

La cuenta de usuario es:

-NOMBRE --> usuario@usuario.com

-CONTRASEÑA --> accesoInvitado2018

El servidor en el que he alojado la aplicación es un dominio temporal.

Para acceder a la aplicación se puede hacer clic en el siguiente enlace o copiando la dirección en la barra de búsqueda de un navegador web: <https://ago.delionoficial.com/>.

REGISTRO DE CAMBIOS DE VERSION

La aplicación AGO ya está utilizando su versión 0.2.

Con respecto a la antigua versión la nueva trae mejoras tanto visuales como funcionales.

Se han incluido nuevas funcionalidades y se ha mejorado la experiencia de usuario.

Además del programa también ha sido modificada la base de datos, añadiendo nuevas funcionalidades y blindándose con nueva seguridad.

La nueva versión incluye:

-CAMBIOS EN LA INTERFAZ:

- La antigua interfaz era confusa y tosca, por lo que se optó por una renovación visual completa en la que se puede ver un logo rediseñado.
- En un primer momento se optó por utilizar una gama de colores distinta a la actual, pero la aplicación se hacía difícil de visualizar y después de un largo rato se hacía cansada.
- Se han re-ordenado los menús para que sean más accesibles y puedan ser utilizados tanto en un ordenador como en un dispositivo móvil.
- Se han añadido nuevas páginas nuevas. La mayor incorporación ha sido la creación de facturas, aunque esta funcionalidad no está terminada al 100%.
- También se han modificado las páginas de datos, optando por una interfaz de tablas más sencilla y fácil de ver.
- La página de administración de usuario se ha rediseñado completamente, añadiendo funcionalidades nuevas y la opción de poder utilizar una imagen de perfil personalizada.
- Se han eliminado todos los botones que eliminan registros en la base de dato, exceptuando los de servicios, materiales y otros gastos en las obras, ya que comprometían la integridad de la base de datos.
- Se ha añadido la opción de buscar y filtrar en las tablas, haciendo así que se puedan recorrer nuestros datos más sencillamente.
- Se ha rediseñado completamente la página que nos permitía añadir datos en las obras, en la cual se han añadido nuevas opciones.

-CAMBIOS EN LA FUNCIONALIDAD:

- El principal cambio de la funcionalidad es el hecho de que ahora podemos invitar usuarios mediante el registro de usuarios.
- Se ha añadido la opción de recuperación de contraseña y de eliminación de cuentas.
- Se han añadido tecnologías nuevas y más eficientes para aumentar la velocidad y características de la aplicación, y se han renovado las tecnologías antiguas, evitando así posibles fallas de seguridad.
- Se han cifrado las contraseñas de usuario y se han restringido ciertas características que comprometían la seguridad de la antigua aplicación.
- Se han añadido controles de acceso a ciertas partes de la aplicación aumentando así las posibilidades.
- Se ha añadido también la opción para administradores de importar nuestras tablas de la base de datos, haciendo así más rápida y sencilla la opción de añadir datos en la aplicación.
- Se ha modificado la manera de almacenar imágenes en la aplicación, pasando de alojarlas en un servidor externo a alojarlas en el propio servidor, aumentando así la velocidad de carga.

-CAMBIOS EN LA BASE DE DATOS:

- La mayor novedad es la disgregación de la tabla de usuarios en dos complementarias, una de datos del usuario y otra con información sensible, aumentando así la seguridad y evitando un posible robo de datos.
- Se han añadido tablas nuevas a las nuevas opciones de la aplicación. Estas tablas son las de facturas y las de otros gastos de las obras.
- Se ha añadido la tabla de control de envío de correos de recuperación de contraseñas.

CONCLUSION

Esta aplicación es la culminación de dos años de constante aprendizaje y se pueden ver resumidos en una herramienta web totalmente funcional.

Este proyecto supone un punto de partida bastante importante, que seguro que ayudará al desarrollo de futuras aplicaciones.

En este punto solamente queda seguir aumentando la funcionalidad de la aplicación y tratar de llegar a tantos clientes como sea posible.