



Bolsa de Trabajo

15/06/2018

Ricardo Alexis Remache Sánchez

2º Desarrollo de Aplicaciones Web

IES Tierno Galván

Madrid (Madrid)

Introducción

Con el objetivo de ayudar a los estudiantes recién graduados de los ciclos formativos del centro a buscar su primera experiencia laboral y para facilitar la comunicación entre empresas que se ponen en contacto con profesores y tutores buscando alumnos.

Hemos desarrollado una bolsa de trabajo donde poder publicar, gestionar ofertas de trabajo y comunicar empresas con alumnos/as.

Objetivos

1. Ser una fuente de empleo fiable para alumnos/as y exalumnos/as del centro.
2. Facilitar la difusión de ofertas de trabajo que llegan al centro para todos/as los/as alumnos/as registrados/as en la intranet.
3. Automatizar la gestión de ofertas de trabajo publicadas.

Especificaciones

El desarrollo de esta herramienta englobada dentro de la intranet del centro, contiene las siguientes especificaciones:

- Un servidor Apache montado en un sistema ubuntu 18.04.
- PHP: lenguaje principal ejecutado por el servidor.
- HTML5: lenguaje de maquetación para representar las vistas.
- Javascript y jQuery: lenguaje para el cliente encargado de animaciones, validaciones, paginación, manipulación de datos y ajax.
- Bootstrap y css para manipulación del contenido representado, así como de dotar de funcionalidades responsive.
- composer manejador de dependencias y paquetes de php, que nos permitirá el uso de librerías externas.
- librerías: conjunto de funcionalidades de terceros para agilizar el desarrollo de la aplicación.
- json: objeto ligero de intercambio de datos utilizado en las llamadas realizadas por Ajax.
-

Herramientas

Para el desarrollo del proyecto hemos hecho uso de las siguientes herramientas:

- Netbeans y PhpStorm como IDE`s principales de desarrollo del código.
- Github: como repositorio para trabajar con el resto de compañeros encargados de la intranet: <https://github.com/tiernogalvan/tiernogalvan>

Librerías

A continuación haremos un desglose de todas las librerías utilizadas y su función en la aplicación.

Librerías declaradas en Composer:

Las siguientes librerías se autogeneran en la carpeta "vendor" ubicada en la raíz de la aplicación:

[Respect/Validation](#): Librería en php encargada de validar los datos recibidos en servidor.

La bolsa de trabajo consta de varios formularios de entrada de datos, ya sea la creación, actualización de ofertas de trabajo o la edición del perfil de usuario.

Por ello es necesario la comprobación que cada uno de los campos cumplan un estándar mínimo, como el tipo de datos que se espera, si es obligatorio rellenar ciertos campos o el tamaño que tiene que tener.

Ejemplo:

- Campo obligatorio, tipo string y con un mínimo de 10 y un máximo de 200 caracteres:

Diagrama de anotación de validación en PHP:

```
v::attribute(ConstantsBolsaTrabajo::TITULO_OFERTA, v::stringType()->length(10, 200))
```

Las flechas indican:

- Referencia a la librería: `v::`
- Campo del objeto a validar: `ConstantsBolsaTrabajo::TITULO_OFERTA`
- Tipo de dato: `stringType()`
- longitud del campo: `length(10, 200)`

Se agregan todos los campos que se quieran y se le pasará el objeto a validar. Esta función nos devolverá un booleano, con el resultado de la validación:

Referencia a la librería

```
if( $validador->validate($ofertaNueva)){ ...}
```

Objeto a validar

[Carbon](#): Librería en php, diseñada para controlar y manipular los campos tipo fecha.

Para la bolsa de Trabajo tenemos en cuenta las fechas para controlar la creación y borrado de ofertas, así como el tiempo que llevan publicadas o desde hace cuanto, el/la alumno/a, ha modificado sus datos personales.

Ejemplo:

- Queremos recuperar hace cuántos días se creó una oferta de trabajo.

Recibimos la fecha de la base de datos como un string.

La convertimos a una variable tipo datetime.

Y calculamos la diferencia en días con respecto a la fecha actual.

```
public function formatCreacion($creacion)
{
    Referencia a la librería
    $caducidad = Carbon::createFromTimeString($creacion);
    $diferencia = Carbon::now()->diffInDays($caducidad);
    return sprintf(MensajesBT::TIEMPO_TRANSCURRIDO, $diferencia);
}
```

[Latitude](#): Es un constructor de consultas SQL.

Además de su fácil uso, para todo tipo de consultas, hemos optado implementarla, porque está preparada para prevenir todo tipo de inyecciones de SQL.

Todas las consultas a base de datos de la bolsa de trabajo están realizadas con esta librería.

Ejemplo:

SELECT a la tabla de ofertas de trabajo para recuperar una oferta por su ID.

```
$engine = new MySqlConnection();
    $factory = new QueryFactory($engine);
    $query = $factory
        ->select()
        ->from(ConstantsBD::TABLA_OFERTA)
        ->where(field(ConstantsBD::ID_OFERTA)->eq($idOferta))
        ->compile();
```

← Compilador que se utiliza, depende del tipo de base de datos

← Constructor de la librería.

[Faker](#): Librería en php para la generación de datos ficticios.

Mientras la base datos estaba todavía por definirse, esta librería nos permite avanzar con el proyecto generando datos falsos, ya sea representado información de una oferta de trabajo o mostrando el perfil de un/a usuario/a.

Ejemplo:

```
$faker = Faker\Factory::create();

echo $faker->name;
// 'Lucy Cechtelar';
echo $faker->address;
// "426 Jordy Lodge"
```

← Creación de la instancia

← Tipo de dato requerido

[PHP file management](#): Librería en php, diseñada para la manipulación de archivos en el servidor:

En el apartado de los perfiles, cada usuario/a tiene la posibilidad de subir una foto y debido que es necesario borrar la foto antigua cada vez que se sube una nueva, esta librería nos ayuda con la eliminación de estos archivos.

Ejemplo:



```
File::delete($perfilExist->FOTO); //Borro la foto previa
$folderParent = explode("/", $perfilExist->FOTO);
$pos = sizeof($folderParent) - 2;
File::deleteEmptyDir(ConfigBolsaTrabajo::DIRECTORIO_PERFILES . '/' . $folderParent[$pos]); //borro el directorio vacío
```

Como la estructura de almacenamiento de las fotos es:

../{directorio perfiles}/{uuid}/{nombre foto}

El método delete de la clase estática File, solo borra el fichero, dejando el directorio vacío, por ello es necesario la segunda instrucción deleteEmptyDir para borrar la carpeta del fichero eliminado.

[Twig](#): Librería en php para la representación de vistas.

Gracias a la colaboración de Oscar, hemos utilizado este sistema para la utilización de plantillas en php. Al cual se le pasa a través de un singleton la referencia del template y un objeto con datos a representar en la vista.

[PHPMailer](#): Librería en php para el envío de correos.

Dado que nuestro objetivo es conectar alumnos/as y empresas, facilitando la comunicación de ofertas de trabajo. Por ello es necesario el uso de esta librería.

Librerías de Javascript

A Través de npm un manejador de paquetes. Cumple la misma función que composer, pero ahora con librerías en javascript siendo almacenadas en la carpeta "node_modules", de la raíz del proyecto.

[Pagination JS](#): librería en javascript para implementar paginación en la vista principal.

```
<link rel="stylesheet" href="https://pagination.js.org/dist/2.1.2/pagination.css" type="text/css">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

<script src="https://pagination.js.org/dist/2.1.2/pagination.js"></script>
```

Dado que en un posible, la bolsa de trabajo albergará un gran número de ofertas era necesario la implementación de una paginación, además que nuestra intención era implementar dos filtros a la hora de presentar las ofertas. Uno por ciclo de formación y otro por fecha de publicación, que se van actualizando a través de ajax.

Para su uso, con una etiqueta identificadora en html, que recuperamos con jQuery.

Llamamos al constructor de paginación y le pasamos sus parámetros de configuración:

- url a la que hacer la peticiones en ajax
- el número de páginas que queremos y la cantidad de ofertas de cada una.
- los parámetros que se envían con cada llamada de ajax.
- y un callback en el que recibe los datos del servidor y la forma en la que serán representados.

```

    container.pagination({
        dataSource: 'index.php?c=bolsa_trabajo&a=request_operation&operacion=pagination',
        locator: 'items',
        alias: {
            pageNumber: 'page',
            pageSize: 'limit'
        },
        className: 'paginationjs-big',
        totalNumberLocator: function (response) {
            return numOfertas;
        },
        pageSize: 10,
        ajax: {
            beforeSend: function () {
                container.prev().html('Cargando Ofertas de trabajo ...');
            },
            data: {
                orden: dataForm.orden,
                fp_oferta: dataForm.fp_oferta
            },
            callback: function (response, pagination) {
                var dataHtml = '<div class="list-group">';
                container.prev().html(dataHtml);
            }
        }
    });

```

Constructor

Url a la que hacer peticiones Ajax

Clase de css del aspecto que presentará los botones de paginación

Orden de las ofertas (Antiguas, Recientes)

Código de ciclo formativo

Respuesta del servidor

html final que se representa en la página

[Fine Uploader](#): Librería en Javascript encargada de la subida de archivos al servidor. Anteriormente explicamos del uso de una librería para manipular archivos en el servidor. Por su parte, esta librería es la encargada de validar el tipo, la cantidad de ficheros permitidos y la subida a través de ajax de los mismos.

Ejemplo:


```

var uploader = new qq.FineUploader({
  debug: false,
  element: document.getElementById('fine-uploader'),
  request: {
    endpoint:
    'index.php?c=bolsa_trabajo&a=request_operation&operacion=upload_file'
  },
  deleteFile: {
    method: 'POST',
    enabled: true,
    endpoint:
    'index.php?c=bolsa_trabajo&a=request_operation&operacion=upload_file'
  },
  retry: {
    enableAuto: true
  },
  validation: {
    itemLimit: 1,
    allowedExtensions: ["jpg", "png", "jpeg"]
  }
});

```

Constructor

Referencia en la que anclar la librería

Url a la que hacer peticiones de subida Ajax

Url a la que hacer peticiones para borrar por Ajax

Reintentar la subida automáticamente.

Cantidad de ficheros permitidos

Tipo de ficheros permitidos

[PACE](#): Librería en javascript que automáticamente muestra una barra de carga, hasta que se termina de renderizar toda la página.

Dado que tenemos funcionalidades con ajax, de esta manera visual advertimos al usuario que se está llevando a cabo una operación en el servidor.

Y solamente con la inclusión de su librería y un tema, esta se encarga de realizar todas las operaciones.

```

<script src="node_modules/pace-js/pace.min.js"></script>
<link rel="stylesheet" href="node_modules/pace-js/themes/blue/pace-theme-corner-indicator.css"
type="text/css">

```

Librería

Tema

[jQuery](#): librería de javascript, para manipular, de manera más sencilla html.

Usado en las vistas, principalmente recupera la referencia de etiquetas html y para todas las llamadas con ajax al servidor.

```
$.ajax({
    url: "index.php?c=bolsa_trabajo&a=crear_oferta&tarea=insert",
    data: {
        nueva_oferta: JSON.stringify(datos)
    },
    success: function (result) {
        $('button').attr('disabled', false);
        $("#build_modal_response").html(buildCodeModalMessage(JSON.parse(result)));
        $('#request_modal_response').modal('show');
    },
    error: function (XMLHttpRequest, textStatus, errorThrown) {
        $('button').attr('disabled', false);
        $("#build_modal_response").html(buildCodeModalMessageError(
            JSON.parse(XMLHttpRequest.responseText)));
        $('#request_modal_response').modal('show');
    }
});
```

Url a la que realizar la petición

Datos en formato json que se envían

Callback de respuesta


Callback de respuesta cuando, existe un fallo en el servidor

Otras herramientas a mencionar:

[CSS Inliner Tool](#): Herramienta on-line que lee un archivo en html junto a su css y lo junta todo, para que en cada etiqueta de html cargue sus estilos.

Con la intención de enviar ofertas de trabajo atractivas a nivel visual, a través de correo, decidimos enviar el cuerpo del mensaje con HTML, pero gestores de correo como gmail no leen las hojas de estilos o ignoran los estilos declarados en una etiqueta <style>.



Dando como resultado mensajes rotos, como este:

 **IES Tierno Galván - Bolsa de Trabajo** <dawcrud@gmail.com>
para mí ▾

...

 logo

I.E.S. Enrique Tierno Galván

 España  UK

▬

- [Inicio](#)
- [Contabilidad](#)
- [Incidencias TIC](#)
- [Documentos](#)
- [Programaciones](#)
- [Libros 2ª mano](#)
- [Tareas para alumnos](#)
- [Bolsa de Trabajo](#)
- [Crud Users](#)

Datos Personales

Ciclos Formativos:



•

Información

Por ello esta herramienta fue de gran ayuda y ahora se envían correos como estos:

Una persona esta interesada por tu oferta de empleo



De **IES Tierno Galván - Bolsa de Trabajo** 
Destinatario **Erasto Entertainment** 
Fecha **Hoy 16:10**

Nuev@ candidat@ apuntad@

Un@ de nuestr@s alumn@s se acaba de apuntar a tu oferta de Trabajo.

Te adjuntamos una breve descripción de su perfil en la bolsa de trabajo, así como te rogamos un pronto conta

Oferta Publicada: Nueva Oferta de Trabajo test

[Ver Oferta en la Intranet](#)

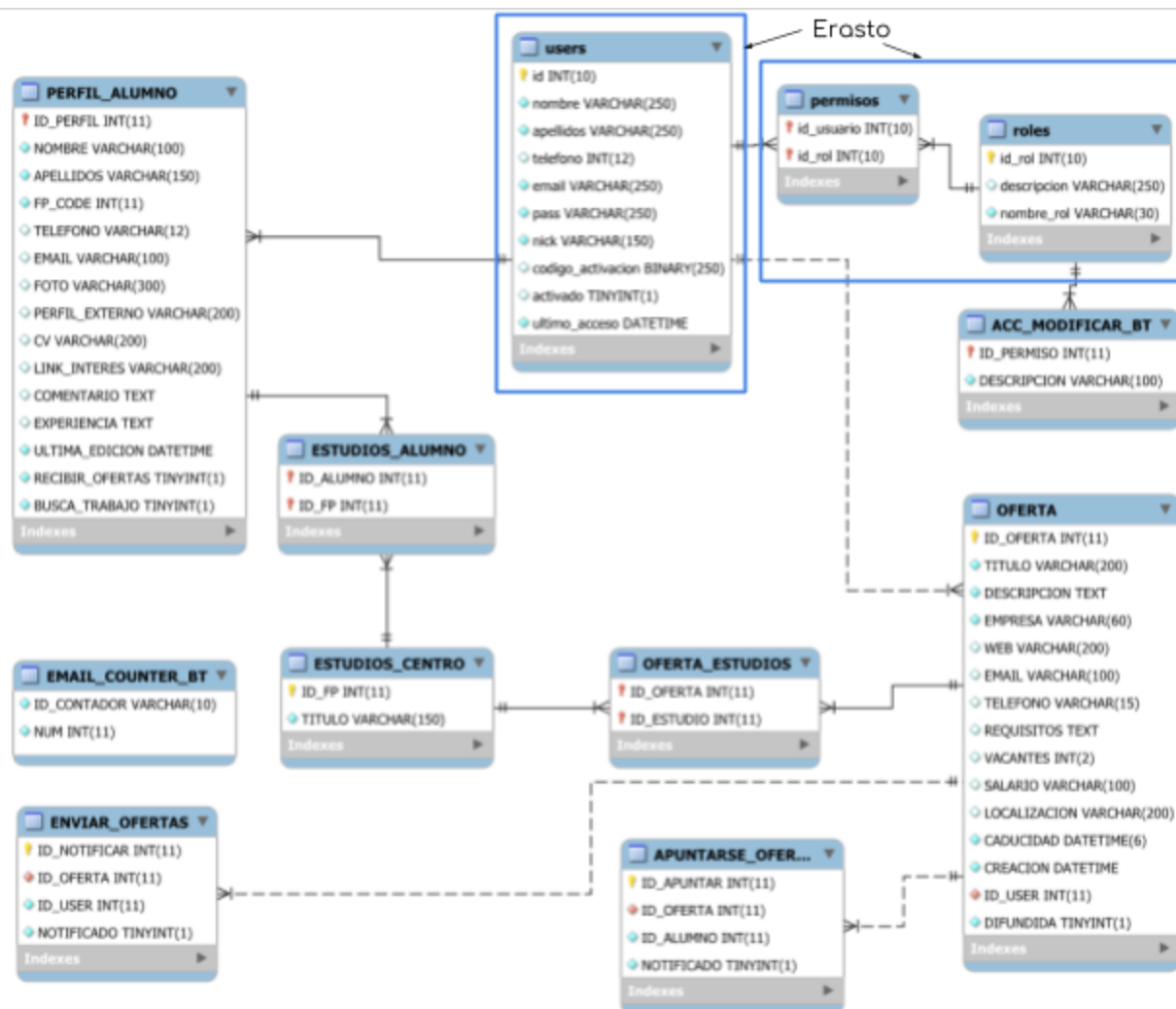
Nombre: Ricardo Remache Sánchez

Descripción:

Información personal

Base de datos

Montado en un servidor mysql, 9 tablas son las que utilizaremos en la bolsa de trabajo, cuya relación es la siguiente:



Descripción por tablas y papel que desempeñan:

Las tablas más importantes son PERFIL_ALUMNO y OFERTA.

OFERTA:

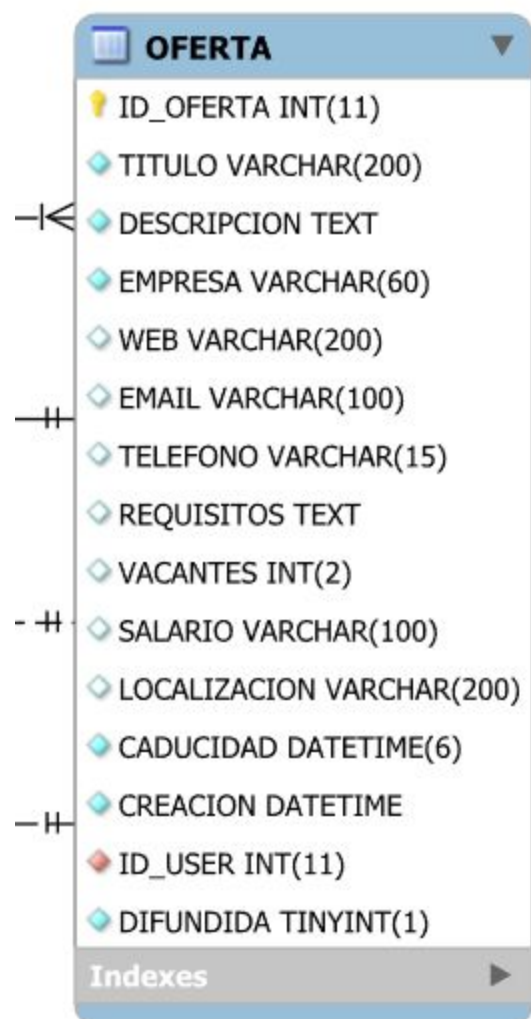


Tabla que albergará todas las ofertas de trabajo publicadas en la bolsa.

Esta además de contener los campos necesarios como título, descripción o requisitos.

Las campos EMAIL, CADUCIDAD, CREACION, ID_USER y DIFUNDIDA son importantes.

EMAIL: Campo necesario para que el/la anunciante reciba correos, con la gente que se va apuntando en la oferta.

CADUCIDAD y CREACIÓN: Son campos necesarios para llevar un mantenimiento de la bolsa de trabajo, borrando aquellas ofertas antiguas.

ID_USER: nos sirve de referencia para identificar al usuario que ha creado una oferta y para que ella la pueda gestionar.

DIFUNDIDA: nos sirve para identificar si la oferta de trabajo ha sido enviada a todos los usuarios de la intranet cuyo ciclo concuerde con esta.

OFERTA_ESTUDIOS:



Tabla en la que almacenamos para que ciclos formativos va destinada la oferta.

PERFIL_ALUMNO:

Column	Type	Primary Key
ID_PERFIL	INT(11)	Yes
NOMBRE	VARCHAR(100)	No
APELLIDOS	VARCHAR(150)	No
FP_CODE	INT(11)	No
TELEFONO	VARCHAR(12)	No
EMAIL	VARCHAR(100)	No
FOTO	VARCHAR(300)	No
PERFIL_EXTERNO	VARCHAR(200)	No
CV	VARCHAR(200)	No
LINK_INTERES	VARCHAR(200)	No
COMENTARIO	TEXT	No
EXPERIENCIA	TEXT	No
ULTIMA_EDICION	DATETIME	No
RECIBIR_OFERTAS	TINYINT(1)	No
BUSCA_TRABAJO	TINYINT(1)	No

En esta tabla almacenaremos toda la información relevante que el/la usuario/a quiere que sean públicas para la bolsa.

A destacar: EMAIL, FOTO, ULTIMA_EDICION, RECIBIR_OFERTAS.

EMAIL: Campo que utilizaremos para enviarle ofertas de trabajo que concuerden con su ciclo formativo y para recibir confirmación de apuntarse a una oferta.

FOTO: almacena la dirección del directorio donde se almacena la foto de perfil.

ULTIMA_EDICION: Campo para calcular el tiempo que lleva sin actualizarse un perfil.

RECIBIR_OFERTAS: Campo que revisamos cada vez que tratamos de enviar una nueva oferta al usuario y proceder según indique el booleano.

ESTUDIOS_ALUMNO

Column	Type	Primary Key
ID_ALUMNO	INT(11)	Yes
ID_FP	INT(11)	No

Tabla en la que almacenamos los ciclos formativos que un alumno ha obtenido en el centro.

ESTUDIOS_CENTRO



Tabla en la que almacenamos todos los ciclos formativos que dispone el centro de estudios.

APUNTARSE_OFERTA



Tabla en la que vamos agregando a los alumnos que se van apuntando a una oferta de trabajo y con ello evitar un envío duplicado de correos a alumnos/as y empresas.

ACC_MODIFICAR_BT (Acceso modificar bolsa de trabajo)

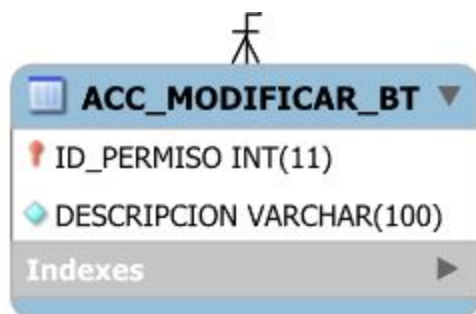


Tabla relacionada con la tabla de roles en la que definimos que tipo de usuarios pueden acceder a la bolsa de trabajo con permisos para crear ofertas de trabajo.

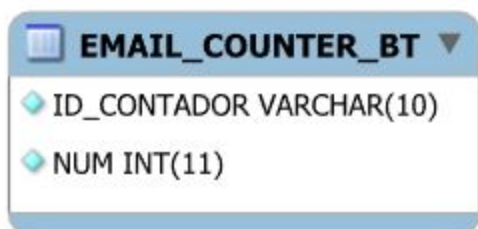
Actualmente se permite a Administradores, Profesores y Empresas.

ENVIAR_OFERTAS



Tabla en la que almacenamos que oferta y a que usuario vamos a enviar un correo cuando se crea una nueva oferta de trabajo.

EMAIL_COUNTER_BT



Actualmente como está montado el servicio de correos con una cuenta de Gmail, esta tabla sirve para llevar un control de los correos enviados, para no superar los [500 emails diarios que permite Gmail](#), antes de bloquear temporal una cuenta.

Se adjunta un script con la base de datos del proyecto.

Estructura del proyecto

Basándonos en el patrón de diseño de MVC (Modelo vista controlador), la bolsa de trabajo se divide en cuatro capas.

La capa de DAO, encargada de todas las operaciones realizadas con la base de datos.

La capa de Servicios encargada del grueso de las operaciones que realiza la aplicación, validaciones, manipulación de datos y archivos.

La capa del controlador, que se encarga de controlar las peticiones entrantes y repartir hacia la operación requerida.

Y la vista a la cual enviamos la información a representar, ya sea través de twig o por medio de ajax.

Distribución del código.

Todo lo que concierne a la bolsa de trabajo se ubica en las siguientes carpetas del proyecto.

El núcleo de la aplicación se ubica dentro de /src.

- /src/config - Tenemos un archivo de configuración que explicaremos en el apartado de configuración necesaria para el funcionamiento de la misma.
- /src/controllers - Desde el index, cualquier petición que tiene que ver con la bolsa de trabajo será enviada aquí.
- /src/dao/bolsaTrabajo - La clase destinada a conectarse con la base de datos.
- /src/model - Directorio donde almacenamos todos los modelos que utilizaremos, como OfertaTrabajo o PerfilBolsaTrabajo.
- /src/servicios - Aquí se define la capa de servicios
- /src/utils/bolsaTrabajo - Almacena un conjunto de clase para distintos propósitos.
 - BuzonCorreo - es un singleton configurado para funcionar con una cuenta de gmail, codificando el contenido del mensaje en UTF-8, permitiendo el uso del envío de html en el mensaje, además de estar configurado para funcionar con dos cuentas de correo distintas, si una de ellas no puede enviar el mensaje, la otra tratará de enviar el correo.
 - Tres clases de constantes. para distintos propósitos.
 - ConstantesDB: contiene el nombre de todas las tabla y campos de la bolsa de trabajo. Es utilizada por la capa de DAO en sus consultas.
 - ConstantesBolsaTrabajo: Alberga todas las constantes usadas por el controller y la vista.
 - MensajesBT: Almacena todos los mensajes informativos que el usuario recibe de la aplicación.
 - UploadHandler: Clase externa encarga de controlar la subida de fotos.
 - GenEmail: Es una clase idéntica a TwigViewer, que en vez de pintar las vistas de la aplicación, recupera el contenido de una plantilla para ser enviado por correo.

/vista/bolsaTrabajo - La parte referente a la vista. Directorio que contiene todas las plantillas de twig utilizadas.

/css/bolsaTrabajo - pequeño conjunto de estilos utilizados para mejorar el aspecto visual de la aplicación.

/img/bolsaTrabajo/perfiles - directorio definido en el archivo de configuración, donde se cargan todas las fotos de los distintos usuarios.

/js/bolsaTrabajo - Directorio con un conjunto de archivos de javascript para todas las operaciones realizadas desde el cliente, recuperar datos, ventanas modales, animaciones y ajax.

/node_modules/ - Directorio donde se almacenan todas las librerías en Javascript

/vendor/ - Directorio donde se almacenan todas las librerías en php.

/cron/bolsaTrabajo: directorio que contiene los scripts a ejecutarse periódicamente por crontab.

Configuraciones:

A la hora de poner en marcha nuestra aplicación hay que tener en cuenta un par de configuraciones para un perfecto funcionamiento del servicio.

Subida de fotos: para que esta característica funcione de manera adecuada, tendremos que otorgar los permisos para manipular archivos:

Situados en la carpeta raíz del proyecto.

```
sudo chmod 766 -R /img/bolsaTrabajo
```

La aplicación hace uso de varios scripts a ejecutar cada cierto tiempo, los cuales se explicaran en detalle en el siguiente apartado. Por ello haremos uso de la herramienta del sistema crontab.

Accedemos al servicio:

```
crontab -e
```

Dentro del archivo que se nos abrirá, añadimos los comandos pertinentes.

```
0 */6 * * * /usr/bin/php7.1
/var/www/html/cron/bolsaTrabajo/poblarEnviarCorreos.php

0 * * * * /usr/bin/php7.1
/var/www/html/cron/bolsaTrabajo/enviarCorreosMasivamente.php

0 0 * * * /usr/bin/php7.1
/var/www/html/cron/bolsaTrabajo/borrarOfertasAntiguas.php
```

Hay que tener en cuenta donde esta alojado el interprete de php, para ello podemos consultarlo de la siguiente manera:



```
whereis php
```

Además uno tiene que estar seguro cual es el directorio absoluto de los scripts.

De esta manera el proyecto está listo para su funcionamiento

CRON

Es un administrador de procesos en segundo plano para ejecutar estos cada cierto tiempo.

Será necesario para llevar a cabo tareas masivas y de mantenimiento de la aplicación.

Detalle de los scripts utilizados.

Poblar Enviar Correos

Script encargado de revisar todas las nuevas ofertas de trabajo, recuperar a qué ciclos formativos va dirigida, buscar los/as alumnos/as registrados/as en este ciclo formativo, para llevarlos a la tabla ENVIAR_OFERTAS donde el siguiente script se encargará del siguiente paso. Finalmente la oferta tratada se modificará para cambiar su estado y no ser tomada en cuenta para el próxima ejecución del script.

Configuración:

- Repetición: cada 6 horas.
- Patrón: 0 */6 * * *
- Comando (intérprete php): /usr/bin/php
/{directorio_host}/cron/bolsaTrabajo/poblarEnviarCorreos.php
- Comando (wget) : wget {hosting}/cron/bolsaTrabajo/poblarEnviarCorreos.php

Enviar Correos Masivamente

Script encargado de recorrer la tabla ENVIAR_OFERTAS buscando usuarios/as a los/as cuales enviar un correo, con una oferta de trabajo acorde con su ciclo formativo.

Se recupera la información relevante de la oferta y se envía por correo al interesado/a.

Configuración:

- Repetición: cada hora.
- Patrón: 0 * * * *
- Comando (intérprete php): /usr/bin/php /{directorio
host}/cron/bolsaTrabajo/enviarCorreosMasivamente.php

- Comando (wget) : wget
{hosting}/cron/bolsaTrabajo/enviarCorreosMasivamente.php

Borrar Ofertas Antiguas

Script encargado de revisar las ofertas de trabajo publicadas y borrar aquellas que han caducado en la tabla de OFERTA.

Configuración:

- Repetición: una vez por día
- Patrón: 0 0 * * *
- Comando (intérprete php): /usr/bin/php /{directorio host}/cron/bolsaTrabajo/borrarOfertasAntiguas.php
- Comando (wget) : wget {hosting}/cron/bolsaTrabajo/borrarOfertasAntiguas.php

Parámetros configurables de la aplicación:

En el directorio /src/config, el archivo configBolsaTrabajo.php tiene una serie de parámetros personalizables:

- **LONGITUD_TEXTO_DESCRIPCION:** define el número de caracteres a mostrar en la vista principal por cada una de las ofertas de trabajo enlistadas.
- **DIRECTORIO_PERFILES:** marca la ruta donde el directorio almacena las fotos que los usuarios suben al servidor.
- **LIMITE_CORREOS_POR_DIA:** Dado que enviamos correos de manera masiva bajo ciertas circunstancias, este parámetro limita el número máximo de correos que pueda enviar por día.
- **LIMITE_CORREOS_POR_HORA:** Es otro paramentro limitante para enviar correos masivamente limitando su máximo por hora.
- **Servidor Email:** Actualmente está configurado con una cuenta de Gmail y recomendamos tener un email exclusivo, ya que en un supuesto con muchos usuarios registrados, por cada nueva oferta registrada la cantidad de correos enviados puede a ser alto.

Funcionamiento:

La aplicación tiene en cuenta dos perfiles de usuarios principalmente, alumnos/as y empresas, con funcionalidades comunes y específicas para cada uno.

De manera global, todo el contenido de la bolsa de trabajo es de acceso exclusivo para personas registradas en la intranet.

Permisos:

Alumnos/as: pueden ver todas las ofertas de trabajo publicadas en la bolsa.

Ver y editar su perfil personal de acceso público a todo usuario de la intranet.

Empresas/Administradores: pueden ver las ofertas publicadas.

Crear, Editar y Borrar ofertas de trabajo.

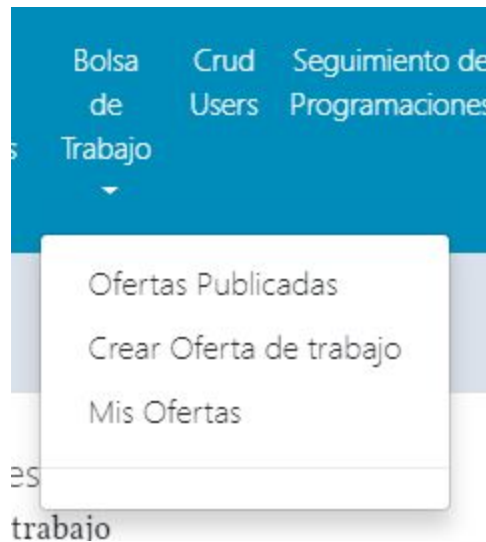
Vista principal para todos los usuarios:

The screenshot displays the main interface of the job application system. On the left, there is a sidebar titled "Opciones de Filtración" (Filtering Options) under the heading "Ciclo Formativo" (Training Cycle). It lists several categories with radio buttons: "Todas las Ofertas" (All Offers), "EEC - Equipos Electrónicos de Consumo" (Consumer Electronic Equipment), "IEA - Instalaciones Eléctricas y Automáticas" (Automatic Electrical Installations), "IPC - Instalaciones de Producción de Calor" (Heat Production Installations), "IFC - Instalaciones Frigoríficas y de climatización" (Refrigeration and Air Conditioning Installations), and "IPCFC - Instalaciones de Producción de..." (Heat Production Installations...). The main area is titled "Ofertas de trabajo" (Job Offers) and shows a list of job postings. The first posting is "Nueva Oferta de Trabajo test" (New Test Job Offer) with a description "Descripción de la oferta Mucho trabajo" (Job description: A lot of work) and a timestamp "Hace 1 días" (1 day ago). The second posting is "oferta de prueba" (test offer) with a description "descripción mucho trabajo" (description: a lot of work) and a timestamp "Hace 3 días" (3 days ago). The third posting is "VEN HOY, EMPIEZA MAÑANA" (Come today, start tomorrow) for "Captación socios Cruz Roja jhee" (Cruz Roja partner recruitment) with a timestamp "Hace 4 días" (4 days ago). The description for this posting reads: "Gracias a nuestra dilatada trayectoria profesional de más de 10 años desarrollamos campañas de ventas y captación de clientes face to face tanto..." (Thanks to our long professional trajectory of more than 10 years, we develop sales and customer recruitment campaigns face to face both...).

Simulación de funcionalidad por tipo de Usuario.

Empresa:

Vista global del menú.



Crear Oferta de Trabajo:

Muestra un formulario donde la empresa anunciante rellena toda la información necesaria para crear una nueva oferta.

Creación Oferta de trabajo

Rellena los siguientes campos, para crear una nueva oferta de trabajo

Título de la oferta *

Describe en pocas palabras, lo que estas ofreciendo

Descripción *

Cuentanos en que consiste el trabajo

Normas para rellenar el formulario

Solo se podrá crear una nueva oferta siempre y cuando se cumplan los siguientes requisitos:

- Todos los campos con (*) son obligatorios
- El título tiene que mínimo de 5 y un máximo de 200 letras.
- La Descripción tiene un min. de 10 y un max. de 1000 letras.
- Los Requisitos tienen un min. de 10 y un max. de 750 letras.
- El Email tienen un min. de 10 y un max. de 80 letras.

Tras rellenar todos los campos según se indican en las normas a cumplir, Guardamos y una petición en ajax es enviada al servidor que se encargará de validar y responder según proceda. Si todo es correcto, nos devolverá un mensaje modal avisandonos de la resolución.

The image shows a web form for creating a job offer. A modal window is displayed in the center with the title "Operación Aceptada" and the message "La inserción de sus datos ha sido aceptada". At the bottom of the modal are two buttons: "Entendido" (green) and "Ver Oferta Creada" (blue). The background form is partially visible, showing fields for "Título de la oferta", "Descripción", and "Requisitos".

Del mismo modo, si surge algún problema, el usuario será informado de la misma manera.

Mis ofertas:

Muestra una lista de oferta de trabajo publicadas por el usuario, las cuales podrá editar su contenido y llegado el caso borrar la oferta.

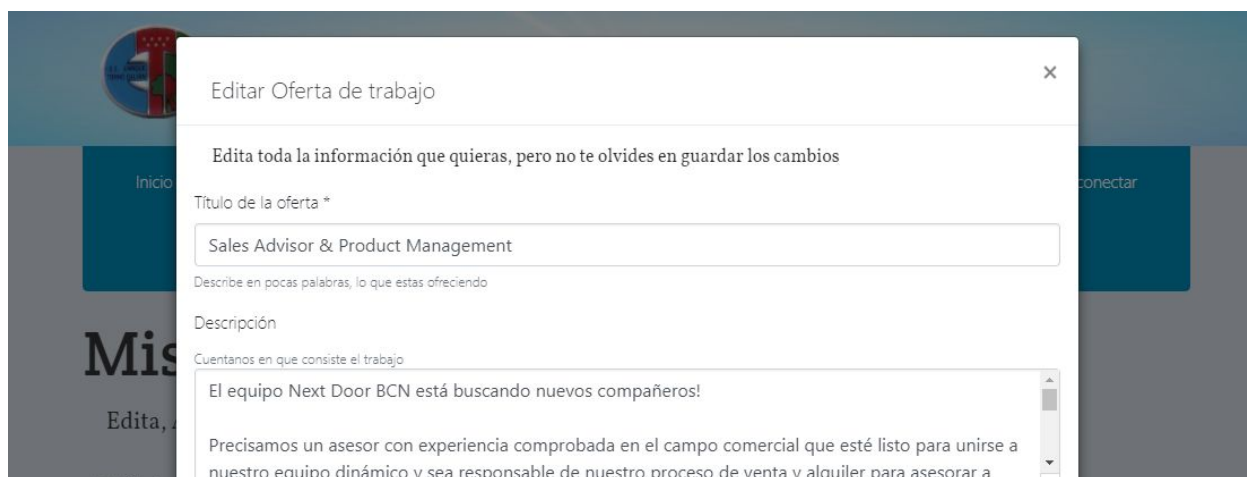
Mis Ofertas

Edita, Actualiza y Borra tus publicaciones

ID Oferta	Título	Acciones
84	Nueva Oferta de Trabajo test	<button>Editar</button> <button>Borrar</button>
85	Sales Advisor & Product Management	<button>Editar</button> <button>Borrar</button>

Editar y borrar realizarán una llamada en ajax.

Editar recuperará la oferta de trabajo y en una pantalla modal desplegará toda la información de la oferta.



Editar Oferta de trabajo

Edita toda la información que quieras, pero no te olvides en guardar los cambios

Título de la oferta *

Sales Advisor & Product Management

Describe en pocas palabras, lo que estas ofreciendo

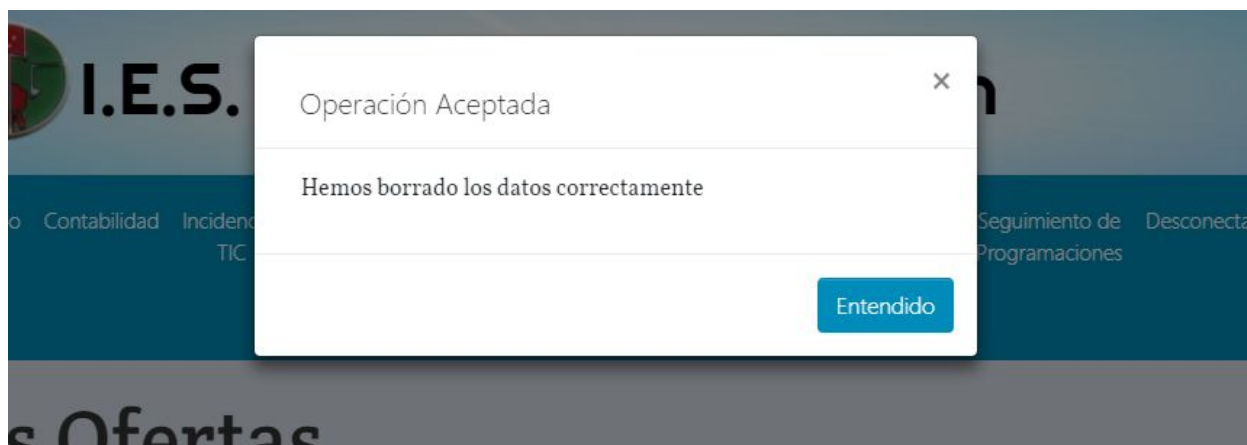
Descripción

Cuentanos en que consiste el trabajo

El equipo Next Door BCN está buscando nuevos compañeros!

Precisamos un asesor con experiencia comprobada en el campo comercial que esté listo para unirse a nuestro equipo dinámico y sea responsable de nuestro proceso de venta y alquiler para asesorar a

Borrar simplemente informará del resultado.



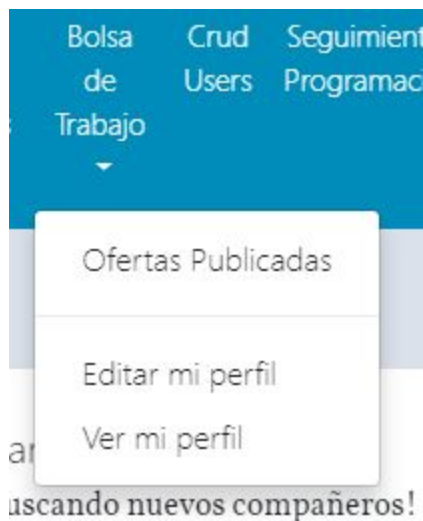
Operación Aceptada

Hemos borrado los datos correctamente

Entendido

Alumno/a:

Vista global del menú.



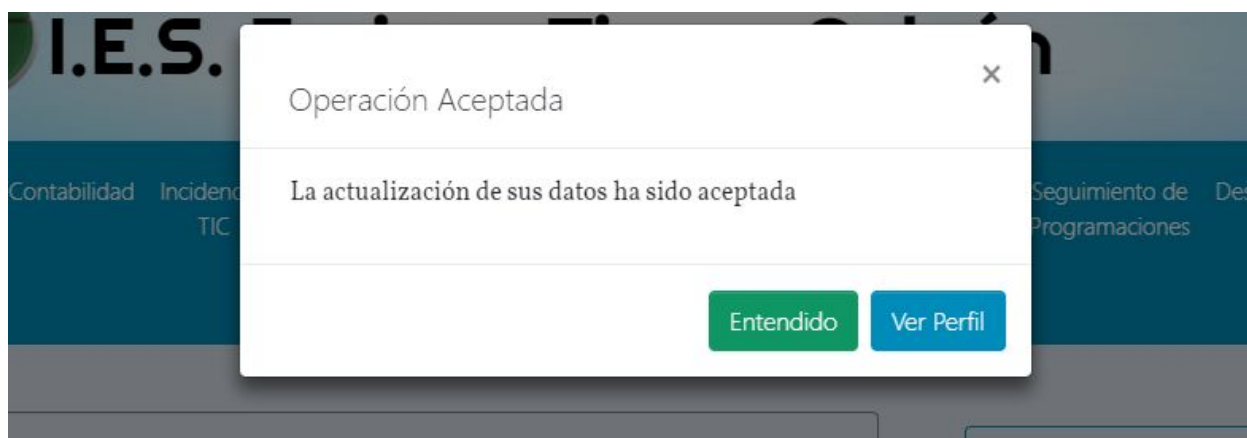
Editar mi perfil:

Muestra un formulario a rellenar con todos los datos más importantes que sirvan de carta de presentación a las empresas.

A screenshot of a web application's 'Editar Perfil' (Edit Profile) form. The form is divided into two main sections. The left section, titled 'Editar Perfil', contains a 'Foto de Perfil' (Profile Photo) area with a 'Sube una foto' (Upload a photo) button and a placeholder text 'Suelta una foto aquí' (Drop a photo here). Below this are input fields for 'Nombre' (Name) and 'Apellidos' (Surnames), each with a placeholder 'Tu nombre, por favor' (Your name, please) and 'Tus Apellidos, por favor' (Your surnames, please) respectively. At the bottom of this section is a list of 'Ciclos Formativos Cursados' (Completed Training Cycles) with two items: 'EEC - Equipos Electrónicos de Consumo' and 'IEA - Instalaciones Eléctricas y Automáticas'. The right section, titled 'Foto Actual' (Current Photo), shows a large blue outline of a person's head and shoulders. Below this is the heading 'Configuraciones adicionales' (Additional Configurations). Under this heading are two radio button questions: 'Estás buscando trabajo?' (Are you looking for work?) with options 'Sí' (Yes) and 'No' (selected), and 'Quieres recibir ofertas de trabajo por E-mail?' (Do you want to receive job offers by email?) with options 'Sí' (Yes) and 'No' (selected).

Además de ello ofrece una pequeña configuración para el usuario, que le permite dejar de recibir nuevas oferta de trabajo si este lo prefiere.

Una vez se haya rellenado todos los campos y guardado el perfil, se realizará una llamada por ajax que se encargará de crear un nuevo registro en la tabla de perfiles o actualizar el perfil existente.



Apuntarme:

Es una funcionalidad exclusiva de los alumnos que les sirve para conectar con las empresas.

Full Stack Developer

Descripción

iEstrategic es una agencia de marketing digital y fabricante de una solución SaaS en crecimiento, y por ello necesitamos incorporar nuevo talento a nuestro equipo.

Como Full Stack Developer en iEstrategic contribuirás al éxito de proyectos en diferentes sectores (turismo, retail, etc) trabajando en un equipo multidisciplinar, joven e internacional.

Participarás en el desarrollo de webs y aplicaciones desde el principio hasta el fin, sobre una potente plataforma SaaS. Trabajarás codo con codo con expertos en SEO, SEM, UX/UI, y marketing digital, aportando tu conocimiento y experiencia en tecnologías como HTML5, CSS3, Javascript y PHP/MySQL.

Estas Interesad@ en esta Oferta de Trabajo?

No esperes más y ponte en contacto con la persona responsable, otr@ lo pudo haber hecho ya.

APÚNTAME

En cada una de las ofertas publicadas, aparecerá el botón apuntame. Realizando una petición ajax, se encarga de registrar al alumno en una oferta de trabajo y envía un correo a la empresa con información del usuario apuntado y un enlace a la intranet donde poder observar su perfil, mientras al alumno se envía un correo de confirmación con información relevante de la oferta, además de incentivar a las dos partes por una pronta comunicación para una entrevista de trabajo.



Ver mi perfil:

Es una vista donde se representarán todos los datos que el usuario quiere mostrar en la intranet.

Datos Personales

Ricardo Remache Sánchez

Ciclo Formativo:

DAW - Desarrollo de Aplicaciones Web

Información


Técnico Superior en Desarrollo de Aplicaciones multiplataforma,
Técnico Superior en Desarrollo de Aplicaciones Web.

Lenguajes de programación:
-Java, PHP, Javascript, python, ABap

Base de datos:
- Oracle y MySql

Otros:
-Android, SAP UI5

Experiencia



Situación Laboral

Trabajo:
[En búsqueda activa de empleo](#)

Ofertas de Trabajo:
[Estoy dispuesto a recibir ofertas de Trabajo](#)

Perfil Actualizado:
— Hace 1 días

Y como hemos dicho antes, la empresa anunciante recibirá un enlace a este perfil.

Mejoras:

- Permitir agregar, más de un ciclo formativo a los alumnos del centro. La tabla ESTUDIOS_ALUMNO está preparada para ello.
- En la vista de Ver mis ofertas, se requiere de una paginación, aunque no es esencial, no se espera que una empresa publique activamente más de 10 ofertas que lo haga necesaria por ahora.
- Subir CVs en PDF.
- Mejorar la representación de fechas las ofertas trabajo.
- Ocultar el botón de apuntar para las empresas y administradores en la vista de una oferta.
- La comunicación entre el usuario y la empresa.
- Un listado con las personas apuntadas a una oferta de trabajo.
- Borrado de fotos residuales del servidor.
- Implementar hilos para tareas muy pesadas o hacerlas de otra manera.
- Validación por jQuery o javascript de los datos en cliente.
- Mejorar la usabilidad
- Ser explícito en qué campos se ha equivocado el usuario, en los formularios.
- Buscador para las ofertas de trabajo.
- Pedir confirmación cuando vas a realizar tareas de borrado.
- Habilitar el borrado de perfiles.

Agradecimientos:

A todo el personal docente del ciclo de DAW y en especial a Santiago por ser un profesor/tutor muy implicado con los alumnos a los que nos tuvo mucha paciencia en muchos momentos y Oscar por enseñarnos tecnologías y metodologías de programación que nos han servido de mucho en las prácticas de empresa.

A nuestros compañeros del grupo encargado de la intranet, por a pesar de muchos riñas internas, se ha intentado por todos los medios posibles ser lo más colaborativos posibles.

Bibliografía:

Respect/Validation - validador de datos - <https://github.com/Respect/Validation>

Faker - Datos ficticios para la aplicación <https://github.com/fzaninotto/Faker#installation>

Carbon - Manipulación de Fechas <https://github.com/briannesbitt/Carbon>

latitude - crear consulta base de datos y previene SQL injection -
<https://shadowhand.github.io/latitude/>

pagination JS - <http://pagination.js.org/index.html>

fine uploader - <https://docs.fineuploader.com>

php file management - gestor de ficheros en el servidor
<https://github.com/Josantonius/PHP-File>

PHP FileUpload <https://github.com/Gargron/fileupload>

CSS Inliner Tool - <https://templates.mailchimp.com/resources/inline-css/>

PACE - <http://github.hubspot.com/pace/docs/welcome/>