

AsoGest
Gestor de Asociaciones

Instrucciones de instalación
Versión 1.4-Beta

Alejandro Campos (acamfue@gmail.com)

Índice

1. ¿Qué es AsoGest y para qué sirve?	2
2. Instalación en ordenador Local	3
3. Configuración básica de la aplicación	4
3.1 Creación de la Base de Datos	4
3.2 Configuración de Google Calendar para la gestión de Eventos	5
3.3 Cambio de imágenes por defecto.	12
3.4 Cambio de los textos por defecto y cambio de idioma.....	14
3.5 Plantillas de correos electrónicos	16
4. Despliegue del entorno para pruebas.....	16
4.1 Posibles Problemas:	19
5. Despliegue en un servidor de internet.....	19
ANEXO A – Base de Datos	21

1. ¿Qué es AsoGest y para qué sirve?



DATOS GENERALES DE LA VERSIÓN v1.4-Beta:

- Laravel Framework 9.40.1
- Apache 2.4.54
- PHP 8.0.7 (cli)
- MySQL – MariaDB - 8.1.10

NOTA: En la actualidad, Asogest se encuentra en fase BETA de pruebas. Realmente, como desarrollador, no tengo mucho tiempo libre, así que probablemente se quede en beta. Esto quiere decir que aún pueden surgir fallos de eventos no controlados, aunque la funcionalidad completa de la aplicación ya ha sido testeada (lleva más de un año funcionando en una asociación)

Asogest es un programa Web orientado a la gestión de Asociaciones Culturales sin ánimo de lucro.

Utiliza tecnología Laravel, de código abierto, así como otros módulos libres. Se precisa un servidor web con PHP y una Base de datos MySQL (MariaDB).

Permite entre otras cosas realizar:

- Gestión de la Secretaría de la asociación:
 - Vista de Secretario con la gestión completa de socios (creación, edición, etc).
 - Visor y gestión de las invitaciones del socio.
 - Habilitar y deshabilitar Socios
 - Información de acceso a Drive o carpetas compartidas (externas). Se puede cambiar por otras opciones, como gestión de llaves, etc., a través de dos variables personalizables (*Acceso Drive* y *Acceso Junta*).
 - Gestión de las vocalías (creación, eliminación y edición)
- Gestión de la Tesorería:
 - Panel de control para el tesorero con visión general del estado de la tesorería.
 - Gestión de apuntes contables (creación, edición) con ajuste al Plan General Contable (Criterio de Devengo).
 - Visionado de los apuntes contables en diferentes vistas con filtrado.
 - Gestión de Cuotas de los socios
 - Creación de tipos de cuota, gestión y renovación de cuotas de socio.
 - Avisos automáticos por email cuando está próximo el vencimiento de la cuota (hay que habilitar el Cron en el servidor)
 - Visionado de las cuotas atrasadas, próximas renovaciones, etc.
 - Moratoria de cuotas
 - Exportación e Importación de los apuntes contables en Excel, copiados en portapapeles o impresión.
 - Creación de certificado en PDF del coste de mantenimiento del local (Alquiler)
 - Creación de informes personalizados de tesorería.
- Gestión de la ficha del socio
 - Datos personales (nombre, apellido, dirección, email, teléfono, etc.)
 - Gestión de eventos creados por el socio (eliminar los eventos y reservas realizadas) (necesaria cuenta de GMAIL)
 - Gestión por parte del socio de sus invitaciones
 - Preguntas sobre Privacidad y permisos del socio para comunicaciones.

- Gestión de preferencias de Vocalías (muestra el interés del socio en esa vocalía. Se puede usar para destinar parte de la cuota a sus vocalías o intereses)
- Gestión del rol del socio (tesorero, secretario, vocal, socio normal, etc...)
- Gestión de eventos creados por el socio (necesaria cuenta de GMAIL).
- Gestión de Vocalías (Secciones o Grupo de Trabajo)
 - Cada vocalía actúa de manera independiente
 - Calendario independiente para cada vocalía (necesaria cuenta de GMAIL)
 - Panel del vocal donde se expone información personalizada
 - Gestión de propuestas de compra con un sistema de votaciones
 - Histórico de compras de esa vocalía
 - Gestión del presupuesto de cada vocalía
 - Envío de email del vocal a los socios interesados en la vocalía o a socios en concreto.
- Gestión de Eventos
 - Calendario de Google con sincronización (necesaria cuenta de GMAIL).
 - Creación de eventos por vocalía, generales o importantes (necesaria cuenta de GMAIL).
 - Visualización de los próximos eventos en la web (diferentes formatos)
- Emails:
 - Envíos de email entre socios (al crear un evento)
 - Envío de email Vocal-Socio (al hacer una compra, para hacer un comunicado de vocalía, etc.)
 - Envío de email de socio a junta directiva.
- Visor de Registros (ver las acciones que se han realizado en la web y quién las ha hecho)
- Gestor Documental (almacenamiento de documentos de la Asociación)

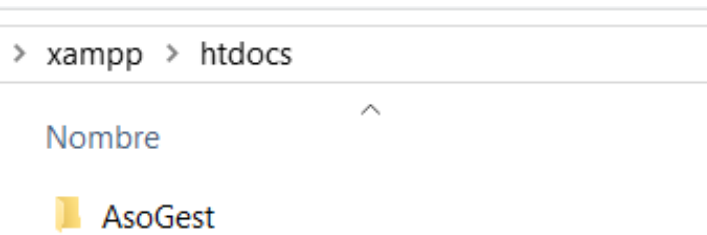
2. Instalación en ordenador Local

Requisitos Mínimos:

Para la prueba de la aplicación en un PC local necesitaremos:

- Una instalación de **XAMPP** o LAMPP con PHP versión 8.0 u otro servidor virtual (en este ejemplo usaremos XAMPP).
- Una Base de Datos MySQL (incluida en XAMPP)
- Composer
- (opcional) Conocimientos de programación de PHP, Laravel, uso de Artisan, Gestión de BD con MySQL, etc (por si se quiere modificar algo)

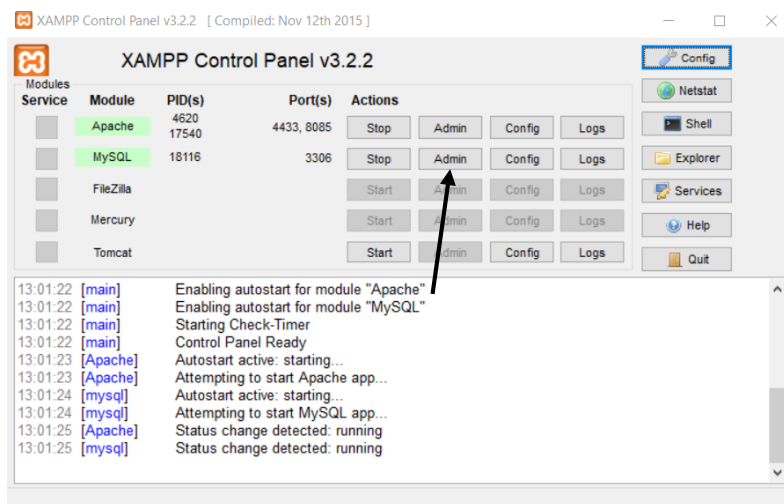
1.- Descomprimos la carpeta con el programa AsoGest en la carpeta HTDOCS de XAMPP



3. Configuración básica de la aplicación

3.1 Creación de la Base de Datos

Lo primero que haremos será crear una Base de datos. Para ello abrimos XAMPP y pinchamos en el icono “Admin” justo al lado de MySQL.



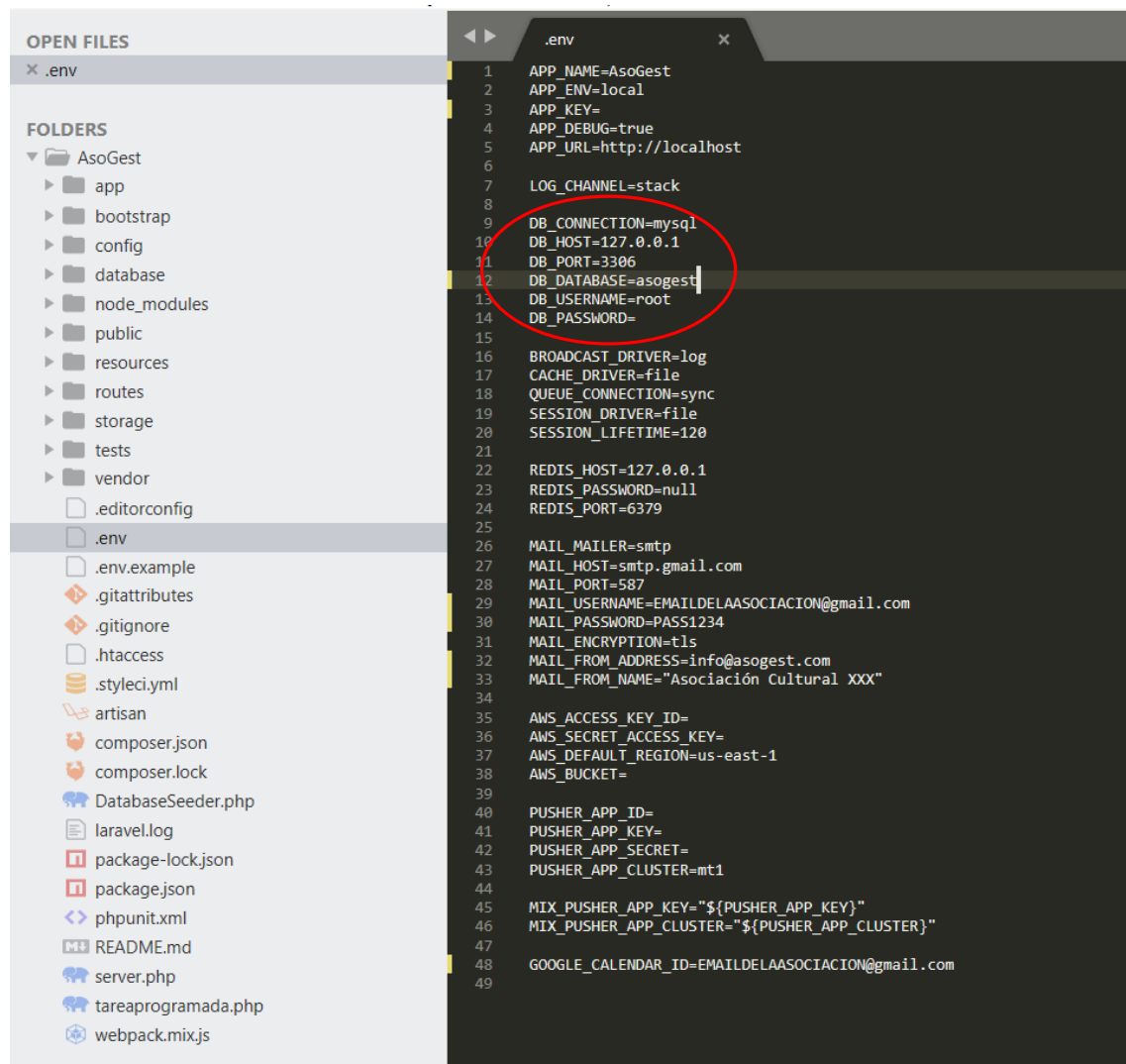
Se nos abrirá la pantalla de gestión de MySQL



- 1.- Seleccionamos “Nueva”
- 2.- Introducimos un **nombre** para la Base de Datos (en el ejemplo “asogest”) y seleccionamos el cotejamiento que queramos. En este ejemplo “utf8_spanish_ci”
- 3.- Clicamos en “Crear”

Ahora nos vamos a la aplicación. Podemos usar algún gestor de archivos, como Notepad++, Atom, SublimeText, Visualcode, etc.

Abrimos el archivo “.env” que se encuentra en la raíz y editamos los datos de la Base de Datos que hemos creado antes:



```
1 APP_NAME=AsoGest
2 APP_ENV=local
3 APP_KEY=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8
9 DB_CONNECTION=mysql
10 DB_HOST=127.0.0.1
11 DB_PORT=3306
12 DB_DATABASE=asogest
13 DB_USERNAME=root
14 DB_PASSWORD=
15
16 BROADCAST_DRIVER=log
17 CACHE_DRIVER=file
18 QUEUE_CONNECTION=sync
19 SESSION_DRIVER=file
20 SESSION_LIFETIME=120
21
22 REDIS_HOST=127.0.0.1
23 REDIS_PASSWORD=null
24 REDIS_PORT=6379
25
26 MAIL_MAILER=smtp
27 MAIL_HOST=smtp.gmail.com
28 MAIL_PORT=587
29 MAIL_USERNAME=EMAILDELAASOCIACION@gmail.com
30 MAIL_PASSWORD=PASS1234
31 MAIL_ENCRYPTION=tls
32 MAIL_FROM_ADDRESS=info@asogest.com
33 MAIL_FROM_NAME="Asociación Cultural XXX"
34
35 AWS_ACCESS_KEY_ID=
36 AWS_SECRET_ACCESS_KEY=
37 AWS_DEFAULT_REGION=us-east-1
38 AWS_BUCKET=
39
40 PUSHER_APP_ID=
41 PUSHER_APP_KEY=
42 PUSHER_APP_SECRET=
43 PUSHER_APP_CLUSTER=mt1
44
45 MIX_PUSHER_APP_KEY="{PUSHER_APP_KEY}"
46 MIX_PUSHER_APP_CLUSTER="{PUSHER_APP_CLUSTER}"
47
48 GOOGLE_CALENDAR_ID=EMAILDELAASOCIACION@gmail.com
49
```

3.2 Configuración de Google Calendar para la gestión de Eventos

Este programa tiene una gestión de eventos completa con Google Calendar. Esto permite a los socios crear eventos (como partidas, juegos, reuniones, citas, actividades, etc.) y compartirlas con el resto de los socios.

Se basa en el Plugin de Spatie “Google-Calendar” que podemos encontrar aquí: <https://github.com/spatie/laravel-google-calendar>

El primer paso es crearnos una cuenta de Google (GMAIL). Podemos usar una si ya la tenemos creada. (Sigue las instrucciones aquí si tienes dudas: <https://support.google.com/accounts/answer/27441?hl=es>)

Cuando lo hayamos creado, debemos introducir los datos en el mismo fichero “.env” de antes:

NOTA (es recomendado): En el apartado **MAIL_*** podemos poner los datos de nuestro actual proveedor de correo electrónico (hosting) en lugar del de Gmail. Esto hará más fluido el envío de email, ya que no tendremos que desactivar las opciones de seguridad de Google.

```
OPEN FILES
x .env

FOLDERS
v AsoGest
  app
  bootstrap
  config
  database
  node_modules
  public
  resources
  routes
  storage
  tests
  vendor
.editorconfig
.env
.env.example
.gitattributes
.gitignore
.htaccess
.styleci.yml
artisan
composer.json
composer.lock
DatabaseSeeder.php
laravel.log
package-lock.json
package.json
phpunit.xml
README.md
server.php
tareaprogramada.php
webpack.mix.js

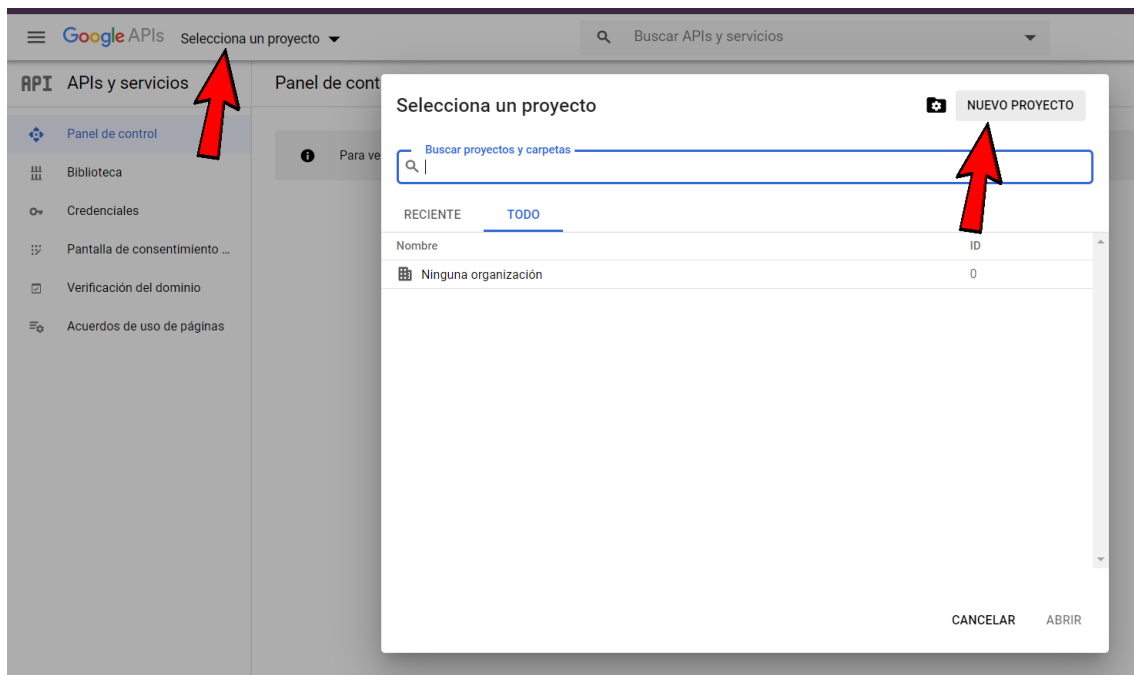
1 APP_NAME=AsoGest
2 APP_ENV=local
3 APP_KEY=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8
9 DB_CONNECTION=mysql
10 DB_HOST=127.0.0.1
11 DB_PORT=3306
12 DB_DATABASE=asogest
13 DB_USERNAME=root
14 DB_PASSWORD=
15
16 BROADCAST_DRIVER=log
17 CACHE_DRIVER=file
18 QUEUE_CONNECTION=sync
19 SESSION_DRIVER=file
20 SESSION_LIFETIME=120
21
22 REDIS_HOST=127.0.0.1
23 REDIS_PASSWORD=null
24 REDIS_PORT=6379
25
26 MAIL_MAILER=smtp
27 MAIL_HOST=smtp.gmail.com
28 MAIL_PORT=587
29 MAIL_USERNAME=asogest2020@gmail.com
30 MAIL_PASSWORD=PASS1234
31 MAIL_ENCRYPTION=tls
32 MAIL_FROM_ADDRESS=info@asogest.com
33 MAIL_FROM_NAME="Asociación Cultural AsoGest"
34
35 AWS_ACCESS_KEY_ID=
36 AWS_SECRET_ACCESS_KEY=
37 AWS_DEFAULT_REGION=us-east-1
38 AWS_BUCKET=
39
40 PUSHER_APP_ID=
41 PUSHER_APP_KEY=
42 PUSHER_APP_SECRET=
43 PUSHER_APP_CLUSTER=mt1
44
45 MIX_PUSHER_APP_KEY="${PUSHER_APP_KEY}"
46 MIX_PUSHER_APP_CLUSTER="${PUSHER_APP_CLUSTER}"
47
48 GOOGLE_CALENDAR_ID=asogest2020@gmail.com
49
```

Ahora vamos a la Consola de Administrador de Google para crear un nuevo proyecto y darle permisos (es recomendable hacerlo desde una **ventana InPrivate**-ventana privada, para que no se contamine con otros inicios de sesión de google):

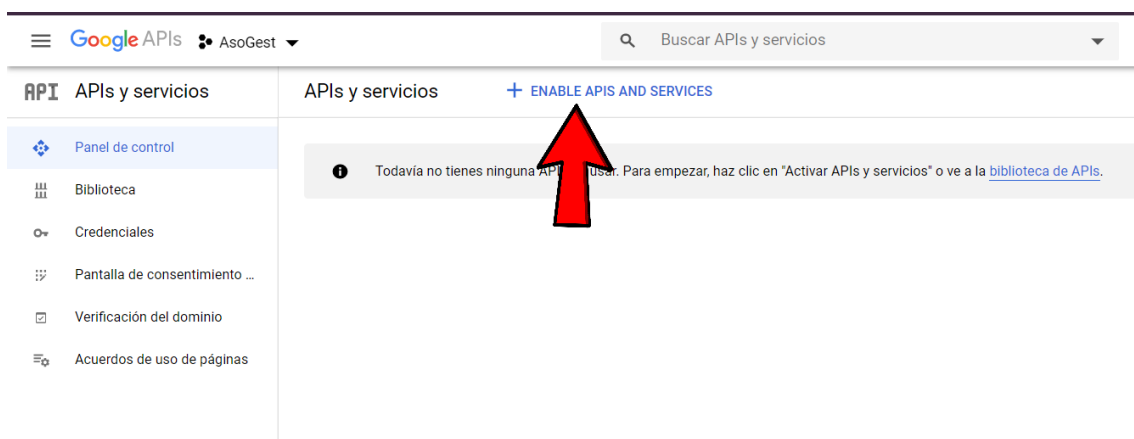
<https://console.developers.google.com/apis>

Iniciamos sesión con las **credenciales que hemos puesto para esa cuenta de Gmail**.

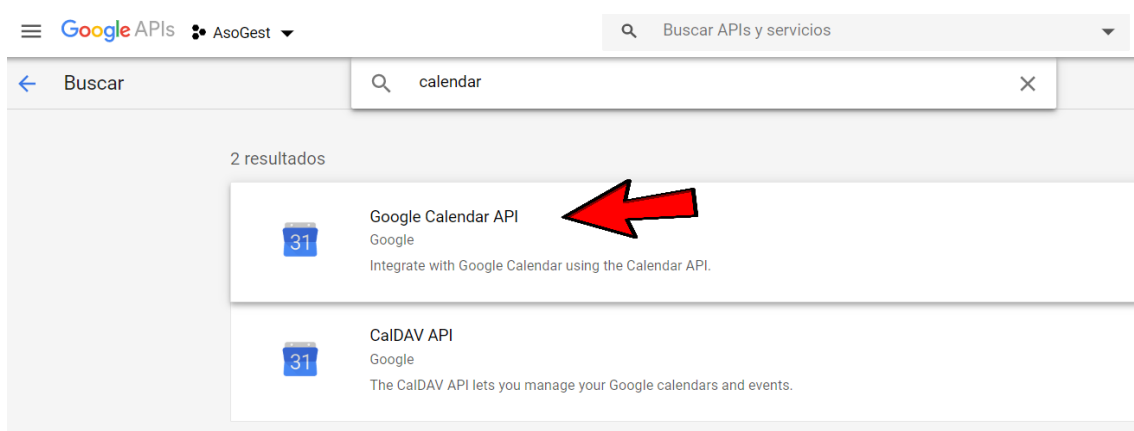
En el Panel de Control, en la parte superior, seleccionamos "Selecciona un Proyecto". A continuación, elegimos "Nuevo Proyecto".



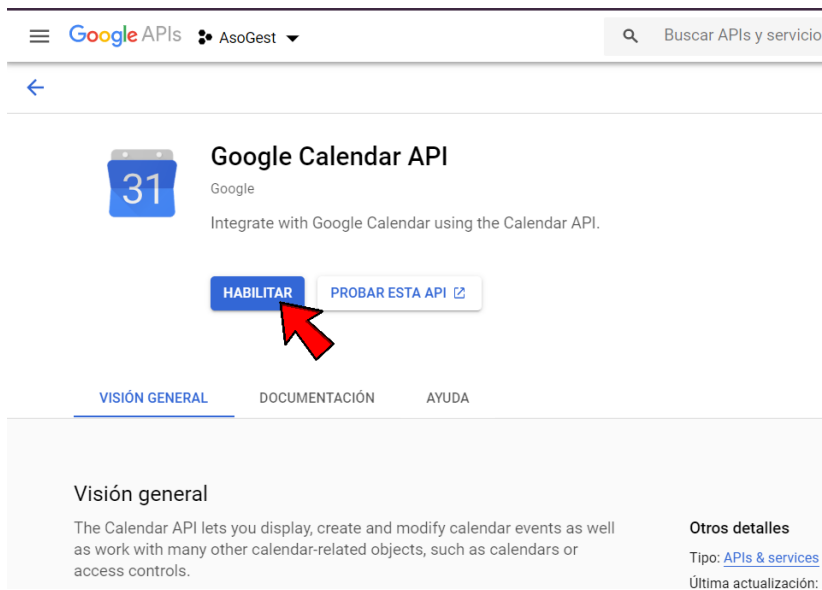
A continuación, elegimos **“ENABLE APIS AND SERVICES”**.



En el cuadro de Búsqueda escribimos **“Calendar”** y seleccionaremos la **Api de Google Calendar**



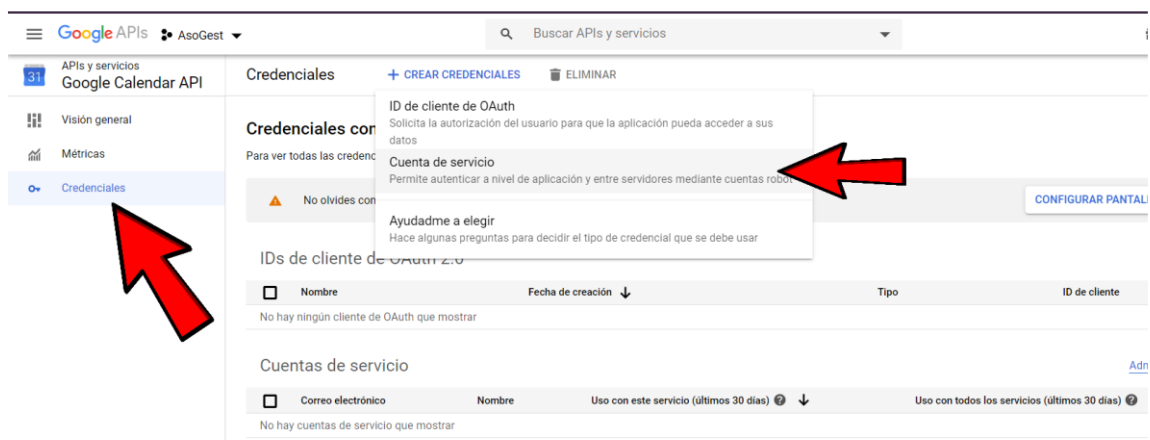
A continuación, pinchamos encima de **“Habilitar”**



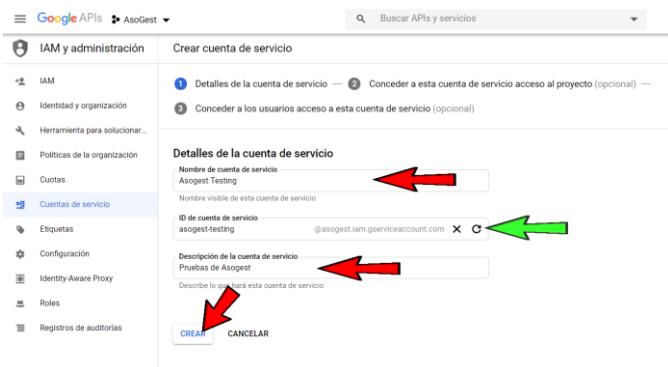
Ahora ya hemos creado un proyecto que tiene acceso a la API de Google Calendar para crear eventos en el Calendario.

Ahora vamos a descargar las credenciales de ese proyecto para que Laravel pueda conectarse a través de él y gestionar el calendario de Google.

Seleccionamos, en la siguiente ventana que sale al darle “Habilitar” en el punto anterior, el botón “Credenciales” del menú de la izquierda y seleccionamos “Cuenta de Servicio”.



Deberemos rellenar dos de las opciones que aparecen.

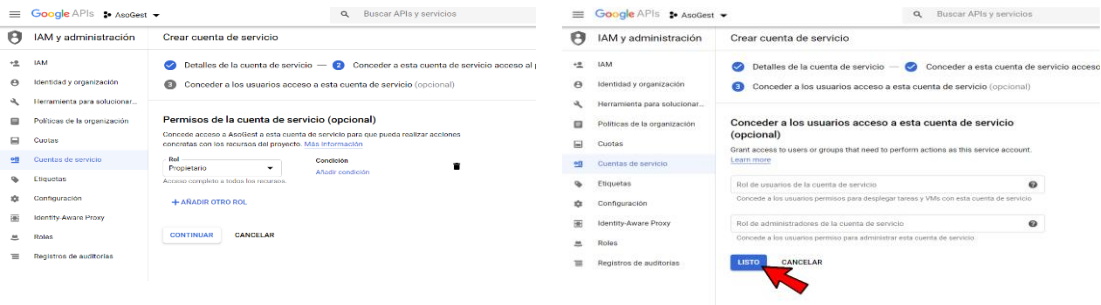


En Nombre debemos darle un nombre a esa cuenta de servicio.

El ID de cuenta aparece automáticamente, y viene bien recordarlo porque es el que usará la aplicación.

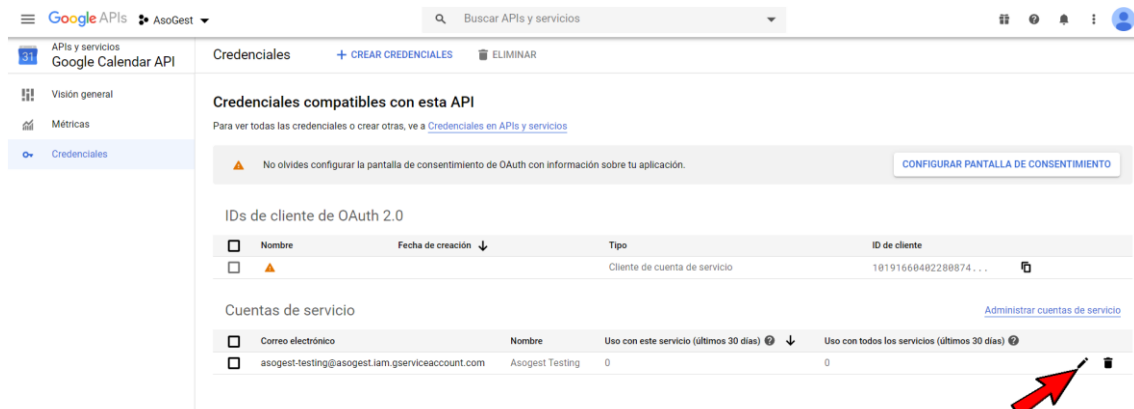
En la descripción podremos poner lo que deseemos, una descripción del uso que le daremos al servicio de la API.

A continuación, nos pedirá un rol. Aunque no es obligatorio, podemos seleccionar **“Propietario”**. En la siguiente ventana no elegiremos nada y pulsaremos en **“Listo”**

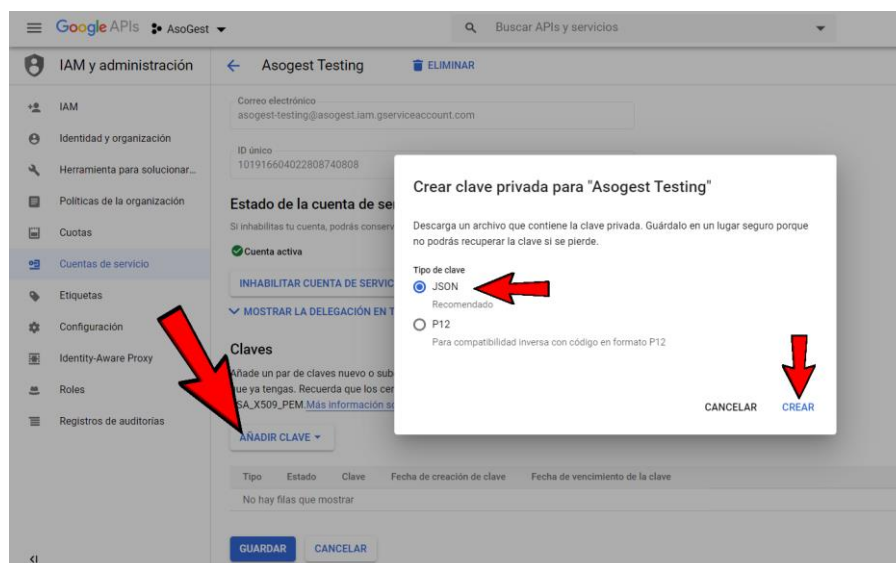


Una vez finalizado el proceso, llegaremos a la pantalla de Credenciales otra vez, donde ya aparecerá nuestra Cuenta de Servicio.

Pulsamos encima del lapicero para editarla.

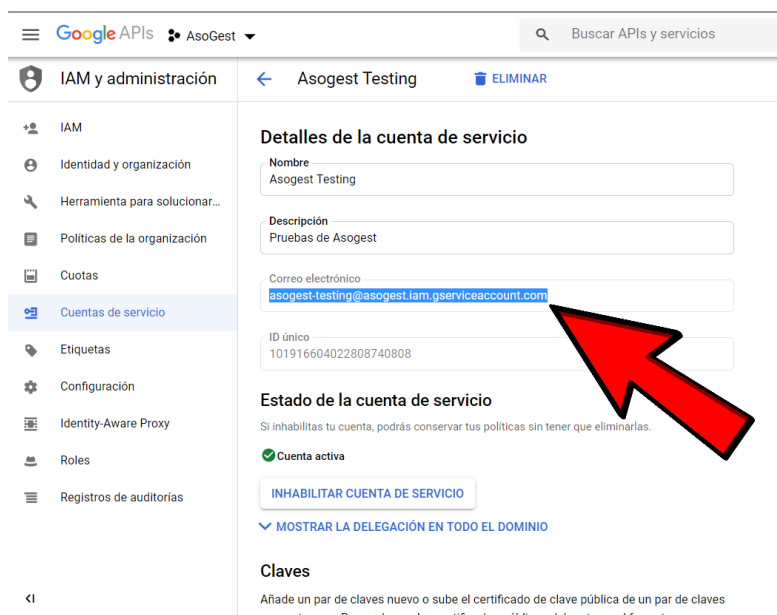


En la nueva ventana que nos sale bajamos un poco y presionamos el botón **“AÑADIR CLAVE”**. Luego seleccionamos el tipo de clave **“JSON”** y pinchamos en **“CREAR”**

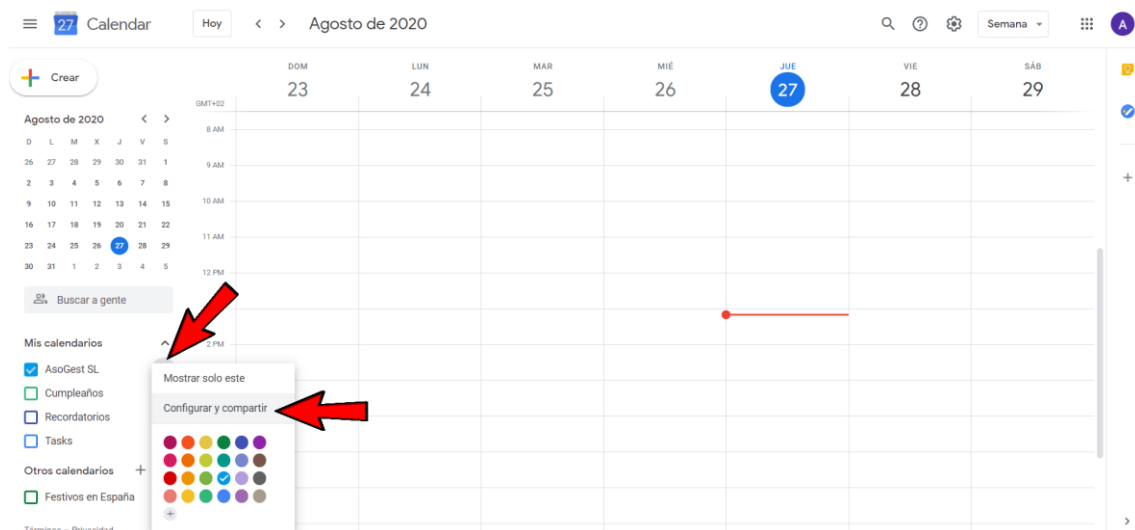


Nos descargará un archivo JSON que **debemos guardar de manera segura y confidencial**.

Un poco más arriba de donde se crean las claves, veremos los datos de antes, nombre de la clave, correo electrónico y descripción. **Seleccionaremos y copiaremos** (doble click encima y CTRL+C) el **email del servicio**

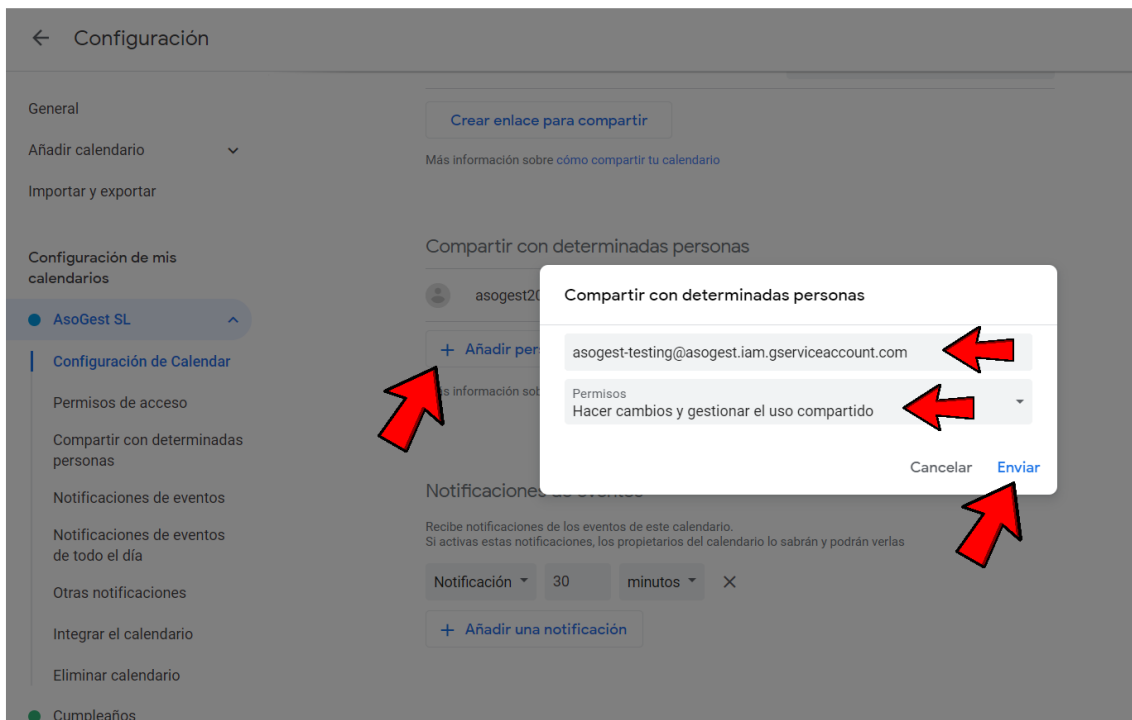


Abrimos una nueva pestaña o ventana en el explorador y nos vamos a **Calendario de Google** (<https://calendar.google.com>). Una vez estemos dentro del calendario, buscaremos nuestro calendario, pulsaremos encima de los tres puntitos (menú) de su derecha y elegiremos **“Configurar y Compartir”**

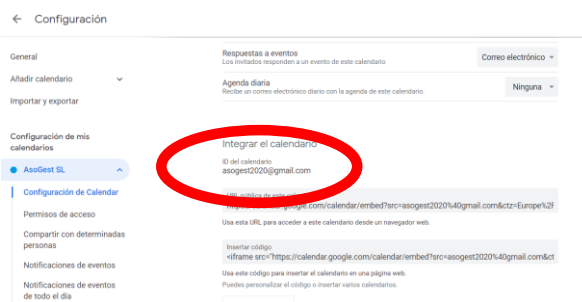


En la ventana de configuración deberemos bajar un poco hacia abajo, y en la sección de **“Compartir con determinadas personas”** seleccionamos la opción **“Añadir persona”**.

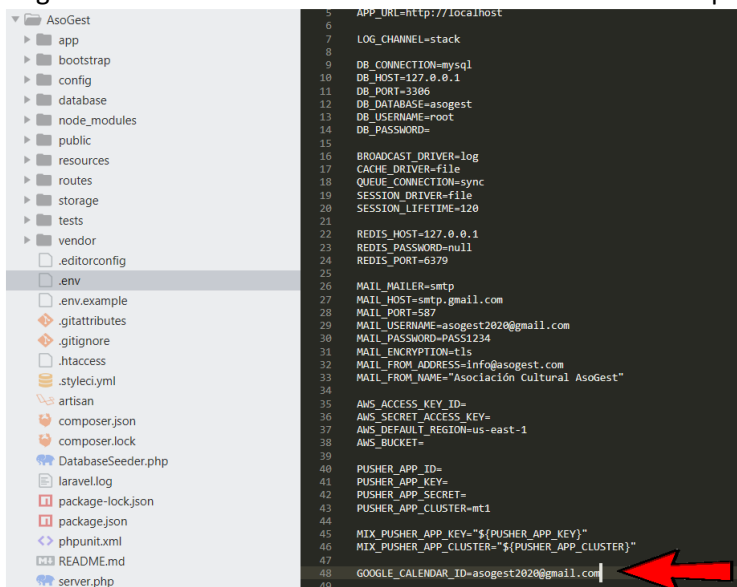
En el recuadro que nos sale, pegamos la dirección de correo electrónico de nuestro servicio, que hemos copiado en el paso anterior, y elegimos el permiso **“Hacer cambios y gestionar el uso compartido”**

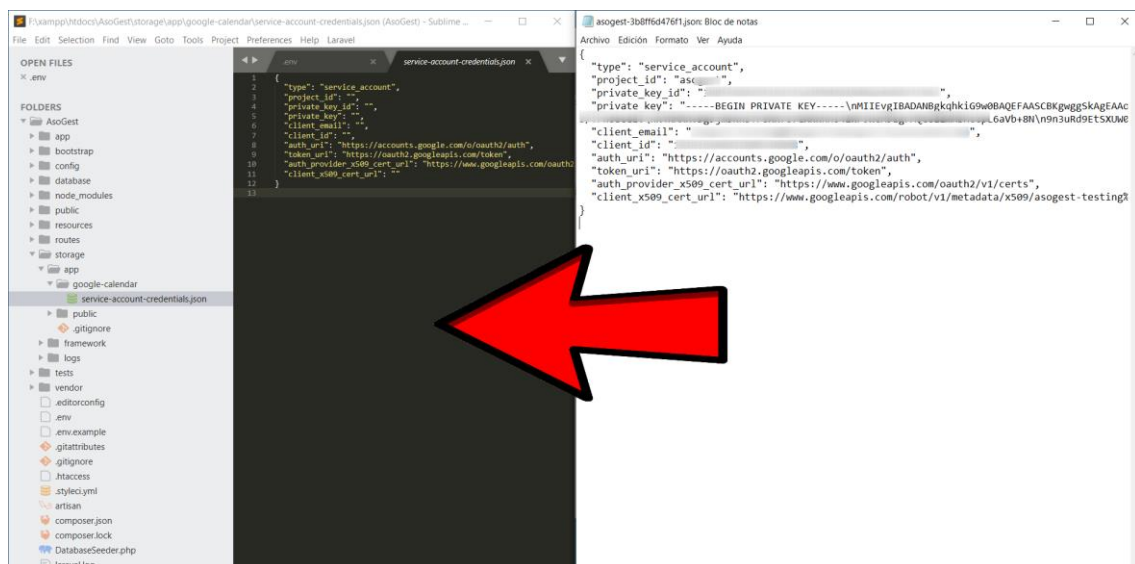


En el siguiente paso, deberemos copiar la **Identificación del calendario** para que sea gestionada por nuestro programa Asogest. Un poco más abajo la tenemos. Esa dirección es la que tendremos que indicarle en la configuración del programa, como **ID del Calendario para eventos importantes**.



Pegaremos esa dirección en el fichero **“.ENV”** de la raíz del programa:





3.3 Cambio de imágenes por defecto.

La aplicación tiene una serie de imágenes por defecto que podremos cambiar (En el caso de los **logotipos** deberían cambiarse para adecuarlos a nuestra asociación). Se encuentran todas en la carpeta **“public->images”**:

AsoGest > public > images					Buscar en imágenes
Nombre	Fecha	Tipo	Tamaño	Dimensiones	
favicon	14/10/2022 11:14	Carpeta de archivos			
fotos	14/10/2022 11:14	Carpeta de archivos			
working	14/10/2022 11:14	Carpeta de archivos			
configuracion.png	02/01/2021 20:32	Archivo PNG	304 KB	523 x 354	
default2.jpg	02/01/2021 20:32	Archivo JPG	13 KB	480 x 480	
documentos.png	02/01/2021 20:32	Archivo PNG	552 KB	664 x 445	
informes.jpg	23/03/2020 13:18	Archivo JPG	42 KB	608 x 399	
logo.png	13/11/2020 11:51	Archivo PNG	25 KB	295 x 290	
logoB.png	13/11/2020 11:58	Archivo PNG	83 KB	854 x 640	
logoCargando.gif	10/06/2021 10:18	Archivo GIF	47 KB	246 x 222	
logoPeq.png	27/10/2022 10:43	Archivo PNG	11 KB	171 x 128	
mesa.jpg	06/08/2010 18:35	Archivo JPG	84 KB	480 x 336	
minis.jpg	02/01/2021 20:32	Archivo JPG	58 KB	421 x 340	
prohibido.gif	02/01/2021 20:32	Archivo GIF	702 KB	338 x 161	

Podemos cambiar estas imágenes por otras que nos gusten más simplemente copiando una nueva imagen encima y poniéndole el mismo nombre de archivo y manteniendo el tamaño de la imagen original.

Las primeras imágenes que deberemos cambiar serán los favicons (en la carpeta favicon), que son las imágenes en miniatura que usará el programa a través del navegador.

Para crear los favicons aconsejo utilizar un servicio online que nos genere todas las imágenes automáticamente. Por ejemplo: <https://www.favicon-generator.org/>

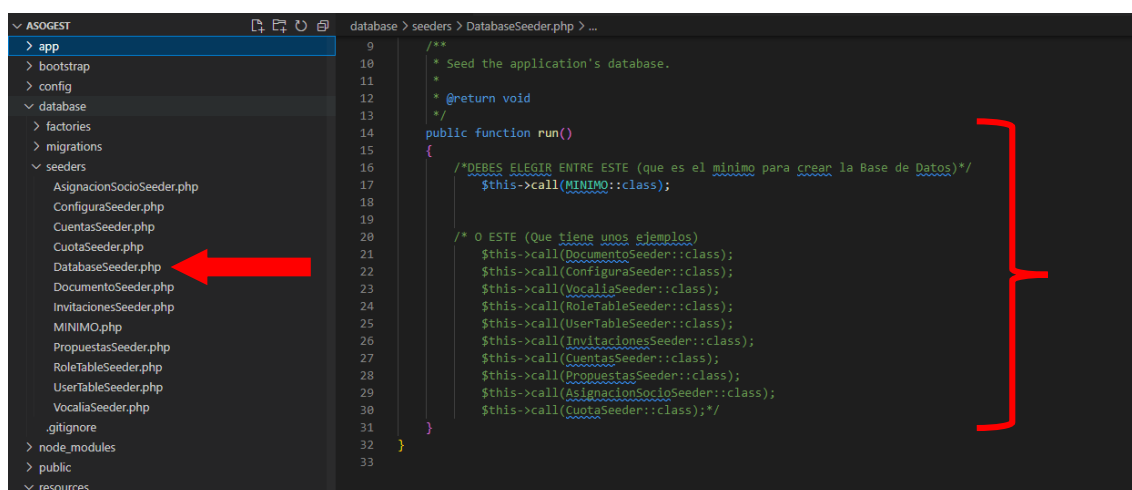
También podremos modificar las imágenes de las vocalías de ejemplo y demás imágenes.

La carpeta fotos contiene las fotos para los datos de prueba y las imágenes que el programa utiliza para cuando un usuario no tiene imagen. Hay varias imágenes de prueba, tanto para hombres como para mujeres, porque el programa saca una aleatoria cada vez... Me pareció divertido.

Existe una **imagen especial** que también debemos modificar. Se encuentra en la base de datos que se ha generado. Para modificar esta imagen (es el logotipo que se usa, por ejemplo, en los **emails o en los PDF**) podremos hacerlo desde el fichero de semillas correspondiente. Existen dos:

1. Fichero de semillas mínimo
2. Fichero de semillas para pruebas

Deberemos elegir uno de ellos, comentando el otro. En la siguiente imagen, está señalado el MINIMO, con lo cual, al ejecutar el comando `db:seed` se alimentará la base de datos con lo mínimo para su funcionamiento. Si queremos alimentar la base de datos con ejemplos y pruebas, deberemos comentar la línea `$this->call(MINIMO::call);` y descomentar lo de abajo

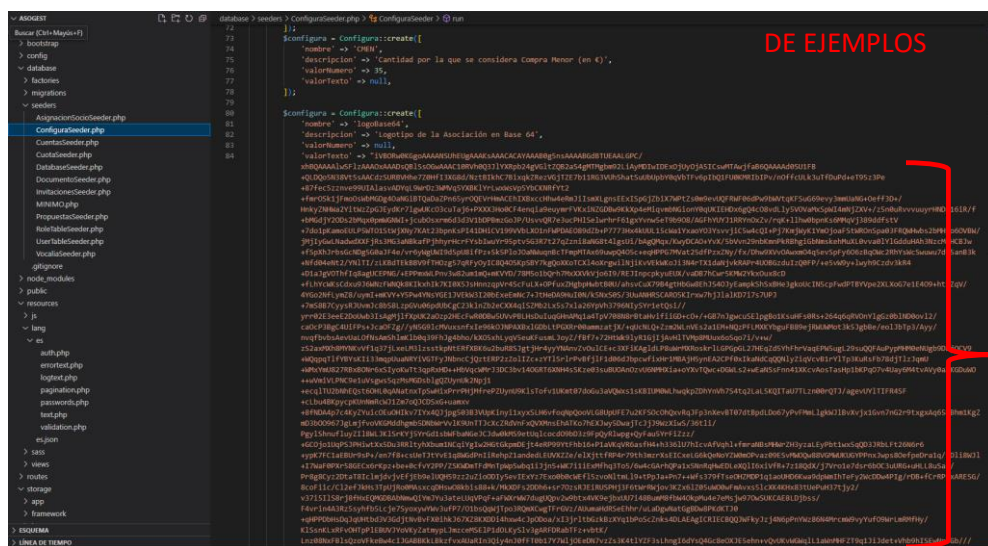


El logotipo en Base 64. Está codificado así porque al crear un documento PDF o un email, el logotipo no se podía insertar como imagen png/jpg, es un fallo conocido y no resuelto del módulo domPDF de Laravel. Así que la solución es crearlo en Base 64 y añadirlo así.

Para cambiar una imagen a Base64 podemos hacer uso de alguna web online: <https://www.base64-image.de/>

Dependiendo de si hemos escogido el **semillero MINIMO o el de ejemplo**, deberemos modificar el apartado correspondiente al logotipo en base 64:

- MINIMO: Editar el archivo en la línea 176 eliminando lo que hay y poniendo el nuevo código generado en Base64
- EJEMPLOS: Editar el archivo



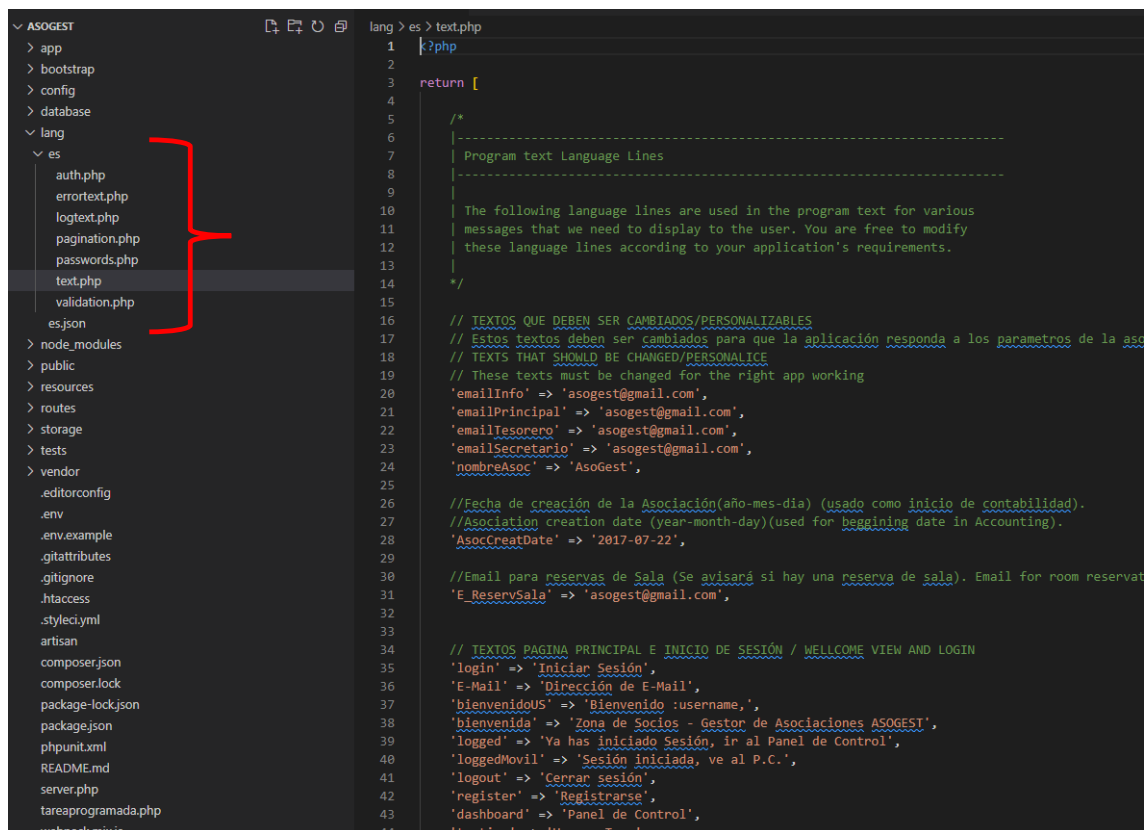
NOTA: Este logotipo especial en Base 64 se puede modificar en cualquier momento en el apartado “Configuración” una vez iniciado el programa

El programa tiene una serie de textos, mensajes y palabras que se pueden modificar. Todos los textos que aparecen en el programa y en las páginas web se gestionan a través de los archivos de lenguaje de Laravel. Se encuentran en un archivo llamado “Text.php” en la carpeta raíz “Lang->es”.

Podemos tener varios idiomas instalados y gestionar el cambio de idioma con un botón (o según la localización geográfica de la persona que acceda a la página. Para gestionar esto, ver la documentación de Laravel:

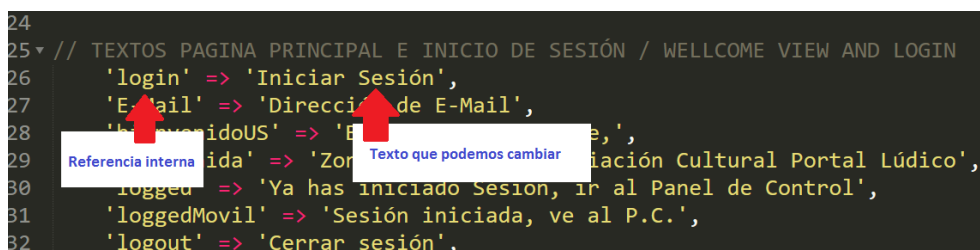
<https://laravel.com/docs/8.x/localization>

Para cambiar los textos de la página web, podremos copiar la carpeta LANG y ponerle el código de idioma (EN, ES, FR, IT, etc...). Luego en app.config decirle que idioma seleccionará.



```
1 <?php
2
3 return [
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
```

Este archivo consta de dos textos en cada línea, el primero es la referencia interna del programa para ese texto y que **NO PODEMOS MODIFICAR**, y el segundo es el texto que podremos modificar.



```
25 // TEXTOS PAGINA PRINCIPAL E INICIO DE SESIÓN / WELLCOME VIEW AND LOGIN
26 'login' => 'Iniciar Sesión',
27 'E-Mail' => 'Dirección de E-Mail',
28 'bienvenidoUS' => 'Bienvenido :username,',
29 'bienvenida' => 'Zona de Socios - Gestor de Asociaciones ASOGEST',
30 'logged' => 'Ya has iniciado Sesión, ir al Panel de Control',
31 'loggedMovil' => 'Sesión iniciada, ve al P.C.',
32 'logout' => 'Cerrar sesión',
```

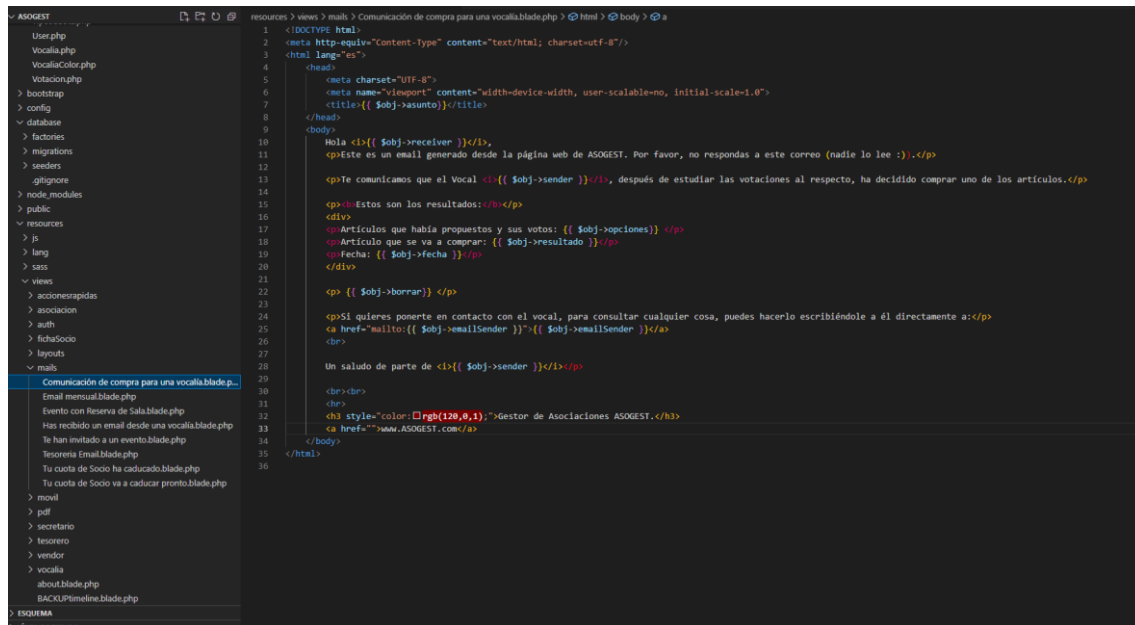
Es importante **cambiar los emails** que aparecen en el comienzo de este archivo, ya que serán los de contacto de la aplicación y entre los usuarios.

En el archivo Helper (APP->Helper->Helper.php) también hay dos funciones, creadas por mí, que **personalizan la salida de fechas** a un formato español. Si queremos cambiar a otros idiomas, podemos modificar también estos formatos de fecha para que se adecuen al idioma.

3.5 Plantillas de correos electrónicos

La aplicación tiene la posibilidad de enviar diversos correos electrónicos a los socios. Estas plantillas de correos se corresponden con las diferentes opciones que da la aplicación, como la de enviar un email cuando la cuota de socio esté a punto de caducar, o cuando se realice una compra dentro de una vocalía.

Las plantillas se encuentran en la ruta “Resources->Views->Mails”.



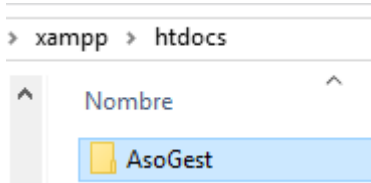
```
1 <DOCTYPE html>
2 <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
3 <html lang="es">
4 <head>
5 <meta charset="UTF-8">
6 <meta name="viewport" content="width=device-width, user-scalable=no, initial-scale=1.0">
7 <title>{{ $obj->asunto }}</title>
8 </head>
9 <body>
10 <p>Hola {{ $obj->receiver }}</p>
11 <p>Este es un email generado desde la página web de ASOGEST. Por favor, no respondas a este correo (nadie lo lee :)).</p>
12 <p>Te comunicamos que el Vocal <{{ $obj->sender }}>, después de estudiar las votaciones al respecto, ha decidido comprar uno de los artículos.</p>
13 <p>Estos son los resultados:</p>
14 <div>
15 <p>Artículos que había propuestos y sus votos: {{ $obj->opciones }}</p>
16 <p>Artículo que se va a comprar: {{ $obj->resultado }}</p>
17 <p>Fecha: {{ $obj->fecha }}</p>
18 </div>
19 <p>{{ $obj->borrar }}</p>
20 <p>Si quieres ponerte en contacto con el vocal, para consultar cualquier cosa, puedes hacerlo escribiéndole a él directamente a:</p>
21 <a href="mailto:{{ $obj->emailSender }}">{{ $obj->emailSender }}</a>
22 <br>
23 <p>Un saludo de parte de {{ $obj->sender }}</p>
24 <br>
25 <div style="color: #000080; font-weight: bold;>
26 <p>Gestor de Asociaciones ASOGEST.</p>
27 <a href="http://www.ASOGEST.com/>
28 </div>
29 </body>
30 </html>
```

Podemos modificar el texto plano que aparece, pero **debemos respetar lo que hay entre llaves {{ XXX }}** ya que son variables que se usan en la aplicación.

Si sabemos programación de PHP, podremos modificar estas variables y poner lo que nosotros deseemos.

4. Despliegue del entorno para pruebas

Asogest viene preparado para realizar pruebas y ver el funcionamiento el programa con unos datos de ejemplo.



Para desplegar estos datos, primero tendremos que tener una versión de XAMPP (con PHP 8.0) instalada y en funcionamiento.

Copiaremos la carpeta AsoGest directamente dentro de la raíz de la carpeta “XAMPP->HTDOCS”.

También tendremos que asegurarnos de contar con Composer instalado y funcionando en el sistema.

<https://www.hostinger.es/tutoriales/como-instalar-composer>

Para saber si tenemos PHP instalado, al menos en PHP 8. Podemos abrir un CMD y ejecutar el comando:

php --version

Para saber si tienes Composer, con una versión actualizada, puedes usar el comando:

composer -v

```
Símbolo del sistema

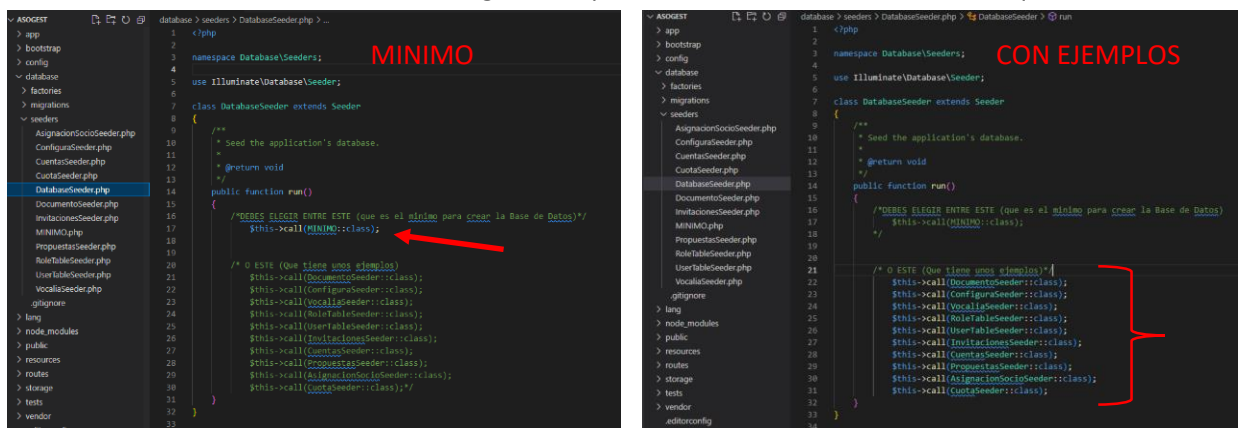
C:\>php --version
PHP 8.0.7 (cli) (built: Jun 2 2021 00:41:03) ( ZTS Visual C++ 2019 x64 )
Copyright (c) The PHP Group
Zend Engine v4.0.7, Copyright (c) Zend Technologies

C:\>composer -v

Composer version 2.4.3 2022-10-14 16:56:41
```

Cuando comprobemos que tenemos las versiones correctas de los programas, deberemos seleccionar un tipo de semilla en el archivo “DatabaseSeeder.php” en la carpeta Database.

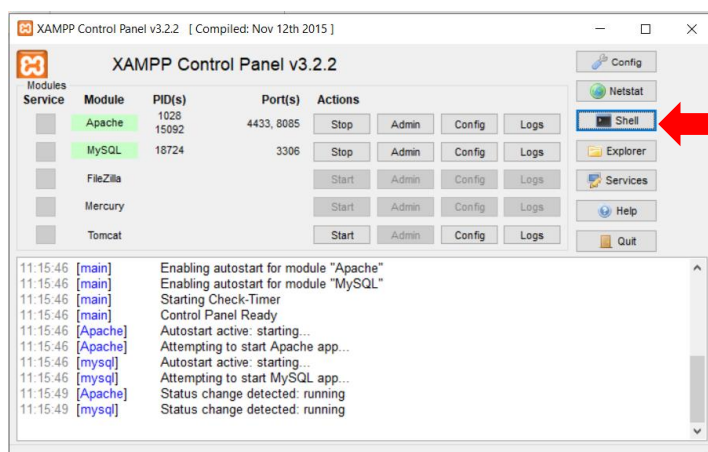
Allí comentaremos el trozo de código correspondiente a la modalidad de datos que deseemos.



Si queremos iniciar la aplicación con los datos mínimos para funcionar, la dejaremos con la primera parte sin comentar. Si queremos ver un ejemplo completo de la aplicación, con usuarios, cuentas, etc, comentaremos la línea de mínimo y quitaremos los comentarios a las de abajo.

Si todo está funcionando nos vamos a la raíz de Asogest con un entorno de línea de comandos.

Ahora lo que vamos a hacer es Crear estructura de la Base de Datos e introducirle los datos de prueba (o los mínimos, según tengamos seleccionados en DatabaseSeeder). Para ello, podemos pulsar el botón “Shell” en el Panel de Control de XAMPP



Una vez abierto el Shell debemos asegurarnos asegurándonos de estar en la raíz del proyecto.



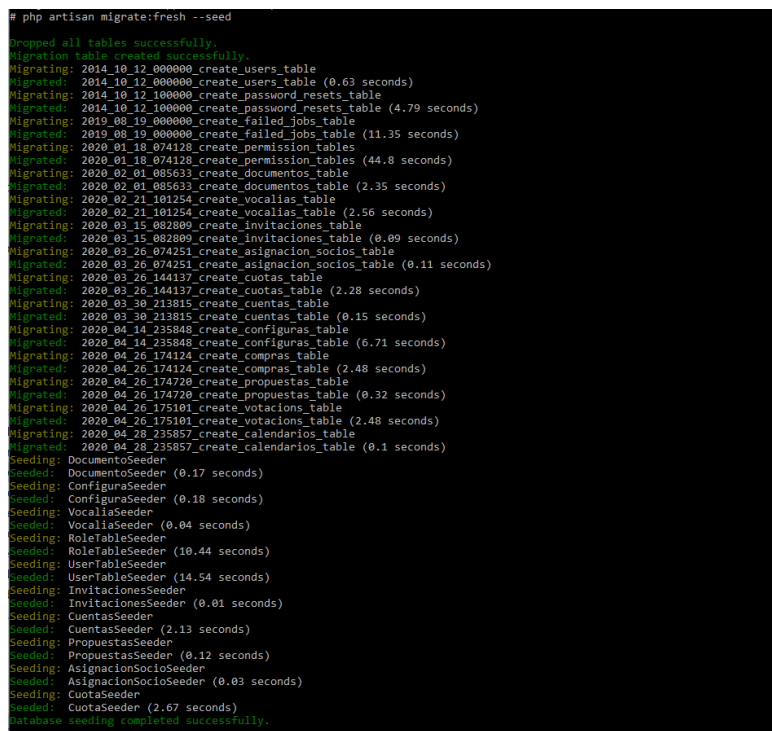
```
XAMPP for Windows
f:\xampp\htdocs
# cd AsoGest
f:\xampp\htdocs\AsoGest
#
```

A continuación, escribimos estos dos comandos, uno tras el otro:

- **php artisan key:generate**
- **php artisan migrate:fresh --seed**

Lo que hará Artisan será generar una nueva Key para nuestro proyecto, resetear toda la base de datos que hayamos indicado en el archivo “.env” y creará de nuevo las tablas y las alimentará con la información que se encuentra en la carpeta Seeder dentro de Database. (NOTA: Debes haber creado antes la base de datos en MySQL)

Nos debe aparecer algo así cuando termine de ejecutar el comando:



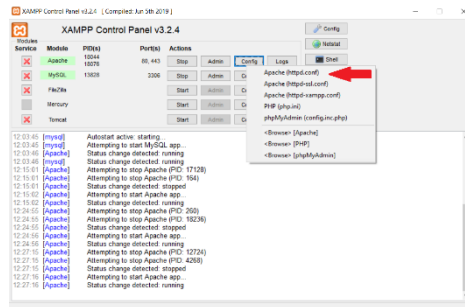
```
# php artisan migrate:fresh --seed
Dropped all tables successfully.
Migrating: 2014_10_12_000000_create_users_table (0.63 seconds)
Migrated: 2014_10_12_000000_create_users_table (0.63 seconds)
Migrating: 2014_10_12_100000_create_password_resets_table (4.79 seconds)
Migrated: 2014_10_12_100000_create_password_resets_table (4.79 seconds)
Migrating: 2019_08_19_000000_create_failed_jobs_table (11.35 seconds)
Migrated: 2019_08_19_000000_create_failed_jobs_table (11.35 seconds)
Migrating: 2020_01_18_074128_create_permission_tables (44.8 seconds)
Migrated: 2020_01_18_074128_create_permission_tables (44.8 seconds)
Migrating: 2020_02_01_085633_create_documentos_table (2.35 seconds)
Migrated: 2020_02_01_085633_create_documentos_table (2.35 seconds)
Migrating: 2020_02_21_101254_create_vocalias_table (2.56 seconds)
Migrated: 2020_02_21_101254_create_vocalias_table (2.56 seconds)
Migrating: 2020_03_15_082809_create_invitaciones_table (0.09 seconds)
Migrated: 2020_03_15_082809_create_invitaciones_table (0.09 seconds)
Migrating: 2020_03_26_074251_create_asignacion_socios_table (0.11 seconds)
Migrated: 2020_03_26_074251_create_asignacion_socios_table (0.11 seconds)
Migrating: 2020_03_26_144137_create_cuotas_table (2.28 seconds)
Migrated: 2020_03_26_144137_create_cuotas_table (2.28 seconds)
Migrating: 2020_03_30_213815_create_cuentas_table (0.15 seconds)
Migrated: 2020_03_30_213815_create_cuentas_table (0.15 seconds)
Migrating: 2020_04_14_235848_create_configuras_table (6.71 seconds)
Migrated: 2020_04_14_235848_create_configuras_table (6.71 seconds)
Migrating: 2020_04_26_174124_create_compras_table (2.48 seconds)
Migrated: 2020_04_26_174124_create_compras_table (2.48 seconds)
Migrating: 2020_04_26_174720_create_propuestas_table (0.32 seconds)
Migrated: 2020_04_26_174720_create_propuestas_table (0.32 seconds)
Migrating: 2020_04_26_175101_create_votaciones_table (2.48 seconds)
Migrated: 2020_04_26_175101_create_votaciones_table (2.48 seconds)
Migrating: 2020_04_28_235857_create_calendarios_table (0.1 seconds)
Migrated: 2020_04_28_235857_create_calendarios_table (0.1 seconds)
Seeding: DocumentoSeeder (0.17 seconds)
Seeded: DocumentoSeeder (0.17 seconds)
Seeding: ConfiguraSeeder (0.18 seconds)
Seeded: ConfiguraSeeder (0.18 seconds)
Seeding: VocaliaSeeder (0.04 seconds)
Seeded: VocaliaSeeder (0.04 seconds)
Seeding: RoleTableSeeder (10.44 seconds)
Seeded: RoleTableSeeder (10.44 seconds)
Seeding: UserTableSeeder (14.54 seconds)
Seeded: UserTableSeeder (14.54 seconds)
Seeding: InvitacionesSeeder (0.01 seconds)
Seeded: InvitacionesSeeder (0.01 seconds)
Seeding: CuentasSeeder (2.13 seconds)
Seeded: CuentasSeeder (2.13 seconds)
Seeding: PropuestasSeeder (0.12 seconds)
Seeded: PropuestasSeeder (0.12 seconds)
Seeding: AsignacionSocioSeeder (0.03 seconds)
Seeded: AsignacionSocioSeeder (0.03 seconds)
Seeding: CuotaSeeder (2.67 seconds)
Seeded: CuotaSeeder (2.67 seconds)
Database seeding completed successfully.
```

Ahora solo deberemos acceder a la dirección: <https://localhost/asogest> para disfrutar de la experiencia. Por defecto se habrá creado un usuario Administrador (**admin**) con la contraseña: **Admin1234**. (Esto puede variar si hemos modificado el archivo en Seeder).

4.1 Posibles Problemas:

En el despliegue normal de esta aplicación es posible que surjan problemas, como que aparezca un “Error 404” al intentar navegar por la Web.

La casuística es demasiado amplia, pero es posible que se deba a la configuración del servidor Apache, que no tiene las redirecciones permitidas. Esto se configura en el archivo “httpd.conf” de Apache:



En ese archivo tendremos que buscar esta línea:

AllowOverride None

Y cambiarla:

AllowOverride All

También debemos eliminar la # de esta línea:

LoadModule rewrite_module modules/mod_rewrite.so

También, si estamos desplegando en un servidor propio o en un hosting compartido, deberemos estar seguros de editar el archivo *php.ini* para **habilitar estas extensiones** quitando el ; del principio de cada línea (o solicitárselo a nuestro proveedor):

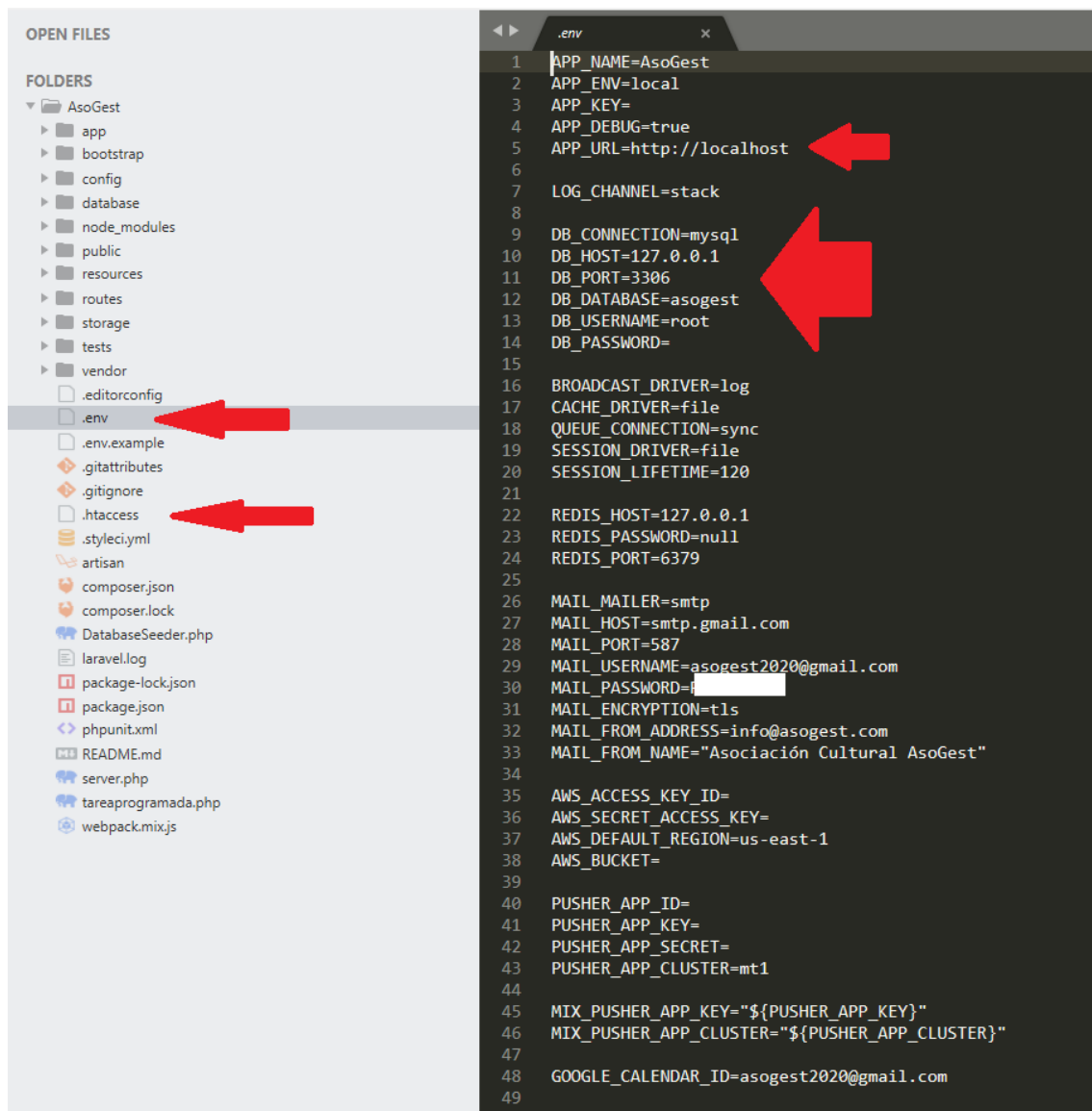
- extension=bz2
- extension=curl
- extension=fileinfo
- extension=gd
- extension=gettext
- extension=mbstring
- extension=exif
- extension=mysqli
- extension=pdo_mysql
- extension=pdo_sqlite

5. Despliegue en un servidor de internet

Para desplegar la aplicación en un servidor de internet, deberemos modificar tres archivos. El primero, y más importante se encuentra en la raíz del programa, y se llama “.env”.

En este fichero tendremos que modificar los valores de APP_URL y poner la correspondiente a nuestra dirección real de internet. Además de editar los otros valores que creamos necesarios (nombre de APP, Key, etc)

También tendremos que modificar los valores correspondientes a la Base de Datos que tengamos contratada con nuestro Hosting.



Con esto ya estaría todo listo para empezar a funcionar con nuestro nuevo Gestor de Asociaciones.



ANEXO A – Base de Datos

Este es el diagrama de la Base de Datos. Algunas tablas son independientes y otras tienen relaciones entre ellas indicadas mediante el nombre de la tabla y la referencia con la que tienen relación. En el siguiente esquema no se han puesto las relaciones para que se muestre más clara la imagen.

