



PROYECTO

FIN DE CICLO

TÍTULO DEL PROYECTO

Nombre: SILVIA MESA COFRADES

DNI: 53598060X

Curso Académico: 2022 / 2024

Profesor tutor del Proyecto : SERGIO MARTOS OLMO.



FPA FORMACIÓN
PROFESIONAL
ANDALUZA

GLOSARIO DE CONTENIDOS

1: INTRODUCCIÓN Descripción del proyecto Objetivos del proyecto	3
¿Qué es GAM?	3
¿Cómo lo consigue?	3
2: ANÁLISIS Requisitos que debe cumplir el sistema propuesto	4
3: DISEÑO UI/UX	5
Diseño UI (Interfaz de Usuario):	5
Diseño UX (Experiencia de Usuario):	7
4: CODIFICACION: Lenguaje de implementación Sistema gestor de BD • Modelo relacional (Tablas, relaciones y restricciones) Herramientas de desarrollo	10
BACK:	10
FRONT:	13
5: MANUALES DE INSTALACIÓN Y USUARIO Instalación y configuración.	13
6: CONCLUSIONES.	13
7: TRABAJO FUTURO.	14
8: BIBLIOGRAFÍA Y REFERENCIAS	14

1: INTRODUCCIÓN Descripción del proyecto Objetivos del proyecto

¿Qué es GAM?

GAM (GUÍAME), nace con el objetivo de aprovechar el aumento del nomadismo puro y del rol de turista, (ver referencia 1 de Bibliografía), para crear conexiones sociales que repercutan en un beneficio para toda la sociedad.

¿Cómo lo consigue?

Actualmente podemos observar que uno de los principales problemas que tiene la sociedad, es el aumento de la soledad, (ver referencia 2 de Bibliografía).

Podemos entender como una consecuencia de tener menos vinculación entre individuos el aumento del nomadismo y el aumento de las tasas de turismo, sobre todo en adultos y adultos jóvenes.

Por otra parte, también encontramos que la desconexión social en individuos con menos recursos económicos o sociales, lo que produce es un recluimiento casero.

Basándome en ésta observación, y entendiendo los recursos que cada rol puede aportar, directamente el uno al otro, nació la idea de GAM.

GAM pretende, conectar a estos dos tipos de personas para que las personas que hacen turismo, puedan expresar al máximo sus experiencias, que realmente es el interés mayor que presentan, pudiendo conectar con individuos autóctonos de cada lugar que visitan, para que sean ellos lo que promocionen y organicen estas experiencias.

A cambio los ciudadanos autóctonos, están obligados a cumplir estrictamente con la descripción de sus experiencias y se aconsejara que los gastos incurrentes corran por parte del turista (entradas de lugares, desplazamientos...etc), pero no obstante,

la aplicación no tiene ningún tipo de interés comercial en este aspecto y se dejará sujeto al acuerdo entre ambas partes.

El ciudadano autóctono de esta manera, consigue un impulso tanto emocional como económico para hacer planes fuera de casa y de esta forma, ambos se verán beneficiados en primera instancia, y como consecuencia final, esto provocará un aumento del turismo en las localidades donde la app esté implantada y un aumento de la economía local, tiendas, bares...etc.

2: ANÁLISIS Requisitos que debe cumplir el sistema propuesto

- 1- Lograr una estructura de los datos, según la lógica de negocio.
- 2- Lograr una persistencia de los datos
- 3- Lograr una interfaz válida para cualquier dispositivo, que sea vistosa y amigable para el usuario y que a su vez, represente de forma conveniente la lógica del negocio.

3: DISEÑO UI/UX

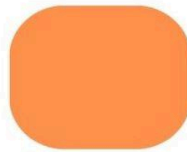
Diseño UI (Interfaz de Usuario):

- Gama de colores: Para los colores, según la psicología de colores, (ver ref 3), voy usar una gama multicolor, con los colores del arcoiris, para el logo y fondo de encabezados. La finalidad es proyectar una imagen solidaria y diversa, donde la fusión de todos los colores en el espacio y modo correcto, quede armonioso y bonito y que así, el usuario perciba, que la diversidad social que le ofrece la app, será algo beneficioso para el. Fomentando su apertura mental y su confianza.
- Logo: El logo será un círculo formado por manos abiertas multicolor.
- Favicon: Será una de las manos.

PALETA



Rojo coral: #ff5757



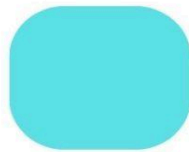
Naranja: #ff914d



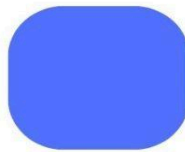
Amarillo: #ffde59



Verde: #c1ff72



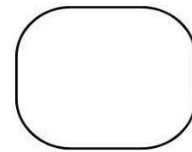
Azul: #5ce1e6



Añil: #5271ff



Morado: #8c52ff



Blanco: #ffffff

LOGO



TIPOGRAFIA: Nerko One Static, regular 400

<style>

@import url('https://fonts.googleapis.com/css2?family=Nerko+One&display=swap')

</style>

Bienvenido a GAM!

Diseño UX (Experiencia de Usuario):

La interfaz de usuario consta de cinco vistas:

- Vista inicio: Vistosa presentación de la aplicación con formulario de login/registro. Única vista en la zona pública.
- Vista muro: Contiene un filtro por código postal, tipo de experiencia y fecha. Y mostrará los resultados de búsqueda. Si no se establece ningún criterio de búsqueda, mostrará todas las experiencias a partir de la fecha actual.
Los resultados de búsqueda, serán unas tarjetas con la información de la experiencia y una parte será la ventana de avatar del usuario con el botón reservar experiencia.
- Publicar Experiencia: Formulario que recoge todos los datos necesarios para añadir una experiencia.
- Detalles de experiencia, mostrará todos los detalles tanto de la experiencia, como del usuario, y si se pulsa, sobre reservar, llevará a la vista de confirmación de reserva
- Datos de contacto y confirmación de reserva: Mostrará una confirmación de la reserva, con un recordatorio de las normas. Tendrá una equis que al pulsarla, volverá al muro.

VISTA INICIO:



VISTA MURO:



VISTA DETALLES:

DETALLES DE EXPERIENCIA

ANFITION@

-FECHA: 25-04-2024 11:30 AM

-COSTE ESTIMADO: 20€ P.P

-PARTIDA: Paza del Ayuntamiento


-TIPO EXPERIENCIA:Cata de vino

-LUGARES :Restarante Los Pinos, Los Pincelines, Taberna "El Vinos".

-NOMBRE: Silvia Mesa Cofrades

-EDAD: 32 AÑOS

-SOBRE MI: Me encanta conocer gente y compartir mi pasión por la enología. Tengo formación reglada sobre ello y me gusta enseñar sobre ello.



RESERVAR

VISTA RESERVA:

DATOS DE CONTACTO



-SILVIA MESA

-TELEFONO: 659-87-98-63

-EMAIL: SMESCOF@GMAIL.ES

NOTA: Es obligatorio avisar al anfitrión si se cancela la cita, para que pueda volver a publicar la experiencia

4: CODIFICACION: Lenguaje de implementación Sistema gestor de BD • Modelo relacional (Tablas, relaciones y restricciones) Herramientas de desarrollo

BACK:

Para implementar la lógica de negocio (back; modelo-controlador) usaré una API-REST en Laravel, que estará conectada con una base de datos relacional MYSQL.

Los Modelos serán los siguientes.

- Usuarios (todo usuario que ingresa en la app)
- Localidades (Localidades donde está implementada la app)
- Experiencias (experiencias que ofrece cada usuario)
- Tipos de experiencia. (tipado de la experiencia, gastronómica, monumental, senderismo...etc).
- Reserva de Experiencia. (datos de dos usuarios, y experiencia)

También deberá satisfacer las siguientes peticiones:

- Todas las experiencias y el usuario asociado a cada una de ellas a partir del día de la consulta. (Sin ningún filtro) .
- Todas las experiencias de una fecha concreta con tipo de experiencia, código postal y el usuario asociado. (Filtrar por fecha , tipo y cp).
- Añadir usuario.
- Consultar usuario.
- Añadir experiencia.
- Añadir localidad.
- Consultar localidad.
- Consultar tipos de experiencia.

Los endpoints finalmente quedarán de la siguiente manera y tendrán el funcionamiento descrito a continuación:

USUARIOS CONTROLLER:

1:___ /consultar-credenciales-usuario GET

Params:

input = dni

input = password

- Si existe en la base de datos un usuario con el DNI y la contraseña que se recibe por parámetro devolverá una respuesta 200 acompañada de los datos completos del usuario. En caso de que no exista la coincidencia devolverá 401.

2:___ /consultar-usuario GET

Params:

input = dni

- Si existe en la base de datos un usuario con el DNI se recibe por parámetro devolverá una respuesta 200 acompañada de los datos completos del usuario. En caso de que no exista la coincidencia devolverá 404.

3:___ /crear-usuario POST

parameters = [

'dni' => 'required|string',

```
'nombre_usuario' => 'required|string',  
  
'password' => 'required|string',  
  
'fecha_nacimiento' => 'required|date',  
  
'sobremi' => 'required|string',  
  
'apellidos' => 'required|string',  
  
'direccion' => 'required|string',  
  
'codigo_postal_usuario' => 'required|string',  
  
'telefono' => 'required|string',  
  
'email' => 'required|email',  
  
]
```

- Si se recibe por parámetro todos los datos necesarios para crear un usuario, el modelo persistirá los datos y devolverá una respuesta 201 acompañada de la id del usuario que se ha persistido. De fallar la validación de los datos y no poderse persistir, el sistema automático de Laravel , devolverá el error, normalmente un 422.

LOCALIDADES CONTROLLER

1__ /insertar-localidad POST

Params:

```
Parameters = [  
  
'codigo_postal' => 'required|string',  
  
'nombre_localidad' => 'required|string',  
  
'provincia_localidad' => 'required|string',
```

```
'pais_localidad' => 'required|string',
```

```
]
```

- Si se recibe por parámetro todos los datos necesarios para crear un usuario, el modelo persistirá los datos y devolverá una respuesta 200 acompañada de la id del usuario que se ha persistido. De fallar la validación de los datos y no poderse persistir, el sistema automático de Laravel , devolverá el error, normalmente un 422.

2__ /consultar-localidades GET

Params:

Input = codigo_postal

- Si existe en la base de datos una localidad con el código postal que se recibe por parámetro devolverá una respuesta 200 acompañada de los datos completos de la localidad. En caso de que no exista la coincidencia devolverá 404.

EXPERIENCIAS CONTROLLER

1__ /todas-las-experiencias-filtro GET

Params:

Input = fecha (yyyy-mm-dd)

Input = tipo_experiencia

Input = codigo_postal

- Si existe en la base de datos una o más experiencias de estado activa que cumplan todos los criterios de búsqueda, las devuelve con una respuesta 200, si no se envía ningún parámetro, se devuelven todas las experiencias. Si no se encuentra, es laravel quien se encarga de mandar un error, normalmente 422.

2__ /cambiar-estado-inactivo GET

Params:

Input = id_experiencia

- Si existe en la base de datos una experiencia que cumpla todos los criterios de búsqueda, modifica el estado a reservada y devuelve 200, si no se envía ningún parámetro, se devuelven todas las experiencias. Si no se encuentra, es laravel quien se encarga de mandar un error, normalmente 422.

3__ /insertar-experiencia POST

Params:

```
Parameters = [  
  
    'titulo_experiencia' => 'required|string',  
  
    'descripcion_experiencia' => 'required|string',  
  
    'lugar_partida' => 'required|string',  
  
    'coste_estimado' => 'required|numeric',  
  
    'fecha_experiencia' => 'required|date',  
  
    'tipo_experiencia' => 'required|integer',
```

```
'codigo_postal_experiencia' => 'required|string',  
  
'dni_proveedor' => 'required|string'  
  
]
```

- Si se reciben correctamente todos los parámetros necesarios para persistir una experiencia, el modelo lo hará y devolverá 201 con la id de la experiencia.

4__ /consultar-experiencia GET

Params:

Input = id_experiencia

- Si existe una experiencia que se corresponda con la id que se envía por parámetro, devolverá todos los datos de la experiencia con una respuesta 200.

TIPO DE EXPERIENCIAS CONTROLLER

1__ /consultar-tipos-experiencias GET

Params:

Input = id_tipo_experiencia

- Si existe un tipo de experiencia que coincida con el dato que se recibe por parámetro, se devuelve una respuesta 200 con todos los datos de un tipo de experiencia.

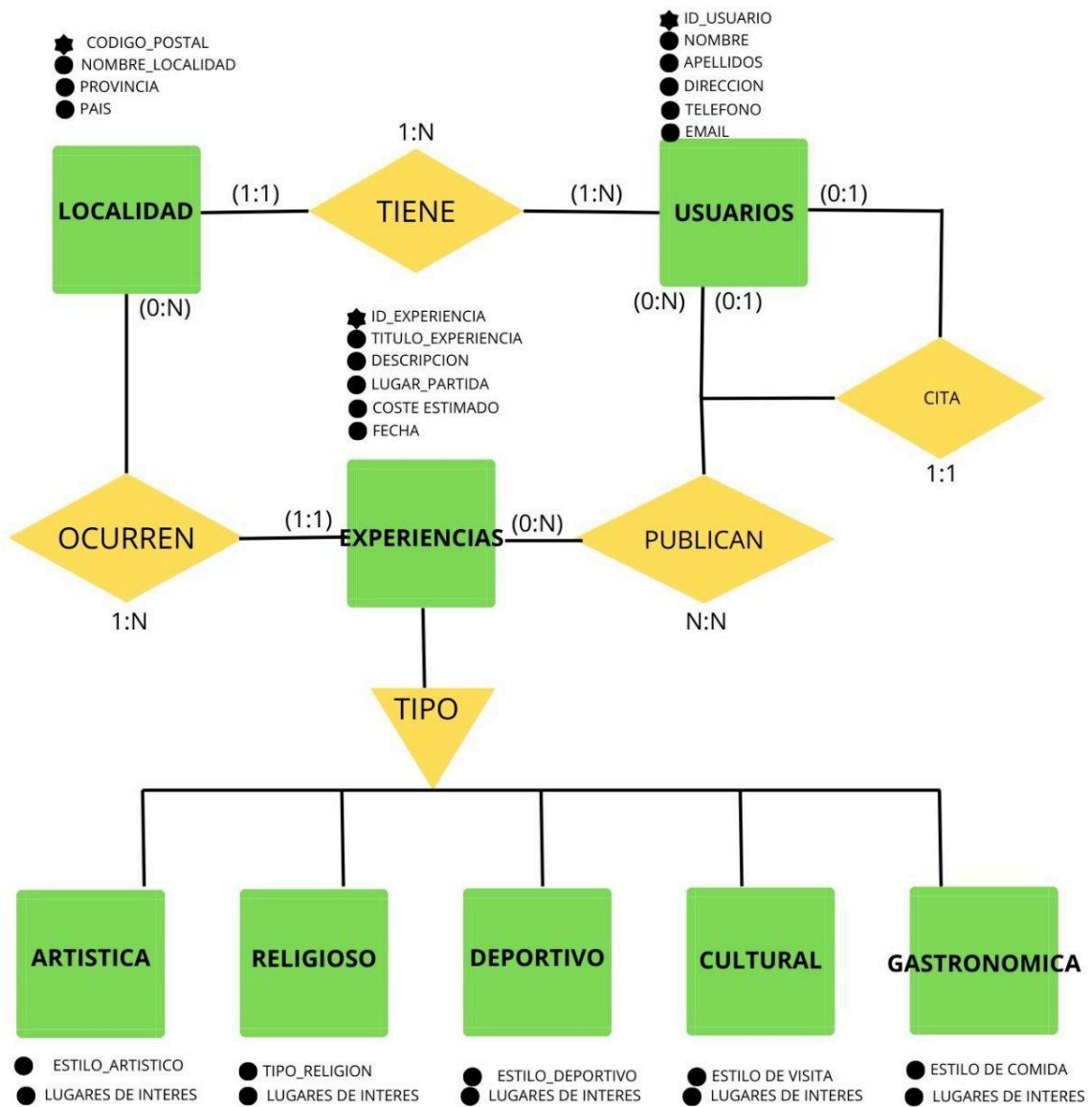
CITAS DE USUARIO CONTROLLER

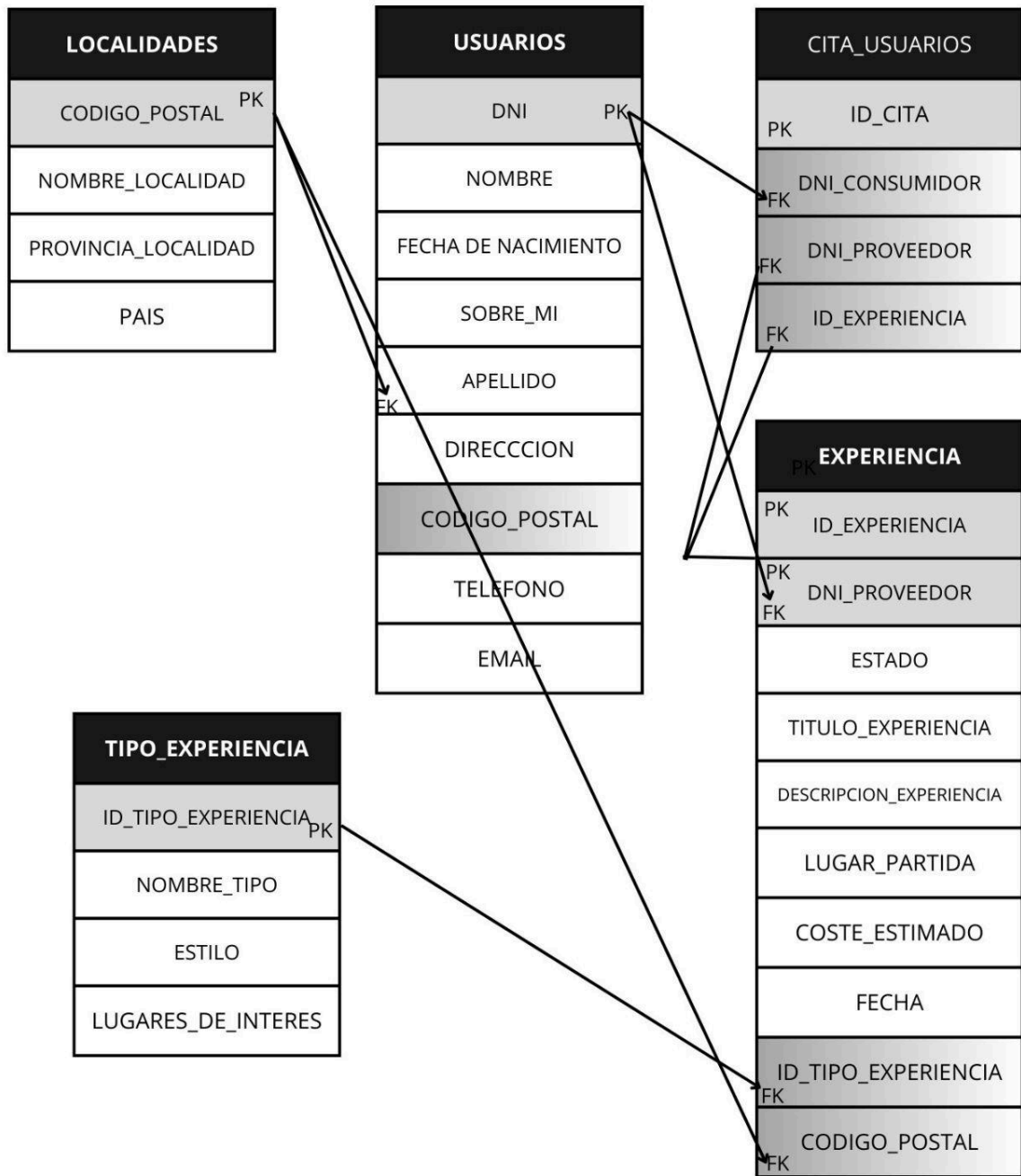
1__ /crear-cita POST

Params:

```
Parameters = [  
  
    'dni_consumidor' => 'required|string',  
  
    'dni_proveedor' => 'required|string',  
  
    'id_experiencia' => 'required|integer',  
  
    ]
```

- El modelo recibirá por parámetro los datos necesarios y persistirá en la base de datos el registro de una cita. Devolverá una respuesta 201, de no recibirse los datos, se devolverá 422.





FRONT:

Para el front (vista), realizaré un cliente a la API en Vue. Éste estará formado por cuatro vistas unidas con RouterLink, y usando sus propiedades de autenticación para controlar que no se acceda a visitas privadas sin estar registrado, pudiendo provocar una grieta de seguridad a los datos.

En resumen las tecnologías usadas serán:

- Php, (Laravel)
- Sql.(Consultas)
- Javascript (Vue),(para el desarrollo de la vista)
- JQuery (peticiones)
- Json (respuestas)
- Html (representación de datos)
- Css (para los estilos)

5: MANUALES DE INSTALACIÓN Y USUARIO Instalación y configuración.

El proyecto está realizado con la intención de que los usuarios puedan acceder a la aplicación a través del navegador una vez que esté desplegada en un servidor, no obstante, se puede poner en marcha en un entorno local de la siguiente manera:

Para la API:

- Instalar Laragon(ver referencia 4).
- Alojar el proyecto en la carpeta www de Laragon.
- Para servir la aplicación localmente ejecutar el comando “php artisan serve”.

Para la interfaz:

- Instalar Vue. (Ver referencia 5)
- Instala las dependencias con el comando “npm install”.
- Ejecuta el proyecto con el comando “npm run dev”.

6: CONCLUSIONES.

En conclusión, la duración del proyecto es de unas 40 horas productivas.

La dificultad la estimo en media - alta, y me he enfrentado a algunos retos como, hacer Api Rest en laravel o implementar seguridad en las rutas de Vue.

Generalmente, el planteamiento que hice en el inicio ha funcionado correctamente y no he tenido que hacer prácticamente ninguna modificación que se haya detectado como necesaria en la fase de desarrollo.

Con el resultado final, estoy generalmente satisfecha atendiendo a los objetivos iniciales, ya que éstos, han sido conseguidos. Soy consciente que la aplicación se podría haber desarrollado con más complejidad y detalle, pero eso conlleva más modelos, más relaciones y en definitiva más tiempo. Creo que he materializado bien una idea propia, el resultado es bastante representativo del funcionamiento y está amoldado al tiempo de proyecto que son 40h. Quedando como digo, muchísimas mejoras o ampliaciones para un futuro.

7: TRABAJO FUTURO.

En un futuro la aplicación aún tendrá mucho margen de trabajo, bien en la mejora de la propia aplicación, como por ejemplo:

- Crear cliente administrador.
- Crear todos los CRUD de todas las tablas.
- Añadir fotografías al proyecto.
- Añadir administración del perfil.
- Añadir servicio de mensajes dentro de la aplicación.
- Etc..

Por otra parte también se puede trabajar en aspectos de comercialización y monetización de la app, como por ejemplo:

- Pensar en la mejor forma de monetización
 - Venta de la app a entidades de turismo públicas o privadas.
 - Incluir anuncios.
 - Añadir pago por uso.
- Implementarla (si es conveniente) en la app.

8: BIBLIOGRAFÍA Y REFERENCIAS

1: <https://pumble.com/learn/es/digital-nomad-visa/statistics/>

2: <https://www.cdc.gov/aging/spanish/features/lonely-older-adults.html#:~:text=La%20soledad%20significa%20sentirse%20solo,solas%20sin%20estar%20socialmente%20aisladas.>

3:

<https://blog.3tcomunicacion.com/2018/02/el-significado-de-los-colores-colores-solidarios/>

4: <https://styde.net/laragon-un-entorno-de-desarrollo-para-laravel-en-windows/>

5:

<https://learn.microsoft.com/es-es/windows/dev-environment/javascript/vue-on-windows>