

GTV

16/12/2022

—

Miguel Moreno Such
Carlos Molina Alcolea
César Verdú Motos

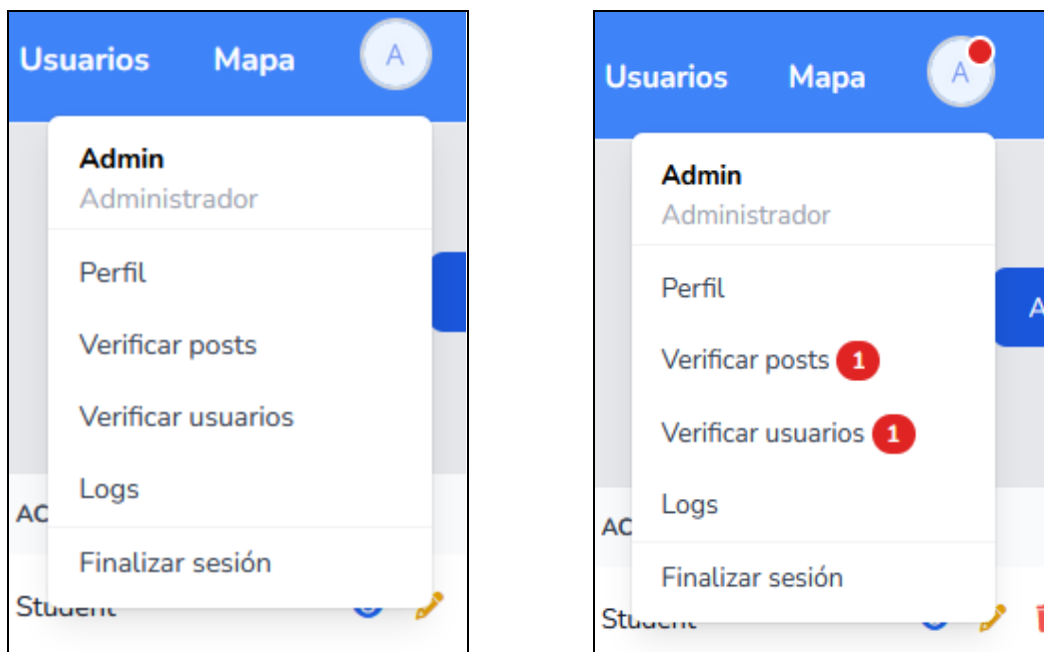
Índice

Verificación de los posts creados por los alumnos	2
Verificación de usuarios nuevos	4
Borrado de posts y usuarios sin verificar	5
Mapa de puntos de interés	7
Log	9
Refactorización de la base de datos	10
Normalización	10
Añadido, borrado y desplazamiento de columnas	10
Validación de tipos de datos	10
Refactorización de los factories	10
Eliminación de modelos relacionados y soft deletes	10
Refactorización de las vistas	11
Búsqueda, ordenación y fechas	11
Vista detalles	11
Segmentación columna en tablas	12
Reparación de subida de vídeos y fotografías	13
Tests automáticos con Github Actions	14
Tests con dusk	14
Despliegue con Laravel Forge	15
Creación de formulario de Registro	16
Limpieza y armonización de código	17
Actualización a Laravel 9.44	17

Verificación de los posts creados por los alumnos

Se ha implementado esta nueva funcionalidad para que profesores y administradores puedan verificar los posts de los alumnos antes de que sean visibles en la página. Lo primero que hemos hecho ha sido añadir la columna *verified* a las tablas puntos de interés, lugares, fotos y vídeos con el valor predeterminado en *false* para, posteriormente, filtrar todos los elementos que llegan a la página de dichas tablas y mostrar sólo los posts que contengan el campo *verified* igualado a *true*(posts verificados).

Después hemos creado un controlador de Livewire con su respectiva vista para sustituir el icono del usuario registrado por un icono reactivo que avise al usuario de que tiene notificaciones sin revisar. Dentro del desplegable del usuario hemos añadido unos contadores donde se muestra la cantidad de posts y alumnos que hay por verificar:



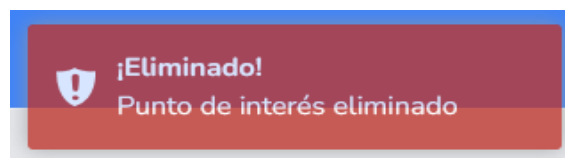
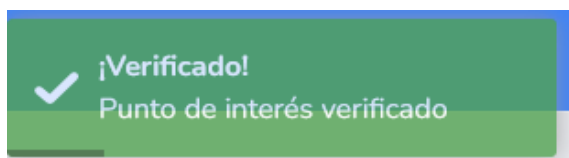
En la vista de verificación de los posts podemos encontrarnos algo como esto:

Elementos por verificar:

Selecciona un filtro **ELIMINAR FILTROS**


PUNTOS DE INTERÉS					
ID	NOMBRE	DESCRIPCIÓN	LUGAR	CREADOR	ACTUALIZADOR
3	Lebsacktown	Repudiandae at quasi velit est aut. Consequuntur n...	Candidashire	Teacher	Admin

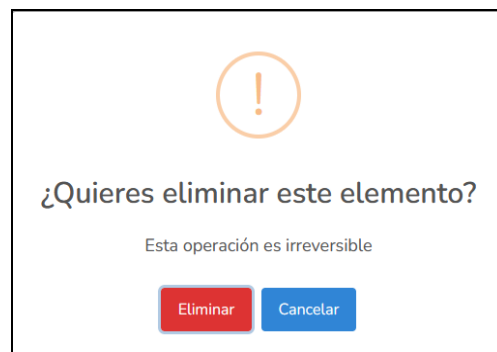
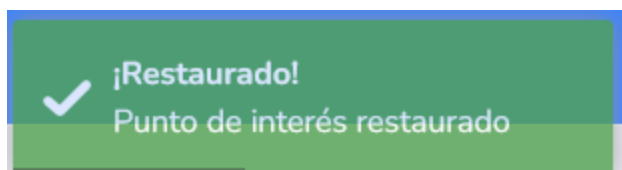
Aquí podemos filtrar los posts que hay por verificar por tipos y también ver los detalles del propio post que esté por verificar. Si verificamos, el elemento se enviará a su respectiva vista con el resto de elementos verificados, y si decidimos no verificar el post, éste será enviado a la papelera, en cualquier caso recibiremos una notificación temporal creada con *toastr*



Si accedemos a la papelera podemos ver el post que hemos enviado ahí con unos iconos de acción distintos, si decidimos restaurarlo, volverá a la vista de elementos por verificar, y si decidimos eliminarlo se nos mostrará un diálogo creado con *SweetAlert* que nos pedirá confirmar la acción.

Elementos no verificados:

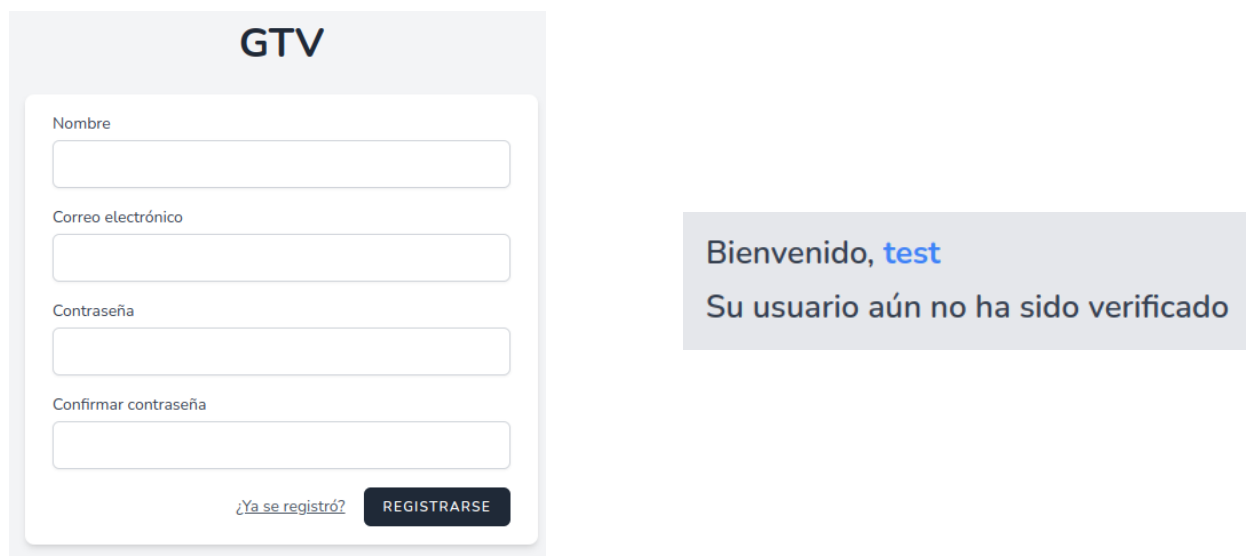
PUNTOS DE INTERÉS							
QR	ID	NOMBRE	LATITUD / LONGITUD	LUGAR	CREADOR	ACTUALIZADOR	FECHA CREACIÓN
	1	Lake Chadchester	-15.5387170000 / 98.2657700000	Paulinemouth	Admin	Admin	2022-12-15 20:59:01



Verificación de usuarios nuevos

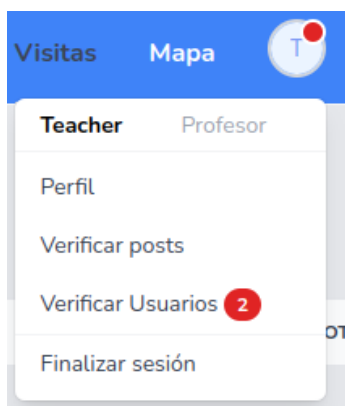
Hemos habilitado el registro de nuevos usuarios para que los alumnos puedan crearse una cuenta en la aplicación. Al registrarse por primera vez los alumnos verán un mensaje informándoles de que no han sido verificados y no podrán acceder al resto de la aplicación.

Los profesores y administradores podrán verificar a los alumnos. Una vez verificados, los alumnos tendrán acceso al resto de la aplicación.



The image shows a registration form for 'GTV' with the following fields: 'Nombre', 'Correo electrónico', 'Contraseña', and 'Confirmar contraseña'. Below the fields is a link '¿Ya se registró?' and a 'REGISTRARSE' button. To the right, a grey box displays the message: 'Bienvenido, test' and 'Su usuario aún no ha sido verificado'.







Para verificar a los alumnos, los profesores deben hacer click en su foto de perfil y mostrar el menú desplegable. En este menú se muestra el número de alumnos pendientes por verificar. Si hay alumnos pendientes por verificar también se muestra un círculo rojo en la foto de perfil.




En la vista para verificar alumnos los profesores pueden ver los detalles del alumno antes de hacer la verificación. Una vez comprobados los detalles pueden verificar al alumno haciendo click en el botón verde, esto mostrará una ventana de confirmación con SweetAlert.

Verificación de usuarios

NOMBRE ELIMINAR FILTROS

	ID ↓	NOMBRE	EMAIL	FECHA CREACIÓN	FECHA ACTUALIZACIÓN	
	5	test	test@gmail.com	2022-12-15 17:56:06	2022-12-15 17:56:06	 
	4	Student2	student2@mail.com	2022-12-15 16:57:09	2022-12-15 16:57:09	 

Detalles del usuario #4



Nombre: Student2


Email: student2@mail.com

Rol: Usuario sin verificar

Fecha de creación: 2022-12-15 16:57:09

Fecha de actualización: 2022-12-15 16:57:09

[Cerrar](#)





¿Quieres verificar este alumno?

[Verificar](#) [Cancelar](#)

Borrado de posts y usuarios sin verificar

Se ha implementado una nueva función para el control del material borrado o los usuarios no verificados. Todo el material que un profesor no haya permitido su subida a la aplicación será enviado a la papelera de reciclaje. Una vez allí los profesores, una vez hayan accedido a la misma clicando en el icono de la papelera pueden eliminar definitivamente el material clicando en el icono rojo o bien restaurar el elemento clicando el icono verde.

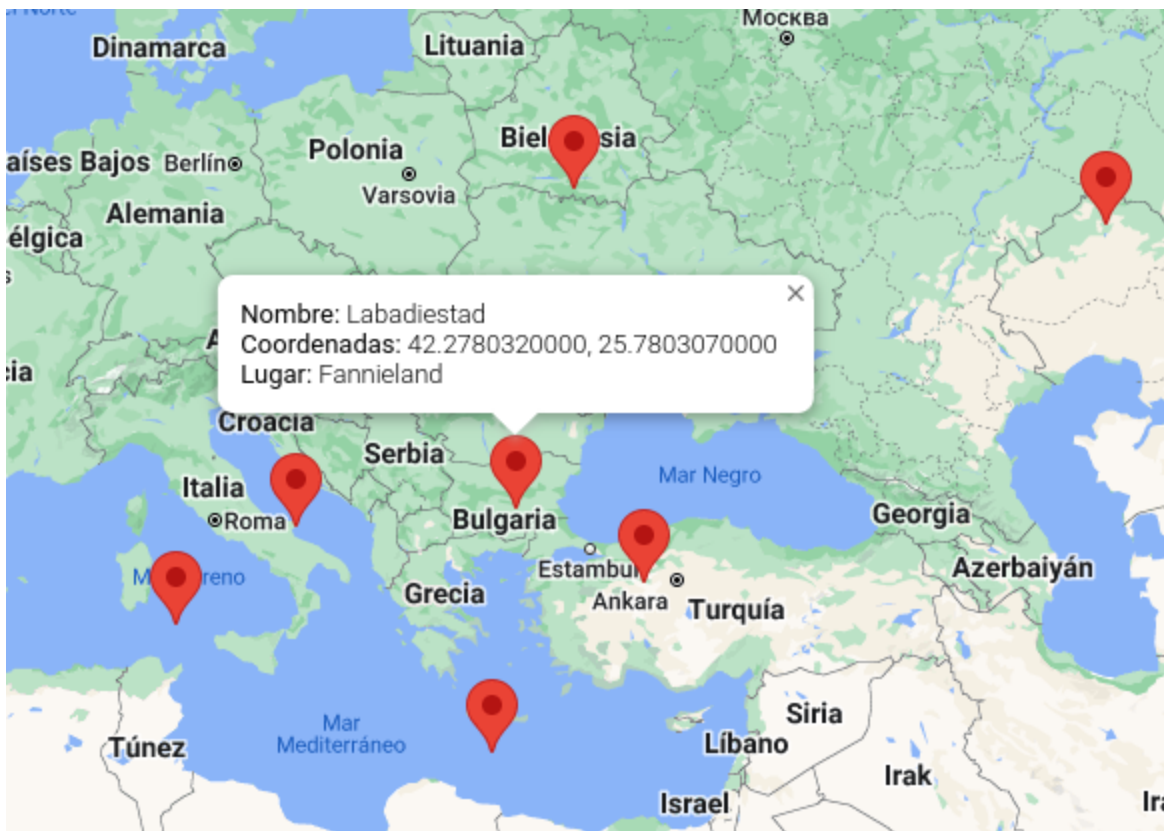
Elementos no verificados: [Volver](#)

PUNTOS DE INTERÉS							
QR	ID	NOMBRE	LATITUD / LONGITUD	LUGAR	CREADOR	ACTUALIZADOR	FECHA CREACIÓN
	101	asease	3.0000000000 / 3.0000000000	West Jude	Student	Teacher	2022-12-16 10:56:05
	102	cedssa	4.0000000000 / 3.0000000000	West Jude	Student	Ninguno	2022-12-16 11:25:11

Por otra parte, si un usuario no ha sido verificado por un profesor o un administrador en un plazo de 7 días desde que se registró en la aplicación, automáticamente se eliminará ese registro y el usuario deberá reiniciar el proceso de registro de nuevo.

Mapa de puntos de interés

Se ha creado una vista nueva con un mapa de Google que muestra los puntos de interés. El mapa tiene un marcador por cada punto de interés. Al hacer click en un marcador se muestra un mensaje con información del punto de interés. Este mensaje desaparece al hacer click en otro sitio.



En la vista del mapa también se ha incluido la lista de todos los puntos de interés. Cada punto tiene un checkbox para mostrar u ocultar el marcador de ese punto. También permite filtrar los puntos que se muestran en el mapa. Se pueden filtrar por áreas temáticas, lugares y nombre. El mapa es dinámico por lo que los filtros se aplican instantáneamente sin necesidad de recargar la web.

Fannieland	neque	Buscar punto de interés...
<input type="checkbox"/> SELECCIONAR TODOS	ID ↑	NOMBRE
<input type="checkbox"/>	12	Isobelside
<input checked="" type="checkbox"/>	23	Port Reynoldhaven
<input checked="" type="checkbox"/>	97	North Javonte

El mapa es un componente livewire reusable lo que nos permite añadir otros mapas a la aplicación muy fácilmente:

Detalles del Punto de Interés #91

Nombre: East Mariloufort

Descripción: Fugiat aut commodi fugiat eos sed incididunt. Et eius dicta quo sequi. Velit est nostrum non ad eveniet nobis sed saepe. Sit dolorem dicta molestiae dolorem necessitatibus.

Latitud: 44.4427700000

Longitud: 108.1907170000

Lugar: North Eldridgestad (6)

Área/s Temática/s: quis (1), corporis (2), veritatis (4),

Creador: Student (3)

Actualizador: Student (3)

Fecha de creación: 2022-12-15 16:57:10


Fecha de actualización: 2022-12-15 16:57:10

QR y mapa:



Mapa

Satélite



Combinaciones de teclas

Datos de mapas ©2022

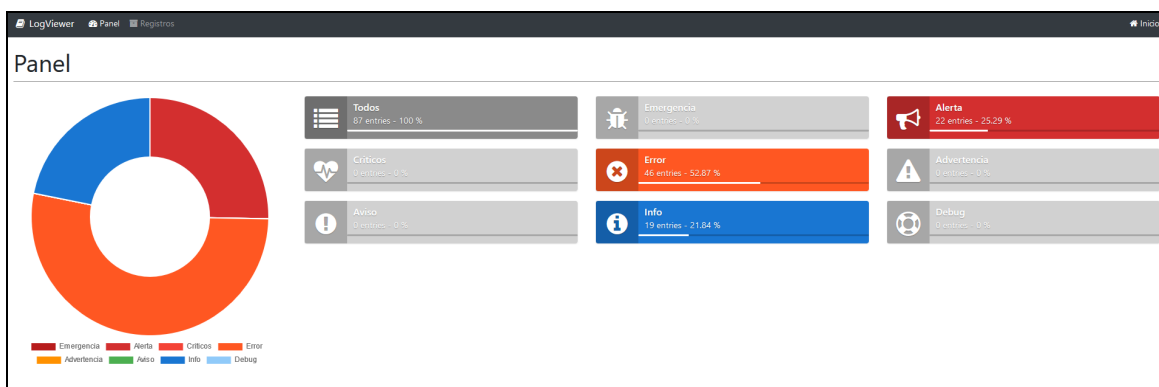
Términos de uso

Cerrar

Log

Los administradores pueden visitar la página del Log de aplicación clicando en el su imagen de usuario y seleccionando la opción Logs.

Allí hemos extendido la dimensión del log de la aplicación a la creación de elementos y los errores de la aplicación en los diferentes entornos (local, test y producción) además de los de edición y eliminación de elementos ya existentes.



local	Info	21:47:16	User with ID 6 was updated a point of interest with name Adamouth	Context
local	Alerta	21:41:56	User with ID 1 verified a student	Context
local	Alerta	21:14:06	User with ID 1 verified a student	Context
local	Alerta	18:38:16	User with ID 1 verified a student	Context
production	Error	13:08:44	htmlspecialchars(): Argument #1 (\$string) must be of type string, array given ("view":{"userid":1,"exception":{"object":{"Spatie\LaravelIgnition\Exceptions\ViewException(code: 0): htmlspecialchars(): Argument #1 (\$string) must be of type string, array given at /home/forge/gtvtabajofinalgrado.xyz/vendor/laravel/framework/src/Illuminate/Support/helpers.php:123})"}}	Stack Context
production	Error	13:08:36	htmlspecialchars(): Argument #1 (\$string) must be of type string, array given ("view":{"userid":1,"exception":{"object":{"Spatie\LaravelIgnition\Exceptions\ViewException(code: 0): htmlspecialchars(): Argument #1 (\$string) must be of type string, array given at /home/forge/gtvtabajofinalgrado.xyz/vendor/laravel/framework/src/Illuminate/Support/helpers.php:123})"}}	Stack Context

Fecha	Idios	Emergencia	Alerta	Críticos	Error	Advertencia	Aviso	Info	Debug	Acciones
2022-12-16	3	0	1	0	0	0	0	0	0	🔍 📄 🗑️
2022-12-15	22	0	0	0	11	0	0	0	0	🔍 📄 🗑️
2022-12-14	23	0	0	0	21	0	0	0	0	🔍 📄 🗑️
2022-12-13	10	0	0	0	0	0	0	1	0	🔍 📄 🗑️

Admin
Administrador

Perfil

Verificar posts

Verificar usuarios 1

Logs

Finalizar sesión

Refactorización de la base de datos

Se han realizado varias mejoras en la base de datos. El objetivo de estas mejoras es minimizar la redundancia de datos, disminuir problemas de actualización de datos y proteger la integridad de datos.

Normalización

Inicialmente la base de datos no estaba normalizada, tenía dependencias funcionales y transitivas en columnas que no eran claves. Debido a estos problemas era violar la integridad de la base de datos de forma que algunas columnas tenían valores que no deberían ser posibles.

La normalización de la base de datos soluciona estos problemas ya que una BBDD completamente normalizada evita este tipo de violaciones de integridad por diseño.

Añadido, borrado y desplazamiento de columnas

Se han añadido columnas nuevas para almacenar nueva información en los modelos. Algunas columnas se han cambiado de tabla porque no estaban bien ubicadas.


Se han borrado varias columnas principalmente porque eran columnas que almacenaban información innecesaria y que no se usaba en ningún sitio de la aplicación. También se han borrado columnas duplicadas ya que había tablas que tenían varias columnas para almacenar la misma información y esto viola la integridad de la base de datos.

Validación de tipos de datos

Al insertar datos nuevos en la BBDD se debe validar que el tipo de datos sea correcto para que no de un error. Estas validaciones estaban incompletas por lo que las hemos terminado de implementar.

Refactorización de los factories

Inicialmente el proyecto contenía factories en el formato de Laravel 7 y también factories en el nuevo formato de Laravel 8. Algunos modelos tenían un factory en



cada uno de estos formatos y se usaban los dos. Hemos refactorizado los factories para actualizarlos al nuevo formato de Laravel 8 y para que cada modelo tenga un único factory.

Eliminación de modelos relacionados y soft deletes

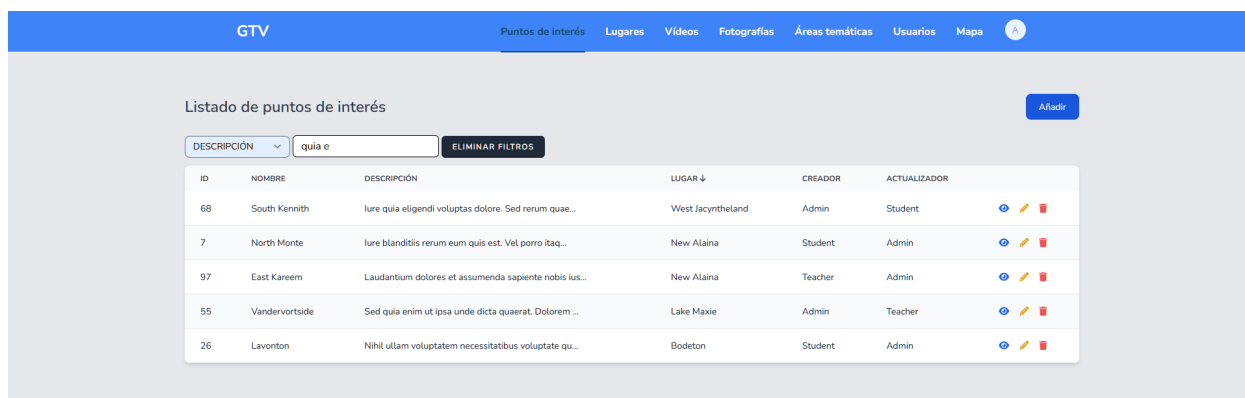
Inicialmente no se controlaba el comportamiento del borrado de los modelos. En ocasiones borrar un modelo rompía las vistas de los modelos relacionados por clave foránea. En otras ocasiones al borrar un modelo se borraban otros modelos que no deberían ser borrados. Por ejemplo, si se borraba un usuario se perdían todos los puntos de interés, áreas temáticas, vídeos, fotografías y lugares que el usuario había actualizado, aunque no las hubiese creado ese usuario.

También hemos incluido soft deletes para los modelos Photography, Place, PointOfInterest, ThematicArea y Video.

Refactorización de las vistas

Búsqueda y ordenación

Añadida búsqueda por nombre y descripción, ordenación de lugar en la tabla puntos de interés, desplazadas fechas de todas las tablas a la vista de detalle para hacerlas mas amigables. Añadida la opción de ordenar por lugar.



GTV

Puntos de interés Lugares Videos Fotografías Áreas temáticas Usuarios Mapa

Listado de puntos de interés [Añadir](#)

DESCRIPCIÓN [ELIMINAR FILTROS](#)

ID	NOMBRE	DESCRIPCIÓN	LUGAR ↓	CREADOR	ACTUALIZADOR	
68	South Kenneth	lure quia eligendi voluptas dolore. Sed rerum quae...	West Jacyntheland	Admin	Student	📍 ✎ 🗑
7	North Monte	lure blanditis rerum eum quis est. Vel porro itaq...	New Alaina	Student	Admin	📍 ✎ 🗑
97	East Kareem	Laudantium dolores et assumenda sapiente nobis ius...	New Alaina	Teacher	Admin	📍 ✎ 🗑
55	Vandervortside	Sed quia enim ut ipsa unde dicta quaerat. Dolorem ...	Lake Maxie	Admin	Teacher	📍 ✎ 🗑
26	Lavonton	Nihil ullam voluptatem necessitatibus voluptate qu...	Bodeton	Student	Admin	📍 ✎ 🗑

Vista detalles

Añadidas las vistas de detalles en las tablas puntos de interés del menú y vista de bienvenida. Refactorización de los QR para que muestren las coordenadas en un mapa y añadido un mapa con el punto.

Detalles del Punto de Interés #101

Nombre: AT&T Stadium

Descripción: El AT&T Stadium es un recinto deportivo ubicado en Arlington, Texas, Estados Unidos. Alberga los partidos que disputan como locales los Dallas Cowboys de la National Football League (NFL) y tiene una capacidad para 80.000 espectadores

Latitud: 90.0000000000

Longitud: 32.0700000000

Lugar: Lake Maxie (10)

Área/s Temática/s:


Creador: Alejandro Martínez (7)

Actualizador: Admin (1)


Fecha de creación: 2022-12-16 09:36:20

Fecha de actualización: 2022-12-16 09:41:16

QR y mapa:



Mapa Satélite



+

-

Google Combinaciones de teclas Datos de mapas ©2022 Términos de uso

[Cerrar](#)

Segmentación columna en tablas

Añadida una segmentación de la columna descripción en todas las tablas para mostrar un resumen a partir de 50 caracteres para conservar el diseño de las mismas.

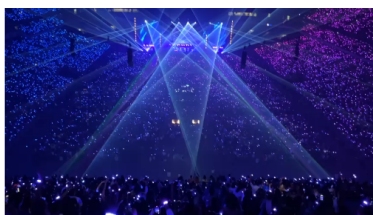
Accusamus expedita soluta voluptatem.

Autem tempora voluptas omnis nemo necessitatibus v...

Reparación fotografías y vídeos

Reparación de la subida de vídeos y fotos en detalles y tablas

Detalles del vídeo #202



Descripción: Concierto de Bad Bunny en San Antonio. Prueba de la respuesta del sistema de sonido en el punto más alejado de la audiencia.

Ruta: /storage/videos/V5o2Y0rGoPUgEqkFdhXSaa7cvsSAPzGW7OD8D.mp4

Orden: 1

Punto de interés: 101

Creador: Alejandro Martínez (7)


Actualizador: ninguno

Fecha de creación: 2022-12-16 10:14:28

Fecha de actualización: 2022-12-16 10:14:28

Cerrar

Detalle de la fotografía #203















Ruta: storage/photos/c2UuNTTCTCXb4hVWyXgxaMJFRXE0AJL-metaSU1HXzlwMjllwOTA3XzlyMzAxNy5qcGc=-.jpg

Orden: 2

Punto de interés: 101

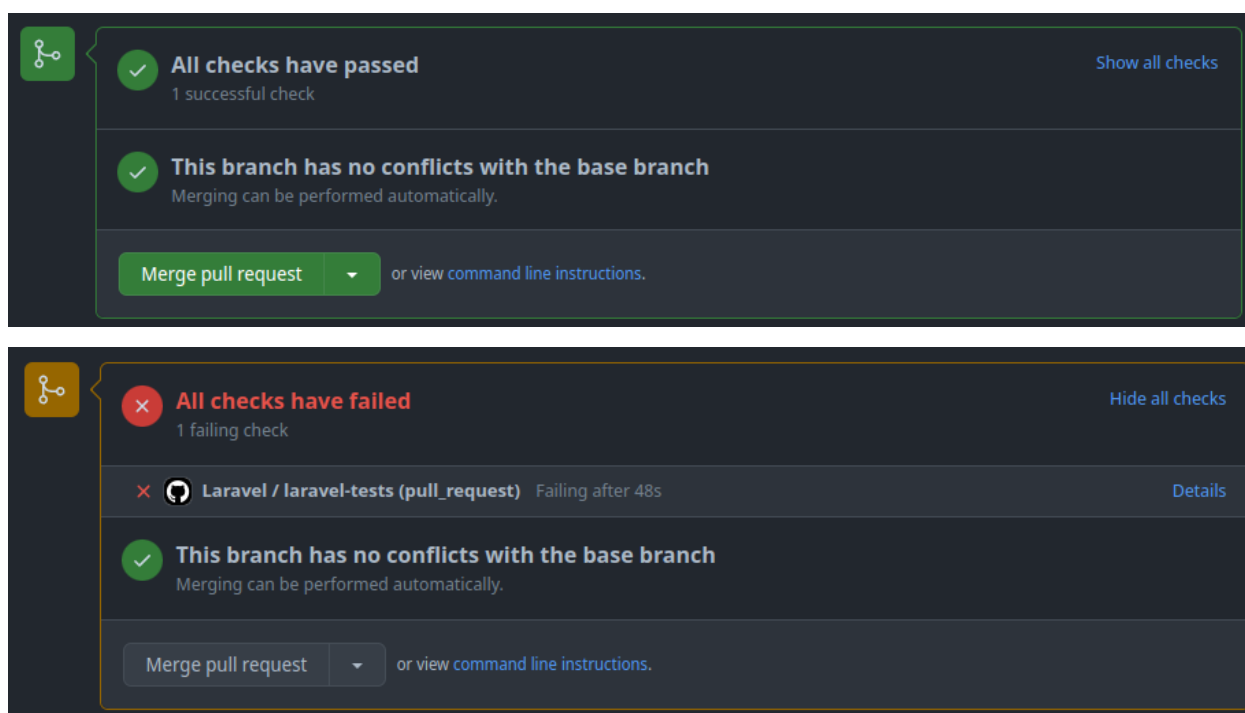
Fecha de creación: 2022-12-16 09:50:39

Cerrar

ID	RUTA	ORDEN	PUNTO DE INTERÉS	CREADOR	ACTUALIZADOR	ACCIONES
203		2	101	Alejandro Martínez (ID: 7)		  
202		1	1	Eduardo Tomas Navarro (ID: 8)		  
201		1	101	Alejandro Martínez (ID: 7)		  

Tests automáticos con Github Actions

GitHub Actions es una plataforma de integración continua y despliegue continuo (CI/CD) que nos permite automatizar la compilación, las pruebas y el despliegue de nuestra aplicación. Nosotros hemos usado Github Actions para automatizar la ejecución de los tests. Los tests se ejecutan cada vez que se hace una pull request y cada vez que se hace un merge a la rama principal:



Tests con dusk

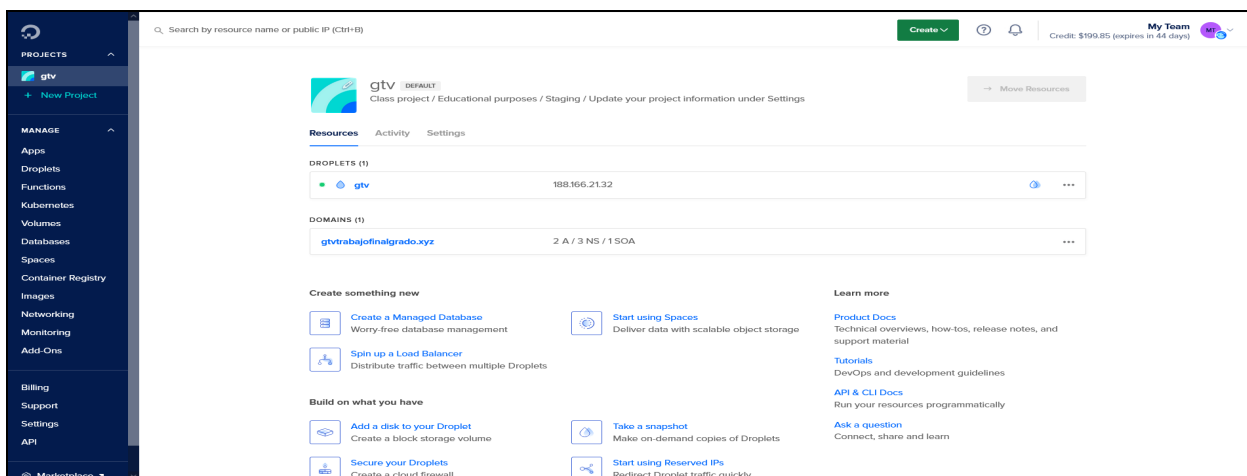
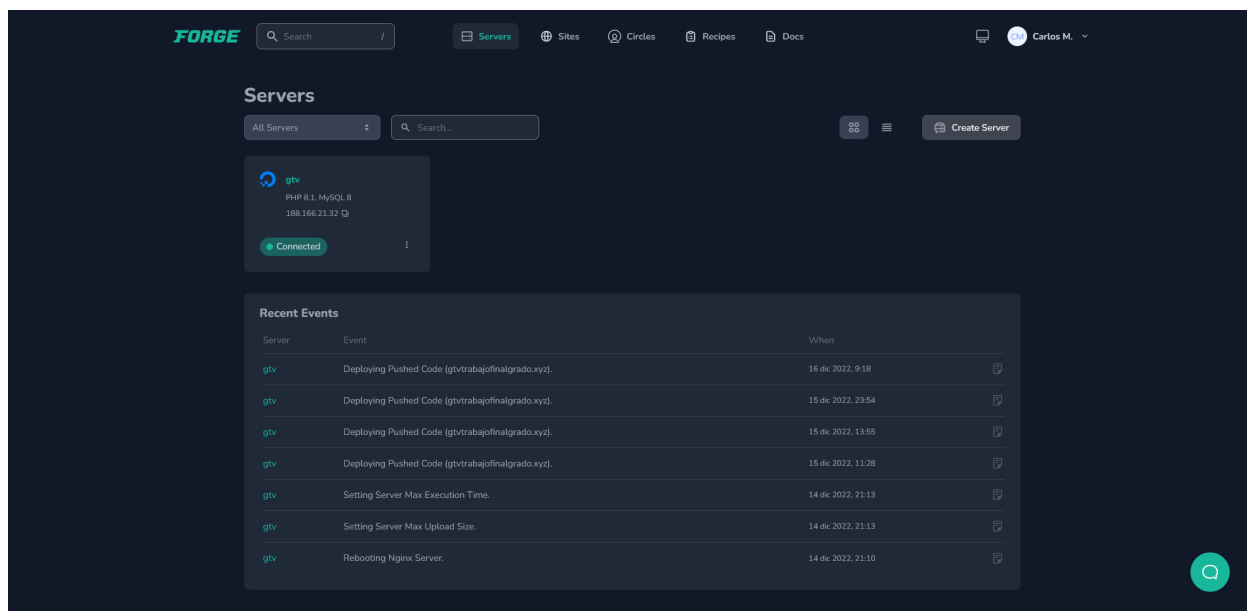
Hemos usado dusk porque hay ciertas cosas que no se pueden testear sin dusk. Un ejemplo es cualquier test relacionado con el mapa ya que el mapa se inserta en la aplicación usando un script de javascript de Google, por lo que es necesario un navegador que ejecute javascript.

Despliegue con Laravel Forge

Para el despliegue de la aplicación hemos utilizado el servicio *Laravel Forge*, que automatiza tanto el despliegue como la posterior gestión de éste. Desde su página web ofrece un gran abanico de herramientas, una vez hayas desplegado tu servidor.

También nos decantamos por este servicio porque algunas de sus herramientas nos parecían muy interesantes y facilitaban bastante las cosas, como por ejemplo la funcionalidad *Push to deploy*, que consiste en desplegar tu aplicación apuntando a su repositorio en github, y cada vez que Forge detecte que se ha hecho un push a la rama predeterminada (normalmente suele ser *master* o *main*) éste se encarga de hacer un pull en el servidor y desplegar nuevamente la aplicación.

Otro aspecto importante de Forge es que es compatible con los Digital Ocean, donde se aloja nuestro *droplet*.



Actualización a Laravel 9.44

El proyecto inicialmente estaba en la versión 9.12 de Laravel. Hemos actualizado a Laravel 9.44.0, la última versión con fecha 14/12/22. La actualización causaba problemas con algunas dependencias pero se han podido solucionar. Todas las dependencias del proyecto también han sido actualizadas a la versión más reciente.

Limpieza y armonización de código

Se ha realizado una limpieza del código eliminando varias clases como OldPointsOfInterest y eliminando por completo la sección visitas que no aportaba nada al proyecto y no realizaba función ninguna. También se ha creado la consulta para la ordenación de lugares en puntos de interés con Eloquent a fin de ayudar a la compresión de código.

```
public function render()
{
    if ($this->sortField === 'place_id') {
        $points = PointOfInterest::select('point_of_interests.*')
            ->join('places', 'places.id', '=', 'point_of_interests.place_id')
            ->where('point_of_interests.' . $this->searchColumn, 'like', '%' . $this->search . '%')
            ->where('point_of_interests.verified', true)
            ->orderBy('places.name', $this->sortDirection)
            ->paginate(10);
    } else {
        $points = PointOfInterest::where($this->searchColumn, 'like', '%' . $this->search . '%')
            ->where('verified', true)
            ->orderBy($this->sortField, $this->sortDirection)
            ->paginate(10);
    }
}
```