



**C.P.R. Liceo “La Paz”**

**Proyecto Fin de Ciclo**

**Desarrollo de Aplicaciones Multiplataforma**

---

**Autor : Javier Debén Chacón**

**Tutor : Jesús Ángel Pérez-Roca Fernández**





### Resumen.

El proyecto proporciona un gestor de automóviles el cual permite introducir y registrar los datos de personas y vehículos asociados a dichas personas. Se tendrán datos concretos relativos tanto a las personas como a los vehículos, y así mismo se almacenarán datos correspondientes a los gastos asociados a cada uno de estos vehículos. Desde la aplicación se tendrán por tanto dos formas principales de acceder a los datos, bien a través de un buscador de personas accediendo de esta forma a los vehículos de los que sea propietario o bien a través de un buscador de marca y modelo, mostrándose de esta forma los coches que cumplan esta selección e indicándose a quien pertenece.

Una vez se tiene seleccionado un vehículo se podrá visualizar los datos básicos de éste (color, matrícula, etc.) así como los gastos que para él tengamos registrados. De la misma forma se podrá acceder a cada uno de estos gastos, recuperar información sobre ellos como por ejemplo: kilometraje en el que se produjo, fecha, descripción, etc. o también litros y precio por litro en el caso de que se trate de un repostaje. Estos gastos podrán ser a modo indicativo los citados repostajes, ITVs, recibos del seguro, reparaciones en taller o intervenciones de mantenimiento, o cualquier otro que el usuario desee registrar para que posteriormente pueda ser tratado o analizado de forma estadística.

Todos estos datos que se han mencionado podrán ser añadidos, modificados, consultados o borrados a través de la aplicación.

### Abstract.

This project provides us with a car manager app which allows to enter and record the data of vehicles and its owners alike. Furthermore, the data related to the expenses of these vehicles is stored. There are two main ways of access, either by a client searchtool (showing the owned vehicles) or using the brand and model search engine (in this case, showing the cars that meet the criteria selected).

Once a vehicle is selected, it will show basic data about itself (colour, license plate...) as well as any other information registered. We can access to any of these expenses correspondingly, plus recover information about them (kilometers, date, description, liters consumed...). These expenses may be indicative of the aforementioned refueling, ITVs, insurance receipts, workshop repairs or maintenance interventions, or any other that the user wishes to record so that it can later be treated or analyzed statistically. All these data that we have mentioned can be added, modified, consulted or deleted through the application.

## Palabras Clave.

Tecnología	Ámbito	Explicación
HTML	Lenguaje de marcas	Utilizado para definir la estructura y los contenidos visualizados en el navegador web por el usuario. Con este lenguaje se define como se estructuran sin influir en su diseño lo cual actualmente se realiza mediante la siguiente tecnología (CSS).
CSS	Lenguaje de estilos	Usado para darle formato visual a los contenidos mostrados en el navegador. Se entiende como formato visual a cuestiones como colores y tamaños de tipografía, colores de fondo y de los diversos componentes, enlaces (pulsados y sin pulsar)... Simplemente cambiando la hoja de estilos se podría modificar el diseño visual del contenido desplegado.
PHP	Lenguaje de programación	Es el lenguaje utilizado en el proyecto para realizar el <i>core</i> o <b>backend</b> , es decir, toda la operativa del proyecto. Es el lenguaje a través del cual se realizan las operaciones necesarias para el funcionamiento de la aplicación, esto es, la recuperación de datos solicitados por el usuario y que serán mostrados en la <i>vista</i> o <b>frontend</b> , el guardado de las nuevas entradas con sus correspondientes comprobaciones necesarias, las modificaciones, etc.
SYMFONY	Framework web	Framework web basado en PHP que implementa el patrón Modelo-Vista-Controlador <b>MVC</b> . Ayuda y facilita en las tareas de escuchar los diferentes endpoints (urls) y mapearlas a los correspondientes métodos que realizan las funciones que implementan el comportamiento buscado para esa operación y que están escritas en PHP. También ayuda a trabajar con las entidades que representan los datos almacenados en la BBDD, así como los repositorios que permiten acceder y consultarlos de forma sencilla.
TWIG	Motor de plantillas	Es el motor por defecto utilizado por Symfony y que permite generar el HTML de forma dinámica para ser enviado a la vista. El contenido de este HTML va a ser función de la consulta que nos devuelva la BBDD. En esta plantilla se define un esqueleto del HTML que será rellenado en cada ejecución con los datos correspondientes enviados por el controlador después de consultar al modelo.

APLICACIÓN WEB	Acceso a servidor web	Es un programa que se codifica en un lenguaje interpretable por los navegadores web en la que se confía la ejecución al navegador. Otorga una independencia del sistema operativo, así como la facilidad para actualizar las aplicaciones web sin distribuir e instalar el software a usuarios potenciales.
BDD (MySQL)	Biblioteca de datos	Conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. MySQL permite almacenar y posteriormente acceder a los datos de forma rápida y estructurada. Se utilizará una base de datos dinámica, la cual permite modificar la información almacenada con operaciones como actualización, borrado y edición de datos (además de la posibilidad de generar consultas).

*Agradecimiento a Mónica Pérez Sueiro, por su ayuda y calidad como profesional.*



## Índice

Resumen.....	4
Abstract.....	5
Palabras Clave.....	6
Introducción/motivación.....	10
Objetivos.....	11
Estado del arte.....	12
Caso de estudio. (Opcional).....	14
Diagramas.....	15
Desarrollo del proyecto.....	17
Viabilidad tecno-económica.....	27
Trabajo futuro.....	29
Conclusiones.....	31
Biblioteca de recursos web y referencias.....	32

## Introducción/motivación.

La primera y fundamental motivación es la condición de que es un requisito imprescindible para la finalización del ciclo formativo y la obtención del título, siendo en segundo lugar un reto personal y una motivación para probar y comprobar todos los conocimientos adquiridos durante la etapa formativa de estos dos últimos años. Este proyecto obliga a integrar todos los conocimientos adquiridos en los diferentes módulos y ponerlos a trabajar en conjunto para la realización de un proyecto completo y funcional, con un grado de extensión y complejidad superior a los realizados en las diferentes evaluaciones.

Para ello se desea utilizar unas tecnologías ampliamente extendidas y utilizadas en el ámbito profesional, para que este desarrollo aporte una experiencia que pueda facilitar la posterior inclusión en el mercado laboral. Para ello se eligen lenguajes como PHP y el framework Symfony, utilizados en las descritas condiciones, son tecnologías modernas y útiles que se han aprendido en el ciclo y que ayudan al interés de crear un proyecto entero desde el principio hasta el fin.

Respecto a la elección del tema, teniendo en cuenta la evolución del mundo de los carburantes y la subida de sus precios y el fenómeno que se está produciendo de la sustitución del vehículo particular por patinetes eléctricos, el car sharing, alquileres de coche, ubers... a costa de perder cierta libertad en la movilidad. Esta aplicación podría ayudar calculando y mostrando los gastos correspondientes a tener un coche en propiedad y mejorando la gestión de éste, proporcionando datos históricos o en determinados periodos.

Las entidades reflejadas en el proyecto son conceptos básicos que todo el mundo conoce, con lo cual no se proporciona un modelo extremadamente difícil y complejo que dificulte el uso de la aplicación por parte de los usuarios. La aplicación está pensada para un uso familiar, pero posteriormente se podría ampliar para la utilización de esta por parte de un taller, de tal forma que registre a los clientes y sus coches, así como las intervenciones realizadas en ellos, pudiendo llegar a intentar conseguir un uso comercial del trabajo aquí realizado. Se desarrollarán estas ideas en el apartado correspondiente al trabajo futuro.

### Objetivos.

Con el desarrollo de esta aplicación web se busca cumplir con una serie de objetivos. El primero de ellos es crear una herramienta sencilla de utilizar por el usuario, accesible e intuitiva, con funcionalidades que generen sensación de utilidad, comodidad y buena experiencia de uso.

El segundo objetivo que se pretende conseguir es buscar una diferenciación en el mercado al compararse con otras aplicaciones similares. El poder administrar cualquier marca de automóviles (pudiendo añadirse en cualquier momento) es una gran ventaja en el mundo cambiante actual (y el auge de las nuevas marcas incipientes de vehículos electrificados). A pesar de no tener porque contener una información completa en un principio, se podrían ir añadiendo en el futuro por los usuarios.

El que sea una aplicación web, permite que el acceso a ella sea totalmente transparente, cómodo y sencillo, pudiendo entrar desde cualquier lugar en el que se encuentre el usuario siendo únicamente necesario que disponga de un dispositivo con un *browser* o navegador web y una conexión a Internet.

Esto nos da una amplitud de posibilidades bastante elevada, ya que abarca opciones como puede ser un *smartphone* (teléfono inteligente), tablet, ordenadores portátiles o de sobremesa, incluso en la actualidad, aunque más incómodos, navegadores integrados en los televisores. Simplemente necesitamos, como decíamos, que tenga conectividad a Internet.

De esta forma será fácil para el usuario, guardar los gastos (el ejemplo más claro son los repostajes) en el momento en que se producen, y por tanto posteriormente tendrá todos esos registros para poder realizar análisis reales de los gastos de su vehículo.

Este enfoque es mucho más versátil que lo que nos puede ofrecer una aplicación de escritorio clásica, donde solo podríamos interactuar con ella a través del equipo en el que estuviese instalada y que probablemente el usuario no lleve encima cuando se producen los gastos asociados al vehículo, y en caso de llevarla (un ordenador portátil) sería incómodo encenderla y registrar en ese momento el citado gasto o intervención.

### Estado del arte.

Para el desarrollo, de la aplicación web se ha buscado y utilizado una tecnología que esté ampliamente extendida, documentada y de la que sea fácil obtener ayuda de otros compañeros programadores del mundo y que permita a su vez, que la aplicación sea mantenida y extendida por cualquiera que pueda trabajar en ella.

Estas cualidades nos las ofrece el lenguaje PHP y el framework Symfony, enfocados claramente en la creación de páginas web dinámicas y concretamente aplicaciones web. Symfony de hecho es uno de los frameworks más utilizados para este lenguaje y usado fundamentalmente para la creación de este tipo de aplicaciones. Sigue el esquema Modelo-Vista-Controlador que se utiliza en este proyecto.

Dentro de las alternativas que podríamos usar (también ampliamente extendidas) podríamos citar al framework Laravel, basado igualmente en el lenguaje PHP, pero se optó por Symfony ya que es el framework que se uso durante el periodo académico que precedió y desembocó en este proyecto y del que se atesora cierto conocimiento previo.

Otras alternativas de frameworks destinados a la creación de aplicaciones web, y que sean ampliamente extendidos y evolucionados implicarían ya cambiar el lenguaje. Entre ellas podríamos citar varias como puede ser Spring (uno de los frameworks de desarrollo web más utilizados en la actualidad) pero basado en el lenguaje Java. Finalmente se puede citar otra alternativa menos extendida, como puede ser Ruby on Rails, basada en el lenguaje de programación Ruby.

Nos hemos decantado en este caso por la opción ofrecida por Symfony ya que como se ha mencionado se puede aprovechar la experiencia adquirida en otros proyectos a lo largo del curso y porque además ofrece una gran facilidad para ser instalado en cualquier equipo y ser desplegado y puesto en producción.

A nivel empresarial el uso de Java es muy común, pero PHP y Symfony le andan a la zaga, siendo la opción elegida en una gran cantidad de webs y grandes y conocidas. La opción de otros lenguajes como Ruby, fue descartada desde el inicio ya que su extensión a nivel empresarial es mucho menor, y el objetivo era utilizar un lenguaje que nos pueda proporcionar una experiencia en un proyecto completo que pueda ser aprovechada en la primera incursión laboral real.

En cuanto a alternativas a nuestra aplicación, conocemos [spritmonitor.de](http://spritmonitor.de) que ofrece una utilidad similar pero con un diseño arcaico y dificultoso, además de no estar adaptado para

## Proyecto Desarrollo de Aplicaciones Web

dispositivos móviles. El diseño de nuestra web será mas minimalista y sencillo de comprender.

## Caso de estudio. (Opcional)

El caso de uso que se va a relatar, describe el caso básico para un primer uso de la aplicación, lo cual sería el proceso completo y más detallado posible, siendo reducido en los usos subsiguientes en los que ya se tenga registrada cierta información.

El objetivo es poder registrar cualquier gasto asociado a un vehículo, distinguiendo por supuesto a que vehículo pertenece y a la persona propietaria / usuaria del citado vehículo.

Para ello por tanto se necesitará dar de alta a una nueva persona que será la propietaria de uno o más vehículos. A continuación podremos seleccionar a esta persona y podremos dar de alta el primer vehículo de los que le pertenecen. Para dar de alta a este vehículo, será necesario elegir un color, una marca y un modelo, viniendo la aplicación con la base de datos precargada con una extensa variedad de opciones y cubriendo todos los modelos lanzados por las principales marcas del sector.

En caso de que fuese un modelo nuevo o perteneciente a una marca minoritaria y poco conocida, sería necesario dar de alta dicho modelo o incluso la marca en el segundo escenario descrito. Primero daríamos de alta una nueva marca en caso de ser necesario, y una vez ya exista podríamos crear un nuevo modelo para ella. Habría que crear igualmente un nuevo color en caso de no existir el que queremos representar.

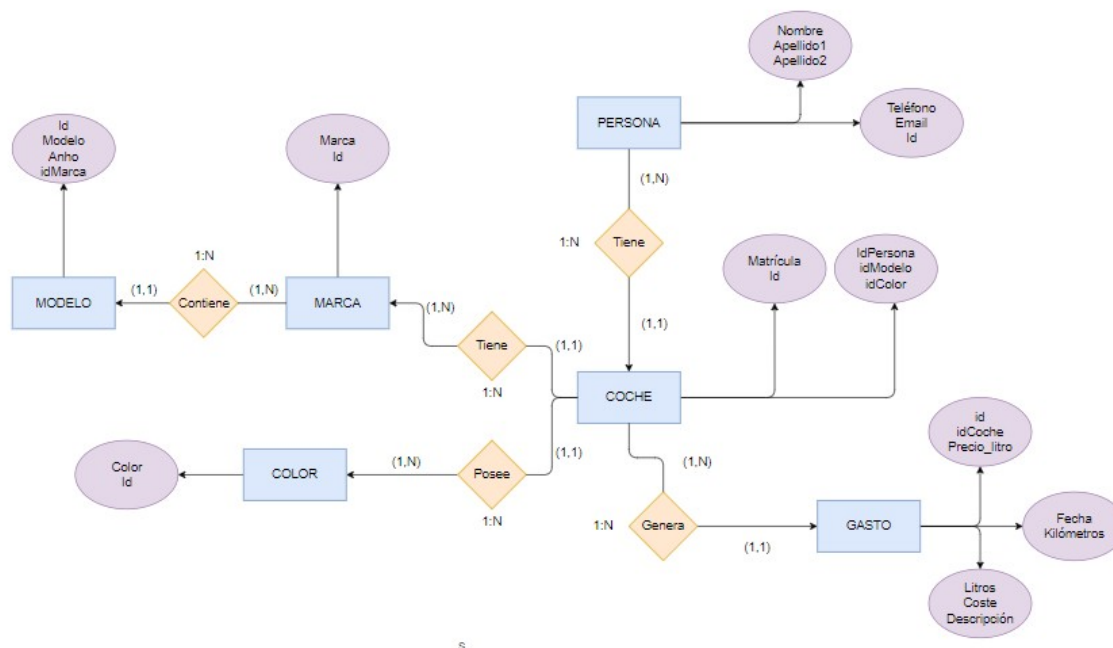
No obstante, lo habitual será que ya exista la marca y modelo y por tanto una vez se cree la persona se pueda dar de alta automáticamente un coche para ella. Y una vez esté creado el coche, ya podría ser seleccionado y por tanto registrar y dar de alta nuevos gastos asociados a él.

Para completar este caso de uso mediante la navegación a través de la aplicación podremos siempre editar o modificar cualquiera de las entidades o registros almacenados. Pudiese ser que cambiase el nombre comercial de alguna marca y por supuesto se podría querer editar los datos asociados a una persona, a un coche o a cualquiera de los gastos.

Así mismo se podría querer borrar cualquiera de las personas, coches o gastos y que no fuesen visibles desde la aplicación. El borrado de un coche implicaría el borrado de todos sus gastos asociados y el borrado de una persona implicaría el borrado de todos los coches de los que es propietaria, siendo a su vez borrados sus respectivos gastos. No tiene sentido permitir el borrado de las marcas, modelos y colores ya que enriquecen la base de datos y además pueden estar siendo usados por otros coches registrados posteriormente.

## Diagramas.

Diagrama Entidad-Relación.



En cuanto a las relaciones del diagrama, vamos a empezar explicando a partir de la entidad coche, que se podría considerar la entidad central del proyecto. Un coche pertenece a una única persona, pero una persona puede tener ninguno o más coches. Esto genera una relación “1:N” entre las entidades persona y coche. Tal y como se explicará en el trabajo futuro, una línea de sugerencia es que un coche pueda ser compartido / usado por más de una persona, en ese caso habría que actualizar esta relación a una “N:M”.

Así mismo, cada uno de estos coches está fabricado o es de un color (y en el proyecto no se contemplan los actuales coches bicolor, que en cualquier caso suelen hacer referencia al techo). Por otra parte, un color puede ser usado por ninguno o muchos coches. Esto genera una relación “1:N”.

Se ha decidido que color sea una entidad separada y no un atributo de coche para que los colores estén normalizados, es decir, si un coche es un Orient Blau de BMW, no se escribe al dar de alta el coche opciones como azul marino, azul oscuro, Orient Blue (la denominación oficial del color en inglés), etc. si no que se selecciona de un combo uno de los colores dados de alta. Se podrá decidir si se catalogan los colores existentes en las diferentes

## Proyecto Desarrollo de Aplicaciones Web

marcas o si por el contrario se usan colores genéricos comunes (azul, gris, plata...), pero en cualquier caso seguirán estando normalizados.

Finalmente, cada coche es un modelo y de este modelo puede haber más de un coche, de hecho de cada modelo se fabrican miles de coches. Por ejemplo un coche puede ser un Peugeot 308 o un Audi A3, pero solo uno de ellos, en cambio sabemos las miles de unidades que hay de éstos citados modelos, por lo que la relación entre las entidades modelo y coche es de "1:N".

Por su parte cada modelo pertenece a una marca, la cual tiene una serie de modelos (que a su vez definen el coche como se ha visto anteriormente). Una marca puede tener varios modelos, pero este modelo solo puede pertenecer a una marca. Por lo tanto se genera una relación "1:N" entre marca y modelo.

Por último, tenemos que relacionar gasto con coche. Un gasto es generado por un coche (echarle gasolina a un coche, pagar el seguro de un coche, cambiar el aceite a un coche...), pero un coche genera muchos gastos, y los gastos no se comparten entre coches. Esto genera una relación "1:N" entre las entidades coche y gasto.



# Desarrollo del proyecto

## 9.1. PASOS PREVIOS PARA EL DESARROLLO DEL PROYECTO E INSTALACIÓN.

Los pasos aquí detallados son necesarios para la realización del trabajo de programación de la aplicación detallada en este proyecto. Algunas de estas instrucciones son comunes y necesarias igualmente para la instalación en el equipo en el que se vaya a desplegar. Se explicitará cuales son comunes y cuales específicas para el desarrollo inicial.

Tanto para desarrollar como para ejecutar nuestro proyecto necesitamos software sobre el que ejecutar el proyecto de gestión de coches. En primer lugar se necesita instalar el intérprete de PHP ya que es el lenguaje de programación en el que se desarrolla la aplicación web y el Composer como gestor de paquetes de PHP para posteriormente obtener Symfony, los cuales podemos descargar desde las páginas web oficiales. Así mismo necesitamos una base de datos en la que almacenar el contenido de nuestra aplicación web y para ello vamos a usar la integrada en el paquete XAMPP, la cual es una MySQL.

Una vez tenemos este software básico instalado (PHP, Composer y XAMPP), procederemos a instalar el framework de Symfony. Para ello necesitamos tener PHP en las variables de entorno y teclearemos en una consola (cmd) lo siguiente:

```
c:\> php -r "readfile('http://symfony.com/installer');" > symfony
```

Hasta aquí sería la parte necesaria para la instalación del paquete de software en cualquier equipo que se quisiese ejecutar y desplegar la aplicación. Para el desarrollo además necesitamos ejecutar y configurar lo siguiente:

Necesitamos instalar Doctrine para que nos ayude a crear tanto entidades como repositorios de forma sencilla, y a continuación las tablas en la BBDD a partir de las entidades y realizar el mapeo de forma automática y coherente. Para lo cual teclearemos:

```
composer require symfony/orm-pack composer require --dev symfony/maker-bundle
```

Una vez tenemos todo el software instalado vamos a crear nuestro proyecto. Estos comandos son solo necesarios para la creación inicial (del proyecto y de entidades). Para ello teclearemos:

```
symfony new --full gestionCoches
```

Abrimos el proyecto con el Visual Studio y editamos el archivo .env para configurar el acceso a la BBDD (la cual tendremos que tener lanzada y configurada desde XAMPP). Este archivo se entregará ya configurado en el proyecto finalizado:

```
DATABASE_URL="mysql://root:1234@127.0.0.1:3306/gestionCoches?  
serverVersion=5.7"
```

También se puede crear la base de datos directamente con Doctrine con el siguiente comando:

```
php bin/console doctrine:database:create
```

Tras esto, se pueden añadir migraciones y "fields" dentro de la base de datos con el siguiente comando:

```
php bin/console make:entity
```

Una vez tenemos creada la entidad, podemos hacer que sea el propio Symfony quien nos cree la tabla correspondiente en la base de datos o nos actualice la existente si la hemos modificado. Para ello, hemos de teclear lo indicado en la última captura al terminar de crear la entidad:

```
php bin/console make:migration
```

Finalmente, después de introducir toda la información necesaria, podemos crear el controller, que nos da las operaciones básicas del CRUD (Create, Read, Update, Delete), por medio del comando:

```
php bin/console make:controller "Nombre"Controller
```

Con esto tendríamos el esqueleto del proyecto creado, tras eso tendríamos que trabajar editando y creando código de forma manual.

En la entrega del proyecto, se proporcionará una imagen de la base de datos (poblada con datos iniciales sobre marcas de coches, modelos y colores), que simplemente será necesario importar en el MySQL instalado en la máquina sobre la que se vaya a realizar el despliegue.

### 9.2. DESPLIEGUE Y EJECUCIÓN DE LA APLICACIÓN

Para ejecutar y desplegar la aplicación simplemente hay que abrir una consola de comandos en la carpeta donde está el proyecto y ejecutar la siguiente instrucción:

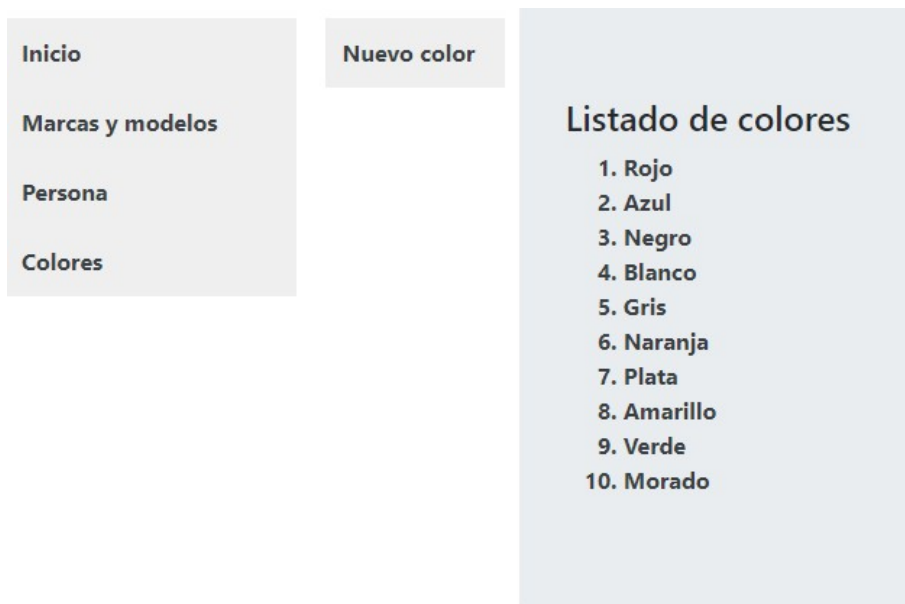
```
Symfony server:start
```

### 9.3. USO DE LA APLICACIÓN.

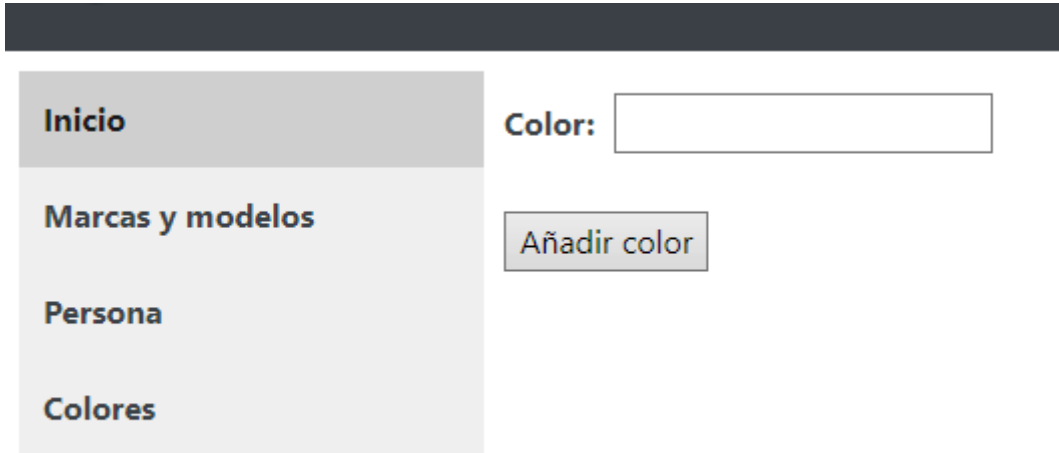
En este apartado se explicará el funcionamiento de la aplicación, comenzando por la pantalla principal.



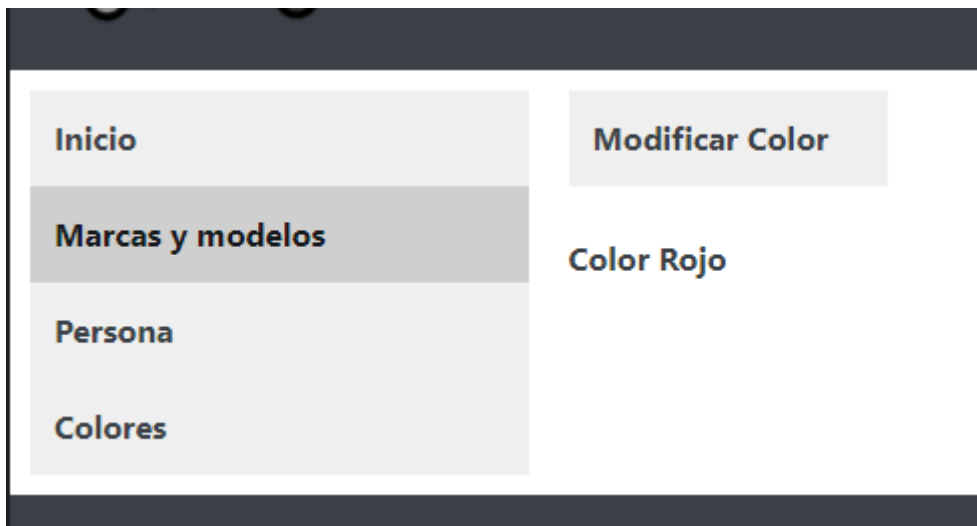
Desde ella podemos acceder a las tres opciones iniciales de la aplicación. Estas son el listado de personas registradas en la aplicación (y que pueden ser poseedoras de un coche), el listado de marcas y el listado de colores. Además, un jumbotron explica de manera breve las opciones de la aplicación. Se analizarán las tres opciones, empezando por el listado de colores.



En esta pantalla se muestra el listado de colores que se han dado de alta en la aplicación, pudiéndose aplicar a cualquier vehículo. Desde aquí se podrá acceder a cualquier color registrado para modificarlo o bien crear un nuevo color en la base de datos.

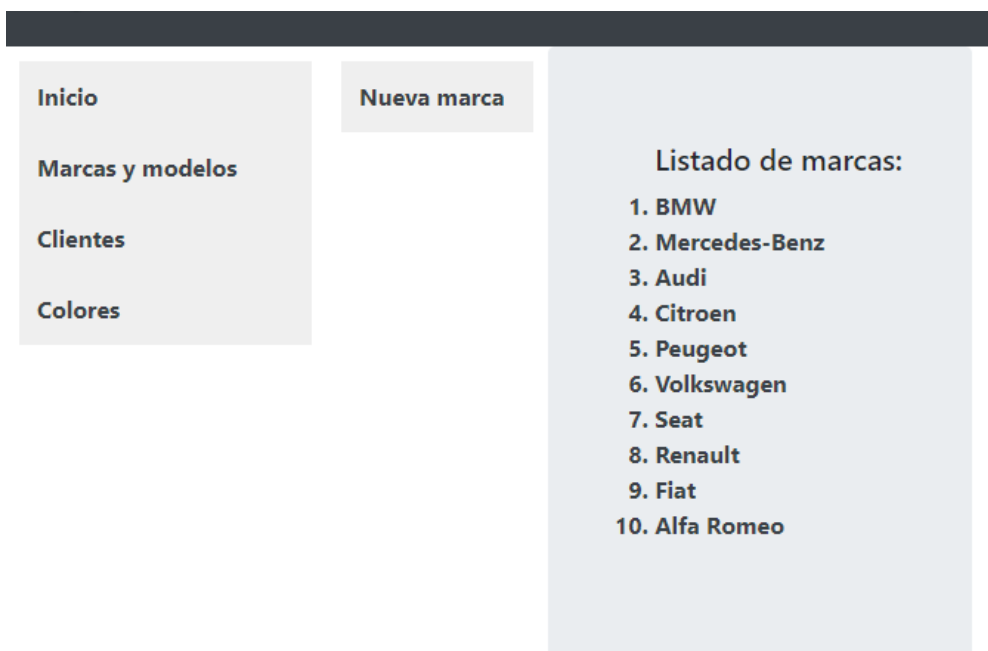


En esta pantalla se mostrará tras clicar en nuevo color. Simplemente escribimos el nuevo color que se quiere registrar y se añade pulsando en el botón.



Si por contra se entra en un color, y se mostrará un botón que da la opción de modificar. Se remarca que un color dado de alta no se puede borrar desde la aplicación ya que podría estar registrado para cualquier número de coches.

Volviendo al menú principal podríamos acceder a la segunda opción de las tres disponibles, el listado de marcas.



**Inicio**

**Nueva marca**

**Listado de marcas:**

1. BMW
2. Mercedes-Benz
3. Audi
4. Citroen
5. Peugeot
6. Volkswagen
7. Seat
8. Renault
9. Fiat
10. Alfa Romeo

Desde aquí se visualiza el listado de marcas completo de la base de datos y se puede o bien añadir una nueva marca o bien visualizar una marca determinada (donde saldrán listados todos sus modelos).



**Inicio**

**Marcas y modelos**

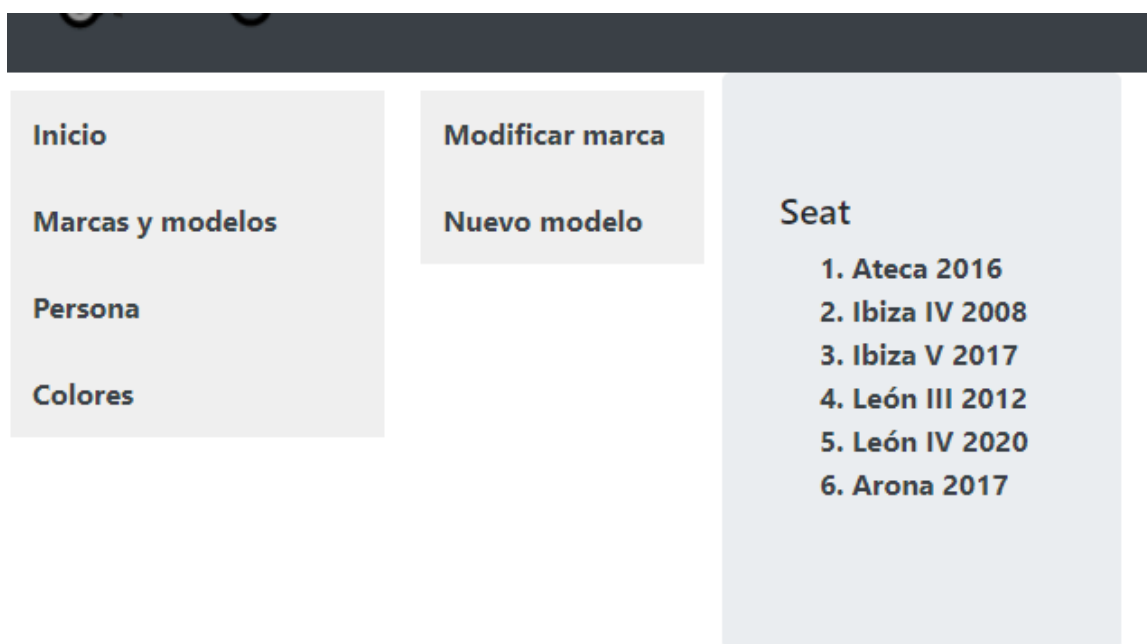
**Persona**

**Colores**

**Marca:**

**Añadir marca**

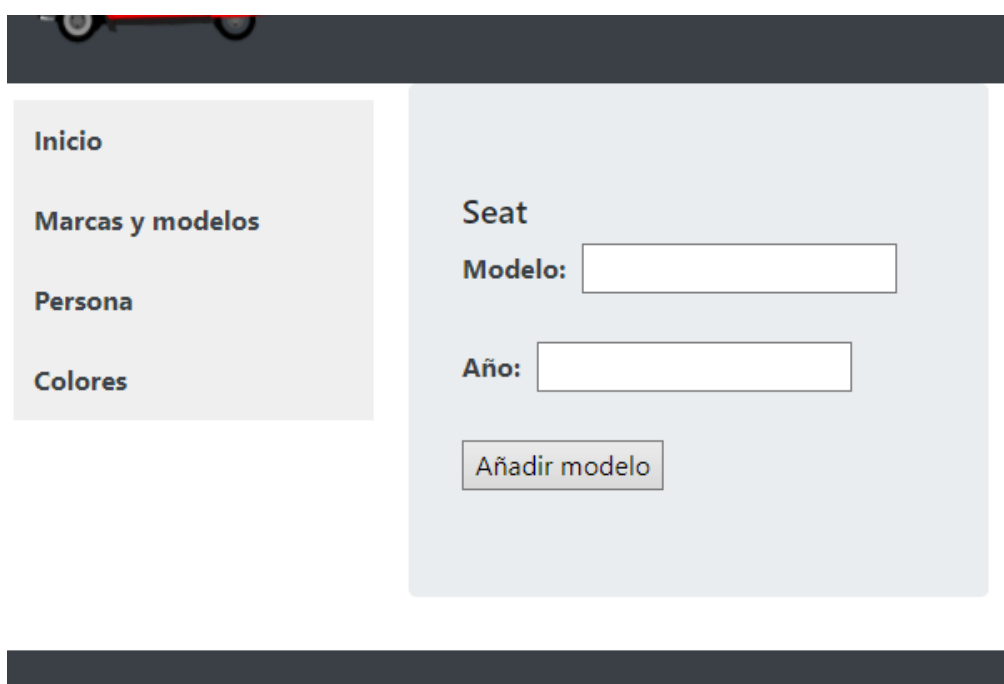
Tras clicar en nueva marca, aparecerá la opción para registrarla como se muestra en la captura.



The screenshot shows a web application interface. On the left, there is a sidebar menu with the following items: **Inicio**, **Marcas y modelos**, **Persona**, and **Colores**. The **Marcas y modelos** item is highlighted. To the right of the sidebar, there is a main content area. At the top of this area, there is a button labeled **Modificar marca** and a button labeled **Nuevo modelo**. Below these buttons, the word **Seat** is displayed. Under **Seat**, there is a list of six items: **1. Ateca 2016**, **2. Ibiza IV 2008**, **3. Ibiza V 2017**, **4. León III 2012**, **5. León IV 2020**, and **6. Arona 2017**.

Si en cambio accedemos a cualquiera de las marcas, visualizaremos todos sus modelos registrados.

En esta pantalla podemos bien modificar la marca (sería el mismo formulario que el de crear una nueva pero con el valor precargado), añadir un nuevo modelo o visualizar uno de los existentes.

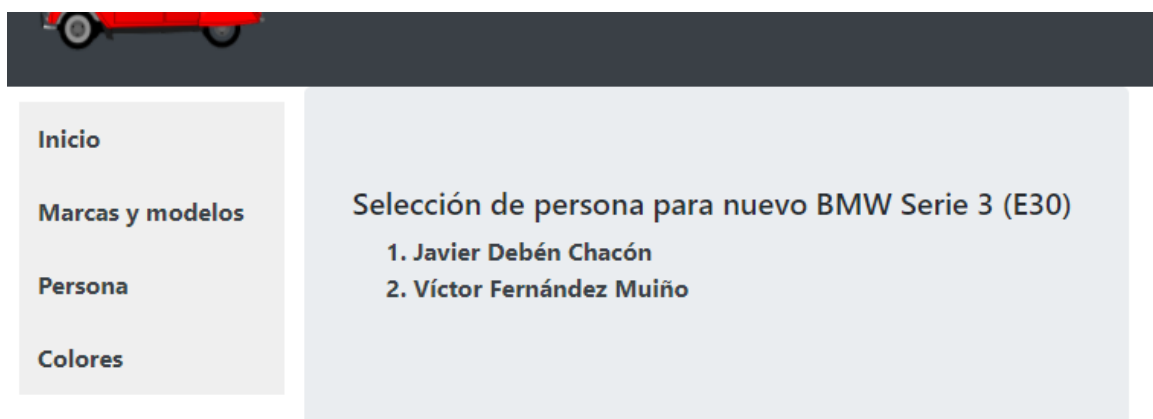


The screenshot shows a web application interface. On the left, there is a sidebar menu with the following items: **Inicio**, **Marcas y modelos**, **Persona**, and **Colores**. The **Marcas y modelos** item is highlighted. To the right of the sidebar, there is a main content area. At the top of this area, the word **Seat** is displayed. Below **Seat**, there are two input fields: **Modelo:** and **Año:**. Below these input fields, there is a button labeled **Añadir modelo**.

En el caso de querer registrar un nuevo modelo tendríamos que cubrir sus datos, añadiendo el nombre del modelo y el año del lanzamiento y guardándolo desde su correspondiente botón.



Si accedemos a uno de los modelos registrados visualizaremos el listado de coches de este modelo cualquiera que sea su propietario. Se visualizan datos de matrícula, color y propietario. Así mismo desde esta pantalla podemos modificar los datos del modelo y añadir un nuevo coche.



En este caso habría que elegir en primer lugar al propietario del listado de personas. Y finalmente ya podemos indicar el resto de datos como color y matrícula. Se guarda desde su correspondiente botón.

Inicio

Marcas y modelos

Persona

Colores

Modificar coche

Borrar coche

Nuevo gasto

**BMW Serie 3 (E30)**

Año: 1983

Matrícula: 8383CWJ

Color: Azul

Listado de gastos:

1. 11/08/2020 50€

2. 05/12/2019 60€

La última opción desde el listado de coches sería entrar a cualquiera de ellos y visualizar el listado de gastos que tengan registrados. Dentro de esta opción se pueden modificar los datos del coche, borrar el vehículo o añadir un nuevo gasto.

Inicio

Marcas y modelos

Persona

Colores

**BMW Serie 3 (E30) 8383CWJ Azul**

Fecha:

Litros:

Precio del litro:

Kilometros:

Coste:



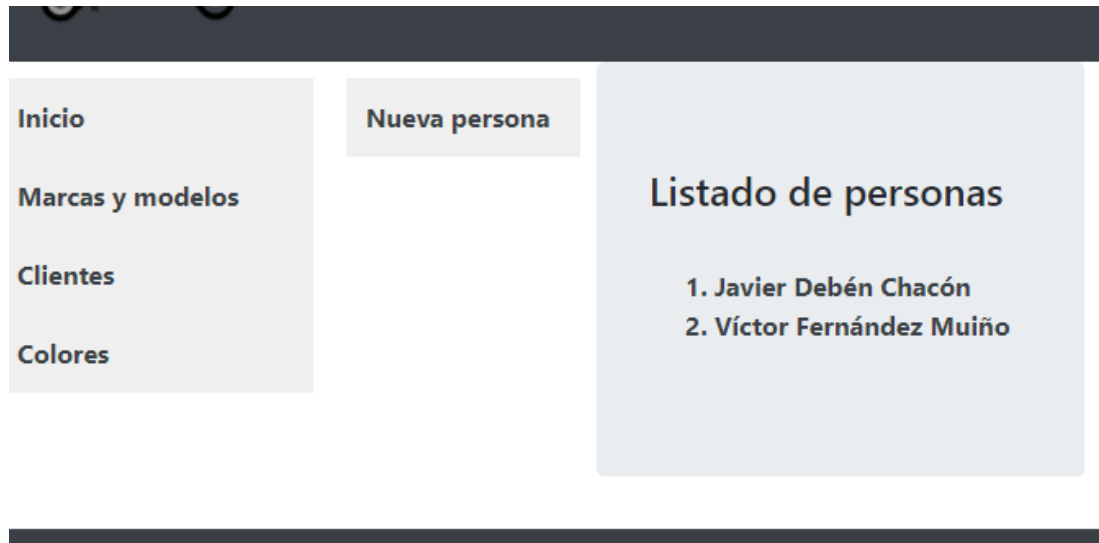
Si queremos añadir un nuevo gasto, se mostrará el formulario con los datos que cubrir, tras clicar en el botón se mostrará con los demás gastos.

<b>Inicio</b>	<b>Modificar gasto</b>	<b>BMW Serie 3 (E30)</b> <b>Matrícula:</b> 8383CWJ <b>Color:</b> Azul <b>Información sobre los gastos:</b> <b>Fecha:</b> 11/08/2020 <b>Kilómetros:</b> 15000 <b>Descripción:</b> <b>Litros:</b> 40 <b>Precio/litro:</b> 1.5€ <b>Coste:</b> 50€
<b>Marcas y modelos</b>	<b>Borrar gasto</b>	
<b>Persona</b>		
<b>Colores</b>		

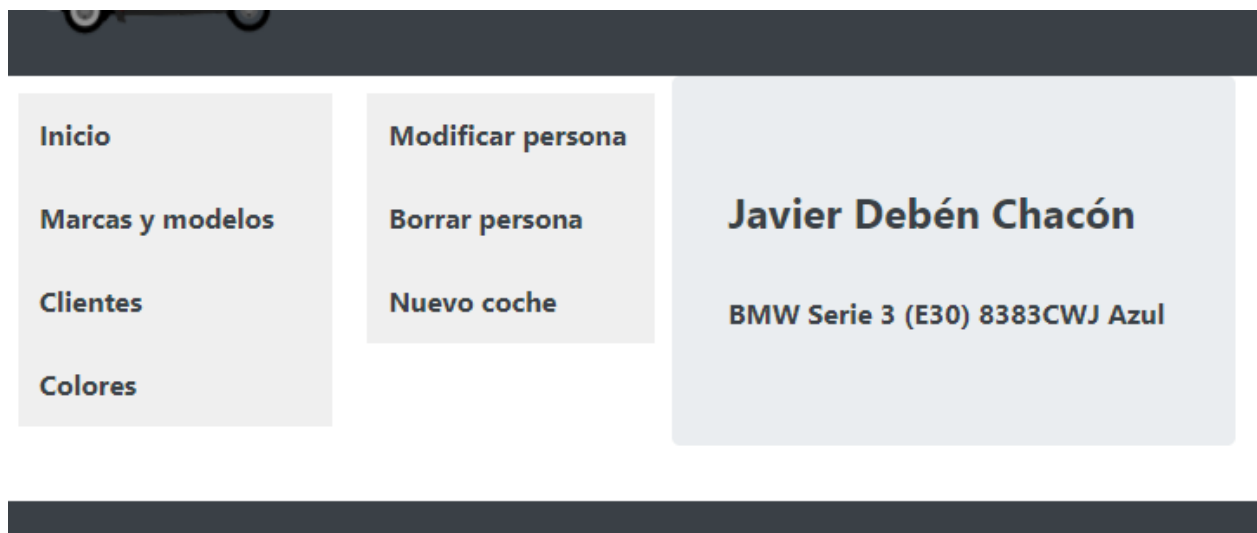
---

Tras clicar en uno de los gastos, este se mostrará con todos los datos específicos, añadiendo las opciones de borrar o modificarlo.

Por último se analizará la categoría clientes.



Tras clicar en la opción del menú, se mostrarán el listado de personas y un nuevo botón para añadir un nuevo cliente.



Tras clicar en uno de los clientes de la lista, aparecerá el nombre de la persona y una lista de sus vehículos, la cual nos permite clicar en un coche para mostrar sus datos (Desde marca o persona llegaremos a la misma información, pero en diferente orden). Además, permite la opción de modificar los datos del cliente, borrarlos o añadir un nuevo vehículo.

## Viabilidad tecno-económica.

Los costes asociados a este proyecto serían los correspondientes tanto a software como a hardware. Relativos al software se van a estudiar los generados por el propio desarrollo del proyecto así como los asociados a las licencias de software de terceros que vamos a utilizar. En la parte relativa al hardware, estará representado tanto el coste del equipo en el que se desarrollará el proyecto, como el de los equipos necesarios para su puesta en producción.

Comenzaremos el análisis por los costes de software. En cuanto a costes de licencias, no supondrán una carga para el proyecto, ya que todo el software utilizado será de código abierto. No hay que pagar regalías por el uso de PHP ni Symfony para la parte de desarrollo ni del IDE utilizado para la programación, utilizándose en este caso el Visual Studio Code.

Así mismo para la parte de base de datos utilizada y en la que se registrarán todos los datos almacenados por la aplicación, se eligió MySQL o MariaDB, las cuales son igualmente de código abierto y gratuitas. Nos falta mencionar la licencia del sistema operativo utilizado por el equipo de desarrollo, pero se podría dar por hecho que esta ya está contemplada en el coste del equipo que mencionaremos a continuación. Otra opción podría ser optar por una alternativa gratuita como una distribución de Linux.

A continuación se analizarán los costes asociados al hardware. Para el equipo de desarrollo se usará un portátil común con una antigüedad medianamente reciente del que podríamos estimar un coste aproximado de entre 600 y 800 euros. Este equipo podría ser sustituido por un ordenador de sobremesa si así se decidiese por temas de comodidad, ergonomía...

Esta aplicación se podría instalar de dos formas. En primer lugar una instancia para cada cliente (varios usuarios de su familia), en cuyo caso la carga a la que estaría sometido el servidor sería muy baja y podía desplegarse la aplicación en producción desde cualquier ordenador común y que estuviese actuando como servidor, aquí se tendría un coste de entre 600 y 800 euros.

Una segunda opción (una vez estuviese realizado el trabajo futuro de securización y login de usuarios) sería disponer de una única instancia a la que accediesen todos los clientes. En este caso la carga sería mayor y variable en función del número de clientes, por lo que en estas circunstancias lo ideal sería desplegar la aplicación a través de un hosting o incluso servicios en la nube como AWS (Amazon web service), Azure, ... Se desconoce los costes asociados a esta opción, que además escalarían en función de la carga.

Finalmente queda por mencionar las horas del programador (o programadores, en cuyo caso se multiplicaría el coste del equipo de desarrollo por el número de programadores)

## Proyecto Desarrollo de Aplicaciones Web

que de momento y tratándose de un proyecto académico, no suponen un coste real. En caso de continuar con el trabajo futuro e intentar monetizar de alguna manera la aplicación ya habría que tener en cuenta estas horas de trabajo que se convertirían en la mayor partida de costes del proyecto

### Trabajo futuro.

En este punto existen una gran cantidad de líneas de expansión para el proyecto, citaremos varias de ellas como ejemplo:

En primer lugar se podría añadir un módulo de estadísticas donde se muestren y seleccionen gastos, clasificándolos según la persona para cada uno de sus vehículos o el conjunto de ellos, pudiendo ser en el histórico completo o añadiendo filtros como periodos de tiempo, marcas o modelos específicos, etc. De esta forma, se podría obtener tanto el gasto realizado por el usuario en términos de movilidad, como los gastos asociados a determinada unidad de su parque móvil, de tal forma que le pueda ayudar a decidir sobre la idoneidad de proceder a la renovación por uno más moderno, en base a parámetros como el gasto asociado al número de averías, consumo de combustible, etc.

Se podrían recuperar los gastos de repostajes de forma filtrada (separándolos de otro tipo de gastos), y sumarlos de tal forma que se puedan calcular los consumos medios (sumando los litros y dividiendo estos entre la diferencia de kilómetros recorridos entre los citados repostajes), podríamos saber el consumo real que tiene el coche en el periodo de tiempo deseado y seleccionado, obteniendo por tanto el consumo real en vez del estimado por el ordenador de abordo del vehículo (en caso de que disponga de él, que no siempre tiene porque ser así).

Una vez se tiene hecho este cálculo, se puede recuperar el mismo valor para otras unidades del mismo o similar modelo (que disponga del mismo motor, etc.) de tal forma que se le proporcione al usuario el dato comparativo y pueda por tanto detectar un consumo excesivo que pueda denotar tanto una avería que deba ser reparada, como una forma de conducir que eleve este valor, y que pueda interesarle corregir en pos de un ahorro económico.

Así mismo otra línea de trabajo sugerida podría ser añadir un módulo de facturación. Esto implicaría abrir el mercado de potenciales clientes a “talleres de barrio”. El taller podría utilizarlo como ERP, gestionando los diferentes coches de los diferentes clientes. En este caso los gastos registrados (con la necesaria modificación) serían realmente las intervenciones realizadas por el taller en los coches. De tal forma que se pudiesen general facturas u obtener todos los trabajos realizados en un periodo de tiempo para obtener los ingresos del taller, realizar las liquidaciones de impuestos, etc.

Finalmente, otra posible evolución del proyecto está relacionada con lo comentado al explicar el modelo entidad-relación en el apartado 8 (diagramas). En él se explica que la relación entre personas y coches es 1:N (una persona puede tener muchos coches, pero un coche es de una única persona), la modificación consistiría en que un coche pudiese estar

## Proyecto Desarrollo de Aplicaciones Web

relacionado con más de una persona de tal forma que cualquiera de estos usuarios o propietarios pudiesen registrar gastos asociados a un determinado coche. Para ello habría que cambiar esta relación a una N:M (para ello habría que crear una tabla de relación entre esas dos entidades) y analizar si se debería añadir una relación entre gasto y persona para en esas posibles estadísticas tratar los datos de los gastos de cada persona.

## Conclusiones.

### 12. CONCLUSIONES.

Honestamente creo que hemos cumplido bastante los objetivos marcados a nivel académico. Se ha realizado un proyecto de cierta magnitud y complejidad superior a la realizada mientras se cursaban los módulos de los dos cursos académicos.

En el proyecto de Symfony realizado en clase, teníamos una única entidad con una única tabla en la base de datos. Aquí tenemos 6 entidades con relaciones entre ellas, un mucho mayor número de pantallas navegables por el usuario e integrando un diseño mejorado. Hemos unido los conocimientos adquiridos en los diferentes módulos dando lugar, como decimos, a un proyecto más grande y más complejo.

Así mismo hemos aprendido a realizar una memoria (aquí presente) sobre todo el trabajo realizado en este proyecto y se han invertido una considerable cantidad de horas para llegar a este punto. También se presentan instrucciones para la instalación y despliegue del proyecto en cualquier ordenador, siendo otros de los puntos que hemos necesitado para el desarrollo del proyecto.

Nos parece un más que digno proyecto de fin de ciclo para este nivel educativo, es cierto que como somos ambiciosos, hemos descrito bastantes líneas de trabajo futuro que aumentarían aún más la complejidad y necesitarían un mayor número de horas de desarrollo, pero realmente hemos cumplido con la idea inicial de la extensión que pretendíamos abarcar con este proyecto.

Igualmente, como corresponde a la realización de un primer proyecto en el que hemos volado libres por primera vez, sabemos que hay muchas cosas que se pueden mejorar. Estamos totalmente abiertos a recibir sugerencias, opiniones y matizaciones por parte del tribunal y profesorado para aprender de ello y mejorar de cara a nuestro futuro.

Así mismo queremos aprovechar esta ocasión para agradecer a los profesores, los conocimientos adquiridos durante estos dos cursos académicos y que nos han permitido la realización y entrega de este proyecto.

## Biblioteca de recursos web y referencias.

PNGEGG

<https://www.pngegg.com/en/png-bbnxz>  
<https://www.pngegg.com/en/png-bdmky>

CLEANPNG

[Free download Car Cartoon . - CleanPNG / KissPNG](#)

W3SCHOOLS

[W3Schools Online Web Tutorials](#)

SYMFONY

[Symfony, High Performance PHP Framework for Web Development](#)

SPRITMONITOR

[Spritverbrauch berechnen und Autokosten verwalten - Spritmonitor.de](#)