

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE SISTEMAS INFORMÁTICOS

UNIVERSIDAD POLITÉCNICA DE MADRID



GRADO EN INGENIERÍA DEL SOFTWARE

TRABAJO DE FIN DE GRADO

**JOBCAR - DESARROLLO DE UNA APLICACIÓN WEB CON
ANGULAR JS**

Autor: Rafael Gutiérrez Cuartero

Curso 2016/2017

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE SISTEMAS INFORMÁTICOS

UNIVERSIDAD POLITÉCNICA DE MADRID



GRADO EN INGENIERÍA DEL SOFTWARE

TRABAJO DE FIN DE GRADO

**JOB CAR - DESARROLLO DE UNA APLICACIÓN WEB CON
ANGULAR JS**

Autor: Rafael Gutiérrez Cuartero

Tutor: Juan Alberto de Frutos Velasco

Curso 2016/2017

AGRADECIMIENTOS

Ha sido largo el camino recorrido, desde el día en el que decidí estudiar la carrera de informática, hasta llegar a este momento en el que escribo estas líneas.

Ha habido momentos duros, en los que se me pasaba por la cabeza abandonar, pero al final todo esfuerzo tiene su recompensa, y tengo la obligación moral de mencionar a las siguientes personas para agradecerles profundamente, el haber estado a mi lado durante estos años, ya que sin ellas, ésto no habría sido posible.

A mi hermano, compañero de viaje, de horas de estudio, de tristezas por los suspensos y de alegrías por los aprobados, no podría haber tenido un compañero mejor.

A mis padres, por su eterna generosidad, y por estar siempre ahí, en los buenos momentos y en los no tan buenos.

A mi novia, por animarme a seguir, por tener una confianza infinita en mí, y por hacerme ver, que los límites se los pone uno mismo, nadie más.

A mi abuela, que aunque ya no se encuentra entre nosotros, si que lo está en mi mente, y ha sido una persona fundamental en mi vida.

A mis amigos, muchos de ellos con titulación de la ETSISI, por marcarme el camino y hacerme ver que ésto se podía conseguir.

A mí tutor, porque guardo un grato recuerdo del año que le tuve como profesor, y en el que pensé siempre como la persona ideal para dirigirme el proyecto.

RESUMEN

En este trabajo de Fin de Grado se ha realizado el desarrollo de una aplicación web llamada **JobCar**, utilizando el framework de JavaScript **Angular JS**. Está inspirada en la popular BlaBlaCar, pero enfocada a los desplazamientos diarios de casa al trabajo.

Es habitual sobre todo en grandes ciudades, los atascos en horas punta debido a la masiva concentración de coches en rutas comunes.

Es fácil comprobar también como en la gran mayoría de estos desplazamientos, el número de ocupantes de estos vehículos se reduce únicamente al conductor del mismo.

JobCar plantea una alternativa para mejorar esta situación, es decir, si hay personas que viven y trabajan relativamente cerca, podrían compartir coche, consiguiendo reducir el tráfico, el consumo de gasolina, y la contaminación.

Sería una alternativa más, y cada persona deberá elegir entre seguir usando el coche propio en solitario, utilizar JobCar, el transporte público o el taxi.

En JobCar, el usuario podrá ser **conductor**, **pasajero**, o las 2 cosas a la vez. Como conductor, será el creador de la ruta, y establecerá el trayecto, el horario, la frecuencia, las vacantes disponibles, y el precio. Como pasajero, podrá comprobar la rutas disponibles en JobCar, y unirse a la que más le convenga.

SUMMARY

In this paper Final Project has made the development of a **JobCar** called web application, using the **Angular JS** JavaScript framework. It is inspired by the popular BlaBlaCar, but focused on commuting from home to work.

It is common especially in large cities, traffic jams at peak times due to the massive concentration of cars in common routes.

It is also easy to see how the vast majority of these displacements, the number of occupants of these vehicles is reduced only to the driver thereof.

JobCar poses an alternative to improve this situation, ie, if there are people who live and work relatively close car could share, managing to reduce traffic, fuel consumption and pollution.

It would be an alternative, and should be the users who decide between continuing to use the car itself alone, use JobCar, public transport or taxi.

In JobCar, the user may be **driver**, **passenger**, or 2 things at once. As a driver, is the creator of the route, and set the route, time, frequency, vacancies available, and price. As a passenger, you can check the available routes in JobCar, and join the one that suits you.

TABLA DE CONTENIDOS

	PÁG
1. INTRODUCCIÓN Y OBJETIVOS	11
1.1 TECNOLOGÍAS.....	12
2. ANGULAR JS	17
2.1 ¿QUÉ ES ANGULAR JS?.....	17
2.2 MVC.....	19
2.3 ÁMBITO	21
2.3.1 Scope hace de enlace entre las vistas y los controladores	21
2.3.2 Ámbitos del scope dentro de la vista.....	22
2.3.3 Alias del scope	22
2.3.4 Ámbito raíz con \$rootScope.....	23
2.3.5 Conociendo \$parent	23
2.3.6 Ejemplo con ámbitos corrientes y ámbito root.....	24
2.3.7 Angular Batarang	25
2.4 DIRECTIVAS.....	27
2.4.1 Tipos de Directivas.....	27
2.5 INYECCIÓN DE DEPENDENCIAS	29
2.5.1 ¿Qué es la inyección de dependencias?.....	29
2.6 SERVICIOS.....	30
2.6.1 ¿Qué son los servicios?	30
2.6.2 Ciclo de vida de Angular JS.....	30
Fase de configuración	30
Fase de ejecución	31
2.6.3 Tipos de Servicios y Ejemplos	32
2.7 PROMESAS EN ANGULAR JS	35
2.7.1 ¿Qué son las promesas?	35
2.7.2 Servicio \$q.....	35
2.8 MÓDULO NGROUTE.....	36
2.8.1 ¿Qué es y para qué sirve?	36
2.8.2 Ejemplo de configuración de rutas.....	37
2.9 CÓMO DESCARGAR ANGULAR JS	38
3. REQUISITOS.....	39
3.1 REQUISITOS FUNCIONALES	39
3.2 REQUISITOS NO FUNCIONALES	40
4. ANÁLISIS	41
4.1 DIAGRAMA ENTIDAD-RELACIÓN.....	41
4.2 CASOS DE USO.....	42
5. DISEÑO.....	55
5.1 PASO A TABLAS	55
5.2 CONTROLADORES, VISTAS Y MODELOS UTILIZADOS.....	57
5.3 EJEMPLOS DEL DISEÑO Y DEL CÓDIGO DESARROLLADO	59
5.3.1 Diseño de Jobcar, una aplicación web de una sola página	59

5.3.2 Ejemplo de uso de directivas nativas, ng-include y ng-view.....	60
5.3.3 Ejemplo de uso de configuración de las rutas.....	60
5.3.4 Ejemplo de uso del scope	61
5.3.5 Ejemplo de uso del rootScope	62
5.3.6 Ejemplo de uso de inyección de dependencias	62
5.3.7 Ejemplo de uso de factoría (factory)	63
5.4 CONFIGURACIÓN EN LA PARTE DEL SERVIDOR.....	64
6. GUÍA DE USUARIO.....	67
6.1 REGISTRO	67
6.2 LOGIN	69
6.3 CREAR RUTA	70
6.4 YO, CONDUCTOR.....	71
6.5 VER PASAJEROS	72
6.6 ELIMINAR PASAJERO	73
6.7 ELIMINAR RUTA	75
6.8 VER RUTAS	76
6.9 UNIRSE A RUTA.....	77
6.10 YO, PASAJERO	79
6.11 SALIRSE DE RUTA.....	80
7. PRUEBAS	83
8. ASPECTOS A MEJORAR.....	97
9. CONCLUSIONES	99
10. BIBLIOGRAFÍA	101

1. INTRODUCCIÓN Y OBJETIVOS

JobCar es una aplicación web que permite crear rutas y publicirlas, para que otros usuarios se puedan unir a ellas. Tras el registro inicial, y el posterior logado, el usuario tiene la posibilidad de ser conductor, pasajero, o ejercer ambos roles simultáneamente.

Como conductor, puede crear una o más rutas, y será el administrador de cada una de ellas. Es decir, puede ver qué pasajeros tiene, darlos de baja, y también puede eliminar las rutas que haya creado.

Como pasajero, puede unirse a una o más rutas, al tener acceso a las rutas con vacantes disponibles. Por supuesto, puede salirse de una ruta en cualquier momento, sin previo aviso.

En cuanto al desarrollo de Jobcar, se ha utilizado Angular JS, junto con otras tecnologías. Angular JS es el popular framework de JavaScript para el desarrollo en el lado del cliente.

Hace uso del patrón Modelo-Vista-Controlador, que permite separar la capa de presentación, la capa lógica y los componentes de la aplicación.

Su principal uso es la creación de las aplicaciones web de una sola página (SPA's – Single Page Applications), dotándolas de una gran fluidez y rapidez.

Ofrece un contenido dinámico, a través de un data binding bidireccional, que permite la sincronización automática de modelos y vistas.

El presente proyecto de fin de grado tiene como objetivos:

- Realizar el análisis, diseño, desarrollo y pruebas para la creación de una aplicación web en Angular JS.
- Describir a grandes rasgos qué es Angular JS, y cómo se ha utilizado.
- La adquisición de conocimientos en otras tecnologías como bootstrap, html, css, php, y mysql.
- Plantear una solución, para aumentar el nivel de ocupación de los vehículos en los desplazamientos diarios al trabajo, contribuyendo a que el tráfico sea más fluido, y reducir tanto el ruido como los niveles de contaminación.
- Consolidarse como punto de encuentro en internet, para que las personas con trayectos similares puedan compartir vehículo.

1.1 Tecnologías

Junto con Angular JS, para poder realizar esta aplicación web, ha sido necesario hacer uso de las siguientes tecnologías:

- **Html:** Sus siglas en inglés significan *HyperText Markup Language* (lenguaje de marcas de hipertexto), y hace referencia al lenguaje de marcado para la elaboración de páginas web. El HTML se escribe en forma de «etiquetas», rodeadas por corchetes angulares (<,>, /). Los elementos son la estructura básica de HTML, y tienen dos propiedades básicas: atributos y contenido. Cada atributo y contenido tienen ciertas restricciones para que se considere válido al documento HTML.

Se ha utilizado html5 para el desarrollo de las **vistas** de la aplicación (lo que el usuario ve cuando navega a través de JobCar).

- **CSS:** Son las siglas de *Cascading Style Sheets* (Hoja de estilo en cascada), es un lenguaje usado para definir y crear la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). El World Wide Web Consortium (W3C) es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los agentes de usuario o navegadores. La idea que se encuentra detrás del desarrollo de CSS es separar la estructura de un documento de su presentación.

La información de estilo puede ser definida en un documento separado (como se ha hecho en JobCar) o en el mismo documento HTML.

- **Animate.css:** Se ha utilizado esta hoja de estilos que contiene animaciones css3, concretamente en el archivo "index.html" del proyecto, donde la imagen del coche realiza una animación al cargar la página.



JobCar

El código utilizado para lograr este efecto, es el siguiente:

```

```

- Sweetalert.css: Es una librería externa, disponible en la página <http://t4t5.github.io/sweetalert/> , mediante la cuál se han podido incluir alertas en la aplicación, logrando una estética más agradable y de forma sencilla. Concretamente, se ha utilizado en los archivos siguientes del proyecto:
 1. controladorConductor.js → En la función eliminarRuta
 2. controladorCrearRutas.js → En la función guardarRuta
 3. controladorPasajero.js → En la función salirseDeRuta
 4. controladorVerRutas.js → En la función unirseARuta
- **Bootstrap:** Twitter Bootstrap es un framework o conjunto de herramientas de código abierto para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como, extensiones de JavaScript adicionales. Es el proyecto más popular en GitHub y es usado por la NASA y la MSNBC junto a demás organizaciones. Bootstrap fue desarrollado por Mark Otto y Jacob Thornton de Twitter, como un marco de trabajo (framework) para fomentar la consistencia a través de herramientas internas. Antes de Bootstrap, se usaban varias librerías para el desarrollo de interfaces de usuario, las cuales guiaban a inconsistencias y a una carga de trabajo alta en su mantenimiento.



- AdminLTE.min.css: Forma parte de un tema gratuito de Bootstrap que se ha utilizado en JobCar, y que se incluye en el código, en el archivo "index.html" de la aplicación.

El uso de Bootstrap ha permitido, mejorar la apariencia de la aplicación de forma sencilla.

- **PHP:** Es un lenguaje de programación de uso general de código del lado del servidor, originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor, que se podían incorporar directamente en el documento HTML, en lugar de llamar a un archivo externo para que procesara los datos.

El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página web resultante. PHP ha evolucionado por lo que ahora incluye también una interfaz de línea de comandos que puede ser usada en aplicaciones gráficas independientes. Puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún coste.

Son los archivos con extensión .php del código de JobCar.



- **Medoo:** Para que el uso de PHP resultará más sencillo de llevar a cabo, se ha utilizado este framework, que permite manejar toda la parte de base de datos, tanto las conexiones como las consultas.
- **MySQL:** Es un sistema de gestión de bases de datos relacional, desarrollado bajo licencia dual GPL/Licencia comercial por Oracle Corporation, y está considerada como la base de datos open source más popular del mundo, y una de las más populares en general junto a Oracle y Microsoft SQL Server, sobre todo para entornos de desarrollo web. MySQL fue inicialmente desarrollado por MySQL AB (empresa fundada por David Axmark, Allan Larsson y Michael Widenius). MySQL A.B. fue adquirida por Sun Microsystems en 2008, y ésta a su vez fue comprada por Oracle Corporation en 2010, la cual ya era dueña desde 2005 de Innobase Oy, empresa finlandesa desarrolladora del motor InnoDB para MySQL.



Dentro de los archivos php, se han realizado en MySQL una parte de las queries necesarias, las demás se han hecho utilizando el framework medoo, para obtener información de la base de datos de JobCar.

- Entorno de desarrollo: **Sublime Text** es un editor de texto y editor de código fuente, está escrito en C++ y Python para los plugins. Desarrollado originalmente como una extensión de Vim, con el tiempo fue creando una identidad propia, por ésto aún conserva un modo de edición tipo vi llamado *Vintage mode*. Se puede descargar y evaluar de forma gratuita. Sin embargo no es software libre o de código abierto y se debe obtener una licencia para su uso continuado, aunque la versión de evaluación es plenamente funcional y no tiene fecha de caducidad.

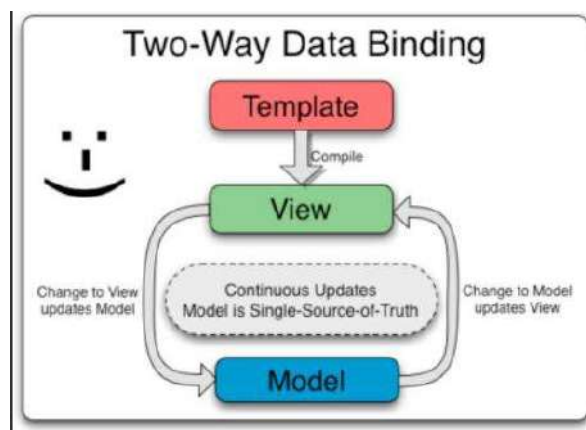
2. ANGULAR JS

2.1 ¿Qué es Angular JS?

AngularJS, o simplemente Angular, es un framework de JavaScript de código abierto, mantenido por Google, que se utiliza para crear y mantener aplicaciones web de una sola página. Su objetivo es aumentar las aplicaciones basadas en navegador con capacidad de modelo-vista-controlador (MVC), en un esfuerzo para hacer que el desarrollo y las pruebas sean más fáciles.



Se centra en el desarrollo Front-End, es decir, en la interfaz de la aplicación web. Este framework adapta y amplía el HTML tradicional, para servir mejor contenido dinámico a través de un data binding bidireccional, que permite la sincronización automática de modelos y vistas.



La biblioteca lee el HTML que contiene atributos de las etiquetas personalizadas adicionales, entonces obedece a las directivas de los atributos personalizados, y une las piezas de entrada o salida de la página a un modelo representado por las variables estándar de JavaScript. Los valores de las variables de JavaScript se pueden configurar manualmente, o recuperados de los recursos JSON estáticos o dinámicos.

AngularJS se puede combinar con el entorno en tiempo de ejecución Node.js, el framework para servidor Express.js y la base de datos MongoDB para formar el conjunto **MEAN**.

AngularJS está construido en torno a la creencia, de que la programación declarativa es la que debe utilizarse para generar interfaces de usuario y enlazar componentes de software, mientras que la programación imperativa es excelente para expresar la lógica de negocio.

AngularJS pone menos énfasis en la manipulación del DOM y mejora la testeabilidad y el rendimiento.

Los objetivos de diseño:

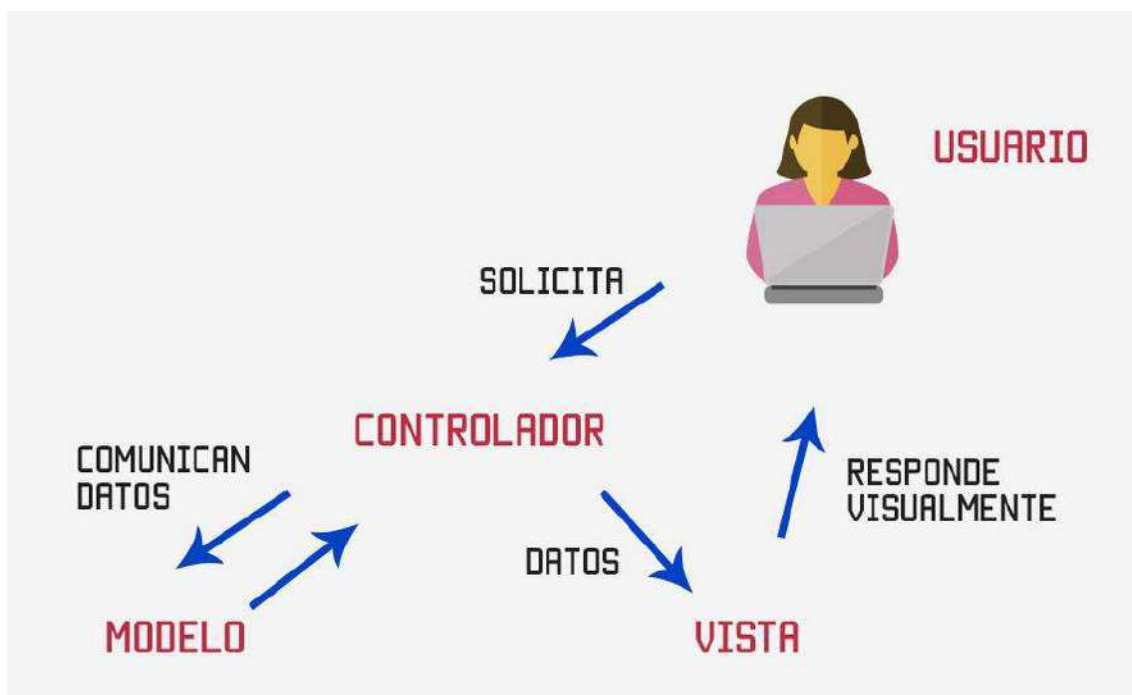
- Separar la manipulación del DOM de la lógica de la aplicación. Esto mejora la capacidad de prueba del código.
- Considerar a las pruebas de la aplicación como iguales en importancia a la escritura de la aplicación. La dificultad de las pruebas se ve reducida drásticamente por la forma en que el código está estructurado.
- Disociar el lado del cliente de una aplicación del lado del servidor. Esto permite que el trabajo de desarrollo avance en paralelo, y permite la reutilización de ambos lados.
- Guiar a los desarrolladores a través de todo el proceso de desarrollo de una aplicación: desde el diseño de la interfaz de usuario, a través de la escritura de la lógica del negocio, hasta las pruebas.

Con el uso de la inyección de dependencias, Angular lleva servicios tradicionales del lado del servidor, tales como controladores dependientes de la vista, a las aplicaciones web del lado del cliente. En consecuencia, gran parte de la carga en el backend se reduce, lo que conlleva a aplicaciones web mucho más ligeras.

2.2 MVC

El modelo–vista–controlador (MVC) es un patrón de arquitectura de software, que separa la parte visual, de la funcionalidad y las estructuras de datos. Para ello el MVC propone la construcción de tres componentes distintos: el **modelo**, la **vista** y el **controlador**, es decir, por un lado define componentes para la representación de la información, y por otro lado para la interacción del usuario.

Este patrón de arquitectura de software se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento.



Los componentes del MVC se pueden definir como:

- **Modelo:** Es la representación de la información con la cual el sistema opera, por lo tanto gestiona todos los accesos a dicha información, tanto consultas como actualizaciones, implementando también los privilegios de acceso que se hayan descrito en las especificaciones de la aplicación (lógica de negocio). Envía a la vista aquella parte de la información que en cada momento se le solicita para que sea mostrada (típicamente a un usuario). Las peticiones de acceso o manipulación de información llegan al modelo a través del controlador.

- **Vista:** Presenta el modelo (información y lógica de negocio) en un formato adecuado para interactuar (usualmente la interfaz de usuario), por tanto requiere de dicho modelo la información que debe representar como salida.
- **Controlador:** Responde a eventos (usualmente acciones del usuario), e invoca peticiones al modelo cuando se hace alguna solicitud sobre la información (por ejemplo, editar un registro en una base de datos). También puede enviar comandos a su vista asociada si se solicita un cambio en la forma en que se presenta el modelo, por tanto se podría decir que el controlador hace de intermediario entre la vista y el modelo.

2.3 Ámbito

La traducción de scope es **ámbito**, sin embargo, el término es tan habitual que se suele usar directamente la palabra en inglés.

En una aplicación en Angular JS, no existe un único ámbito, hay diferentes scopes en diferentes partes del código, anidados o en paralelo. Los scope se crean automáticamente para cada uno de los controladores de una aplicación, y además existe un scope raíz para toda la aplicación (\$rootScope).

2.3.1 Scope hace de enlace entre las vistas y los controladores

El scope es la magia que permite que los datos que se manejan en los controladores, pasen a las vistas, y viceversa. Es el enlace que traslada esos datos de un lugar a otro, sin que el programador tenga que hacer nada.

En los controladores se inyecta el scope mediante el parámetro \$scope. Posteriormente se podrán añadir datos como propiedades a ese objeto inyectado. Ejemplo:

```
.controller('otroCtrl', function($scope){  
  $scope.algo = "probando scope...";  
  $scope.miDato = "otra cosa..."  
});
```

Lo anterior generará un dato en el scope llamado "algo" y otro llamado "miDato".

Desde las vistas se puede acceder a datos del scope usando la sintaxis de dobles llaves:

```
{{ algo }}
```

Por lo tanto, "algo" es un dato que está almacenado en el scope desde el controlador. Con esta expresión se traslada a la página. También se puede acceder al scope al usar la directiva ng-model y definiendo un dato en el modelo.

```
<input type="text" ng-model="miDato">
```

El scope sirve para trasladar datos entre el controlador y la vista. Para poder acceder a "algo" o a "miDato" no es necesario enviarlo desde el controlador a la vista ni de la vista al controlador, sólo hay que crear esa propiedad en el \$scope. Una de las ventajas de Angular JS, es el Binding, cuando el dato se modifica, ya sea porque se le asigna un nuevo valor desde el controlador o porque se cambia lo que hay escrito en el INPUT de la vista, la otra parte es consciente de ese cambio sin tener que estar suscritos a eventos.

2.3.2 Ámbitos del scope dentro de la vista

Cada scope tiene su ámbito restringido a una parte del código.

Si revisamos el siguiente HTML:

```
<div ng-controller="miAppController">
  <p>Contenido de la vista acotado por un controlador</p>
  <p>{{ cualquierCosa }}</p>
</div>
<section>
  Desde fuera de esa división no hay acceso al scope del controlador
</section>
```

En este ejemplo, hay un scope creado por el controlador "miAppController", y tiene validez dentro de ese código HTML en el lugar donde se ha definido la directiva ngController. Es decir, los elementos que se asignen a \$scope dentro de este controlador se podrán ver desde la etiqueta DIV y todas sus hijas, pero no desde etiquetas que se encuentren fuera, como es el caso del SECTION que hay después.

2.3.3 Alias del scope

Existe otra modalidad de enviar datos al scope y es por medio de un alias. Para conseguir ésto, al declarar los controladores en el HTML (directiva ng-controller) hay que utilizar la sintaxis "controller..as".

```
<div ng-controller="pruebaAppCtrl as vm">
```

En este caso se podrán generar datos en el scope de dos maneras, o bien a través del mismo objeto \$scope que se puede inyectar, como está explicado en el apartado anterior, o bien a través de la variable this dentro de la función.

```
.controller('pruebaAppCtrl', function($rootScope){
  var modeloDeLaVista = this;
  modeloDeLaVista.otroDato = "Esto está ahora en el scope, para acceder a través de un alias";
});
```

Gracias al alias en la vista se puede acceder a ese dato con una expresión como ésta:

```
{{ vm.otroDato }}
```

Este alias del scope facilita el trabajo cuando se está trabajando con varios ámbitos.

2.3.4 Ámbito raíz con \$rootScope

En AngularJS existe un ámbito o scope raíz, sobre el que se pueden insertar datos. Ese ámbito es accesible desde cualquier lugar de la aplicación, y es útil para definir valores a los que se puede acceder desde cualquier punto del HTML, independientemente del controlador de la aplicación en el que se esté trabajando. Se puede acceder al scope raíz de Angular desde el controlador, por medio de \$rootScope, que necesita ser inyectado en la función del controlador si se desea usar:

```
.controller('pruebaAppCtrl', function($scope, $rootScope){
    $scope.scopeNormal = "Esto lo coloco en el scope normal de este controlador...";
    $rootScope.scopeRaiz = "Esto está en el scope raíz";
});
```

Al guardar algo en el ámbito raíz, se puede acceder a ese valor, desde la vista, a través del nombre de la propiedad que se ha creado en \$rootScope:

```
<div ng-controller="pruebaAppCtrl">
  {{ scopeNormal }}
  <br />
  {{ scopeRaiz }}
</div>
```

En este ejemplo se accede de la misma manera a un dato que está en el scope de este controlador y a un dato que está en el scope raíz. Ésto sucede porque en AngularJS, si una variable del modelo no se encuentra en el scope actual, el propio sistema busca el dato en el scope padre. Si no, en el padre y así sucesivamente.

2.3.5 Conociendo \$parent

Un problema aparece cuando se tiene un dato con el mismo nombre en dos scopes distintos.

```
.controller('pruebaAppCtrl', function($scope, $rootScope){
    //sobreescribo una variable del scope
    //en realidad son dos datos distintos con dos valores distintos
    //uno está en el scope del controlador y el otro en el scope raíz "root"
    $scope.repetido = "Algo en el scope normal";
    $rootScope.repetido = "Algo en el scope raíz";
});
```

En este caso, si en la vista se escribe {{ repetido }} existirá una ambigüedad, pues ese valor puede significar dos cosas, dependiendo de si se revisa un scope u otro. Angular resuelve esa situación devolviendo el dato del scope más específico. De esta manera, retorna el valor que se encuentra en \$scope.repetido y no se puede acceder a \$rootScope.repetido, salvo que se use \$parent.

Por lo tanto, `$parent` permite acceder al scope "padre", en el caso que haya dos controladores, uno dentro de otro. Si sólo hay un controlador, con `$parent` se podrá acceder al scope raíz.

Si se quiere acceder a ambos valores, se puede utilizar el siguiente HTML:

```
<div ng-controller="pruebaAppCtrl">
  {{ repetido }} --- {{ $parent.repetido }}
</div>
```

El primer "repetido" permite acceder al dato que está en el scope actual. Por su parte, con `$parent.repetido`, se puede acceder al dato que está guardado sobre `$rootScope`.

2.3.6 Ejemplo con ámbitos corrientes y ámbito root

Lo visto en este apartado, se puede resumir con el siguiente ejemplo:

HTML

```
div ng-app="pruebaApp" ng-controller="pruebaAppCtrl">
  {{ scopeNormal }}
  <br />
  {{ scopeRaiz }}
  <br />
  {{ algo }} --- {{ $parent.algo }}
</div>
```

Javascript

```
.controller('pruebaAppCtrl', function($scope, $rootScope){
  $scope.scopeNormal = "Esto se coloca en el scope normal de este controlador...";
  $rootScope.scopeRaiz = "Esto está en el scope raíz";

  //si se sobrescribe una variable del scope

  $scope.algo = "Algo en el scope normal";
  $rootScope.algo = "Algo en el scope raíz";
});
```

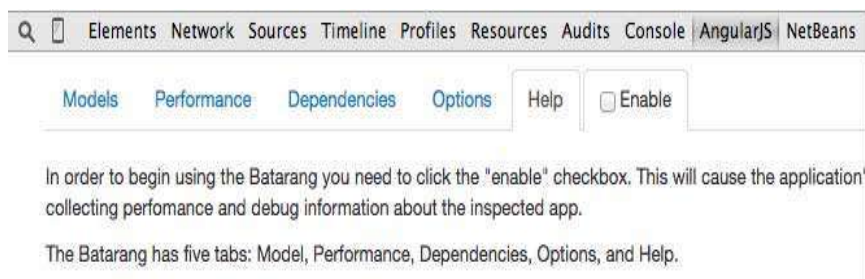
Para apreciar mejor, y de una forma visual, los diferentes ámbitos, se puede utilizar la extensión Angular Batarang.

2.3.7 Angular Batarang

Es una herramienta que se integra en Google Chrome que sirve de inspector de scopes en AngularJS. Permite ver de una manera rápida el contenido de los objetos que se encuentran en los diferentes scopes.

De esta forma, las herramientas para desarrolladores de Google Chrome tienen ahora una nueva pestaña que permite examinar los distintos scopes de una aplicación.

Para instalar el plugin de Batarang, hay que acudir [a la Chrome Web Store](#). Una vez instalado, para comenzar a utilizarlo, se debe activar la nueva pestaña que aparecerá, denominada Angular JS.



Una vez activado, se puede ver qué objetos hay en cada uno de los diferentes \$scopes de la aplicación:



Seleccionando un \$scope en concreto, se puede acceder a los modelos que contiene:

Models for (2)

```
{
  listaFacturas:
    [ {
      id: A
      concepto: mac
      importe: 1000
    }, {
      id: B
      concepto: Iphone
      importe: 700
    } ]
}
```

2.4 Directivas

Las directivas son marcas en los elementos del árbol DOM, en los nodos del HTML, que indican al compilador de Angular que debe asignar cierto comportamiento a dichos elementos o transformarlos según corresponda.

Podríamos decir que las directivas nos permiten añadir comportamiento dinámico al árbol DOM haciendo uso de las nativas del propio AngularJS o extender la funcionalidad hasta donde necesitemos creando las nuestras propias. Se recomienda que sea en las directivas en el único sitio donde se manipule el árbol DOM, para que entre dentro del ciclo de vida de compilación, binding y renderización del HTML.

Las directivas son la técnica que nos va a permitir crear nuestros propios componentes visuales, encapsulando las posibles complejidades en la implementación, normalizando y parametrizándolos según nuestras necesidades.

2.4.1 Tipos de Directivas

Hay varios tipos de directivas, las directivas nativas del propio Angular, y las creadas por nosotros mismos.

Las **directivas nativas** de Angular, comienzan por el prefijo “ng-“. Ejemplos:

1. **ng-app**: Inicializa una aplicación de Angular JS.
2. **ng-controller**: Sirve para enlazar el controlador con la vista.
3. **ng-include**: Se usa para cargar trozos de HTML en la página.
4. **ng-view**: Se utiliza para indicar en qué parte del HTML vamos a inyectar las vistas.
5. **ng-model**: Enlaza datos de la aplicación con el HTML.
6. **ng-submit**: Especifica la función que se debe ejecutar cuando el formulario es enviado.
7. **ng-repeat**: Se encarga de repetir un elemento HTML.
8. **ng-click**: Se utiliza para especificar un evento click, es decir, el código que se debe ejecutar tras hacer click sobre un elemento del HTML en el que se ha incluido esta directiva.
9. **ng-show**: Permite que un elemento de la página esté visible (si está a true) o invisible (si está a false) en función de un valor de nuestro modelo.
10. **ng-cloak**: Esta directiva pausa el navegador, para que Angular realice las acciones necesarias, y a continuación se muestran los resultados esperados.
11. **ng-class**: Nos permite definir clases que posteriormente se aplican a los elementos.

Las **directivas propias** de Angular JS se crean de la forma siguiente:

```
.directive('nombreDirectiva', function(){  
  })
```

Se establece una propiedad llamada Restrict, que define cómo será utilizada la directiva. Puede tomar los siguientes valores:

A: Restringe a la directiva ser adjunta utilizando un atributo por ejemplo,

```
<div nombreDirectiva></div>
```

E: Permite a la directiva ser utilizada cómo un elemento personalizado,

```
<nombreDirectiva></nombreDirectiva>
```

C: Permite a la directiva ser usada añadiéndole una clase al elemento,

```
<div class="nombreDirectiva"></div>
```

M: Permite a la directiva ser ejecutada desde un comentario HTML,

Por defecto AngularJs establece la directiva cómo atributo solamente, pero se pueden usar combinaciones de valores. Por ejemplo:

```
.directive('nombreDirectiva', function(){  
  return {  
    restrict: 'AE'  
  }  
})
```

2.5 Inyección de Dependencias

La Inyección de Dependencias (DI - Dependency Injection) es un pilar fundamental de AngularJS y es que la inyección de dependencias se relaciona con la forma en la que se hacen referencias desde el código.

2.5.1 ¿Qué es la inyección de dependencias?

Es un patrón de diseño orientado a objetos. Este patrón nos dice que los objetos necesarios en una clase serán suministrados y que por tanto no necesitamos que la propia clase cree estos objetos.

El framework de AngularJS gestiona la inyección de dependencias, por lo tanto, las dependencias, como por ejemplo de servicios, serán suministradas por AngularJS. Por ello, al crear un componente de AngularJS se deben especificar las dependencias que se esperan y será el propio framework el que proporcione los objetos que se solicitan. Por ejemplo, si se necesita utilizar un servicio en un controlador, al crear el controlador se debe especificar la dependencia del servicio y no intentar crear un objeto del servicio.

2.6 Servicios

2.6.1 ¿Qué son los servicios?

Los **servicios** son objetos singleton, inyectables por Dependency Injection (inyección de dependencias), donde se define la lógica de negocio de la aplicación, con el objetivo de que sea reutilizable e independiente de las vistas.

Los módulos de Angular exponen 5 métodos para definir servicios:

- constant
- value
- service
- factory
- provider

Constant es un servicio al que se le pasa directamente el valor de dicho servicio. Su principal característica es que se puede inyectar en cualquier sitio. Se define llamando al método constant de un módulo. A dicho método se le pasa el nombre de la constante y su valor.

Value es un objeto javascript que se le pasa al controlador y se instancia en la fase de configuración.

Por otro lado están, provider, factory y service, que permiten crear objetos mucho más complejos que dependen de otros objetos. Cada uno es un caso más concreto del anterior, y como gran elemento diferencial, provider permite generar una API para configurar el servicio resultante.

2.6.2 Ciclo de vida de Angular JS

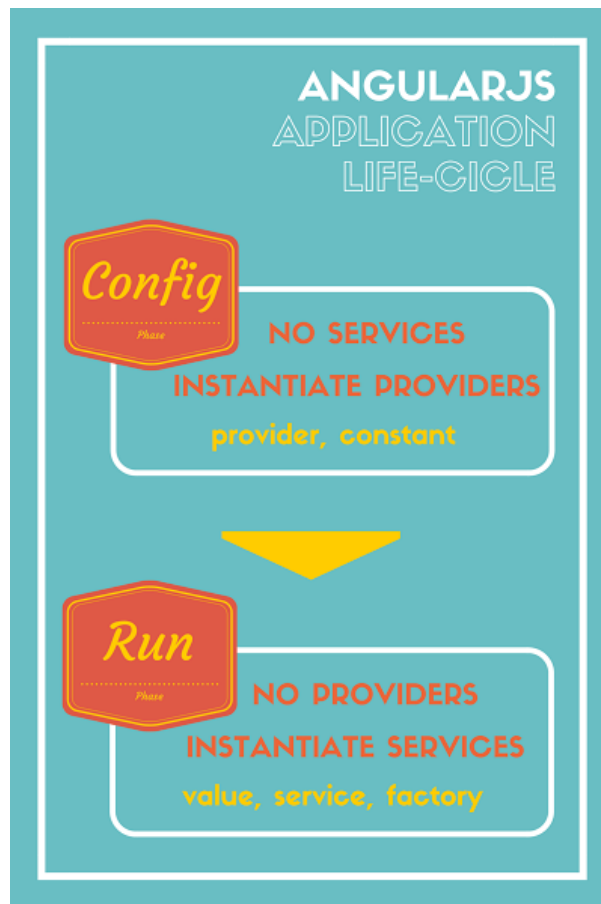
Para entender las diferencias entre cada uno de estos elementos, es importante explicar el ciclo de vida AngularJS. El ciclo de vida de una aplicación en AngularJS se divide en dos fases: la de **configuración** y la de **ejecución**.

Fase de configuración

La fase de configuración se ejecuta en primer lugar. Durante esta fase los servicios aún no pueden instanciarse (no pueden pasarse servicios por inyección de dependencias). El objetivo de esta fase es definir cuál va a ser la configuración de la aplicación a la hora de ejecutarse. Hay que tener precaución si se quiere configurar un servicio en la resolución de un método asíncrono en esta fase, porque el ciclo de configuración podría haber finalizado antes. En esta fase, solo se pueden inyectar constants y providers.

Fase de ejecución

La fase de ejecución es donde se ejecuta toda la lógica de la aplicación y empieza una vez ha concluido la fase de configuración. La interacción entre las vistas, controladores y servicios de la aplicación, sucede en esta fase. En esta fase se pueden inyectar constants, values, services y factories.



2.6.3 Tipos de Servicios y Ejemplos

Constant:

Sirve para almacenar valores simples de cualquier tipo que no deben cambiar, no se pueden inyectar dependencias en su definición, y tampoco es configurable, pero si se puede inyectar en funciones de configuración.

Ejemplo de definición de constante:

```
myApp.constant('SERVERS',{ DEVELOPMENT: "http://localhost:8080/app", PRODUCTION:"http://myDomain.com/app"});
```

Value:

Permite definir objetos simples y primitivas que se pueden inyectar únicamente durante la fase de ejecución. No se pueden inyectar dependencias en su definición ni es configurable.

Ejemplo de definición de value:

```
myApp.value('randomize',function(){
    return Math.floor(Math.random()*10000);
})
myApp.value('token','a1234567890');
myApp.value('User',{'id': 'someId'})
```

Service:

Un servicio es una función constructor que define el servicio. Este servicio se puede inyectar únicamente durante la fase de ejecución. No obstante, si se pueden inyectar dependencias en su definición, aunque no es configurable.

Internamente, Angular utiliza el método new sobre este constructor a la hora de instanciar el servicio, por lo que se le pueden añadir propiedades con **this**. Ese objeto this es exactamente lo que devuelve el servicio.

A continuación, se puede ver un ejemplo de definición de servicio, donde se inyecta una dependencia (el value token del punto anterior):

```
myApp.service('AuthBearer', ['token', function(token) {
    this.authValue = "bearer " + token;
}]);
```


Factory:

Una factoría es un caso más genérico de service, más enfocado a la inicialización del servicio, dado que no devuelve el constructor sino el objeto en sí mismo. Como en el servicio, se puede inyectar únicamente durante la fase de ejecución, y si se pueden inyectar dependencias en su definición, aunque no es configurable.

Un ejemplo de definición sería el siguiente:

```
myApp.factory('apiToken', ['$window', 'clientId', function apiTokenFactory($window, clientId) {
  var encrypt = function(data1, data2) {
    // NSA-proof encryption algorithm:
    return (data1 + ':' + data2).toUpperCase();
  };
  var secret = $window.localStorage.getItem('myApp.secret');
  var apiToken = encrypt(clientId, secret);

  return apiToken;
}]);
```

Y se inyecta como un servicio:

```
myApp.run(['apiToken', function(apiToken){
  console.log(apiToken);
}])
```

Provider

El provider es el caso más genérico de servicio, que además de generar un servicio inyectable durante la fase de ejecución e inyectar dependencias en su definición, proporciona una API para la configuración del servicio antes de que se inicie la aplicación.

Un provider se define de la siguiente forma:

```
myApp.provider('logger', function(){
  var logToConsole = false;

  this.enableConsole = function(flag){
    logToConsole = flag;
  };

  this.$get = function(){
    return {
      debug: function(msg){ if(logToConsole){ console.log(msg);} }
    };
  };
});
```

Donde los métodos de `this` conforman la API de configuración, y el método `this.$get` equivale a una factoría.

Para configurar el servicio `logger`, se tiene que usar su API en la fase de configuración, inyectando el `loggerProvider`:

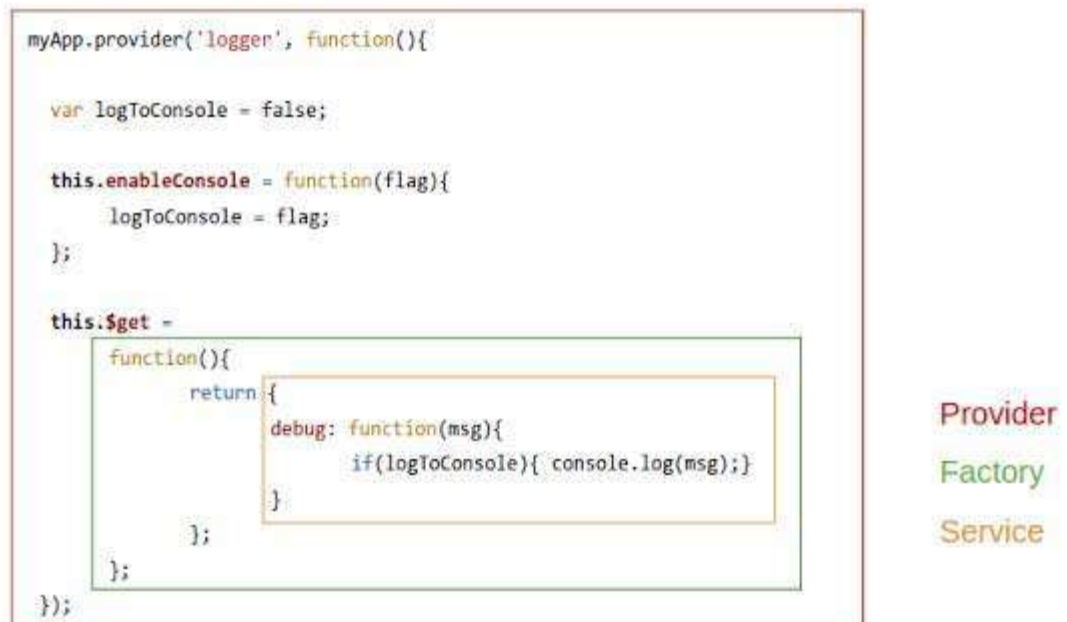
```
myApp.config(['loggerProvider', function(loggerProvider){
  loggerProvider.enableConsole(true);
}])
```

En la fase de ejecución, se utiliza el servicio `logger`:

```
myApp.run(['logger', function(logger){
  logger.debug('Hello world');
}])
```

Relación provider/factory/service

Se puede decir que la relación entre los distintos métodos que existen para crear servicios complejos, es la que ilustra el siguiente esquema:



2.7 Promesas en Angular JS

2.7.1 ¿Qué son las promesas?

Una promesa es un objeto, que actúa como proxy en los casos en los que no se puede retornar el verdadero valor porque aún no se conoce, pero no se puede bloquear la función esperando a que llegue.

Por ejemplo, al hacer una llamada Ajax con `$http`, la llamada a `$http` no retorna ningún valor ya que aún no tiene dicho valor, pero tampoco se puede bloquear esperando a que llegue. En realidad el servicio `$http` sí que retorna un valor. Lo que retorna es una promesa, y la promesa es un proxy que en un futuro contendrá el valor.

La promesa tendrá en un futuro el valor, pero para saber en qué momento la promesa tendrá el valor, se usa una función callback para trabajar asincrónicamente. Las funciones callback son piezas de código ejecutable que se pasan como argumentos a otro código. Este último es el encargado de ejecutar este argumento cuando sea posible.

2.7.2 Servicio `$q`

El servicio de `$q` es un servicio de AngularJS que contiene toda la funcionalidad de las promesas. Está basado en la implementación de Kris Kowal. AngularJS ha hecho su propia versión para que esté todo integrado en el propio framework.

Se puede comparar el sistema de promesas al problema del productor-consumidor. La similitud es que hay una parte que generará la información, por ejemplo el método `$http` y otra parte que consumirá la información, por ejemplo el código de la aplicación que se está realizando. Esta separación es importante ya que hay 2 objetos con los que se tiene que tratar.

En la nomenclatura de AngularJS al productor se le llama `deferred` y al consumidor se le llama `promise`. Mediante el servicio de `$q`, se obtiene el objeto `deferred` llamando al método `defer()`, y a partir de él, se obtiene el objeto `promise` llamando a la propiedad `promise`.

Una vez creada mediante el método `defer()`, una promesa se puede encontrar en alguno de los siguiente 3 estados:

- **Pendiente:** Aún no se sabe si se podrá o no obtener el resultado.
- **Resuelta:** Se ha podido obtener el resultado. Se llega a este estado llamando al método `deferred.resolve()`.
- **Rechazada:** Ha habido algún tipo de error y no se ha podido obtener el resultado. Se llega a este estado llamando al método `deferred.reject()`.

2.8 Módulo ngRoute

2.8.1 ¿Qué es y para qué sirve?

Es el módulo de Angular JS que permite crear rutas profundas en la aplicación e intercambiar vistas dependiendo de la ruta.

El módulo ("module" en la terminología anglosajona de Angular) ngRoute es un potente paquete de utilidades para configurar el enrutado y asociar cada ruta a una vista y un controlador. Sin embargo este módulo no está incluido en la distribución de base de Angular, sino que para usarlo hay que instalarlo y luego inyectarlo como dependencia en el módulo principal de la aplicación.

Instalación de ngRoute:

Para instalarlo hay que incluir el script del código Javascript del módulo ngRoute:

```
<script src="angular-route.js"></script>
```

Este script se tiene que incluir después de haber incluido el script principal de Angular JS.

Inyección de dependencias:

El segundo paso es inyectar la dependencia con ngRoute en el módulo general de la aplicación. Esto se hace en la llamada al método module() con el que se inicia cualquier programa AngularJS, indicando el nombre de las dependencias a inyectar en un array:

```
angular.module("app", ["ngRoute"])
```

Configurar el sistema de enrutado con \$routeProvider:

El sistema de enrutado de AngularJS permite configurar las rutas que se quieren crear en la aplicación de una manera declarativa.

Tiene un par de métodos: el primero es when(), que sirve para indicar qué se debe hacer en cada ruta que se desee configurar, y el método otherwise(), que sirve para marcar un comportamiento cuando se intente acceder a cualquier otra ruta no declarada.

Esta configuración se debe realizar dentro del método `config()`, que pertenece al modelo. De hecho, solo se puede inyectar `$routeProvider` en el método `config()` de configuración:

```
angular.module("app", ["ngRoute"])
  .config(function($routeProvider){
    //configuración y definición de las rutas
  });
```

Las rutas se configuran por medio del método `when()` que recibe dos parámetros. Por un lado la ruta que se está configurando y por otro lado un objeto que tendrá los valores asociados a esa ruta. Los fundamentales son:

- "controller", para indicar el controlador.
- "templateUrl", indica el nombre del archivo, o ruta, donde se encuentra el HTML de la vista que se debe cargar cuando se acceda a la ruta.

2.8.2 Ejemplo de configuración de rutas

```
angular.module("app", ["ngRoute"])
  .config(function($routeProvider){
    $routeProvider
      .when("/", {
        controller: "appCtrl",
        templateUrl: "home.html"
      })
      .when("/descargas", {
        controller: "appCtrl",
        templateUrl: "descargas.html"
      })
      .when("/opciones", {
        controller: "appCtrl",
        templateUrl: "opciones.html"
      });
  })
  .controller("appCtrl", function(){
    //código del controlador
  });
<script src="app.js"></script>
</body>
```

En este ejemplo se ha utilizado un único controlador, pero cada vista puede tener su propio controlador.

2.9 Cómo descargar Angular JS

Acceder a la página oficial de Angular JS:

<https://angularjs.org/>



Download AngularJS

Branch: 1.5.x (stable) 1.2.x (legacy)

Build: Minified Uncompressed Zip

CDN: `https://ajax.googleapis.com/ajax/libs/angularjs/1.5.8/angular.min.js`

Bower: `bower install angular#1.5.8`

npm: `npm install angular@1.5.8`

Extras: [Browse additional modules](#)

[Previous Versions](#) [Download](#)

Incluir el enlace que aparece en CDN en el proyecto a realizar (será necesario estar conectado a internet para poder utilizarlo), o directamente descargar el archivo “angular.min.js” desde la opción Download (no será necesaria una conexión a internet para su utilización).

3. REQUISITOS

3.1 Requisitos funcionales

Los **requisitos funcionales** son los servicios que provee una aplicación. En el caso de JobCar, son los siguientes:

1. Registro:

Es el primer paso a realizar para poder utilizar JobCar. Datos a facilitar:

- Nombre
- Usuario
- Contraseña
- Email

2. Login:

Una vez que el usuario se ha registrado, tendrá que logarse poniendo su usuario y su contraseña. Si los datos introducidos son correctos, accederá a la aplicación.

3. Crear Ruta:

Si el usuario quiere ejercer el rol de conductor, es decir, quiere crear una ruta, únicamente tendrá que rellenar los datos solicitados y pulsar en Guardar Ruta. De esta manera su ruta será publicada en la comunidad de JobCar, y será visible para el resto de usuarios.

4. Yo, Conductor:

Desde esta opción se verán todas las rutas activas que haya creado un usuario.

5. Ver Pasajeros:

El conductor/creador de la ruta, podrá ver los usuarios que se han unido a su ruta.

6. Eliminar Ruta:

Desde esta opción, el conductor/creador de la ruta podrá darla de baja.

7. Ver Ruta:

Se visualizarán las rutas publicadas en JobCar por el resto de usuarios, siempre que tengan vacantes disponibles. No se verán rutas con vacantes igual a cero.

8. Unirse a Ruta:

El usuario podrá unirse a la ruta en la que está interesado, ejerciendo el rol de pasajero. Tras hacerlo, el número de vacantes disponibles disminuirá en una unidad.

9. Yo, Pasajero:

Desde esta opción el usuario podrá ver en que rutas activas está unido.

10. Salirse de Ruta:

El pasajero podrá salirse de la ruta, es decir, dejará de ser pasajero en dicha ruta, y el número de vacantes aumentará en una unidad.

11. Eliminar Pasajero:

El conductor/creador de la ruta, podrá eliminar a uno o más pasajeros de su ruta, por lo que el número de vacantes disponibles aumentará.

3.2 Requisitos no funcionales

Especifican criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos. Se refieren a todos los requisitos que no describen información a guardar, ni funciones a realizar, sino características de funcionamiento. Se han identificado los siguientes:

1. **Disponibilidad:** Debe estar disponible al usuario, permitiéndole el acceso a su información.
2. **Estabilidad:** Se pretende conseguir que el sistema tenga el menor número de fallos posible.
3. **Mantenibilidad:** Se pretende minimizar el esfuerzo necesario para conservar el correcto funcionamiento de la aplicación.
4. **Rendimiento:** Las aplicaciones de una sola página, como JobCar, implementada con Angular JS, permiten tiempos de carga muy cortos.
5. **Escalabilidad:** La aplicación permitirá incorporar nuevas funciones, sin afectar a las ya existentes.

4. ANÁLISIS

4.1 Diagrama Entidad-Relación

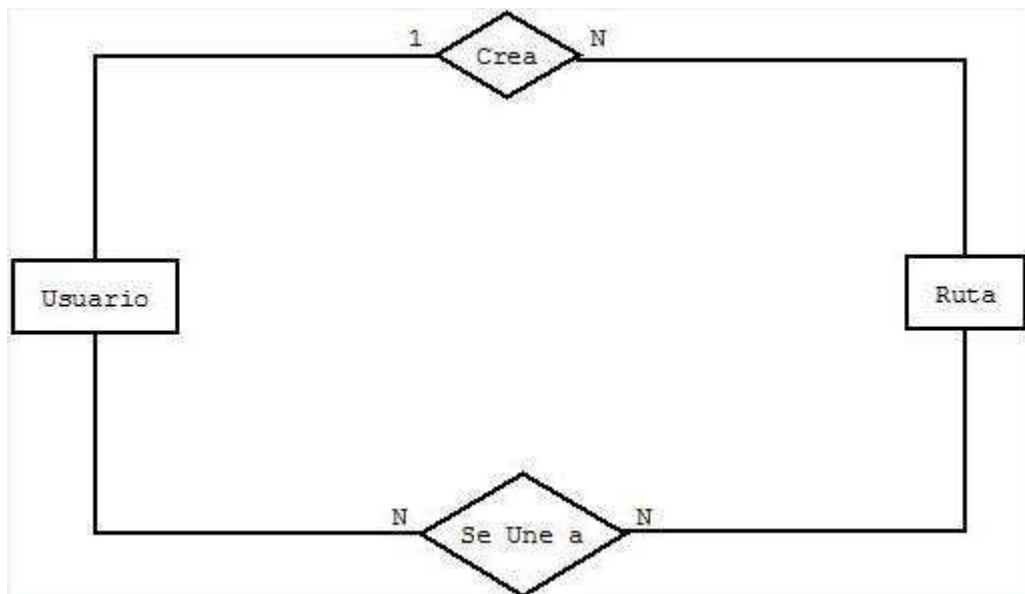
Una vez detectadas las funciones que queremos que realice nuestro sistema, es necesario identificar las **entidades** y las **relaciones** existentes entre ellas, para poder realizar el modelado de nuestra base de datos:

Entidades:

1. **Usuario**: Representado por cada una de las personas que se registren, se loguen, y formen parte de JobCar.
2. **Ruta**: Cada trayecto creado por un usuario de JobCar.

Relaciones:

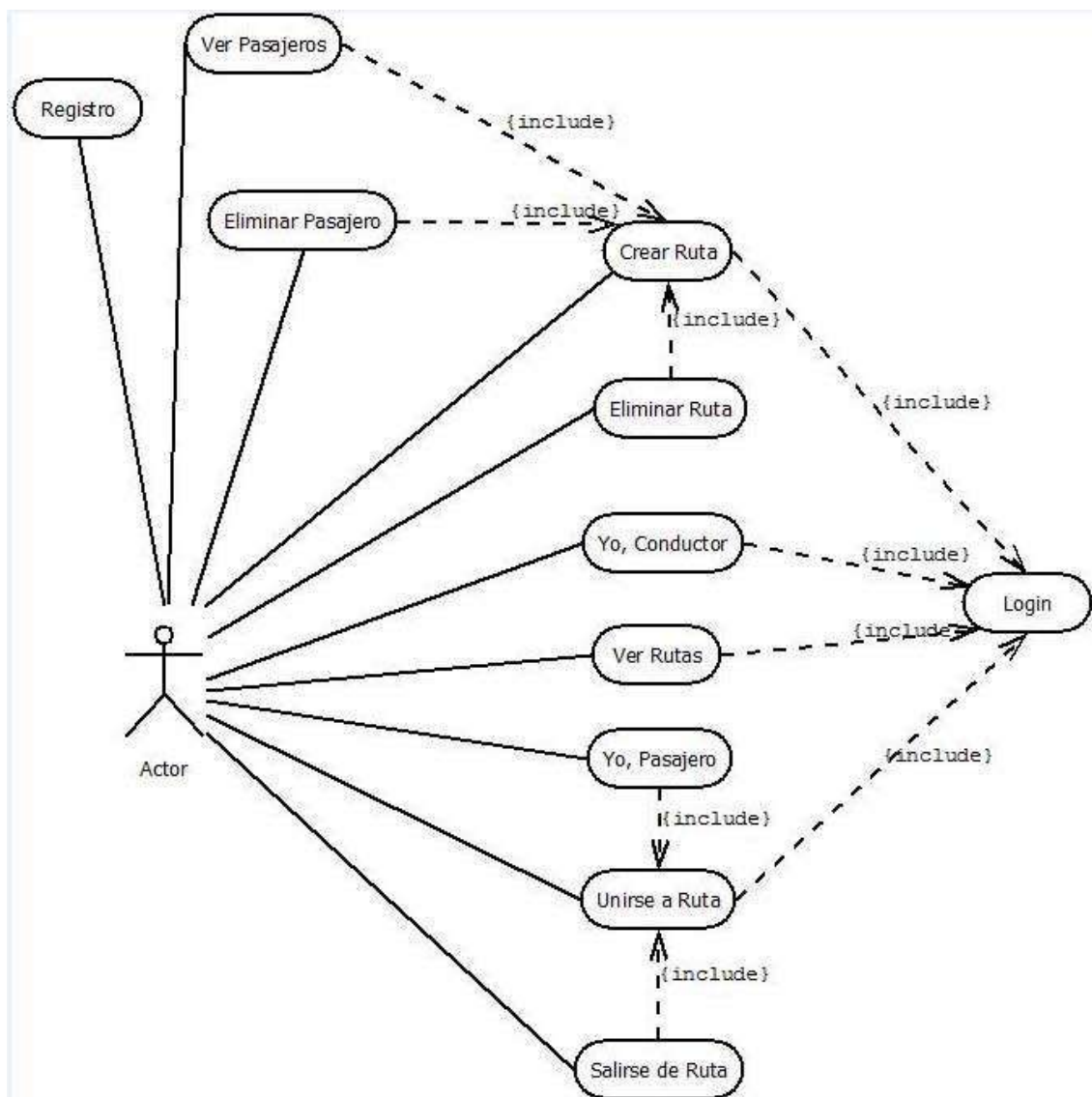
1. **Crea**: Un usuario puede crear N rutas como máximo, y una ruta puede ser creada por 1 usuario como máximo. Esta relación es **1 a N**, no crea tabla.
2. **Se une a**: Un usuario se puede unir a N rutas como máximo, y a una ruta se le pueden unir N usuarios como máximo. Esta relación es **N a N** y crea tabla en la base de datos.



4.2 Casos de Uso

En este punto se procederá a detallar y documentar todas las acciones posibles que pueden llevarse a cabo en la aplicación de JobCar. Para realizar esta tarea, se usará como herramienta el **Modelo de los Casos de Uso** especificado por el **Lenguaje Unificado de Modelado (UML)**.

El **Lenguaje Unificado de Modelado (UML)**, es una metodología utilizada para definir y documentar cada una de las acciones y/o actividades que pueden realizarse, dentro de un sistema de información. En nuestro caso, se emplearán los **Casos de Uso**, para representar a grandes rasgos qué acciones se pueden realizar dentro del sistema.



El actor principal y único es el **usuario** de JobCar.

A continuación se detallará un listado con todos los casos de uso:

Código	Nombre
CU01	Registrar usuario
CU02	Login
CU03	Crear Rutas
CU04	Ver Rutas
CU05	Yo, Conductor
CU06	Ver Pasajeros
CU07	Eliminar Pasajero
CU08	Yo, Pasajero
CU09	Eliminar Ruta
CU10	Unirse a Ruta
CU11	Salirse de Ruta

CU01 – Registrar Usuario

Nombre	Registrar usuario
Actores	Usuario
Objetivo	Registro del usuario en el sistema
Precondiciones	
Postcondiciones	Usuario registrado
Escenario básico	<ol style="list-style-type: none"> 1. El usuario carga la página web, y se dirige a la opción para realizar el registro. 2. Se solicita que rellene la información correspondiente. 3. El usuario introduce los datos solicitados. 4. El sistema valida los campos. 5. Se guardan los datos, quedando el usuario registrado en JobCar.
Escenario alternativo	Si el usuario ya existe en el sistema, se le mostrará un mensaje de error informándole de lo ocurrido, y no se realizará el registro en JobCar.

CU02 – Login

Nombre	Login
Actores	Usuario
Objetivo	Logado del usuario en el sistema
Precondiciones	Usuario registrado en el sistema
Postcondiciones	Sesión abierta
Escenario básico	<ol style="list-style-type: none"> 1. El usuario carga la página de JobCar. 2. El sistema solicita que introduzca el usuario y la contraseña. 3. El usuario pulsa en el botón “Entrar”. 4. El sistema comprueba los datos introducidos. 5. Accede a JobCar, al haber introducido los datos correctamente.
Escenario alternativo	<ol style="list-style-type: none"> 1. El usuario carga la página de JobCar. 2. El sistema solicita que introduzca el usuario y la contraseña. 3. El usuario pulsa en el botón “Entrar”. 4. El sistema comprueba los datos introducidos. 5. Aparece un mensaje de error, indicando que los datos de acceso son incorrectos.

CU03 – Crear Rutas

Nombre	Crear Rutas
Actores	Usuario
Objetivo	Crear una ruta en JobCar
Precondiciones	Usuario registrado y logado en el sistema
Postcondiciones	Una nueva ruta quedará registrada en la base de datos.
Escenario básico	<ol style="list-style-type: none"> 1. El usuario pulsa sobre la sección “Crear Rutas”. 2. Rellena los datos necesarios para crear la ruta. 3. Pulsa en en el botón “Guardar Ruta”. 4. Se carga automáticamente la sección “Yo, Conductor”, donde se ven las rutas creadas por el usuario.

CU04 – Ver Rutas

Nombre	Ver Rutas
Actores	Usuario
Objetivo	Ver las rutas disponibles en JobCar
Precondiciones	Usuario registrado y logado en el sistema
Postcondiciones	
Escenario básico	<ol style="list-style-type: none"> 1. El usuario pulsa sobre la sección “Ver Rutas”. 2. Aparecen las rutas con vacantes disponibles de la comunidad de JobCar.
Escenario alternativo	El sistema no muestra ninguna ruta al no haber vacantes disponibles en ninguna ruta.

CU05 – Yo, Conductor

Nombre	Yo, Conductor
Actores	Usuario
Objetivo	Ver las rutas en las que el usuario es el conductor/creador de la ruta.
Precondiciones	Usuario registrado, y logado en el sistema.
Postcondiciones	
Escenario básico	<ol style="list-style-type: none"> 1. El usuario pulsa sobre la sección “Yo, Conductor”. 2. Aparece/n la/s ruta/s creada/s por el usuario.
Escenario alternativo	<ol style="list-style-type: none"> 1. El usuario pulsa sobre la opción “Yo, Conductor”. 2. El sistema no muestra ninguna ruta, al no haber creado ninguna el usuario.

CU06 – Ver Pasajeros

Nombre	Ver Pasajeros
Actores	Usuario
Objetivo	Ver los pasajeros que el usuario tiene en su ruta
Precondiciones	Usuario registrado y logado en el sistema, con al menos una ruta creada.
Postcondiciones	
Escenario básico	<ol style="list-style-type: none"> 1. El usuario pulsa sobre la sección “Yo, Conductor”. 2. Pulsa sobre “Ver Pasajeros” de la ruta. 3. El sistema muestra el/los pasajero/s incluidos en la ruta.
Escenario alternativo	<ol style="list-style-type: none"> 1. El usuario pulsa sobre la sección “Yo, Conductor”. 2. Pulsa sobre “Ver Pasajeros” de la ruta. 3. El sistema no muestra ningún pasajero, ya que ningún usuario se ha unido a la ruta.

CU07 – Eliminar Pasajero

Nombre	Eliminar Pasajero
Actores	Usuario
Objetivo	Eliminar un pasajero que está unido a la ruta
Precondiciones	Usuario registrado y logado en el sistema, con una ruta creada, y al menos 1 pasajero unido a ella.
Postcondiciones	
Escenario básico	<ol style="list-style-type: none"> 1. El usuario pulsa sobre la sección “Yo, Conductor”. 2. Pulsa sobre “Ver Pasajeros” de la ruta. 3. El sistema muestra el/los pasajero/s incluidos en la ruta. 4. Al presionar sobre el botón “Eliminar Pasajero”, el pasajero desaparece de la pantalla, y se aumenta en una unidad el número de vacantes de esta ruta. El pasajero eliminado recibirá una notificación por email informándole que el conductor le ha dado de baja.

CU08 – Yo, Pasajero

Nombre	Yo, Pasajero
Actores	Usuario
Objetivo	Ver las rutas en las que el usuario es pasajero
Precondiciones	Usuario registrado y logado en el sistema, unido al menos a una ruta.
Postcondiciones	
Escenario básico	<ol style="list-style-type: none"> 1. El Usuario pulsa sobre la sección “Yo, Pasajero”. 2. Aparece/n la/s ruta/s en la/s que el usuario es pasajero.

CU09 – Eliminar Ruta

Nombre	Eliminar Ruta
Actores	Usuario
Objetivo	Eliminar una ruta creada por el usuario
Precondiciones	Usuario registrado y logado en el sistema, con al menos una ruta creada.
Postcondiciones	
Escenario básico	<ol style="list-style-type: none"> 1. El usuario pulsa sobre la sección “Yo, Conductor”. 2. Pulsa en el botón “Eliminar” de una de sus rutas. 3. La ruta desaparece de esta pantalla.

CU10 – Unirse a Ruta

Nombre	Unirse a Ruta
Actores	Usuario
Objetivo	El usuario se une a una ruta de JobCar (ejercerá el rol de pasajero)
Precondiciones	Usuario registrado y logado en el sistema
Postcondiciones	Que la ruta no sea eliminada por su creador/conductor
Escenario básico	<ol style="list-style-type: none"> 1. El usuario pulsa sobre la sección “Ver Rutas”. 2. Presiona el botón “Unirse” de la ruta en la que esté interesado. 3. Se carga automáticamente la sección “Yo, Pasajero”, donde se ve/n la/s ruta/s en la/s que el usuario es pasajero, y se disminuye en una unidad el número de vacantes de esta ruta. El conductor de la ruta recibirá una notificación por email, donde se le informará que tiene un nuevo pasajero.

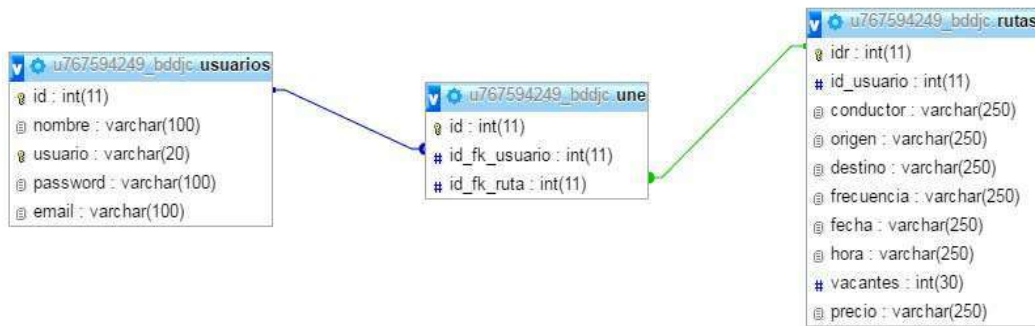
CU11 – Salirse de Ruta

Nombre	Salirse de Ruta
Actores	Usuario
Objetivo	El usuario se sale de una ruta
Precondiciones	Usuario registrado y logado en el sistema, unido al menos a una ruta
Postcondiciones	
Escenario básico	<ol style="list-style-type: none"> 1. El usuario pulsa sobre la sección “Yo, Pasajero”. 2. Presiona el botón “Salirse” de la ruta correspondiente. 3. Aumenta en una unidad el número de vacantes de esta ruta. El conductor de la ruta recibirá una notificación por email, donde se le informará que un pasajero ha abandonado la ruta.

5. DISEÑO

5.1 Paso a tablas

Tras la fase de análisis, se realiza el paso a tablas, y finalmente con la solución planteada, se obtienen 3 tablas:



1. **Usuarios:** Donde se recoge la información de los usuarios registrados en JobCar. Para cada usuario, se almacenará:
 - id (clave numérica auto-incremental)
 - nombre
 - usuario (que será único)
 - password
 - email
2. **Rutas:** En esta tabla se almacenan las rutas creadas por los usuarios. Para cada ruta, se almacenará:
 - idr (id de la ruta, clave numérica auto-incremental)
 - id_usuario (clave foránea procedente de la tabla usuarios)
 - conductor (será el nombre del usuario)
 - origen
 - destino
 - frecuencia
 - fecha
 - hora
 - vacantes
 - precio

3. **Une:** Cuando un usuario se une a una ruta, el id del usuario y el id de la ruta quedan recogidos en esta tabla.
- id (clave numérica auto-incremental)
 - id_fk_usuario (clave foránea procedente de la tabla usuarios → id)
 - id_fk_ruta (clave foránea procedente de la tabla rutas → idr)

Imagen de la base de datos de JobCar en Hostinger:

The screenshot shows the Hostinger database interface for a local database named 'u767594249_bddjc'. It displays the structure of the 'rutas' table, which has three columns: 'id', 'id_fk_usuario', and 'id_fk_ruta'. The interface includes tabs for 'Estructura', 'SQL', 'Buscar', 'Generar una consulta', and 'Exportar'. Below the table structure, there are icons for 'Examinar', 'Estructura', 'Buscar', 'Insertar', 'Vaciar', and 'Eliminar' for each table. The 'rutas' table is highlighted, and it shows 3 tables and the number of rows.

Tabla Usuarios:

+ Opciones

<div>← T →</div>		id	nombre	usuario	password	email		
<input type="checkbox"/>	Editar	Copiar	Borrar	1	Rafael Gutiérrez	FITA	12	rafinene@gmail.com
<input type="checkbox"/>	Editar	Copiar	Borrar	2	Marcos Martín	MARC	12	marc321@gmail.com
<input type="checkbox"/>	Editar	Copiar	Borrar	3	Francisco Jiménez	PACO	12	pacojim53@gmail.com
<input type="checkbox"/>	Editar	Copiar	Borrar	4	Luis García	LUIS1987	12	luis1987@gmail.com

↑

Marcar todos / Desmarcar todos

Para los elementos que están marcados:

Cambiar

Borrar

Tabla Rutas:

				idr	id_usuario	conductor	origen	destino	frecuencia	fecha	hora	vacantes	precio
<input type="checkbox"/>	Editar	Copiar	Borrar	112	1	Rafael Gutiérrez	C/ Albasanz 12	Avenida de Maspalomas 33	L - V	20/02/2017	08:00	2	1 euro
<input type="checkbox"/>	Editar	Copiar	Borrar	114	1	Rafael Gutiérrez	C/ Dublin 12	Avenida Europa 36	L - V	20/11/2016	09:00	2	1 euro
<input type="checkbox"/>	Editar	Copiar	Borrar	123	1	Rafael Gutiérrez	Calle Lope de Vega 13	Calle San Marcos 15	L - V	01/12/2016	08:00	3	1 euro
<input type="checkbox"/>	Editar	Copiar	Borrar	116	2	Marcos Martín	Hacienda de Pavones 6	Glorieta de Bilbao 12	L - V	10/11/2016	07:00	3	1 euro
<input type="checkbox"/>	Editar	Copiar	Borrar	118	2	Marcos Martín	Avenida de Los Poblados 23	Ronda de Atocha 10	L - V	01/12/2016	08:00	2	1 euro
<input type="checkbox"/>	Editar	Copiar	Borrar	111	3	Francisco Jiménez	C/ Fuente de la Hoz 20	Avenida de la Albufera 120	L - V	10/11/2016	10:00	2	1 euro
<input type="checkbox"/>	Editar	Copiar	Borrar	119	3	Francisco Jiménez	Calle Preciados 13	Ronda de Andalucía 2	L - V	15/11/2016	07:00	3	1 euro
<input type="checkbox"/>	Editar	Copiar	Borrar	121	3	Francisco Jiménez	Paseo de la Castellana 20	Calle Miguel Hernández 26	L - V	28/11/2016	08:00	3	1 euro
<input type="checkbox"/>	Editar	Copiar	Borrar	128	4	Luis García	Avenida de los Reyes Católicos (28040-Madrid)	Calle Fuencarral 50 (28004-Madrid)	L - V	15/01/2017	08:00	3	0,50 euros

Tabla Une:

				id	id_fk_usuario	id_fk_ruta
<input type="checkbox"/>	Editar	Copiar	Borrar	142	1	111
<input type="checkbox"/>	Editar	Copiar	Borrar	143	3	118
<input type="checkbox"/>	Editar	Copiar	Borrar	147	2	112
<input type="checkbox"/>	Editar	Copiar	Borrar	148	2	114

5.2 Controladores, Vistas y Modelos utilizados

En este apartado se va a indicar, para cada controlador, el modelo de datos que utiliza, y la vista que tiene asociada:

- **controladorBienvenida:** Se encarga de mostrar un mensaje de bienvenida al usuario que accede a la aplicación. Tiene como modelo de datos 2 propiedades del scope, “visible” y “nombre”. El archivo “bienvenida.html” es la vista asociada.
- **controladorConductor:** Dentro de este controlador, se realiza un acceso a la base de datos de JobCar, para mostrar las rutas en las que el usuario que se ha logado, figura como conductor. Dichas rutas, se almacenan en la propiedad del scope “rutas”, que es el modelo de datos. El archivo “conductorRutas.html” es la vista asociada.
- **controladorCrearRutas:** Este controlador, permite almacenar una ruta en la base de datos de JobCar. La información que se recoge del formulario, se almacena en la propiedad del scope llamada “ruta”, que es el modelo de datos. Tiene como vista asociada el archivo “crearRutas.html”.
- **controladorFinSesion:** Se encarga de realizar el logout de la aplicación. Tiene como modelo de datos una propiedad del rootScope llamada “loginValido”. El archivo “login.html” es la vista asociada.
- **controladorGestionaFoco:** Con este controlador se consigue, que al pulsar sobre una opción del menú, ésta aparezca con el foco activo. Tiene como modelo de datos una propiedad del scope llamada “focoActivo”. Este controlador realiza su función en el archivo “menu.html”.
- **controladorInicial:** Este controlador se encarga de gestionar el login y el registro en la aplicación. Tiene como modelo de datos, 2 propiedades del scope, “dato” y “registro”. Es el único controlador de JobCar que tiene 2 vistas asociadas, los archivos “login.html” y “registro.html”.
- **controladorPasajero:** En este controlador, se realiza un acceso a la base de datos de JobCar, para mostrar las rutas en las que el usuario que se ha logado, figura como pasajero. El modelo de datos, es una propiedad del scope llamada “rutas”. El archivo “pasajeroRutas.html” es la vista asociada.

- **controladorVerPasajeros:** Dentro de este controlador, se realiza un acceso a la base de datos de JobCar, para mostrar los pasajeros de la ruta creada por el usuario logado en la aplicación. El modelo de datos, es una propiedad del scope llamada “pasajeros”. Tiene como vista asociada el archivo “verPasajeros.html”.
- **controladorVerRutas:** Este controlador realiza un acceso a la base de datos de JobCar, para mostrar las rutas con vacantes disponibles creadas por otros usuarios, para que el usuario logado en la aplicación, se pueda unir a ellas. Tiene como modelo de datos una propiedad del scope llamada “rutas”. El archivo “verRutas.html” es la vista asociada.

5.3 Ejemplos del diseño y del código desarrollado

5.3.1 Diseño de Jobcar, una aplicación web de una sólo página

Angular JS permite diseñar aplicaciones de una sólo página, tal y como está explicado en el apartado 1.1. Para lograr este objetivo, el archivo “index.html” tendrá 4 partes diferenciadas:

- Header
- Menú
- Vistas
- Footer

JobCar
La mejor forma para desplazarse al trabajo

Inicio Crear Rutas Ver Rutas Yo, Conductor Yo, Pasajero Salir FITA

Rutas disponibles en JobCar
Únete a la ruta que quieras

Conductor	Origen	Destino	Frecuencia	Fecha de inicio	Hora	Vacantes	Precio	Opciones
Marcos Martín	Hacienda de Pavones 6	Glorieta de Bilbao 12	L - V	10/11/2016	07:00	3	1 euro	Unirse
Marcos Martín	Avenida de Los Poblados 23	Ronda de Atocha 10	L - V	01/12/2016	08:00	2	1 euro	Unirse
Francisco Jiménez	Calle Preciados 13	Ronda de Andalucía 2	L - V	15/11/2016	07:00	3	1 euro	Unirse
Francisco Jiménez	Paseo de la Castellana 20	Calle Miguel Hernández 26	L - V	28/11/2016	08:00	3	1 euro	Unirse
Luis García	Avenida de los Reyes Católicos (28040-Madrid)	Calle Fuencarral 50 (28004-Madrid)	L - V	15/01/2017	08:00	3	0,50 euros	Unirse

Anteriores Siguiénte

Todos los derechos reservados, JobCar 2016 ©

Cuando el navegador cargue la url de Jobcar, cargará estas 4 partes. Pero a partir de ese momento, las partes con el marco de color negro (header, menú, y footer) no tendrán que ser recargadas de nuevo. Cuando cambiemos de opción en el menú, lo único que cargará, será la nueva vista, ahorrando trabajo al navegador, y permitiendo que haya fluidez al movernos por la página.

5.3.2 Ejemplo de uso de directivas nativas, ng-include y ng-view

Siguiendo con el ejemplo anterior, para poder cargar el menú en el html, utilizo la directiva nativa **ng-include**, y para ir inyectando las vistas en la página, utilizo la directiva **ng-view**:

```
<div class="container">
  <div ng-include=" 'partes/menu.html' "></div>
</div>

<div class="container">
  <div ng-view></div>
</div>
```

Código ubicado en el archivo "index.html"

5.3.3 Ejemplo de uso de configuración de las rutas

A través del menú, navegaremos por la página, por lo tanto, cada opción del menú irá vinculado a una ruta, y por lo tanto a una vista. En la imagen inferior, podemos comprobar las rutas a las que están vinculadas las opciones del menú. Por ejemplo, la opción "Crear Rutas" tiene vinculada la ruta "/crearruta".

```
<div class="collapse navbar-collapse navbar-ex6-collapse sha" ng-cloak>
  <ul class="nav navbar-nav">
    <li ng-class="{active: focoActivo('/')}"><a href="#"><span class="glyphicon glyphicon-home"></span></a></li>
    <li ng-show="loginValido" ng-class="{active: focoActivo('/crearruta')}"><a href="#/crearruta">
      <span> Crear Rutas</span></a></li>
    <li ng-show="loginValido" ng-class="{active: focoActivo('/verruta')}"><a href="#/verruta"><span>
      <span> Ver Rutas</span></a></li>
    <li ng-show="loginValido" ng-class="{active: focoActivo('/conductor')}"><a href="#/conductor">
      <span> Yo, Conductor</span></a></li>
    <li ng-show="loginValido" ng-class="{active: focoActivo('/pasajero')}"><a href="#/pasajero"><span>
      <span> Yo, Pasajero</span></a></li>
    <li ng-show="loginValido" ng-class="{active: focoActivo('/finsesion')}"><a href="#/finsesion">
      <span> Salir</span></a></li>
    <li ng-show="loginValido"><a><b>{{ nombreUsuario }}</b></a></li>
```

Código ubicado en el archivo "menu.html"

En la imagen inferior podemos comprobar la configuración de las rutas. Por ejemplo, cuando estemos en la ruta “/bienvenida”, se cargará la vista “bienvenida.html” y el controlador encargado de manejar la lógica de esta vista será el “controladorBienvenida”:

```
.when('/bienvenida', {
    templateUrl: 'partes/bienvenida.html',
    controller: 'controladorBienvenida'
})
.when('/crearruta', {
    templateUrl: 'partes/crearRutas.html',
    controller: 'controladorCrearRutas'
})
.when('/verruta', {
    templateUrl: 'partes/verRutas.html',
    controller: 'controladorVerRutas'
})
.when('/conductor', {
    templateUrl: 'partes/conductorRutas.html',
    controller: 'controladorConductor'
})
.when('/verpasajeros/:id', {
    templateUrl: 'partes/verPasajeros.html',
    controller: 'controladorVerPasajeros'
})
})
```

Código ubicado en el archivo “app.configRutas.js”

5.3.4 Ejemplo de uso del scope

En las 2 imágenes siguientes se muestra cómo se establece la comunicación, entre el “controladorBienvenida” y la vista “bienvenida.html”, a través del scope:

```
angular
    .module('JobCar')
    .controller('controladorBienvenida', function($scope, pasoIdUsuario){

        $scope.visible = pasoIdUsuario.obtengoLogin();
        $scope.nombre = pasoIdUsuario.obtengoNombre();

    });
```



```
<div id="alineado" ng-show="visible"><h4>¡¡ Login realizado correctamente !!</h4>
<h4>Bienvenido <b>{{ nombre }}</b></h4>
</div><br>
```

5.3.5 Ejemplo de uso del rootScope

Como se explicó en el apartado 1.3.4, además del scope, existe el rootScope, en el que se pueden insertar datos, y su ámbito es accesible desde cualquier lugar de la aplicación.

```
$rootScope.nombreUsuario = data.usuario;
$rootScope.loginValido = pasoIdUsuario.obtengoLogin();
$location.url("/bienvenida");
```

Código ubicado en el archivo "controladorInicial.js"

Gracias al rootScope, puedo acceder a estas variables desde el archivo "menu.html", aunque sea otro el controlador (controladorGestionaFoco) el que gestiona esta vista.

```
<li ng-show="loginValido" ng-class="{active: focoActivo('/finsesion') }">
  span> Salir</a></li>
<li ng-show="loginValido"><a><b>{{ nombreUsuario }}</b></a></li>
</ul>
```

Código ubicado en el archivo "menu.html"

5.3.6 Ejemplo de uso de inyección de dependencias

En Angular JS, existe el servicio \$http que permite realizar la comunicación con el servidor, usando el protocolo http. Para poder usar este servicio dentro del "controladorVerRutas", se tiene que inyectar.

En las imágenes siguientes se puede apreciar, la inyección de la dependencia en el controlador, y su uso posterior en la función "unirseARuta" ubicada dentro de este controlador.

```
angular
  .module('JobCar')
  .controller('controladorVerRutas', function($scope, $http, pasoIdUsuario, $location, $route){

    $http.post('./php/unirseARuta.php', insercion).success(function(datos){

      $location.url("/pasajero");
    });
  });
```

Código ubicado en el archivo "controladorVerRutas.js"


5.3.7 Ejemplo de uso de factoría (factory)

En el desarrollo de JobCar he necesitado crear una factoría llamada “pasoIdUsuario”, mediante la cuál he podido compartir datos entre mis controladores. Esta factoría se encuentra dentro del archivo “app.ConfigRutas.js”, y contiene una serie de métodos.

```
.factory("pasoIdUsuario", function() {
  var id = "";
  var nombre = "";
  var nombreUsuario = "";
  var loginValido = false;
  var cerrarSesion = false;
  var intercambio = {
    obtengoID: function(){
      return id;
    },
    pongoID: function(valor){
      id = valor;
    },
    obtengoLogin: function(){
      return loginValido;
    },
    pongoLogin: function(valor){
      loginValido = valor;
    },
    pongoNombre: function(valor){
      nombre = valor;
    },
    obtengoNombre: function(){
      return nombre;
    },
    pongoVacantes: function(valor){
      numVacantes = valor;
    },
    obtengoVacantes: function(){
      return numVacantes;
    }
  }
  return intercambio;
});
```

Posteriormente, para poder utilizar esta factoría, he tenido que inyectarla en los controladores necesarios.

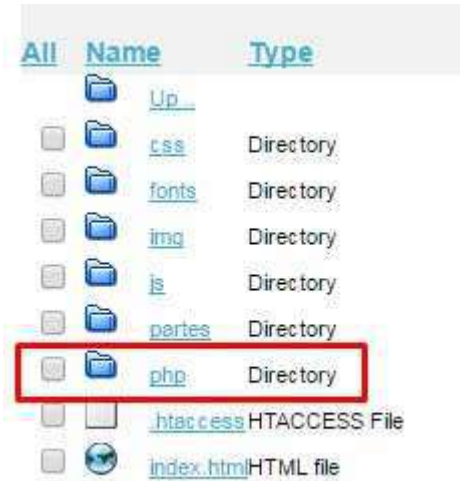
```
angular
  .module('JobCar')
  .controller('controladorPasajero', function($scope,$http,pasoIdUsuario,$route){
```



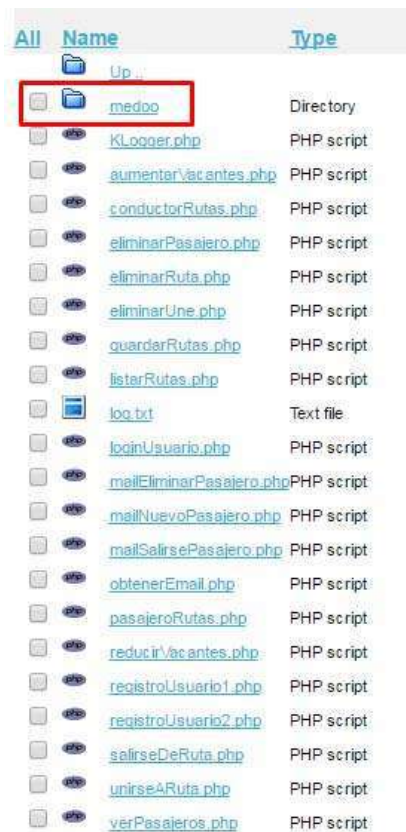
Ejemplo de inyección de la factoría en el controlador “controladorPasajero”

5.4 Configuración en la parte del servidor

En el servidor, todo el trabajo realizado contra la base de datos, se encuentra en el interior de la carpeta php:



Si accedemos al interior de esta carpeta, encontramos la carpeta **medoo**, y todos los archivos, donde se realizan las consultas y las modificaciones en la base de datos:



Tal y como se explicó en el apartado “2.1 Tecnologías”, a través de este framework de php, se han realizado las conexiones con la base de datos de JobCar:

```
$database = new medoo([
    'database_type' => 'mysql',
    'database_name' => 'u767594249_bddjc',
    'server' => 'localhost',
    'username' => 'u767594249_rafa',
    'password' => 'tfq2016',
    'charset' => 'utf8',
]);
```

Las operaciones de consulta/modificación de la base de datos, se han hecho de 2 formas:

1. Utilizando directamente la sintaxis de medoo:

```
$datas = $database->insert(
    'rutas', [
        'id_usuario' => $id_usuario,
        'conductor' => $conductor,
        'origen' => $origen,
        'destino' => $destino,
        'frecuencia' => $frecuencia,
        'fecha' => $fecha,
        'hora' => $hora,
        'vacantes' => $vacantes,
        'precio' => $precio
    ]
);

echo json_encode( $datas );
```

Ejemplo de inserción, realizado en el archivo “guardarRutas.php”:

2. Combinando el lenguaje MySQL, con medoo y php:

```
$datas = $database->query("SELECT rutas.idr, rutas.id_usuario, rutas.conductor, rutas.origen, rutas.destino, rutas.frecuencia, rutas.fecha, rutas.hora, rutas.vacantes, rutas.precio FROM rutas WHERE rutas.id_usuario != ".$id_usuario." AND rutas.vacantes > 0 AND rutas.idr NOT IN (SELECT une.id_fk_ruta FROM une WHERE une.id_fk_usuario = ".$id_usuario.")")->fetchAll(PDO::FETCH_CLASS);

echo json_encode( $datas );
```

Ejemplo de consulta, realizada en el archivo “listarRutas.php”

6. GUÍA DE USUARIO

En las páginas siguientes se va a mostrar una guía para que el usuario de JobCar, pueda conocer paso a paso las opciones disponibles en esta aplicación web:

6.1 Registro

Tras cargar la dirección web de JobCar, el registro en la aplicación es el primer paso a realizar, para ello habrá que pulsar en la opción “Si no eres usuario, regístrate aquí”.



JobCar

La mejor forma para desplazarse al trabajo

[Inicio](#)


Introduce tu usuario y tu contraseña

Entrar

[Si no eres usuario, regístrate aquí](#)

Todos los derechos reservados, JobCar 2016 ©

En la pantalla propia del registro habrá que facilitar el nombre, un código de usuario que ha de ser único en toda la comunidad de JobCar, una contraseña, y el correo electrónico. Tras pulsar en “Enviar”, se comprobará que el usuario sea único, en caso de ser así se dará de alta, y nos llevará automáticamente a la página para hacer el login.




JobCar


La mejor forma para desplazarse al trabajo


[Inicio](#)


Registro en JobCar

Introduzca los siguientes datos











Todos los derechos reservados, JobCar 2016 ©

6.2 Login

Se introduce el usuario, la contraseña, y se pulsa en “Entrar”. Si ambos datos son correctos, el usuario LUIS1987 accederá a la aplicación web de JobCar.

Introduce tu usuario y tu contraseña






Entrar

[Si no eres usuario, regístrate aquí](#)

Todos los derechos reservados, JobCar 2016 ©



JobCar

La mejor forma para desplazarse al trabajo

[Inicio](#)
[Crear Rutas](#)
[Ver Rutas](#)
[Yo, Conductor](#)
[Yo, Pasajero](#)
[Salir](#)

LUIS1987

¡¡ Login realizado correctamente !!

Bienvenido **Luis García**

Todos los derechos reservados, JobCar 2016 ©

6.3 Crear Ruta

Desde esta opción, el usuario puede crear una ruta, para que otros usuarios de JobCar se unan a ella. Tendrá que rellenar los campos solicitados en pantalla, pulsar sobre “Guardar Ruta”, confirmar la acción, y automáticamente se cargará la sección de “Yo, Conductor”, donde podrá ver su ruta/s publicada/s.

CREA TUS RUTAS CON FACILIDAD
Rellena los siguientes datos, y tu ruta formará parte de la comunidad de JobCar

Dirección origen:
C/ Joaquín Maroto 25

Dirección destino:
Avenida de la Paz 48

Frecuencia de la ruta:
L-V

Fecha de inicio:
10/11/2015

Hora:
09:00

Vacantes:
3

Precio:
0,50 euros

Guardar Ruta **Limpiar datos**

Todos los derechos reservados, JobCar 2016 ©

Vas a crear esta ruta en JobCar, estás seguro?

A partir de ahora los usuarios de JobCar, se podrán unir a tu ruta

Cancel **Si, quiero crear mi ruta!**

Todos los derechos reservados, JobCar 2016 ©

6.4 Yo, Conductor

El usuario puede consultar en esta sección las rutas que tiene creadas. Estas rutas son visibles para el resto de usuarios de la comunidad de JobCar desde la opción “Ver Rutas”, para que se puedan unir a ella. El usuario podrá comprobar los pasajeros que se han unido a su ruta, desde la opción “Ver Pasajeros”.



JobCar
La mejor forma para desplazarse al trabajo

Inicio Crear Rutas Ver Rutas **Yo, Conductor** Yo, Pasajero Salir LUIS1987

Éstas son tus rutas como conductor en JobCar
Puedes ver los pasajeros incluidos en tu ruta, o darla de baja si lo deseas

Conductor	Origen	Destino	Frecuencia	Fecha de inicio	Hora	Vacantes	Precio	Opción 1	Opción 2
Luis García	Calle de la Imagen 25	Avenida de las Margaritas 20	L - V	02/10/2016	10:00	3	1 euro	Ver Pasajeros	Eliminar Ruta
Luis García	C/ Joaquín Maroto 25	Avenida de la Paz 48	L - V	10/11/2016	09:00	3	0,50 euros	Ver Pasajeros	Eliminar Ruta

[Anteriores](#) [Sigüientes](#)

Todos los derechos reservados, JobCar 2016 ©

6.5 Ver Pasajeros

Siguiendo el punto anterior, desde esta opción, el conductor/creador de la ruta, podrá ver si tiene usuarios unidos a su ruta.



The screenshot shows the JobCar application interface. At the top, there's a navigation bar with options: Inicio, Crear Rutas, Ver Rutas, Yo, Conductor (selected), Yo, Pasajero, and Salir. Below the navigation bar, a message states: "Éstas son tus rutas como conductor en JobCar. Puedes ver los pasajeros incluidos en tu ruta, o darla de baja si lo deseas". A table lists routes with columns: Conductor, Email, Origen, Destino, Frecuencia, Fecha de Inicio, Hora, Vacantes, Precio, and Opciones. The first row shows a route created by Luis García. In the 'Opciones' column, the 'Ver Pasajeros' button is highlighted with a red arrow.

Conductor	Email	Origen	Destino	Frecuencia	Fecha de Inicio	Hora	Vacantes	Precio	Opciones
Luis García	luis1987@gmail.com	Avenida de la Hispanidad 13	Calle Alcalá 140	L - V	01/10/2016	09:00	2	1 euro	Ver Pasajeros, Eliminar Ruta

En el ejemplo mostrado en la foto inferior, se puede comprobar como Rafael Gutiérrez es el único pasajero de la ruta creada por Luis García.

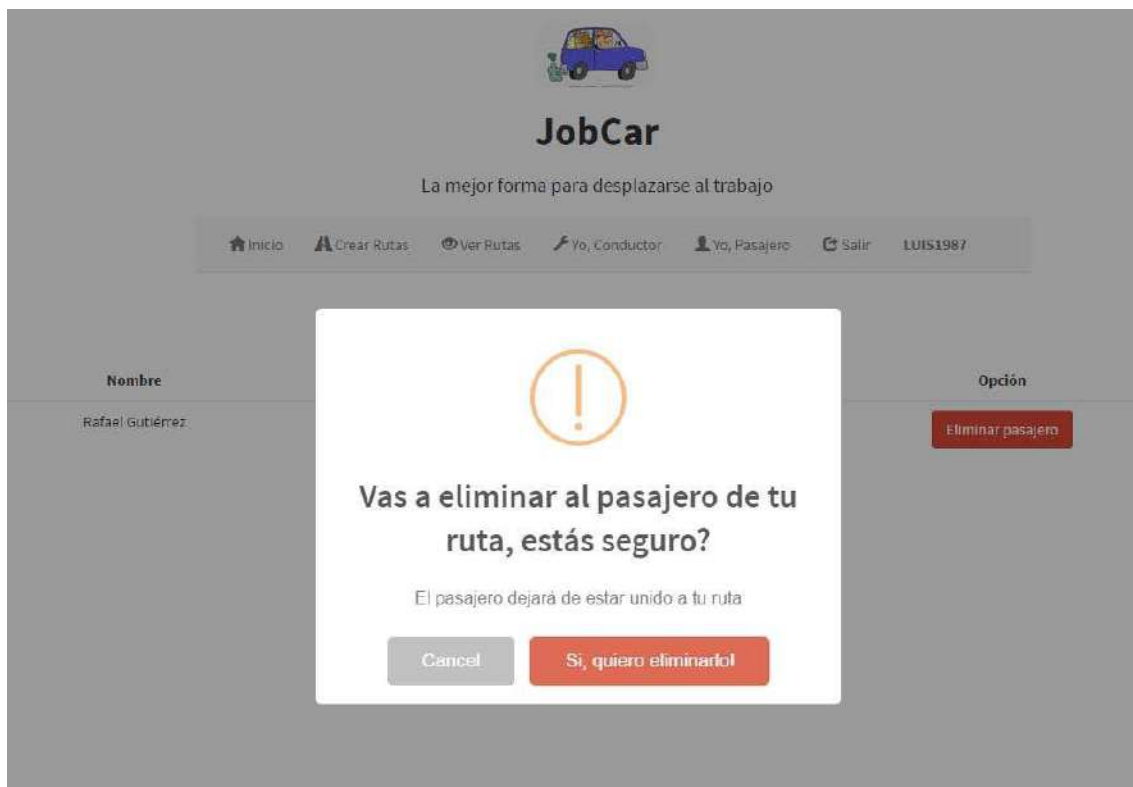


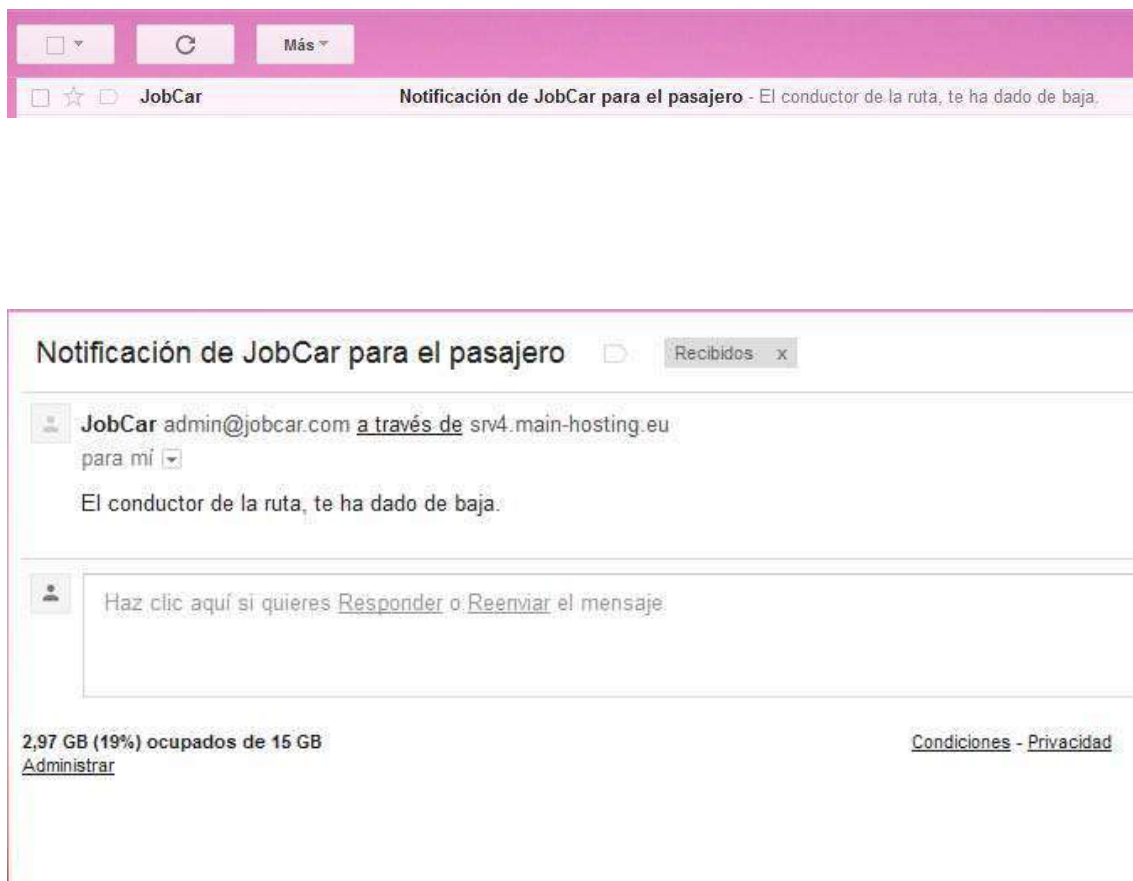
The screenshot shows the 'Ver Pasajeros' page of the JobCar application. The navigation bar is the same as in the previous screenshot. Below the navigation bar, the title 'Ver Pasajeros' is displayed. A table lists passengers with columns: Nombre, Email, and Opción. The first row shows Rafael Gutiérrez. In the 'Opción' column, there is a red button labeled 'Eliminar pasajero'.

Nombre	Email	Opción
Rafael Gutiérrez	rafine@gmail.com	Eliminar pasajero

6.6 Eliminar Pasajero

Si el conductor de la ruta quiere dar de baja a un pasajero, tiene que pulsar sobre “Eliminar Pasajero”, y confirmar la acción. Automáticamente el número de vacantes de esta ruta aumentará en una unidad. El pasajero recibirá una notificación por correo electrónico, donde se le informará que el conductor de la ruta le ha dado de baja.





6.7 Eliminar Ruta

Cuando el creador de la ruta quiere dar de baja su ruta, únicamente tendrá que pulsar sobre el botón “Eliminar Ruta”, y confirmar su acción.



Inicio Crear Rutas Ver Rutas Yo, Conductor Yo, Pasajero Salir LUIS1987

Éstas son tus rutas como conductor en JobCar
Puedes ver los pasajeros incluidos en tu ruta, o darla de baja si lo deseas

Conductor	Origen	Destino	Frecuencia	Fecha de inicio	Hora	Vacantes	Precio	Opción 1	Opción 2
Luis García	Calle de la Imagen 25	Avenida de las Margaritas 20	L - V	01/10/2016	10:00	3	1 euro	Ver Pasajeros	Eliminar Ruta
Luis García	C/ Joaquín Maroto 25	Avenida de la Paz 48	L - V	10/11/2016	09:00	3	0,50 euros	Ver Pasajeros	Eliminar Ruta

Anteriores Sigüientes

Todos los derechos reservados, JobCar 2016 ©



Inicio Crear Rutas Ver Rutas Yo, Conductor Yo, Pasajero Salir LUIS1987

Puedes ver los pasajeros incluidos en tu ruta, o darla de baja si lo deseas.

Vas a eliminar esta ruta, estás seguro?

Esta operación eliminará la ruta de JobCar


Cancel Si, quiero eliminarla!

Conductor	Origen	Destino	Precio	Opción 1	Opción 2
Luis García	Calle de la Imagen 25	Avenida de las Margaritas 20	1 euro	Ver Pasajeros	Eliminar Ruta
Luis García	C/ Joaquín Maroto 25	Avenida de la Paz 48	0,50 euros	Ver Pasajeros	Eliminar Ruta

Anteriores Sigüientes

6.8 Ver Rutas

Desde esta opción se ven las rutas creadas por otros usuarios de JobCar, a las que no se ha unido el usuario logado en la aplicación, y que actualmente tienen vacantes (vacantes > 0).



JobCar

La mejor forma para desplazarse al trabajo

Inicio Crear Rutas Ver Rutas Yo, Conductor Yo, Pasajero Salir LUIS1987

Rutas disponibles en JobCar
Únete a la ruta que quieras


Conductor	Origen	Destino	Frecuencia	Fecha de inicio	Hora	Vacantes	Precio	Opciones
Francisco Jiménez	C/ Fuente de la Hoz 20	Avenida de la Albufera 120	L - V	10/11/2016	10:00	3	1 euro	Unirse
Rafael Gutiérrez	C/ Albasanz 12	Avenida de Maspalomas 33	L - V	20/02/2017	08:00	3	1 euro	Unirse
Rafael Gutiérrez	C/ Dublin 12	Avenida Europa 36	L - V	20/11/2016	09:00	3	1 euro	Unirse
Marcos Martín	Hacienda de Pavones 6	Glorieta de Bilbao 12	L - V	10/11/2016	07:00	3	1 euro	Unirse
Marcos Martín	Avenida de Los Poblados 23	Ronda de Atocha 10	L - V	01/12/2016	08:00	3	1 euro	Unirse
Francisco Jiménez	Calle Preciados 13	Ronda de Andalucía 2	L - V	10/11/2016	07:00	3	1 euro	Unirse
Francisco Jiménez	Paseo de La Castellana 20	Calle Miguel Hernández 26	L - V	20/11/2016	08:00	3	1 euro	Unirse
Rafael Gutiérrez	Calle Lope de Vega 13	Calle San Marcos 15	L - V	01/12/2016	08:00	3	1 euro	Unirse

[Anteriores](#) [Siguientes](#)

Todos los derechos reservados, JobCar 2016 ©

6.9 Unirse a Ruta

Cuando un usuario esté interesado en una ruta, y quiera formar parte de ella para sus desplazamientos al trabajo, sólo tendrá que pulsar sobre el botón “Unirse”, y confirmar su acción. Automáticamente el número de vacantes de esta ruta descenderá en una unidad. El conductor de dicha ruta, recibirá una notificación por correo electrónico, donde se le informará que tiene un nuevo pasajero.



JobCar
La mejor forma para desplazarse al trabajo


Inicio | Crear Rutas | Ver Rutas | Yo, Conductor | Yo, Pasajero | Salir | LUIS1987

Rutas disponibles en JobCar
Únete a la ruta que quieras

Conductor	Origen	Destino	Frecuencia	Fecha de Inicio	Hora	Vacantes	Precio	Opciones
Francisco Jiménez	C/ Fuente de la Hoz 20	Avenida de la Albufera 120	L - V	10/11/2016	10:00	3	1 euro	Unirse
Rafael Gutiérrez	C/ Albasanz 12	Avenida de Maspalomas 33	L - V	20/02/2017	08:00	3	1 euro	Unirse
Rafael Gutiérrez	C/ Dublin 12	Avenida Europa 36	L - V	20/11/2016	09:00	3	1 euro	Unirse
Marcos Martín	Hacienda de Pavones 6	Glorieta de Bilbao 12	L - V	10/11/2016	07:00	3	1 euro	Unirse
Marcos Martín	Avenida de Los Poblados 23	Ronda de Atocha 10	L - V	01/12/2016	09:00	3	1 euro	Unirse
Francisco Jiménez	Calle Preciados 13	Ronda de Andalucía 2	L - V	15/11/2016	07:00	3	1 euro	Unirse
Francisco Jiménez	Paseo de la Castellana 20	Calle Miguel Hernández 26	L - V	28/11/2016	08:00	3	1 euro	Unirse
Rafael Gutiérrez	Calle Lope de Vega 13	Calle San Marcos 15	L - V	01/12/2016	08:00	3	1 euro	Unirse

[Anteriores](#) [Sigüientes](#)

Todos los derechos reservados, JobCar 2016 ©



JobCar
La mejor forma para desplazarse al trabajo

Inicio | Crear Rutas | Ver Rutas | Yo, Conductor | Yo, Pasajero | Salir | LUIS1987

Vas a unirte a esta ruta, estás seguro?

Esta operación te unirá a esta ruta de JobCar

[Cancelar](#) [Sí, quiero unirme!](#)

Conductor	Origen	Destino	Frecuencia	Fecha de Inicio	Hora	Vacantes	Precio	Opciones
Francisco Jiménez	C/ Fuente de la Hoz 20	Avenida de la Albufera 120	L - V	10/11/2016	10:00	3	1 euro	Unirse
Rafael Gutiérrez	C/ Albasanz 12	Avenida de Maspalomas 33	L - V	20/02/2017	08:00	3	1 euro	Unirse
Rafael Gutiérrez	C/ Dublin 12	Avenida Europa 36	L - V	20/11/2016	09:00	3	1 euro	Unirse
Marcos Martín	Hacienda de Pavones 6	Glorieta de Bilbao 12	L - V	10/11/2016	07:00	3	1 euro	Unirse
Marcos Martín	Avenida de Los Poblados 23	Ronda de Atocha 10	L - V	01/12/2016	09:00	3	1 euro	Unirse
Francisco Jiménez	Calle Preciados 13	Ronda de Andalucía 2	L - V	15/11/2016	07:00	3	1 euro	Unirse
Francisco Jiménez	Paseo de la Castellana 20	Calle Miguel Hernández 26	L - V	28/11/2016	08:00	3	1 euro	Unirse
Rafael Gutiérrez	Calle Lope de Vega 13	Calle San Marcos 15	L - V	01/12/2016	08:00	3	1 euro	Unirse

[Anteriores](#) [Sigüientes](#)

Todos los derechos reservados, JobCar 2016 ©



6.10 Yo, Pasajero

Una vez que el usuario se ha unido a la ruta en el paso anterior, podrá ver desde esta opción las rutas en las que es pasajero.



JobCar

La mejor forma para desplazarse al trabajo

Inicio Crear Rutas Ver Rutas Yo, Conductor **Yo, Pasajero** Salir LUIS1987

Éstas son tus rutas como pasajero en JobCar

Puedes salirte de la ruta cuando lo desees

Conductor	Origen	Destino	Frecuencia	Fecha de Inicio	Hora	Vacantes	Precio	Opciones
Francisco Jiménez	C/ Fuente de la Hoz 20	Avenida de la Albufera 120	L - V	10/11/2016	10:00	2	1 euro	Salirse

Anteriores Siguientes

Todos los derechos reservados, JobCar 2016 ©

6.11 Salirse de Ruta

Al pulsar sobre la opción “Salirse”, y confirmar la acción, el usuario dejará de ser pasajero de esta ruta. Automáticamente el número de vacantes de esta ruta aumentará en una unidad. El conductor, recibirá una notificación por correo electrónico, donde se le informará que un pasajero se ha dado de baja.



JobCar
La mejor forma para desplazarse al trabajo


Inicio Crear Rutas Ver Rutas Yo, Conductor Yo, Pasajero Salir LUIS1987

Éstas son tus rutas como pasajero en JobCar
Puedes salirte de la ruta cuando lo desees

Conductor	Origen	Destino	Frecuencia	Fecha de Inicio	Hora	Vacantes	Precio	Opciones
Francisco Jiménez	C/ Fuente de la Hoz 20	Avenida de la Albufera 120	L - V	10/11/2016	10:00	2	1 euro	Salirse

Anteriores Siguientes

Todos los derechos reservados, JobCar 2016 ©



JobCar
La mejor forma para desplazarse al trabajo

Inicio Crear Rutas Ver Rutas Yo, Conductor Yo, Pasajero Salir LUIS1987

Conductor Origen

Francisco Jiménez C/ Fuente de la Hoz 20

Anteriores Siguientes

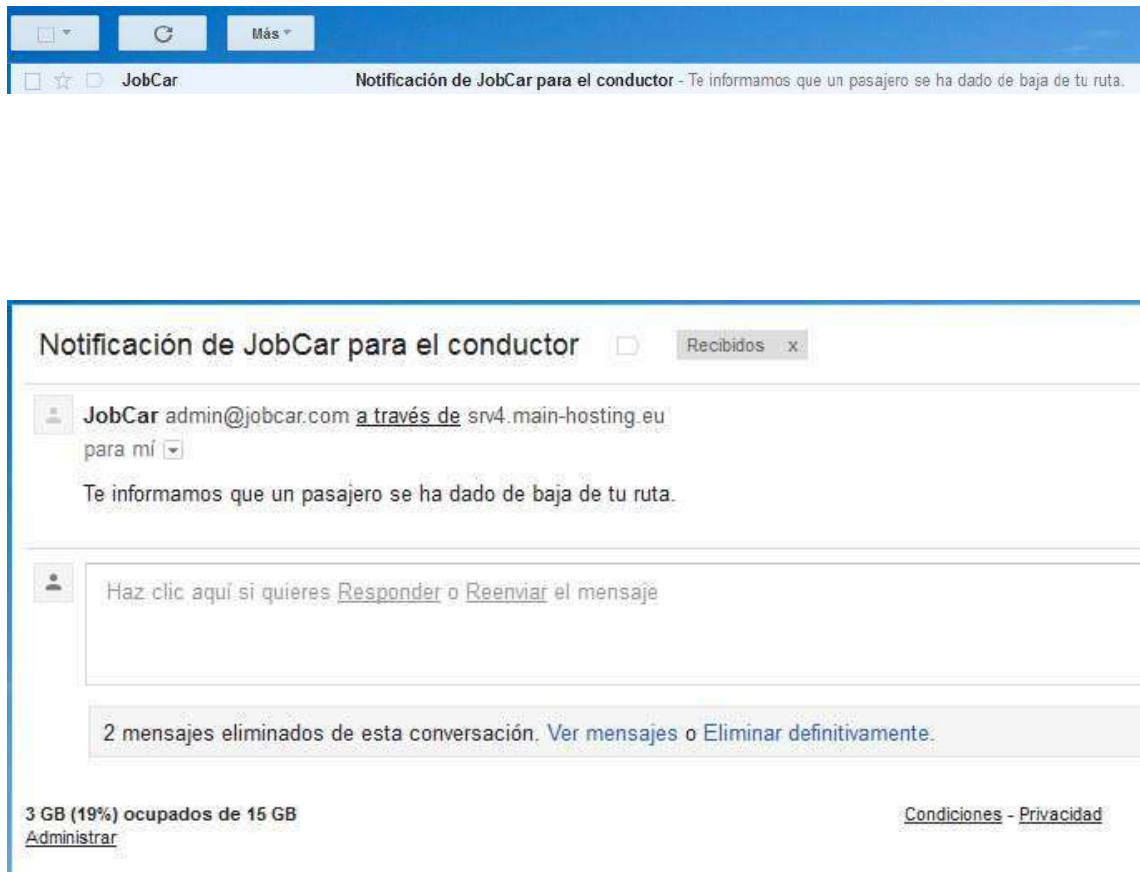
Hora Vacantes Precio Opciones

10:00 2 1 euro Salirse

Vas a salirte de esta ruta, estás seguro?

Dejarás de estar incluido en esta ruta de JobCar

Cancel Si, quiero salirme



7. PRUEBAS

En este apartado, se han realizado 8 pruebas, en las que se aportarán evidencias procedentes de la interfaz de JobCar, y de la base de datos.

En las 3 primeras pruebas, se quiere verificar que sólo los usuarios registrados y correctamente logados, pueden acceder a la aplicación.

Las pruebas restantes, tienen como objetivo comprobar, que tras la ejecución de las opciones clave de JobCar (crear ruta, eliminar ruta, unirse a ruta, salirse de ruta, y eliminar pasajero), el resultado sea el esperado.

Nº de Prueba	Nombre de la prueba	Resultado esperado
1	Alta de usuario que ya existe en la B.D	El sistema no permite realizar el alta

+ Opciones:

	id	nombre	usuario	password	email
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	1	Rafael Gutiérrez	FITA	12	rafinene@gmail.com
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	2	Marcos Martín	MARC	12	marc321@gmail.com
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	3	Francisco Jiménez	PACO	12	pacojim53@gmail.com
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	4	Luis García	LUIS1987	12	luis1987@gmail.com

⬆ Marcar todos / Desmarcar todos Para los elementos que están marcados: ☐ Cambiar ☐ Borrar

Registro en JobCar

Introduzca los siguientes datos

Enviar

Todos los derechos reservados, JobCar 2016 ©

Tras pulsar en el botón enviar:

Registro en JobCar

Introduzca los siguientes datos

Enviar

!! EL USUARIO INTRODUCIDO YA EXISTE !!

Todos los derechos reservados, JobCar 2016 ©


Nº de Prueba	Nombre de la prueba	Resultado esperado
2	Login en JobCar con usuario correcto y password incorrecta	El sistema no permite el acceso a JobCar


+ Opciones

	id	nombre	usuario	password	email
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	1	Rafael Gutiérrez	FITA	12	rafinene@gmail.com
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	2	Marcos Martín	MARC	12	marc321@gmail.com
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	3	Francisco Jiménez	PACO	12	pacojim53@gmail.com
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	4	Luis García	LUIS1987	12	luis1987@gmail.com

☐ Marcar todos / ☐ Desmarcar todos. Para los elementos que están marcados: ☐ Cambiar ☐ Borrar

Introduce tu usuario y tu contraseña







Entrar

[Si no eres usuario, regístrate aquí](#)

Tras pulsar en el botón entrar:

Introduce tu usuario y tu contraseña





Entrar

[Si no eres usuario, regístrate aquí](#)

!! DATOS DE ACCESO INCORRECTOS !!

Todos los derechos reservados, JobCar 2016 ©

Nº de Prueba	Nombre de la prueba	Resultado esperado
3	Login en JobCar con usuario incorrecto	El sistema no permite el acceso a JobCar

+ Opciones

	id	nombre	usuario	password	email
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	1	Rafael Gutiérrez	FITA	12	rafinene@gmail.com
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	2	Marcos Martín	MARC	12	marc321@gmail.com
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	3	Francisco Jiménez	PACO	12	pacojim53@gmail.com
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	4	Luis García	LUIS1987	12	luis1987@gmail.com

⬆️ Marcar todos / Desmarcar todos Para los elementos que están marcados: ☐ Cambiar ☐ Borrar

Introduce tu usuario y tu contraseña

Entrar

[Si no eres usuario, regístrate aquí](#)

Todos los derechos reservados, JobCar 2016 ©

Tras pulsar en el botón entrar:

Introduce tu usuario y tu contraseña


Entrar

[Si no eres usuario, regístrate aquí](#)

!! DATOS DE ACCESO INCORRECTOS !!

Todos los derechos reservados, JobCar 2016 ©

Nº de prueba	Nombre de la prueba	Resultado esperado
4	Crear Ruta	Alta de ruta en base de datos



JobCar

La mejor forma para desplazarse al trabajo

Inicio
Crear Rutas
Ver Rutas
Yo, Conductor
Yo, Pasajero
Salir
LUIS1987

CREA TUS RUTAS CON FACILIDAD

Rellena los siguientes datos, y tu ruta formará parte de la comunidad de JobCar

Dirección origen:
C/ Camino de la Arboleda 12

Dirección destino:
Calle María de Molina 35

Frecuencia de la ruta:
L - V

Fecha de inicio:
20/11/2016

Horas:
08:00

Vacantes:
3

Precio:
1 euro


Guardar Ruta
Limpiar datos

Todos los derechos reservados, JobCar 2016 ©

+ Opciones

	idr	id_usuario	conductor	origen	destino	frecuencia	fecha	hora	vacantes	precio
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	112		1 Rafael Gutiérrez	C/ Albasanz 12	Avenida de Maspelomas 33	L - V	20/02/2017	08:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	114		1 Rafael Gutiérrez	C/ Dublin 12	Avenida Europa 36	L - V	20/11/2016	09:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	123		1 Rafael Gutiérrez	Calle Lopo de Vega 13	Calle San Marcos 15	L - V	01/12/2016	08:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	116		2 Marcos Martín	Hacienda de Pavones 6	Glorieta de Bilbao 12	L - V	10/11/2016	07:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	118		2 Marcos Martín	Avenida de Los Poblados 23	Ronda de Atocha 10	L - V	01/12/2016	08:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	111		3 Francisco Jiménez	C/ Fuente de la Hoz 20	Avenida de la Albufera 120	L - V	10/11/2016	10:00	2	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	119		3 Francisco Jiménez	Calle Preciados 13	Ronda de Andalucía 2	L - V	15/11/2016	07:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	121		3 Francisco Jiménez	Paseo de la Castellana 20	Calle Miguel Hernández 26	L - V	28/11/2016	08:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	126		4 Luis García	C/ Camino de la Arboleda 12	Calle María de Molina 35	L - V	20/11/2016	08:00	3	1 euro

☐ Marcar todos / ☐ Desmarcar todos Para los elementos que están marcados:
 ☐ Cambiar
☐ Borrar
☐ Exportar



JobCar

La mejor forma para desplazarse al trabajo

[Inicio](#) [Crear Rutas](#) [Ver Rutas](#) [Yo, Conductor](#) [Yo, Pasajero](#) [Salir](#) LUIS1987

Éstas son tus rutas como conductor en JobCar

Puedes ver los pasajeros incluidos en tu ruta, o dárla de baja si lo deseas

Conductor	Origen	Destino	Frecuencia	Fecha de inicio	Hora	Vacantes	Precio	Opción 1	Opción 2
Luis Garcia	C/ Camino de la Arboleda 12	Calle Maria de Molina 35	L-V	20/11/2016	08:00	3	1 euro	Ver Pasajeros	Eliminar Ruta

[Anteriores](#) [Sigüientes](#)

Todos los derechos reservados, JobCar 2016 ©

Nº de prueba	Nombre de la prueba	Resultado esperado
5	Eliminar Ruta	Borrado de la ruta en base de datos



JobCar

La mejor forma para desplazarse al trabajo

Inicio
Crear Rutas
Ver Rutas
Yo, Conductor
Yo, Pasajero
Salir
LUIS1987

Éstas son tus rutas como conductor en JobCar

Puedes ver los pasajeros incluidos en tu ruta, o darla de baja si lo deseas

Conductor	Origen	Destino	Frecuencia	Fecha de inicio	Hora	Vacantes	Precio	Opción 1	Opción 2
Luis García	C/ Camino de la Arboleda 12	Calle María de Molina 35	L - V	20/11/2016	08:00	3	1 euro	Ver Pasajeros	Eliminar Ruta

Anteriores
Sigüientes

Todos los derechos reservados, JobCar 2016 ©

Tras pulsar en “Eliminar Ruta”, comprobamos que la ruta es borrada de la base de datos, y desaparece también de la sección “Yo, Conductor” del usuario.

+ Opciones

←
→

Idr.
Id usuario

conductor
origen
destino
frecuencia
fecha
hora
vacantes
precio

112	1	Rafael Gutiérrez	C/ Albasanz 12	Avenida de Maspalomas 33	L - V	20/02/2017	08:00	3	1 euro
114	1	Rafael Gutiérrez	C/ Dublin 12	Avenida Europa 35	L - V	20/11/2016	09:00	3	1 euro
123	1	Rafael Gutiérrez	Calle Lope de Vega 13	Calle San Marcos 15	L - V	01/12/2016	08:00	3	1 euro
116	2	Marcos Martín	Hacienda de Pavones 6	Glorieta de Bilbao 12	L - V	10/11/2016	07:00	3	1 euro
118	2	Marcos Martín	Avenida de Los Poblados 23	Ronda de Atocha 10	L - V	01/12/2016	08:00	3	1 euro
111	3	Francisco Jiménez	C/ Fuente de la Hoz 26	Avenida de la Albufera 120	L - V	10/11/2016	10:00	2	1 euro
119	3	Francisco Jiménez	Calle Prociados 13	Ronda de Andalucía 2	L - V	15/11/2016	07:00	3	1 euro
121	3	Francisco Jiménez	Paseo de la Castellana 20	Calle Miguel Hernández 25	L - V	28/11/2016	08:00	3	1 euro

Marcar todos / Desmarcar todos
Para los elementos que están marcados:
Cambiar
Borrar
Exportar

Inicio
Crear Rutas
Ver Rutas
Yo, Conductor
Yo, Pasajero
Salir
LUIS1987

Éstas son tus rutas como conductor en JobCar

Puedes ver los pasajeros incluidos en tu ruta, o darla de baja si lo deseas

Conductor	Origen	Destino	Frecuencia	Fecha de inicio	Hora	Vacantes	Precio	Opción 1	Opción 2
-----------	--------	---------	------------	-----------------	------	----------	--------	----------	----------

Anteriores
Sigüientes

Todos los derechos reservados, JobCar 2016 ©

Nº de prueba	Nombre de la prueba	Resultado esperado
6	Unirse a Ruta	La ruta aparece en la sección Yo, Pasajero

Inicio
Crear Rutas
Ver Rutas
Yo, Conductor
Yo, Pasajero
Salir
LUIS1987

Rutas disponibles en JobCar
Unete a la ruta que quieras

Conductor	Origen	Destino	Frecuencia	Fecha de Inicio	Hora	Vacantes	Precio	Opciones
Rafael Gutierrez	C/ Albasanz 12	Avenida de Maspalomas 33	L - V	20/02/2017	08:00	3	1 euro	Unirse
Rafael Gutierrez	C/ Dublin 12	Avenida Europa 36	L - V	20/11/2016	09:00	3	1 euro	Unirse
Marcos Martin	Hacienda de Pavones 6	Glorieta de Bilbao 11	L - V	10/11/2016	07:00	3	1 euro	Unirse
Marcos Martin	Avenida de Los Poblados 23	Ronda de Atocha 10	L - V	01/12/2016	08:00	3	1 euro	Unirse
Francisco Jimenez	Calle Preciados 13	Ronda de Andalucía 2	L - V	15/11/2016	07:00	3	1 euro	Unirse
Francisco Jimenez	Paseo de la Castellana 20	Calle Miguel Hernandez 26	L - V	28/11/2016	08:00	3	1 euro	Unirse
Rafael Gutierrez	Calle Lope de Vega 13	Calle San Marcos 15	L - V	01/12/2016	08:00	3	1 euro	Unirse

Anteriores
Siguientes

Todos los derechos reservados, JobCar 2016 ©

Tras pulsar en “Unirse”, la ruta aparece en la sección “Yo, Pasajero”

Inicio
Crear Rutas
Ver Rutas
Yo, Conductor
Yo, Pasajero
Salir
LUIS1987

Éstas son tus rutas como pasajero en JobCar
Puedes salirte de la ruta cuando lo desees

Conductor	Origen	Destino	Frecuencia	Fecha de Inicio	Hora	Vacantes	Precio	Opciones
Rafael Gutierrez	Calle Lope de Vega 13	Calle San Marcos 15	L - V	01/12/2016	08:00	2	1 euro	Salirse

Anteriores
Siguientes

Todos los derechos reservados, JobCar 2016 ©

En la base de datos, podemos comprobar que el campo vacantes disminuye en una unidad, y que se genera la fila correspondiente en la tabla “Une”.

+ Opciones

	idr	id_usuario	conductor	origen	destino	frecuencia	fecha	hora	vacantes	precio
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	112		1 Rafael Gutiérrez	C/ Albasanz 12	Avenida de Maspalomas 33	L - V	20/02/2017	08:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	114		1 Rafael Gutiérrez	C/ Dublin 12	Avenida Europa 36	L - V	20/11/2016	09:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	123		1 Rafael Gutiérrez	Calle Lope de Vega 13	Calle San Marcos 15	L - V	01/12/2016	08:00	2	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	116		2 Marcos Martín	Hacienda de Pavones 6	Glorieta de Bilbao 12	L - V	10/11/2016	07:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	118		2 Marcos Martín	Avenida de Los Poblados 23	Ronda de Atocha 10	L - V	01/12/2016	08:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	111		3 Francisco Jiménez	C/ Fuente de la Hoz 20	Avenida de la Albufera 120	L - V	10/11/2016	10:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	119		3 Francisco Jiménez	Calle Preciados 13	Ronda de Andalucía 2	L - V	15/11/2016	07:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	121		3 Francisco Jiménez	Paseo de la Castellana 20	Calle Miguel Hernández 25	L - V	25/11/2016	08:00	3	1 euro

⬆ Marcar todos / Desmarcar todos Para los elementos que están marcados: ☐ Cambiar ☐ Borrar ☐ Exportar

+ Opciones


	id	id_fk_usuario	id_fk_ruta
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	142	1	111
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	143	3	118
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	141	4	123

⬆ Marcar todos / Desmarcar todos Para los elementos que están

El conductor recibirá una notificación por correo electrónico, informándole que tiene un nuevo pasajero en su ruta.

☐ ☆ □ JobCar Notificación de JobCar para el conductor - Enhorabuena !! Un nuevo pasajero se ha unido a tu ruta.

Notificación de JobCar para el conductor

 JobCar admin@jobcar.com a través de srv4.main-hosting.eu para mí

Enhorabuena !! Un nuevo pasajero se ha unido a tu ruta.

Nº de prueba	Nombre de la prueba	Resultado esperado
7	Salirse de Ruta	La ruta desaparece de la sección “Yo, Pasajero”



Inicio Crear Rutas Ver Rutas Yo, Conductor Yo, Pasajero Salir LUIS1987

Éstas son tus rutas como pasajero en JobCar
Puedes salirte de la ruta cuando lo desees

Conductor	Origen	Destino	Frecuencia	Fecha de Inicio	Hora	Vacantes	Precio	Opciones
Rafael Gutiérrez	Calle Lope de Vega 13	Calle San Marcos 15	L - V	01/12/2016	08:00	2	1 euro	Salirse

Anteriores Siguyentes

Todos los derechos reservados, JobCar 2016 ©

Tras pulsar en “Salirse”, desaparece la ruta de la sección “Yo, Pasajero”, y se elimina la fila correspondiente en la tabla “Une”.



Inicio Crear Rutas Ver Rutas Yo, Conductor Yo, Pasajero Salir LUIS1987

Éstas son tus rutas como pasajero en JobCar
Puedes salirte de la ruta cuando lo desees

Conductor	Origen	Destino	Frecuencia	Fecha de Inicio	Hora	Vacantes	Precio	Opciones
-----------	--------	---------	------------	-----------------	------	----------	--------	----------

Anteriores Siguyentes

Todos los derechos reservados, JobCar 2016 ©



+ Opciones

	id	id_fk_usuario	id_fk_ruta
Editar Copiar Borrar	142	1	111
Editar Copiar Borrar	143	3	118

Marcar todos / Desmarcar todos Para los elementos que están

El conductor recibirá una notificación por correo electrónico, informándole que un pasajero se ha dado de baja de su ruta.

JobCar Notificación de JobCar para el conductor - Te informamos que un pasajero se ha dado de baja de tu ruta.

Notificación de JobCar para el conductor



Recibidos x



JobCar admin@jobcar.com a través de srv4.main-hosting.eu
para mí ▾

Te informamos que un pasajero se ha dado de baja de tu ruta.

Nº de prueba	Nombre de la prueba	Resultado esperado
8	Eliminar Pasajero	Borrado de la fila en la tabla “Une”, y aumento de vacantes en dicha ruta en una unidad

Esta ruta ha sido creada por el usuario LUIS1987, y tiene actualmente 2 vacantes.

[Inicio](#)
[Crear Rutas](#)
[Ver Rutas](#)
[Yo, Conductor](#)
[Yo, Pasajero](#)
[Salir](#)
LUIS1987

Estas son tus rutas como conductor en JobCar

Puedes ver los pasajeros incluidos en tu ruta, o darla de baja si lo deseas

Conductor	Origen	Destino	Frecuencia	Fecha de inicio	Hora	Vacantes	Precio	Opción 1	Opción 2
Luis Garcia	Avenida de los Reyes Católicos (28040-Madrid)	Calle Fuencarral 50 (28004-Madrid)	L - V	15/01/2017	08:00	2	0,50 euros	Ver Pasajeros	Eliminar Ruta

[Anteriores](#)
[Siguietes](#)

Todos los derechos reservados, JobCar 2016 ©

+ Opciones

	id_r	id_usuario	conductor	origen	destino	frecuencia	fecha	hora	vacantes	precio
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	112	1	Rafael Gutiérrez	C/ Albasanz 12	Avenida de Maspalomas 33	L - V	20/02/2017	08:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	114	1	Rafael Gutiérrez	C/ Dublin 12	Avenida Europa 36	L - V	20/11/2016	09:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	123	1	Rafael Gutiérrez	Calle Lope de Vega 13	Calle San Marcos 15	L - V	01/12/2016	08:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	116	2	Marcos Martín	Hacienda de Pavones 6	Glorieta de Bilbao 12	L - V	10/11/2016	07:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	118	2	Marcos Martín	Avenida de Los Poblados 23	Ronda de Atocha 10	L - V	01/12/2016	08:00	2	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	111	3	Francisco Jiménez	C/ Fuente de la Hoz 20	Avenida de la Albufera 120	L - V	10/11/2016	10:00	2	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	119	3	Francisco Jiménez	Calle Preciados 13	Ronda de Andalucía 2	L - V	15/11/2016	07:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	121	3	Francisco Jiménez	Paseo de la Castellana 20	Calle Miguel Hernández 26	L - V	28/11/2016	08:00	3	1 euro
<input type="checkbox"/> Editar <input type="checkbox"/> Copiar <input type="checkbox"/> Borrar	128	4	Luis Garcia	Avenida de los Reyes Católicos (28040-Madrid)	Calle Fuencarral 50 (28004-Madrid)	L - V	15/01/2017	08:00	2	0,50 euros

Tras pulsar en “Ver Pasajeros” comprobamos que Rafael Gutiérrez es el único pasajero de la ruta.

Ver Pasajeros		
Nombre	Email	Opción
Rafael Gutiérrez	rafine@gmail.com	Eliminar pasajero

Todos los derechos reservados, JobCar 2016 ©

+ Opciones

					id	id_fk_usuario	id_fk_ruta
<input type="checkbox"/>	Editar	Copiar	Borrar	142		1	111
<input type="checkbox"/>	Editar	Copiar	Borrar	143		3	118
<input type="checkbox"/>	Editar	Copiar	Borrar	145		1	128

⬆ Marcar todos / Desmarcar todos Para los elementos que están

Al pulsar en “Eliminar Pasajero”, desaparece de la pantalla dicho usuario, se elimina la fila correspondiente de la tabla “Une”, y en la tabla rutas se actualiza el campo vacantes de dicha ruta, aumentando en una unidad. El pasajero recibirá una notificación por correo electrónico, informándole que le han dado de baja de la ruta.

Inicio	Crear Rutas	Ver Rutas	Yo, Conductor	Yo, Pasajero	Salir	LUIS1987
------------------------	-----------------------------	---------------------------	-------------------------------	------------------------------	-----------------------	----------

Ver Pasajeros		
Nombre	Email	Opción
Todos los derechos reservados, JobCar 2016 ©		

8. ASPECTOS A MEJORAR

- Adaptar JobCar para dispositivos móviles.
- Integración de JobCar en redes sociales.
- Posibilidad de editar la ruta creada por el conductor.
- Incluir la opción de Buscar, dentro de la sección Ver Rutas, para poder filtrar las búsquedas por determinados campos.
- Añadir un panel de administrador, para realizar la gestión de los usuarios.
- Incluir un sistema de votos, para que los pasajeros puedan mostrar su agrado o su descontento con el conductor de la ruta.
- Incluir un mapa con el itinerario seguido en la ruta.

9. CONCLUSIONES

Una vez finalizado mi trabajo de Fin de Grado, y redactada la memoria, estoy cerca de cerrar mi etapa como estudiante de la ETSISI.

Este trabajo final, me ha requerido un esfuerzo adicional, ya que actualmente trabajo a jornada completa, por lo que he tenido que desarrollarlo en las pocas horas que tenía disponibles.

He podido familiarizarme con Angular JS, un Framework de Javascript que desconocía (se necesitan bastantes años de trabajo para dominarlo), pero es gratificante poder ver hecha realidad, la idea que concebí meses atrás.

Actualmente soy tester de software, no trabajo en el entorno web, pero este aprendizaje me servirá para enfrentarme con otros desafíos que me esperarán seguro en el mundo laboral.

10. BIBLIOGRAFÍA

API de Angular JS (enlace activo a 22-10-2016):

- <https://docs.angularjs.org/api>

Tutorial de Angular JS (enlace activo a 22-10-2016):

- <http://www.w3schools.com/angular/default.asp>

Web oficial de Bootstrap (enlace activo a 22-10-2016):

- <http://getbootstrap.com/>

Manual de Bootstrap (enlace activo a 22-10-2016):

- https://librosweb.es/libro/bootstrap_3/

Manual de PHP en español (enlace activo a 22-10-2016):

- <https://secure.php.net/manual/es/>

Web de documentación de Medoo (enlace activo a 22-10-2016):

- <http://medoo.in/doc>

Web de MYSQL (enlace activo a 22-10-2016):

- <http://www.mysql.com/>

Web para descargar el editor de texto Sublime Text (enlace activo a 22-10-2016):

- <https://www.sublimetext.com/>

Web de información general (enlace activo a 22-10-2016):

- <https://es.wikipedia.org/>

Curso de Angular JS de Jesús Conde (enlace activo a 22-10-2016):

- <https://www.youtube.com/playlist?list=PLEtcGQaT56cgHfdvGguisToK90z321pR!>

Curso de Angular JS (enlace activo a 22-10-2016):

- <http://www.arielmax.com.ar/category/angular-js/page/4/>