



Escuela
Politécnica
Superior

Aplicación web para buscar aparcamiento y compartir coche



Grado en Ingeniería Informática

Trabajo Fin de Grado

Autor:

Ouadi Chamit

Tutor/es:

Estela Saquete Boro

Septiembre 2017



Universitat d'Alacant
Universidad de Alicante

Tabla de contenido

INTRODUCCION	4
RESUMEN	4
MOTIVACION Y ESTUDIO DE MERCADO	5
➤ BlaBlaCar	5
➤ Uber	5
➤ Amovens	5
➤ Carpooling	6
➤ Wazypark	6
➤ Parkopedia	6
➤ Aparca&Go	6
HERRAMIENTAS, TECNOLOGIAS Y LENGUAJES DE PROGRAMACION EMPLEADOS	7
phpMyAdmin	7
Github	7
Balsamiq	8
Codigniter	8
Ventajas de utilizar un framework como CodeIgniter	8
PhpStorm	9
Draw.io	9
Selenium IDE	9
REQUERITOS FUNCIONALES Y NO FUNCIONALES	10
Requisitos funcionales	10
Requisitos no funcionales Los requisitos no funcionales serán:	10
DISEÑO	11
Mockups	11
Wireframe	15
DISEÑO FINAL	16
ARQUITECTURA	36
IMPLEMENTACION	38
- Front end	38
Html/css	38
Y por ultimo Incluir CSS en los elementos HTML	41
Javascript/Jquery	43
Peticiones Ajax/json	45

- Back end	47
Base de datos	52
Esquema conceptual	52
Modelo Relacional.....	53
Diccionario de Datos	55
SEGURIDAD	62
Balancear riesgo y usabilidad	62
Rastrear el paso de los datos	62
XSS Prevention	62
Ataques de inyección SQL	62
Contraseña del usuario	63
APIS UTILIZADAS.....	65
Api Google maps	65
SIGUA	68
PRUEBAS.....	71
CONCLUSIONES, TRABAJOS FUTUROS Y POSIBLES MEJORAS	81
REFERENCIAS.....	81

INTRODUCCION

RESUMEN

El proyecto consiste en la creación de una aplicación web, que ayuda a los usuarios a publicar anuncios para compartir coche o buscar aparcamiento en la universidad de Alicante, con el fin de facilitar encontrar un aparcamiento libre, o poder compartir coche para ir y venir de la universidad.

El proyecto nace con la idea de crear una aplicación web dedicada a todos los usuarios de la universidad de Alicante, en concreto a aquellos usuarios que no tienen medios para venir a la universidad o los que no tienen suficiente gasto de los viajes y ayudar a los propietarios de los coches de poder ahorrar gastos. También sirve para buscar aparcamiento de forma cómoda, ya que la universidad consta de varios parkings, cada uno se identifica por una referencia y está ubicado en una zona, por lo que resulta difícil, encontrar un hueco en una zona predeterminada, según el interés de los usuarios, y como resultado, perder tiempo, gastos de energía etc.

MOTIVACION Y ESTUDIO DE MERCADO

Siendo alumno de la universidad, siento esa necesidad, y la de los demás usuarios, por lo que me motiva desarrollar esa aplicación y poder resolver uno de los problemas que afrontan los usuarios de la universidad que se encuentran a día hoy, y poder proveer una herramienta, para poner fin a esa necesidad o por lo menos paliar el problema.

Al analizar la competencia nos hemos dado cuenta de que no existe una aplicación en concreto con el mismo objetivo que gestiona tanto el aparcamiento como compartir coche a la vez específicamente para el sector universitario. No obstante, hay algunas aplicaciones encontradas que coinciden en algún apartado/sección, y que nos ha servido para mejorar nuestra idea.

Las aplicaciones buscadas han sido, por ejemplo:

➤ BlaBlaCar

Es una red social de viajes de largas y cortas distancias en coche compartido. Permite a los usuarios ponerse en contacto cuando quieren realizar un trayecto común y coinciden para hacerlo el mismo día, así como permitir a los conductores ahorrar el coste del litro de gasolina por cada pasajero.

Los usuarios comparten los gastos del viaje sin que el conductor tenga beneficios, para eso la red social recomienda a los usuarios por cada viaje la aportación de 0.06 euros por cada kilómetro cada uno, y limita la aportación máxima que pueden solicitar los conductores de tal manera que no se superen estos gastos.

➤ Uber

Uber es una aplicación para compartir viajes rápidos y fiables en cuestión de minutos, de día o de noche. No hace falta aparcar ni esperar taxis o autobuses.

Mediante la cual, comunicas tu ubicación, el destino, el tipo de vehículo y el tiempo de espera estimado. Tras el viaje, lo puedes pagar con tarjeta, efectivo o con tu teléfono móvil y el 20% se lo queda la propia empresa a modo de comisión

➤ Amovens

es similar a Uber en el concepto urbano de la aplicación, pero el objetivo de esta herramienta es encontrar a gente que realice el mismo trayecto que tú a diario y poder crear vínculos de transporte entre las personas, para ir al trabajo o a la universidad, sin que la empresa se lleve ningún tipo de comisión ni facilite modos de pago.

➤ **Carpooling**

es una aplicación similar a BlaBlaCar. Sirve exactamente para lo mismo, consiste en compartir nuestros viajes en auto con otras personas de forma cotidiana.

El carpooling es una práctica popular en Estados Unidos y Europa, donde se realiza de manera organizada para lograr aumentar el número de viajes compartidos y que estos sean concretados con otras personas además de nuestros vecinos y amigos.

Sin embargo ofrece elementos como la elección de viajes solo entre mujeres o fijar un radio de búsqueda a tu alrededor para empezar el viaje.

➤ **Wazypark**

Se trata de una aplicación de inteligencia colectiva en la que mediante geolocalización del móvil en un mapa, podemos saber dónde hay una plaza libre en las calles de nuestra ciudad para aparcar. Wazypark mejora a medida que aumenta el número de usuarios, que deben registrarse ellos mismos tanto como a su vehículo. Cada usuario avisa desde su móvil cuando deja un hueco libre en un aparcamiento, de modo que otro usuario cercano nos pueda detectar y aparcar en la plaza. Disponible para Android e iOS.

➤ **Parkopedia**

Tiene el mismo espíritu que Wazypark pero con elementos de Foursquare, en la que los usuarios dejan información de zonas de aparcamientos libres en plena calle o de aparcamientos públicos o privados, con los precios por hora o día de cada parking. Su interés reside en que hay información de numerosas ciudades. Tanto en Android como en iOS.

➤ **Aparca&Go**

Disponible solo para Android como aplicación, pero se puede usar desde la web. Se ha especializado en la gestión de aparcamientos mediante acuerdos por los que el usuario puede ganar un descuento en su ticket. La aplicación gestiona los pagos -hay que dejar los datos de una tarjeta-, por lo que las molestias son mínimas; además se puede reservar la plaza por adelantado. Un valor fuerte de esta aplicación es la creación de aparcamientos propios y alternativos a los de aeropuertos y estaciones de tren, pero con precios reducidos. De modo que ellos gestionan el transporte hasta el aeropuerto desde sus aparcamientos y de vuelta.

HERRAMIENTAS, TECNOLOGIAS Y LENGUAJES DE PROGRAMACION EMPLEADOS

Para llevar a cabo el proyecto he utilizado diversas tecnologías y herramientas seleccionadas para cumplir con ciertos propósitos concretos.

Para la realización de la aplicación, he utilizado el lenguaje de programación PHP en el lado Backend y el framework bootstrap, HTML, CSS, JAVASCRIPT, JQUERY en lado de FrontEnd.

Por último, para realizar la base de datos he utilizado SQL. La herramienta utilizada para manejar la base de datos es phpMyAdmin.

[phpMyAdmin](#)

phpMyAdmin es la herramienta que he utilizado para el manejo de la base de datos MySQL está alojado en el servidor junto con la página web. La versión de MySQL es la 10.1.25-MariaDB - mariadb.org binary distribution y la versión de phpMyAdmin utilizada es la 4.7.0 y está alojada sobre un servidor Apache/2.4.26 (Win32) OpenSSL/1.0.2I PHP/7.1.7.



[Github](#)

GitHub es una plataforma de **desarrollo colaborativo de software** para alojar proyectos utilizando el sistema de control de versiones Git.



Sirve para alojar un repositorio de código y brinda herramientas muy útiles para el trabajo en equipo, dentro de un proyecto.

Además de todo eso, se ofrecen varias herramientas útiles para el trabajo en equipo

- Una wiki para el mantenimiento de las distintas versiones de las páginas.
- Un sistema de seguimiento de problemas que permiten a los miembros de tu equipo detallar un problema con tu software o una sugerencia que deseen hacer.
- Una herramienta de revisión de código, donde se pueden añadir anotaciones en cualquier punto de un fichero y debatir sobre determinados cambios realizados en un commit específico.
- Un visor de ramas donde se pueden comparar los progresos realizados en las distintas ramas de nuestro repositorio

Balsamiq

He utilizado Balsamiq Mockups (versión escritorio). Balsamiq es la herramienta que he utilizado para realizar los mockups¹ de las pantallas que va a tener mi aplicación. Con el objetivo claro de conseguir un enfoque sobre el diseño básico en etapas más tempranas, es decir, antes de comenzar a diseñar las pantallas ya sé cuál es el diseño que van a tener.



Codigniter

CodeIgniter es un entorno de desarrollo abierto que utiliza el **MVC** y permite crear webs dinámicas con PHP . Su principal objetivo es ayudar a que los desarrolladores, puedan realizar proyectos mucho más rápido que creando toda la estructura desde cero, proveyendo un rico juego de librerías para tareas comúnmente necesarias, así como una interface simple y estructura lógica para acceder a esas librerías.



Ventajas de utilizar un framework como CodeIgniter

- Las páginas se procesan más rápido, el núcleo de CodeIgniter es bastante ligero.
- Es sencillo de instalar, basta con subir los archivos al ftp y tocar un archivo de configuración para definir el acceso a la bd.
- Reutilización de código, desarrollo ágil.
- Existe abundante documentación en la red.
- Facilidad de edición del código ya creado.
- Facilidad para crear nuevos módulos, páginas o funcionalidades.
- Acceso a librerías públicas y clases. Entre otras, hay librerías para el login, paginador, calendarios, fechas,....
- Estandarización del código. Fundamental cuando hay que tocar código hecho por otra persona o cuando trabaja más de una persona en un mismo proyecto.
- URLs amigables con SEO. Hoy en día creo que nadie duda de la importancia del posicionamiento web.
- Separación de la lógica y arquitectura de la web, el MVC.

- CodeIgniter es bastante menos rígido que otros frameworks. Define una manera de trabajar, pero podemos seguirlo o no (esto puede convertirse en un inconveniente también)
- Cualquier servidor que soporte PHP+MySQL sirve para CodeIgniter.
- CodeIgniter se encuentra bajo una licencia open source, es código libre.
- CodeIgniter usa una versión modificada del Patrón de Base de Datos Active Record. Este patrón permite obtener, insertar y actualizar información en tu base de datos con mínima codificación. Permite queries más seguras, ya que los valores son escapadas automáticamente por el sistema.

PhpStorm

es un IDE de programación desarrollado por JetBrains. Es uno de los entornos de programación más completos de la actualidad, permite editar código de varios lenguajes de programación aunque su nombre empieza por php.



algunas de las características principales.

- Proporciona un fácil autocompletado de código.
- Soporta el trabajo con PHP 5.5
- Sintaxis abreviada.
- Permite la gestión de proyectos fácilmente.

Draw.io

es una aplicación web interesante que es de gran ayuda para la realización de diagramas tales como: UML, diagramas de flujo, diagramas BPMN, creación de mockups, sin necesidad de instalar nada en la PC.



Selenium IDE

SeleniumHQ es un conjunto de herramientas de pruebas open-source para automatizar pruebas funcionales sobre aplicaciones Web



REQUISITOS FUNCIONALES Y NO FUNCIONALES

Requisitos funcionales

Los principales requisitos funcionales de la aplicación son:

- Cada usuario registrado es dado de alta, también existe el caso de dar de alta al coche si ese usuario es dueño de un coche en la cual comparte los viajes a/y aparca en la universidad.
- Cada usuario puede dar de alta a una notificación de compartir su coche cuando tiene plazas libres.
- Cada usuario puede buscar aparcamiento, poniendo la referencia del aparcamiento y ver las zonas libre.
- Cada usuario cuando sale del aparcamiento, debe actualizar la zona y ponerla como libre.
- Cuando usuario que se aparca, debe poner como ocupada la zona.
- Cada usuario puede comentar un coche cuando la comparte, así se lo permite a los demás usuarios conocer a las situaciones tanto del dueño como el coche .
- Cuando el usuario pretende compartir ruta, realiza una reserva a la espera de la aceptación del propietario del anuncio.
-

Requisitos no funcionales Los requisitos no funcionales serán:

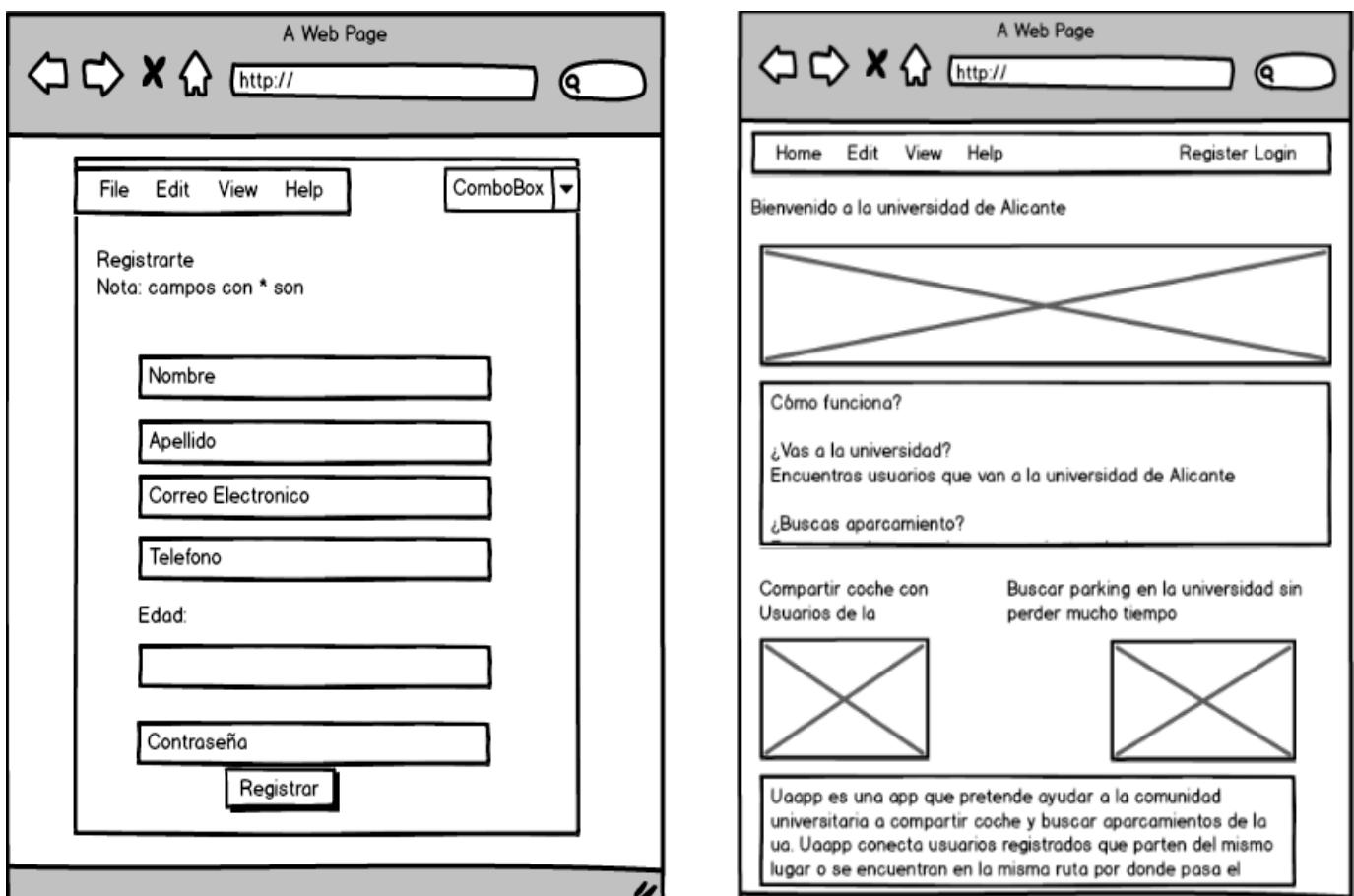
- Fiabilidad. La aplicación debe ser fiable en los datos que guarde. Si en algún momento no se puede acceder a internet, la aplicación deberá proporcionar otra forma alternativa de acceder a esos datos (guardados localmente).
- Usabilidad. La usabilidad en la aplicación será esencial ya que la mayoría de los usuarios que utilizarán la aplicación no estarán relacionados con el entorno informático. La tasa de errores cometidos por el usuario deberá ser menor. El sistema debe proporcionar mensajes de error que sean informativos y orientados a usuario final. El sistema debe poseer interfaces gráficas bien formadas.
- Eficiencia. El sistema debe ser capaz de procesar N transacciones por segundo. Toda funcionalidad del sistema y transacción de negocio debe responder al usuario en menor tiempo posible. El sistema debe ser capaz de operar adecuadamente con hasta N usuarios con sesiones concurrentes. Los datos modificados en la base de datos deben ser actualizados para todos los usuarios que acceden en menor tiempo.
- Dependibilidad. El sistema debe tener una disponibilidad del 99,99% de las veces en que un usuario intente accederlo. El tiempo para iniciar o reiniciar el sistema no podrá ser mayor. La tasa de tiempos de falla del sistema no podrá ser mayor. El promedio de duración de fallas no podrá ser mayor. La probabilidad de falla del Sistema no podrá ser mayor.

DISEÑO

Mockups

Antes de comenzar a implementar el proyecto, he creado los mockups de la aplicación. Consiguiendo así definir el diseño básico que tendrán las pantallas en fases tempranas del desarrollo .

Por ejemplo vamos a ver el mockup de la pantallas de la aplicación.



Pantalla Login/Actualizar datos usuario

The image displays two wireframe diagrams of web pages, likely representing a login screen and a user profile update screen.

Left Screen (Login):

- Header: "A Web Page" with browser controls (back, forward, stop, home, search).
- Address Bar: "http://".
- Menu Bar: "File Edit View Help" and a "ComboBox" dropdown.
- Content Area:
 - "Identificate"
 - "Correo Electronico" input field.
 - "Contraseña" input field.
 - "¿Has olvidado" link.
 - "Login" button.

Right Screen (User Profile Update):

- Header: "A Web Page" with browser controls (back, forward, stop, home, search).
- Address Bar: "http://".
- Top Navigation: "Home" (highlighted), "Editar perfil", "Viajes publicados", "Buscar", "Publicar", and a "ComboBox" dropdown.
- Content Area:
 - "Información personal" section with note: "Nota: campos con * son obligatorios".
 - Left sidebar:
 - "Información del perfil" (with "Foto" and "Coche" options)
 - "Opiniones" (with "Opiniones recibidas" and "Opiniones dejadas" options)
 - Main form:
 - "Nombre" input field.
 - "Apellidos" input field.
 - "Correo electronico" input field.
 - "Telefono" input field.
 - "Edad" input field.
 - "Contraseña" input field.
 - "Actualizar" button.
 - Footer:
 - "Posted by: Hege Refsnes"
 - "Contact information: someone@example.com"

Pantalla Coche/Foto usuario

A Web Page
http://

Home Editor perfil Viajes publicados Buscar Publicar ComboBox

Información del perfil
Información personal
Nota: campos con * son obligatorios

Matrícula
Marca
Modelo
Color
Categoría
Acentos

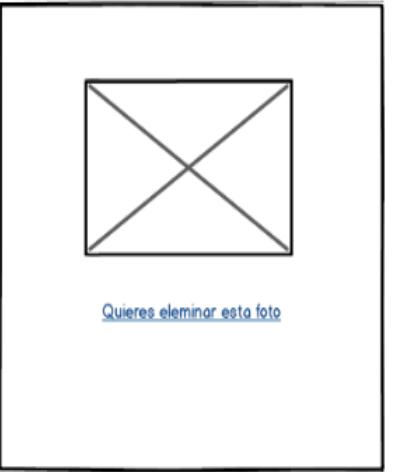
Insertar

Posted by: Hege Refsnes
Contact information: someone@example.com

A Web Page
http://

Home Editor perfil Viajes publicados Buscar Publicar ComboBox

Información del perfil
Información personal
Nota: campos con * son obligatorios



Quieres eliminar esta foto

Posted by: Hege Refsnes
Contact information: someone@example.com



Pantalla Publicar anuncio/Buscar coche para compartir

A Web Page
http://

Home Editar perfil Viajes publicados Buscar **Publicar** ComboBox ▾

Publicar Ruta
Nota: campos con * son obligatorios

coche:

Salida:

Precio:

Plazas disponible

Detalles
Ejemplo: Voy a La universidad de Alicante todos los días, salgo de Torrevieja, tengo clases solo por las mañanas

Button

Posted by: Hege Refsnes
Contact information: someone@example.com.

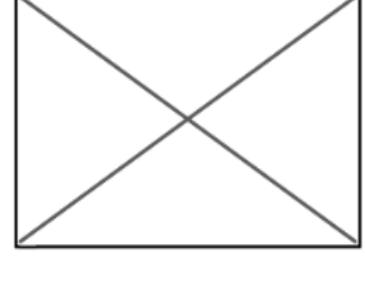
A Web Page
http://

Home Editar perfil Viajes publicados Buscar **Publicar** ComboBox ▾

Buscar Ruta
Nota: campos con * son obligatorios

Salida:

Buscar



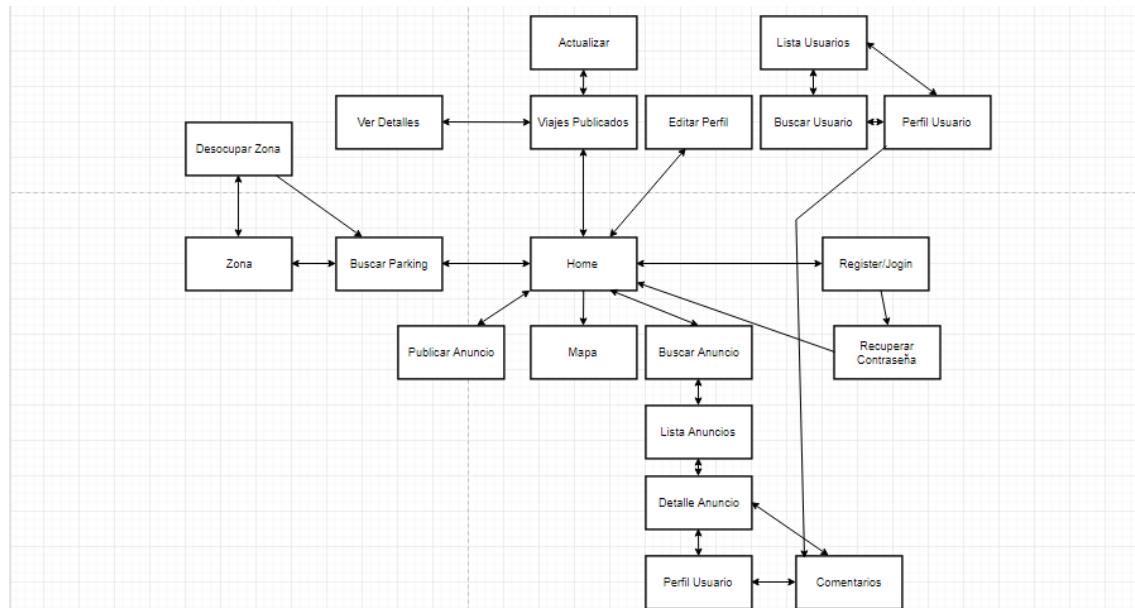
Posted by: Hege Refsnes
Contact information: someone@example.com.



§ Wireframe

Desde home podemos acceder a las páginas funciones principales del menú de la cabecera. Además también podrás acceder a tu cuenta mediante login y registro. Una vez se registra o se inicia sesión, se podrá acceder a todas las funcionalidades de la app, por ejemplo editar perfil, donde el usuario puede actualizar sus datos, viajes publicados donde el usuario tiene todo el historial de sus publicaciones, con los cuales puede ver información al respecto o eliminarlos.

Tambien puede buscar anuncio, buscar usuario, ver perfil de usuario, ver detalle de anuncio, etc.



DISEÑO FINAL

Pantalla principal de la aplicación

En la pantalla principal, se encuentra la formación que provee la aplicación, Como las funcionalidades, el registro y el login, Además cualquier usuario tiene la opción de ver los parkings de la ua, sin que este, este dado de alta.

The screenshot shows the main page of the application. At the top, there is a navigation bar with links for Home, Quien somos, Contact, Register, and Login. The main content area features a large image of a modern university building reflected in water. Below the image, there is a section titled "Cómo funciona?" (How it works) with six items arranged in a 2x3 grid:

¿Vas a la universidad?	Confirmas la reserva	¡A viajar!
Encuentras usuarios que van a la universidad de Alicante.	Los usuarios reservan las plazas libres online.	Compartes coche y dejas una opinión a tus compañeros.
¿Buscas aparcamiento?	Confirmas el aparcamiento	¡Aparcar!
Encuentras hueco en los aparcamientos de la ua.	Los usuarios reservan las plazas libres online.	una vez sales de tu aparcamiento, ha de indicar que el espacio está libre.

At the bottom of the main content area, there are three buttons: "Compartir coche con Usuarios", "Ver parkings de la universidad", and "Buscar parking en la". The taskbar at the bottom of the screen shows various application icons.

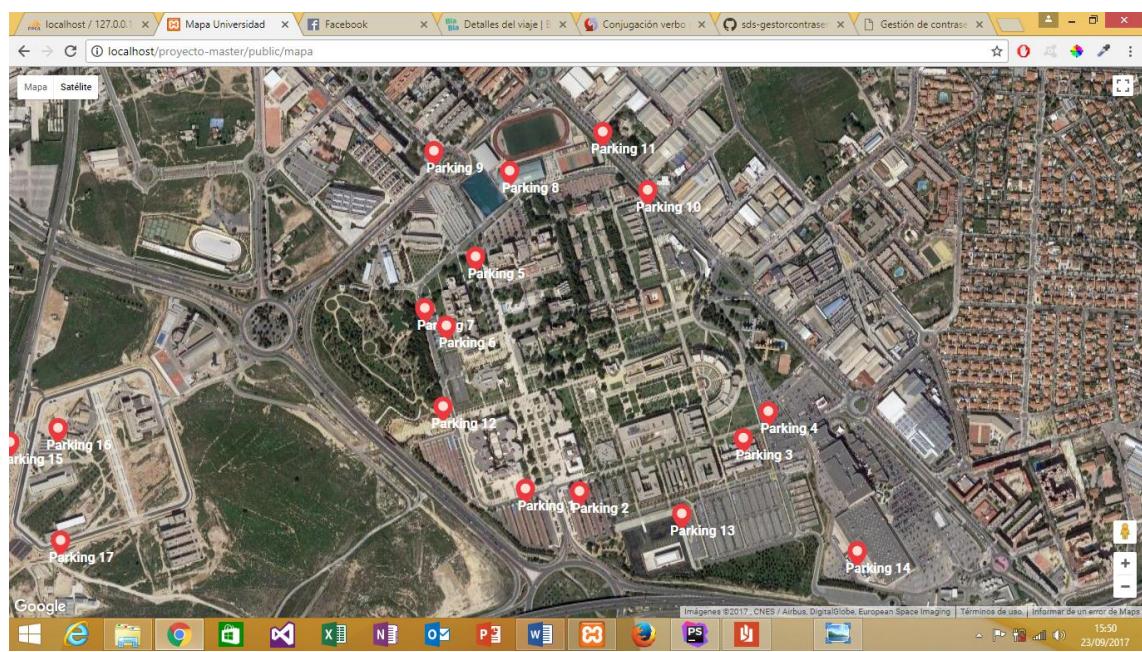
The screenshot shows the main page of the application. The top navigation bar and the "Cómo funciona?" section are identical to the previous screenshot. The main content area has been simplified, showing only three functional descriptions:

Compartir coche con Usuarios de la universidad	Ver parkings de la universidad	Buscar parking en la universidad sin perder mucho tiempo
--	--------------------------------	--

Below each description is a small image and a brief explanation:

- Compartir coche con Usuarios de la universidad:** An image of four people in a car, with text explaining that users can find someone to share a ride.
- Ver parkings de la universidad:** An aerial map of the university campus, with text explaining that users can see available parking spots.
- Buscar parking en la universidad sin perder mucho tiempo:** An image of a full parking lot, with text explaining that users can quickly find available spots without wasting time.

At the bottom of the main content area, there is a note about the app's purpose: "Uaapp es una app que pretende ayudar a la comunidad universitaria a compartir coche y buscar aparcamientos de la ua. Uaapp conecta usuarios registrados que parten del mismo lugar o se encuentran en la misma ruta por donde pasa el propietario del coche, así como encontrar un aparcamiento, sin perder mucho tiempo." The taskbar at the bottom of the screen shows various application icons.



Pantalla de Registro de Usuario

Esta pantalla permite a los usuarios crear una nueva cuenta. Como se puede observar,

Consta de varios campos todos son obligatorios, indicados con el asterisco rojo, controlados con la propiedad de html required, la cual impide hacer un submit si un campo esta vacío.

Además, se comprueba el campo correo con Ajax para que no sea duplicado en la base de datos, ya que es un dato único. También el campo teléfono y contraseña son controlados con Jquery para que sean datos validos.

Una vez se comprueba todo que es correcto, cuando el controller registro recibe los datos de parte del cliente, se mantiene todos los datos en plano, menos la contraseña que se almacena de forma cifrada, para realizar el cifrado, se Utiliza SHA-512 como hash sobre la contraseña, se crea una salt aleatoria. Esta salt servirá para “mezclarla” con el hash creado y realizar un encriptado con “crypt”, y por ultimo realizar el almacenamiento en la base de datos.

Nota: campos con * son obligatorios

The screenshot shows a user registration form with several fields and validation messages:

- Nombre:** ouadi (highlighted in yellow, required field)
- Apellido:** chamit (highlighted in yellow, required field)
- Correo:** usuario30@gmail.com (highlighted in yellow, required field)
Error message: *Ya hay un usuario con esta cuenta, introduzca otra cuenta*
- Teléfono:** 55 (highlighted in yellow, required field)
Error message: *El telefono debe empezar por 6 o 7*
- Contraseña:** (highlighted in yellow, required field)
Error message: *La contraseña debe contener mas de 7 caracteres*
- Edad:** 20 (highlighted in yellow, required field)
- Descripción (max 300 caracteres):** En pocas palabras... Vivo en torrevieja y trabajo en Valencia, me tengo que desplazar casi todos los días de un a sitio a otro.

A blue "Registrar" button is at the bottom of the form.

Cuando se realiza el registro de forma correcta, se le envía al usuario un correo electrónico, para confirmar el registro pulsando sobre el enlace enviado, en caso contrario, no puede acceder al sistema.

Email Test Recibidos x

 **chamit** 20 sept. (hace 3 días) ☆ ↶ ⌄

para mí ▼

haga click en el enlace abajo para activar tu cuenta, y poder utilizar la aplicación <http://localhost/proyecto-master/confirmRegistro/everis@gmail.com>

 Haz clic aquí para [Responder](#) o para [Reenviar](#)

12,95 GB (86%) ocupados de 15 GB [Administrar](#)

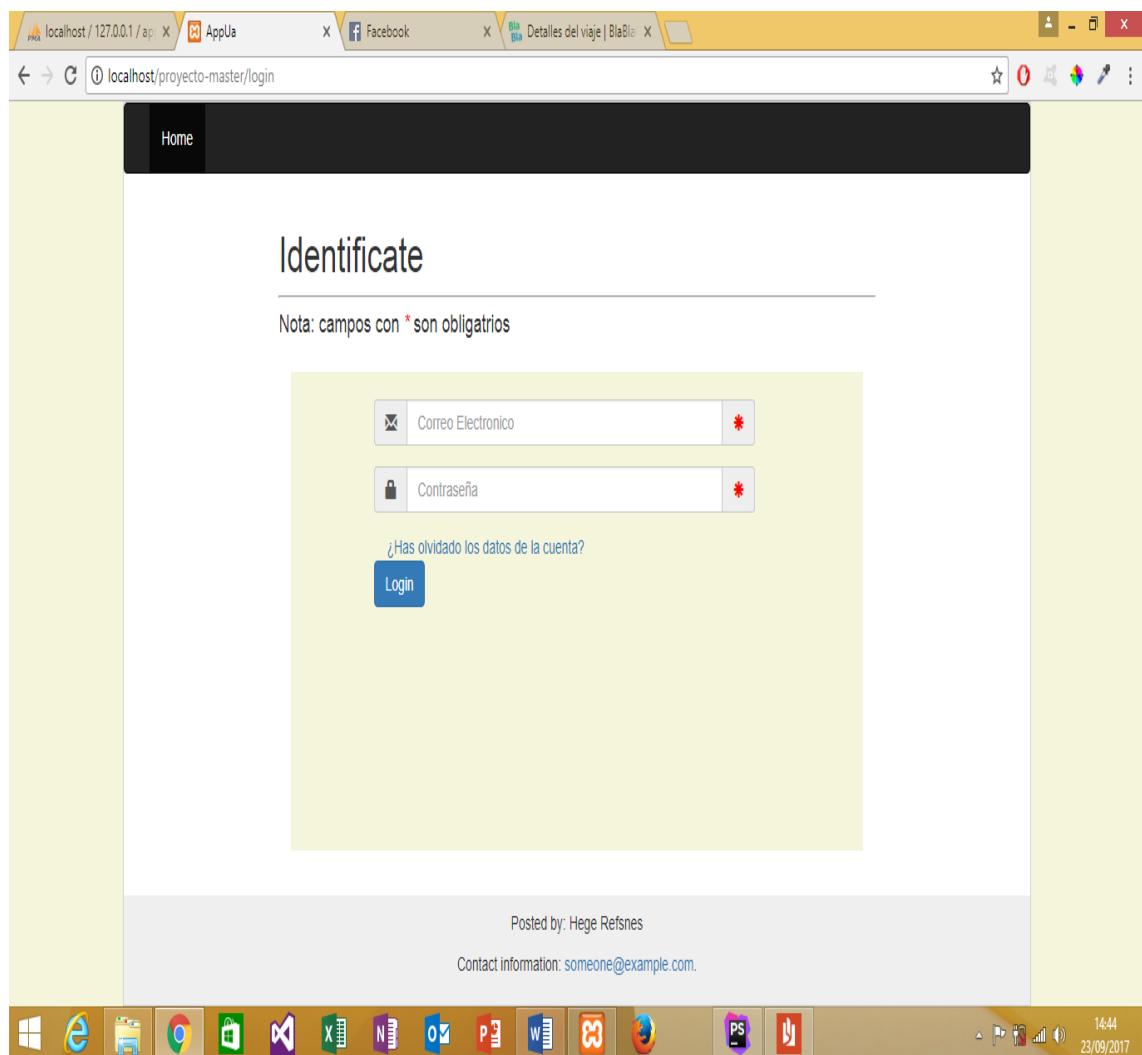
[Condiciones](#) - [Privacidad](#)

Última actividad de la cuenta: hace 4 días
[Información detallada](#)

Pantalla de Inicio de Sesión

La pantalla de inicio de sesión sirve para que los usuarios puedan acceder con sus credenciales a la aplicación.

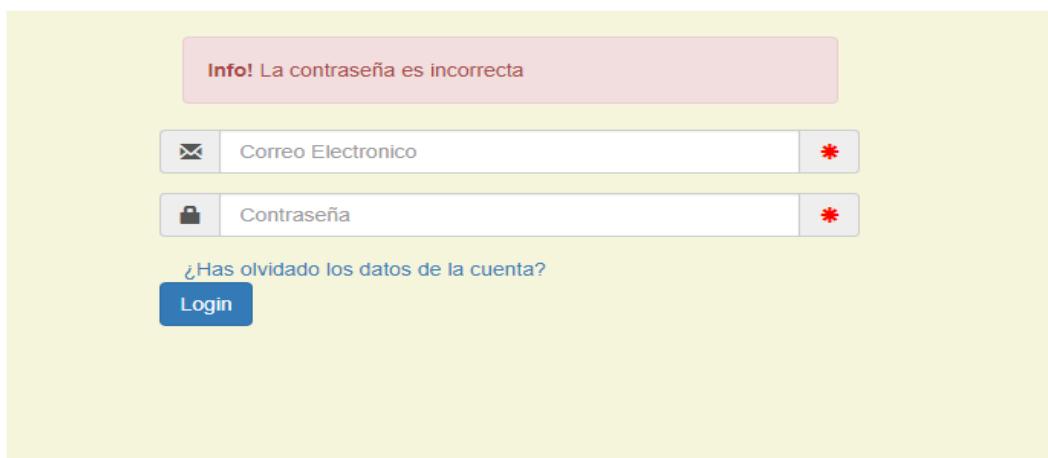
La pantalla tendrá varias campos, un botón para realizar la petición, un enlace en caso de perder la contraseña y dos campos para introducir texto en el cuál se deberán introducir las credenciales de acceso del usuario, esto es, nombre de usuario y contraseña.



Una vez esta diseñada la pantalla, se implementa una función para comprobar la coincidencia de los datos proveídos con los datos que están en la base de datos, esta función Utiliza SHA-512 como hash sobre la contraseña, se recupera la salt aleatoria de la base de datos a base del correo introducido, porque cada usuario tiene un campo salt, con la cual se hace la mezcla con el hash con la función crypt y se guarda la contraseña mezclada. Esta salt servirá para “mezclarla” con el hash relalizado sobre la contraseña y realizar un encriptado con “crypt” y por tanto comprobar la coincidencia de las contraseñas si son correctas.

Identificate

Nota: campos con * son obligatorios



The screenshot shows a login form with two input fields. The first field, labeled 'Correo Electronico', has a red asterisk (*) indicating it is required. The second field, labeled 'Contraseña', also has a red asterisk (*). Above the fields, a pink info box displays the message 'Info! La contraseña es incorrecta' (Info! The password is incorrect). Below the fields is a link '¿Has olvidado los datos de la cuenta?' (Forgot your account details?) and a blue 'Login' button.

Como se observa en la captura, si la contraseña es incorrecta, se indica al usuario con el mensaje con color morado. En caso que el correo no existe en la base de datos se le indica con el mensaje reflejado en la captura siguiente:

Identificate

Nota: campos con * son obligatorios



The screenshot shows a login form with two input fields. The first field, labeled 'Correo Electronico', has a red asterisk (*) indicating it is required. The second field, labeled 'Contraseña', also has a red asterisk (*). Above the fields, a pink info box displays the message 'Info! Los datos son incorrectos' (Info! The data is incorrect). Below the fields is a link '¿Has olvidado los datos de la cuenta?' (Forgot your account details?) and a blue 'Login' button.

Pantalla recuperación de contraseña

Si el usuario no se acuerda de su contraseña, bajo cualquier constancia, tiene la posibilidad de recuperar su cuenta, solo introduzca su correo, botón send, recibirá un correo con un enlace mediante el cual se lleva a otra vista donde introduzca su nueva contraseña.

Home

Introduzca tu correo

Nota: campos con * son obligatorios

usuario30@gmail.com *

Send

chamit para mí ▾ 16:02 (hace 0 minutos) ★ ↗ ▾

haga click en el enlace abajo para actualizar su contraseña.
<http://localhost/proyecto-master/private/recoverUpdatePassword/usuario30@gmail.com>

Haz clic aquí para [Responder](#) o para [Reenviar](#)

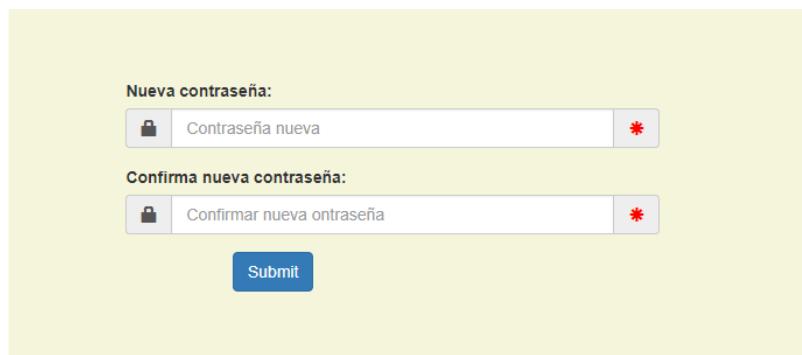
12,95 GB (86%) ocupados de 15 GB
[Administrar](#)

[Condiciones](#) - [Privacidad](#)

Última actividad de la cuenta: hace 4 días
[Información detallada](#)

Gestión de la clave

Nota: campos con * son obligatorios



Este formulario es para cambiar la contraseña. Los campos obligatorios están marcados con un asterisco (*).

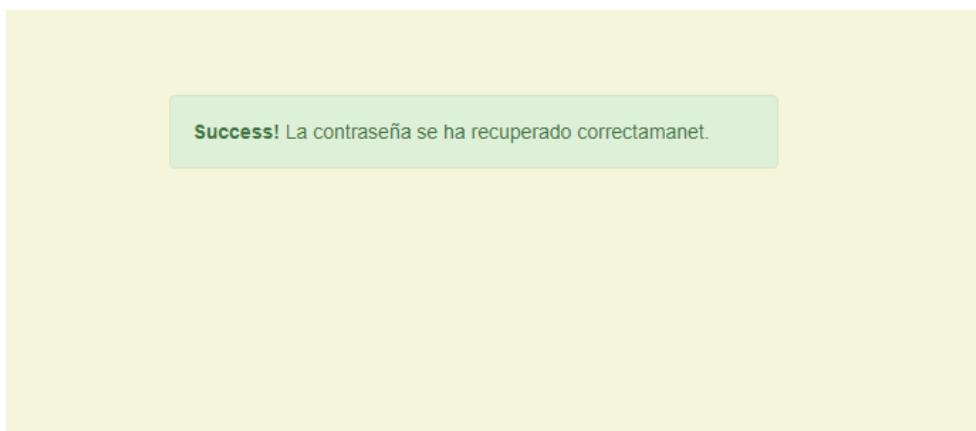
Nueva contraseña:
 Contraseña nueva *

Confirma nueva contraseña:
 Confirmar nueva contraseña *

Como resultado de la recuperación de la cuenta, se le notifica al usuario con el mensaje “La contraseña se ha recuperado correctamente”

Gestión de la clave

Nota: campos con * son obligatorios



Success! La contraseña se ha recuperado correctamente.

Pantalla Editar Perfil

Como se observa en la captura, cuando el usuario inicia sesión, aparecen los menús, dos horizontales y dos verticales, los cuales llevan a todas las funcionalidades de la aplicación, por ejemplo en la captura siguiente, el usuario puede editar la información personal, su foto, coche, ver comentarios hechos y recibidos y actualizar contraseña etc.

La captura muestra la interfaz de usuario de una aplicación web. En la parte superior, hay un menú horizontal con enlaces a "Home", "Editar perfil", "Viajes publicados", "Buscar anuncio", "Publicar anuncio", "Buscar parking" y "Bienvenido usuario30". El enlace "Bienvenido usuario30" despliega un menú suspenso con las siguientes opciones: "Editar perfil", "CSS", "Viajes publicados", "Buscar usuario" y "Cerrar sesion".

El contenido principal es un formulario titulado "Información personal". Se incluye una nota que dice: "Nota: campos con * son obligatorios". El formulario contiene los siguientes campos:

- Nombre: *
- Apellido: *
- Correo electrónico: *
- Teléfono: *
- Edad: *

En el centro del formulario hay un botón azul "Submit".

Esta captura muestra la misma interfaz de usuario que la anterior, pero con un mensaje de éxito en la parte superior del formulario: "Success! Los datos se han actualizado correctamente."

El resto del formulario y su estructura es idéntica al de la primera captura, incluyendo los campos obligatorios y el botón "Submit".

Editar foto

En esta pantalla el usuario, puede poner foto, borrarla y cambiarla por otra.

Screenshot of the 'Foto de perfil' (Profile Photo) section of a user profile page. The page includes a navigation bar with links like 'Home', 'Editar perfil', 'Viajes publicados', 'Buscar anuncio', 'Publicar anuncio', 'Buscar parking', and 'Bienvenido usuario30'. On the left, there's a sidebar with sections for 'Información de perfil' (Personal info), 'Opiniones' (Reviews), 'Cuenta' (Account), and 'Contraseña' (Password). The main area shows a placeholder for a profile photo with a message: 'Añade una foto para tu perfil. A los demás usuarios les tranquilizará saber con quién van a viajar, y a ti te será más fácil encontrar un viaje. Las fotos os ayudarán a reconoceros en el punto de encuentro.' Below the placeholder is a small blue button labeled 'Quieres eliminar esta foto' (Do you want to delete this photo?).

Cuando el usuario, no tiene foto o la borra, en esta vista puede incluir una.

Screenshot of the 'Foto de perfil' (Profile Photo) section of a user profile page. The layout is identical to the previous screenshot, including the navigation bar and sidebar. However, the main area now features a placeholder image of a Sony digital camera, indicating that no photo has been uploaded. Below the camera image is a file upload interface with a 'Upload:' label, a 'Seleccionar archivo' (Select file) button, and a message 'Ningún archivo seleccionado' (No file selected). A 'Submit' button is located at the bottom of the form.

Vista de opiniones realizados

Home Editar perfil Viajes publicados [Buscar anuncio](#) [Publicar anuncio](#) [Buscar parking](#) Bienvenido usuario30 ▾

Informacion de perfil
informacion personal
foto
coche

Opiniones
Opiniones recibidas
Opiniones hechas

Cuenta
Contraseña
Darse de baja

Comentario realizados

Opiniones

	usuario30 usuario30 kjkjbkbkjkb
	usuario30 usuario30 jbrkjbrekjg
	usuario30 usuario30 capullo

1234 [Siguiente](#) [>](#) [Last](#) >

Comentarios recibidos

Home Editar perfil Viajes publicados [Buscar anuncio](#) [Publicar anuncio](#) [Buscar parking](#) Bienvenido usuario30 ▾

Informacion de perfil
informacion personal
foto
coche

Opiniones
Opiniones recibidas
Opiniones hechas

Cuenta
Contraseña
Darse de baja

Comentario recibidos

Opiniones

	usuario30 usuario30 jbrkjbrekjg
---	---

Pantalla publicar anuncio

En esta vista el usuario, publica un anuncio, para buscar usuarios con el fin de compartir el coche, el usuario, pone el punto de salida, el precio, las plazas disponibles y un detalle del anuncio.

Si el usuario no dispone de un coche no puede publicar ningún anuncio, ya que cuando introduzca los datos y los envíe al servidor, se le notifica con un mensaje de que debe introducir datos del su coche

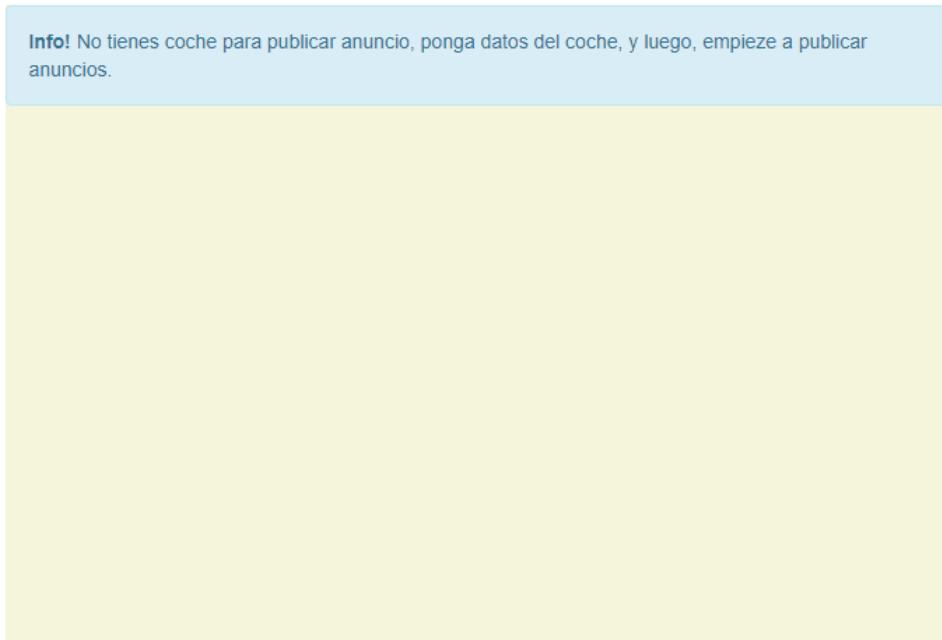
The screenshot shows a user interface for posting a route. At the top, there is a navigation bar with links: Home, Editar perfil, Viajes publicados, Buscar anuncio (Search ad), Publicar anuncio (Post ad), Buscar parking (Search parking), and Bienvenido usuario30 (Welcome user30). The main section is titled "Pblicar Ruta" (Post Route). Below the title, a note says "Nota: campos con * son obligatorios" (Note: fields with * are mandatory). The form consists of several input fields:

- Salida:** A text input field labeled "Text" with placeholder "Ejemplo: Elche". A red asterisk (*) is positioned to the right of the input field.
- Precio:** A text input field labeled "€" with placeholder "Ejemplo: 3". A red asterisk (*) is positioned to the right of the input field.
- Plazas disponibles:** A text input field labeled "N" with placeholder "Ejemplo: 3". A red asterisk (*) is positioned to the right of the input field.
- Detalles:** A text area containing placeholder text: "Ejemplo: Voy a La universidad de Alicante todos los dias, salgo de Torrevieja, tengo clases solo por las mañanas".

At the bottom of the form is a blue "Publicar" (Post) button.

Publicar Ruta

Nota: campos con * son obligatorios



Pantalla buscar parking

Cuando el usuario busca parking, la opción Buscar parking lo lleva a una vista con el mapa de la universidad, con los parkings indicados con los puntos rojos.

The screenshot shows a user interface for searching parking. At the top, there is a navigation bar with links: Home, Editar perfil, Viajes publicados, Buscar anuncio (with a magnifying glass icon), Publicar anuncio (with a car icon), Buscar parking (highlighted in blue), and Bienvenido usuario30 (with a dropdown arrow). Below the navigation bar, the title "Buscar Parking" is displayed. Underneath the title, the text "Elige parking" is followed by a dropdown menu labeled "Elige parking:" containing the option "parking 1". A blue button labeled "Buscar aparcamiento" is positioned below the dropdown. To the right of the dropdown is a satellite map of a university campus. The map shows various buildings, roads, and green spaces. Several red circular markers with white numbers (Z6, Z7, Z8, Z9, Z10, Z11) are placed on the map, indicating the locations of different parking zones. There are also two small buttons at the top left of the map labeled "Mapa" and "Satélite".

Pantalla Eligir zona.

Mediante la opción del select, cuando el usuario elige un parking mediante su numero, se le lleva otra vista con las zonas vacías de este parking.

Elige Zona para aparcar y actualizar para indicar como zona ocupada:

Z 461

Actualizar Zona como ocupada



Pantalla Desocupar zona.

Cuando el usuario indica que ha ocupado una zona dada, se le lleva a una vista para desocupar esta zona, y además, si no la desocupa, cuando vuelve a buscar zonas vacías en los parkings,

siempre se le lleva a esa vista para marcar como desocupada la zona, así puede volver a buscar huecos.

Info! Debes desocupar la zona ocupada y luego puedes volver a aparcar.

Zona ocupada

461

Desocupar zona ocupada

Mapa Satélite

A satellite map of a city, likely Madrid, showing a dense urban area with a grid of streets. A red marker is placed on the map at coordinates Z 461, indicating the location of the occupied parking zone. The map also shows various landmarks, roads, and green spaces.

Pantalla buscar anuncio

En esta pantalla el usuario, pone un origen, en el campo origen del formulario, como resultado, le salen todos los anuncios desde ese origen con plazas disponibles.

Screenshot of the 'Buscar Ruta' (Search Route) page. The top navigation bar includes 'Home', 'Editar perfil', 'Viajes publicados', 'Buscar anuncio' (with a magnifying glass icon), 'Publicar anuncio' (with a pen icon), 'Buscar parking', and 'Bienvenido usuario30'. Below the navigation is a search form with a 'Text' input field containing 'Ejemplo: Elche' and a red asterisk indicating it's required. A 'Buscar' button is next to the input. To the right is a map showing routes from 'Elche' to 'San Vicente del Raspeig'. The map highlights two main routes: one via AP-7/A-31 (52 min, 64.3 km) and another via N-332 (50 min, 71.4 km). The bottom of the map shows coastal towns like Santa Pola and Guardamar del Segura.

Como resultado, le sale una vista en modo paginación para ver todos los anuncios.

Screenshot of the '12 viajes disponibles de a la universidad de alicante' (12 available journeys to the University of Alicante) page. The top navigation bar is identical to the previous page. The main content displays four travel ads in a grid:

Usuario	Origen	Destino	Precio	Otros
aux aux 22 años	Elche	Universidad de Alicante	3 € por plaza	2 plazas disponibles
usuario2 2 años	Elche	Universidad de Alicante	4 € por plaza	2 plazas disponibles
usuario3 2 años	Elche	Universidad de Alicante	3 € por plaza	3 plazas disponibles
usuario4 3 años	Elche	Universidad de Alicante	3 € por plaza	3 plazas disponibles

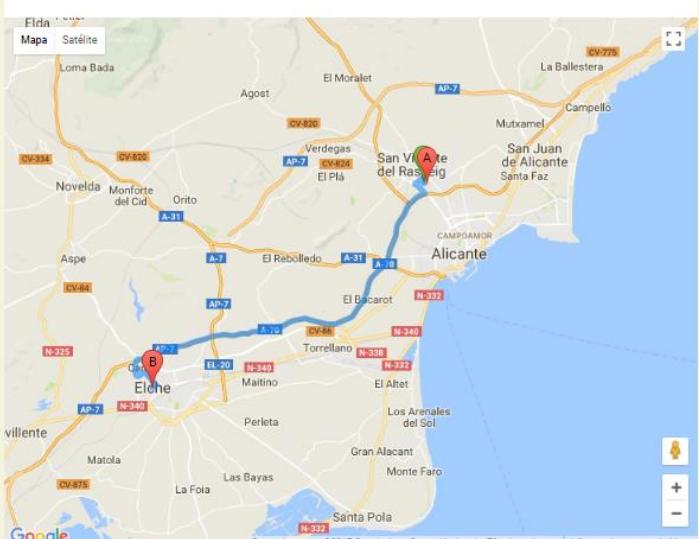
	usuario4 usuario4 3 años	Elche -> Universidad de Alicante	3 € por plaza 3 plazas disponibles
	usuario5 usuario5 2 años	Elche -> Universidad de Alicante	3 € por plaza 3 plazas disponibles

123Siguiente ->

Cuando el usuario selecciona un anuncio le sale todos los datos del anuncio en otra vista

Con una mapa indicando la ruta desde el punto de salida hasta la universidad

Elche --> Universidad de Alicante

Salida:	Elche	Precio	3 €
Fecha Publicacion	2017-07-09	Plazas	2 disponibles
Detalles: Los puntos de recogida y dejada son en Valencia plaza de toros, mestalla, rotonda de los anzuelos, en Alicante renfe, estación de autobuses, aeropuerto, telepizza en san vicente, en Elche estación de tren elche carrus y parque, estación de autobuses, en torrevieja estación de autobuses.			
		Conductor  aux aux 22 años	
Coche  A15 Categoría: TURISMO Color: BLANCO		Opiniones  usuario30usuario30 capullo usuario30usuario30 khjhvhvjhbh	
Ver todos los comentarios			
Actividades Ver Perfil			

Administracion

En la parte de administración, donde el usuario que tiene privilegios, puede gestionar la aplicación, dar de alta, modificar, borrar, actualizar toda la información contenida en la aplicación, en esta parte he utilizado una librería que provee el framework, la librería permite crear un formulario que contiene toda la información de una tabla en la base de datos, así como permitir hacer las operaciones crud, y buscar información por filtro. Para llevar a cabo el desarrollo, en el controlador se realiza los pasos siguientes:

```
$crud = new grocery_CRUD();
$crud->set_table('coche');
$crud->set_subject('coche');
$crud->set_theme("datatables");
```

En el ejemplo anterior la librería crea un objeto coche como un formulario que tiene toda la información de los coches dados de alta en el sistema, en este formulario, se pueden realizar todas las operaciones, incluso buscar por filtro.

Coches Gestión de coches								
Add coche								
Print								
Matricula	Modelo	Color	Acientos	Usuario	Marca	Categoría	ImageFoto	
11111	10	BLANCO	4	occ10@alu.ua.es	A15	TURISMO		View Edit
3214BKC	10	BLANCO	4	usuario10@gmail.com	A15	TURISMO	ferrari2.jpg	View Edit
3214BKK	10	BLANCO	2	occ10@alu.ua.es	A15	TURISMO		View Edit
3214LLL	10	BLANCO	2	usuario30@gmail.com	A15	TURISMO	ferrari22.jpg	View Edit
auxx	10	BLANCO	2	aux@aux.com	A15	TURISMO		View Edit
usuario1	10	BLANCO	2	usuario1@gamail.com	A15	TURISMO		View Edit
usuario10	10	BLANCO	2	usuario10@gmail.com	A15	TURISMO		View Edit
usuario11	10	BLANCO	2	usuario11@gmail.com	A15	TURISMO		View Edit
usuario12	10	BLANCO	2	usuario12@gmail.com	A15	TURISMO		View Edit
usuario2	10	BLANCO	2	usuario2@gmail.com	A15	TURISMO		View Edit
Search Matricu	Search Mod	Search Cl	Search Acient	Search Usuario	Search Ma	Search Catego	Search ImageFo	Clear

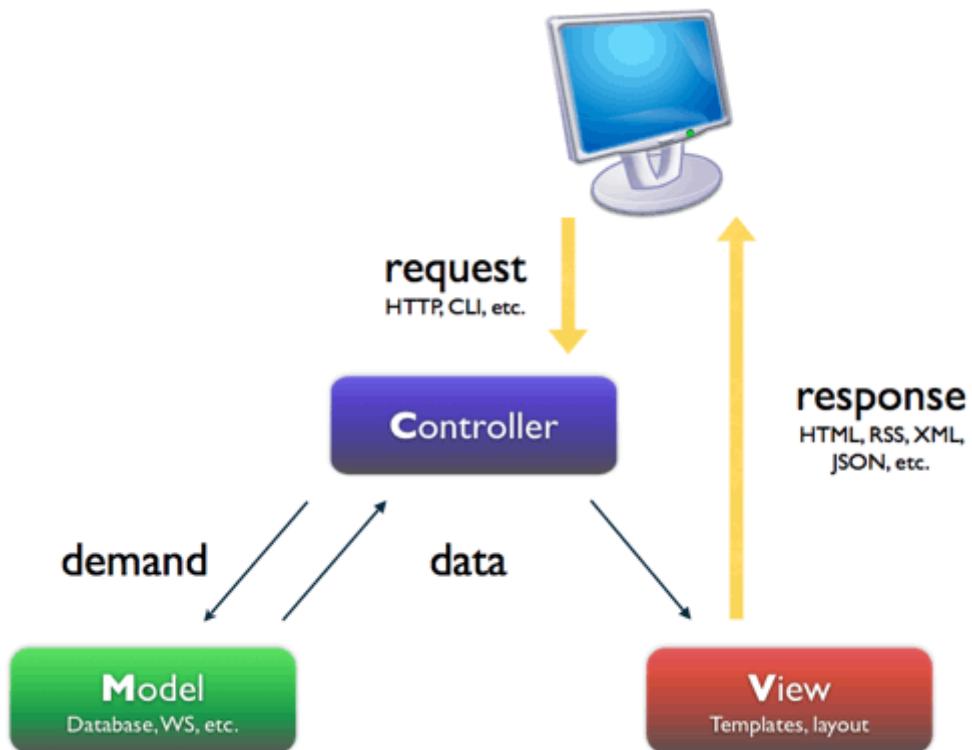
Screenshot of a web application interface titled "AppUA Admin". The main title is "Coches Gestión de coches". The left sidebar contains navigation links: Inicio, Usuarios, Coches, Ruta, Parking, Zona, Comentarios, and Historial Parking. The main content area shows a table of car data with columns: Matricula, Modelo, Color, Acientos, Usuario, Marca, Categoría, and ImageFoto. The table has 10 entries. The bottom of the screen shows a taskbar with various icons and a system tray indicating the date and time.

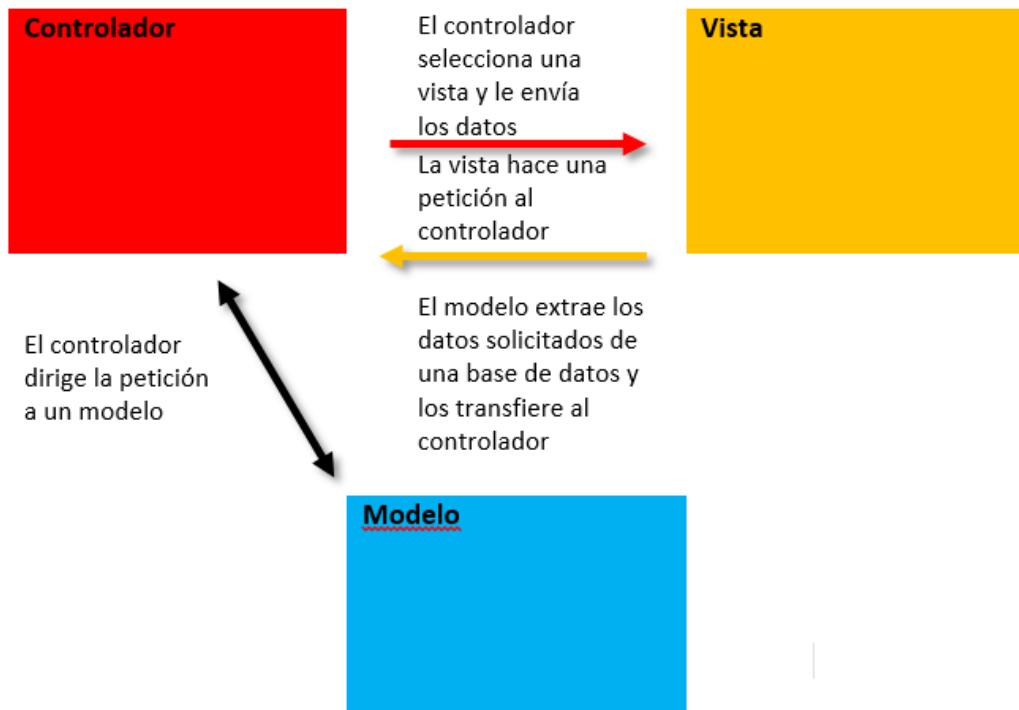
Matricula	Modelo	Color	Acientos	Usuario	Marca	Categoría	ImageFoto		
11111	10	BLANCO	4	occ10@alu.ua.es	A15	TURISMO			
3214BKC	10	BLANCO	4	usuario10@gmail.com	A15	TURISMO	ferrari2.jpg		
3214BKK	10	BLANCO	2	occ10@alu.ua.es	A15	TURISMO			
3214LLL	10	BLANCO	2	usuario30@gmail.com	A15	TURISMO	ferrari22.jpg		
auxx	10	BLANCO	2	aux@aux.com	A15	TURISMO			
usuario1	10	BLANCO	2	usuario1@gmail.com	A15	TURISMO			
usuario10	10	BLANCO	2	usuario10@gmail.com	A15	TURISMO			
usuario11	10	BLANCO	2	usuario11@gmail.com	A15	TURISMO			
usuario12	10	BLANCO	2	usuario12@gmail.com	A15	TURISMO			
usuario2	10	BLANCO	2	usuario2@gmail.com	A15	TURISMO			

ARQUETECTURA

El modelo utilizado en el desarrollo es Modelo vista controlador mediante el framework CodeIgniter, el cual hace uso del patrón MVC, un patrón de desarrollo que separa una aplicación en 3 capas:

- **Modelo:** es la capa de acceso a datos, en ella se realizan las conexiones y consultas a la base de datos, y los datos que esta obtiene son recibidos por la capa Controlador.
- **Controlador:** aquí encontramos la lógica del programa, esta capa manipula la información que recupera de la base de datos, y la envía a la Vista. También recibe las acciones del usuario capturadas por la Vista y responde a ellas.
- **Vista:** en esta parte se define la interfaz de usuario, especifica cómo se mostrará la información obtenida del Controlador y capture los eventos del usuario.





IMPLEMENTACION

- Front end

Html/css

En cuanto a la vista, en el Front end, he utilizado el lenguaje de marcado HTML, es el elemento de construcción más básico de una página web y se usa para crear y representar visualmente una página web. Determina el contenido de la página web, pero no su funcionalidad. Otras tecnologías distintas de HTML son usadas generalmente para describir la apariencia/presentación de una página web (CSS) o su funcionalidad (JavaScript/jquery).

HTML le da "valor añadido" a un texto estándar en español. Hiper Texto se refiere a enlaces que conectan una página Web con otra, ya sea dentro de una página web o entre diferentes sitios web. los vínculos son un aspecto fundamental de la Web. Al subir contenido a Internet y vincularlo a páginas de otras personas, te haces participante activo de esta Red Mundial.

HTML usa "markup" o marcado para anotar textos, imágenes, y otros contenidos que se muestran en el Navegador Web. El lenguaje de marcado HTML incluye "elementos" especiales tales como <head>, <title>, <body>, <header>, <article>, <section>, <p>, <div>, , , y muchos otros más.

Para dar estilo a las páginas webs, he utilizado CSS Hojas de Estilo en Cascada (Cascading Style Sheets) y Bootstrap.

CSS es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla, o cómo se va a imprimir, o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos.

CSS funciona a base de reglas, es decir, declaraciones sobre el estilo de uno o más elementos. Las hojas de estilo están compuestas por una o más de esas reglas aplicadas a un documento HTML o XML. La regla tiene dos partes: un selector y la declaración. A su vez la declaración está compuesta por una propiedad y el valor que se le asigne.

`h1 {color: red;}`

h1 es el selector

{color: red;} es la declaración

El selector funciona como enlace entre el documento y el estilo, especificando los elementos que se van a ver afectados por esa declaración. La declaración es la parte de la regla que establece cuál será el efecto. En el ejemplo anterior, el selector **h1** indica que todos los elementos **h1** se verán afectados por la declaración donde se establece que la propiedad **color** va a tener el valor **red** (rojo) para todos los elementos **h1** del documento o documentos que estén vinculados a esa hoja de estilos.

Las tres formas que he utilizado para dar estilo a un documento son las siguientes:

Utilizando una hoja de estilo externa que estará vinculada a un documento a través del elemento `<link>`, el cual debe ir situado en la sección `<head>`.

```
<!DOCTYPE html>
<html lang="en">
<html>
  <head>
    <title>Título</title>
    <link href="<?php echo base_url();?>assets/css/perfil.css"
rel="stylesheet">
  </head>
  <body>
    .
    .
    .
    .
  </body>
</html>
```

Utilizando el elemento `<style>`, en el interior del documento al que se le quiere dar estilo, y que generalmente se situaría en la sección `<head>`. De esta forma los estilos serán reconocidos antes de que la página se cargue por completo.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>hoja de estilo interna</title>
    <style>

      .navbar {
        margin-bottom: 0;
        border-radius: 0;
      }

      .sidenav {
        padding-top: 20px;
        background-color: #f1f1f1;
        height: 100%;
      }

      /* On small screens, set height to 'auto' for sidenav and grid */
      @media screen and (max-width: 767px) {
        .sidenav {
          height: auto;
          padding: 15px;
        }
      }

      ul {
        list-style-type: none;
      }

      .paginacion{
        background-color: #F5F5F5;
        padding: 20px 0 0 0;
```

```
        margin: auto;
        width: 500px;
        height: 80px;
        text-align: center;
        margin-top: 40px;

    }

.paginacion a{
    text-decoration: none;
    padding: 0;
    background-color: #4CAF50;
    color: white;
    border-radius: 5px;
    margin: 0;
    width: 80%;

}

.paginacion a:hover{
    background-color: #333333;
    color: #C0C0C0;

}

.pagination a:hover:not(.active) {
    background-color: #ddd;
    border-radius: 5px;
}

.pagination a.active {
    background-color: #4CAF50;
    color: white;
    border-radius: 5px;
}

.actual{
    color: #FFFFFF;
    padding: 4px;
}

.flex-container {
    display: -webkit-flex;
    display: flex;
    width: 100%;
    height: 100%;
    /* background-color: lightgrey; */
}

</style>
</head>
<body>
    <h1>Aquí se aplicará el estilo de letra para el Título</h1>
</body>
</html>
```

Y por ultimo Incluir CSS en los elementos HTML.

```
<div class="col-md-4" style="border-right-style:groove;height:80%>
```

En cuanto al Bootstrap, para utilizarlo, se puede descargar de dos maneras, compilado o mediante el código fuente original. En mi caso he utilizado la versión compilada, la cual tiene la siguiente estructura de archivos y directorios:

```
bootstrap/
  └── css/
    ├── bootstrap.css
    ├── bootstrap.min.css
    ├── bootstrap-theme.css
    └── bootstrap-theme.min.css
  └── js/
    ├── bootstrap.js
    └── bootstrap.min.js
  └── fonts/
    ├── glyphicons-halflings-regular.eot
    ├── glyphicons-halflings-regular.svg
    ├── glyphicons-halflings-regular.ttf
    └── glyphicons-halflings-regular.woff
```

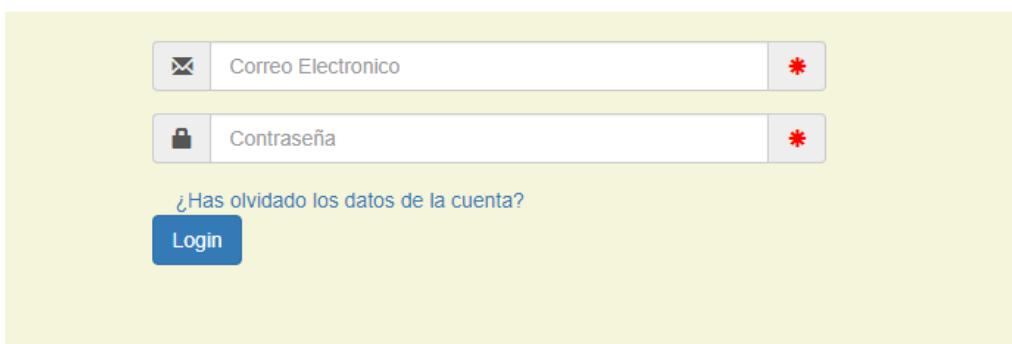
Después de descargarlo y ponerlo en una ruta especificada, para poder tener el acceso, y utilizar esas librerías, se debe incluir la ruta en el elemento <head> en el doc html como el ejemplo siguiente:

```
<link href="php echo<br/base_url(); ?>assets/include/css/bootstrap.min.css" rel="stylesheet">
```

Con este mecanismo, se puede acceder y utilizar la librería, como en este ejemplo:

```
<form action="php echo site_url('loginUser') ?&gt;" name="myForm"<br/class="form-horizontal" method="post">  
    <div class="form-group">  
        <div class="input-group">  
            <span class="input-group-addon"><i class="glyphicon glyphicon-envelope"></i></span>  
            <input type="text" name="correo" class="form-control" id="correoElectronico" placeholder="Correo Electronico" required><span  
class="input-group-addon"><i style="color:red;" class="glyphicon glyphicon-asterisk"></i></span>  
        </div>  
    </div>  
    <div class="form-group">  
        <div class="input-group">  
            <span class="input-group-addon"><i class="glyphicon glyphicon-lock"></i></span>  
            <input class="form-control" name="contraseña" type="password" id="password1" placeholder="Contraseña" required><span  
class="input-group-addon"><i style="color:red;" class="glyphicon glyphicon-asterisk"></i></span>  
        </div>  
    </div>  
    <div> <a href="php echo site_url("public/recoverPass") ?&gt;"¿Has  
olvidado los datos de la cuenta?</a></div>  
    <div class="form-group">  
        <input type="submit" value="Login" class="btn btn-primary">  
    </div>  
</form>
```

Como se observa todas las clases son de la librería Bootstrap, con la cual se le da un estilo a los elementos de html, en este caso al los elementos input tienen dos elementos laterales, uno que demuestra el tipo del elemento de forma grafica y el otro como es un campo obligatorio, también el botón de login, mediante la clase btn btn-primary se le da buen estilo, con color azul y esquinas redondas.



Javascript/Jquery

para dar dinamismo a las páginas web y incorporar efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario, he utilizado la librería JQuery, es una biblioteca multiplataforma de JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web.

Se puede incluir de dos formas, una interna y otra externa:

La forma interna, se trata de incluir la librería descargada de la pagina <http://getbootstrap.com/> oficial, o cualquier fuente, que la contiene, una vez esta descargada la librería, se incluye un link que contiene la ruta hasta su ubicación, este ejemplo refleja su la manera como se incluye en la aplicación web:

```
<script src="<?php echo base_url(); ?>assets/include/js/jquery-3.2.1.min.js"></script>
```

La forma externa, se puede utilizar desde Google o Microsoft. En mi caso he utilizado el enlace de Google.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>

<script src="https://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.2.1.min.js"></script>
```

En este ejemplo, se controla el campo nombre, mediante el cual se busca un usuario con su nombre, solo esta permitido nombres con letras, si no es asi, se le muestra un mensaje al usuario, y se desactiva el botón submit, hasta que el usuario corrija el error, una vez desaparece el error, el usuario puede realizar la consulta.

```
$("#boBuscarUsuarioN").click(function(event){  
    var valor= $("#usuarioNombre").val();  
    var characters = /^[a-z," "]+$/i;  
    if (!characters.test(valor)) {  
        $("#messageTipoN").html("El nombre solo debe contener letras");  
        return false  
    }else{  
        //$("#messageTipo").html("");  
    }  
});
```

Peticiones Ajax/json

En cuanto a la realización de peticiones de consulta de datos, para evitar duplicidad, a la hora de almacenar información en la base de datos y además sin recarga de la página, he utilizado la tecnología Ajax, la cual permite intercambiar información entre el servidor y el cliente (navegadores) sin la necesidad de recargar la página. De esta forma, ganamos en usabilidad, experiencia y productividad del usuario final.

En este ejemplo cuando el usuario pone el email en el campo de registro, una vez sale del campo, se realiza una petición al servidor, para comprobar si existe el dato:

```
$ (function () {
    $("#correoElectronico").blur(function (event)
    {

        event.preventDefault();
        var correo= $("#correoElectronico").val();
        var url= "<?php echo base_url(); ?>private/correo/" +correo;

        $.ajax(
        {
            type:"GET",
            url: url,
            success:function (response)
            {

                if(response=='false') {
                    $('#messageCorreo').html("Ya hay un usuario
con esta cuenta, introduzca otra cuenta");
                    $('form
input[id='boUsuario']").prop("disabled", true);
                }else{
                    $('#messageCorreo').html("");
                    $('form
input[id='boUsuario"]').prop("disabled", false);
                }

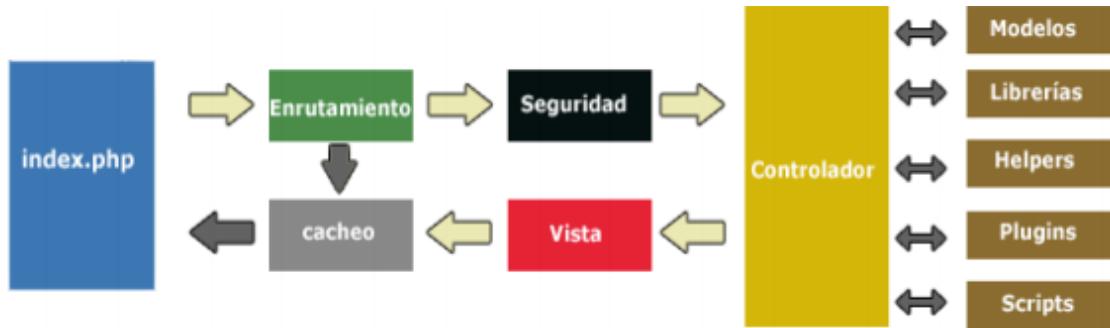
            },
            error: function(error)
            {
                $('#message').html(error);
            }
        }
    );
});
```

El servicio que responde la consulta anterior en el lado del servidor:

```
public function correo($correo) {
    $resultado=$this->RegistroModel->consultarCorreo($correo);
    if(!$resultado)
    {
        echo "false";
    }
    else
    {
        echo "true";
    }
}
```

- Back end

existe un procedimiento para atender una solicitud de página del cliente. Este proceso se realiza internamente por el propio CodeIgniter y de manera transparente. Durante el proceso participan varios módulos como el enrutamiento de la solicitud, la caché interna, etc.



el flujo de aplicación que implementa CodeIgniter, puedes seguir los siguientes puntos:

1. Toda solicitud de una página a partir de CodeIgniter comienza en un index.php que hay en la raíz del framework.
2. Luego se realiza un filtrado de la URL para saber cuál es elemento que tiene que procesar esta página.
3. Si la página se había generado antes y está en la caché de CodeIgniter, se devuelve el archivo de la caché ya generado, con lo que se ahorra procesamientos repetidos. La caché se puede configurar y si lo deseamos, incluso deshabilitar.
4. Antes de continuar con el proceso se realiza un tratamiento de seguridad sobre la entrada que tengamos, tanto de la información que haya en la URL como de la información que haya en un posible POST, si se ha configurado así.
5. El controlador adecuado realiza el procesamiento de la solicitud. CodeIgniter decide el controlador que debe procesar la solicitud en función de la URL solicitada.
6. El controlador comunica con una serie de módulos, los que necesite, para producir la página.
7. A través de las vistas adecuadas, el controlador genera la página, tal cual se tiene que enviar al navegador.
8. Si la página no estaba en la caché, se introduce, para que las futuras solicitudes de esta página sean más rápidas.

el funcionamiento empieza con un URL: para dirigirse al controlador, como unidad de control central entre la vista y el modelo, es necesario introducir un URL en la barra de búsqueda del navegador web. Para ello, los desarrolladores crean las denominadas **clases de controlador**, archivos PHP con diversas **funciones** que permiten cargar librerías, extensiones o helpers, establecer conexiones a bases de datos, integrar un modelo o seleccionar una vista determinada.

El flujo de aplicación de CodeIgniter se basa en el siguiente esquema de URL básico:

example.com/class/function/parameter

el caso de class/function existe la posibilidad que el usuario, pueda darle palabras intuitivas, por ejemplo si el usuario necesita registrarse, la clase puede ser Registro, y el método darseDeAlta, en el fichero de enrutamiento se puede relacionar cualquier ruta con la clase y el método, por ejemplo, si quisiera darse de alta, puede relacionar *esa ruta example.com/public/registro/parameter con el controlador y el método con la forma siguiente Registro/darseDeAlta, con eso el sistema llama al controlador Registro y al método darseDeAlta, prepara la vista y devuelve el resultado a la petición.*

CodeIgniter permite crear controladores individuales como clases definidas por el usuario. Para ello crean un archivo PHP separado para cada controlador en el directorio *application/controllers/*. Los controladores se crean como subclases de la clase *CI_Controller*. En el código fuente se realiza con ayuda de la palabra clave *extends*.

```
class Registro extends CI_Controller {  
}
```

Registro hereda todas las funciones de la visibilidad public y protected de la clase CI_Controller.

Cada clase también tiene un constructor, con lo cual se pueden integrar librerías, un modelo de datos, bases de datos o clases helper. Desde PHP5, se utiliza __construct() como estándar.

```
public function __construct() {  
    parent::__construct();  
  
    $this->load->database(); // podría hacerlo desde el autoload  
    $this->load->helper('url');  
    $this->load->library('session');  
    $this->load->model('CocheModel');  
  
}
```

Una vez creada la clase y definido el constructor, se crean los métodos, los cuales se invocan vistas o se interacciona con un modelo de datos integrado.

```
//funcion borrar ruta
public function borrarRutaCoches($id) {
    if($this->session->userdata('user')) {
        $data = array(
            'id' => $id,
        );
        $Resultado['ruta'] = $this->RutaModel-
>borrarRutaUsuario($data);
        //echo "<pre>";
        // print_r($Resultado);
        // echo "<pre>";
        $data2 = array(
            'usuario' => $this->session->userdata('user')->correo,
        );
        $output['rutas'] = $this->RutaModel-
>obtenerRutasUsuario($data2);
        if ($Resultado == true) {
            $output['Exito'] = 'exito';
        } else {
            $output['Error'] = 'error';
        }

        $this->load->view('private/listadoRutas', $output);
    }else{
        $this->load->view('public/home');
    }
}
```

En el ejemplo anterior, después que un usuario intente borrar una ruta, el cual se llama el método borrarRutaCoches, se comprueba si el usuario ha iniciado sesión, se coge el parámetro, se llama al modelo y se borra la ruta, prepara la respuesta, levanta la vista pasándole los datos del proceso, para reflejarlos al usuario, como resultado de la acción borrar ruta.

Para conectarse con la base de datos, y recuperar información solicitada, se utilizan los Modelos, Las clases modelo permiten a los desarrolladores definir funciones de forma individual para las operaciones de bases de datos. En este ejemplo se crea el modelo coche con el cual se realizan todas las operaciones posibles con la entidad coche, por ejemplo insertar, consultar, borrar y actualizar.

```
class CocheModel extends CI_Model
{
    function insertaCoche($data)
    {
        //Inserta coche en la base de datos
        $this->db->insert('coche', $data);
        return ($this->db->affected_rows() > 0);

    }
}
```

En el ejemplo anterior, en el modelo insertacoche, recibe los datos de un coche y los almacena en la base de datos.

En este ejemplo se hace una consulta para recuperar todos los datos del coche.

```
function get_contents($data)
{
    $this->db->select('*');
    $this->db->from('coche');
    $this->db->where('usuario', $data['usuario']);
    // $this->db->where('contraseña', $data['contraseña']);

    $query = $this->db->get();
    return $result = $query->row();
}
```

Enrutamiento con CodeIgniter

La dirección URL solicitada por el usuario, indica a CodeIgniter qué Controlador y qué método son invocados. Para ello, el framework utiliza el esquema clase/función/parámetro, un esquema básico que se puede modificar según las necesidades. CodeIgniter dispone del archivo routes.php en el directorio application/config/ que contiene un array denominado \$route, que es el que permite a los desarrolladores definir sus propios criterios de enrutamiento.

```
$route['registro'] = 'registro/index';
$route['registroUser'] = 'registro/indexRegister';
$route['confirmRegistro/(:any)'] = 'registro/confirmRegistro/$1';

$route['login'] = 'login/index';
$route['loginUser'] = 'login/indexLogin';
$route['cerrarSesion'] = 'login/unset_session_data'; // cerrar sesión

$route['private/modifyPerfil'] = 'modifyPerfil/index';
$route['private/coche'] = 'coche/index';
// foto del perfil
$route['private/fotoPerfil'] = 'foto/index';
// borrar foto
$route['private/borrarFoto'] = 'foto/borrarFoto';
// actualizar usuario
$route['updateUser'] = 'modifyPerfil/updateUser';
```

Cuando se requiere crear una regla de enrutamiento nueva para una dirección dinámica, los desarrolladores web disponen con CodeIgniter de dos opciones: definir entradas de enrutamiento para URL dinámicos con Wildcards (comodines) o con expresiones regulares.

El documento routes.php soporta dos tipos de comodín:

Comodines de routes.php	Descripción
:num	Actúa de comodín para números enteros
:any	Actúa de comodín para una cadena de caracteres (string)

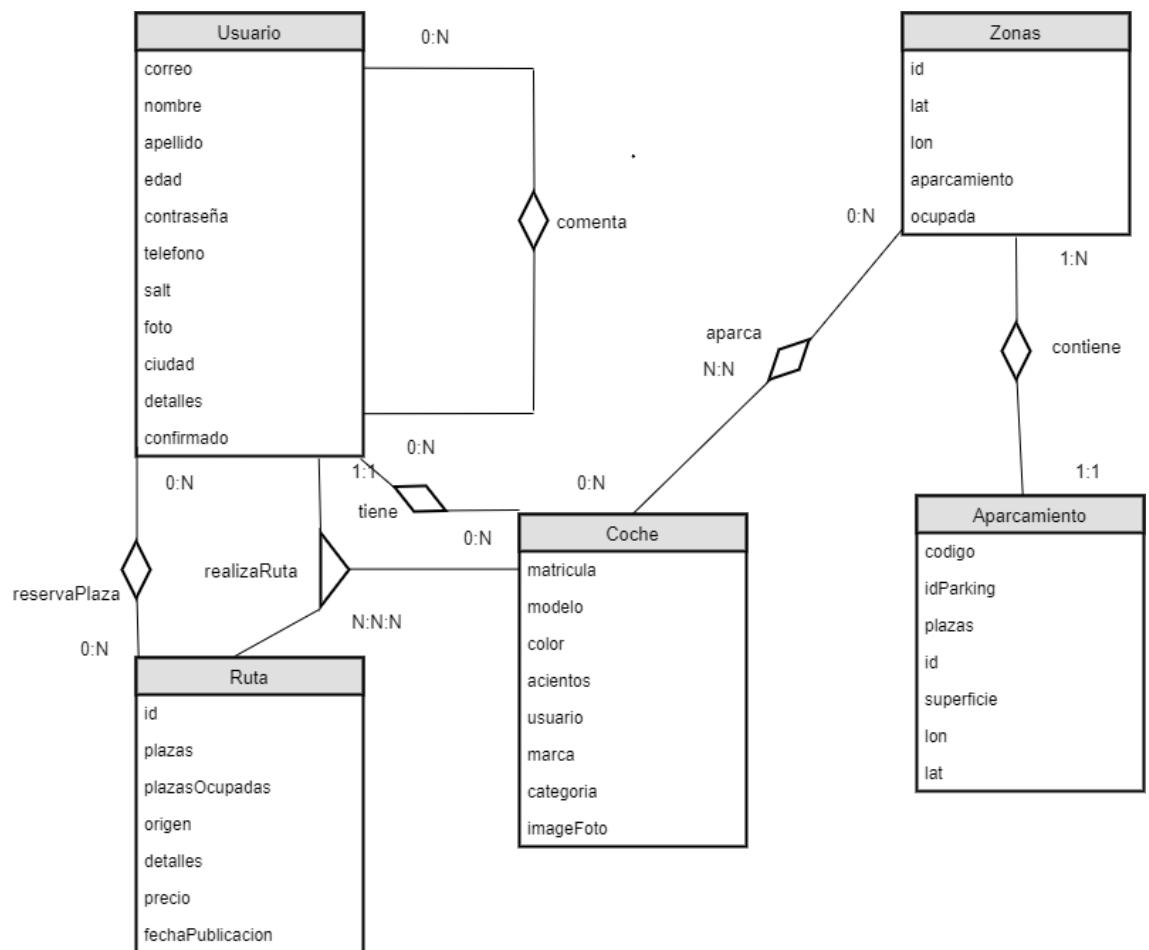
Aquí hay ejemplos.

```
$route['private/buscarAnuncios/(:num)'] = 'ruta/buscarAnuncios/$1';
//cambiar foto coche
$route['private/cambiarFotoCoche/(:any)'] = 'coche/fotoCoche/$1';
```

Base de datos

Esquema conceptual

Para visualizar la base de datos de una forma más clara he tomado una captura un esquema Entidad-Relación que representa las entidades de la base de datos así como las relaciones entre las distintas entidades



Modelo Relacional

El modelo relacional nos aporta de una forma sencilla e intuitiva la representación de la base de datos.

Usuario(correo, nombre, apellidos, edad, contraseña, teléfono, salt, foto, ciudad, detalles, confirmado,opcion)

C.P.: correo

V.N.N.: correo, nombre, apellidos, edad, contraseña, teléfono, salt, ciudad, detalles, confirmado,opcion.

Coche(matricula, modelo, color, acientos, usuario, marca, categoría, iamgeFoto)

C.P.: matricula

C.aj.: usuario -> usuario

VNN.: modelo, color, asientos, usuario, marcia, categoría.

Ruta(id, plazas, plazasOcupadas, origen, detalles , precio, fechaPublicacion)

C.P.: Id

V.N.N.: plazas, plazasOcupadas, origen, destino, precio, fechaPublicacion.

Aparcamiento(Id, código, idParking, plazas, superficie, lon, lat)

C.P.: Id

V.N.N.: código, idParking, plazas, superficie, lon, lat.

Realiza_Ruta(usuario, coche, ruta)

C.P.: (usuario, coche, ruta)

C.aj.: usuario -> usuario

C.aj.: coche -> coche

C.aj.: ruta -> ruta

ReservaPlaza(usuario,ruta)

C.P.: (usuario, ruta)

C.aj.: usuario -> usuario

C.aj.: ruta -> ruta

Comenta(UsuarioA, UsuarioB, Comentario)

C.P.: (UsuarioA, UsuarioB)

C. Ajena: UsuarioA -> Usuario

C. Ajena: UsuarioB -> UsuarioB

V.N.N.: Comentario

Zonas(id, lat, lon, aparcamiento, ocupada)

C.P.: (id)

C. Ajena: aparcamiento -> aparcamiento

V.N.N.: lat, lon, aparcamiento, ocupada.

UsuarioAparcaCoche(coche, zona, fecha)

C.P.: (coche, zona)

C. Ajena: coche -> coche

C. Ajena: zona -> zonas

V.N.N.: fecha.

Diccionario de Datos

Otra forma de representar la información que contienen las tablas es a través del diccionario de datos, por ejemplo el diccionario de datos de la tabla usuario es la siguiente:

Donde podemos observar el nombre de la columnas, el tipo de datos, si es posible que el valor de los campos sea nulo, el valor por defecto que tendrán los campos si no se establecen y el comentario que sirve como nota aclaratoria.

Usuario

Columna	Tipo	Nulo	Predeterminado	Comentarios
Correo	Varchar	No		Correo del usuario
Nombre	Varchar	No		Nombre del usuario
Apellido	Varchar	No		Apellido del usuario
Edad	Int	No		Edad del usuario
Contraseña	Varchar	No		Contraseña del usuario para login
Telefono	Varchar	No		Teléfono del usuario
Salt	Varchar	No		Salt generada para crear la contraseña junto con el hash
Foto	Varchar	Sí	Null	Foto del usuario
detalles	Varchar	No		Detalles del usuario
Confirmado	Enum	No	0	1 indica que el usuario ha confirmado su registro

				mediante el correo recibido, o todavía el usuario no ha confirmado su registro
--	--	--	--	--

Coche

Columna	Tipo	Nulo	Predeterminado	Comentarios
matricula	Varchar	No		Correo del coche
modelo	Varchar	No		Modelo del coche
Color	Varchar	No		Color del coche
Asientos	Int	No		Acientos del coche
Usuario	Varchar	No		Usuario dueño del coche
Marca	Varchar	No		Marca del coche
Categoría	Varchar	No		Categoría del coche
ImagenFoto	Varchar	Sí	Null	Foto del coche

Ruta

Columna	Tipo	Nulo	Predeterminado	Comentarios
id	Int	No		Identificador de la ruta
Plazas	Int	No		Plazas disponibles
Plazas ocupadas	Int	No		Plazas ocupadas, es decir, las que han sido solicitadas por los usuarios
Origen	VARCHAR	No		Punto de partida, es el origen donde sale el usuario
Detalles	VARCHAR	No		Detalles del viaje
Precio	Decimal	No		Precio por cada persona
FechaPublicacion	Date	No		Fecha cuando el usuario se ha dado de alta a la publicación de la ruta

Aparcamiento

Columna	Tipo	Nulo	Predeterminado	Comentarios
Id	Int	No		Numero cremental, se asigna a cada parking
Codigo	Varchar	No		Identificador del parking
IdParking	Int	No		Identificador de actividad según la tipología del aparcamiento
Plazas	Int	No		Numero de plazas que contiene
Superficie	Double	No		La superficie del parking
Lon	Double	No		Cordinada y
Lat	Double	No		Cordinada X

Zonas

Columna	Tipo	Nulo	Predeterminado	Comentarios
Id	Int	No		Identificador de la zona
Lat	Double	No		Cordinada x de la zona respecto del gps
Lon	Double	No		Cordinada y de la zona respecto a gps
Aparcamiento	Varchar	No		El aparcamiento al que pertenece la zona
Ocupada	Enum	No		0 indica que la zona no esta ocupada, 1 indica el contrario

Realiza ruta

Columna	Tipo	Nulo	Predeterminado	Comentarios
Usuario	Varchar	No		Usuario que publica la ruta
Coche	Varchar	No		Coche del usuario que publica la ruta
Ruta	Int	No		Identificador de la ruta

Reserva Ruta

Columna	Tipo	Nulo	Predeterminado	Comentarios
Usuario	Varchar	No		Usuario que publica la ruta
Ruta	Int	No		Identificador de la ruta

Usuario aparca coche

Columna	Tipo	Nulo	Predeterminado	Comentarios
Coche	Varchar	No		Coche la cual aparcá en una zona de un aparcamiento dado
Zona	Int	No		Zona de un aparcamiento
Fecha	Date	No		Fecha cuando el coche se aparcó

Comenta

Columna	Tipo	Nulo	Predeterminado	Comentarios
Id	Int	No		Identificador del comentario
UsuarioComentado	Varchar	No		Usuario que recibe el comentario
UsuarioComenta	Varchar	No		Usuario que realiza el comentario
Comentario	Varchar	No		El comentario introducido por el usuario

SEGURIDAD

Balancear riesgo y usabilidad

Si bien la usabilidad y la seguridad en una aplicación web no son excluyentes la una de la otra, alguna medida tomada para incrementar la seguridad con frecuencia afecta a la usabilidad. Es conveniente emplear medidas de seguridad que sean transparentes a los usuarios y que no resulten engorrosas en su empleo. Por ejemplo, el uso de un login que solicita el nombre de usuario y contraseña, permite controlar el acceso de los usuarios hacia secciones restringidas de la aplicación.

Rastrear el paso de los datos

Existen funciones globales en la aplicación que sirven para identificar de forma clara las entradas proporcionadas por el usuario. En el caso de nuestra web se ha utilizado Session, mediante la cual se almacena la información del usuario y con esto se puede seguir el rastro durante su recorrido en la aplicación.

XSS Prevention

XSS significa cross-site scripting. CodeIgniter viene con XSS filtro de seguridad. Ese filtro evita cualquier javascript malicioso o cualquier código que intenta crear cookies y hacer actividades maliciosas, para filtrar los datos a través de XSS, se usa el método clean como se muestra abajo.

```
$data = $this->security->xss_clean($data);
```

Ademas de filtrar código, también existe otra manera para filtrar la fotos que se suben a la base de datos, poniendo otro campo opcional como true.

```
$data = $this->security->xss_clean($data, true);
```

Ataques de inyección SQL

Para evitar los ataques mediante inyección de SQL se filtrarán las entradas de datos para evitar los caracteres “especiales” que permiten realizar dichos ataques. Es muy importante realizar el filtrado en todas las posibles entradas de textos porque un ataque de inyección SQL puede suponer una gran pérdida de datos.

Contraseña del usuario

La seguridad de contraseñas es una de las partes mas importantes de la aplicación, así para evitar daños, o robo de información y como consecuencia manipular y poner en peligro información de los usuarios, he utilizado un cifrado muy complejo, utiliza SHA-512 como HASH y un salt aleatoria que se mezcla con el hash y construir la contraseña mediante el método crypt.

Cuando se lleva acabo el desarrollo, se guarda el resultado después de aplicar el método crypt, como contraseña cifrada, además se guarda el salt, porque cuando se realiza el login, se aplica el mismo método, el usuario introduce su contraseña, se le aplica el hash, se consulta el salt creado mediante el correo cuando se hizo el registro, se hace la mezcla y se compara con la contraseña cifrada en la base de datos.

Aquí esta el ejemplo del registro

```
public function indexRegister()
{
    $pass = hash('sha512', $this->input->post('password'));
    $salt = uniqid(mt_rand(), true);
    $pass = crypt($pass, $salt);
    $correo = $this->input->post('correo');
    $data = array(
        'correo' => $this->input->post('correo'),
        'nombre' => $this->input->post('nombre'),
        'apellido' => $this->input->post('apellido'),
        'edad' => $this->input->post('bday'),
        'contraseña' => $pass,
        'telefono' => $this->input->post('telefonoReg'),
        'detalles' => $this->input->post('detalles'),
        'salt' => $salt
    );

    if ($this->send_mail($correo)) {
        $this->RegistroModel->insertaUsuario($data);
        $Resultado = $this->RegistroModel->get_contents($data);
        $data['message'] = $Resultado;
    } else {
        $data['message'] = "el correo introducido no es
incorrecto";
    }
    $this->load->view('public/confirmRegister', $data);
}
```

Y aquí esta el ejemplo del login:

```
public function indexLogin()
{
    //generar el hash a traves de la contraseña propuesta del usuario
    $pass = hash('sha512', $this->input->post('contraseña'));
    $user = $this->input->post('correo');
    $data = array(
        'correo' => $user,
    );
    $this->load->model('LoginModel');
    $salt=$this->LoginModel->get_salt($data);

    if(!empty($salt)) {
        $pass=crypt($pass,$salt);
        $data2 = array(
            'correo' => $this->input->post('correo'),
            'contraseña' => $pass
        );
        $Resultado = $this->LoginModel->get_contents($data2);

        if (!empty($Resultado)) {
            if ($Resultado->confirmado == 'SI') {

                //Se guarda el objeto
                $this->session->set_userdata('user', $Resultado);

                redirect('/');
            }else{
                $output['Error'] = 'Debes confirmar tu registro mediante el correo mandado con la direccion registrada';
            }
        }else{
            $output['Error'] = 'La contraseña es incorrecta';
        }
    }else{
        $output['Error'] = 'Los datos son incorrectos';
    }
    $this->load->view('public/log', $output);
}
```

APIS UTILIZADAS

Api Google maps

Para reflejar los parkings de la ua, y facilitar a los usuarios ver los parkings y las zonas que se dispone la universidad, he utilizado la Api google maps que permite agregar un mapa de Google con un marcador al sitio web y así demostrar las zonas con puntos rojos.

Los pasos a seguir para crear un mapa de Google con un marcador en la página web:

Paso 1: En la pagina HTML, creo un elemento div con un identificador:

```
div id="map" style="width: 100%; height: 600px;"></div>
```

Paso 2: Agrega un mapa con un marcador

```
<script>
//var jsn = [];
var jsn = <?php if (isset ($Resultado)) {
    echo json_encode($Resultado);?>

jsn = JSON.parse(jsn);

<?php } else{ echo json_encode($corrdenes); }
?>

function initMap() {

    var posArray = [];
    var count = 0;
    for(var i=0; i<jsn.length; i++) {
        //if (jsn[i].ocupada == '1') {
        <?php if (isset ($Resultado)){ ?>
            posArray[count] = {lat: parseFloat(jsn[i].lat), lng:
parseFloat(jsn[i].lon), id: jsn[i].id}
            count++;
        <?php } else{?>
            posArray[i] = {lat: parseFloat(jsn[i].lat), lng:
parseFloat(jsn[i].lon), id: jsn[i].id}
        <?php } ?>

        //}
    }

    var center = {lat: 38.385189, lng: -0.514053};

    var map = new google.maps.Map(document.getElementById('map'),
{
    zoom: 15,
    center: center,
    mapTypeId: 'satellite'
});

    var markerLabel = 'Z ';
    for (i = 0; i < posArray.length; i++) {
        console.log(posArray[i]);
        var marker = new google.maps.Marker({
            position: posArray[i],
            map: map,
            title: ""+i,
            label: {
                text: markerLabel+(posArray[i].id),
                color: "#FFFFFF",
                fontSize: "10px",
                fontWeight: "bold",
                borderColor: "#000000",
            }
        });
    }
}

</script>
```

Paso 3: Obtener una clave de API

Para conseguir la clave, ha de seguir unos pasos y conseguir la clave, como se refleja aquí.

Credenciales

Añadir credenciales al proyecto

- Averigua qué tipo de credenciales necesitas
Llamando a Google Maps JavaScript API

- 2 Ya tienes unas credenciales adecuadas para este fin

¿No quieres usar esta clave de API ya disponible? [Crear una nueva clave de API](#)

API key

Clave	AlzaSyAbvgfahBKagC52oJxu7necSaHbVBKIXzA
Tipo	Ninguna
Fecha de creación	5 may. 2017 18:54:39

[Listo](#) [Cancelar](#)

Y por ultimo agregar la clave al código.

```
<script async defer
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyAG9NywQiRlbr3FJu
UhYBFCSFXZI79IF-tE&callback=initMap">
</script >
```



SIGUA

SIGUA proporciona un interfaz de programación (API) para desarrolladores de aplicaciones que necesiten consumir datos de la base de datos geográfica. Se trata de un servicio web de tipo REST (*Representational State Transfer*), un estilo de arquitectura de software que emplea el conjunto básico de instrucciones HTTP para realizar operaciones de creación, lectura, modificación y borrado de datos en entornos web en contraposición a protocolos más complejos como SOAP o RPC. El API REST de SIGUA está basado en Slim para PHP 5, un *micro framework* de código abierto, y su principal consumidor es la propia aplicación WebGIS de SIGUA.

El API REST se estructura en varios niveles, entre los que destacan los de acceso público, que pueden ser consumidos tanto por las unidades y departamentos de la Universidad de Alicante que lo precisen como por desarrolladores externos. En este sentido, se ofrecen recursos para la obtención de datos referidos a entidades comunes de la base de datos geográfica (sedes, edificios, estancias, actividades, ubicación de puestos de trabajo, etc)

Para consumir los recursos públicos del API REST se debe consultar previamente la documentación y utilizar el punto de acceso cuyas URL se indican a continuación:

URL de la Documentación	URI Base
https://bitbucket.org/SIGUA/apirest-doc/src/master/specs.mkd	http://www.sigua.ua.es/api

Esquema:

```
        "required":true,
        "properties":{
            "codigo": {
                "type":"string",
                "required":true,
                "description":"Código SIGUA del aparcamiento"
            },
            "lon": {
                "type":"number",
                "required":true,
                "description":"Longitud del centroide en WGS84"
            }
            "lat": {
                "type":"number",
                "required":true,
                "description":"Latitud del centroide en WGS84"
            }
            "denominacion": {
                "type":["string", "null"],
                "required":true,
                "description":"Denominación conocida o código corporativo del
aparcamiento"
            },
            "id_actividad": {
                "type":"integer",
                "required":true,
                "description":"Identificador de actividad según la tipología del
aparcamiento"
            },
            "nombre_actividad": {
                "type":"string",
                "required":true,
                "description":"Denominación de la tipología del aparcamiento"
            },
            "superficie": {
                "type":"number",
                "required":true,
                "description":"Superficie en metros cuadrados"
            },
            "plazas": {
                "type":"number",
                "required":true,
                "description":"Número de plazas"
            }}}},
        "type": {
            "type":"string",
            "required":true
        }}}},
    "type": {
        "type":"string",
        "required":true
    }}}
```

GET Requests:

- Recupera todas las zonas de aparcamiento en batería
 - /pub/aparcamientos/bateria/items
- Recupera todas las zonas de aparcamiento en línea
 - /pub/aparcamientos/linea/items
- Recupera todas las zonas de aparcamiento de autobuses
 - /pub/aparcamientos/bus/items
- Recupera todas las plazas de aparcamiento adaptadas
 - /pub/aparcamientos/adaptado/items

PRUEBAS

Para realizar las pruebas sobre la aplicación voy a realizar pruebas funcionales con la herramienta Selenium IDE, El cual es un conjunto de herramientas de pruebas open-source para automatizar pruebas funcionales sobre aplicaciones Web.

La forma más sencilla de escribir scripts de pruebas con Selenium es utilizar la herramienta Selenium IDE. Se instala como un plugin de Firefox, y nos permite crear scripts de pruebas utilizando la aplicación web tal y como un usuario haría normalmente: a través del navegador. Además, el mismo script de pruebas se puede ejecutar en diferentes navegadores

El objetivo de realizar pruebas es encontrar el mayor número de errores posibles realizando el menor número de pruebas. Las pruebas son importantes porque de ellas dependen el éxito del proyecto, las pruebas las realizamos para contribuir al éxito del proyecto en la satisfacción al cliente. Las pruebas más relevantes realizadas son las siguientes:

Casos de prueba de la pantalla de registro.

registro		
open	http://localhost/proyecto-master/	
clickAndWait	link=Register	
type	id=inputNombre	usuario300
type	id=Apellido	usuario300
type	id=correoElectronico	usuario.gmail.com
fireEvent	id=correoElectronico	blur
verifyText	id=messageCorreο	El formato del correo no es correcto
type	id=correoElectronico	usuario30@gmail.com
fireEvent	id=correoElectronico	blur
verifyText	id=messageCorreο	Ya hay un usuario con esta cuenta, introduzca otra cuenta
type	id=correoElectronico	occc10@hotmail.com
fireEvent	id=correoElectronico	blur
type	id=telefonoReg	56478
fireEvent	id=telefonoReg	blur
verifyText	id=messageReg	El telefono debe empezar por 6 o 7
type	id=telefonoReg	654789
fireEvent	id=telefonoReg	blur
verifyText	id=messageReg	El numero debe contener 9 cifras
type	id=telefonoReg	654789654
fireEvent	id=telefonoReg	blur
type	id=passwordReg	kjbjhff
fireEvent	id=passwordReg	blur
verifyText	id=passwordRegid	La contraseña debe contener mas de 7 caracteres
type	id=passwordReg	aaaaaaa1
fireEvent	id=passwordReg	blur
verifyText	id=passwordRegid	La contraseña debe contener al menos una letra mayuscula
type	id=passwordReg	aaaaaaAA

fireEvent	id=passwordReg	blur
verifyText	id=passwordRegid	La contraseña debe contener al menos un digito
type	id=passwordReg	aaaaaaA1
fireEvent	id=passwordReg	blur
type	id=bdays	22
type	id=detalles	suelo salir de Alicante sobre las 8 de la
type	id=bdays	22
clickAndWait	id=boUsuarioReg	
verifyText	css=h1	Informacion
verifyText	css=strong	Success!

Como se ve en la pantalla anterior, en la primera columna contiene los comandos, la segunda contiene la localización del elemento sobre el cual realizar el evento y por ultimo el valor, los comandos se pueden generar automáticamente cuando el usuario interactua con la aplicación a testear o crear los comandos de forma manual.

El caso anterior, se comprueba todos los datos y sus posibles mensajes cuando hay un error, se introduzca un dato incorrecto y cuando se dispara un evento mediante jquery se crea un elemento con un mensaje del tipo del error y luego se comprueba con selenium IDE si el resultado es el deseado, Como resultado después de ejecutar el tests

The screenshot shows the Selenium IDE interface with the following details:

- Title Bar:** registro (untitled suite) - Selenium IDE 2.9.1
- Toolbar:** Archivo (F), Editar, Actions, Options, Ayuda; Base URL: http://localhost/
- Test Case Panel:** Shows "registro" as the current test case.
- Command Table:** A table view of the recorded test steps. The columns are Command, Target, and Value.

Command	Target	Value
verifyText	id=passwordRegid	La contraseña debe contener...
type	id=passwordReg	aaaaaaA1
fireEvent	id=passwordReg	blur
type	id=bdays	22
type	id=detalles	suelo salir de Alicante sobre l...
type	id=bdays	22
clickAndWait	id=boUsuarioReg	
verifyText	css=h1	Informacion
verifyText	css=strong	Success!

- Log Panel:** Displays the command history and results of the execution.

```
[info] Executing: |type | id=bdays | 22 |
[info] Executing: |type | id=detalles | suelo salir de Alicante sobre las 8 de la |
[info] Executing: |type | id=bdays | 22 |
[info] Executing: |clickAndWait | id=boUsuarioReg | |
[info] Executing: |verifyText | css=h1 | Informacion |
[info] Executing: |verifyText | css=strong | Success! |
[info] Test case passed
```

Casos de prueba de la pantalla de intento Inicio de Sesión con mensaje de confirmar rehistro

En este caso cuando el usuario se registra y todo es correcto, cuando intenta iniciar sesión sin confirmar el registro a través de un correo mandado, le sale el mensaje que debe confirmar el registro, con este script se comprueba que si un usuario no confirma el registro no inicia sesión y le sale el mensaje correspondiente.

login		
open	http://localhost/proyecto-master/	
clickAndWait	link=Login	
verifyText	css=h1	Identificate
type	id=correoElectronico	occc10@hotmail.com
type	id=password1	aaaa
clickAndWait	css=input.btn.btn-primary	
verifyText	css=div.alert.alert-danger	Info! La contraseña es incorrecta
type	id=correoElectronico	occc10@hotmail.com
type	id=password1	aaaaaaA1
clickAndWait	css=input.btn.btn-primary	
verifyText	css=div.alert.alert-danger	Info! Debes confirmar tu registro mediante el correo mandado con la dirección registrada

Resultado de ejecución de este script

The screenshot shows the Selenium IDE interface with the following details:

- Test Case:** login
- Runs:** 1
- Failures:** 0
- Log:**
 - [info] Executing: |verifyText | css=div.alert.alert-danger | Info! La contraseña es incorrecta |
 - [info] Executing: |type | id=correoElectronico | occc10@hotmail.com |
 - [info] Executing: |type | id=password1 | aaaaaaA1 |
 - [info] Executing: |clickAndWait | css=input.btn.btn-primary |
 - [info] Executing: |verifyText | css=div.alert.alert-danger | Info! Debes confirmar tu registro mediante el correo mandado con la dirección registrada |
 - [info] Test case passed

Casos de prueba de la pantalla de Inicio de Sesión después de confirmar registro

loginConfirm		
open	http://localhost/proyecto-master/	
clickAndWait	link=Login	
verifyText	css=h1	Identificate
type	id=correoElectronico	occc10@hotmail.com
type	id=password1	aaaaaaAa
clickAndWait	css=input.btn.btn-primary	
verifyText	css=div.alert.alert-danger	Info! La contraseña es incorrecta
type	id=correoElectronico	occc10@hotmail.com
type	id=password1	aaaaaaA1
type	id=correoElectronico	occ@hotmail.com
clickAndWait	css=input.btn.btn-primary	
verifyText	css=div.alert.alert-danger	Info! Los datos son incorrectos
type	id=password1	
type	id=correoElectronico	occc10@hotmail.com
type	id=password1	ddddddD1
clickAndWait	css=input.btn.btn-primary	
verifyText	css=h1	Bienvenido a la universidad de Alicante
click	//button[@type='button']	
clickAndWait	link=Editar perfil	
verifyText	css=h1	Información personal
click	css=body	
click	//button[@type='button']	
clickAndWait	link=Cerrar sesion	

Ejecución del test.

loginConfirm (untitled suite) - Selenium IDE 2.9.1

Archivo (F) Editar Actions Options Ayuda

Base URL: http://localhost/

Test Case: loginConfirm

Runs: 1 Failures: 0

Table Source

Command	Target	Value
type	id=password1	aaaaaaA1
clickAndWait	css=input.btn.btn-primary	
verifyText	css=h1	Bienvenido a la universidad d...
click	//button[@type='button']	
clickAndWait	link=Editar perfil	
verifyText	css=h1	Información personal
click	css=body	
click	//button[@type='button']	
clickAndWait	link=Cerrar sesion	

Command: type
Target: id=password1
Value: aaaaaaA1

Log Reference UI-Element Rollup

type(locator, value)

Arguments:

- locator - an element locator
- value - the value to type

Sets the value of an input field, as though you typed it in.

Can also be used to set the value of combo boxes, check boxes, etc. In these cases, value should be the value of the option selected, not the visible text.

Caso de prueba para modificar un dato

modificarInformacion		
open	http://localhost/proyecto-master/	
clickAndWait	css=span.glyphicon.glyphicon-log-in	
type	id=password1	aaaaaaA2
type	id=correoElectronico	occc10@hotmail.com
type	id=password1	ddddddD1
clickAndWait	css=input.btn.btn-primary	
click	//button[@type='button']	
clickAndWait	link=Editar perfil	
type	name=bdy	24
storeValue	name=bdy	edad
clickAndWait	id=boModifyPefil	
verifyText	css=strong	Success!
clickAndWait	link=informacion personal	
echo	\${edad}	
verifyValue	name=bdy	\${edad}
click	//button[@type='button']	
clickAndWait	link=Cerrar sesion	
verifyText	css=h1	Bienvenido a la universidad de Alicante

Resultado ejecución del test

modificarInformacion (untitled suite) - Selenium IDE 2.9.1

Archivo (F) Editar Actions Options Ayuda

Base URL <http://localhost/>

Test Case
modificarInformacion

Runs: 1
Failures: 0

Table Source

Command	Target	Value
storeValue	name=bday	edad
clickAndWait	id=boModifyPerfil	
verifyText	css=strong	Success!
clickAndWait	link=informacion personal	
echo	\${edad}	

Command type
Target id=password1
Value aaaaaaA1

Log Reference UI-Element Rollup Info Clear

```
[info] Executing: |clickAndWait| link=Editar perfil ||  
[info] Executing: |type| name=bday | 24 |  
[info] Executing: |storeValue| name=bday | edad |  
[info] Executing: |clickAndWait| id=boModifyPerfil ||  
[info] Executing: |verifyText| css=strong | Success! |  
[info] Executing: |clickAndWait| link=informacion personal ||  
[info] Executing: |echo| ${edad} ||  
[info] echo: 24  
[info] Executing: |verifyValue| name=bday | ${edad} |  
[info] Executing: |click| //button[@type='button'] ||  
[info] Executing: |clickAndWait| link=Cerrar sesion ||  
[info] Executing: |verifyText| css=h1 | Bienvenido a la universidad de Alicante |  
[info] Test case passed
```

Caso de prueba insertar coche, publicar ruta, y actualizar información del anuncio.

plazasOcupadas		
open	http://localhost/proyecto-master/	
clickAndWait	link=Login	
type	id=correoElectronico	occc10@hotmail.com
type	id=password1	ddddddD1
clickAndWait	css=input.btn.btn-primary	
clickAndWait	link=Publicar anuncio	
verifyText	css=h3	Publicar Ruta
verifyText	css=div.alert.alert-info	Info! No tienes coche para publicar anuncio, ponga datos del coche, y luego, empieze a publicar anuncios.
click	//button[@type='button']	
clickAndWait	css=ul.dropdown-menu > li > a	
verifyText	css=h1	Información personal
clickAndWait	link=coche	
type	id=matricula	3652HHH
type	name=fotofile	C:\Users\walid\Desktop\Escritorio\ferrari.jpg
clickAndWait	id=boCoche	
verifyText	css=strong	Success!
clickAndWait	link=Publicar anuncio	
type	id=origen	Elche
type	id=precio	6
type	id=plaza	4
type	id=detalle	suelo salir una hora antes.
clickAndWait	css=input.btn.btn-primary	
verifyText	css=strong	Success!
clickAndWait	link=Viajes publicados	
verifyText	css=h2	Rutas publicadas
verifyText	css=td	Elche
verifyText	//td[2]	4

clickAndWait	link=Ver Detalles	
verifyText	css=td	Elche
clickAndWait	link=Viajes publicados	
clickAndWait	link=Actualizar	
type	id=plazaOcupadas	2
storeValue	id=plazaOcupadas	valor
clickAndWait	css=input.btn.btn-primary	
verifyText	css=h3	Actualizar Ruta
verifyText	css=strong	Success!
clickAndWait	link=Viajes publicados	
clickAndWait	link=Ver Detalles	
verifyText	//td[3]	\${valor}
selectWindow	null	
click	//button[@type='button']	
clickAndWait	css=ul.dropdown-menu > li > a	
clickAndWait	link=coche	
clickAndWait	link=Eliminar coche	
verifyText	css=strong	Success!
click	//button[@type='button']	
clickAndWait	link=Cerrar sesion	

Resultado ejecución del test

plazasOcupadas (untitled suite) - Selenium IDE 2.9.1

Archivo (F) Editar Actions Options Ayuda

Base URL <http://localhost/>

Test Case **plazasOcupadas**

Runs: 1 Failures: 0

Table Source

Command	Target	Value
open	http://localhost/proyecto-mast...	
clickAndWait	link=Login	
type	id=correoElectronico	occc10@hotmail.com
type	id=password1	aaaaaaA1
clickAndWait	css=input.htm.htm-primary	

Log Reference UI-Element Rollup Info Clear

```
[info] Executing: |verifyText | css=strong | Success! |
[info] Executing: |clickAndWait | link=Viajes publicados ||
[info] Executing: |clickAndWait | link=Ver Detalles ||
[info] Executing: |verifyText | //td[3] | ${valor} |
[info] Executing: |selectWindow | null ||
[info] Executing: |click | //button[@type='button'] ||
[info] Executing: |clickAndWait | css=ul.dropdown-menu > li > a ||
[info] Executing: |clickAndWait | link=coche ||
[info] Executing: |clickAndWait | link=Eliminar coche ||
[info] Executing: |verifyText | css=strong | Success! |
[info] Executing: |click | //button[@type='button'] ||
[info] Executing: |clickAndWait | link=Cerrar sesion ||
[info] Test case passed
```

Caso de prueba buscar aparcamiento.

buscarParking		
open	http://localhost/proyecto-master/	
clickAndWait	link=Login	
type	id=password1	aaaaaaA2
type	id=correoElectronico	occc10@hotmail.com
type	id=password1	ddddddD1
clickAndWait	css=input.btn.btn-primary	
clickAndWait	link=Buscar parking	
verifyText	css=h3	Buscar Parking
clickAndWait	css=input.btn.btn-primary	
verifyText	css=label	Elige Zona para aparcar y actualizar para indicar como zona ocupada:
clickAndWait	css=input.btn.btn-primary	
verifyText	css=h3	Zona ocupada
verifyText	css=div.alert.alert-info	Info! Debes desocupar la zona ocupada y luego puedes volver a aparcar.
clickAndWait	link=Buscar parking	
verifyText	css=h3	Buscar Parking
clickAndWait	css=input.btn.btn-primary	
verifyText	css=div.alert.alert-info	Info! Debes desocupar la zona ocupada y luego puedes volver a buscar aparcamiento.
clickAndWait	css=input.btn.btn-primary	
verifyText	css=h3	Buscar Parking
clickAndWait	css=input.btn.btn-primary	
verifyText	css=label	Elige Zona para aparcar y actualizar para indicar como zona ocupada:
clickAndWait	css=input.btn.btn-primary	
verifyText	css=div.alert.alert-info	Info! Debes desocupar la zona ocupada y luego puedes volver a aparcar.
clickAndWait	css=input.btn.btn-primary	
verifyText	css=h3	Buscar Parking
click	//button[@type='button']	
clickAndWait	link=Cerrar sesion	

Resultado ejecución del script:

The screenshot shows the Selenium IDE interface with the following details:

- Test Case:** buscarparking
- Runs:** 1
- Failures:** 0
- Log:**

```
[info] Executing: |clickAndWait| css=input.btn.btn-primary |
[info] Executing: |verifyText| css=div.alert.alert-info | Info! Debes desocupar la zona ocupada y luego puedes volver a buscar aparcamiento. |
[info] Executing: |clickAndWait| css=input.btn.btn-primary |
[info] Executing: |verifyText| css=h3 | Buscar Parking |
[info] Executing: |clickAndWait| css=input.btn.btn-primary |
[info] Executing: |verifyText| css=label | Elige Zona para aparcar y actualizar para indicar como zona ocupada: |
[info] Executing: |clickAndWait| css=input.btn.btn-primary |
[info] Executing: |verifyText| css=div.alert.alert-info | Info! Debes desocupar la zona ocupada y luego puedes volver a aparcar. |
[info] Executing: |clickAndWait| css=input.btn.btn-primary |
[info] Executing: |verifyText| css=h3 | Buscar Parking |
[info] Executing: |click| //button[@type='button'] |
[info] Executing: |clickAndWait| link=Cerrar sesion |
[info] Test case passed
```

CONCLUSIONES, TRABAJOS FUTUROS Y POSIBLES MEJORAS

REFERENCIAS

<https://www.formget.com/codeigniter-uri-segment/>

<https://validator.w3.org/nu/#textarea>

https://www.w3schools.com/bootstrap/bootstrap_templates.asp

<https://www.w3schools.com/html/default.asp>

<https://www.w3schools.com/css/default.asp>

<https://www.w3schools.com/js/default.asp>

<https://www.w3schools.com/sql/default.asp>

<https://www.w3schools.com/php/default.asp>

<https://www.w3schools.com/jquery/default.asp>

<http://librosweb.es/>

https://www.codeigniter.com/user_guide/

<https://codepen.io/benske/pen/iAgpq>

<https://web.ua.es/es/sigua/api-rest.html>

<https://bitbucket.org/SIGUA/apirest-doc/src/master/specs.mkd?fileviewer=file-view-default>

https://www.tutorialspoint.com/codeigniter/codeigniter_security.htm

http://www.w3ii.com/es/codeigniter/codeigniter_security.html

https://codeigniter.com/user_guide/database/queries.html