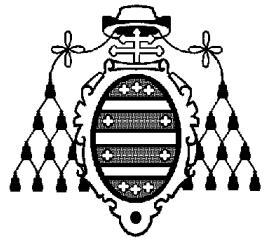




# **UNIVERSIDAD DE OVIEDO**



**ESCUELA POLITÉCNICA DE INGENIERÍA DE GIJÓN**

**PROYECTO FIN DE CARRERA**

**“Red social orientada a temática ciclista con geolocalización”**

**DIRECTOR: Fernando Álvarez García**

**AUTOR: Alejandro López Núñez**

**MAYO 2015**

**MEMORIA PRESENTADA POR**  
**D. Alejandro López Núñez**  
**PARA OPTAR AL TÍTULO DE**  
**INGENIERO TÉCNICO en Informática de Sistemas**



1. Prólogo.....	10
1.1. Descripción del Proyecto .....	10
2. Estudio de Viabilidad del Sistema .....	13
2.1. Establecimiento del Alcance del Sistema .....	13
2.2. Estudio de la Situación Actual.....	14
2.3. Estudio de las Alternativas existentes.....	15
2.4. Definición de Requisitos del Sistema.....	17
2.4.1. Requisitos Funcionales.....	17
2.4.2. Requisitos No Funcionales .....	25
2.5. Alternativas Server-Side.....	26
2.5.1. Elección del Gestor de Base de Datos .....	26
2.5.2. Elección del Lenguaje de Programación .....	28
2.5.3. Valoración de las alternativas .....	31
2.5.4. Descripción de las alternativas seleccionadas .....	32
3. Análisis del Sistema de Información .....	37
3.1. Descripción del Sistema.....	37
3.1.1. Determinación del Alcance del Sistema.....	37
3.1.2. Identificación del Entorno Tecnológico.....	38
3.1.3. Identificación de los Usuarios Participantes .....	38
3.2. Especificación del Sistema .....	38
3.2.1. Contexto del Sistema.....	38
3.3. Diagrama de Subsistemas .....	38
3.4. Especificación de los Subsistemas .....	41
3.4.1. Subsistema Gestión de Usuarios.....	41
3.4.2. Subsistema Gestión de Comentarios .....	56
3.4.3. Subsistema Gestión de Conversaciones .....	61
3.4.4. Subsistema Gestión de Rutas .....	66
3.4.5. Subsistema Gestión de Eventos.....	76
3.4.6. Subsistema Gestión de Contacto .....	85
4. Diseño del Sistema de la Información .....	88
4.1. Diagrama de despliegue .....	88
4.2. Modelo de Clases del Sistema.....	90
4.2.1. Diagrama de Clases .....	90
4.2.2. Descripción de las Clases.....	91
4.3. Comportamiento de las clases de análisis.....	106
4.3.1. Catálogo de eventos. Relación con los casos de uso. ....	106

4.3.2. Matriz Evento - Clase.....	108
4.4. Diagramas de secuencia.....	110
4.5. Diseño Físico de Datos .....	130
4.5.1. Estructura Física de la Base de Datos.....	130
4.5.2. Estructura de Colecciones.....	131
<b>5. Pruebas del Sistema.....</b>	<b>133</b>
5.1. Pruebas de Caja Negra .....	134
5.1.1. Conjunto de pruebas “Acceso de usuarios registrados” .....	134
5.1.2. Conjunto de pruebas “Registro de nuevo usuario” .....	136
5.1.3. Conjunto de pruebas de “Insertar nuevo comentario” .....	141
5.1.4. Conjunto de pruebas de “Insertar nueva ruta” .....	143
5.1.5. Conjunto de pruebas para “Insertar nuevo evento” .....	146
5.1.6. Conjunto de pruebas para “Insertar nueva conversación”.....	150
5.1.7. Conjunto de pruebas para “Enviar formulario de contacto” .....	152
<b>6. Presupuesto .....</b>	<b>155</b>
6.1. Planificación del proyecto .....	155
6.1.1. Desglose en etapas .....	155
6.1.2. Planificación temporal.....	156
6.2. Presupuesto para el desarrollo de la aplicación .....	157
6.2.1. Recursos hardware.....	157
6.2.2. Recursos software .....	157
6.2.3. Mano de obra.....	157
6.2.4. Detalle del coste .....	159
<b>7. Manual de Usuario.....</b>	<b>161</b>
7.1. Introducción a Voy en Bici.....	161
7.1.1. Registro de usuario.....	161
7.1.2. Recordar contraseña .....	162
7.1.3. Acceso a la plataforma.....	162
7.2. Acceso como usuario registrado .....	163
7.2.1. Enlaces rápidos de usuario.....	163
7.2.2. Mi muro .....	163
7.2.3. Escribir un comentario en mi muro .....	164
7.2.4. Editar un comentario escrito por mí .....	164
7.2.5. Eliminar un comentario .....	165
7.2.6. Buscador de usuarios .....	165
7.2.7. Ver perfil de un usuario registrado .....	166
7.2.8. Añadir un usuario a los usuarios que sigo .....	166

7.2.9. Quitar a un usuario de los usuarios que sigo .....	166
7.2.10. A quién sigo.....	167
7.2.11. Mis seguidores .....	167
7.2.12. Mis conversaciones .....	168
7.2.13. Detalle de una conversación.....	168
7.2.14. Escribir un mensaje en una conversación .....	169
7.2.15. Ver rutas de un usuario .....	169
7.2.16. Dar de alta una ruta .....	169
7.2.17. Edición de Ruta.....	171
7.2.18. Eliminar ruta.....	171
7.2.19. Buscar rutas.....	172
7.2.20. Detalle de una ruta.....	172
7.2.21. Descargar fichero de ruta .....	173
7.2.22. Comentar una ruta.....	173
7.2.23. Ver eventos de un usuario .....	174
7.2.24. Dar de alta un evento.....	174
7.2.25. Edición de Evento.....	175
7.2.26. Eliminar evento .....	175
7.2.27. Ver detalle de un evento .....	176
7.2.28. Comentar un evento .....	176
7.2.29. Buscar eventos .....	177
7.2.30. Mis favoritos.....	177
7.2.31. Añadir un elemento a favoritos .....	177
7.2.32. Quitar un elemento de favoritos .....	178
7.2.33. Formulario de contacto .....	178
7.3. Acceso como usuario Administrador .....	179
7.3.1. Pantalla inicial usuario administrador .....	179
7.3.2. Acceso a listado completo de usuarios.....	180
7.3.3. Editar datos de usuario .....	180
7.3.4. Eliminar usuario .....	180
7.3.5. Acceso a listado completo de Comentarios .....	181
7.3.6. Editar comentario.....	181
7.3.7. Eliminar comentario .....	181
7.3.8. Acceso a listado completo de Rutas .....	182
7.3.9. Editar ruta .....	182
7.3.10. Eliminar ruta .....	182
7.3.11. Acceso a listado completo de Eventos.....	183
7.3.12. Editar evento.....	183

7.3.13. Eliminar evento .....	183
<b>8. Manual del programador.....</b>	<b>185</b>
8.1. Instalación de la plataforma Voy en Bici .....	185
8.1.1. Requisitos Hardware.....	185
8.1.2. Requisitos Software .....	185
8.1.3. Instalación.....	186
8.2. Guía de instalación paso a paso.....	186
8.2.1. Actualización de repositorios de nuestro sistema.....	186
8.2.2. Instalar una versión de Ruby .....	186
8.2.3. Instalación de MongoDB.....	188
8.2.4. Arrancando Voy en Bici.....	189
8.2.5. Instalación de Nginx.....	190
8.2.6. Despliegues automatizados.....	192
8.3. Desarrollo del back-end .....	194
8.3.1. Estructura de carpetas.....	194
8.3.2. Ficheros de configuración.....	195
8.3.3. Ciclo de vida de una petición.....	196
8.3.4. Controladores en Ruby on Rails .....	197
8.3.5. Modelos en Ruby on Rails .....	200
8.3.6. Vistas en Ruby on Rails.....	204
8.3.7. Estructura del código fuente .....	205
<b>9. Anexo .....</b>	<b>213</b>
9.1. Conclusión del Proyecto .....	213
9.2. Futuras ampliaciones .....	214
9.3. Bibliografía .....	215
Libros, libros online y referencias consultadas en internet .....	215



# 1. Prólogo

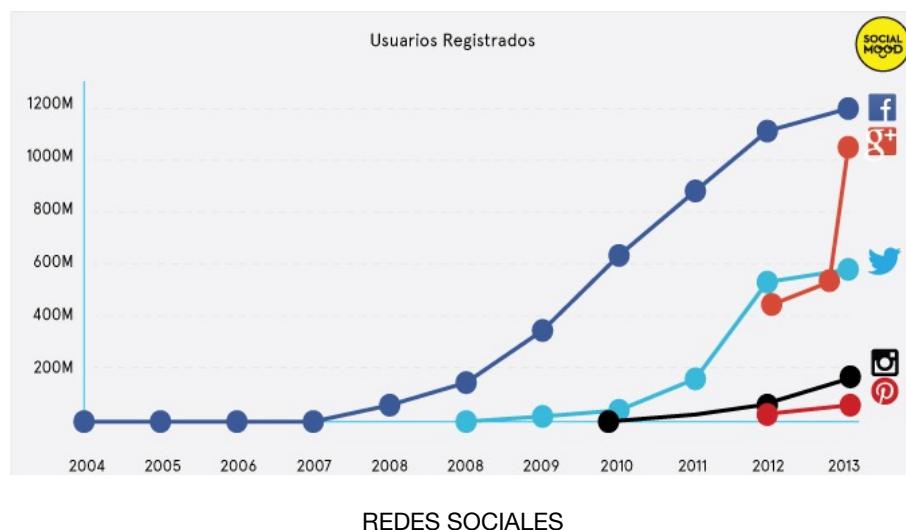
El proyecto se basa en la realización de una **aplicación web disponible tanto en dispositivos de escritorio como móviles** en la que facilitar la comunicación entre ciclistas y grupos de ciclismo.

En dicha aplicación se buscará que los propios usuarios participantes sean los que nutran de información a la aplicación de manera periódica con los nuevos eventos que aparezcan en sus distintas localidades así como con las rutas que realicen buscando de esta manera fomentar la participación de los usuarios.

## 1.1. Descripción del Proyecto

La idea del proyecto surge a través de conversaciones mantenidas entre compañeros en la que echábamos de menos un *punto de reunión de cicloturistas, un único lugar donde compartir rutas, eventos en los que participar y como ponerse en contacto con otros usuarios con las mismas aficiones.*

Toda estos datos permitirían a un usuario recibir información periódica de la manera más actualizada y cómoda posible, a través del acceso en la aplicación tanto desde un ordenador de escritorio como desde un móvil.



Para llevar a cabo la aplicación **se ha optado por la realización de una red social**. Actualmente las redes sociales acaparan al 20% de las personas que disponen de una conexión a internet, existiendo al menos 1200 millones de personas que disponen de un usuario en al menos una red social y accediendo de media un par de veces al día como norma habitual.

A todos estos datos hay que **sumar la movilidad y por tanto acceso directo a la información que nos ofrecen los dispositivos móviles**. El número de móviles en España ha superado al número de habitantes, esto ha implicado que el uso de internet desde móviles supere en la actualidad a las de dispositivos de escritorio y ésta cifra seguirá aumentando durante los próximos años.

Para la realización de una red social que cumpliese con los objetivos buscados se ha realizado un breve estudio de los puntos fuertes y débiles de cada una de las redes sociales, página disponibles con rutas, eventos y cómo aprovecharlas en un portal que fuese un nexo de unión entre ellas.

Uno de los aspectos más importantes y valorado durante el estudio y desarrollo es nuestro posible **target social** y es que se busca incluir a todas las edades posibles por lo que la sencillez era un elemento clave puesto que será un punto de unión tanto de gente joven con un uso habitual de las redes sociales y de personas de mayor edad que pueden estar menos habituadas al uso de las mismas.

Se busca facilitar que los usuarios conozcan a nuevas personas y que a la larga esto se pueda llegar a trasladar a una relación que les lleve fuera de la misma, facilitando la salida de rutas cicloturistas de grupos cada vez más grandes. Para ello se proveerán de los aspectos más comunes de las redes sociales facilitando a los usuarios conversaciones privadas y mensajería pública a todos los usuarios simulando un muro de mensajes.

La red social realizada **facilitará la visualización de las rutas y la descarga de ficheros GPX compatibles con los GPS disponibles en la actualidad** y permitir que otros usuarios escriban comentarios en las rutas ofreciendo nuevas posibilidades, compartiendo sus experiencias y conocer gente nueva a la que seguir y con la que quedar en un futuro para compartir sus aficiones a las rutas de ciclismo.

Por último se facilitará que los usuarios comparten eventos de manera que de lugar a que el mayor número posible de personas se enteren de la existencia de un evento que pueda suceder cerca de su zona.

El proyecto inicialmente estará localizado al territorio español y su idioma principal será el castellano aunque **la ejecución del mismo a nivel de desarrollo se planteará adaptado a otros idiomas** y se facilitará como ejemplo el idioma inglés pudiendo ser traducido a cualquier otro idioma que consideremos oportuno mediante diferentes ficheros de configuración.



## 2. Estudio de Viabilidad del Sistema

---

### 2.1. Establecimiento del Alcance del Sistema

Se desea realizar una aplicación que permita la **interacción entre usuarios de manera sencilla**, facilitando la labor de **compartir información relativa a elementos asociados al ciclismo entre usuarios con las mismas aficiones**.

En la aplicación desarrollada en este proyecto se identificarán distintos tipos de usuario que tendrán entornos distintos asociados a sus necesidades y permisos disponibles.

Los usuarios por tanto accederán a la plataforma para consultar, comentar, publicar novedades que consideren oportunas facilitando la interacción con otros usuarios del sistema.

En función del perfil de usuario se tendrá acceso a distintas funcionalidades:

#### **Administrador del sistema**

Es el encargado de controlar el funcionamiento de la aplicación.

Tendrá un *control total sobre los elementos que hay en la plataforma*, dispone de la posibilidad de editar y eliminar usuarios, crear, editar y eliminar rutas y eventos y filtrar los comentarios que haya en la plataforma.

Por tanto su labor será la de mantener una red social en la que se permita una libertad de expresión suficiente a los usuarios participantes pero controlando en cierto modo los mensajes que se produzcan en la misma.

#### **Usuario registrado**

Es un usuario registrado en la plataforma que tiene acceso y un *control total a todos los elementos que haya publicado él mismo*.

A este tipo de usuarios no se le asigna mayor privilegio o rol que el de poder editar su propia información, rutas y eventos creados así como la posibilidad de eliminarlos en un momento dado.

Como usuario registrado disponen también de la posibilidad de descargarse ficheros de rutas u otros elementos compartidos en eventos.

#### **Usuario anónimo**

Es un usuario no registrado que accede a la plataforma.

Este tipo de usuarios *no dispone de ningún tipo de privilegio más allá de la consulta de información disponible en el portal como son las rutas y eventos*. No pueden crear ningún

tipo de información que se comparta en el sistema ni descargarse elementos exclusivos de usuarios registrados.

Como es objetivo de este proyecto **la aplicación será una red social en tiempo real desarrollada** bajo arquitectura Web la cual permitirá una **visualización correcta en cualquier dispositivo** desde el que se visualice la plataforma.

Uno de los puntos más importantes y en los que más hincapié se realizará es la **velocidad de ejecución de la aplicación** ya que los estudios demuestran que *un 40% de los usuarios abandonan una web si ésta tarda más de 3 segundos en ejecutarse*.

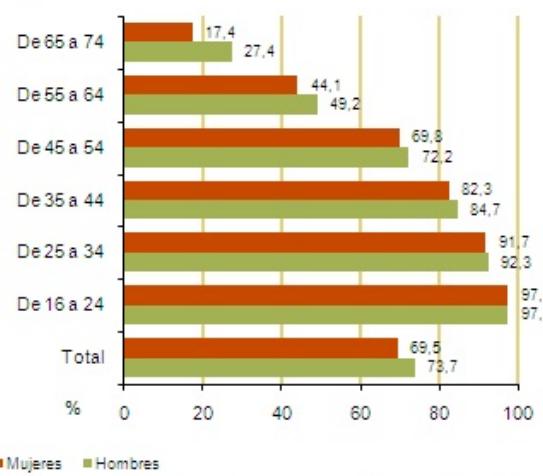
Por ello se hará uso de las últimas técnicas disponibles en el *front-end y back-end* de la aplicación para mejorar estos aspectos de la aplicación.

## 2.2. Estudio de la Situación Actual

Hoy en día las redes sociales copan un elevado tanto por ciento de toda la información que se comparte en Internet. **Una gran parte de la población dispone de una cuenta en una o varias redes sociales** en las que dispone de una amistad con gente con la que comparte información personal o publicaciones de otros medios de manera que se informe a todos aquellos usuarios que nos siguen.

En este caso la idea surge de un vacío que existe en las redes sociales deportivas en nuestro país y dotar a las ya existentes de una vertiente más social.

Para ello se ha optado por unir los puntos que se veían que funcionaban correctamente de cada una de las redes sociales que se han estudiado y ver que opciones prioritarias a añadir y como **facilitar a los usuarios compartir la información de las rutas que realizan**.



Fuente: Encuesta sobre Equipamiento y Uso de Tecnologías de la Información y Comunicación en los Hogares. INE

USO DE INTERNET DE LA POBLACIÓN ESPAÑOLA

En la actualidad **en España un 72% de la población dispone de Internet** aumentando este número paulatinamente e igualmente el número de ciclistas por nuestras ciudades no deja de aumentar siendo ya el uso en algunas ciudades cercano al 50% de la población.

Una de los principales handicaps de la aplicación es la necesidad de facilidad de uso para los usuarios ya que la gestión de la información visualizada correrá a cargo de los propios usuarios.

Debido al carácter de la aplicación **los propios usuarios serán los encargados de la compartir información** aunque existirá un usuario administrador que controlará todo el proceso de manera que exista un control sobre los elementos que se visualizarán en la aplicación.

## 2.3. Estudio de las Alternativas existentes

Desde un primer momento en el que surge la idea hasta su definición más concreta se ha mantenido contacto con gente del mundo del ciclismo en la que mostraron su interés e ideas de posibles funcionalidades a integrar.

De estas reuniones se *concluyeron los puntos básicos que tendría que cumplir la aplicación* para que fuese utilizada por cualquier tipo de persona y que permitiese a los usuarios compartir información de manera sencilla.

Para lograr estos requisitos se plantearon las siguientes mejoras que no cumplían otras alternativas ya existentes:

### Responsive design

*Diseño adaptable a todos los dispositivos donde se visualiza la aplicación web.* Éste es un punto crucial de la aplicación ya que actualmente en España hay más dispositivos móviles que habitantes y es un punto de acceso cada vez más común gracias a la mejora de las redes móviles y de los propios dispositivos que permiten una mejor visualización de las páginas web que se visitan a diario.



RESPONSIVE DESIGN

## **Red social**

Casi en su totalidad las alternativas que ya existen no disponen de un *componente social que integre a los usuarios en la aplicación*, suelen ser páginas web asociadas a una serie de rutas o de eventos sin fomentar la participación y relación entre los usuarios participantes.

## **Integración de rutas**

La integración de rutas se realiza única y exclusivamente a través de ficheros GPX o de manera manual. Nuestra aplicación hará uso de estas dos opciones que son las que más comúnmente conocen los usuarios añadiendo además la *posibilidad de integrar la información de alguna URL ya existente en Internet*.

## 2.4. Definición de Requisitos del Sistema

### 2.4.1. Requisitos Funcionales

En este apartado especificamos los requisitos funcionales del proyecto. Para ello deberemos diferenciar los requisitos propios de la aplicación y no sobre la plataforma que realizamos el desarrollo de la aplicación.

#### Plataforma

La plataforma deberá cumplir ciertos requisitos necesarios para que se comporte adecuadamente y pueda tener las características deseadas.

Identificador	Prioridad	Descripción
RF 1.1	Alta	<i>Adaptación a estándares</i>  Con el fin de asegurar una independencia del navegador y / o software de acceso a la plataforma.
RF 1.2	Alta	<i>Accesibilidad a los recursos</i>  Capacidad de mezclar recursos de una forma sencilla, satisfaciendo las necesidades del usuario.
RF 1.3	Alta	<i>Sensible a contexto</i>  El contexto deberá reunir cualquier información generada durante la interacción con el usuario, generando un valor añadido en sucesivas interacciones.
RF 1.4	Alta	<i>Participación activa del usuario</i>  Dotar a los usuarios de la capacidad de compartir su información y recursos con otros.
RF 1.5	Alta	<i>Tiempo real</i>  Resulta determinante el sector y el entorno en el que funcionará la aplicación. Se exige la máxima fluidez en los procesos y en el acceso a los datos compartidos.
RF 1.6	Alta	<i>Modularidad</i>  Su configuración modular permitirá la adaptación de la solución a distintos entornos.
RF 1.7	Alta	<i>Usabilidad</i>  El usuario final deberá manejar la aplicación tanto con dispositivos hardware como puedan ser el teclado y ratón o desde dispositivos táctiles tales como los smartphones y tablets.
RF 1.8	Alta	<i>Control del terminal</i>  La aplicación tendrá la capacidad para ocultar total o parcialmente las funciones propias del SO del terminal donde se ejecute.

<b>Identificador</b>	<b>Prioridad</b>	<b>Descripción</b>
RF 1.9	Alta	<p><i>Seguridad</i></p> <p>Control de versiones, control de accesos, validación de usuarios y demás métodos necesarios para un entorno seguro de información.</p>
RF 1.10	Alta	<p><i>Actualizaciones</i></p> <p>La aplicación se actualizará de manera inherente al usuario siendo éste inconsciente de cuando se produzcan dichas actualizaciones.</p>
RF 1.11	Alta	<p><i>Trazabilidad</i></p> <p>Generación de trazas a nivel de aplicación para facilitar el desarrollo y depuración. Generación de trazas de control / seguimiento de operaciones de cada usuario.</p>
RF 1.12	Muy alta	<p><i>Seguridad de datos</i></p> <p>La información contenida y generada en el terminal es de una gran importancia tanto empresarial como fiscal. La gestión de réplicas de la base de datos y las copias de seguridad deben ser consideradas como acciones fundamentales de la aplicación.</p>
RF 1.13	Muy alta	<p><i>Protección de datos</i></p> <p>Tanto por su importancia empresarial como por la LOPD es necesario que los intercambios de información se realicen de forma encriptada.</p>

## Aplicación

A continuación se presentan los requisitos de la aplicación. Para facilitar la comprensión se han dividido en varios subsistemas:

- Subsistema de Gestión de la Aplicación
- Subsistema de Gestión de Usuarios
- Subsistema de Gestión de Comentarios
- Subsistema de Gestión de Rutas
- Subsistema de Gestión de Eventos
- Subsistema de Gestión de Conversaciones
- Subsistema de Consulta de Contacto

Dentro de cada uno de los subsistemas se especifican los requisitos funcionales de los mismos.

## Gestión de la Aplicación

Identificador	Prioridad	Descripción
RF 2.1	Alta	<p><i>Baja y Modificación de Usuarios</i></p> <p>El usuario administrador es el encargado de hacer una gestión de los usuarios que acceden a la aplicación. Para ello dispone de un panel de administración desde el cual realizar las operaciones de modificación y baja de usuarios de la plataforma.</p>
RF 2.2	Alta	<p><i>Alta, Baja y Modificación de Rutas</i></p> <p>El usuario administrador dispone de un panel de control de las rutas creadas en el sistema por usuarios registrados pudiendo modificar o eliminar las ya existentes así como crear nuevas rutas en la aplicación.</p>
RF 2.3	Alta	<p><i>Alta, Baja y Modificación de Eventos</i></p> <p>El usuario administrador dispone de un panel de control de los eventos creados en el sistema por usuarios registrados pudiendo modificar o eliminar los eventos ya existentes en el sistema así como crear nuevos eventos que todavía no estuviesen en el sistema.</p>
RF 2.4	Alta	<p><i>Borrado de Conversaciones</i></p> <p>El usuario administrador dispone de un panel de control de las conversaciones existentes entre los distintos usuarios pudiendo de dicha manera hacer un control de las mismas por si hubiese algún problema entre usuarios. Se le permite realizar el borrado de las mismas sin consentimiento de parte de los usuarios.</p>
RF 2.5	Media	<p><i>Informe de Usuarios</i></p> <p>El usuario administrador dispone la posibilidad de realizar un informe en formato Excel o CSV de los usuarios registrados en el sistema a partir de unos filtros realizados previamente por el mismo usuario administrador desde su panel de administración.</p>
RF 2.6	Media	<p><i>Informe de Rutas</i></p> <p>El usuario administrador dispone la posibilidad de realizar un informe en formato Excel o CSV de las rutas almacenadas en el sistema a partir de unos filtros realizados previamente por el mismo usuario administrador desde su panel de administración.</p>
RF 2.7	Media	<p><i>Informe de Eventos</i></p> <p>El usuario administrador dispone la posibilidad de realizar un informe en formato Excel o CSV de los eventos registrados en el sistema a partir de unos filtros realizados previamente por el mismo usuario administrador desde su panel de administración.</p>

## Gestión de Usuarios

Identificador	Prioridad	Descripción
RF 3.1	Alta	<p><i>Formulario de Alta de Usuario</i></p> <p>La aplicación deberá permitir la creación de nuevos usuarios en la plataforma en cualquier momento.</p> <p>El alta de un nuevo usuario requiere la siguiente información:</p> <ul style="list-style-type: none"> <li>• Nombre</li> <li>• Email</li> <li>• Contraseña</li> <li>• Confirmación de contraseña</li> </ul> <p>Y con las siguientes características y limitaciones impuestas en el sistema:</p> <ul style="list-style-type: none"> <li>• Email con formato de email (<u>xxx@xxx.xx</u>)</li> <li>• Email debe ser único en el sistema</li> <li>• Contraseña y confirmación de contraseña deben ser el mismo texto y deben tener al menos 6 caracteres</li> </ul>
RF 3.2	Alta	<p><i>Formulario de Recordar Contraseña</i></p> <p>La aplicación deberá permitir a un usuario recordar su contraseña para volver a acceder a la plataforma en caso de que no la recordase.</p> <p>La solicitud de una nueva contraseña requiere de la siguiente información:</p> <ul style="list-style-type: none"> <li>• Email</li> </ul>
RF 3.3	Alta	<p><i>Formulario de Acceso a la Plataforma</i></p> <p>La aplicación deberá permitir el acceso a los usuarios registrados previamente en la plataforma.</p> <p>El acceso a la misma requiere de la siguiente información:</p> <ul style="list-style-type: none"> <li>• Email</li> <li>• Contraseña</li> </ul>
RF 3.4	Alta	<p><i>Modificación de contraseña tras solicitud de recordatorio</i></p> <p>La aplicación permitirá a un usuario modificar su contraseña si existía previamente en el sistema y ha recibido el enlace de seguridad que permite hacer una modificación de contraseña.</p> <p>Los datos requeridos para el cambio de contraseña son:</p> <ul style="list-style-type: none"> <li>• Contraseña</li> <li>• Confirmación de contraseña</li> </ul>

Identificador	Prioridad	Descripción
RF 3.5	Alta	<p><i>Edición de Datos del Usuario</i></p> <p>Un usuario registrado podrá realizar modificaciones en sus datos de usuario de manera que los otros usuarios que accedan a su perfil vean información actualizada.</p> <p>También dispondrá de la posibilidad de realizar una modificación de su nombre, email y contraseña de acceso al sistema.</p>
RF 3.6	Alta	<p><i>Borrado de Usuario</i></p> <p>Un usuario puede eliminar su cuenta del sistema de manera que se eliminen de la misma todos los comentarios y conversaciones que tuviese en la misma quedando las rutas y eventos asociadas a la administración del sistema.</p>

## Gestión de Comentarios

Identificador	Prioridad	Descripción
RF 4.1	Alta	<p><i>Formulario de Alta de Comentario</i></p> <p>Un usuario registrado puede realizar un comentario en su muro o en el de otro usuario rellenando la siguiente información:</p> <ul style="list-style-type: none"> <li>• Comentario</li> </ul> <p>Y cumpliendo las limitaciones impuestas de 255 caracteres en el comentario.</p>
RF 4.2	Alta	<p><i>Borrado de comentarios</i></p> <p>Un usuario registrado puede borrar un comentario que haya escrito previamente en su muro, en una ruta o en un evento.</p>
RF 4.3	Media	<p><i>Marcar comentario como ofensivo</i></p> <p>Un usuario registrado puede marcar un comentario de otro usuario registrado como ofensivo, esto provocará una alerta en la administración de manera que dicho comentario quedará a valoración de los usuarios administradores.</p>

## Gestión de Rutas

Identificador	Prioridad	Descripción
RF 5.1	Alta	<p><i>Formulario de Alta de Ruta importación externa</i></p> <p>Un usuario registrado puede añadir una nueva ruta al sistema a partir de una URL de otro portal en la que ya disponga de rutas almacenadas.</p> <p>El campo obligatorio y requerido es una URL bien formada y que pertenezca a alguno de los portales de los que se puede leer información.</p>

<b>Identificador</b>	<b>Prioridad</b>	<b>Descripción</b>
RF 5.2	Alta	<p><i>Formulario de Alta de Ruta mediante fichero GPX</i></p> <p>Un usuario registrado puede añadir una nueva ruta al sistema a partir de un fichero GPX que almacena los puntos por los que transcurrirá dicha ruta.</p> <p>El alta de una nueva ruta mediante un fichero GPX necesita la siguiente información siendo todos los campos obligatorios:</p> <ul style="list-style-type: none"> <li>• Nombre</li> <li>• Descripción</li> <li>• Fichero GPX</li> <li>• Tipo de ruta</li> <li>• Dificultad</li> <li>• Permitir comentarios</li> <li>• Visibilidad</li> </ul>
RF 5.3	Alta	<p><i>Formulario de Alta de Ruta manualmente</i></p> <p>Un usuario registrado puede añadir una nueva ruta al sistema de manera manual situando puntos en un mapa y almacenando la información de cada uno de ellos.</p> <p>El alta de una nueva ruta de manera manual necesita la siguiente información siendo todos los campos obligatorios:</p> <ul style="list-style-type: none"> <li>• Nombre</li> <li>• Descripción</li> <li>• Puntos sobre un mapa (al menos dos puntos)</li> <li>• Tipo de ruta</li> <li>• Dificultad</li> <li>• Permitir comentarios</li> <li>• Visibilidad</li> </ul>
RF 5.4	Alta	<p><i>Formulario de Edición de Ruta</i></p> <p>Un usuario registrado puede editar una de sus rutas previamente almacenadas en el sistema.</p> <p>Son obligatorios los mismos campos que para los formularios de Alta de Ruta de manera manual o mediante fichero GPX.</p>
RF 5.5	Alta	<p><i>Borrado de Ruta</i></p> <p>Un usuario registrado puede eliminar una ruta que previamente hubiese almacenado en el sistema.</p>
RF 5.6	Alta	<p><i>Búsqueda de rutas</i></p> <p>Un usuario que acceda a la aplicación puede realizar una búsqueda de rutas a partir de una localización introducida, mediante el desplazamiento por el mapa y mediante los filtros de Tipo de ruta y dificultad de la misma.</p>
RF 5.7	Media	<p><i>Marcar Ruta como favorita</i></p> <p>Un usuario registrado puede marcar una ruta como favorita de manera que a posteriori aparezca dentro de su listado de rutas favoritas.</p>

## Gestión de Eventos

Identificador	Prioridad	Descripción
RF 6.1	Alta	<p><i>Formulario de Alta y Edición de Evento</i></p> <p>Un usuario registrado puede dar de alta un nuevo evento en el sistema, para ello debe llenar la siguiente información siendo todos los campos obligatorios excepto el campo URL del evento:</p> <ul style="list-style-type: none"> <li>• Nombre del evento</li> <li>• Descripción</li> <li>• Fecha</li> <li>• URL del evento</li> <li>• Localización (sobre un mapa) y Dirección asociada</li> <li>• Tipo de evento</li> <li>• Permitir comentarios</li> <li>• Visibilidad</li> </ul>
RF 6.2	Alta	<p><i>Borrado de Evento</i></p> <p>Un usuario registrado que haya dado de alta previamente un evento puede eliminar el mismo del sistema.</p>
RF 6.3	Alta	<p><i>Búsqueda de Eventos</i></p> <p>Un usuario que acceda a la aplicación puede realizar una búsqueda de eventos a partir de una localización, mediante el desplazamiento por el mapa y mediante el filtro de Tipo de Evento.</p>
RF 6.4	Media	<p><i>Marcar un Evento como favorito</i></p> <p>Un usuario registrado puede marcar un evento como favorito de manera que a posteriori le aparezca dentro de su listado de eventos favoritos.</p>

## Gestión de Conversaciones

Identificador	Prioridad	Descripción
RF 7.1	Alta	<p><i>Iniciar Conversación con otro Usuario</i></p> <p>Un usuario registrado puede iniciar una conversación con otro usuario mediante el identificador RF 7.2 consistente en la escritura de un mensaje siempre que no hubiese mensajes previos entre ellos.</p>
RF 7.2	Alta	<p><i>Formulario de Mensaje entre usuarios</i></p> <p>Un usuario registrado puede escribir un mensaje a otro usuario registrado en el sistema mediante el formulario de Mensaje. Dicho formulario consta de un único campo obligatorio denominado Mensaje.</p> <p>Si el mensaje es el primero entre los usuarios previamente se crea una Conversación entre usuarios a la que irá asociada el mensaje escrito.</p>
RF 7.3	Alta	<p><i>Enviar Conversación a Archivo</i></p> <p>Un usuario registrado que tenga una conversación con otro usuario puede enviar dicha conversación a Archivo de manera que no le lleguen notificaciones de las mismas.</p>

## Gestión de Contacto

Identificador	Prioridad	Descripción
RF 8.1	Alta	<p>Formulario de Contacto</p> <p>Un usuario visitante o registrado puede hacer una solicitud de información a la administración del portal.</p> <p>Los campos del formulario solicitados siendo todos ellos obligatorios son:</p> <ul style="list-style-type: none"><li>• Nombre</li><li>• Email</li><li>• Tipo de necesidad informativa</li><li>• Comentario</li></ul>

## 2.4.2. Requisitos No Funcionales

<b>Identificador</b>	<b>Prioridad</b>	<b>Descripción</b>
RNF 1.1	Alta	<p>Control de Acceso</p> <p>Para controlar el acceso al sistema cada usuario dispone de un email único y una contraseña de acceso.</p>
RNF 1.2	Alta	<p>Permisos</p> <p>Los usuarios sólo podrán visualizar, editar y eliminar aquellos elementos sobre los que tienen permiso de actuación. El administrador del sistema podrá modificar o eliminar cualquier elemento de la aplicación.</p>
RNF 1.3	Alta	<p>Control de acceso a servidor</p> <p>Será necesaria una configuración robusta y segura para el acceso al servidor de producción de manera que sólo puedan acceder los usuarios registrados controlando el acceso mediante firewall con control de IP.</p>
RNF 1.4	Alta	<p>Diseño gráfico</p> <p>La apariencia es importante para un buen uso de la aplicación para ello es necesaria una buena Interfaz de Usuario de manera que sea sencillo adivinar el funcionamiento de todos los elementos así como que no se diferencie demasiado de las redes sociales a las que ya están habituados los usuarios.</p>
RNF 1.5	Alta	<p>Responsive design</p> <p>La multitud de dispositivos desde los que puede acceder un usuario implicarán un control de la visualización de la aplicación en la mayoría de las resoluciones disponibles.</p>
RNF 1.6	Alta	<p>Control de Introducción de Datos</p> <p>Se ha de controlar el tipo de datos introducidos de manera que cumplan las condiciones indicadas y las validaciones introducidas a nivel de base de datos.</p>
RNF 1.7	Media	<p>Agilidad de conexiones</p> <p>Se hará hincapié en las peticiones realizadas al servidor de manera que sean las mínimas posibles y haciendo el mínimo uso posible de la red.</p>
RNF 1.8	Media	<p>Tipos de datos</p> <p>La aplicación hará uso exhaustivo de los datos en formato JSON, para ello será vital un buen uso de la información que nos llegue en dicho formato y su posterior visualización en pantalla.</p>
RNF 1.9	Alta	<p>Seguridad de la información</p> <p>Las copias de respaldo serán incrementales diariamente y se hará una copia de respaldo completa semanalmente tanto de los ficheros como de la base de datos de la aplicación.</p>

## 2.5. Alternativas Server-Side

Una vez decididos los requisitos funcionales y no funcionales de los que hará uso nuestra aplicación se necesita decidir qué tecnologías se usarán durante el desarrollo. Como ya se ha indicado la aplicación será una página web por lo que su funcionamiento básico y su visualización en pantalla se realizarán mediante la combinación de **HTML5, CSS3 y Javascript**.



Pero toda esta capa de presentación o también conocida como *front-end* se apoya en otra capa de servidor o *back-end* que contiene la lógica y el acceso a datos.

Las plataformas escogidas tanto a nivel de gestor de base de datos como de lenguaje de programación será las que se han considerado como la mejor opción tanto a nivel de funcionamiento, escalabilidad, tiempo de respuesta, aprendizaje y legibilidad de código entre otros aspectos.

A continuación describimos algunos de los diferentes *gestores de bases de datos* y *frameworks web* existentes.

### 2.5.1. Elección del Gestor de Base de Datos

La primera elección que se ha realizado es la elección de un gestor de base de datos que proporcionase por encima de todo una **cantidad concurrente de usuarios lo más elevada posible y una alta tasa de velocidad de lectura y escritura**.

Contando con estas dos premisas damos por supuesto que el sistema elegido proporcionará los servicios necesarios para almacenar, modificar y eliminar toda la información que se manejará en la aplicación.

La elección de la gestión de la base de datos se ha centrado en las siguientes opciones:

#### Oracle

Desarrollador: *Oracle Corporation*

Licencia: *Uso comercial*



Sistema de gestión de base de datos de tipo objeto-relacional siendo el gestor de base de datos más usado a nivel mundial pudiendo ejecutarse en todas las plataformas.

Por encima de todo hay que destacar que entre sus principales características están el *soporte de transacciones, la estabilidad, escalabilidad y el soporte multiplataforma* ya que el software del servidor puede ejecutarse en multitud de sistemas operativos.

La principal desventaja que nos encontramos en Oracle es su precio ya que su licencia de uso tiene un coste muy elevado.

### **Microsoft Access**

Desarrollador: *Microsoft Corporation*

Licencia: *Uso comercial*



Es el sistema de bases de datos proporcionado por Microsoft dentro de su suite de ofimática Microsoft Office.

Es una base de datos con un *estilo muy amigable y muy versátil*. Tiene una escasa *complejidad de uso inicial* y su formación se puede realizar de manera on-line.

Es una opción que se desecha debido a su escasa potencia en cuanto a tiempo de respuesta y concurrencia de usuarios.

### **Microsoft SQL Server**

Desarrollador: *Microsoft Corporation*

Licencia: *Uso comercial*



Constituye la alternativa de Microsoft a otros potentes sistemas gestores de bases de datos de la competencia como pueden ser Oracle, PostgreSQL o MySQL.

Es un gestor de base de datos *mucho más potente que la suite Microsoft Access* pero su mayor desventaja consideramos que es la *necesidad de tener que tener la aplicación en un servidor con Windows* lo que implicaría un coste añadido de licencias en la aplicación.

### **MySQL**

Desarrollador: *Sun Microsystems*

Licencia: *GPL o Uso comercial*



Es un sistema de gestión de base de datos de *tipo relacional, multihilo y multiusuario* que no tiene ningún tipo de coste para el cliente.

En desarrollos en los que el *número de peticiones por segundo es muy elevado con gran cantidad de datos en el sistema tiene peor rendimiento que otros gestores de base de datos*.

## **PostgreSQL**

Desarrollador: *PostgreSQL Global Development Group*

Licencia: *BSD*



Se trata de un sistema de gestión de base de datos de tipo relacional orientado a objetos y de software libre publicado bajo la licencia *BSD* que permite su uso en desarrollos cerrados.

Presenta una gran escalabilidad con un funcionamiento eficaz. Como mayor desventaja consume una gran cantidad de recursos lo que lo convierte en una opción en la que deberíamos valorar el coste que implicaría su uso a nivel de servidor.

## **MongoDB**

Desarrollador: *MongoDB, Inc.*

Licencia: *GNU AGPL (Apache)*



Es una base de datos no relacional de tipo documental. Está creada por la empresa de su mismo nombre MongoDB y es de libre distribución.

Sus principales características son un alto rendimiento de lectura y escritura siendo en este punto mucho más rápida que cualquiera de las opciones planteadas y de muy sencilla escalabilidad y réplica en servidores.

### 2.5.2. Elección del Lenguaje de Programación

Nuestro siguiente paso será elegir el lenguaje de programación y framework con el que vamos a desarrollar la aplicación.

## **CakePHP**

Lenguaje: *PHP*

Licencia: *MIT License*



CakePHP es un *framework Open Source* escrito en PHP, modelado bajo los conceptos de Ruby on Rails, y distribuido bajo una Licencia MIT. Su desarrollo, que comenzó en 2005, ha ido con el tiempo creciendo, llegando incluso a contar con una propia forja de sub-proyectos basados en CakePHP.

Como Ruby on Rails, CakePHP hace más fácil para el usuario interactuar con la base de datos mediante su ORM, Active Records. Recomienda el uso del patrón arquitectónico MVC (modelo vista controlador), pero no obliga a su uso.

Entre sus características, destacan:

- Compatible con PHP4 y PHP5
- CRUD integrado para la base de datos y consultas simplificadas
- Request dispatcher con URLs personalizadas
- Plantillas de sintaxis PHP con métodos de ayuda (*helpers*) integrados.
- Facilidades para las vistas con AJAX, Javascript y formularios HTML
- Validación built-in
- Listas de control de acceso (ACL)
- Aplicación de scaffolding
- Componentes manejo de peticiones, sesiones y seguridad
- Aplicación para la gestión de servicios de afiliación y fuentes externas
- Estudio de Viabilidad del Sistema
- Caché para las vistas

La Fundación Mambo anunció en 2007 que utilizaría el framework CakePHP para las futuras versiones de su ampliamente usado CMS, denominando a CakePHP como una "elección sólida y ciertamente una de los mejores frameworks disponibles hoy día".

Existen algunas implementaciones de REST sobre CakePHP, pero no existe ninguna integración con el propio framework, por lo que no son soluciones completas y requerirían de un análisis más extenso.

## Django

Lenguaje: *Python*

Licencia: *BSD License*



Django es un framework de desarrollo Web de código abierto, escrito en Python, que cumple en cierta medida el paradigma del Modelo Vista Controlador (salvo la nomenclatura) bajo la filosofía DRY (don't repeat yourself).

Fue desarrollado en origen para gestionar varias páginas orientadas a noticias de la World Company de Lawrence, Kansas, y fue liberada al público bajo una licencia BSD en julio de 2005, tras largo tiempo funcionando en producción.

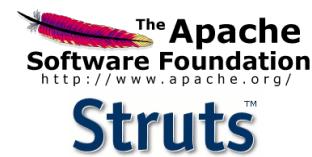
Entre sus características principales podemos enumerar soporte para múltiples bases de datos con un mapeador de tipo objeto relacional, diseños de urls elegantes, cache,

internacionalización, documentación incorporada accesible a través de la aplicación administrativa y una gran comunidad de desarrolladores.

## Struts

Lenguaje: Java

Licencia: Apache



Struts es una herramienta de soporte para el desarrollo de aplicaciones Web bajo la plataforma J2EE (Java 2, Enterprise Edition). Struts se desarrollaba como parte del proyecto Jakarta de la Apache Software Foundation, pero actualmente es un proyecto independiente conocido como Apache Struts.

Struts permite reducir el tiempo de desarrollo. Tiene carácter de "software libre" y compatibilidad con todas las plataformas en que Java Enterprise esté disponible.

Struts se basa en el patrón del Modelo-Vista-Controlador (MVC) el cual se utiliza ampliamente y es considerado de gran solidez.

## Ruby on Rails

Lenguaje: Ruby

Licencia: Licencia MIT



Es un lenguaje de programación dinámico y de código abierto enfocado en la simplicidad y productividad. Su lenguaje de programación Ruby es considerado un lenguaje flexible ya que permite a los usuarios programadores alterarlo libremente.

Sigue el paradigma de la arquitectura Modelo Vista Controlador (MVC). Combina la simplicidad con la posibilidad de desarrollar aplicaciones del mundo real escribiendo menos código que con otros frameworks y con un mínimo de configuración. El lenguaje de programación Ruby permite la metaprogramación, de la cual Rails hace uso, lo que resulta en una sintaxis que muchos de sus usuarios encuentran muy legible.

Rails utiliza un sistema de paquetería llamado RubyGems, sobre el que se distribuyen las librerías y aplicaciones Ruby.

Los principios fundamentales de Ruby on Rails son *DRY* (Don't repeat yourself, No te repitas) y *Convención sobre configuración*.

Entre sus características principales destacan las siguientes:

- Mapeador objeto-relacional
- Soporte de múltiples bases de datos teniendo acceso a PostgreSQL, MySQL o SQLite3 entre otras siendo esta última la opción elegida por defecto al crear un proyecto desde cero.

- Diseño de URLs elegantes pudiendo ser modificadas en cualquier momento sin tener que alterar código para que se ajusten todos los links de la plataforma.
- Sistema de plantillas pudiendo ser modificado libremente por el usuario programador y escoger cualquier otro tipo de plantilla.
- Sistema de cache de tipo completo, parcial, caché de modelo o de consulta.
- Internacionalización con el gestor I18n el cual nos permitirá tener la aplicación traducida en cualquier idioma elegido.
- Aplicaciones denominadas gemas que pueden instalarse en cualquier página gestionada con RubyOnRails, ya que una de las propuestas de RubyOnRails es la metaprogramación, programas que escriben programas.
- Una API de base de datos robusta.
- Un despachador de URLs basado en expresiones regulares.
- Un sistema "middleware" para desarrollar características adicionales que nos permite realizar modificaciones en las rutas antes de acceder a la aplicación, control de tokens de seguridad de formularios y cualquier otra posibilidad que se nos ocurra.

### 2.5.3. Valoración de las alternativas

Entre los gestores de bases de datos se ha optado entre *dos características principales*, por un lado *el coste que conllevaría el gestor elegido en la aplicación y la tasa de lectura / escritura junto al número de usuarios concurrentes que es capaz de gestionar la base de datos*.

Por tanto la tecnología proporcionada por Microsoft (Microsoft SQL Server y Access) y por Oracle han sido descartadas debido a los costes que conllevaría hacer uso de dichos gestores.

Entre las otras opciones valoradas tanto **PostgreSQL como MongoDB** son las **opciones mejor valoradas debido a la tasa de concurrencia de usuarios que pueden manejar**.

En cuanto a los frameworks valorados se hizo la elección de los mismos en base a cuatro lenguajes de programación muy distintos pero que a su vez tienen un gran peso en la comunidad.

*Java y PHP* son dos lenguajes de sobra conocidos que *tienen un funcionamiento muy robusto pero que por su tipología de lenguaje y de framework no favorecen modificaciones rápidas* en el sistema que tan habitualmente suceden en las aplicaciones web hoy en día surgiendo nuevas modificaciones casi a diario.

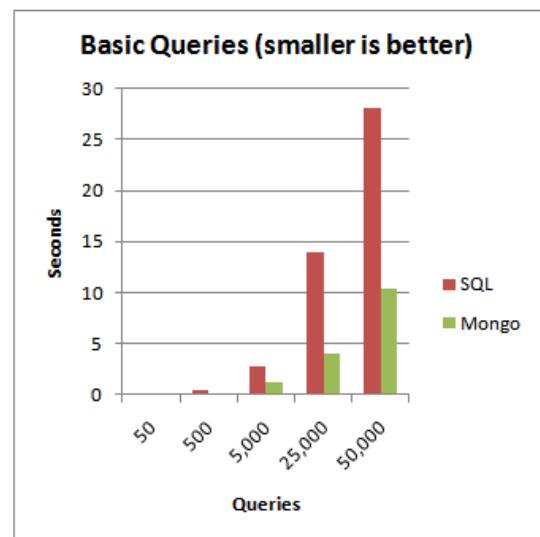
Sin embargo **Python** y **Ruby** son dos lenguajes relativamente “jóvenes” en una creciente evolución gracias a la **facilidad de desarrollo y potencia** que poseen (son necesarias muchas menos líneas de código que con PHP y Java).

Ambos son lenguajes similares, debido a la forma de programar y la manera de estructurar las aplicaciones aunque tienen matices que los diferencian entre sí.

#### 2.5.4. Descripción de las alternativas seleccionadas

Tras el estudio y la valoración de las alternativas disponibles para el desarrollo del proyecto, vamos a detallar la conclusión sobre la elección del sistema de gestión de base de datos y el lenguaje de programación.

Como sistema gestor de bases de datos **se ha optado por MongoDB** siendo la principal ventaja frente a PostgreSQL en que **su tasa de velocidad de lectura / escritura es superior** y que por su estructura interna de almacenamiento facilitará la labor en el front-end ya que MongoDB trabaja con un tipo de dato denominado BSON (similar al formato JSON) del que hará uso nuestra aplicación web ya que su funcionamiento será el de una API Web en formato REST mediante peticiones JSON.



Además MongoDB dispone de una serie de funciones por defecto que facilitan la labor del trabajo con elementos geolocalizables que serán de ayuda a la hora del desarrollo de la aplicación.

La elección entre los lenguajes de programación se centraba principalmente entre Ruby y Django.

Las opciones que hicieron decantar la balanza a favor de **Ruby** es que es un lenguaje de programación muy claro y conciso donde la convención prima sobre la configuración lo que implica escribir el menor número de líneas de código facilitando la migración a nuevas utilidades dentro de la aplicación.

Entre otras de las características que conllevaron la elección de *Ruby On Rails* es el propio lenguaje Ruby donde todos los tipos de datos son un objeto, incluidas las clases y tipos que otros lenguajes definen como primitivas. Toda función es un método. Las variables siempre son referencias a objetos, no los objetos mismos.

Esto facilita el uso de Ruby, porque las reglas que se aplican a los objetos son aplicables a todo Ruby pudiendo ser modificadas, sobreescritas en cualquier momento.

La filosofía de Ruby on Rails se puede resumir en tres principios fundamentales:

**DRY:** "No te repitas" (Don't Repeat Yourself) significa que las definiciones deberían hacerse una sola vez. Dado que Ruby on Rails es un framework, los componentes están integrados de manera que no hace falta establecer puentes entre ellos. Por ejemplo, en ActiveRecord, las definiciones de las clases no necesitan especificar los nombres de las columnas; Ruby puede averiguarlos a partir de la propia base de datos, de forma que definirlos tanto en el código como en el programa sería redundante.

**COC:** "Convención sobre configuración" (Convention Over Configuration) significa que el programador sólo necesita definir aquella configuración que no es convencional. Por ejemplo, si hay una clase Producto en el modelo, la tabla correspondiente de la base de datos es productos, pero si la tabla no sigue la convención debe ser especificada manualmente. Así, cuando se diseña una aplicación partiendo de cero sin una base de datos preexistente, el seguir las convenciones de Rails significa usar menos código.

**Agilidad:** este punto de la filosofía es heredado de los otros dos puntos. Si sabemos utilizar correctamente el DRY y el COC agilizaremos de forma notable el desarrollo de nuestros aplicativos, consiguiendo reducir al máximo nuestro tiempo de desarrollo.

Un punto muy importante dentro de la programación es el patrón Modelo Vista Controlador (MVC) donde se separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

- **Modelo:** representación específica de la información con la cual el sistema opera.
- **Vista:** responsable de presentar la interfaz y la información del controlador.
- **Controlador:** organiza la aplicación. Recibe eventos del exterior, interactúa con el modelo y actualiza la información de las vistas.

El patrón MVC se ve frecuentemente en aplicaciones Web, donde la vista es la página HTML, el control es el código que provee de datos dinámicos a la página, y el modelo contiene clases representativas de la aplicación.

### **Modelo:**

Ésta es la representación específica del dominio de la información sobre la cual funciona la aplicación. El modelo es otra forma de llamar a la capa de dominio. La lógica de dominio añade significado a los datos; por ejemplo, calculando si hoy es el cumpleaños

del usuario o los totales, impuestos o portes en un carrito de la compra.

En Rails, las clases del Modelo son gestionadas por el patrón ActiveRecord, en el cual, el objeto contiene los datos que representan a un registro de nuestra tabla o vista, además de encapsular la lógica necesaria para acceder a la base de datos. Por lo general, lo único que tiene que hacer el programador es heredar de la clase ActiveRecord::Base, y el programa averiguará automáticamente qué tabla usar y qué columnas tiene.

El modelo representa a las colecciones de la base de datos y para cada atributo hay un método asociado.

### **Vista:**

Presenta el modelo en un formato adecuado para interactuar, usualmente un elemento de interfaz de usuario.

En Rails la vista se genera usando RHTML (HTML con código Ruby) o RXML (XML con código Ruby). El controlador elige qué vista usar, y hace disponible los datos que necesita.

### **Controlador:**

Las clase del controlador responden a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista.

Aunque se pueden encontrar diferentes implementaciones de MVC, el flujo que sigue el control generalmente es el siguiente:

El usuario interactúa con la interfaz de usuario de alguna forma (por ejemplo, el usuario pulsa un botón)

El controlador recibe (por parte de los objetos de la interfaz-vista) la notificación de la acción solicitada por el usuario. El controlador gestiona el evento que llega, frecuentemente a través de un gestor de eventos (handler) o callback.

El controlador accede al modelo, actualizándolo y posiblemente modificándolo de forma adecuada a la acción solicitada por el usuario.

El controlador delega a los objetos de la vista la tarea de desplegar la interfaz de usuario. La vista obtiene sus datos del modelo para generar la interfaz apropiada para el usuario donde se refleja los cambios en el modelo. El modelo no debe tener conocimiento directo sobre la vista.

Ventajas de utilizar MVC:

- Código limpio.
- DRY.
- Facilita el trabajo en equipo.

Rails se distribuye a través de RubyGems, que es el formato oficial de paquetes y el canal de distribución de librerías y aplicaciones Ruby.

Rubygems provee de una aplicación llamada gem que permite instalar, desinstalar y consultar sobre las librerías o gemas que tengamos instalada en la computadora para implementar en el desarrollo.

- Fácil de instalar y desinstalar los paquetes de RubyGems, con la posibilidad de resolver las dependencias.
- Gestión y Control local de paquetes.
- Consulta, búsqueda tanto local como remotamente todos los paquetes.
- Múltiples versiones de apoyos para los paquetes instalados.
- Interfaz basada en web para ver las gemas o paquetes instalados.
- Interfaz fácil de usar para la construcción de las gemas.



### 3. Análisis del Sistema de Información

---

#### 3.1. Descripción del Sistema

##### 3.1.1. Determinación del Alcance del Sistema

Se desarrollará una aplicación mediante la cual aficionados al ciclismo puedan compartir sus aficiones, rutas y eventos en los que participen.

Las principales características serán:

**Acceso sencillo** en cualquier momento ya que la plataforma será desarrollada en base a **Responsive Web Design**, de manera que su visualización en cualquier dispositivo sea correcta y acorde a la pantalla en la que se visualice.

**Sincronización en tiempo real**, de manera que un usuario registrado siempre tenga disponible la última información disponible en el sistema.

**Creación de rutas** mediante tres opciones disponibles. Una de ellas mediante la **importación de rutas desde otra página web**, una segunda opción mediante un **fichero GPX** y por último de **manera manual** situando puntos en el mapa.

**Relaciones entre usuarios** permitiendo la *amistad* entre usuarios, conversaciones y comentarios en los muros de usuario.

La **edición de datos del perfil** de los usuarios y un **control sobre la información** que se hace visible al resto de usuarios de la plataforma.

Todos estos puntos se unen para facilitar la interacción entre los usuarios de la plataforma ofreciendo una red social con información actualizada y de fácil manejo.

La aplicación se realizará sobre una plataforma Web desarrollada mediante herramientas que faciliten la gestión de todos los elementos que componen el sistema permitiendo la categorización tanto de tipo como dificultad de rutas y eventos creados por los propios usuarios, la disposición de los mismos en mapas que permitan a un usuario saber con exactitud la localización de los elementos disponibles así como la interacción social entre usuarios mediante comentarios y conversaciones.

### 3.1.2. Identificación del Entorno Tecnológico

Como se ha decidido en el análisis previo del apartado anterior la aplicación se desarrollará una *plataforma web* bajo el *framework Ruby on Rails* basado en el lenguaje *Ruby* y con una base de datos de tipo *NoSQL* (*MongoDB*).

Estas dos opciones nos proporcionarán flexibilidad en la integración del proyecto, código mantenible con posibilidades de actualizaciones futuras sin que repercuta en un coste elevado y una alta tasa de peticiones por segundo a la plataforma sin sobrecarga.

### 3.1.3. Identificación de los Usuarios Participantes

Desde el punto de vista de los usuarios finales existirán tres tipos de usuarios:

- **Administrador del Sistema:** Encargado de la gestión de la aplicación.
- **Usuario registrado:** Gestiona su información visible en la aplicación.
- **Usuario anónimo:** Sólo puede consumir cierta información de la aplicación.

## 3.2. Especificación del Sistema

### 3.2.1. Contexto del Sistema

Descripción de los actores

Los actores que deben interactuar con el sistema y que lo usan de alguna forma son:

**Administrador del Sistema:** Encargado de gestionar el correcto funcionamiento de la aplicación. Su principal cometido será el de mantener una red social ejemplar y que impida problemas entre usuarios.

**Usuario Registrado:** Son usuarios que se han registrado en la plataforma y que por tanto le permite la creación de comentarios, conversaciones, rutas y eventos que serán visualizados por los usuarios de la plataforma.

**Usuario No Registrado:** Usuarios que acceden al sistema para visualizar los eventos y rutas que se hayan compartido de alguna manera.

### 3.3. Diagrama de Subsistemas

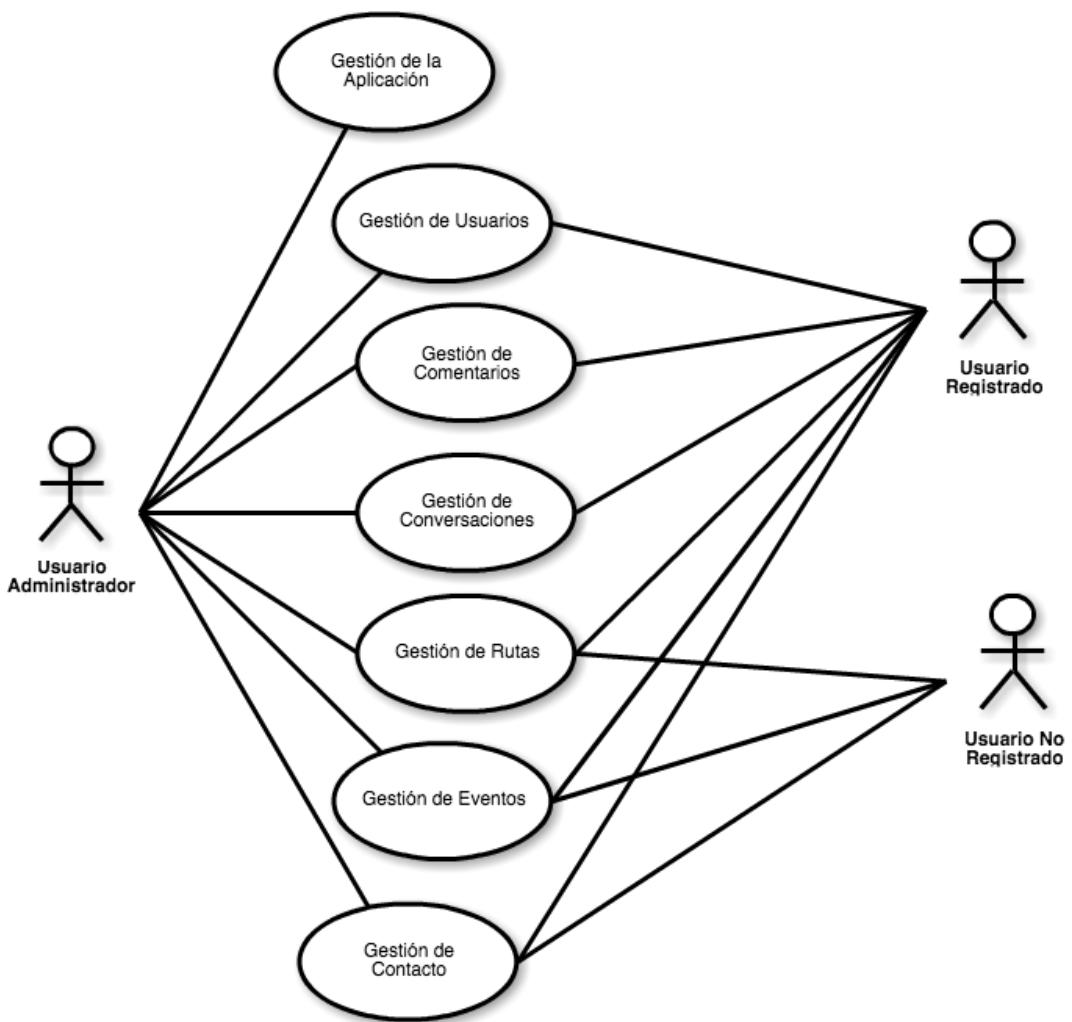


DIAGRAMA DE SUBSISTEMAS

### Gestión de la aplicación

Subsistema encargado de la gestión de todos los elementos que interviene en la aplicación. El administrador podrá controlar la gestión de usuarios, comentarios, conversaciones, rutas y eventos del sistema.

### Gestión de Usuarios

Subsistema encargado de la gestión de Usuarios del sistema. Es uno de los subsistemas fundamentales ya que sólo los usuarios registrados y el usuario administrador pueden crear información que visualicen otros usuarios visitantes.

### **Gestión de Comentarios**

Subsistema encargado de la gestión de comentarios en la plataforma. Sólo puede ser ejecutado por usuarios registrados en la plataforma y en la actualidad se integra dentro del muro de los usuarios, rutas y eventos con posibilidad de en un futuro integrarlo en nuevos módulos a desarrollar.

### **Gestión de Conversaciones**

Subsistema encargado de la gestión de conversaciones de la plataforma. Una conversación necesita al menos dos participantes registrados.

### **Gestión de Rutas**

Subsistema que se encarga de la gestión de las rutas almacenadas en el sistema. Permitirá al usuario registrado la creación, edición y borrado de las rutas que administre.

### **Gestión de Eventos**

Subsistema que se encarga de la gestión de las eventos almacenados en el sistema. Permitirá al usuario registrado la creación, edición y borrado de las rutas que administre.

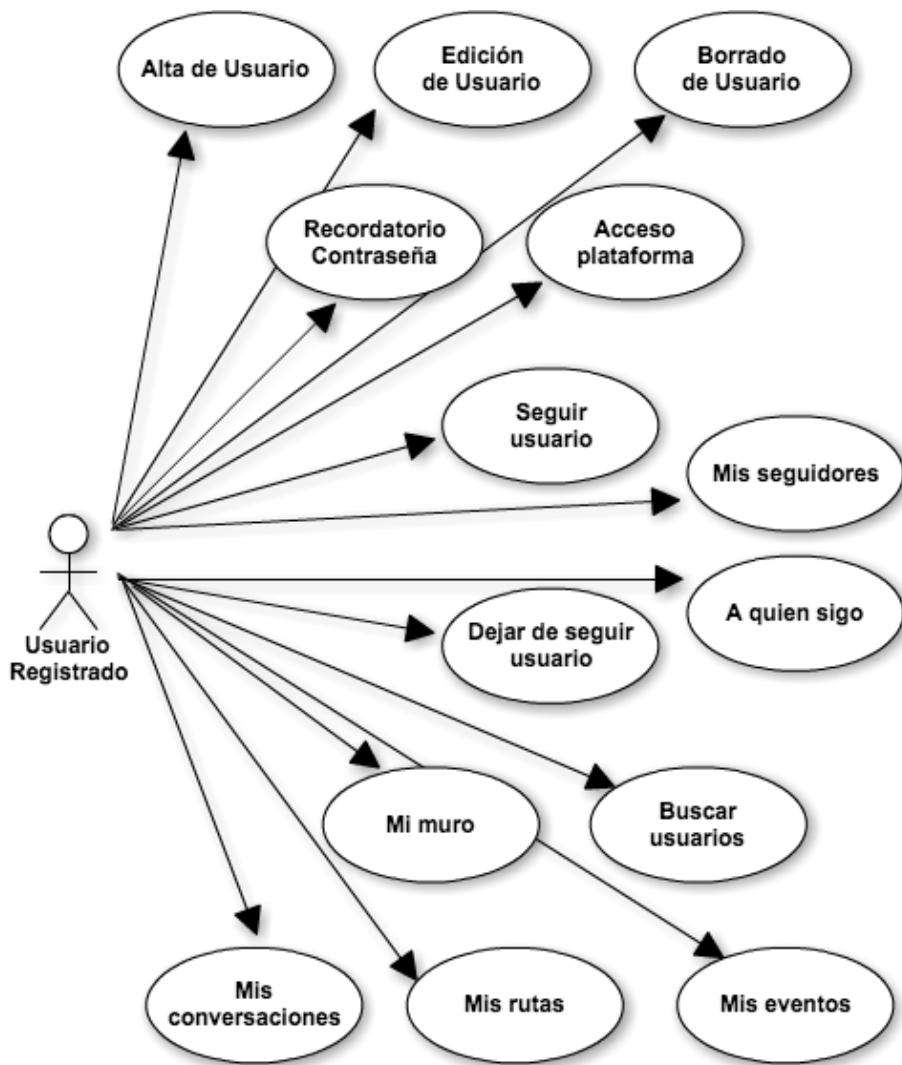
### **Gestión de Contacto**

Subsistema encargado del envío de la información solicitada por un usuario visitante de la web al usuario administrador del Sistema para que le informen de la solicitud realizada.

## 3.4. Especificación de los Subsistemas

### 3.4.1. Subsistema Gestión de Usuarios

#### Modelo de Casos de Uso



SUBSISTEMA GESTIÓN DE USUARIOS

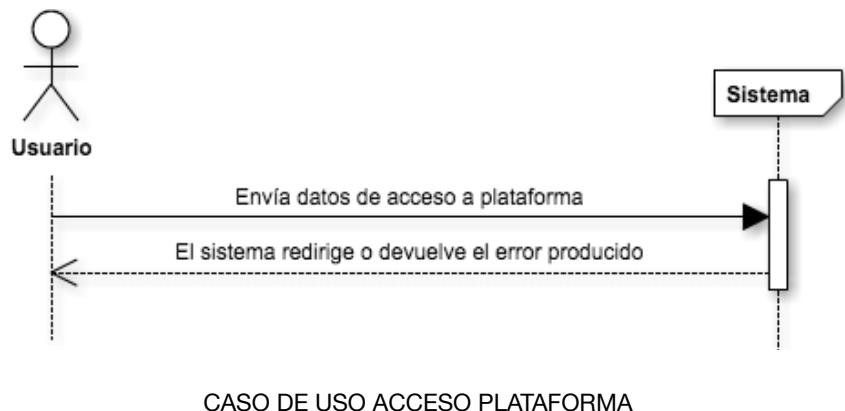
## Descripción de Casos de Uso

### Caso de Uso Acceso Plataforma

#### Descripción

Este caso de uso se encarga del control de acceso de los usuarios a la plataforma. Se solicita mediante un formulario los datos de acceso que corroboran que el usuario es un usuario registrado en la plataforma.

#### Diagrama de interacción



CASO DE USO ACCESO PLATAFORMA

#### Escenario: Acceso plataforma

##### Numeración: 1.1

**Precondiciones:** El usuario no dispone de la cookie que controla el acceso a la plataforma.

**Postcondiciones:** Si el usuario existe en el sistema será redirigido a su muro si se han introducido los datos correctos. Si son datos erróneos será redirigido a la pantalla de acceso a la plataforma para que vuelva a intentar acceder como usuario registrado.

**Quién lo comienza:** Usuario invitado

**Quién lo finaliza:** Usuario registrado

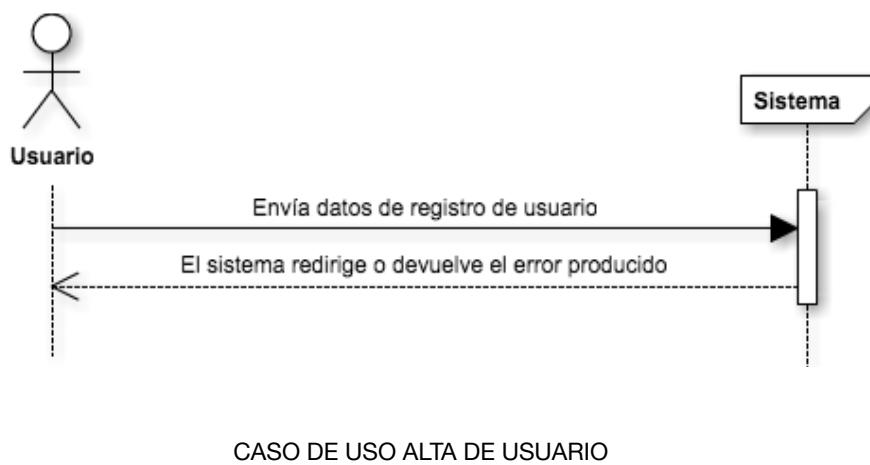
**Descripción:** Pantalla de inicio de sesión en la que se solicita un correo electrónico y una contraseña para acceder al sistema. Si el usuario introduce datos incorrectos se devuelve a la pantalla de acceso a la plataforma con un mensaje de error indicándole que no existen dichos datos. Si el usuario introduce datos correctos se envía al muro del usuario para que pueda ver los últimos acontecimientos que han sucedido en el sistema.

## Caso de Uso Alta de Usuario

### Descripción

Caso de uso encargado del registro de usuarios en la plataforma. Se solicita a través de un formulario la información mínima requerida para la creación de un usuario y se le notifica al usuario mediante un error en pantalla o la redirección al interior de la aplicación.

### Diagrama de interacción



CASO DE USO ALTA DE USUARIO

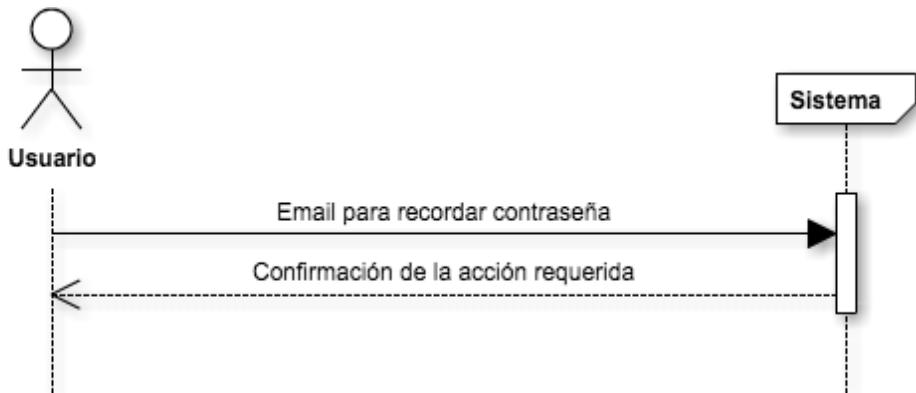
Escenario: Alta de usuario	
<b>Numeración:</b>	1.2
<b>Precondiciones:</b>	El usuario no dispone de la cookie que controla el acceso al sistema.
<b>Postcondiciones:</b>	El usuario accede a la plataforma ya como usuario registrado y se le envía un correo de agradecimiento por el registro.
<b>Quién lo comienza:</b>	Usuario visitante que se registra
<b>Quién lo finaliza:</b>	Usuario visitante que se registra
<b>Descripción:</b>	El usuario visitante que quiere registrarse puede llenar la información obligatoria solicitada en el formulario de registro. El sistema comprueba que los datos sean correctos, ya sea por petición de datos obligatorios como por la existencia de emails no repetidos. Si todo es correcto los datos del usuario se almacenan en el sistema y se envía un correo electrónico al email suministrado para que confirme su cuenta de usuario.

## Caso de Uso Recordar Contraseña

### Descripción

Este caso de uso se encarga de enviar al usuario solicitante un correo electrónico con un enlace único para que recupere su contraseña durante un tiempo máximo. Dicho enlace le servirá para realizar la modificación de su contraseña si fuese necesario.

### Diagrama de interacción



CASO DE USO RECORDAR CONTRASEÑA

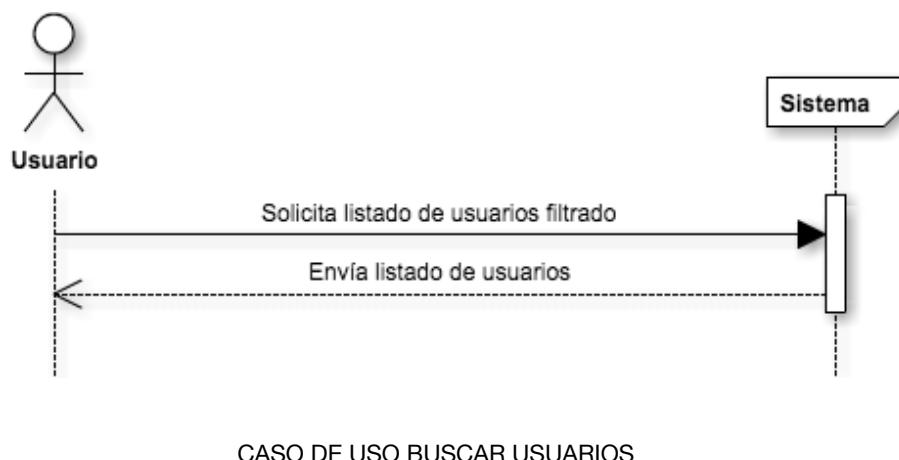
Escenario: Recordar contraseña	
<b>Numeración:</b>	1.3
<b>Precondiciones:</b>	El usuario no dispone de la cookie de control de haber accedido a la plataforma.
<b>Postcondiciones:</b>	Si introduce un correo electrónico almacenado en el sistema se enviará un email con una url de acceso para modificar la contraseña del usuario que tiene por correo electrónico introducido en el formulario.
<b>Quién lo comienza:</b>	Usuario visitante que no recuerda su contraseña
<b>Quién lo finaliza:</b>	Usuario visitante que no recuerda su contraseña
<b>Descripción:</b>	Un usuario visitante tiene la posibilidad de ingresar un email para que el sistema le remita un correo electrónico con una url en la que modificar su contraseña actual.

## Caso de Uso Buscar Usuarios

### Descripción

Este caso de uso se encarga de devolver el listado de usuarios que cumplan las condiciones de búsqueda realizada.

### Diagrama de interacción



CASO DE USO BUSCAR USUARIOS

#### Escenario: Buscar usuarios

**Numeración:** 1.4

**Precondiciones:** Usuario registrado.

**Postcondiciones:** Listado de usuarios a partir de la búsqueda realizada.

**Quién lo comienza:** Usuario registrado

**Quién lo finaliza:** Usuario registrado

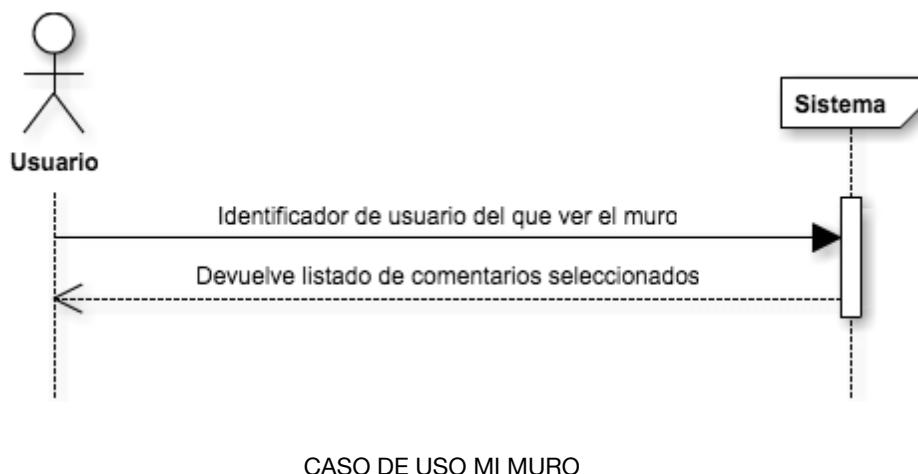
**Descripción:** Se realiza una búsqueda de usuarios a partir de un texto introducido. Dicha información se buscará a partir del nombre de usuario y devolverá un listado de los usuarios que cumplan con dicha condición.

## Caso de Uso Mi Muro

### Descripción

Este caso de uso permite a un usuario ver el muro de otro usuario o el suyo propio y los comentarios asociados al mismo.

### Diagrama de interacción



#### Escenario: Mi muro

##### Numeración: 1.5

**Precondiciones:** Usuario registrado tiene permiso para ver el contenido del muro del usuario seleccionado.

**Postcondiciones:** Usuario registrado dispone un listado de los comentarios asociados al muro del usuario visualizado.

**Quién lo comienza:** Usuario registrado

**Quien lo finaliza:** Usuario registrado

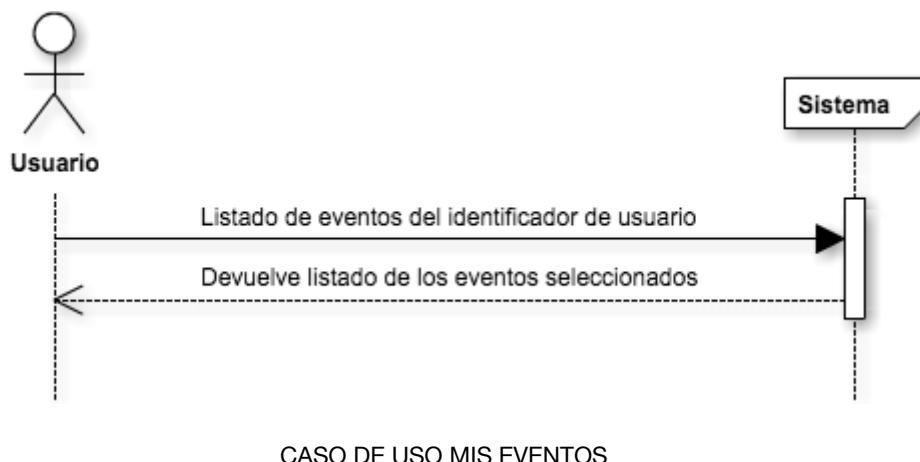
**Descripción:** Un usuario registrado puede ver aquellas notificaciones del muro de un segundo usuario sobre las que tenga permisos y sean visibles para dicho usuario o su propio muro con las notificaciones correspondientes de los usuarios a los que sigue.

## Caso de Uso Mis Eventos

### Descripción

Este caso de uso devuelve el listado de eventos que haya compartido el usuario visualizado.

### Diagrama de interacción



CASO DE USO MIS EVENTOS

#### Escenario: Mis eventos

**Numeración:** 1.6

**Precondiciones:** Usuario registrado

**Postcondiciones:** Lista de eventos del usuario visualizado.

**Quién lo comienza:** Usuario registrado

**Quién lo finaliza:** Usuario registrado

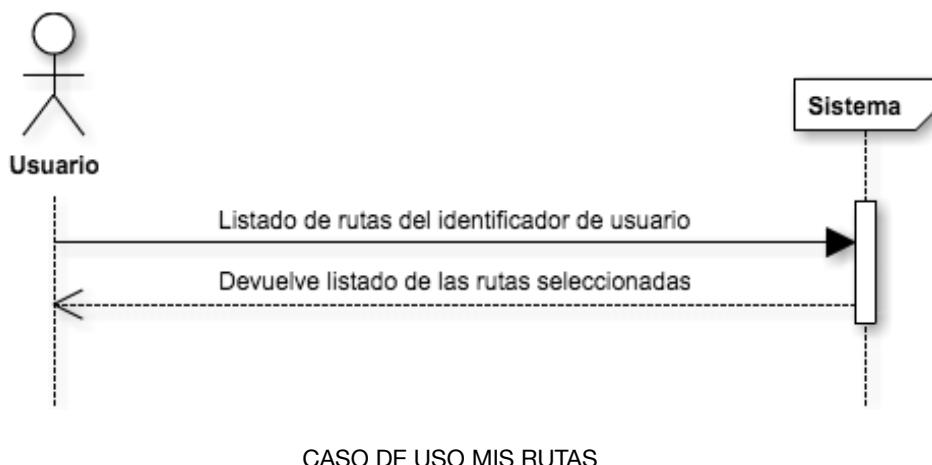
**Descripción:** Un usuario registrado tiene posibilidad de ver en una pantalla el listado de todos los eventos que hubiese creado en el portal web permitiéndole de un vistazo rápido el editar o eliminar la información que viese necesaria o ver un listado de los eventos que haya compartido otro usuario del portal sin posibilidad de edición o borrado, sólo visualización.

## Caso de Uso Mis Rutas

### Descripción

Este caso de uso devuelve el listado de rutas que haya compartido el usuario visualizado, en este caso se muestran todas las rutas las tenga publicadas o no.

### Diagrama de interacción



### Escenario: Mis rutas

**Numeración:** 1.7

**Precondiciones:** Usuario registrado

**Postcondiciones:** Lista de rutas del usuario visualizado.

**Quién lo comienza:** Usuario registrado

**Quien lo finaliza:** Usuario registrado

**Descripción:** Un usuario registrado tiene posibilidad de ver en una pantalla el listado de todas las rutas que hubiese creado en el portal web permitiéndole de un vistazo rápido el editar o eliminar la información que viese necesaria o ver un listado de las rutas que haya compartido otro usuario del portal sin posibilidad de edición o borrado, sólo visualización.

## Caso de Uso Mis Conversaciones

### Descripción

Este caso de uso devuelve el listado de conversaciones en las que está incluido el usuario que ha accedido a la plataforma.

### Diagrama de interacción



CASO DE USO MIS CONVERSACIONES

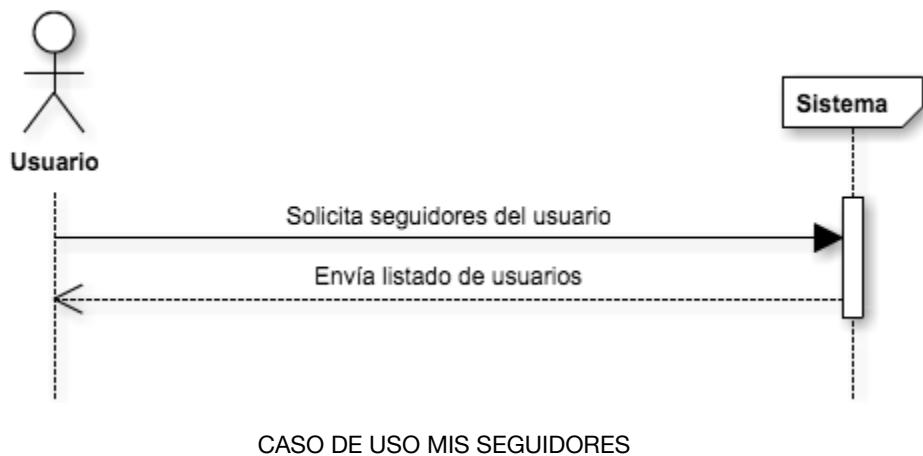
Escenario: Mis conversaciones	
<b>Numeración:</b>	1.8
<b>Precondiciones:</b>	Usuario registrado
<b>Postcondiciones:</b>	Listado de las conversaciones en las que participa el usuario registrado
<b>Quién lo comienza:</b>	Usuario registrado
<b>Quien lo finaliza:</b>	Usuario registrado
<b>Descripción:</b>	Un usuario registrado ve un listado con todos los mensajes que tiene con otros usuarios de la plataforma de manera que pueda acceder al detalle de los mismos.

## Caso de Uso Mis Seguidores

### Descripción

Este caso de uso devuelve un listado de todos los usuarios que siguen al usuario que accede a la plataforma.

### Diagrama de interacción



### Escenario: Mis seguidores

**Numeración:** 1.9

**Precondiciones:** Usuario registrado

**Postcondiciones:** Usuario registrado ve un listado de todos los usuarios que le siguen en la plataforma

**Quién lo comienza:** Usuario registrado

**Quién lo finaliza:** Usuario registrado

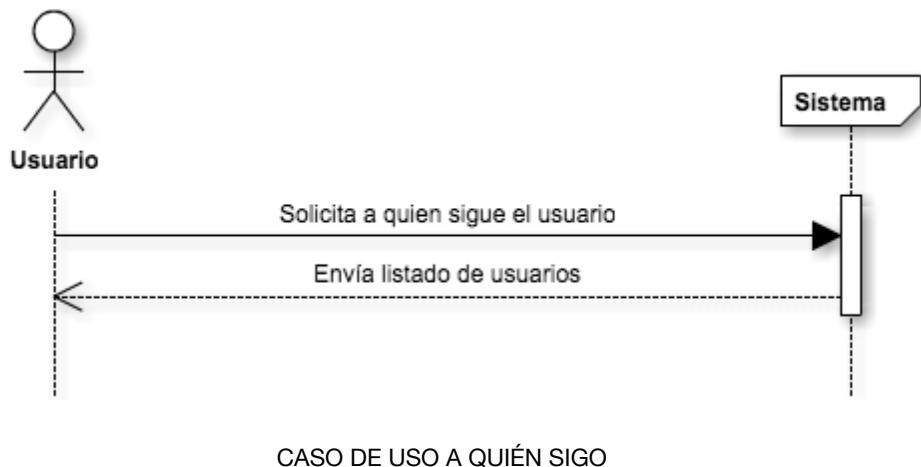
**Descripción:** Un usuario registrado ve un listado de todos los usuarios que le están siguiendo en la plataforma, de manera que pueda acceder a perfil de cada uno de ellos.

Caso de Uso A quién sigo

### Descripción

Este caso de uso devuelve un listado de todos los usuarios que está siguiendo el usuario que accede a la plataforma.

### Diagrama de interacción



### Escenario: A quién sigo

**Numeración:** 1.10

**Precondiciones:** Usuario registrado

**Postcondiciones:** Listado de usuarios que sigue el usuario que accede al sistema en la plataforma.

**Quién lo comienza:** Usuario registrado

**Quién lo finaliza:** Usuario registrado

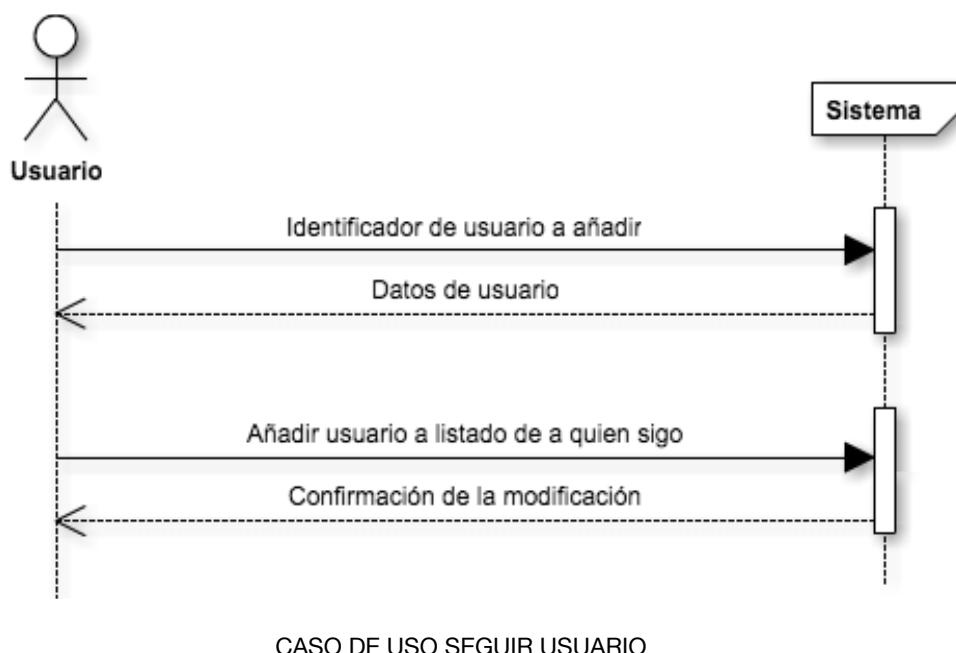
**Descripción:** Un usuario registrado pasa a ver un listado de todos los usuarios que está siguiendo en la plataforma, de manera que pueda acceder a perfil de cada uno de ellos.

## Caso de Uso Seguir Usuario

### Descripción

Este caso de uso hace que el usuario que accede a la plataforma añada a su lista de usuarios a los que sigue al usuario seleccionado.

### Diagrama de interacción



CASO DE USO SEGUIR USUARIO

#### Escenario: Seguir usuario

**Numeración:** 1.11

**Precondiciones:** Usuario registrado no sigue previamente al usuario seleccionado

**Postcondiciones:** Usuario registrado añade al usuario seleccionado como uno de los que sigue en la plataforma

**Quién lo comienza:** Usuario registrado

**Quién lo finaliza:** Usuario registrado

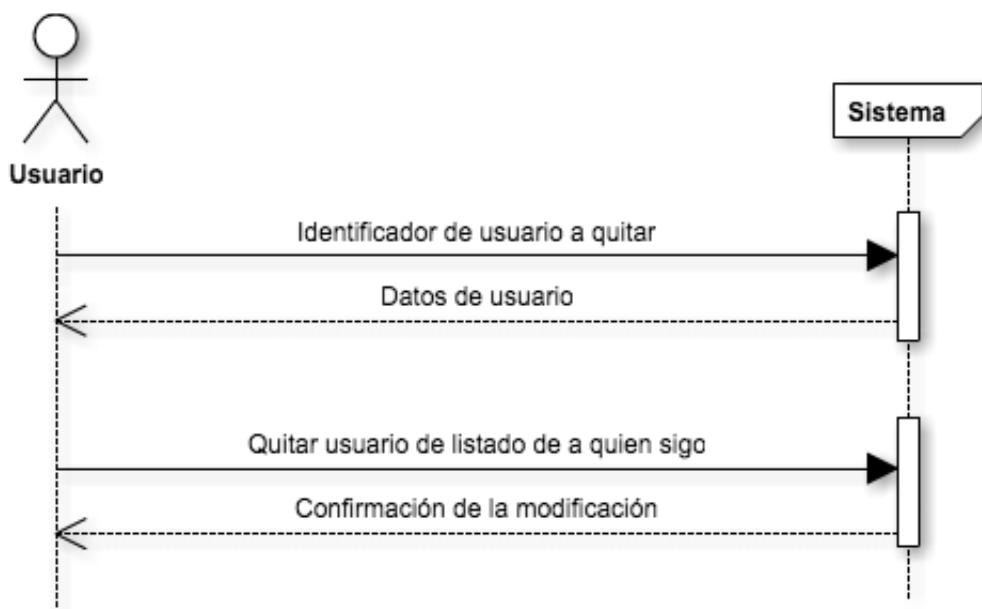
**Descripción:** Un usuario registrado que no seguía a otro usuario del sistema lo añade a su listado de usuarios a los que sigue en la plataforma.

## Caso de Uso Dejar de Seguir Usuario

### Descripción

Este caso de uso hace que el usuario que accede a la plataforma quite de su lista de usuarios a los que sigue al usuario seleccionado.

### Diagrama de interacción



CASO DE USO DEJAR DE SEGUIR USUARIO

#### Escenario: Dejar de seguir usuario

**Numeración:** 1.12

**Precondiciones:** Usuario registrado selecciona a un usuario que sigue previamente

**Postcondiciones:** Usuario registrado deja de seguir al usuario seleccionado

**Quién lo comienza:** Usuario registrado

**Quién lo finaliza:** Usuario registrado

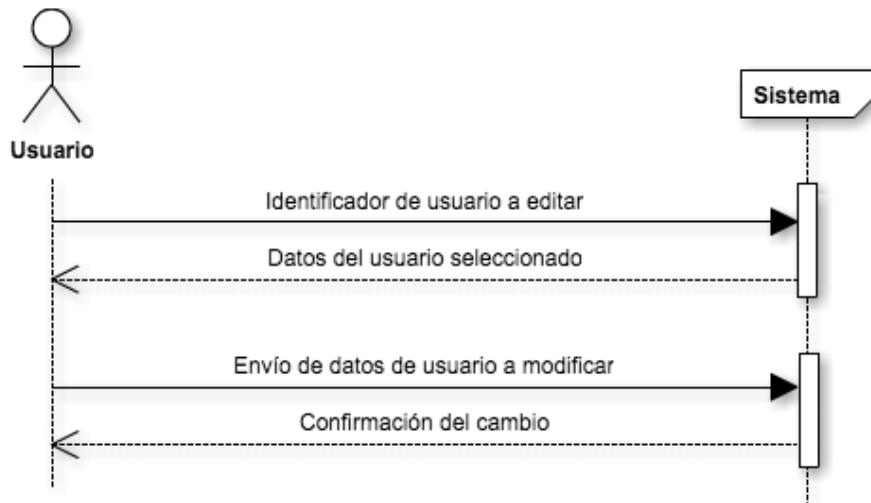
**Descripción:** Un usuario registrado que seguía a otro usuario del sistema deja de seguirlo en la plataforma.

## Caso de Uso Edición de Usuario

### Descripción

Este caso de uso permite al usuario que accede a la plataforma editar sus datos.

### Diagrama de interacción



CASO DE USO EDICIÓN DE USUARIO

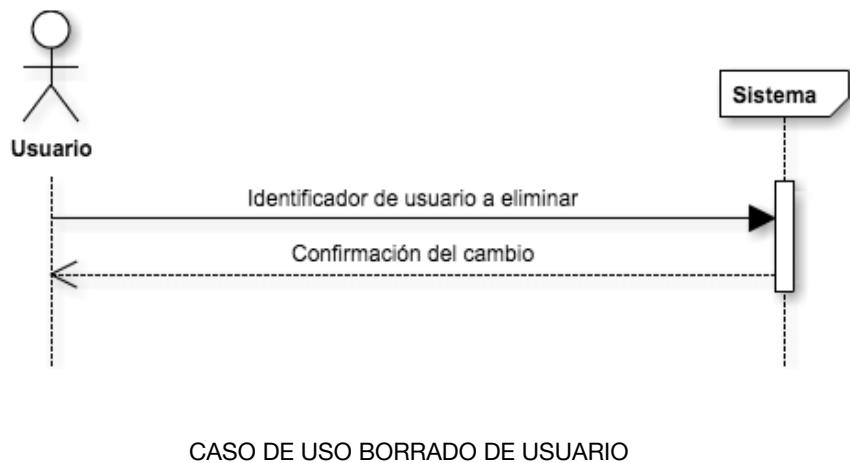
Escenario: Edición de usuario	
<b>Numeración:</b>	1.13
<b>Precondiciones:</b>	Usuario registrado y que ha accedido a la plataforma.
<b>Postcondiciones:</b>	Usuario que accede modifica su información personal.
<b>Quién lo comienza:</b>	Usuario registrado
<b>Quién lo finaliza:</b>	Usuario registrado
<b>Descripción:</b>	Un usuario registrado y con permisos de edición en el usuario seleccionado modifica la información disponible.

## Caso de Uso Borrado de Usuario

### Descripción

Este caso de uso permite al usuario que accede a la plataforma eliminar su cuenta de usuario impidiéndole el acceso en el futuro y eliminar todos sus datos relacionados.

### Diagrama de interacción



CASO DE USO BORRADO DE USUARIO

#### Escenario: Borrado de usuario

**Numeración:** 1.14

**Precondiciones:** Usuario registrado y que ha accedido a la plataforma

**Postcondiciones:** Usuario registrado deja de seguir al usuario seleccionado

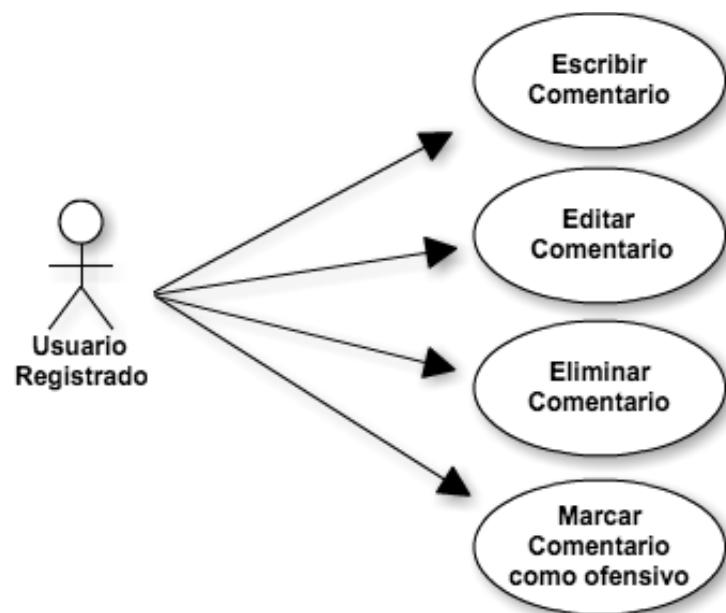
**Quién lo comienza:** Usuario registrado

**Quién lo finaliza:** Usuario anónimo

**Descripción:** Un usuario registrado elimina su cuenta de la plataforma eliminándose toda su información relativa en el portal.

### 3.4.2. Subsistema Gestión de Comentarios

#### Modelo de Casos de Uso



SUBSISTEMA GESTIÓN DE COMENTARIOS

## Descripción de Casos de Uso

Caso de Uso Escribir Comentario

### Descripción

Este caso de uso se encarga de la asociación de un comentario dentro de los elementos que aceptan comentarios dentro de la plataforma.

### Diagrama de interacción



#### Escenario: Escribir Comentario

##### Numeración: 2.1

**Precondiciones:** Usuario registrado que ha accedido a la plataforma y está dentro de un elemento en el que tiene permiso para escribir un comentario.

**Postcondiciones:** Usuario registrado deja comentario en elemento seleccionado.

**Quién lo comienza:** Usuario registrado

**Quién lo finaliza:** Usuario registrado

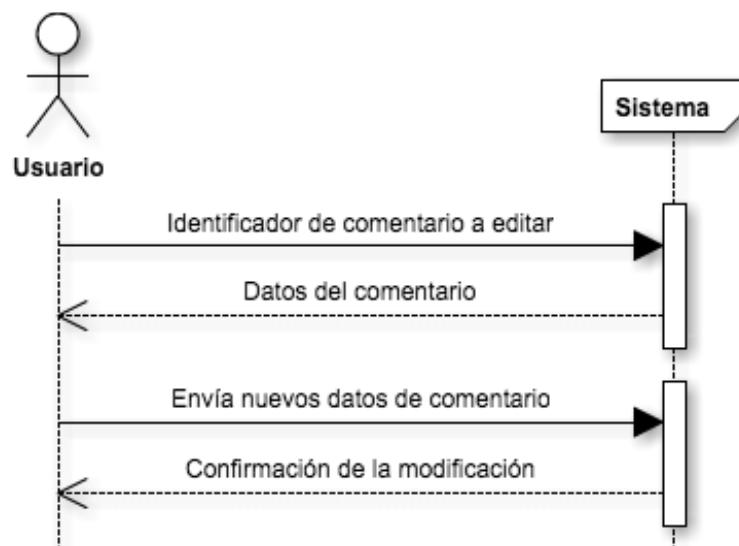
**Descripción:** Un usuario que tiene permiso dentro de un elemento al que ha accedido deja un comentario asociado al mismo.

## Caso de Uso Edición de Comentario

### Descripción

Este caso de uso permite al usuario que ha escrito un comentario dentro de un elemento de la plataforma editarlo y modificar el contenido del mismo.

### Diagrama de interacción



CASO DE USO EDICIÓN DE COMENTARIO

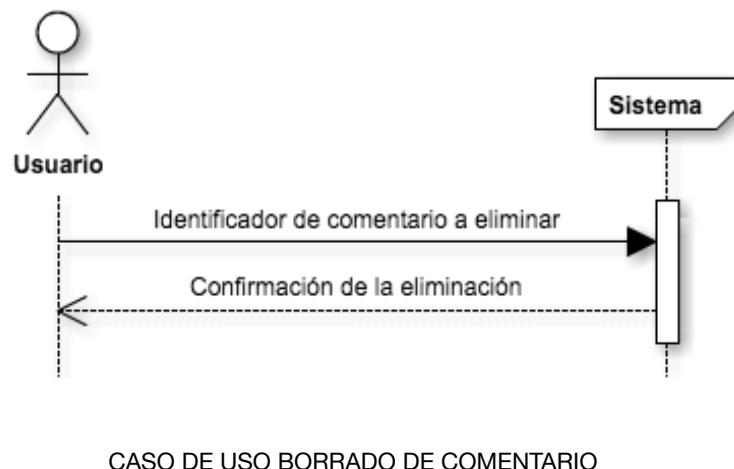
Escenario: Edición Comentario	
<b>Numeración:</b>	2.2
<b>Precondiciones:</b>	Usuario registrado que ha accedido a la plataforma y selecciona un comentario sobre el que tiene permiso.
<b>Postcondiciones:</b>	Se realizan modificaciones sobre el comentario seleccionado.
<b b="" comienza:<="" lo="" quién=""></b>	Usuario registrado con permisos de edición sobre el comentario seleccionado.
<b b="" finaliza:<="" lo="" quién=""></b>	Usuario registrado con permisos de edición sobre el comentario seleccionado.
<b b="" descripción:<=""></b>	Un usuario con permisos de administración en la ruta seleccionada puede realizar modificaciones en la información que se muestra en la misma.

## Caso de Uso Borrado de Comentario

### Descripción

Este caso de uso permite al usuario que ha escrito el comentario dentro de un elemento de la plataforma eliminarlo de la misma.

### Diagrama de interacción



CASO DE USO BORRADO DE COMENTARIO

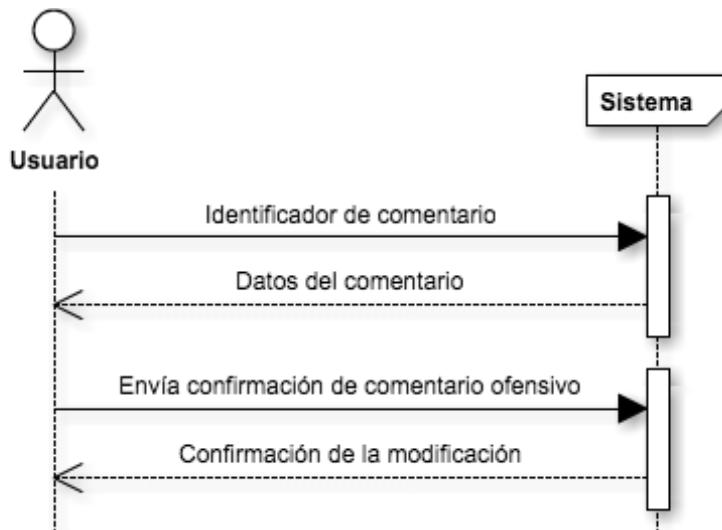
Escenario: Borrado Comentario	
<b>Numeración:</b>	2.3
<b>Precondiciones:</b>	Usuario registrado que accede a la plataforma y tiene permiso de borrado sobre el comentario seleccionado.
<b>Postcondiciones:</b>	Se elimina el comentario seleccionado.
<b>Quién lo comienza:</b>	Usuario registrado con permisos de borrado sobre el comentario seleccionado.
<b>Quién lo finaliza:</b>	Usuario registrado con permisos de borrado sobre el comentario seleccionado.
<b>Descripción:</b>	Un usuario con permisos de borrado sobre el comentario seleccionado lo elimina de la aplicación.

## Caso de Uso Marcar comentario como ofensivo

### Descripción

Este caso de uso permite a un usuario registrado en la plataforma avisar a la administración de la misma de que considera ofensivo un comentario.

### Diagrama de interacción



CASO DE USO MARCAR COMENTARIO COMO OFENSIVO

#### Escenario: Marcar Comentario como Ofensivo

**Numeración:** 2.4

**Precondiciones:** Usuario registrado

**Postcondiciones:** Comentario marcado como ofensivo a la espera de actuación de la administración de la aplicación.

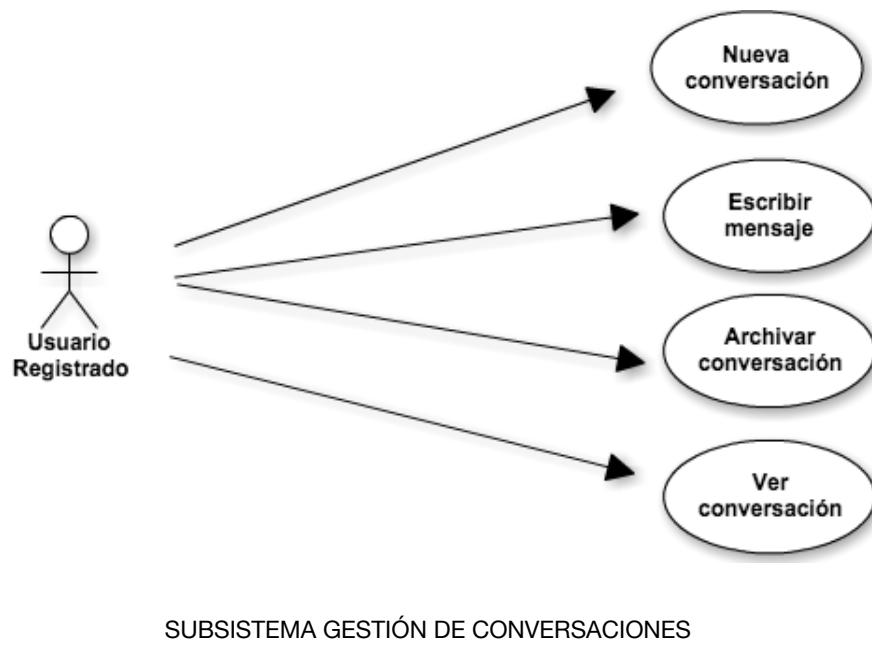
**Quién lo comienza:** Usuario registrado

**Quién lo finaliza:** Usuario registrado

**Descripción:** Un usuario registrado tiene posibilidad de marcar un comentario como ofensivo de manera que los usuarios administradores decidirán que pasos seguir con dicho comentario y con el usuario que lo haya creado.

### 3.4.3. Subsistema Gestión de Conversaciones

#### Modelo de casos de uso



## Descripción de Casos de Uso

Casos de Uso Nueva Conversación

### Descripción

Este caso de uso permite a un usuario registrado en la plataforma iniciar conversación con otro usuario registrado.

### Diagrama de interacción



### Escenario: Nueva Conversación

**Numeración:** 3.1

**Precondiciones:** Usuario registrado y que ha accedido a la plataforma.

**Postcondiciones:** Conversación creada con el usuario registrado y seleccionado.

**Quién lo comienza:** Usuario registrado.

**Quien lo finaliza:** Usuario registrado.

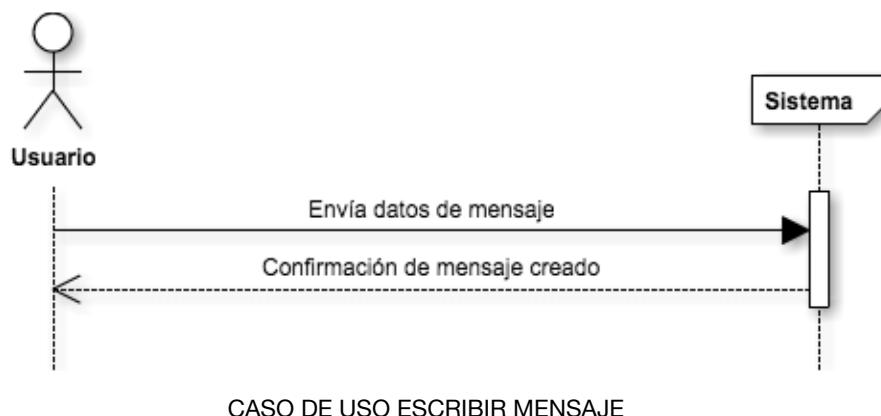
**Descripción:** Un usuario registrado inicia conversación con otro usuario registrado de la plataforma.

## Casos de Uso Escribir Mensaje

### Descripción

Este caso de uso permite a un usuario registrado que había iniciado conversación con otro usuario escribir un mensaje.

### Diagrama de interacción



### Escenario: Escribir un mensaje

**Numeración:** 3.2

**Precondiciones:** Usuario registrado que ha iniciado una nueva conversación con otro usuario registrado.

**Postcondiciones:** Mensaje que se anexa en la conversación existente entre dos usuarios registrados en la plataforma.

**Quién lo comienza:** Usuario registrado.

**Quien lo finaliza:** Usuario registrado.

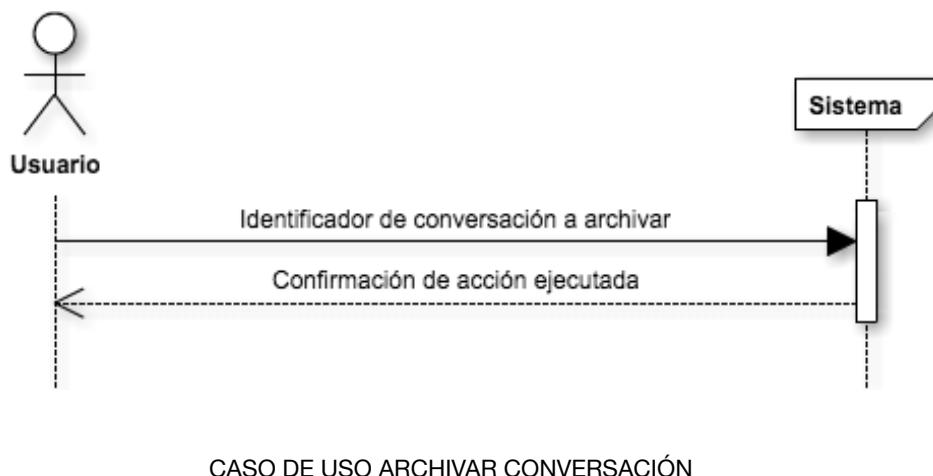
**Descripción:** Un usuario registrado que había iniciado conversación con otro usuario registrado le escribe un mensaje que se concatenará a los ya existentes previamente.

## Casos de Uso Archivar Conversación

### Descripción

Este caso de uso permite a un usuario registrado participante en una conversación archivar la misma para que no le aparezca dentro de sus conversaciones iniciales.

### Diagrama de interacción



CASO DE USO ARCHIVAR CONVERSACIÓN

### Escenario: Archivar Conversación

#### Numeración: 3.3

**Precondiciones:** Usuario registrado que ha iniciado una nueva conversación con otro usuario registrado.

**Postcondiciones:** Conversación se actualiza como archivada para que no aparezca en sus conversaciones principales.

**Quién lo comienza:** Usuario registrado.

**Quien lo finaliza:** Usuario registrado.

**Descripción:** Un usuario registrado que había iniciado conversación con otro usuario registrado pasa dicha conversación a su listado de conversaciones archivadas.

## Casos de Uso Ver Conversación

### Descripción

Este caso de uso permite a un usuario registrado participante en una conversación acceder la misma para ver los mensajes que se ha escrito con el otro usuario participante.

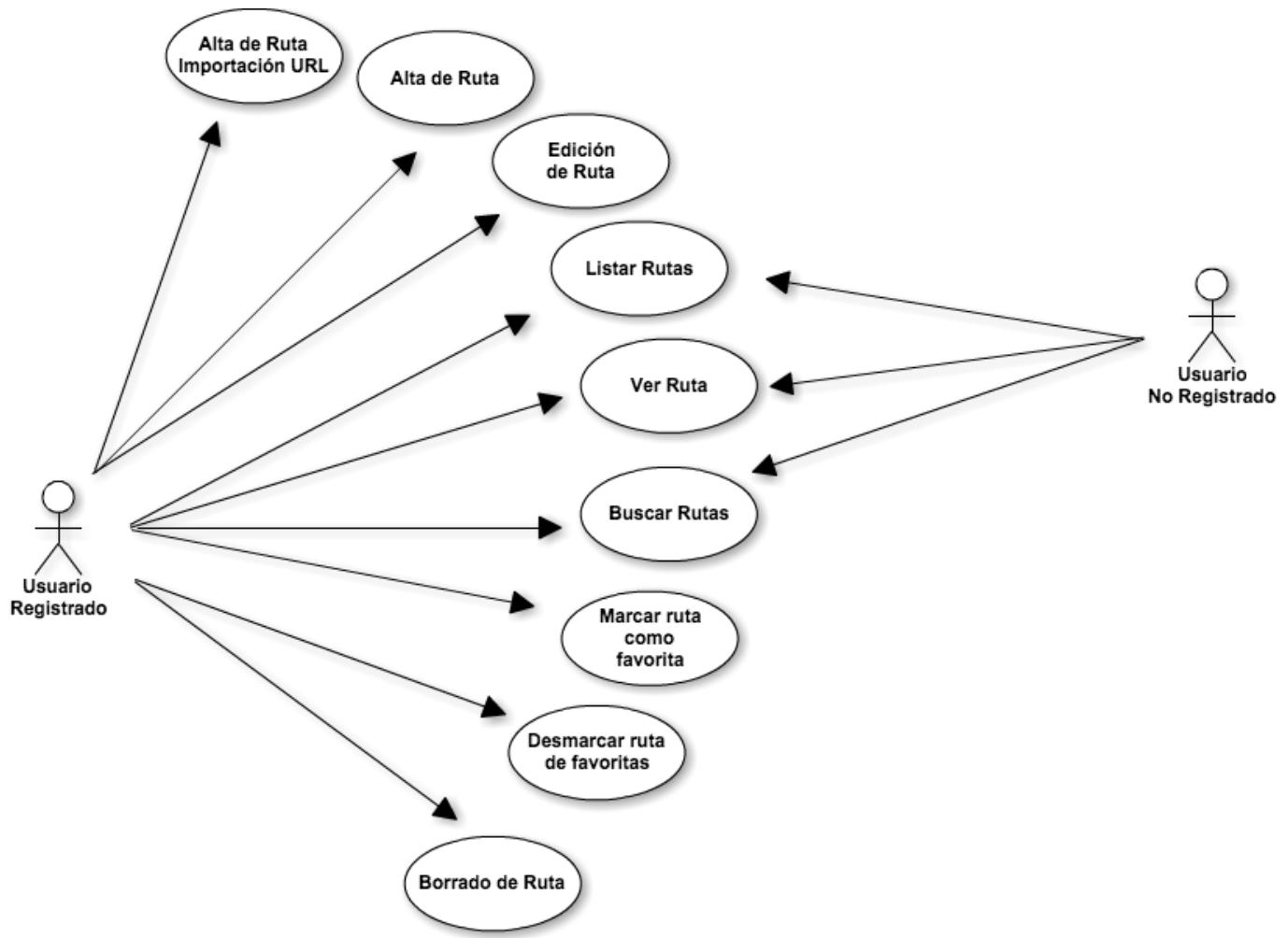
### Diagrama de interacción



Escenario: Ver Conversación
<b>Numeración:</b> 3.4
<b>Precondiciones:</b> Usuario registrado que ha iniciado una nueva conversación con otro usuario registrado previamente.
<b>Postcondiciones:</b> Usuario registrado que accede a la conversación marca la misma como leída.
<b>Quién lo comienza:</b> Usuario registrado.
<b>Quien lo finaliza:</b> Usuario registrado.
<b>Descripción:</b> Un usuario registrado que había iniciado conversación con otro usuario registrado accede a la conversación y puede ver todos los mensajes que se hubiese escrito previamente con dicho usuario.

### 3.4.4. Subsistema Gestión de Rutas

#### Modelo de Casos de Uso



SUBSISTEMA GESTIÓN DE RUTAS

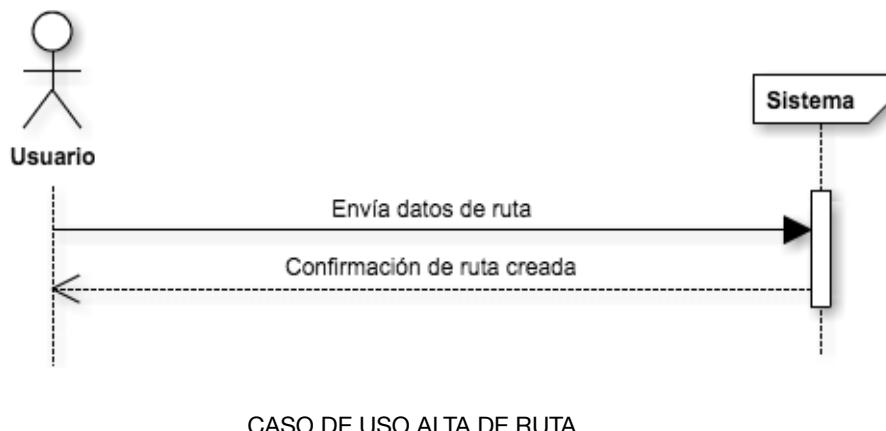
## Descripción de Casos de Uso

### Caso de Uso Alta de Ruta

#### Descripción

Este caso de uso permite a un usuario registrado en el sistema crear una ruta a partir de un fichero GPX o de manera manual sobre un mapa.

#### Diagrama de interacción



#### Escenario: Alta de Ruta

##### Numeración: 4.1

**Precondiciones:** Usuario registrado en el sistema que quiere crear una nueva ruta.

**Postcondiciones:** Un usuario registrado crea la ruta con los datos indicados.

**Quién lo comienza:** Usuario registrado

**Quién lo finaliza:** Usuario registrado

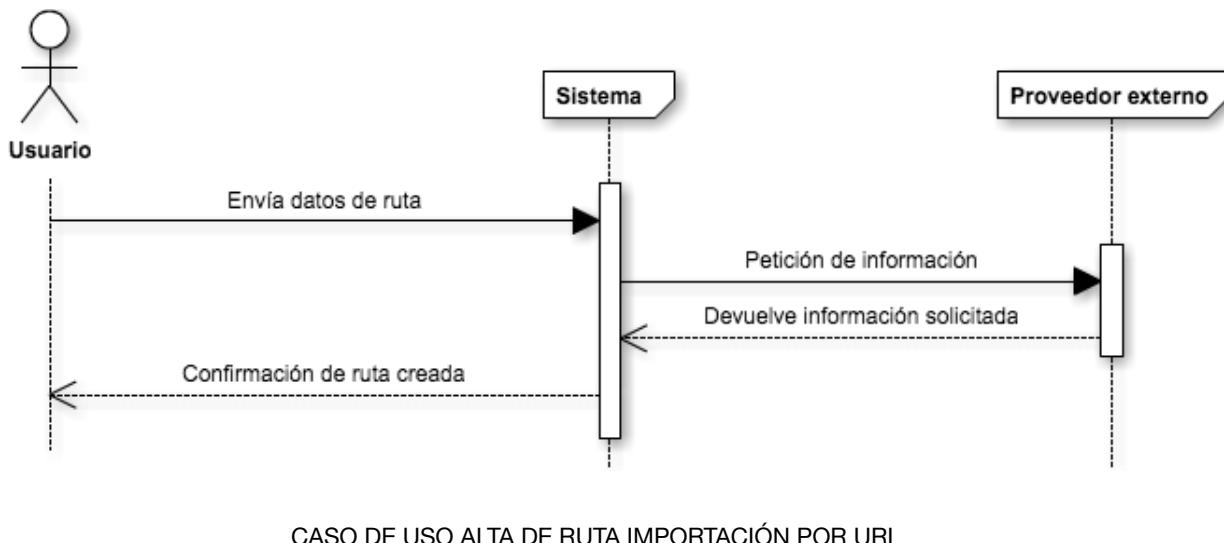
**Descripción:** Un usuario registrado en el sistema puede crear una ruta que será visible para el resto de usuarios visitantes del portal.

## Caso de Uso Alta de Ruta Importación por URL

### Descripción

Este caso de uso permite a un usuario registrado en el sistema crear una ruta a partir de una URL externa aceptada por la aplicación.

### Diagrama de interacción



#### Escenario: Alta de Ruta Importación URL

**Numeración:** 4.2

**Precondiciones:** Usuario registrado en el sistema que quiere crear una nueva ruta.

**Postcondiciones:** Un usuario registrado crea la ruta a partir de la URL indicada.

**Quién lo comienza:** Usuario registrado

**Quién lo finaliza:** Usuario registrado

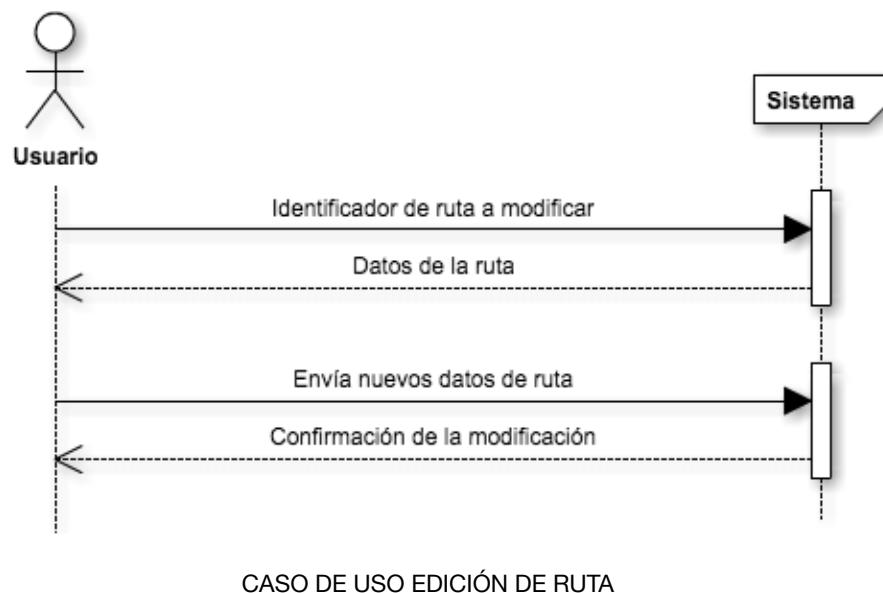
**Descripción:** Un usuario registrado en el sistema puede crear una ruta a partir de la URL indicada que será visible para el resto de usuarios visitantes del portal.

## Caso de Uso Edición de Ruta

### Descripción

Este caso de uso permite al usuario que ha creado la ruta seleccionada editar la misma para ajustar los contenidos que se muestran.

### Diagrama de interacción



CASO DE USO EDICIÓN DE RUTA

#### Escenario: Edición de Ruta

**Numeración:** 4.3

**Precondiciones:** Usuario registrado con permisos de edición sobre la ruta seleccionada.

**Postcondiciones:** Se realizan modificaciones sobre la ruta seleccionada

**Quién lo comienza:** Usuario registrado con permisos de edición sobre la ruta seleccionada

**Quién lo finaliza:** Usuario registrado con permisos de edición sobre la ruta seleccionada

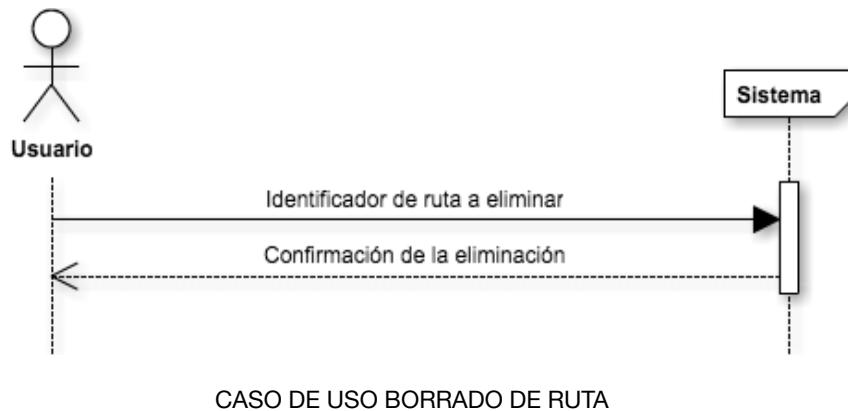
**Descripción:** Un usuario con permisos de edición en la ruta seleccionada puede realizar modificaciones en la información que se muestra en la misma.

## Caso de Uso Borrado de Ruta

### Descripción

Este caso de uso permite al usuario que ha creado la ruta seleccionada borrar la misma del sistema.

### Diagrama de interacción



### Escenario: Borrado de Ruta

**Numeración:** 4.4

**Precondiciones:** Usuario registrado con permisos para eliminar la ruta seleccionada.

**Postcondiciones:** La ruta seleccionada se elimina del sistema.

**Quién lo comienza:** Usuario registrado con permisos de borrado sobre la ruta seleccionada.

**Quién lo finaliza:** Usuario registrado registrado con permisos de borrado sobre la ruta seleccionada.

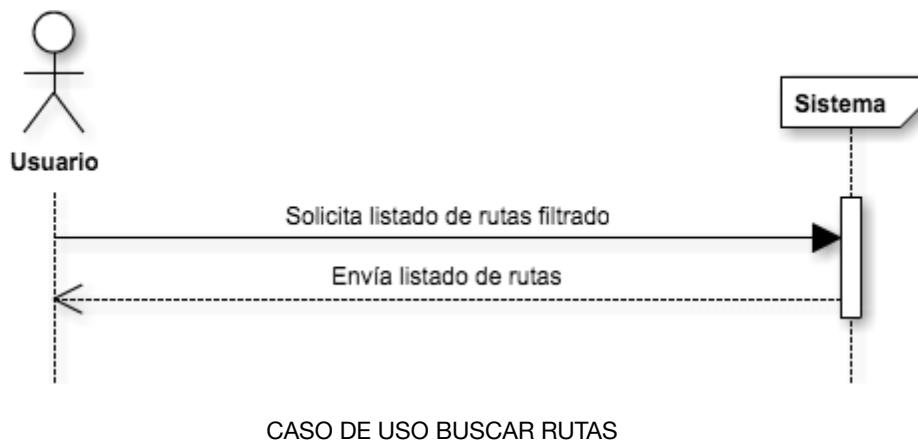
**Descripción:** Un usuario con permisos de borrado sobre la ruta seleccionada tiene posibilidad de eliminarla del sistema imposibilitando la visualización de la misma.

## Caso de Uso Buscar Rutas

### Descripción

Este caso de uso permite a un usuario visitante en la web obtener un listado de las rutas a partir de un filtro realizado.

### Diagrama de interacción



#### Escenario: Búsqueda de rutas

**Numeración:** 4.5

**Postcondiciones:** Lista de rutas filtradas por los parámetros indicados

**Quién lo comienza:** Usuario anónimo o registrado

**Quién lo finaliza:** Usuario anónimo o registrado

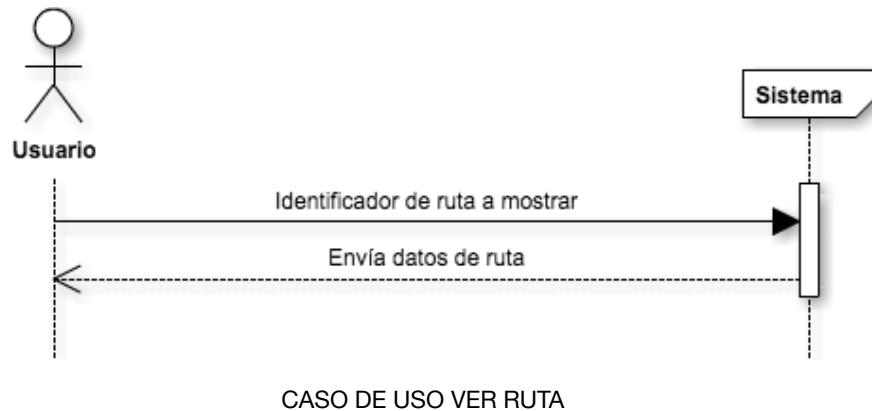
**Descripción:** Se mostrará por pantalla un listado con todas las rutas que se hayan encontrado con los filtros indicados durante la búsqueda.

## Caso de Uso Ver Ruta

### Descripción

Este caso de uso permite a un usuario visitante ver el detalle de la ruta seleccionada.

### Diagrama de interacción



### Escenario: Ver ruta

**Numeración:** 4.6

**Postcondiciones:** Pantalla de detalle de la ruta seleccionada.

**Quién lo comienza:** Usuario anónimo o registrado

**Quién lo finaliza:** Usuario anónimo o registrado

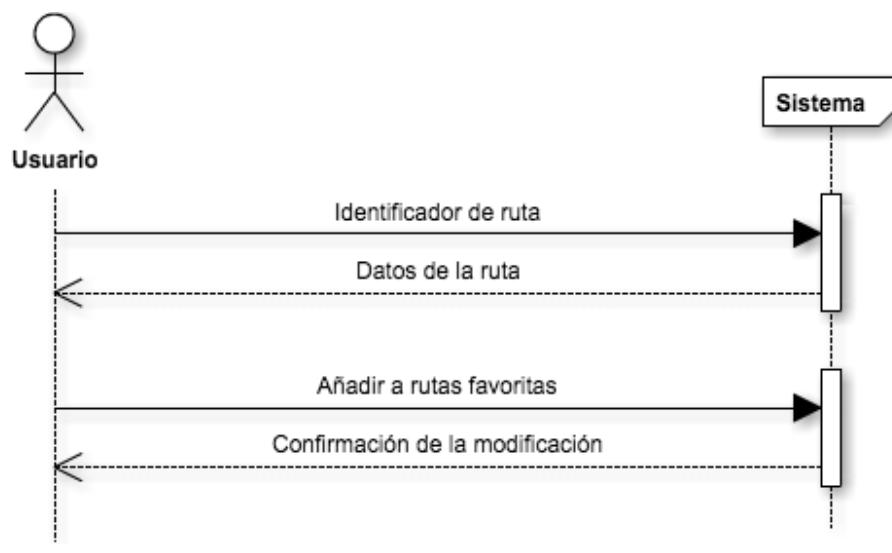
**Descripción:** Se muestra una pantalla la ruta de manera organizada permitiendo ver al usuario toda la información relacionada con la misma.

## Caso de Uso Marcar Ruta como Favorita

### Descripción

Este caso de uso permite a un usuario registrado marcar la ruta seleccionada como una de sus favoritas.

### Diagrama de interacción



#### Escenario: Marcar Ruta como Favorita

**Numeración:** 4.7

**Precondiciones:** Usuario registrado añade una ruta no añadida previamente a sus favoritas.

**Postcondiciones:** La ruta seleccionada pasa a estar dentro del listado de rutas favoritas del usuario.

**Quién lo comienza:** Usuario registrado

**Quién lo finaliza:** Usuario registrado

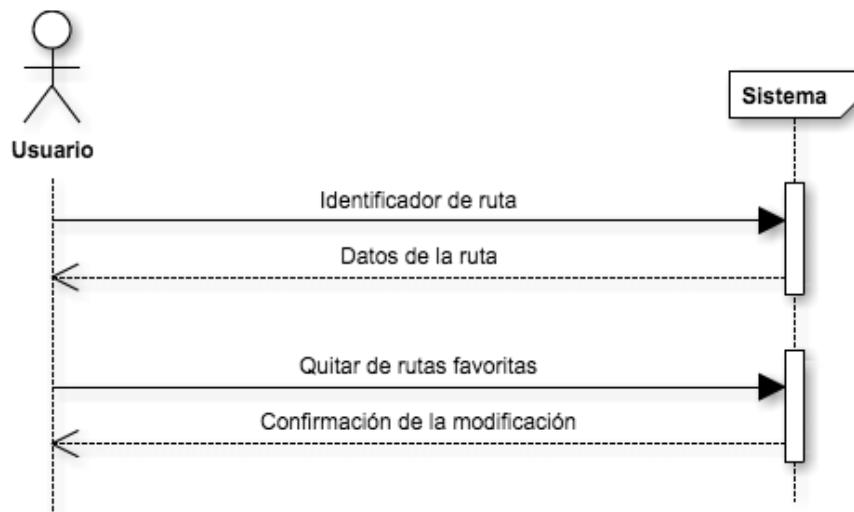
**Descripción:** Un usuario registrado puede marcar una ruta como una de sus favoritas, pasando a estar dentro de su listado de rutas favoritas.

## Caso de Uso Desmarcar Ruta como Favorita

### Descripción

Este caso de uso permite a un usuario registrado quitar de entre sus rutas favoritas la ruta seleccionada.

### Diagrama de interacción



CASO DE USO DESMARCAR RUTA COMO FAVORITA

#### Escenario: Desmarcar Ruta como Favorita

**Numeración:** 4.8

**Precondiciones:** Usuario registrado desmarca una ruta que previamente estaba en su listado de rutas favoritas.

**Postcondiciones:** La ruta seleccionada se elimina del listado de rutas favoritas del usuario.

**Quién lo comienza:** Usuario registrado

**Quién lo finaliza:** Usuario registrado

**Descripción:** Un usuario registrado puede desmarcar una ruta de sus favoritas, eliminándose del listado de favoritas en el que estaba previamente.

## Caso de Uso Listar Rutas

### Descripción

Este caso de uso permite a un usuario listar todas las rutas visibles en el sistema de manera que pueda acceder al detalle de las mismas.

### Diagrama de interacción

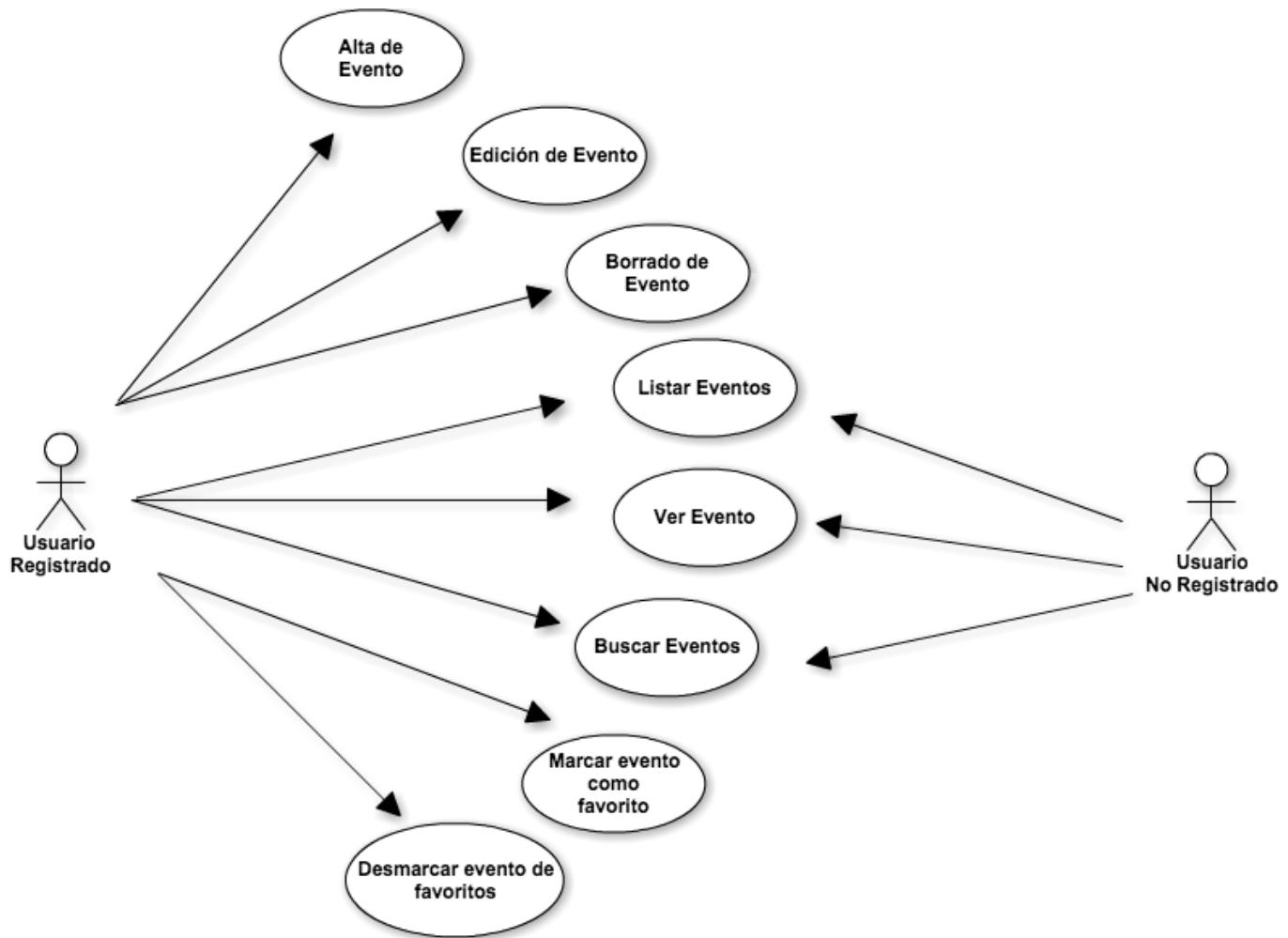


CASO DE USO LISTAR RUTAS

Escenario: Listar Rutas	
<b>Numeración:</b>	4.9
<b>Postcondiciones:</b>	Lista de las rutas que tenemos en el sistema
<b>Quién lo comienza:</b>	Usuario
<b>Quién lo finaliza:</b>	Usuario
<b>Descripción:</b>	Se ofrece un listado de las rutas disponibles en el sistema, no es necesario ser usuario registrado para ver dicho listado y nos permite acceder al detalle de cada una de las rutas disponibles.

### 3.4.5. Subsistema Gestión de Eventos

#### Modelo de Casos de Uso



SUBSISTEMA GESTIÓN DE EVENTOS

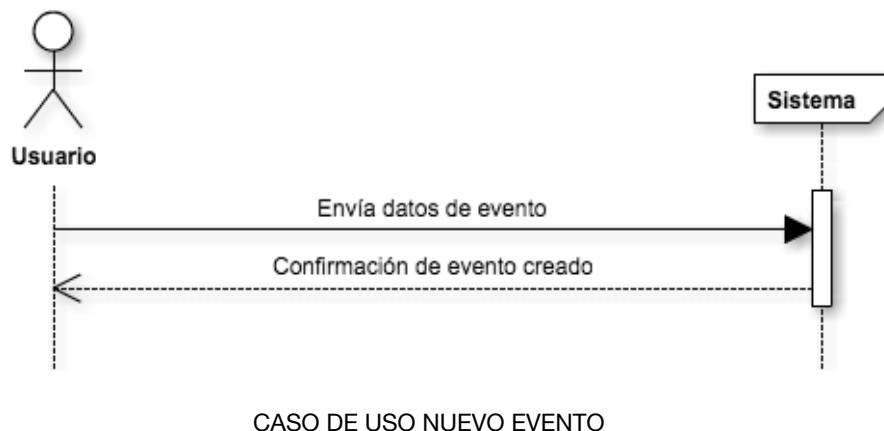
## Descripción de Casos de Uso

Caso de Uso Alta de Evento

### Descripción

Este caso de uso permite a un usuario registrado en la aplicación dar de alta un nuevo evento.

### Diagrama de interacción



#### Escenario: Nuevo evento

**Numeración:** 5.1

**Precondiciones:** Usuario registrado y que ha accedido al sistema.

**Postcondiciones:** Usuario registrado crea el evento a partir de la información añadida.

**Quién lo comienza:** Usuario registrado

**Quién lo finaliza:** Usuario registrado

**Descripción:** Un usuario registrado tiene posibilidad de crear un evento (quedada ciclista, competición, etc) permitiéndole crear la información en el portal y tener un lugar donde transmitirla.

## Caso de Uso Edición de Evento

### Descripción

Este caso de uso permite a un usuario registrado y con permiso de edición sobre un evento modificar los datos del mismo.

### Diagrama de interacción



### Escenario: Edición de evento

**Numeración:** 5.2

**Precondiciones:** Usuario registrado con permisos de edición sobre el evento seleccionado.

**Postcondiciones:** Usuario registrado con permisos de edición sobre el evento seleccionado modifica la información del mismo.

**Quién lo comienza:** Usuario registrado con permisos de edición sobre el evento seleccionado.

**Quién lo finaliza:** Usuario registrado con permisos de edición sobre el evento seleccionado.

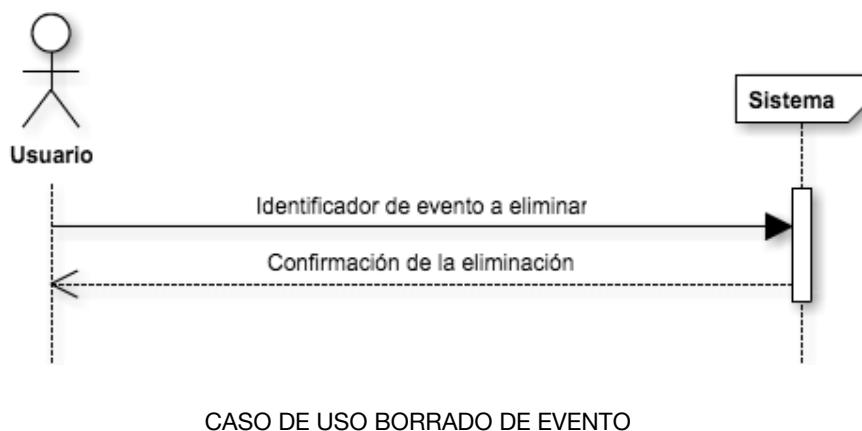
**Descripción:** Un usuario con permisos de administración sobre el evento tiene la posibilidad de realizar modificaciones en la información que se muestra en el mismo.

## Caso de Uso Borrado de Evento

### Descripción

Este caso de uso permite a un usuario registrado con permiso de borrado sobre un evento eliminarlo de la aplicación.

### Diagrama de interacción



CASO DE USO BORRADO DE EVENTO

#### Escenario: Borrado de Evento

**Numeración:** 5.3

**Precondiciones:** Usuario registrado con permisos de borrado sobre el evento seleccionado.

**Postcondiciones:** El usuario elimina el evento del sistema.

**Quién lo comienza:** Usuario registrado con permisos de administración sobre el evento

**Quién lo finaliza:** Usuario registrado con permisos de administración sobre el evento

**Descripción:** Un usuario puede eliminar la información del evento. Dicho evento no muestra ningún tipo de información más ya que es eliminado del sistema.

## Caso de Uso Ver Evento

### Descripción

Este caso de uso permite a un usuario visitante ver el detalle del evento seleccionado.

### Diagrama de interacción



CASO DE USO VER EVENTO

#### Escenario: Detalle de evento

**Numeración:** 5.4

**Postcondiciones:** El usuario visitante tiene acceso al detalle del evento seleccionado.

**Quién lo comienza:** Usuario anónimo o registrado

**Quién lo finaliza:** Usuario anónimo o registrado

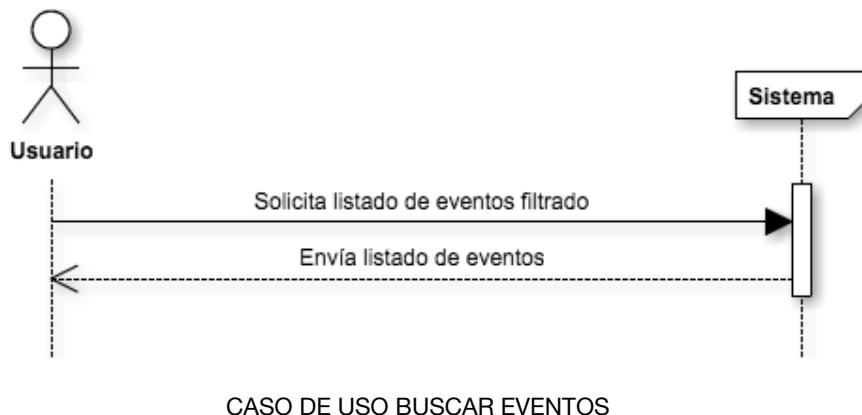
**Descripción:** Un usuario anónimo o registrado tiene la posibilidad de ver la información a mostrar de un evento. Como usuario registrado podrá ver más información que la que puede ver un usuario anónimo.

## Caso de Uso Buscar Eventos

### Descripción

Este caso de uso permite a un usuario visitante obtener un listado de eventos a partir del filtro de búsqueda realizado.

### Diagrama de interacción



### Numeración: Búsqueda de eventos

**Numeración:** 5.5

**Postcondiciones:** Lista de eventos a partir de los filtros de búsqueda seleccionados.

**Quién lo comienza:** Usuario anónimo o registrado.

**Quién lo finaliza:** Usuario anónimo o registrado.

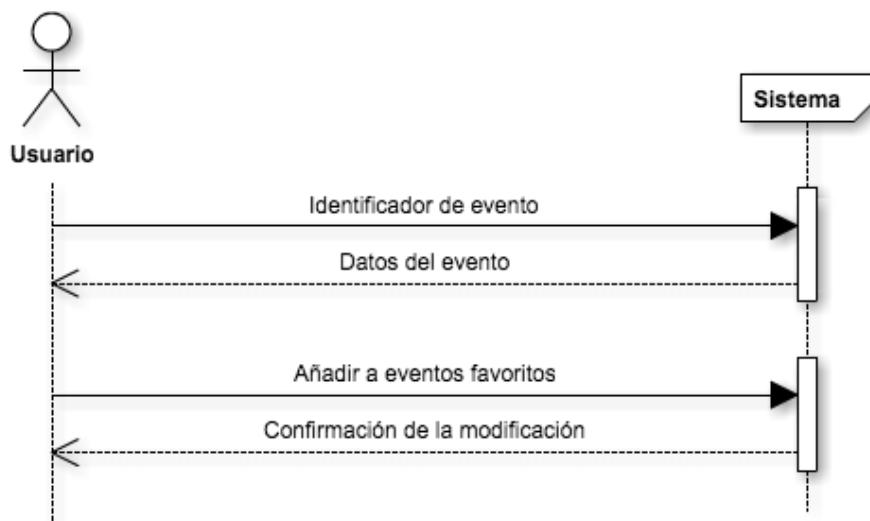
**Descripción:** Un usuario anónimo o registrado puede hacer una búsqueda de eventos por distintos filtros obteniendo un listado con los eventos que cumplan dichas condiciones.

## Caso de Uso Marcar Evento como Favorito

### Descripción

Este caso de uso permite a un usuario registrado marcar un evento seleccionado como uno de sus eventos favoritos.

### Diagrama de interacción



CASO DE USO MARCAR EVENTO COMO FAVORITO

#### Escenario: Marcar evento como favorito

**Numeración:** 5.6

**Precondiciones:** Usuario registrado que no tenga dentro de sus eventos favoritos al evento seleccionado.

**Postcondiciones:** Usuario registrado añade al listado de sus eventos favoritos el evento seleccionado.

**Quién lo comienza:** Usuario registrado

**Quién lo finaliza:** Usuario registrado

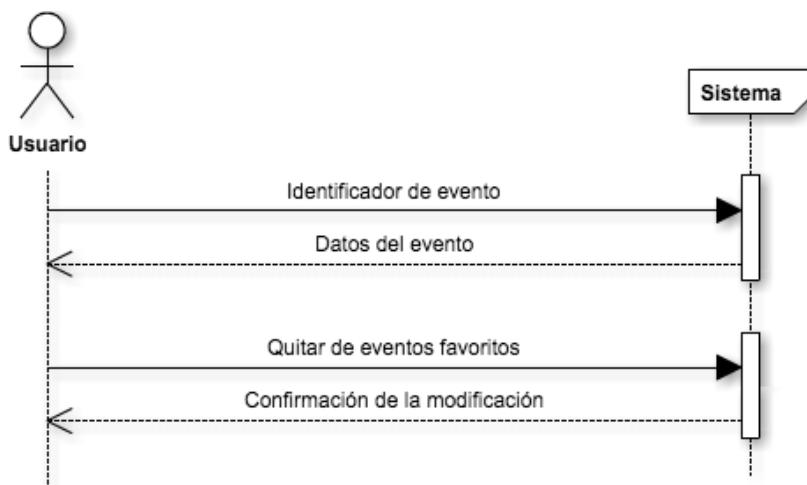
**Descripción:** Un usuario puede añadir el evento seleccionado a su listado de eventos favoritos de manera que pueda acceder de manera más rápida al mismo.

## Caso de Uso Desmarcar Evento de Favoritos

### Descripción

Este caso de uso permite a un usuario registrado desmarcar un evento que previamente estuviese seleccionado como favorito.

### Diagrama de interacción



CASO DE USO DESMARCAR EVENTO DE FAVORITOS

### Escenario: Desmarcar Evento de Favoritos

**Numeración:** 5.7

**Precondiciones:** Usuario registrado que tenga dentro de sus eventos favoritos al evento seleccionado.

**Postcondiciones:** Usuario registrado elimina de sus favoritos el evento seleccionado.

**Quién lo comienza:** Usuario registrado

**Quién lo finaliza:** Usuario registrado

**Descripción:** Un usuario puede quitar un evento que tuviese dentro de su listado de favoritos de manera que no aparezca en dicho listado.

## Caso de Uso Listar Eventos

### Descripción

Este caso de uso permite a un usuario ver un listado de todos los eventos disponibles en la plataforma.

### Diagrama de interacción



### Escenario: Listar Eventos

**Numeración:** 5.8

**Postcondiciones:** Usuario recibe un listado de los eventos disponibles.

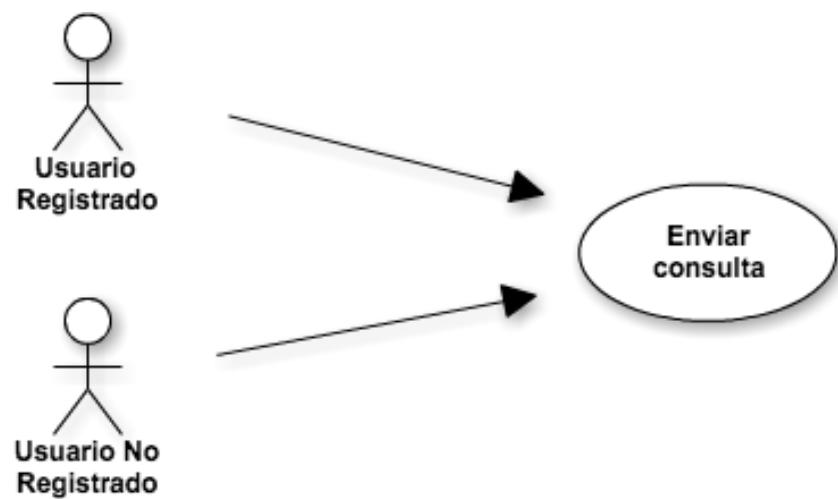
**Quién lo comienza:** Usuario.

**Quién lo finaliza:** Usuario.

**Descripción:** Un usuario puede ver un listado de todos los eventos disponibles en la plataforma y de dicha manera acceder al detalle de cada uno de ellos.

### 3.4.6. Subsistema Gestión de Contacto

#### Modelo de Casos de Uso



SUBSISTEMA GESTIÓN DE CONTACTO

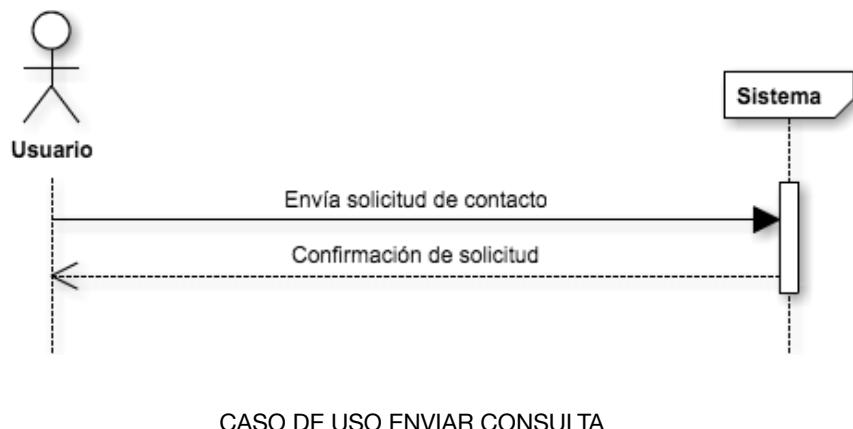
## Descripción de Casos de Uso

Caso de Uso Enviar Consulta

### Descripción

Este caso de uso permite a un usuario visitante de la aplicación enviar una consulta a la administración de la aplicación.

### Diagrama de interacción



#### Escenario: Solicitar información

##### Numeración: 6.1

**Postcondiciones:** Se envía un email a la administración de la aplicación con la consulta realizada por el usuario.

**Quién lo comienza:** Usuario anónimo o registrado

**Quién lo finaliza:** Usuario anónimo o registrado

**Descripción:** Cualquier usuario anónimo o registrado puede acceder al formulario de solicitud de información que se encuentra tanto en la parte pública como privada de la aplicación. Una vez cumplimentados todos los datos obligatorios del formulario el usuario podrá realizar el envío de su solicitud de información.

Los administradores de la aplicación recibirán un correo electrónico en el que se mostrará toda la información introducida previamente en el formulario.

Una vez enviado se mostrará un mensaje de aviso al usuario que envía la solicitud.



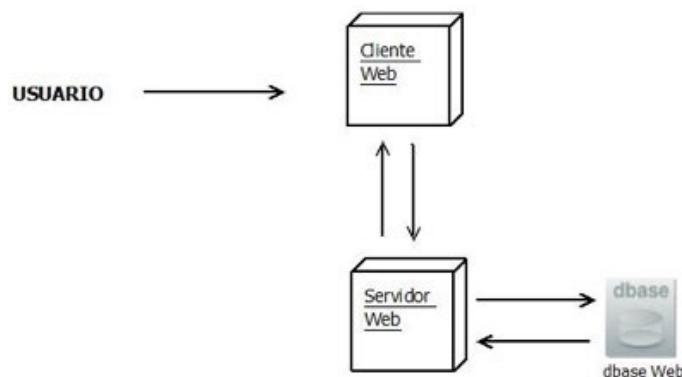
# 4. Diseño del Sistema de la Información

## 4.1. Diagrama de despliegue

El diagrama de despliegue muestra el particionamiento físico del sistema con sus distintos niveles. En nuestro caso podemos ver como un usuario que acceder a la web a través de un navegador accede a la *plataforma* y esta a su vez en los casos en los que sea necesario accederá a los *datos almacenados en el gestor de base de datos utilizado*.

La aplicación se conecta a la web mediante una interfaz remota de métodos y servicios, con peticiones de tipo HTTP vía método GET / POST / PUT / DELETE según las peticiones realizadas.

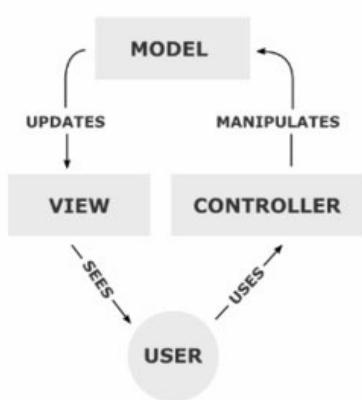
En dichas peticiones, siempre, se establece una conexión segura validando el usuario que está identificado en la aplicación como un usuario existente en el sistema, lo cual establece la seguridad necesaria para el conjunto del sistema.



Las respuestas de la aplicación se realizarán en formato JSON y de manera que se podrán utilizar en el front-end en una SPA (Single Page Application). En caso de que el usuario desactive Javascript será necesario que funcione igualmente mediante peticiones en formato HTML.

La aplicación a realizar utilizará un patrón de arquitectura de software de tipo Modelo-Vista-Controlador (MVC).

Una de las características principales de una aplicación web es que tiene que tener una interfaz que interactúe con el usuario, lo que se denomina como *vista*. Deberá representar la información con la cual el sistema opera, el *modelo*. Además, al ser una aplicación en la que el usuario decide cómo utilizar la funcionalidad ofrecida, deberá



responder a eventos, usualmente acciones del usuario, e invocar peticiones al modelo, el *controlador*.

Aunque se pueden encontrar diferentes implementaciones del MVC, el flujo que sigue el control generalmente es el siguiente:

*El usuario interactúa con la interfaz de usuario, la vista, de alguna forma (por ejemplo, el usuario pulsa un botón, enlace, etc.)*

*El controlador recibe, por parte de los objetos de la interfaz, la notificación de la acción solicitada por el usuario.* El controlador gestiona el evento que llega, frecuentemente a través de un gestor de eventos.

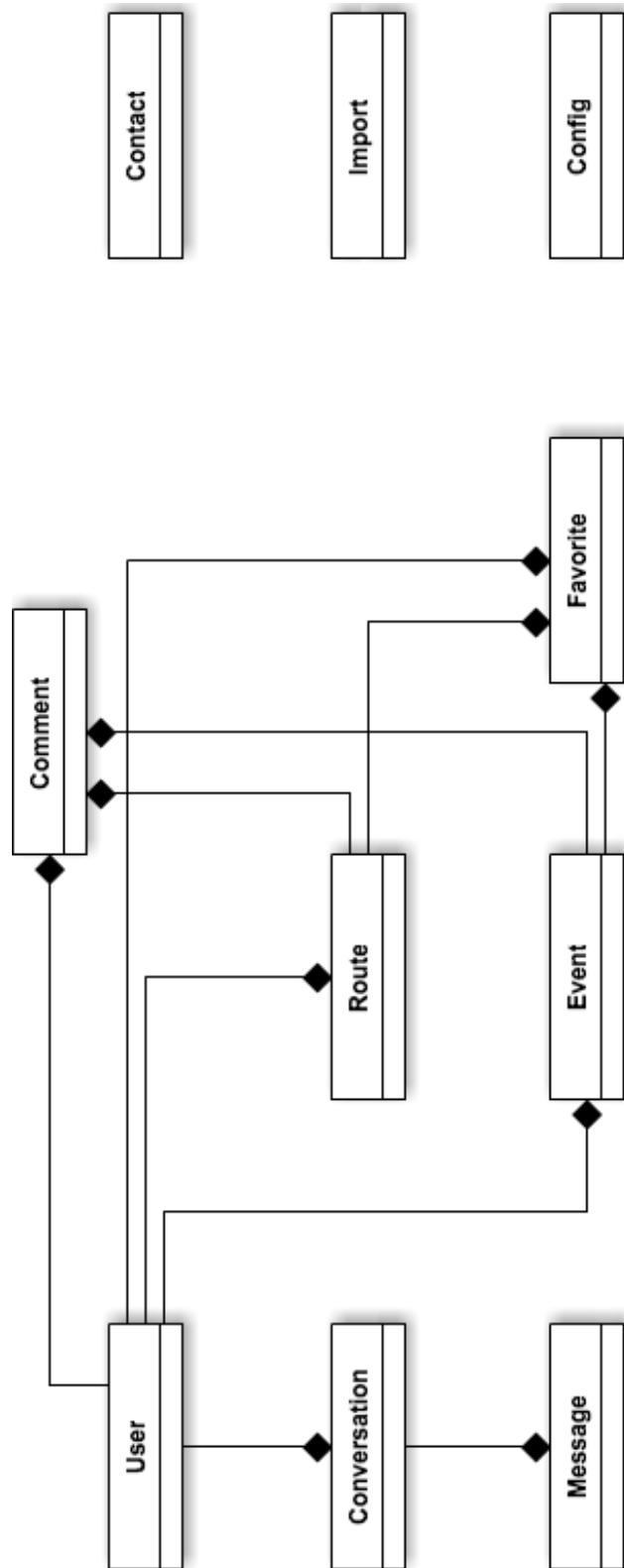
*El controlador accede al modelo, actualizándolo, posiblemente modificándolo de forma adecuada a la acción solicitada por el usuario (por ejemplo, el controlador actualiza los datos de un usuario).*

El controlador delega en los objetos de la vista la tarea de desplegar la interfaz de usuario. La vista obtiene sus datos del modelo para generar la interfaz apropiada para el usuario donde se reflejan los cambios en el modelo (por ejemplo, produce un listado de la búsqueda de sesiones). El modelo no debe tener conocimiento directo sobre la vista. Un objeto vista puede registrarse con el modelo y esperar los cambios, pero aun así el modelo en sí mismo sigue sin saber nada de la vista. El controlador no pasa objetos de dominio a la vista aunque puede dar la orden a la vista para que se actualice.

La interfaz de usuario espera nuevas interacciones del usuario, comenzando el ciclo nuevamente.

## 4.2. Modelo de Clases del Sistema

### 4.2.1. Diagrama de Clases



#### 4.2.2. Descripción de las Clases

##### User

Representa al usuario que accede al sistema, contiene toda la información relativa a un objeto usuario, desde su nombre, fecha de nacimiento, seguidores y a quien sigue así como otra serie de campos informativos.

Se relaciona con las clases Evento, Ruta y Comentarios almacenando la información que envía en cada momento.

##### Atributos

Atributo	Tipo	Requerido	Automático	Contenido almacenado
_id	BSON:ObjectId	Sí	Sí	Identificador único de usuario, se crea automáticamente a la hora de crear el objeto User correspondiente.
email	String	Sí	No	Email del usuario utilizado para el acceso a la plataforma
encrypted_password	String	Sí	No	Contraseña del usuario, se almacena de manera encriptada de manera que nadie más que él pueda saber la contraseña guardada.
reset_password_token	String	No	No	Cuando un usuario solicita un recordatorio de contraseña se genera un token único para que dicho usuario pueda acceder al formulario de recordatorio de contraseña.
reset_password_token_at	DateTime	No	No	Hora en la que el usuario solicita el recordatorio de contraseña, se usará para controlar la fecha de validez del token generado previamente.
remember_create_at	DateTime	No	No	Hora en la que el usuario ha seleccionado que se le recuerde el acceso a la plataforma.
sign_in_count	Integer	No	No	Número de veces que se ha conectado a la plataforma, un valor estadístico.
current_sign_in_at	DateTime	No	No	Hora de la conexión actual.
last_sign_in_at	DateTime	No	No	Hora de la última conexión realizada por el usuario.
about_me	String	No	No	Información del usuario para mostrar al resto de usuarios de la plataforma.

Atributo	Tipo	Requerido	Automático	Contenido almacenado
about_me	String	No	No	Información del usuario para mostrar al resto de usuarios de la plataforma.
birthday	DateTime	No	No	Fecha de nacimiento del usuario, totalmente informativo.
count_messages	Hash	No	No	Es un campo en el que se almacenará el número total de mensajes, numero de conversaciones archivadas y sin leer.
follow_ids	Array	No	No	Array que almacena los ids de los usuarios que seguimos en la plataforma.
follower_ids	Array	No	No	Array que almacena los ids de los usuarios que me siguen en la plataforma.
name	String	Sí	No	Nombre del usuario, usado en la plataforma para que otros usuarios nos encuentren.
privacity	Integer	No	No	Campo que controla la privacidad que el usuario desea disponer en la plataforma.
roles	Array	No	No	Contiene los distintos roles de los que dispone el usuario, se corresponden con los permisos en la plataforma.
tutorial_seen_at	DateTime	No	No	Contiene la hora en la que el usuario ha visto el tutorial, es obligatorio acceder a dicha página
web_page	String	No	No	Texto informativo sobre la página web del usuario si dispone de alguna.
created_at	DateTime	Sí	Sí	Hora de creación del usuario, se lanza automáticamente tras el evento que se produce en la creación del usuario.
updated_at	DateTime	Sí	Sí	Hora de actualización del usuario, se lanza cada vez que se ejecuta una actualización en los datos del usuario.

## Métodos

Método	Parámetros	Descripción
search	_params = {}	Método de búsqueda de usuarios a partir de los parámetros recibidos. Devuelve un array de objetos User.
to_csv	_users, _columns, options, params	Método para transformar un array de objetos User enviados como parámetro en formato CSV de salida.

Método	Parámetros	Descripción
add_favorite	_id, _type	Método que añade a favoritos el identificador del elemento pasado y del tipo enviado en el parámetro _type .
add_follow	_user	Añadir a seguidores el usuario enviado como parámetro.
admin?		Devuelve un boolean considerando si el usuario pertenece al grupo de administradores o no.
as_json	options	Método de Rails sobreescrito de manera que según las opciones enviadas como parámetro devolverá un tipo de JSON u otro, con más o menos información según la que se necesite para cada caso.
conversation	_user_id	Devuelve las conversaciones en las que participe el usuario y el usuario pasado como parámetro.
conversations		Devuelve todas las conversaciones en las que participa el usuario que invoca a éste método.
del_favorite	_id, _type	Elimina de favoritos al elemento que se corresponde con el tipo e identificador enviados como parámetros y del usuario que lo invoca.
del_follows	_user	Elimina de seguidores al usuario enviado como parámetro.
follows		Devuelve un array de objetos User con todos los usuarios que se corresponden con los identificadores almacenados en el atributo follow_ids.
followers		Devuelve un array de objetos User con todos los usuarios cuyo identificador se corresponde con el almacenado en el atributo follower_ids.
id_to_s		Devuelve el identificador de User en formato String.
mail_welcome		Envío de email de bienvenida al usuario registrado.
num_conversations		Devuelve el número de conversaciones que están incluidas en tipo inbox (no archivadas).
num_events		Número de eventos creados por el usuario.
numFavorites		Número de elementos incluidos por el usuario en favoritos.
num_routes		Número de rutas creadas por el usuario.
photo		Devuelve la dirección del sistema donde está almacenada la foto del usuario.
objects_favorites	_type	Devuelve todos los favoritos del objeto _type del usuario que realiza la invocación del método.
event_favorites		Devuelve todos los favoritos de tipo Event del usuario.
route_favorites		Devuelve todos los favoritos de tipo Route del usuario.
set_count_messages		Actualiza un atributo de la colección almacenando el número de conversaciones, conversaciones archivadas y conversaciones sin leer.

## Comment

Representa a todos los comentarios que se realizan en el sistema, desde el realizado en el muro de un usuario a los realizados en los detalles de Eventos y Rutas. Gestionados por el propio usuario que crea el evento y por los usuarios de rol administrador.

### Atributos

Atributo	Tipo	Requerido	Automático	Contenido almacenado
_id	BSON:ObjectId	Sí	Sí	Identificador del comentario, único dentro de la colección Comments.
participant_ids	Array	No	No	Array que contiene a todos los participantes dentro del comentario, preparado para un futuro con la posibilidad de incluir respuestas en un comentario.
object_id	String	No	No	Indica el identificador al que está asociado el comentario si es que se realiza sobre algún elemento.
object_type	String	No	No	Contiene la clase del elemento sobre el que se realiza el comentario si es que se realiza sobre algún elemento.
text	String	Sí	No	Contiene el texto que almacena el comentario realizado por el usuario.
user_id	BSON:ObjectId	Sí	No	Almacena el identificador de usuario que ha creado la ruta.
created_at	DateTime	Sí	Sí	Hora de creación de la ruta indicada.
updated_at	DateTime	Sí	Sí	Hora de actualización de la ruta indicada.

### Métodos

Método	Parámetros	Descripción
search	_params = {}	Método de búsqueda de comentarios a partir de los parámetros recibidos. Devuelve un array de objetos Comment.
to_csv	_comments, _columns, options, params	Método para transformar un array de objetos Comment enviados como parámetro en formato CSV de salida.
as_json	options	Método sobreescrito para devolver un JSON con la información necesaria a mostrar en pantalla.
from		Devuelve un hash con los datos mínimos del User que ha escrito el comentario
id_to_s		Transformación del identificador en formato String
is_comment?		Booleano para saber si el objeto Comment no está asociado a un objeto de la clase Route o Event.

Método	Parámetros	Descripción
is_event?		Booleano para saber si el objeto Comment está asociado a un objeto de la clase Event.
is_route?		Booleano para saber si el objeto Comment está asociado a un objeto de la clase Route.
timestamp		Devuelve la fecha de creación del objeto Comment en formato Unix Timestamp.
type_comment		Devuelve la clase a la que está asociada el objeto Comment.
user_id_to_s		Devuelve el identificador del objeto User que ha creado el comentario en formato String.
get_object_comments	_object, _timestamp	Devuelve los objetos Comment que pertenezcan al objeto enviado como parámetro y que cumplan que su timestamp es superior que el enviado como parámetro.
user_wall	_user, _timestamp	Devuelve los objetos Comment que verá un objeto User de sus seguidores.
user_tweets	_user, _timestamp	Devuelve los objetos Comment pertenecientes al objeto User.
set_participant_ids		Almacena en el atributo participant_ids los identificadores de objeto User que participan en el objeto Comment.

## Conversation

Representa a las conversaciones existentes entre dos usuarios, contienen información de los identificadores asociados a cada usuario, así como la fecha de creación y del último mensaje que existe entre los usuarios participantes.

Son gestionadas por los propios usuarios participantes así como por los usuarios que disponen del rol administrador.

### Atributos

Atributo	Tipo	Requerido	Automático	Contenido almacenado
_id	BSON:ObjectId	Sí	Sí	Identificador de la conversación, es único para cada conversación.
deleted_by_users	Array	No	No	Contiene un conjunto de usuarios que han archivado la conversación.
from	String	No	Sí	Identificador de usuario que ha creado inicialmente la conversación
last_to	String	No	No	Identificador del usuario que recibe el último mensaje.

Atributo	Tipo	Requerido	Automático	Contenido almacenado
last_to_message_at	DateTime	No	No	Hora de la creación del último mensaje que se ha creado.
to	String	No	Sí	Identificador de usuario que recibe el primer mensaje de la conversación.
unread	Hash	No	No	Hash que contiene como clave a cada uno de los identificadores de usuario con el número de mensajes que no han leído todavía.
users	Array	No	Si	Array que contiene todos los identificadores de usuario participantes en la conversación.
created_at	DateTime	Sí	Sí	Hora de creación de la conversación.
updated_at	DateTime	Sí	Sí	Hora de actualización de la conversación.

## Métodos

Método	Parámetros	Descripción
search	_params = {}	Método de búsqueda de objetos Conversation a partir de los parámetros recibidos. Devuelve un array de objetos Conversation.
add_msg	msg, _from	Método para crear un nuevo objeto Message asociado al objeto Conversation almacenando el mensaje enviado y el usuario al que pertenece.
as_json	options	Método sobreescrito para devolver un JSON con la información necesaria a mostrar en pantalla.
delete_by_user	_user	Añade la conversación del objeto User enviado como parámetro a archivados, para ello se añade el identificador del objeto User dentro del atributo <i>deleted_by_users</i> .
id_to_s		Transformación del identificador en formato String
last_message		Devuelve el último objeto Message asociado al objeto Conversation.
mark_as_read_by	_user	Marca el objeto Conversation como leído por el objeto User enviado como parámetro.
msgs		Devuelve un array de objetos Message asociados al objeto Conversation en formato JSON.
user_from		Devuelve un hash con la información necesaria asociada al objeto User que inicia la conversación.
user_to		Devuelve un hash con la información necesaria asociada al objeto User que recibe la conversación.
fix_users		Método que hace recuento de las conversaciones de los objetos User.

Método	Parámetros	Descripción
set_count_user_messages		Recorre los objetos User almacenados en el atributo <i>users</i> y vuelve a almacenar su información.
set_unreads_messages		Setter inicial de conversación marcando el número de mensajes no leídos a cero.
set_users		Setter inicial de conversación añadiendo los identificadores de los objetos User participantes de la conversación.

## Message

Representa a los mensajes que existen dentro de cada conversación. Son elementos embebidos dentro del propio objeto Conversation y disponen un registro del usuario que ha creado el mensaje y la hora del mismo.

Con la creación de cada mensaje se ejecuta una actualización automática que modifica el valor almacenado en el objeto Conversation al que pertenece para indicar la hora del último mensaje creado.

### Atributos

Atributo	Tipo	Requerido	Automático	Contenido almacenado
_id	BSON:ObjectId	Sí	Sí	Identificador del mensaje, es único para cada mensaje escrito entre usuarios.
from	String	Sí	No	Identificador de usuario que ha creado el mensaje.
text	String	Sí	No	Texto del mensaje que ha escrito el usuario.
conversation_id	BSON:ObjectId	Sí	No	Identificador de la conversación al que está asociado el mensaje.
created_at	DateTime	Sí	Sí	Hora de creación del mensaje escrito por el usuario.
updated_at	DateTime	Sí	Sí	Hora de actualización del mensaje escrito por el usuario.

### Métodos

Método	Parámetros	Descripción
set_html_message		Parseo del texto para ajustarlo a ciertos caracteres HTML.

## Event

Representa a los eventos que se almacenan en el sistema y contienen toda la información relativa al evento como su nombre, descripción, fecha del evento y localización. Son gestionadas por los propios usuarios que las crean así como por los usuarios de rol administrador.

Disponen a su vez de relaciones con la clase User (usuario que crea el evento) y con la clase Comment ya que pueden contener comentarios asociados a un evento.

### Atributos

Atributo	Tipo	Requerido	Automático	Contenido almacenado
_id	BSON:ObjectId	Sí	Sí	Identificador del evento, es único para cada evento creado en el sistema.
address	String	Sí	No	Dirección en la que se celebra el evento, almacenada en formato texto con la dirección completa.
allow_comments	Boolean	No	No	Indicador de si el evento acepta comentarios o no.
description	String	Sí	No	Descripción completa del evento, cualquier texto que quiera añadir el usuario que crea el evento.
event_type	Integer	Sí	No	Tipo de evento que creamos, se corresponde si es para BMX, Carretera, Ciclocross, etc
init_date	DateTime	Sí	No	Fecha en la que se celebra el evento.
last_visit_at	DateTime	No	No	Última visita que ha recibido el evento, es un valor estadístico.
location	Array	Sí	No	Almacena en formato Array la longitud y latitud en la que se sitúa el evento, está asociado a la dirección almacenada en el atributo address.
name	String	Sí	No	Nombre que se le asocia al evento.
private	Boolean	No	No	Privacidad del evento, indica si es un evento privado (no visible) o evento público (visible)
static_image	String	No	No	Es un valor que almacena la URL que se codifica una imagen estática de Google con la posición indicada en el campo location.
summary	String	No	No	Almacena el resumen del evento, un par de frases indicativas del mismo.

Atributo	Tipo	Requerido	Automático	Contenido almacenado
url	String	No	No	Si el evento dispone de una URL asociada se almacena en este atributo.
user_id	BSON:ObjectId	Sí	No	Almacena el identificador del objeto User que ha creado el evento.
created_at	DateTime	Sí	Sí	Hora de creación del evento indicado.
updated_at	DateTime	Sí	Sí	Hora de actualización del evento indicado.

## Métodos

Método	Parámetros	Descripción
search	_params = {}	Método de búsqueda de objetos Event a partir de los parámetros recibidos. Devuelve un array de objetos Event.
to_csv	_events, _columns, options, params	Método para transformar un array de objetos Event enviados como parámetro en formato CSV de salida.
as_json	options	Método sobreescrito para devolver un JSON con la información necesaria a mostrar en pantalla.
get_type		Devuelve el tipo de evento en formato String.
ico_event_type		Devuelve el ícono asociado al tipo de evento del objeto Event.
id_to_s		Devuelve el identificador del objeto Event en formato String.
locale_init_date		Devuelve la fecha del evento traducida al idioma del navegador del usuario.
is_commentable		Devuelve un booleano informando si el objeto Event es comentable o no por el resto de usuarios.
is_private		Devuelve un booleano informando si el objeto Event es privado o no al resto de usuarios.
lat		Devuelve la latitud almacenada en el atributo <i>location</i> del objeto Event.
lng		Devuelve la longitud almacenada en el atributo <i>location</i> del objeto Event.
marker		Devuelve un hash con la latitud / longitud almacenada en el atributo location.
update_last_visit_at		Actualización de la última visita realizada al objeto Event.
static_image		Devuelve el string con la URL de Google Maps que visualizará un mapa estático con la localización del objeto Event.
user_id_to_s		Devuelve el identificador del objeto User que ha creado el evento.
type_events		Devuelve un array con los tipos de evento disponibles.

Método	Parámetros	Descripción
set_descripción_event_type		Devuelve la descripción relacionada al tipo de evento del objeto Event.
set_icon_event_type	_route	Devuelve la ruta física del ícono que se mostrará asociado al tipo de evento.
set_location		Setter para almacenar en el atributo <i>location</i> la latitud y longitud seleccionada.

## Favorite

Representa a los elementos que un usuario ha almacenado dentro de su listado de favoritos.

Es una tabla polimórfica donde se almacenan el identificador del objeto asociado y el tipo del objeto almacenado así como el identificador de usuario que ha almacenado información en dicha clase.

### Atributos

Atributo	Tipo	Requerido	Automático	Contenido almacenado
_id	BSON:ObjectId	Sí	Sí	Identificador del objeto Favorite almacenado en la aplicación.
user_id	BSON:ObjectId	Sí	No	Almacena el identificador del objeto User que ha creado el objeto Favorite.
object_type	String	Sí	No	Almacena la clase asociada al objeto que el usuario almacena como favorito.
object_id	String	Sí	No	Almacena el identificador asociado al objeto que el usuario almacena como favorito.
created_at	DateTime	Sí	Sí	Hora de creación del evento indicado.
updated_at	DateTime	Sí	Sí	Hora de actualización del evento indicado.

## Route

Representa a las rutas almacenadas en un sistema. Son gestionadas por los propios usuarios que las crean así como por los usuarios de rol administrador. Se relaciona con la clase Comment ya que pueden contener comentarios asociados en su detalle y con la clase User ya que un objeto Route pertenece a un User que lo crea.

### Atributos

Atributo	Tipo	Requerido	Automático	Contenido almacenado
_id	BSON:ObjectId	Sí	Sí	Identificador de la ruta, es único para cada ruta creada en el sistema.
allow_comments	Boolean	No	No	Indicador de si la ruta acepta comentarios o no.
description	String	Sí	No	Descripción completa de la ruta, cualquier texto que quiera añadir el usuario que crea la ruta.
difficulty	Integer	Sí	No	Dificultad considerada por el usuario que crea la ruta.
distance	Float	No	No	Distancia de la ruta realizada.
downloads	Integer	No	No	Número de descargas que se han realizado de la ruta seleccionada.
duration	Hash	No	No	Duración de la ruta, se almacena en un hash con horas valor, minutos valor.
gpx	String	No	No	Valor del sistema donde se almacena el fichero GPX generado para cada ruta.
last_download_at	DateTime	No	No	Última descarga que ha recibido la ruta, un valor estadístico.
last_visit_at	DateTime	No	No	Última visita que ha recibido la ruta, es un valor estadístico.
location	Array	Sí	No	Punto inicial de la ruta, se almacena la longitud y latitud del primer punto.
name	String	Sí	No	Nombre que se le asocia a la ruta.
points	Array	Sí	No	Array con todos los puntos almacenados para la ruta.
private	Boolean	No	No	Privacidad de la ruta, para indicar si es visible o no.
route_type	String	Sí	No	Indica el tipo de ruta que se está creando, Carretera, BMX u otro cualquiera.
static_image	String	No	No	Es un valor que almacena la URL que se codifica una imagen estática de Google con la posición indicada en el campo location.
static_trackpoints	String	No	No	Almacena los puntos codificados de manera estática para no tener que realizar peticiones a Google cada vez que se muestre la ruta en el mapa.

Atributo	Tipo	Requerido	Automático	Contenido almacenado
summary	String	No	No	Almacena el resumen del evento, un par de frases indicativas del mismo.
url	String	No	No	Si el evento dispone de una URL asociada se almacena en este atributo.
user_id	BSON:ObjectId	Sí	No	Almacena el identificador de usuario que ha creado la ruta.
created_at	DateTime	Sí	Sí	Hora de creación de la ruta indicada.
updated_at	DateTime	Sí	Sí	Hora de actualización de la ruta indicada.

## Métodos

Método	Parámetros	Descripción
search	_params = {}	Método de búsqueda de objetos Route a partir de los parámetros recibidos. Devuelve un array de objetos Route.
to_csv	_events, _columns, options, params	Método para transformar un array de objetos Event enviados como parámetro en formato CSV de salida.
as_json	options	Método sobreescrito para devolver un JSON con la información necesaria a mostrar en pantalla.
first_point		Devuelve el primer punto de la ruta almacenada.
get_difficulty		Devuelve la dificultad asociada al objeto Route.
get_duration		Devuelve la duración del objeto Route en formato HH:MM:SS.
get_elevations		Devuelve las elevaciones que nos permitirán crear la gráfica en la vista del detalle del objeto Route.
get_route_type		Devuelve el tipo de ruta del objeto Route.
gpx		Genera el fichero de tipo GPX si es que no existe en el sistema previamente.
ico_route_type		Devuelve el ícono asociado al tipo de ruta del objeto Route.
id_to_s		Devuelve el identificador del objeto Route en formato String.
is_commentable		Devuelve un booleano indicando si el objeto Route permite comentarios por parte de otros usuarios o no.
is_private		Devuelve un booleano indicando si el objeto Route es privado o no.
last_point		Devuelve el punto final del objeto Route.
lat		Devuelve la latitud asociada al objeto Route y que está almacenada en el atributo <i>location</i> .

Método	Parámetros	Descripción
lng		Devuelve la longitud asociada al objeto Route y que está almacenada en el atributo <i>location</i> .
parse_gpx_file		Método que a partir de un fichero GPX lee todos su atributos y los almacena en el sistema.
set_location		Almacena la latitud y longitud del primer punto en el sistema en el atributo <i>location</i> .
set_static_points		Genera un array de puntos estáticos que se usarán para la generación de la ruta sobre un mapa.
static_image		Devuelve el string con la URL de Google Maps que visualizará un mapa estático con la localización del objeto Route.
update_last_download_at		Actualización de la última descarga realizada.
update_last_visit_at		Actualización de la última visita realizada.
user_id_to_s		Devuelve el identificador del objeto User que ha creado la ruta en formato String.
validate_gpx_file?		Validación que comprueba si el fichero GPX compartido es válido o no.
generate_gpx_file		Método que crea un fichero GPX válido a partir de la información almacenada por el usuario.
set_description_route_type		Devuelve el tipo de ruta en formato String.
set_icon_route_type		Devuelve el ícono asociado al tipo de ruta.
set_summary		El resumen se genera a partir de los primeros 80 caracteres de la descripción y se almacena en sistema.
split_duration		Divide el hash duration en horas, minutos y segundos.

## Config

Representa a las configuraciones almacenadas en el sistema. Son gestionadas por el usuario administrador y almacenarán valores como el remitente de los correos electrónicos, el receptor de los emails de contacto, etc.

## Atributos

Atributo	Tipo	Requerido	Automático	Contenido almacenado
_id	BSON:ObjectId	Sí	Sí	Identificador de la configuración.
key	String	Sí	No	Valor que identifica a la configuración usada.

Atributo	Tipo	Requerido	Automático	Contenido almacenado
value	String	Sí	No	Campo que controla la privacidad que el usuario desea disponer en la plataforma.
created_at	DateTime	Sí	Sí	Hora de creación del documento Config.
updated_at	DateTime	Sí	Sí	Hora de actualización del documento Config.

## Métodos

Método	Parámetros	Descripción
method_missing	id, args*	Devuelve el valor de la clave asociada al parámetro id.

## Contact

Representa a los objetos Contacto que los usuarios visualizan en formato formulario. No son almacenados en base de datos y su cometido es la de enviar un correo electrónico indicando las solicitudes de los usuarios.

## Atributos

Atributo	Tipo	Requerido	Automático	Contenido almacenado
comments	String	Sí	No	Comentario que realiza el usuario
email	String	Sí	No	Email del usuario.
name	String	Sí	No	Nombre del usuario.
subject	String	Sí	No	Asunto de la consulta.
terms_of_service	Boolean	Sí	No	Aceptación de condiciones.

## Métodos

Método	Parámetros	Descripción
initialize		Creación del objeto que se enviará por email.

## Import

Clase encargada de la importación de información a partir de la URL enviada por un usuario. En este punto se parseará el HTML o se obtendrá la información si hay disponible una API para el sitio web introducido y se devolverá la información final.

### Atributos

Atributo	Tipo	Requerido	Automático	Contenido almacenado
description	String	No	No	Descripción de la ruta.
difficulty	String	No	No	Dificultad de la ruta.
distance	String	No	No	Distancia de la ruta.
duration	String	No	No	Duración de la ruta.
location	String	No	No	Localización de la ruta, se obtendrá a partir del primer punto del atributo <i>points</i> .
name	String	No	No	Nombre de la ruta.
points	Array	No	No	Puntos asociados a la ruta.
url	String	No	No	Url desde la que se lee la información.

### Métodos

Método	Parámetros	Descripción
initialize	_url, get	Método desde donde se inicializa toda la lectura de información de la ruta almacenada en la URL indicada.  El parámetro get sirve para indicar si hay que leer o no la información, es de tipo booleano.
get_info		Lectura de la información, es una llamada a todos los métodos de los que se leen los datos.
parse_descripción		Obtiene la descripción de la ruta leída.
parse_difficulty		Obtiene la dificultad de la ruta leída.
parse_distance		Obtiene la distancia de la ruta leída.
parse_duration		Obtiene la duración de la ruta leída.
parse_name		Obtiene el nombre de la ruta leída.
parse_points		Obtiene los puntos de la ruta leída.

## 4.3. Comportamiento de las clases de análisis

### 4.3.1. Catálogo de eventos. Relación con los casos de uso.

#### Eventos de inserción

<b>Id</b>	<b>Nombre</b>	<b>Casos de Uso</b>	<b>Descripción</b>
I1	Alta de usuario	Alta de usuario	Un usuario no registrado crea un usuario para acceder a la plataforma.
I2	Seguir usuario	Seguir Usuario	Un usuario añade a los usuarios que sigue a otro usuario de la plataforma.
I3	Escribir comentario	Escribir comentario	Un usuario escribe un comentario en su muro o en un elemento indicado.
I4	Nueva conversación	Nueva conversación	Un usuario comienza una conversación con otro usuario.
I5	Escribir mensaje	Escribir mensaje	Un usuario escribe un mensaje en una conversación con otro usuario.
I6	Alta de ruta	Alta de Ruta Importación URL Alta de Ruta	Un usuario registrado crea una ruta nueva en el sistema
I7	Marcar ruta como favorita	Marcar ruta como favorita	Un usuario añade una ruta seleccionada a sus favoritas.
I8	Alta de evento	Alta de evento	Un usuario registrado crea un nuevo evento en el sistema.
I9	Marcar evento como favorito	Marcar evento como favorito.	Un usuario registrado añade un evento seleccionado a sus favoritos.

#### Eventos de modificación

<b>Id</b>	<b>Nombre</b>	<b>Casos de Uso</b>	<b>Descripción</b>
M1	Edición de usuario	Edición de usuario	Un usuario registrado y que ha accedido a la plataforma edita su información.
M2	Recordatorio de contraseña	Recordatorio de contraseña	Un usuario ha solicitado un recordatorio de contraseña y hace una modificación de la misma.
M3	Editar comentario	Editar comentario	Un usuario edita un comentario dentro de la plataforma.
M4	Marcar comentario como ofensivo	Marcar comentario como ofensivo.	Un usuario marca un comentario de otro usuario como ofensivo llegando un email al administrador de dicha solicitud.
M5	Archivar conversación	Archivar conversación	Un usuario archiva una conversación en la que estuviese participando.

<b>Id</b>	<b>Nombre</b>	<b>Casos de Uso</b>	<b>Descripción</b>
M6	Edición de ruta	Edición de ruta	Un usuario edita una ruta que hubiese creado previamente.
M7	Edición de evento	Edición de evento	Un usuario edita un evento que hubiese creado previamente.

## Eventos de borrado

<b>Id</b>	<b>Nombre</b>	<b>Casos de Uso</b>	<b>Descripción</b>
B1	Borrado de usuario	Borrado de usuario	Un usuario registrado desea eliminar su cuenta y borra su usuario de la plataforma.
B2	Dejar de seguir usuario	Dejar de seguir usuario	Un usuario registrado deja de seguir a otro usuario.
B3	Eliminar comentario	Eliminar comentario	Un usuario registrado hace un borrado de un comentario propio.
B4	Borrado de ruta	Borrado de ruta	Un usuario borra una ruta que hubiese creado previamente.
B5	Desmarcar ruta de favoritas	Desmarcar ruta de favoritas	Un usuario borra de sus favoritas una ruta seleccionada.
B6	Borrado de evento	Borrado de evento	Un usuario borra un evento que hubiese creado previamente.
B7	Desmarcar evento de favoritos	Desmarcar evento de favoritos	Un usuario borra de sus favoritos un evento seleccionado.

## Eventos de consulta

<b>Id</b>	<b>Nombre</b>	<b>Casos de Uso</b>	<b>Descripción</b>
C1	Mis seguidores	Mis seguidores	Un usuario visualiza los usuarios que le siguen en la plataforma.
C2	A quién sigo	A quién sigo	Un usuario visualiza los usuarios que está siguiendo en la plataforma.
C3	Buscar usuarios	Buscar usuarios	Un usuario registrado hace una búsqueda de otros usuarios dentro de la plataforma.
C4	Mi muro	Mi muro	Un usuario accede a su muro o al de otro usuario consultando los datos introducidos.
C5	Mis eventos	Mis eventos	Lista de eventos asociados al usuario.
C6	Mis rutas	Mis rutas	Lista de rutas asociadas al usuario.
C7	Mis conversaciones	Mis conversaciones	Lista de conversaciones asociadas al usuario.

<b>Id</b>	<b>Nombre</b>	<b>Casos de Uso</b>	<b>Descripción</b>
C8	Ver conversación	Ver conversación	Un usuario accede al detalle de una conversación.
C9	Buscar rutas	Buscar rutas	Un usuario hace una búsqueda de rutas a partir del filtro existente.
C10	Ver ruta	Ver ruta	Un usuario accede al detalle de una ruta.
C11	Buscar eventos	Buscar eventos	Un usuario hace una búsqueda de eventos a partir del filtro existente.
C12	Ver evento	Ver evento	Un usuario accede al detalle de un evento.

#### 4.3.2. Matriz Evento - Clase

##### Matriz Evento - Clase de inserción

Leyenda: I = Inserción

<b>Clave / Evento</b>	<b>I1</b>	<b>I2</b>	<b>I3</b>	<b>I4</b>	<b>I5</b>	<b>I6</b>	<b>I7</b>	<b>I8</b>	<b>I9</b>
<b>Usuarios</b>	I	I							
<b>Rutas</b>					I	I			
<b>Eventos</b>							I	I	
<b>Comentarios</b>			I						
<b>Conversaciones</b>				I	I				

##### Matriz Evento - Clase de modificación

Leyenda: M = Modificación

<b>Clave / Evento</b>	<b>M1</b>	<b>M2</b>	<b>M3</b>	<b>M4</b>	<b>M5</b>	<b>M6</b>	<b>M7</b>
<b>Usuarios</b>	M	M					
<b>Rutas</b>					M		
<b>Eventos</b>						M	
<b>Comentarios</b>			M	M			
<b>Conversaciones</b>					M		

## Matriz Evento - Clase de borrado

Leyenda: B = Borrado

Clave / Evento	B1	B2	B3	B4	B5	B6	B7
Usuarios	B	B					
Rutas				B	B		
Eventos						B	B
Comentarios			B				
Conversaciones							

## Matriz Evento - Clase de consulta

Leyenda: C = Consulta

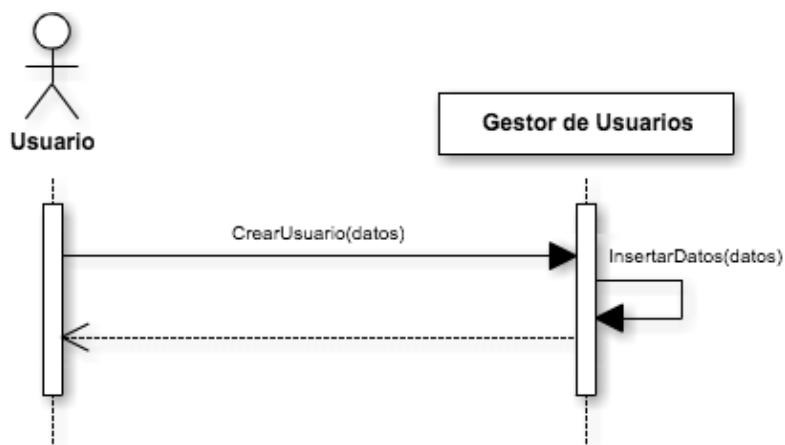
Clave / Evento	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12
Usuarios	C	C	C	C	C	C	C					
Rutas									C	C		
Eventos											C	C
Comentarios												
Conversaciones								C				

## 4.4. Diagramas de secuencia

A continuación se mostrarán los diagramas de secuencia creados a partir de los casos de uso documentados previamente.

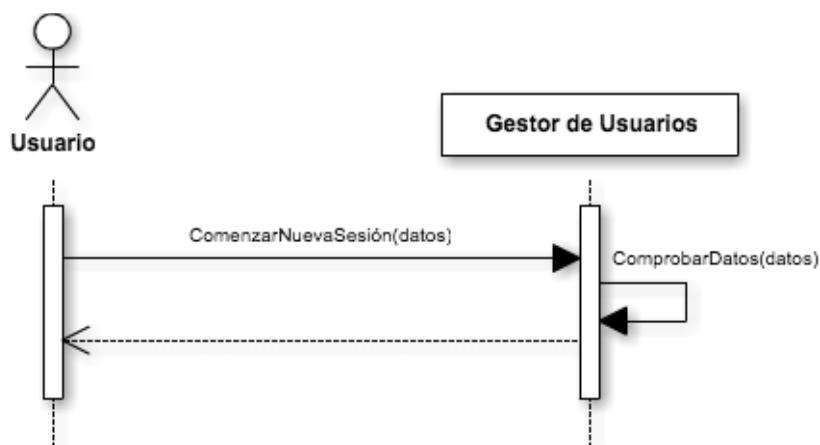
### Diagrama de secuencia Alta de usuario

El usuario solicita información al gestor de usuarios para comprobar si cumple las validaciones necesarias en el sistema. En caso de que todos los pasos sean correctos el gestor de usuarios almacenará la información en base de datos y devolverá la ejecución nuevamente al usuario.



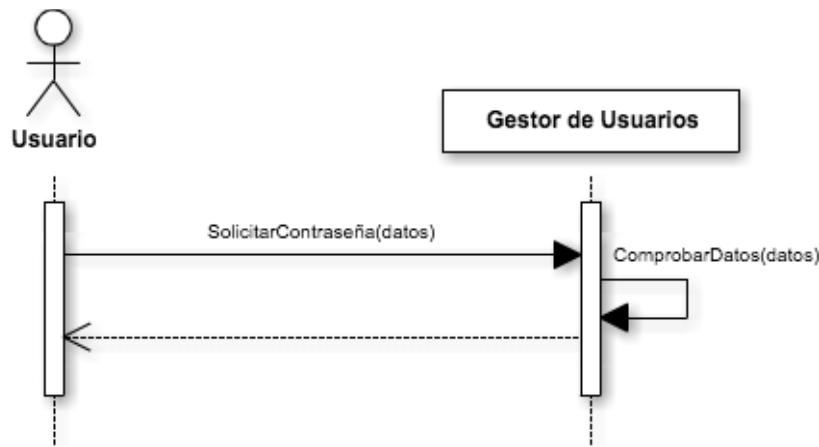
### Diagrama de secuencia Acceso a plataforma

El usuario solicita al gestor de usuarios la comprobación de datos que se envía a través del formulario de acceso a la plataforma para comprobar si se le asigna una sesión de acceso o se le indica el error producido.



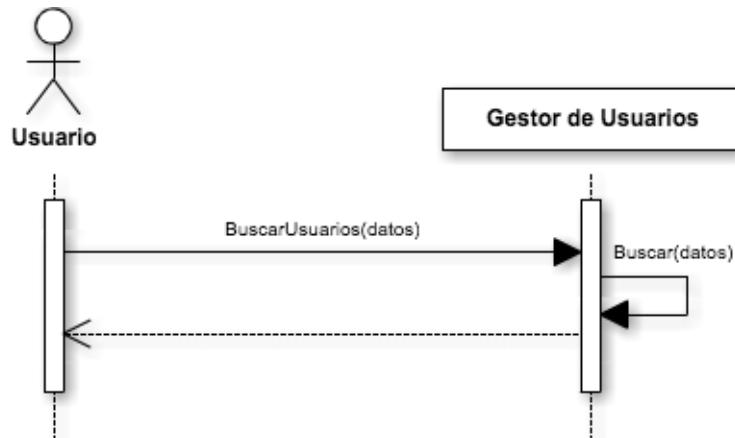
## Diagrama de secuencia Solicitar Contraseña

El usuario solicita al Gestor de usuarios la comprobación de los datos para en caso afirmativo enviar un correo para que se acredite y modifique su contraseña.



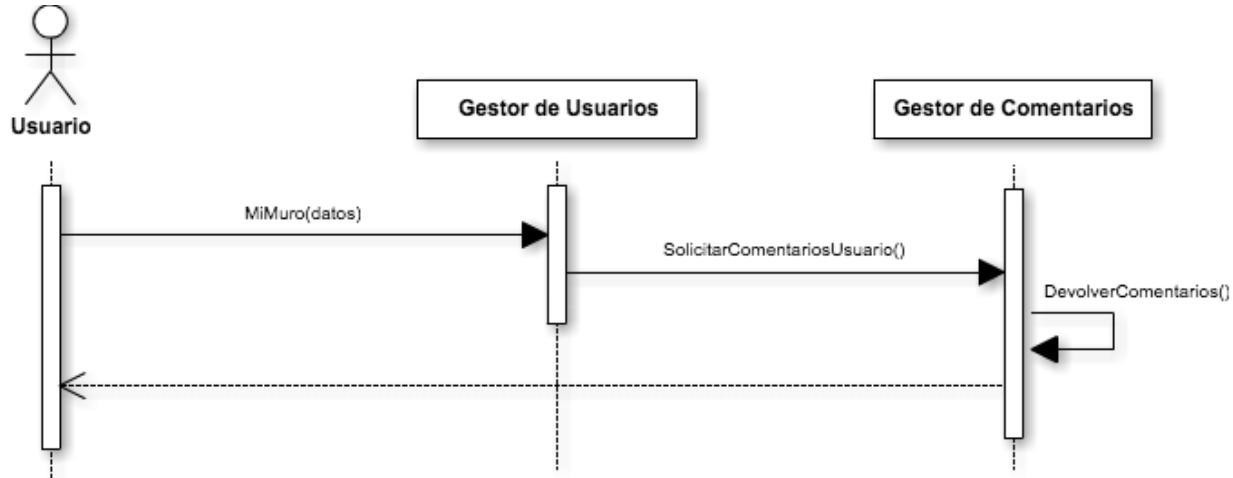
## Diagrama de secuencia Buscar Usuarios

El usuario realiza una solicitud de información de usuarios, para ello enviará la información a través del formulario y en el modelo se realizará una consulta parametrizada y devolverá la información encontrada al usuario.



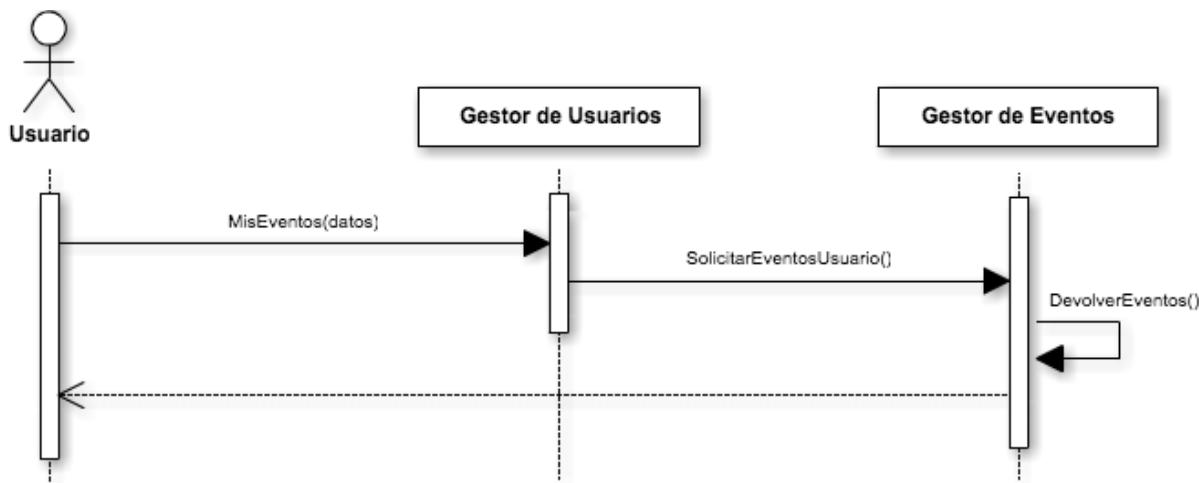
## Diagrama de secuencia Mi Muro

El usuario solicita al gestor de usuarios la información relativa al usuario al que está accediendo, una vez obtenida la información del usuario seleccionado se solicitan los comentarios asociados a dicho usuario y se devuelven en pantalla.



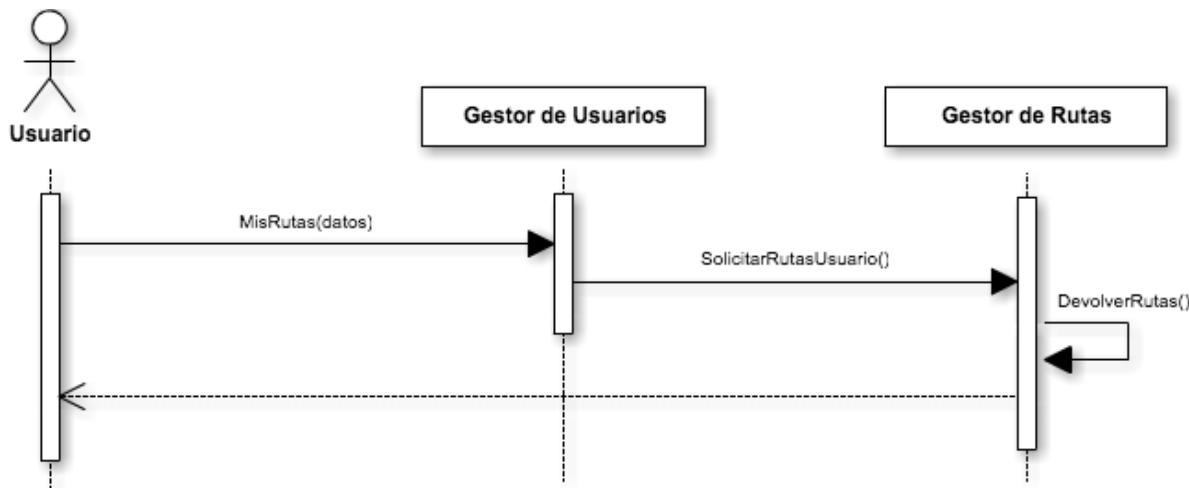
## Diagrama de secuencia Mis Eventos

El usuario solicita la información relativa al usuario al que se está accediendo, con dicha información se accederá al gestor de eventos para que devuelva toda la información relativa a los eventos creados por el usuario y mostrarlos en pantalla.



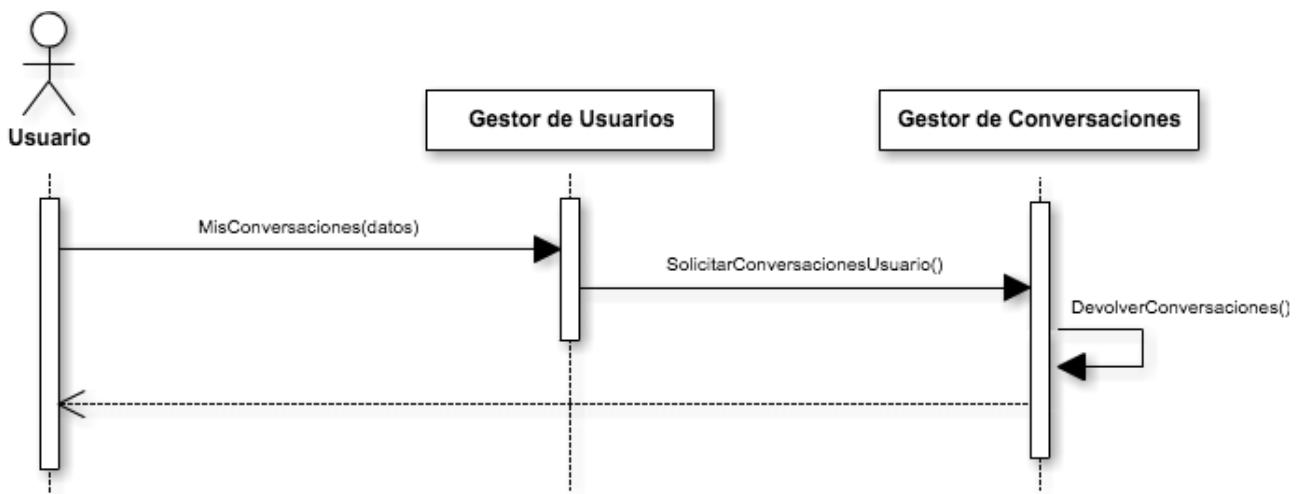
## Diagrama de secuencia Mis Rutas

El usuario solicita la información relativa al usuario al que se está accediendo, con dicha información se accederá al gestor de rutas para que devuelva toda la información relativa a las rutas creadas por el usuario y mostrarlas en pantalla.



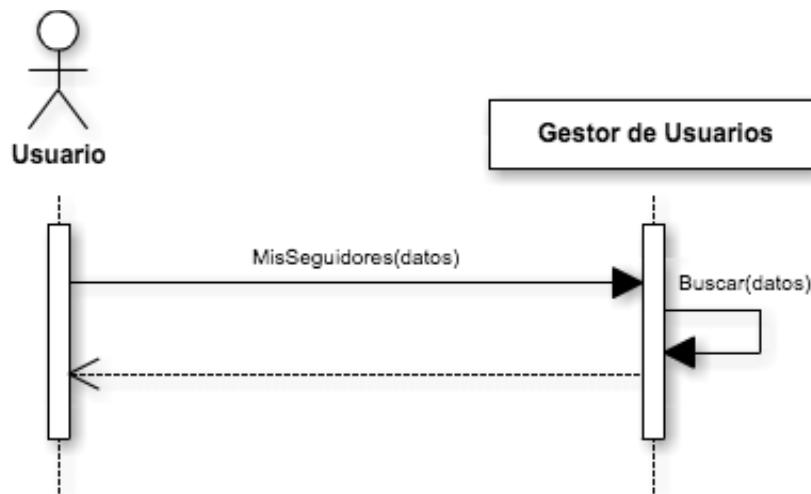
## Diagrama de secuencia Mis Conversaciones

El usuario solicita su información y a continuación al Gestor de Conversaciones la información relativa a todas las conversaciones en las que participa.



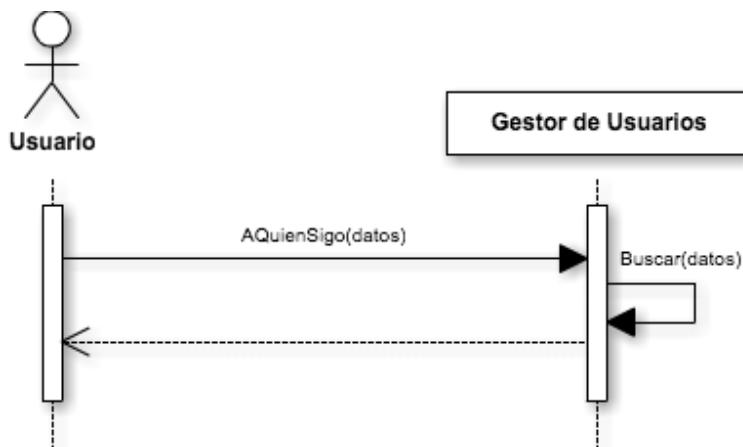
## Diagrama de secuencia Mis Seguidores

El usuario solicita al gestor de usuarios información relativa a todos los usuarios que tiene almacenados como seguidores. Se devolverán al usuario impresos en pantalla.



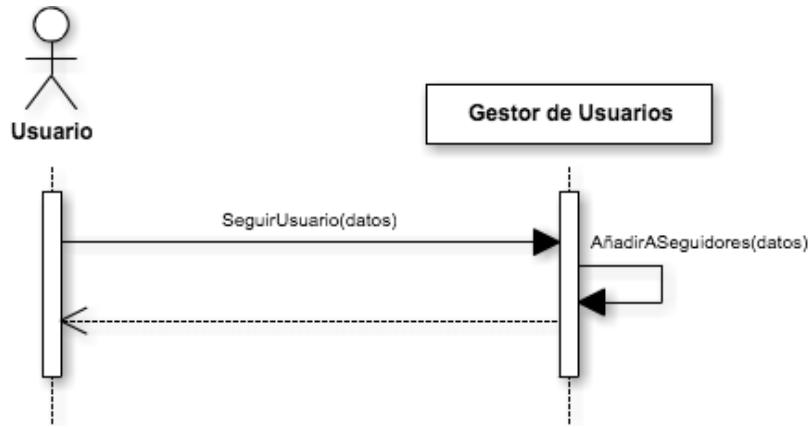
## Diagrama de secuencia A quién sigo

El usuario solicita información relativa a los usuarios que sigue al gestor de usuarios y se hará la solicitud al modelo para recuperar la información a mostrar.



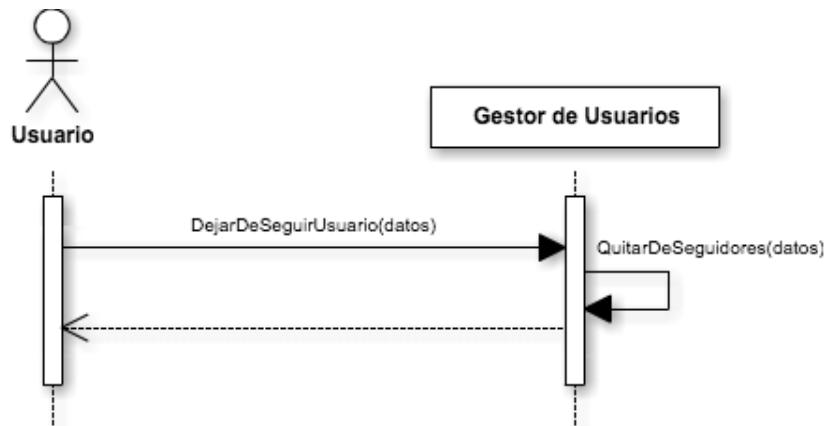
### Diagrama de secuencia Seguir Usuario

El usuario hace una solicitud al gestor de usuarios para comprobar la existencia del usuario y ejecuta la inserción en el sistema para añadir al usuario elegido dentro de la gente que sigue el usuario inicial.



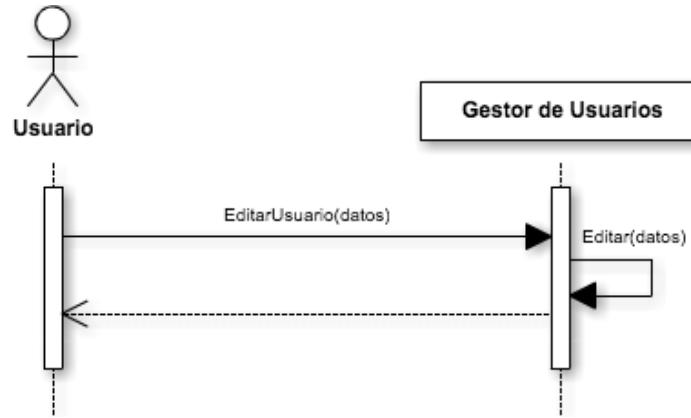
### Diagrama de secuencia Dejar de Seguir Usuario

El usuario hace una solicitud al gestor de usuarios para comprobar la existencia del usuario y ejecuta el borrado del usuario que sigue en el sistema.



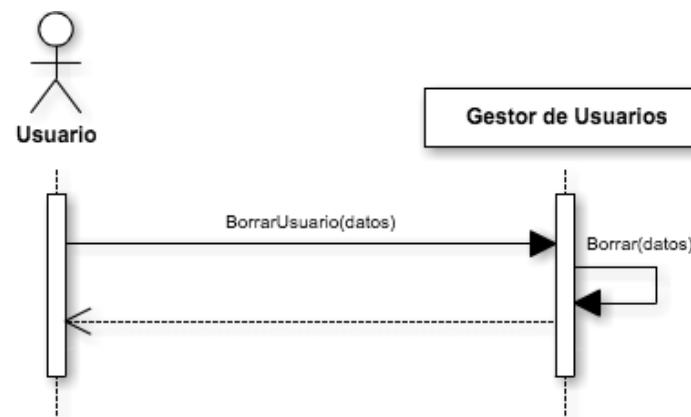
### Diagrama de secuencia Edición de Usuario

El usuario envía una solicitud al Gestor de usuarios enviándole información relativa a la modificación de datos del usuario y se realiza la edición del mismo.



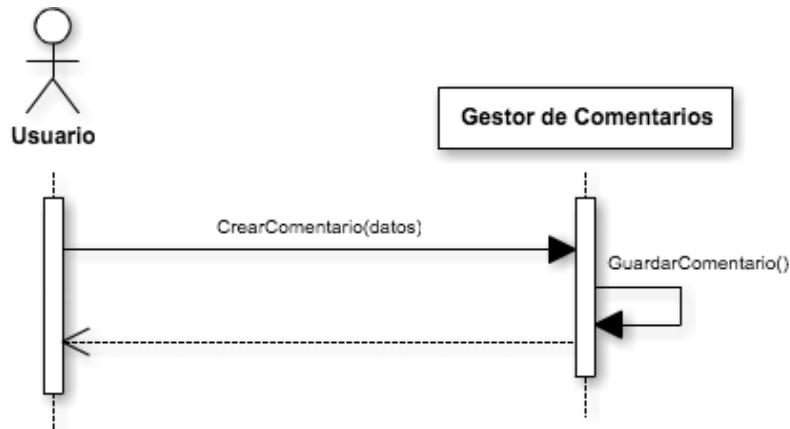
### Diagrama de secuencia Borrado de Usuario

El usuario envía la solicitud al Gestor de usuarios para ejecutar el borrado del mismo.



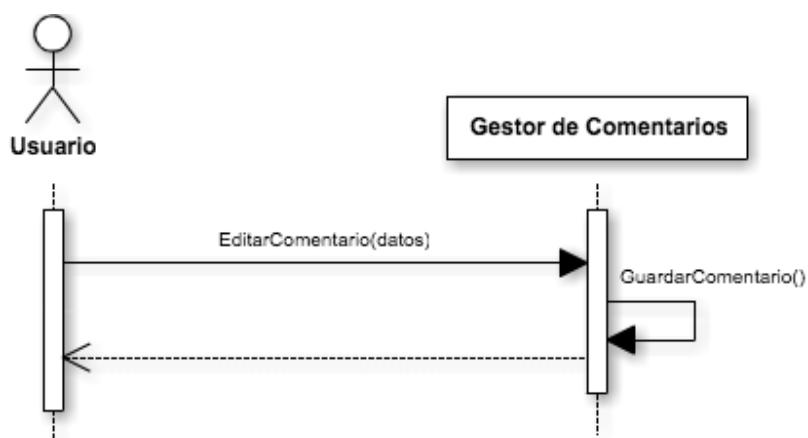
### Diagrama de secuencia Escribir Comentario

El usuario envía la información para la creación del comentario, el Gestor de Comentarios será el encargado de ejecutar la inserción del mismo en el sistema.



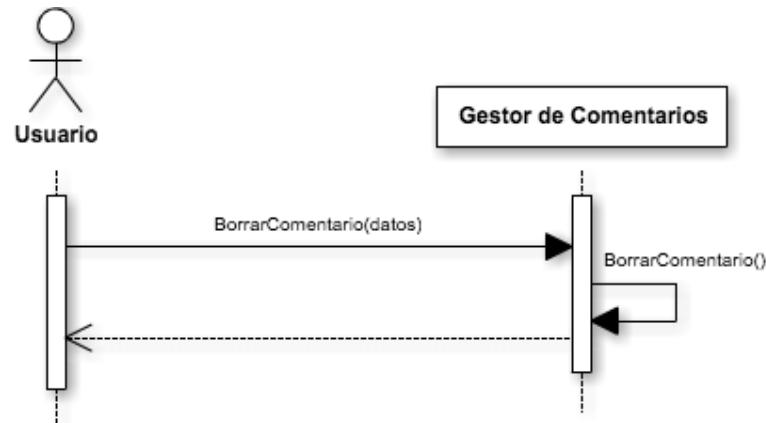
### Diagrama de secuencia Edición de Comentario

El usuario solicita al Gestor de comentarios la edición del mismo y ejecutará la instrucción de guardado del mismo en el sistema.



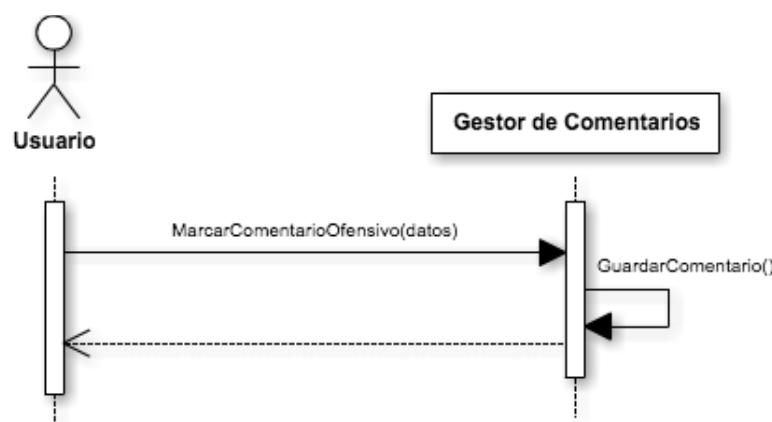
### Diagrama de secuencia Borrado de Comentario

El usuario solicita al Gestor de comentarios el borrado del mismo y se encarga de ejecutar la tarea de borrado de comentario.



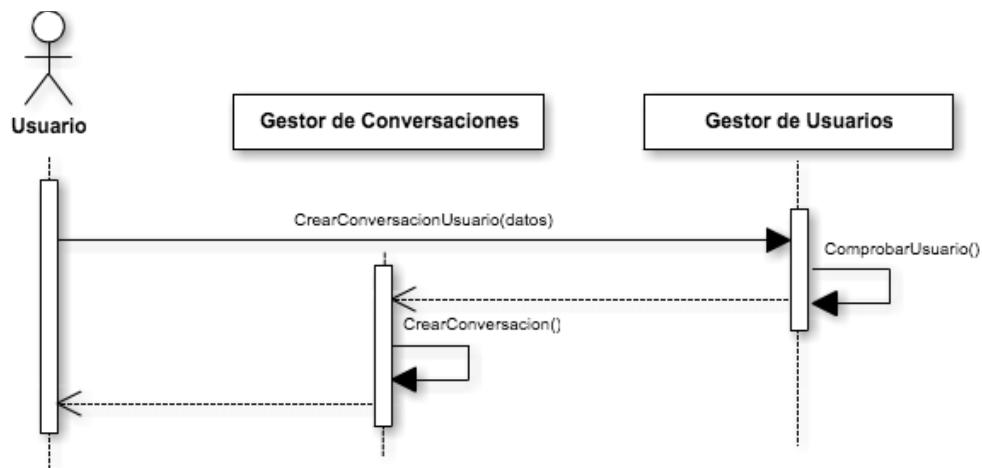
### Diagrama de secuencia Marcar comentario como ofensivo

El usuario solicita al Gestor de comentarios modificar el comentario seleccionado y ejecutar la tarea de modificación del comentario.



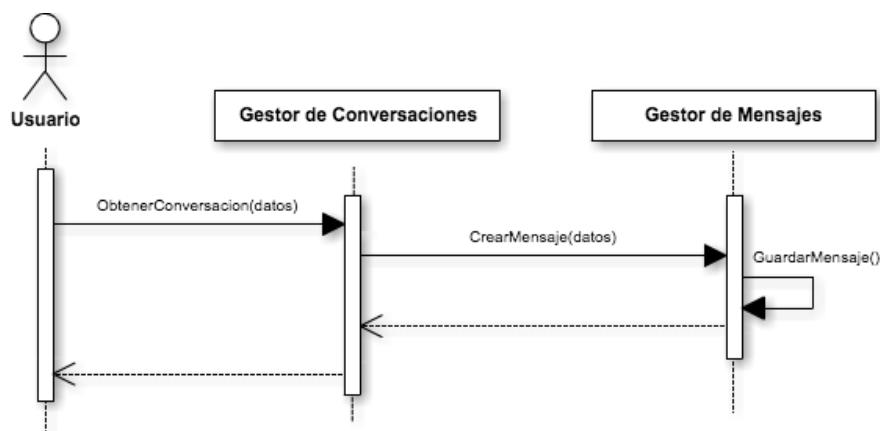
## Diagrama de secuencia Nueva Conversación

El usuario solicita la creación de una nueva conversación, inicialmente debemos acceder al Gestor de usuarios para comprobar que el usuario existe. A continuación el Gestor de conversaciones será el encargado de crear la conversación y devolver la ejecución al usuario inicial.



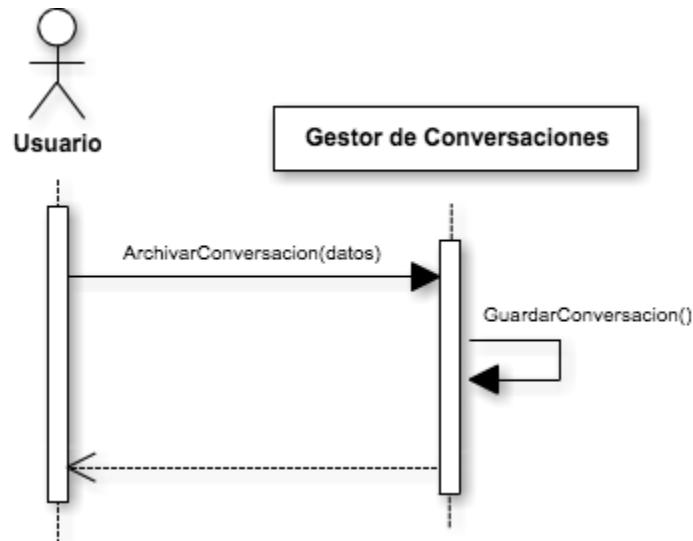
## Diagrama de secuencia Escribir Mensaje

El usuario solicita la creación de un mensaje, para ello accederemos inicialmente a la conversación elegida a través del Gestor de conversaciones. Una vez seleccionada pasamos al Gestor de mensajes que se encargará de almacenar la información y devolver el control al usuario.



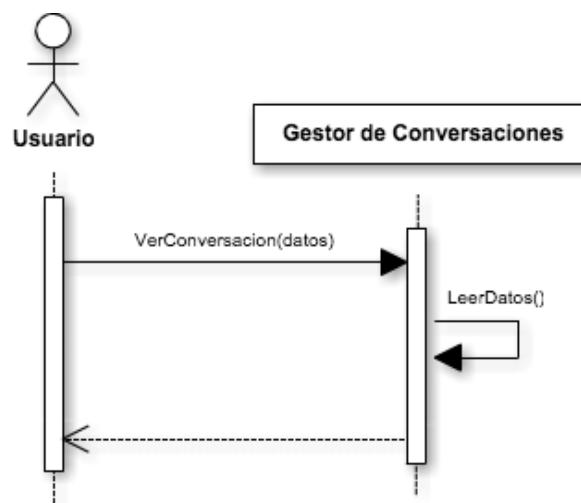
### Diagrama de secuencia Archivar Conversación

El usuario solicita la información relativa a la conversación al Gestor de conversaciones y se encarga de la modificación de la conversación devolviendo a posteriori el control al usuario.



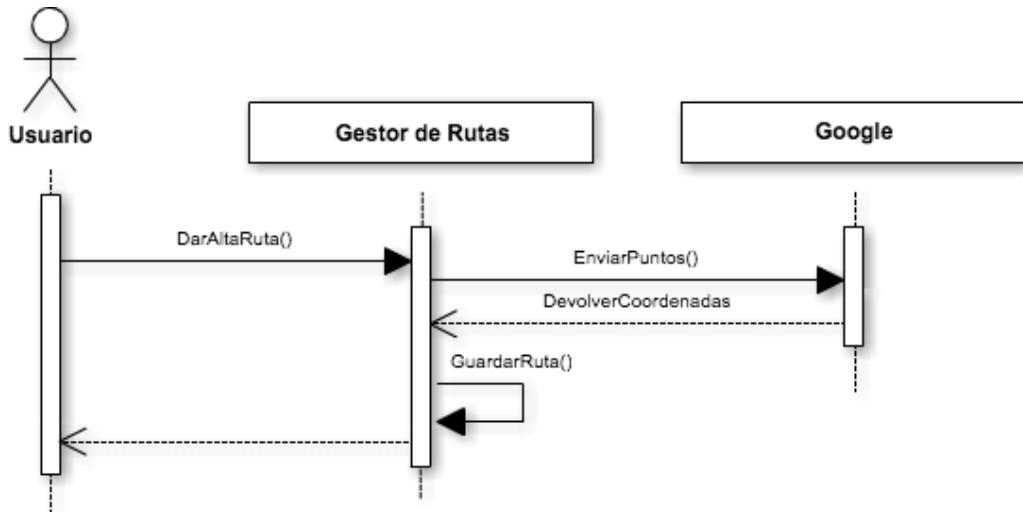
### Diagrama de secuencia Ver Conversación

El usuario solicita información relativa a la conversación al Gestor de conversaciones, lee la información y le devuelve el control al usuario.



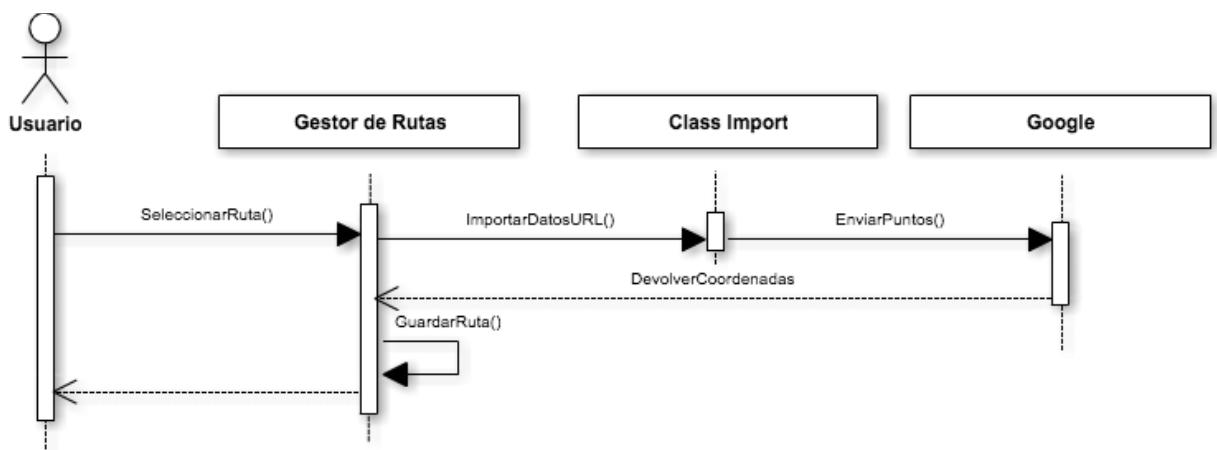
## Diagrama de secuencia Alta de Ruta

El usuario solicita la creación de una nueva ruta. El gestor de rutas toma el control al llegarle la información relativa a la misma, una vez con toda la información se hace una petición al servicio web de Google para devolver toda la información relativa a los puntos enviados y así obtener tanto las posiciones como la elevación de cada uno de ellos y devuelve toda la información tratada al Gestor de rutas que completará la ejecución.



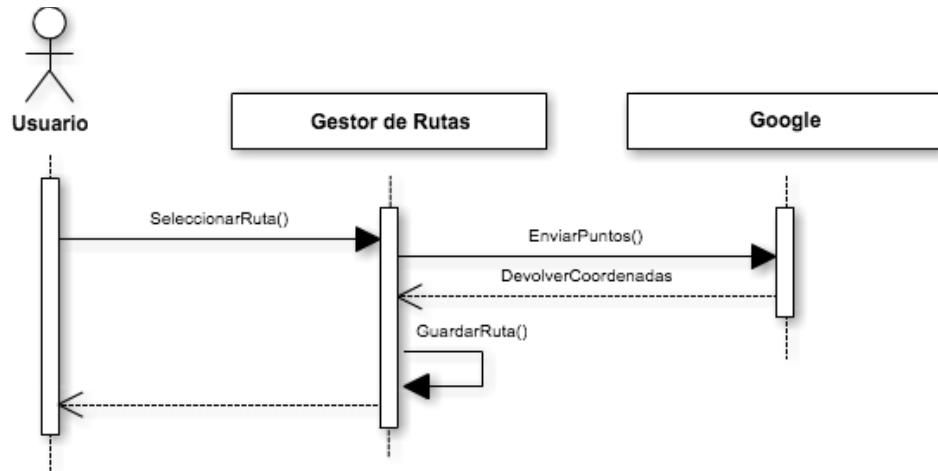
## Diagrama de secuencia Alta de Ruta Importación por URL

La llamada inicial es controlada por el Gestor de Rutas que enviará la información a la clase Import encargada de filtrar la información, a continuación enviará los puntos obtenidos a Google para obtener la latitud, longitud y elevación de cada punto y finalmente devolverá toda la información unida y se guardará la ruta en el Gestor de rutas.



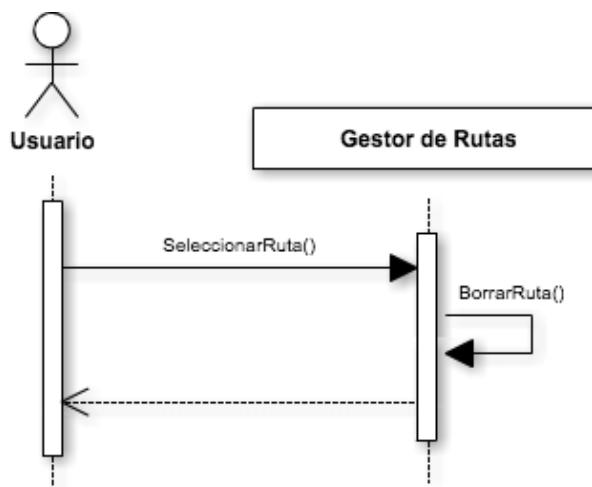
## Diagrama de secuencia Edición de Ruta

El usuario para editar la ruta realiza una llamada al Gestor de Rutas y tras la modificación de puntos solicitará información a Google de cada uno de ellos y por último se ejecutará la acción de guardado.



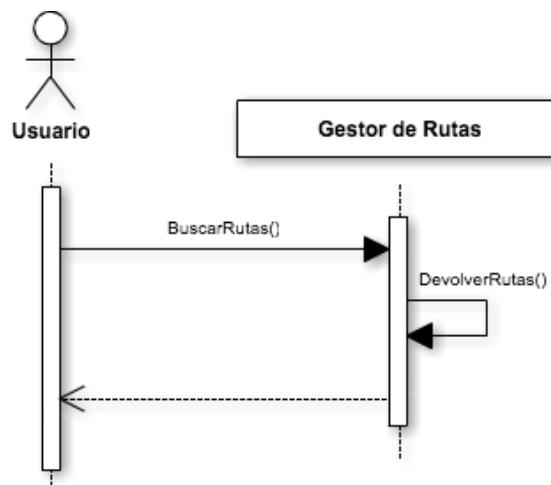
## Diagrama de secuencia Borrado de Ruta

El usuario solicita información de la ruta al Gestor de rutas y ejecuta la acción de borrado de la misma.



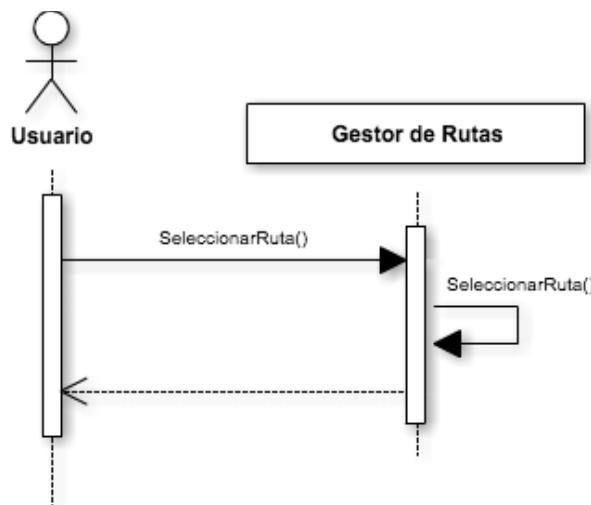
## Diagrama de secuencia Buscar Rutas

El usuario solicita información al Gestor de rutas de las que cumplan los parámetros solicitados.



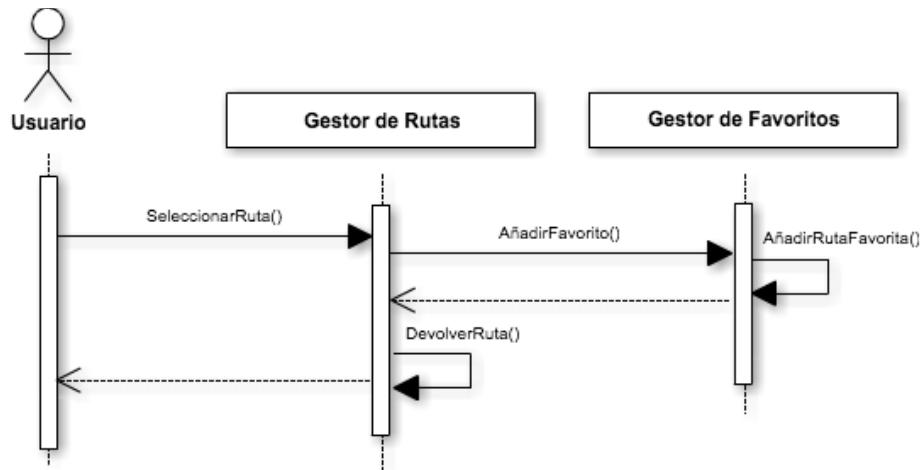
## Diagrama de secuencia Ver Ruta

El usuario solicita al Gestor de rutas información de la ruta seleccionada y le devuelve información de la misma al usuario.



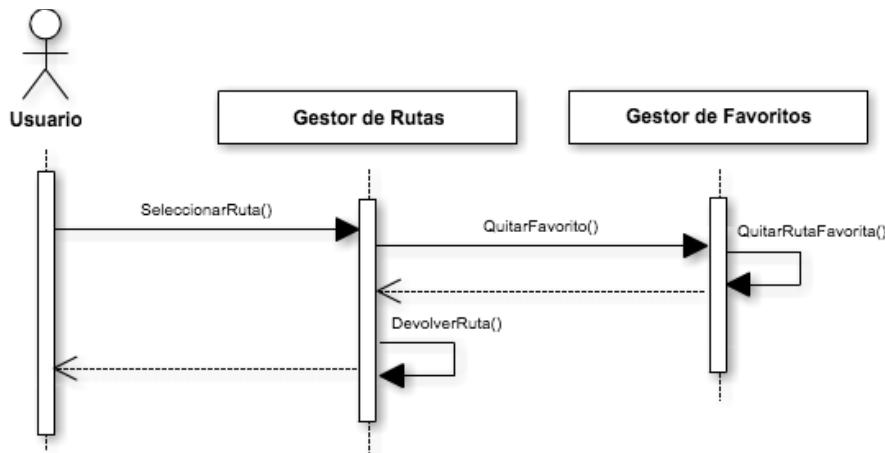
## Diagrama de secuencia Marcar Ruta como Favorita

El usuario solicita al Gestor de Rutas información relativa a la ruta seleccionada. Una vez que disponemos de dicha ruta se ejecuta el Gestor de favoritos que se encargará de almacenar en sistema la ruta seleccionada asociada al usuario inicial. Por último el usuario recibe información de la ruta elegida.



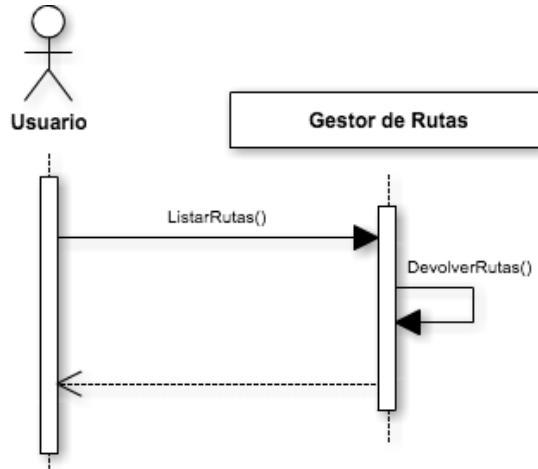
## Diagrama de secuencia Desmarcar Ruta como Favorita

El usuario solicita al Gestor de Rutas información relativa a la ruta seleccionada. Una vez que disponemos de dicha ruta se ejecuta el Gestor de favoritos que se encargará de eliminar la ruta seleccionada de las rutas favoritas asociadas al usuario.



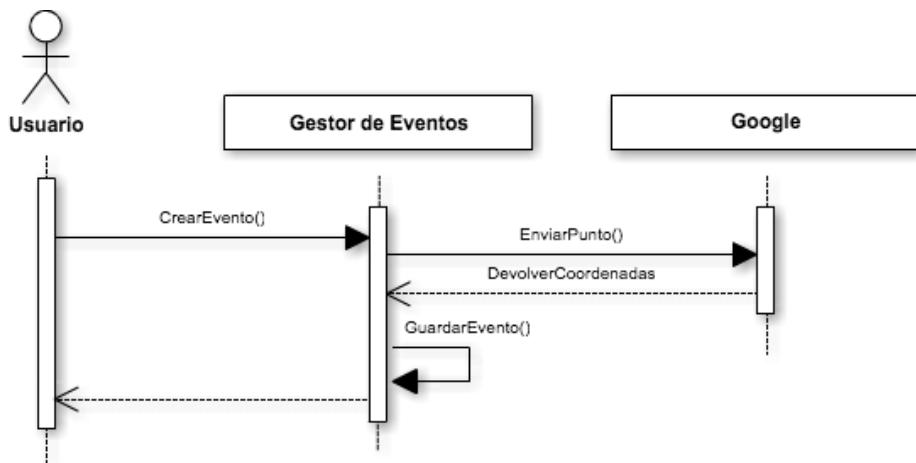
## Diagrama de secuencia Listar Rutas

El usuario solicita del Gestor de Rutas información de todas las rutas del sistema y las lista en pantalla.



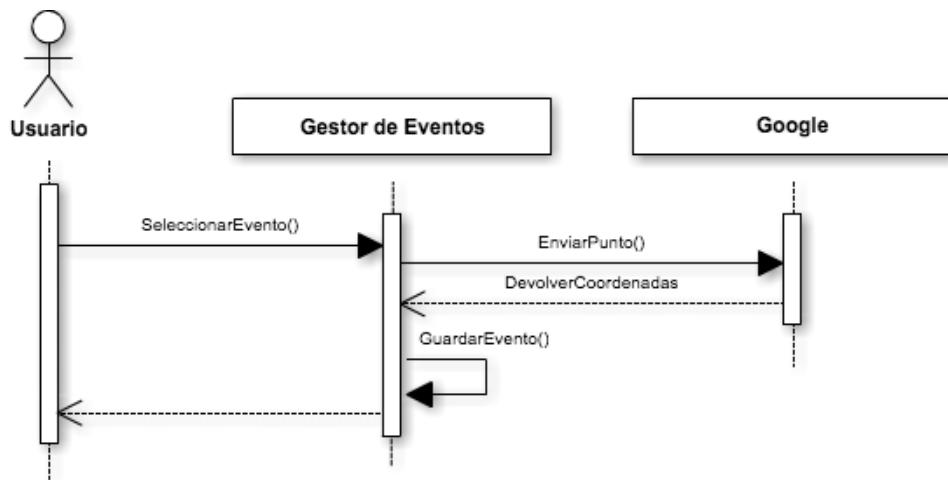
## Diagrama de secuencia Alta de Evento

El usuario solicita al Gestor de eventos la creación de un nuevo evento. Una vez enviada la información se solicita a Google información relativa al punto elegido y una vez que el Gestor de eventos recibe nuevamente el control se ejecuta el guardado del mismo en la aplicación.



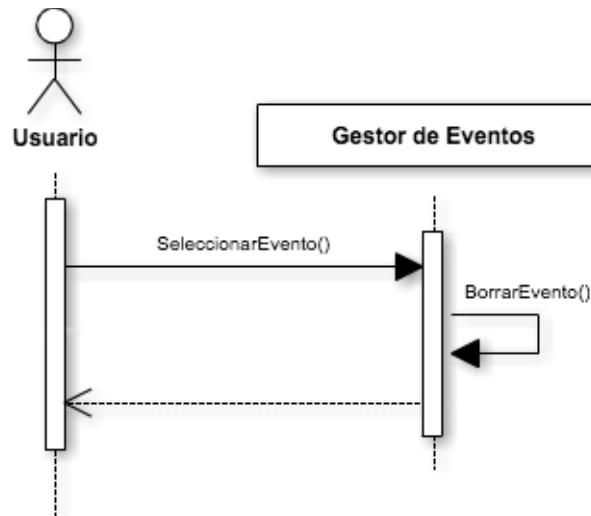
## Diagrama de secuencia Edición de Evento

El usuario solicita al Gestor de eventos la edición del evento elegido. Una vez enviada la información Google devolverá la información relativa al punto seleccionado y el Gestor de eventos cuando tenga nuevamente el control ejecutará la acción de guardado en el sistema.



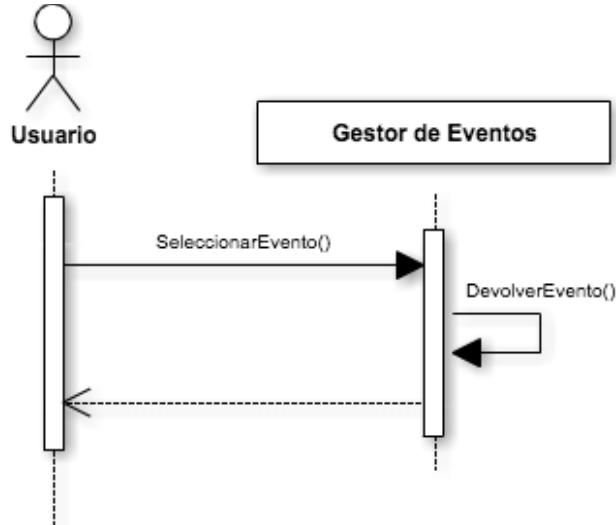
## Diagrama de secuencia Borrado de Evento

El usuario solicita al Gestor de eventos información relativa al mismo y ejecuta la acción de borrado del sistema devolviendo a continuación el control al usuario.



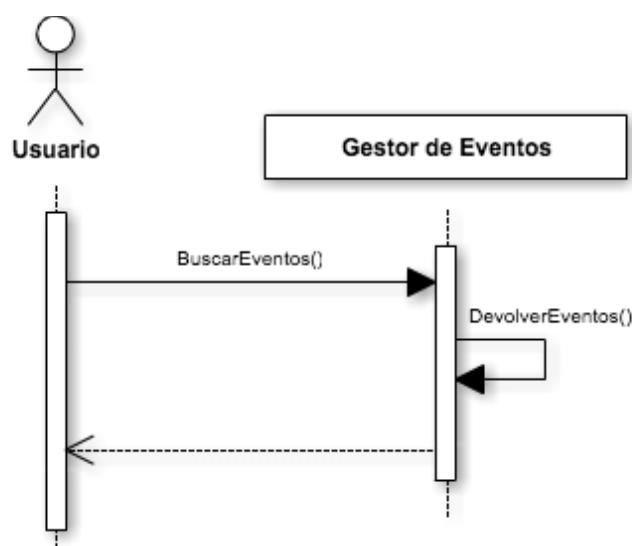
## Diagrama de secuencia Ver Evento

El usuario solicita al Gestor de eventos información relativa al evento elegido y se le devuelve el control al usuario con la información del evento elegido.



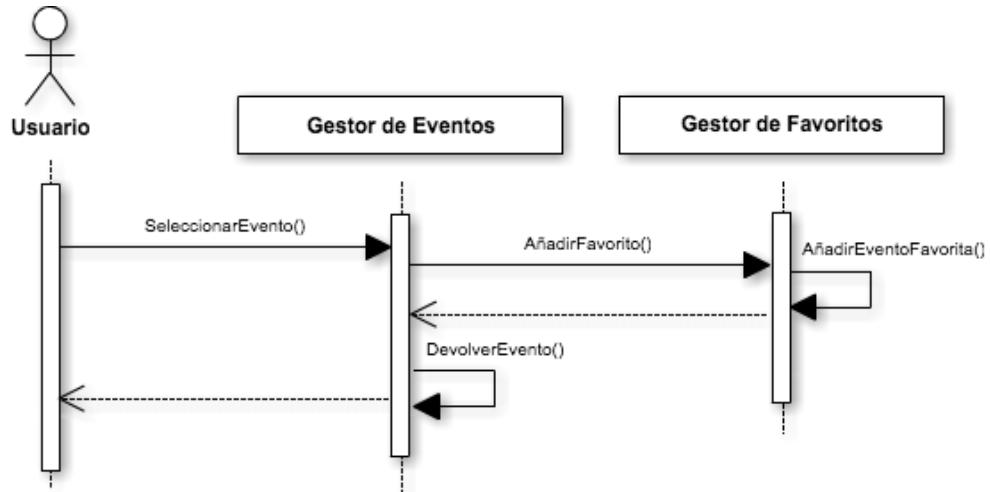
## Diagrama de secuencia Buscar Eventos

El usuario solicita al Gestor de eventos información relativa a los eventos que cumplen los parámetros solicitados por el usuario. Se le devolverá el control de la ejecución con un listado de los eventos que cumplen las condiciones seleccionadas.



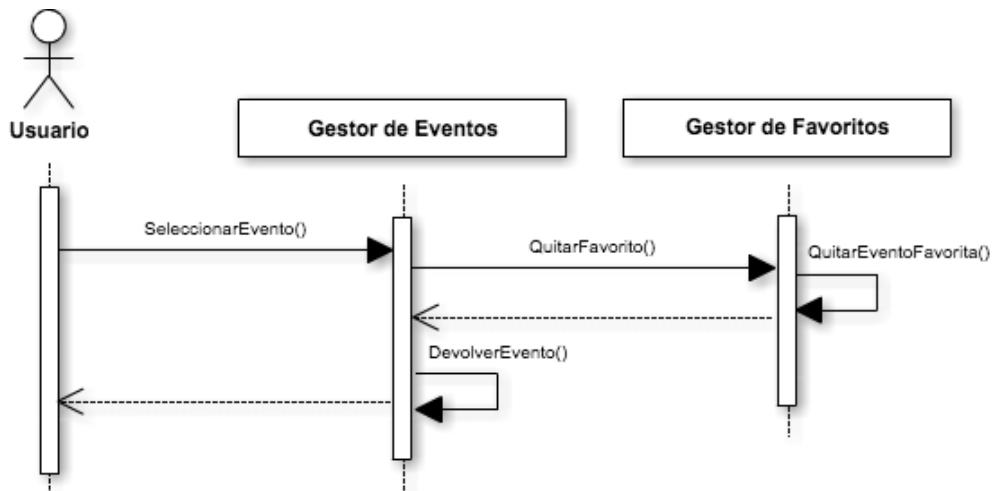
## Diagrama de secuencia Marcar Evento como Favorito

El usuario solicita al Gestor de eventos información relativa al evento seleccionado. Una vez que disponemos de dicho evento se ejecuta el Gestor de favoritos que se encargará de almacenar en sistema el evento seleccionado asociado al usuario que lo ejecuta. Por último el usuario recibe información del evento elegido.



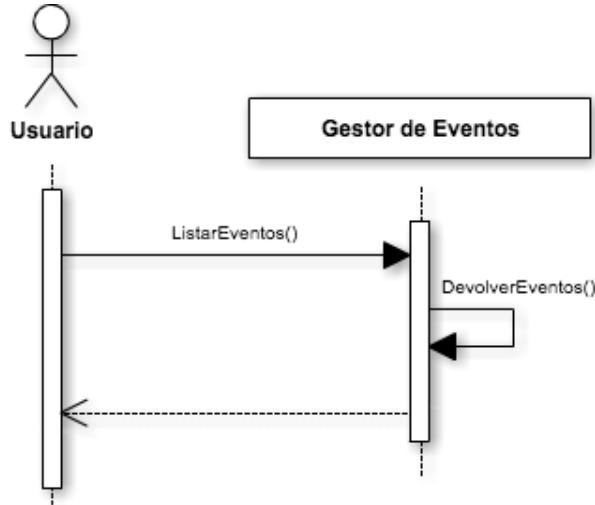
## Diagrama de secuencia Desmarcar Evento de Favoritos

El usuario solicita al Gestor de eventos información relativa al evento seleccionado. Una vez que disponemos de dicho evento se ejecuta el Gestor de favoritos que se encargará de eliminar el evento seleccionado de los eventos favoritos asociados al usuario.



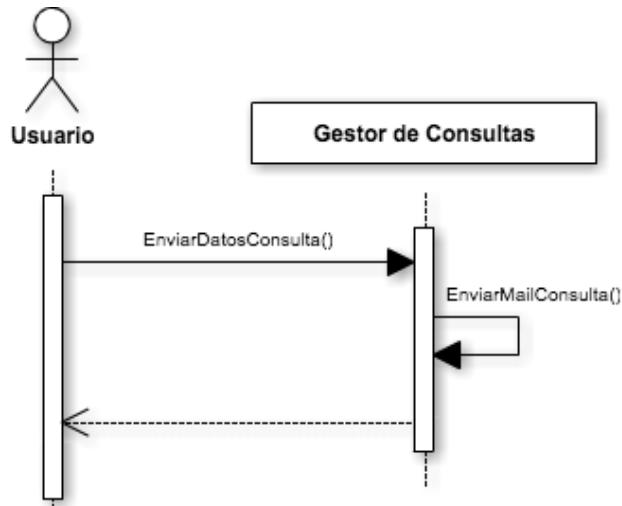
## Diagrama de secuencia Listar Eventos

El usuario solicita al Gestor de eventos un listado de todos los eventos disponibles.



## Diagrama de secuencia Enviar Consulta

El usuario solicita el envío de una consulta a través del Gestor de consultas. En el Gestor se realiza el envío del email que se realizará al usuario administrador del portal y se devolverá el control al usuario que ejecuta la acción inicial.



## 4.5. Diseño Físico de Datos

### 4.5.1. Estructura Física de la Base de Datos

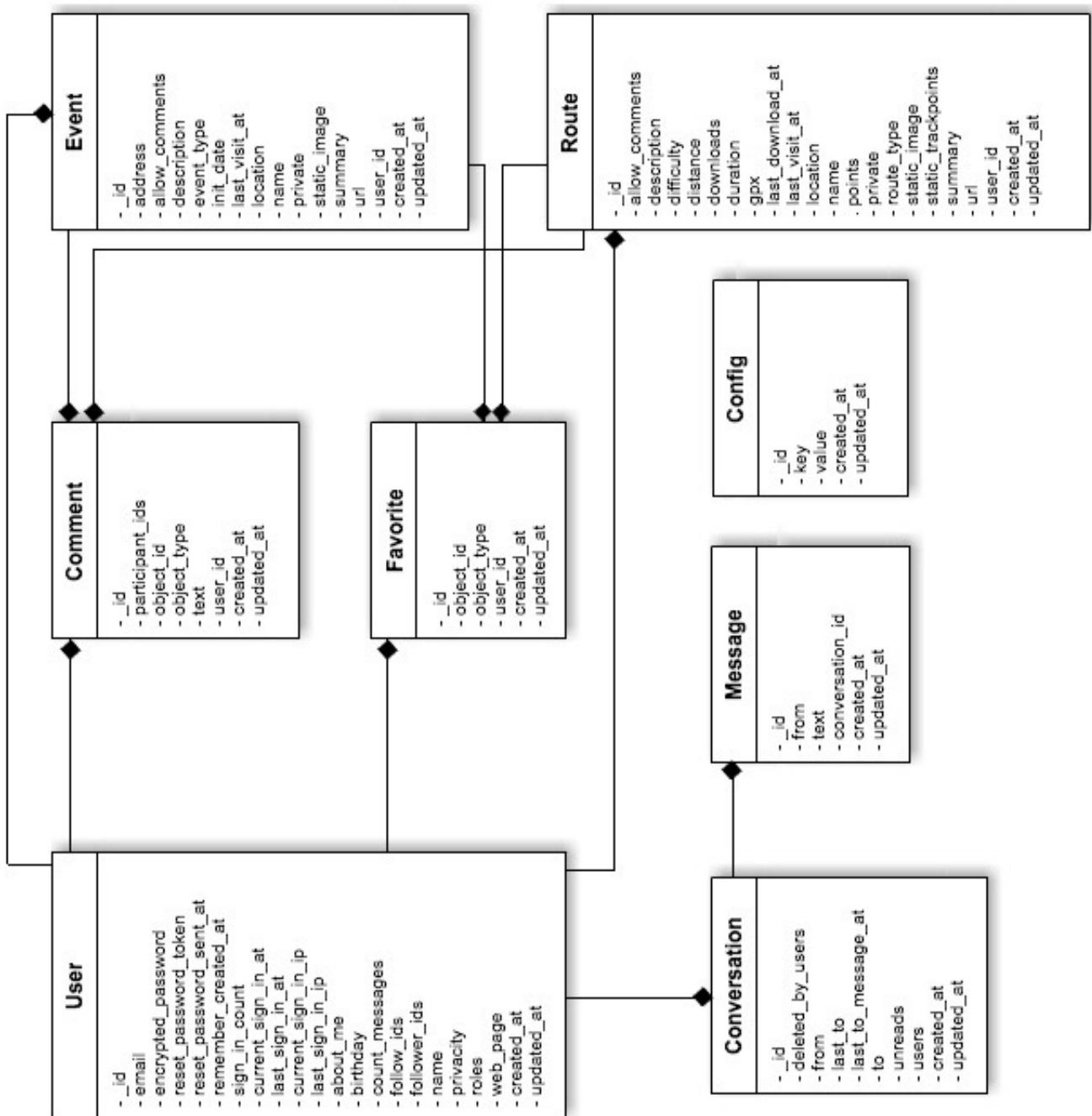
El sistema de gestión de base de datos utilizado es MongoDB. Aunque es un sistema de base de datos NoSQL de tipo documental en la que por defecto no existen relaciones entre colecciones se simulará mediante el uso de atributos que harán el funcionamiento de claves externas entre las mismas.

Para modelar los datos no será necesario crear las colecciones de manera manual, sino que el propio framework Ruby on Rails junto al plugin Mongoid usado en el proyecto crearán automáticamente al iniciar la aplicación toda la estructura de la base de datos almacenada en los distintos modelos de la aplicación ya que proporcionan un método de traducción automático de modelos a colecciones.

De este modo, sólo será necesario crear un modelo por cada colección indicando en el mismo los atributos de los que va a disponer de dicho modelo con sus características propias y el framework se encargará de la conexión con la colección y sus atributos.

En capítulos posteriores se explicará de manera detallada como crear los modelos en Ruby on Rails y los atributos relacionados con la colección a crear.

#### 4.5.2. Estructura de Colecciones





## 5. Pruebas del Sistema

---

En este apartado vamos a desarrollar un plan de pruebas para verificar el correcto funcionamiento de la aplicación.

Puesto que sería imposible probar todas y cada una de las funcionalidades de la aplicación, el proceso de pruebas se centrará en los módulos críticos, es decir, en aquellos puntos que se considera que pueden dar lugar a errores. especialmente, aquellos casos de uso en los que el usuario juega un papel especialmente activo, ya que ha de introducir datos para ejecutarlos, hecho que hace a dichos módulos más propensos a la aparición de errores frente a otros en los que el usuario no maneja la información directamente. Así mismo, es importante detectar los errores en casos de uso que impliquen el borrado de información, para evitar perdidas accidentales de datos.

A todo esto, hay que añadir que, las tareas de inserción y modificación de información, se realizan de manera muy similar en cada uno de los componentes, por tanto, se entiende que se han realizado las mismas pruebas para todos ellos, pero sólo queda reflejado un caso de uso para no extender demasiado el documento.

Para la realización de estas pruebas se considera la aplicación como un todo, con la que solamente se interacciona a través de sus entradas y salidas. Es decir, se realizarán pruebas exhaustivas de las mismas, que evalúen el comportamiento de la aplicación de acuerdo con las especificaciones dadas, no siendo necesario el código de la aplicación para ejecutar este tipo de pruebas, este tipo de pruebas se denominan “Pruebas de Caja Negra”.

Para realizar las pruebas se utiliza la técnica de Clases de Equivalencia, que consiste en dividir las entradas en grupos donde se pueden encontrar errores. Estos grupos se conocen como:

- **Clases de equivalencia válida:** son aquellas para las cuales, la aplicación debería dar una respuesta válida, es decir, las que cumplen con las especificaciones de la aplicación.
- **Clases de equivalencia inválida:** son aquellas para las cuales, la aplicación da una respuesta errónea, es decir, las que no cumplen con las especificaciones de la aplicación.

Para este tipo de pruebas también hay que tener en cuenta las salidas obtenidas. Además del tipo de pruebas ya comentadas, se analizará la usabilidad y la accesibilidad de la aplicación web teniendo en cuenta las pautas del W3C.

## 5.1. Pruebas de Caja Negra

### 5.1.1. Conjunto de pruebas “Acceso de usuarios registrados”

La motivación para escoger las pruebas sobre el caso de uso “Acceso de usuarios registrado” es que no se puede permitir el acceso al área privada de la aplicación a cualquier persona que lo intente.

Clases de equivalencia para “Acceso de usuarios registrados”

Condición	Clases válidas	Clases inválidas
Existencia del usuario en la base de datos	V1. El email y la contraseña introducidos son correctos.	I1. El email introducido no es correcto I2. El email introducido es correcto, pero no lo es la contraseña.
Obligatoriedad de los campos	V2. Introducir email y contraseña	I3. No introducir el email I4. No introducir la contraseña I5. No introducir ninguno de los dos

Casos de prueba para “Acceso de usuarios registrados”

Para reflejar los datos de entrada de los diferentes casos de prueba de este módulo se especifica a continuación una serie de datos de entrada genéricos que servirán para todos los casos de prueba, aunque en el desarrollo de cada caso de prueba se sustituya algún dato para escenificar el caso de prueba en concreto.

Email:	<u>alejandro.lopez@uniovi.es</u>
Contraseña:	voyenbici

A partir de los datos genéricos indicados anteriormente se realizarán los diferentes casos de prueba especificados.

<b>Número de caso de prueba</b>	2.1.2.1
<b>Objetivo</b>	Probar el caso de acceso al muro de usuario una vez que rellena correctamente email y contraseña.
<b>Clases de equivalencia</b>	V1, V2
<b>Datos de entrada</b>	Datos genéricos
<b>Salida esperada</b>	El usuario accede a su muro de contenido donde puede crear nueva información o revisar la existente.
<b>Salida obtenida</b>	La esperada

<b>Número de caso de prueba</b>	2.1.2.2
<b>Objetivo</b>	Probar el caso de acceso al muro de usuario cuando se rellenan los dos campos pero el usuario es incorrecto
<b>Clases de equivalencia</b>	I1
<b>Datos de entrada</b>	Datos genéricos, cambiando el email por uno incorrecto <b><u>alejan@uniovi.es</u></b>
<b>Salida esperada</b>	Mensaje de error identificando que “El usuario y / o la contraseña introducidos son incorrectos”
<b>Salida obtenida</b>	La esperada (Ilustración I1)

<b>Número de caso de prueba</b>	2.1.2.3
<b>Objetivo</b>	Probar el caso de acceso al muro de usuario cuando se rellenan los dos campos pero la contraseña es incorrecta
<b>Clases de equivalencia</b>	I2
<b>Datos de entrada</b>	Datos genéricos, cambiando la contraseña por una incorrecta <b>veb</b>
<b>Salida esperada</b>	Mensaje de error identificando que “El usuario y / o la contraseña introducidos son incorrectos”
<b>Salida obtenida</b>	La esperada (Ilustración I1)

<b>Número de caso de prueba</b>	2.1.2.4
<b>Objetivo</b>	Probar el caso de acceso al muro de usuario cuando no se rellena alguno de los dos campos
<b>Clases de equivalencia</b>	I3, I4
<b>Datos de entrada</b>	Sin información en alguno de los dos campos
<b>Salida esperada</b>	Mensaje de error identificando que “El usuario y / o la contraseña introducidos son incorrectos”
<b>Salida obtenida</b>	La esperada (Ilustración I1)

<b>Número de caso de prueba</b>	2.1.2.5
<b>Objetivo</b>	Probar el caso de acceso al muro de usuario cuando no se rellena ninguno de los dos campos
<b>Clases de equivalencia</b>	I5
<b>Datos de entrada</b>	Sin información en ninguno de los dos campos
<b>Salida esperada</b>	Mensaje de error identificando que “El usuario y / o la contraseña introducidos son incorrectos”
<b>Salida obtenida</b>	La esperada (Ilustración I1)

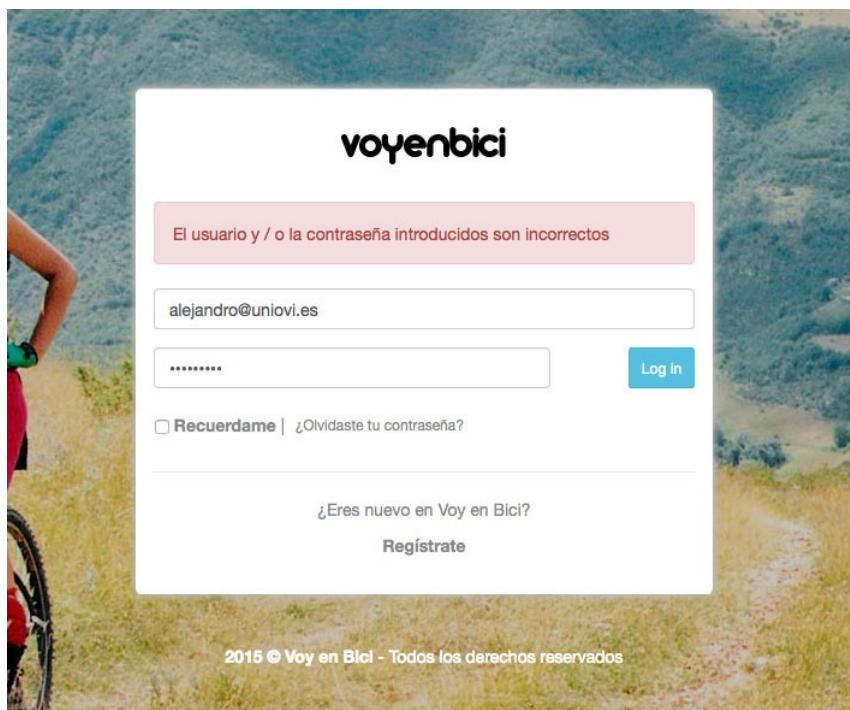


ILUSTRACIÓN I1. ERROR EN USUARIO Y / O CONTRASEÑA

### 5.1.2. Conjunto de pruebas “Registro de nuevo usuario”

El proceso de inserción de nuevo usuario requiere que un usuario no registrado rellene los datos obligatorios del formulario y que estos cumplan ciertas condiciones por lo que es un punto de la aplicación propenso de errores.

## Clases de equivalencia para “Registro de nuevo usuario”

Condición	Clases válidas	Clases inválidas
Existencia del usuario en la base de datos	V1. En la base de datos no existe ningún usuario registrado con el mismo email con el del usuario registrado que se intenta crear.	I1. En la base de datos existe un usuario cuyo email de usuario coincide con el del usuario registrado que se intenta crear.
Obligatoriedad de los campos	V2. El usuario introduce información en todos los campos obligatorios.	I2. El usuario registrado no introduce todos los campos obligatorios.
Formato de datos	V3. La contraseña tiene como mínimo seis caracteres  V4. La contraseña coincide exactamente igual que la confirmación de la contraseña.  V5. El email tiene un formato válido <u>xxx@xxx.xx</u> .	I3. El usuario registrado introduce una contraseña de cinco caracteres o menos.  I4. El usuario registrado introduce una confirmación de contraseña que no coincide con la contraseña.  I5. El usuario registrado introduce un email que no tiene un formato válido <u>xxx@xxx.xx</u> .

## Casos de prueba para “Registro de nuevo usuario”

Para reflejar los datos de entrada de los diferentes casos de prueba de este módulo se especifica a continuación una serie de datos de entrada genéricos que servirán para todos los casos de prueba, aunque en algún caso se puede sustituir alguno de ellos para alguna prueba concreta.

<b>Nombre:</b>	Alejandro López
<b>Email:</b>	<u>alejandro.lopez@uniovi.es</u>
<b>Contraseña:</b>	voyenbici
<b>Confirmación de contraseña:</b>	voyenbici

Las pruebas a realizar con los datos genéricos indicados serán las siguientes:

<b>Número de caso de prueba</b>	2.2.2.1
<b>Objetivo</b>	Caso de uso de insertar un usuario registrado cuando todos los campos han sido rellenados en el formato correcto y no existe en base de datos ningún otro usuario con los mismos datos.
<b>Clases de equivalencia</b>	V1, V2, V3, V4, V5
<b>Datos de entrada</b>	Datos genéricos
<b>Salida esperada</b>	El usuario se registra correctamente y accede a su muro de contenido donde puede crear nueva información o revisar la existente.
<b>Salida obtenida</b>	La esperada

<b>Número de caso de prueba</b>	2.2.2.2
<b>Objetivo</b>	Caso de uso de insertar un usuario registrado cuando todos los campos han sido rellenados en el formato correcto y existe en base de datos otro usuario con el mismo email.
<b>Clases de equivalencia</b>	I1
<b>Datos de entrada</b>	Datos genéricos cuando ya se ha realizado el caso de prueba 2.2.2.1 y por tanto ya existe en base de datos un usuario con el email <a href="mailto:alejandro.lopez@uniovi.es">alejandro.lopez@uniovi.es</a>
<b>Salida esperada</b>	El usuario recibe un mensaje de error indicándole que el email ya está siendo utilizado por otro usuario registrado.
<b>Salida obtenida</b>	La esperada (Ilustración I2)



ILUSTRACIÓN I2. MENSAJE INDICANDO QUE EMAIL YA ESTÁ EN USO.

<b>Número de caso de prueba</b>	2.2.2.3
<b>Objetivo</b>	Caso de uso de insertar un usuario registrado cuando el usuario deja algún campo obligatorio sin llenar.
<b>Clases de equivalencia</b>	I2
<b>Datos de entrada</b>	Datos genéricos dejando sin llenar información
<b>Salida esperada</b>	El usuario recibe un mensaje de error indicándole que campo o campos obligatorios no ha llenado.
<b>Salida obtenida</b>	La esperada (Ilustración I3)



ILUSTRACIÓN I3. USUARIO NO HA RELLENADO TODA LA INFORMACIÓN OBLIGATORIA.

<b>Número de caso de prueba</b>	2.2.2.4
<b>Objetivo</b>	Caso de uso de insertar un usuario registrado con una contraseña inferior a ocho caracteres
<b>Clases de equivalencia</b>	I3
<b>Datos de entrada</b>	Datos genéricos modificando la contraseña a <b>voyenbici</b>
<b>Salida esperada</b>	El usuario recibe un mensaje de error indicándole que la contraseña que ha introducido tiene menos de ocho caracteres.
<b>Salida obtenida</b>	La esperada (Ilustración I4)



ILUSTRACIÓN I4. LA CONTRASEÑA ES INFERIOR A 8 CARACTERES (MÍNIMA OBLIGATORIA).

<b>Número de caso de prueba</b>	2.2.2.5
<b>Objetivo</b>	Caso de uso de insertar un usuario registrado con una confirmación de contraseña no coincidente con la contraseña
<b>Clases de equivalencia</b>	I4
<b>Datos de entrada</b>	Datos genéricos modificando la confirmación de contraseña por <b>voyen</b>
<b>Salida esperada</b>	El usuario recibe un mensaje de error indicándole que la confirmación de contraseña no coincide con la contraseña que ha introducido.
<b>Salida obtenida</b>	La esperada (Ilustración I5)



ILUSTRACIÓN I5. LA CONFIRMACIÓN Y LA CONTRASEÑA NO COINCIDEN.

<b>Número de caso de prueba</b>	2.2.2.6
<b>Objetivo</b>	Caso de uso de insertar un usuario registrado con un email no válido.
<b>Clases de equivalencia</b>	I5
<b>Datos de entrada</b>	Datos genéricos modificando el email por alejandro.lopez@uniovi
<b>Salida esperada</b>	El usuario recibe un mensaje de error indicándole que el email que está introduciendo no tiene un formato válido.
<b>Salida obtenida</b>	La esperada (Ilustración I6)



ILUSTRACIÓN I6. EL EMAIL NO TIENE UN FORMATO VÁLIDO.

### 5.1.3. Conjunto de pruebas de “Insertar nuevo comentario”

El proceso de inserción de nuevo comentario requiere que un usuario registrado rellene los datos obligatorios del formulario de comentario por lo que es un punto de la aplicación propenso de errores.

#### Clases de equivalencia para “Insertar nuevo comentario”

Condición	Clases válidas	Clases inválidas
Obligatoriedad de los campos	V1. El usuario registrado introduce información en todos los campos obligatorios con el formato indicado.	I1. El usuario registrado no introduce todos los campos obligatorios.
Formato de datos	V2. El usuario introduce un comentario con un texto comprendido entre 1 y 256 caracteres	I2. El usuario registrado introduce un comentario con más de 256 caracteres.

#### Casos de prueba para “Insertar nuevo comentario”

Para reflejar los datos de entrada de los diferentes casos de prueba de este módulo se especifica a continuación una serie de datos de entrada genéricos que servirán para todos los casos de prueba, aunque en algún caso se puede sustituir alguno de ellos para alguna prueba concreta.

<b>Comentario:</b>	Este es un comentario de prueba para comprobar la validación de obligatoriedad de campos.
--------------------	---

Las pruebas a realizar con los datos genéricos aquí indicados serán las siguientes:

<b>Número de caso de prueba</b>	2.3.2.1
<b>Objetivo</b>	Caso de uso de insertar un comentario con todos los campos obligatorios con el formato indicado.
<b>Clases de equivalencia</b>	V1
<b>Datos de entrada</b>	Datos genéricos
<b>Salida esperada</b>	El usuario escribe un comentario en su muro o en el elemento sobre el que está creando el comentario.
<b>Salida obtenida</b>	La esperada

<b>Número de caso de prueba</b>	2.3.2.2
<b>Objetivo</b>	Caso de uso de insertar un comentario sin campos obligatorios llenados.
<b>Clases de equivalencia</b>	I1
<b>Datos de entrada</b>	Datos genéricos modificando el comentario e introduciendo un comentario sin texto.
<b>Salida esperada</b>	El usuario recibe un mensaje de error indicándole que el comentario no tiene texto y no se puede crear.
<b>Salida obtenida</b>	La esperada (Ilustración I7)

El comentario no puede estar en blanco

Escribe tu mensaje

Publicar

ILUSTRACIÓN I7. EL COMENTARIO NO SE PUEDE CREAR POR FALTA DE CAMPOS OBLIGATORIOS.

<b>Número de caso de prueba</b>	2.3.2.3
<b>Objetivo</b>	Caso de uso de insertar un comentario con formato no válido.
<b>Clases de equivalencia</b>	I2
<b>Datos de entrada</b>	Datos genéricos modificando el comentario e introduciendo un texto superior a 256 caracteres.
<b>Salida esperada</b>	El usuario recibe un mensaje de error indicándole que el comentario tiene más caracteres de los permitidos.
<b>Salida obtenida</b>	La esperada (Ilustración I8)

El comentario no puede ser superior a 256 caracteres

Hoy he disfrutado como una enana de la última ruta que he hecho, os recomiendo a todos que visitéis los Oscos en primavera :)Hoy he disfrutado como una enana de la última ruta que he hecho, os recomiendo a todos que

Te quedan -120 caracteres

Publicar

ILUSTRACIÓN I8. EL COMENTARIO NO SE PUEDE CREAR POR SUPERAR EL TAMAÑO MÁXIMO.

#### 5.1.4. Conjunto de pruebas de “Insertar nueva ruta”

El proceso de inserción de nuevo ruta requiere que un usuario registrado rellene los datos obligatorios del formulario y que estos cumplan ciertas condiciones por lo que es un punto de la aplicación propenso de errores.

#### Clases de equivalencia para “Insertar nueva ruta”

Condición	Clases válidas	Clases inválidas
Obligatoriedad de los campos	V1. El usuario registrado introduce información en todos los campos obligatorios.	I1. El usuario registrado no introduce todos los campos obligatorios.
Formato de datos	V2. El usuario introduce una ruta con un fichero con formato válido GPS.  V3. El usuario introduce una dificultad que está entre las opciones disponibles (Muy fácil, Fácil, Normal, Difícil, Muy difícil)  V4. El usuario introduce un tipo de ruta entre las opciones disponibles (Mountain Bike, Carretera, BMX, BMT, Ciclocross ...)  V5. El usuario introduce una serie de puntos válidos y con un número mínimo de puntos de 2 elementos.	I2. El usuario registrado introduce una ruta con un fichero que no es de tipo GPS.  I3. El usuario introduce una dificultad que no se encuentra entre las disponibles (Muy fácil, Fácil, Normal, Difícil, Muy difícil)  I4. El usuario introduce un tipo de ruta que no está entre los disponibles en la aplicación (Mountain Bike, Carretera, BMX, BMT, Ciclocross ...)  I5. El usuario no añade puntos o introduce un único punto

#### Casos de prueba para “Insertar nueva ruta”

Para reflejar los datos de entrada de los diferentes casos de prueba de este módulo se especifica a continuación una serie de datos de entrada genéricos que servirán para todos los casos de prueba, aunque en algún caso se puede sustituir alguno de ellos para alguna prueba concreta.

<b>Nombre de ruta:</b>	Ruta VoyEnBici
<b>Descripción de la ruta:</b>	Esta es una ruta de prueba realizada para comprobar el funcionamiento correcto de los casos de prueba realizados para insertar una nueva ruta.
<b>Dificultad:</b>	Fácil
<b>Fichero de ruta:</b>	ruta_gps.gpx (formato de ruta)
<b>Tipo de ruta:</b>	Mountain Bike
<b>Puntos:</b>	[43,5322015, -5,6611195], [43,5322015,-5,6711195], [43,5322015,-5,6781195]

Las pruebas a realizar con los datos genéricos aquí indicados serán las siguientes:

<b>Número de caso de prueba</b>	2.4.2.1
<b>Objetivo</b>	Caso de uso de insertar una ruta con campos obligatorios rellenados y con el formato correcto.
<b>Clases de equivalencia</b>	V1, V2, V3, V4, V5
<b>Datos de entrada</b>	Datos genéricos
<b>Salida esperada</b>	Mensaje de agradecimiento al usuario por haber creado una ruta. Se le devuelve al detalle de la misma para que vea el formato adoptado.
<b>Salida obtenida</b>	La esperada

<b>Número de caso de prueba</b>	2.4.2.2
<b>Objetivo</b>	Caso de uso de insertar una ruta sin todos los campos obligatorios rellenados.
<b>Clases de equivalencia</b>	I1
<b>Datos de entrada</b>	Datos genéricos modificando la ruta para no introducir el nombre de la misma.
<b>Salida esperada</b>	El usuario recibe un mensaje de error indicándole que la ruta no se ha podido crear debido a que falta uno o varios campos por llenar obligatoriamente.
<b>Salida obtenida</b>	La esperada (Ilustración I9)

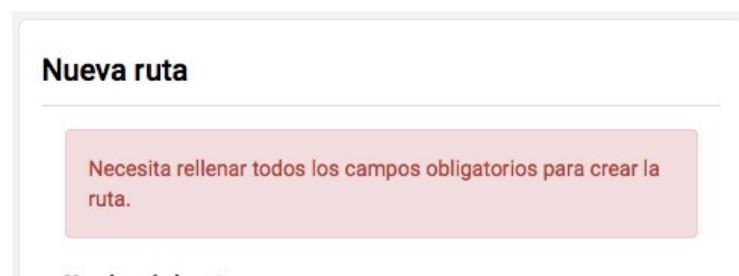


ILUSTRACIÓN I9. LA RUTA NO SE PUEDE CREAR POR NO INCLUIR TODOS LOS CAMPOS OBLIGATORIOS.

<b>Número de caso de prueba</b>	2.4.2.3
<b>Objetivo</b>	Caso de uso de insertar una ruta con un fichero de ruta con formato no válido
<b>Clases de equivalencia</b>	I2
<b>Datos de entrada</b>	Datos genéricos modificando el fichero a introducir para la ruta por <b>ruta_web.doc</b>
<b>Salida esperada</b>	El usuario recibe un mensaje de error indicándole que la ruta no se ha podido crear debido a que el formato del fichero introducido no es correcto.
<b>Salida obtenida</b>	La esperada (Ilustración I10)

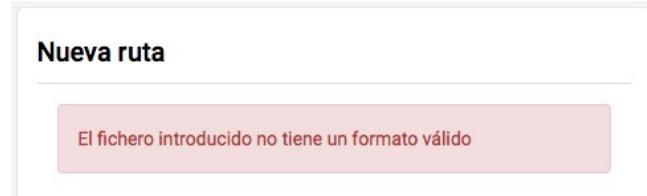


ILUSTRACIÓN I10. LA RUTA NO SE PUEDE CREAR POR QUE EL FICHERO NO TIENE UN FORMATO VÁLIDO.

<b>Número de caso de prueba</b>	2.4.2.4
<b>Objetivo</b>	Caso de uso de insertar una ruta con una dificultad que no se encuentra entre las disponibles en la aplicación
<b>Clases de equivalencia</b>	I3
<b>Datos de entrada</b>	Datos genéricos modificando la dificultad de la ruta por <b>Regular</b> .
<b>Salida esperada</b>	El usuario recibe un mensaje de error indicándole que la ruta no se ha podido crear debido a que el tipo de ruta introducido no es válido.
<b>Salida obtenida</b>	La esperada (Ilustración I11)

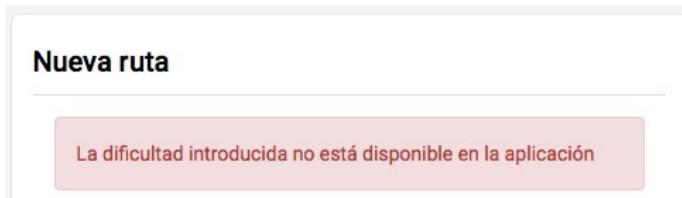


ILUSTRACIÓN I11. LA RUTA NO SE PUEDE CREAR POR INCLUIR UNA DIFICULTAD NO VÁLIDA.

<b>Número de caso de prueba</b>	2.4.2.5
<b>Objetivo</b>	Caso de uso de insertar una ruta con un tipo de ruta que no está entre las disponibles.
<b>Clases de equivalencia</b>	I4
<b>Datos de entrada</b>	Datos genéricos modificando el valor del tipo de ruta por <b>Ruta en Quad.</b>
<b>Salida esperada</b>	El usuario recibe un mensaje de error indicándole que la ruta no se ha podido crear debido a que el tipo de ruta no es uno de los disponibles en el sistema.
<b>Salida obtenida</b>	La esperada (Ilustración I12)

**Nueva ruta**

Debe introducir un tipo de ruta válida

ILUSTRACIÓN I12. LA RUTA NO SE PUEDE CREAR POR INCLUIR UN TIPO DE RUTA NO VÁLIDO.

<b>Número de caso de prueba</b>	2.4.2.6
<b>Objetivo</b>	Caso de uso de insertar una ruta con menos puntos de los necesarios o sin ellos.
<b>Clases de equivalencia</b>	I5
<b>Datos de entrada</b>	Datos genéricos modificando el valor de puntos por un valor nulo.
<b>Salida esperada</b>	El usuario recibe un mensaje de error indicándole que la ruta no se ha podido crear debido a que es necesario introducir puntos o al menos dos de ellos.
<b>Salida obtenida</b>	La esperada (Ilustración I13)

**Nueva ruta**

Necesita incluir al menos dos puntos de ruta

Nombre de la ruta:

ILUSTRACIÓN I13. LA RUTA DEBE TENER COMO MÍNIMO DOS PUNTOS PARA SER VÁLIDA.

### 5.1.5. Conjunto de pruebas para “Insertar nuevo evento”

El proceso de inserción de nuevo evento requiere que un usuario registrado rellene los datos obligatorios del formulario y que estos cumplan ciertas condiciones por lo que es un punto de la aplicación propenso de errores.

## Clases de equivalencia para “Insertar nuevo evento”

Condición	Clases válidas	Clases inválidas
Obligatoriedad de los campos	V1. El usuario registrado introduce información en todos los campos obligatorios.	I1. El usuario registrado no introduce todos los campos obligatorios.
Formato de datos	V2. El usuario introduce una fecha con formato válido.  V2. El usuario introduce un evento con una dirección válida.  V3. El usuario introduce un tipo de evento de entre las opciones disponibles	I2. El usuario introduce una fecha no válida.  I3. El usuario registrado introduce un evento con una dirección vacía o con una latitud / longitud no válidas.  I4. El usuario introduce un tipo de evento no existente de entre las opciones disponibles.

## Casos de prueba para “Insertar nuevo evento”

Para reflejar los datos de entrada de los diferentes casos de prueba de este módulo se especifica a continuación una serie de datos de entrada genéricos que servirán para todos los casos de prueba, aunque en algún caso se puede sustituir alguno de ellos para alguna prueba concreta.

<b>Nombre evento:</b>	Presentación de Proyecto fin de carrera
<b>Descripción:</b>	Reunido con el departamento docente para que evalúen la presentación realizada para el proyecto Voy en Bici.
<b>Fecha:</b>	19/02/2015
<b>Latitud:</b>	43.5322015
<b>Longitud:</b>	-5.6611195
<b>Tipo evento:</b>	BMX

Las pruebas a realizar con los datos genéricos aquí indicados serán las siguientes:

<b>Número de caso de prueba</b>	2.5.2.1
<b>Objetivo</b>	Caso de uso de insertar un nuevo evento con los campos obligatorios llenados y con el formato correcto
<b>Clases de equivalencia</b>	V1, V2, V3, V4
<b>Datos de entrada</b>	Datos genéricos
<b>Salida esperada</b>	El usuario crea el evento y se le redirige al mismo mostrándole un mensaje de agradecimiento del mismo.
<b>Salida obtenida</b>	La esperada

<b>Número de caso de prueba</b>	2.5.2.2
<b>Objetivo</b>	Caso de uso de insertar un evento sin introducir todos los campos obligatorios
<b>Clases de equivalencia</b>	I1
<b>Datos de entrada</b>	Datos genéricos sin añadir el nombre del evento
<b>Salida esperada</b>	El usuario recibe un mensaje de error indicándole que el evento no se ha podido crear debido a que uno o varios campos no tienen información.
<b>Salida obtenida</b>	La esperada (Ilustración I14)

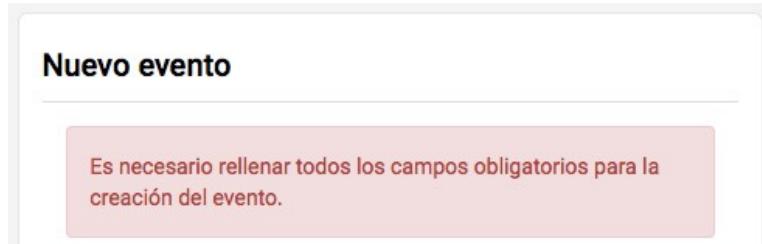


ILUSTRACIÓN I14. EL EVENTO NO SE PUEDE CREAR POR NO INCLUIR TODOS LOS CAMPOS OBLIGATORIOS.

<b>Número de caso de prueba</b>	2.5.2.3
<b>Objetivo</b>	Caso de uso de insertar un evento cuando la fecha de inicio no tiene el formato indicado
<b>Clases de equivalencia</b>	I2
<b>Datos de entrada</b>	Datos genéricos modificando la fecha de inicio por <b>02/19/15</b>
<b>Salida esperada</b>	El usuario recibe un mensaje de error indicándole que el evento no se ha podido crear debido a que la fecha de inicio no tiene el formato indicado.
<b>Salida obtenida</b>	La esperada (Ilustración I15)

Introduce una fecha válida

Fecha	13/13/2015
-------	------------

ILUSTRACIÓN I15. EL EVENTO NO SE PUEDE CREAR POR INSERTAR UNA FECHA INCORRECTA.

<b>Número de caso de prueba</b>	2.5.2.4
<b>Objetivo</b>	Caso de uso de insertar un evento con un formato no válido de latitud.
<b>Clases de equivalencia</b>	I3
<b>Datos de entrada</b>	Datos genéricos modificando la latitud por <b>Gijón</b>
<b>Salida esperada</b>	El usuario recibe un mensaje de error indicándole que el evento no se ha podido crear debido a que la latitud no tiene el formato válido.
<b>Salida obtenida</b>	La esperada (Ilustración I16)

**Nuevo evento**

Debe introducir una localización válida.

ILUSTRACIÓN I16. EL EVENTO NO SE PUEDE CREAR POR AÑADIR DATOS DE GEOLOCALIZACIÓN INCORRECTOS.

<b>Número de caso de prueba</b>	2.5.2.5
<b>Objetivo</b>	Caso de uso de insertar un tipo de evento no existente.
<b>Clases de equivalencia</b>	I4
<b>Datos de entrada</b>	Datos genéricos modificando el tipo de evento por <b>Motociclismo</b>
<b>Salida esperada</b>	El usuario recibe un mensaje de error indicándole que el evento no se ha podido crear debido a que no existe ningún tipo de evento con esa denominación.
<b>Salida obtenida</b>	La esperada (Ilustración I17)

**Nuevo evento**

Debe introducir un tipo de evento válido

ILUSTRACIÓN I17. EL EVENTO NO SE PUEDE CREAR POR NO EXISTIR EL TIPO DE EVENTO ASOCIADO.

### 5.1.6. Conjunto de pruebas para “Insertar nueva conversación”

El proceso de inserción de nuevo evento requiere que un usuario registrado rellene los datos obligatorios del formulario y que estos cumplan ciertas condiciones por lo que es un punto de la aplicación propenso de errores.

#### Clases de equivalencia para “Insertar nueva conversación”

Condición	Clases válidas	Clases inválidas
Existencia de la conversación entre usuarios en la base de datos	V1. En la base de datos no existe ninguna conversación en la que los participantes sean los dos usuarios registrados.	I1. En la base de datos existe una conversación en la que participan ambos usuarios registrados.
Obligatoriedad de los campos	V2. El usuario que inicia la conversación introduce todos los campos obligatorios.	I2. El usuario que inicia la conversación no introduce todos los campos obligatorios.
Existencia de los usuarios entre los que se inicia la conversación	V3. Los usuarios participantes en la conversación existen.	I3. No existe el usuario con el que se intenta mantener la conversación.

#### Casos de prueba para “Insertar nueva conversación”

Para reflejar los datos de entrada de los diferentes casos de prueba de este módulo se especifica a continuación una serie de datos de entrada genéricos que servirán para todos los casos de prueba, aunque en algún caso se puede sustituir alguno de ellos para alguna prueba concreta.

<b>Usuario inicial:</b>	Usuario registrado ( <a href="mailto:alejandro.lopez@uniovi.es">alejandro.lopez@uniovi.es</a> )
<b>Usuario final:</b>	Usuario registrado ( <a href="mailto:test@uniovi.es">test@uniovi.es</a> )
<b>Mensaje:</b>	Este es un mensaje de prueba inicial para testing de conversaciones entre usuarios.

Las pruebas a realizar con los datos genéricos aquí indicados serán las siguientes:

<b>Número de caso de prueba</b>	2.6.2.1
<b>Objetivo</b>	Caso de uso de crear una nueva conversación entre dos usuarios entre los que no existe una conversación previa
<b>Clases de equivalencia</b>	V1, V2, V3
<b>Datos de entrada</b>	Datos genéricos
<b>Salida esperada</b>	Mensaje creado entre los dos usuarios viendo el mensaje escrito entre ellos.
<b>Salida obtenida</b>	La esperada

<b>Número de caso de prueba</b>	2.6.2.2
<b>Objetivo</b>	Caso de uso de crear una nueva conversación entre dos usuarios entre los que ya existe una conversación previa
<b>Clases de equivalencia</b>	I1
<b>Datos de entrada</b>	Datos genéricos tras haber realizado previamente el caso de prueba 2.6.2.1
<b>Salida esperada</b>	Redirección a la conversación previamente creada.
<b>Salida obtenida</b>	La esperada (Ilustración I18)

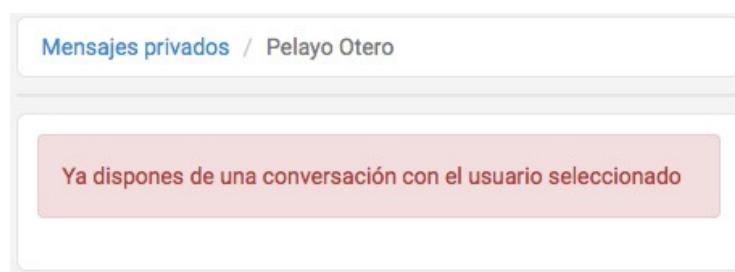


ILUSTRACIÓN I18. LA CONVERSACIÓN NO SE PUEDE CREAR AL EXISTIR DOS USUARIOS QUE YA DISPONEN DE UNA CONVERSACIÓN PREVIA.

<b>Número de caso de prueba</b>	2.6.2.3
<b>Objetivo</b>	Caso de uso en el que un usuario crea una conversación sin llenar todos los campos obligatorios rellenados.
<b>Clases de equivalencia</b>	I2
<b>Datos de entrada</b>	Datos genéricos sin mensaje.
<b>Salida esperada</b>	Se le muestra al usuario un mensaje de error indicándole que rellene toda la información obligatoria.
<b>Salida obtenida</b>	La esperada (Ilustración I19)

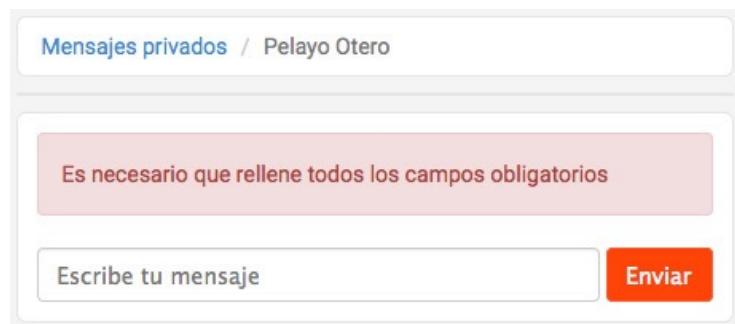


ILUSTRACIÓN I19. LA CONVERSACIÓN NO SE PUEDE CREAR AL SER NECESARIOS TODOS LOS CAMPOS OBLIGATORIOS.

<b>Número de caso de prueba</b>	2.6.2.4
<b>Objetivo</b>	Caso de uso de crear una nueva conversación entre dos usuarios cuando uno de ellos no existe en el sistema.
<b>Clases de equivalencia</b>	I3
<b>Datos de entrada</b>	Datos genéricos modificando el usuario final por <a href="mailto:inexistente@uniovi.es">inexistente@uniovi.es</a>
<b>Salida esperada</b>	Mensaje de error indicando al usuario que el usuario con el que intenta mantener la conversación no existe en base de datos.
<b>Salida obtenida</b>	La esperada (Ilustración I20)

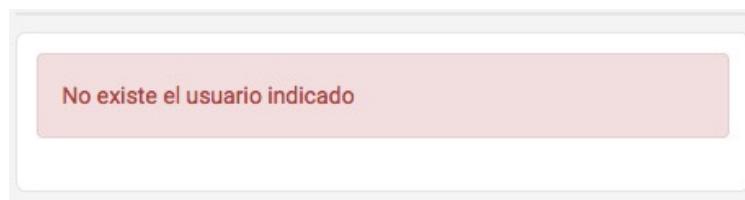


ILUSTRACIÓN I20. LA CONVERSACIÓN NO SE PUEDE CREAR AL NO EXISTIR LOS DOS USUARIOS QUE PARTICIPARÁN EN LA CONVERSACIÓN.

### 5.1.7. Conjunto de pruebas para “Enviar formulario de contacto”

El proceso de envío del formulario de contacto requiere que un usuario rellene los datos obligatorios del formulario y que estos cumplan ciertas condiciones por lo que es un punto de la aplicación propenso de errores.

#### Clases de equivalencia para “Enviar formulario de contacto”

Condición	Clases válidas	Clases inválidas
Obligatoriedad de los campos	V1. El usuario realiza el envío de información a partir de lo solicitado en el formulario de contacto.	I1. El usuario que intenta realizar el envío de información no rellena todos los campos obligatorios.

#### Casos de prueba para “Enviar formulario de contacto”

Para reflejar los datos de entrada de los diferentes casos de prueba de este módulo se especifica a continuación una serie de datos de entrada genéricos que servirán para todos los casos de prueba, aunque en algún caso se puede sustituir alguno de ellos para alguna prueba concreta.

<b>Nombre:</b>	Alejandro López
<b>Email:</b>	<u>alejandro.lopez@uniovi.es</u>
<b>Asunto:</b>	Funcionamiento de la aplicación
<b>Comentario:</b>	Este es un comentario de prueba para la documentación del proyecto fin de carrera de la red social de ciclismo.

Las pruebas a realizar con los datos genéricos aquí indicados serán las siguientes:

<b>Número de caso de prueba</b>	2.7.2.1
<b>Objetivo</b>	Caso de uso en el que un usuario realiza el envío de información a partir de lo solicitado en el formulario de contacto
<b>Clases de equivalencia</b>	V1
<b>Datos de entrada</b>	Datos genéricos
<b>Salida esperada</b>	Mensaje de agradecimiento por la participación con el envío de información
<b>Salida obtenida</b>	La esperada

<b>Número de caso de prueba</b>	2.7.2.2
<b>Objetivo</b>	Caso de uso en el que un usuario intenta realizar el envío de información a partir de lo solicitado en el formulario de contacto sin llenar toda la información obligatoria.
<b>Clases de equivalencia</b>	I1
<b>Datos de entrada</b>	Datos genéricos modificando campos obligatorios y dejándolos vacíos.
<b>Salida esperada</b>	Mensaje de error indicando los campos obligatorios que faltan por llenar.
<b>Salida obtenida</b>	La esperada (Ilustración I21)

## Datos de la consulta

Si desea ponerse en contacto con nosotros por favor rellene el siguiente formulario y nos pondremos en contacto tan pronto como sea posible.

No se pudo guardar este/a contacto porque se encontraron 6 × errores

- Comentarios no puede estar en blanco
- Email no puede estar en blanco
- Email no es válido
- Nombre no puede estar en blanco
- Asunto no puede estar en blanco
- Política de privacidad debe ser aceptado

ILUSTRACIÓN I21. EL FORMULARIO DE CONTACTO NO SE PUEDE ENVIAR PORQUE ES NECESARIO RELLENAR TODOS LOS CAMPOS OBLIGATORIOS.



# 6. Presupuesto

---

## 6.1. Planificación del proyecto

### 6.1.1. Desglose en etapas

Las diferentes etapas por las que pasa este proyecto en su realización son las que se muestran a continuación:

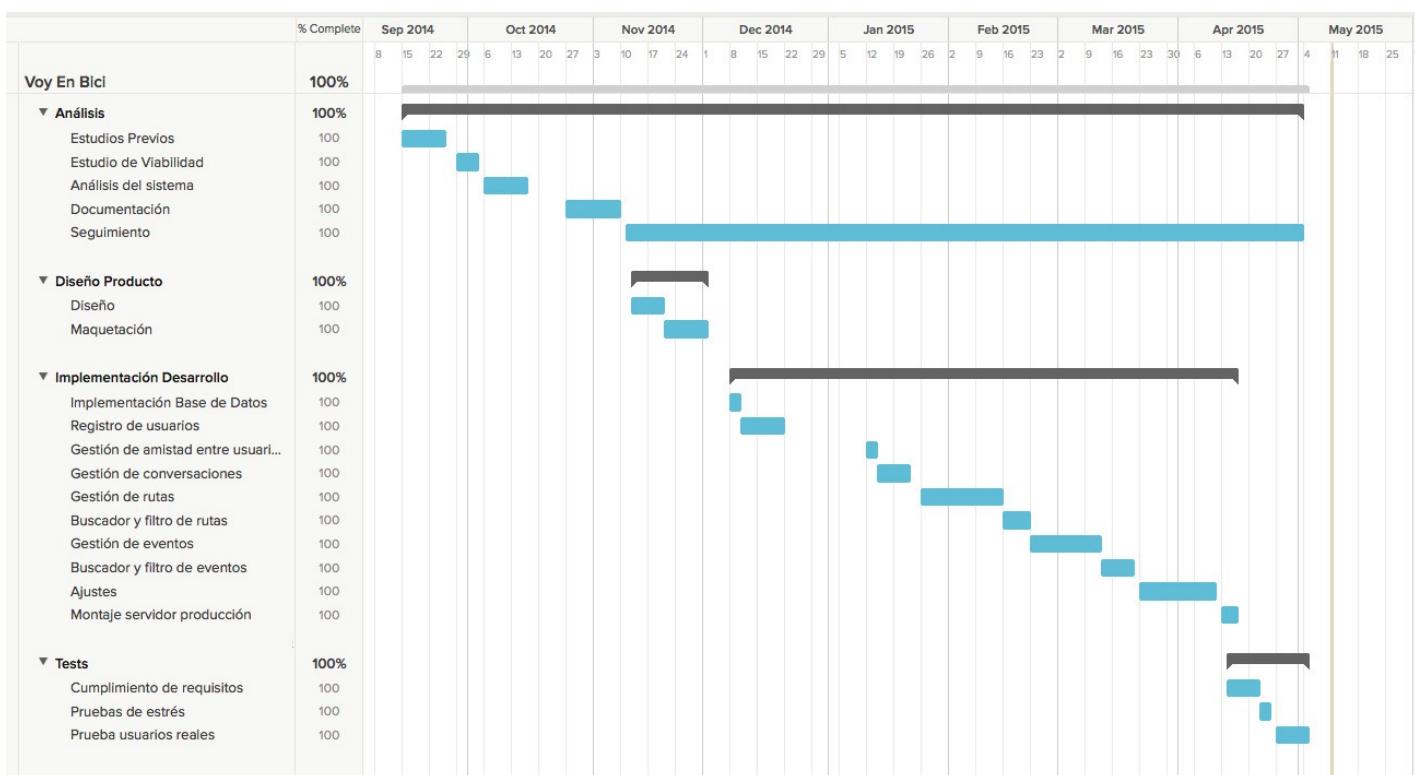
- **Análisis:** El punto de partida inicial del proyecto se desarrolló como un nexo de unión para que personas aficionadas al ciclismo pudiesen tener un punto de reunión en Internet en el que almacenar las rutas realizadas, competiciones, preparar salidas, comunicarse con otros usuarios o grupos de ciclistas. Tras la idea inicial se desarrollaron unos requisitos funcionales del sistema que fueron el inicio del desarrollo del mismo.
- **Diseño:** Como objetivo de la etapa de diseño se propondrá definir la arquitectura del sistema en partes lógicas (subsistemas de diseño) y en detallar el diseño de cada uno de ellos. También se estudiarán las diferentes estructuras de datos necesarias para llevar a cabo el sistema. La especificación de todos estos objetivos se hará de forma que guíe la codificación posterior, a través del documento de Diseño obtenido durante el desarrollo de esta etapa.
- **Instalación:** Preparación del entorno software y hardware donde se realizarán las labores de desarrollo del sistema, así como de los distintos entornos que se utilizarán durante el desarrollo.
- **Programación:** Consiste en la programación de la aplicación utilizando el framework Ruby on Rails para el back-end y para la parte front-end mediante plantillas javascript ECO y peticiones JSON a la api REST generada en el back-end.
- **Pruebas:** Consiste en la realización de pruebas de la aplicación para detectar posibles fallos y probar su fiabilidad y efectividad, consiguiendo de esta manera una aplicación robusta.
- **Documentación:** Aunque la generación de documentos será constante durante todo el proceso de construcción del software, la última etapa será específica para la

elaboración de diferentes manuales para la utilización del software por parte de los usuarios finales.

### 6.1.2. Planificación temporal

La duración del proyecto se estima en 6 meses, aquí muestro una aproximación del tiempo dedicado dividido en las etapas citadas anteriormente:

- **Análisis:** 100 horas
- **Diseño:** 40 horas
- **Instalación:** 5 horas
- **Programación:** 470 horas
- **Pruebas:** 20 horas
- **Documentación:** 80 horas



## 6.2. Presupuesto para el desarrollo de la aplicación

Para el desarrollo del sistema serán necesarios una serie de recursos tanto hardware como software.

### 6.2.1. Recursos hardware

PC compatible para desarrollar las técnicas de análisis y diseño, así como la codificación del sistema y redacción de la documentación.

Este PC deberá ser compatible para tomar el papel de servidor web y servidor de base de datos en etapa de codificación y pruebas.

Hemos utilizado un único terminal con las siguientes características:

- Procesador Intel Core i5
- Memoria Ram 4Gb
- Disco duro 500Gb
- Tarjeta Gráfica Intel HD Graphics 4000 1024 MB

### 6.2.2. Recursos software

El conjunto de aplicaciones software usadas durante el desarrollo de la aplicación son las siguientes:

**Sistema operativo:** Mac OS X 10.10 Yosemite

**Sistema de Bases de Datos:** MongoDB 3.0.3

**Entorno de programación:** Sublime Text

**Navegador web:** Google Chrome / Mozilla Firefox / Safari (en sus últimas versiones actualizadas)

**Generación y maquetado de la documentación:** Pages

**Creación de diagramas:** Cacoo (Página web - <https://cacoo.com>)

**Manipulación de imágenes:** Adobe Photoshop CC

### 6.2.3. Mano de obra

Para hacer este cálculo, nos basaremos en la contabilización de las horas invertidas en la realización del proyecto y que se han comentado en el apartado de planificación temporal.

Los datos de los precios / hora han sido obtenidos de diferentes páginas web donde se comentan las remuneraciones para el desarrollo de aplicaciones web tanto en su parte de diseño gráfico, análisis, jefatura de proyecto y desarrollador web.

**Analista programador:** Realiza las tareas de planificación, análisis, diseño y estudios previos.

Tarea	Horas
Seguimiento	15
Estudio de viabilidad	10
Estudios previos	60
Análisis del sistema	50
Documentación	80
<b>TOTAL</b>	<b>215</b>

**Desarrollador web:** Realiza las tareas de diseño, transformación a HTML / CSS / JS y desarrollo.

Tarea	Horas
Diseño	30
Maquetación	40
Implementación del desarrollo	400
<b>TOTAL</b>	<b>470</b>

**Director del proyecto:** Realiza tareas de seguimiento del proyecto y verificación del funcionamiento.

Tarea	Horas
Definición de requisitos	4
Estudios previos	10
Cumplimiento de requisitos	20
Validación del diseño	4
Seguimiento implementación	12
<b>TOTAL</b>	<b>50</b>

#### 6.2.4. Detalle del coste

TIPO	CONCEPTO	HORAS	IMPORTE	TOTAL
<b>Hardware</b>	Procesador Intel Core i5 Memoria RAM 4GB Seagate 500GB Intel HD Graphics 4000		1129 €	
				1129 €
<b>Software</b>	Adobe Photoshop CC		12€ mensuales	
				72 € (6 meses de proyecto)
<b>Mano de Obra</b>	Analista	215	55 € / hora	11825 €
	Desarrollador	470	42 € / hora	19740 €
	Director proyecto	50	75 € / hora	3750 €
				35315 €
<b>Total Costes del Proyecto</b>				36520 €
<b>IVA (21%)</b>				7669,2 €
<b>TOTAL FINAL</b>				44189,2 €



# 7. Manual de Usuario

Todo el proyecto se ha desarrollado pensando siempre en la sencillez de uso de manera que un usuario sin conocimientos específicos de informática sea capaz de aprovechar la funcionalidad al máximo.

Así mismo se cree necesaria unos ciertos conocimientos previos básicos de lo que supone la navegación por internet y el uso de dispositivos móviles smartphone o tablets. Dado el carácter plenamente orientado al colectivo de deportistas o aficionados al ciclismo que tiene este proyecto, se supone también en este manual cierto grado de familiarización con el entorno de dicho deporte.

## 7.1. Introducción a Voy en Bici

Para acceder a la plataforma deberemos abrir un navegador web, por ejemplo Mozilla Firefox, y escribimos en la barra de direcciones lo siguiente:

<http://localhost:3000>

si ha sido ejecutada en el puerto 3000, el cual es el puerto por defecto en el que se arrancan las aplicaciones en Ruby on Rails aunque este puerto se puede ajustar durante el arranque de la aplicación o mediante la configuración del servidor utilizado.

Tras acceder a la plataforma la primera pantalla que se visualizará es la pantalla de login de la aplicación en la que un usuario ya registrado podrá acceder a la plataforma introduciendo su email y contraseña.

En caso de no disponer de usuario registrado o de no recordar los datos de acceso puede leer los dos siguientes puntos.



### 7.1.1. Registro de usuario

Si no dispone de un usuario en la plataforma dispone de la posibilidad de crearlo pulsando en el texto de Registro, el cual proporcionará un formulario de registro de usuario donde se le solicitarán unos campos básicos de registro.

Una vez rellenados dichos campos correctamente se le hará un envío de un email agradeciendo su participación en la plataforma y por último se le redirigirá a un tutorial para que vea de un vistazo las características más importantes.

## 7.1.2. Recordar contraseña



Si dispone de un usuario registrado pero no recuerda los datos de acceso a la plataforma puede solicitar un recordatorio de contraseña.

Para ello debe pulsar en el enlace que le indica Recordar contraseña. Dicho enlace le llevará a una pantalla donde se le solicitará un correo electrónico el cual debe coincidir con alguno de los almacenados en la plataforma.

Si existe el correo electrónico seleccionado se enviará un email a dicho correo informando de que ha solicitado la modificación de la contraseña y puede cambiarla pulsando sobre un enlace generado única y exclusivamente para el usuario

solicitante. Dicho enlace tiene una validez de 6 horas, transcurrido ese período de tiempo no tendrá validez y si intentase acceder le redirigirá a la pantalla inicial de la plataforma.

Si pulsa en dicho enlace dentro de las 6 horas iniciales de la solicitud del cambio de contraseña accederá a una página donde podrá introducir la nueva contraseña asociada al usuario del email con el que solicitó el recordatorio de contraseña.

## 7.1.3. Acceso a la plataforma



Si ya dispone de un usuario en la plataforma Voy en Bici puede acceder a la misma desde la pantalla inicial del proyecto incluyendo el email y la contraseña con la que realizó el registro.

Si los datos introducidos son correctos se le enviará a ***Mi muro*** donde se accederá a los últimos comentarios realizados en la plataforma por los usuarios que seguimos en la plataforma.

Si es su primer acceso previamente se le enviará a un tutorial donde se le explicarán unos conceptos básicos sobre la plataforma Voy en Bici.

## 7.2. Acceso como usuario registrado

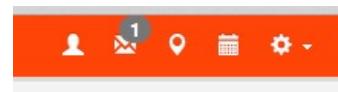
Tras acceder a la plataforma **por primera vez** lo primero que verá será un tutorial en el que mediante unas imágenes descriptivas se explicará el funcionamiento estándar de Voy en Bici.

### 7.2.1. Enlaces rápidos de usuario

El usuario se encuentra con dos opciones de enlaces rápidos desde los que pueda acceder a los distintos aspectos de la aplicación.

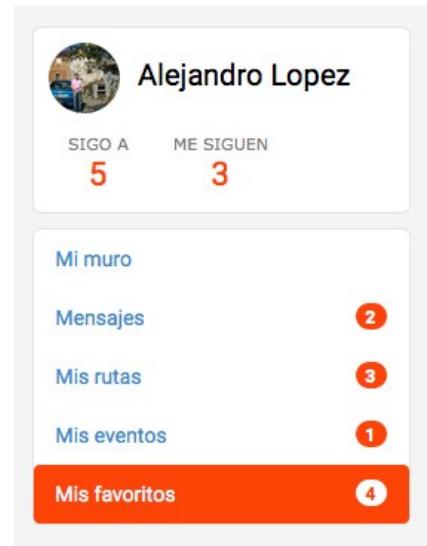
En la parte superior de la aplicación en la parte izquierda disponemos de un buscador en el que poder hacer búsquedas de los distintos usuarios almacenados en la aplicación y en la parte derecha y de izquierda a derecha accesos a:

- mi muro
- mis conversaciones
- rutas
- eventos
- perfil de usuario (edición de perfil, contacto y cerrar sesión)



En el menú lateral izquierdo se puede acceder a los puntos específicos de cada usuario siendo en orden:

- muro de usuario
- conversaciones
- mis rutas
- mis eventos
- mis favoritos



### 7.2.2. Mi muro

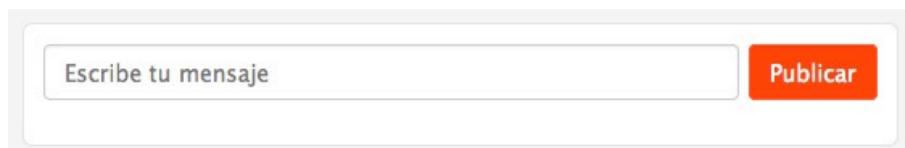
Un usuario al acceder a mi muro tiene acceso a un menú para acceder al resto de sus opciones como puedan ser mis eventos o mis rutas y a la derecha se despliega un listado con todos los comentarios de la gente a la que sigue en la plataforma así como sus propios comentarios ordenados en orden descendente a como se han realizado.

Primero aparecen los más nuevos y siguen hacia los más antiguos. Dicha pantalla se va recargando automáticamente mostrando la nueva información que se haya podido originar durante el intervalo de tiempo.

En la parte superior del listado de los comentarios del usuario se dispone de un formulario para escribir un nuevo comentario. Los comentarios disponen de un máximo de 256 caracteres, superada dicha cifra se impedirá el envío del formulario quedando inhabilitado para ello el botón de enviar y notificando un error por pantalla para avisar al usuario del problema indicado.

### 7.2.3. Escribir un comentario en mi muro

Como usuario registrado dispones de la posibilidad de compartir información con otros usuarios a través de Mi muro. Este punto consiste en un conjunto de todas tus ideas o los comentarios que quieras compartir con el resto de usuarios de Voy en Bici y que entren a tu perfil o te sigan.



Para ello sólo debes acceder a tu muro de usuario, pulsar en la caja de texto e introducir un mensaje inferior a 256 caracteres. Una vez que se pulse sobre publicar tu comentario será visible para el resto de usuarios.

### 7.2.4. Editar un comentario escrito por mí

Como usuario registrado y tras haber escrito un comentario tenemos la posibilidad de editarlo. Para ello en el propio comentario nos aparecerá un botón de editar que nos permitirá hacer modificaciones en el comentario.



Se nos abrirá una ventana modal o emergente que nos mostrará un formulario para modificar el contenido. Una vez que pulsemos sobre el botón de actualizar nuestro comentario será modificado.

### 7.2.5. Eliminar un comentario

Igualmente como la opción de editar un comentario se puede dar el caso de que queramos eliminar un comentario que hayamos escrito previamente. Para ello sólo deberemos pulsar en el botón de eliminar asociado a nuestro comentario y se nos pedirá una confirmación para eliminar el comentario seleccionado.



Una vez que pulsemos en el botón de confirmación nuestro comentario desaparecerá del sistema.

### 7.2.6. Buscador de usuarios

A screenshot of the user search interface. The search bar at the top shows the letters "al". Below it, two user profiles are listed: "Alejandro Lopez" and "Alberto Contador". Each profile includes a small profile picture, the user's name, and a "SIGO A" (Follows) button with the number "0" next to it. To the right of the profiles is a search icon. Below the profiles, there is a section with the text "No disp..." and "Quizás quier...".

En la parte superior de la aplicación y siempre visible a los usuarios se encuentra el menú principal de Voy en Bici de los usuarios registrados donde disponemos de un buscador de usuarios y enlaces al muro del usuario, a las conversaciones del usuario, a todas las rutas, a todos los eventos y por último a las opciones

de usuario.

El buscador funciona de tal manera que hay que introducir al menos 3 caracteres para que nos devuelva información relativa a los usuarios que coinciden con dicha búsqueda. No es sensible a las mayúsculas y las minúsculas y nos mostrará los resultados que vaya encontrando para que podamos acceder al muro del usuario seleccionado y de esta manera enviarle un mensaje privado o añadirlo o eliminarlo de los seguidores.

## 7.2.7. Ver perfil de un usuario registrado

Raquel Rozos

SIGUE A 1 LE SIGUEN 1

Muro

Rutas 2

Eventos 1

Favoritos 0

Raquel Rozos 30 m Hoy he disfrutado co primavera :)

Raquel Rozos 23 d: Que llegue ya el vera

Para acceder al perfil de un usuario se puede realizar desde varios puntos de la aplicación como son el buscador de usuarios, desde el listado de usuarios que sigue otro usuario registrado en la aplicación o desde algún comentario que haya realizado en la aplicación. En el perfil del usuario que estemos visualizando disponemos de un número de rutas, número de eventos y

comentarios que ha creado así como el número de usuarios que sigue o que le siguen para acceder a dichos listados.

Desde el perfil del usuario disponemos también de la opción de seguir a dicho usuario así como de empezar una nueva conversación privada o acceder a la ya existente.

## 7.2.8. Añadir un usuario a los usuarios que sigo

¿Has encontrado a un usuario que quieras seguir en Voy en Bici? Es tan sencillo como acceder a su perfil de usuario, ya sea desde el buscador, desde algún comentario en rutas o eventos o desde el listado de usuarios que te siguen y pulsar en el enlace de **Seguir**.

David Muñiz

Seguir

## 7.2.9. Quitar a un usuario de los usuarios que sigo

Si quieres eliminar a un usuario de los usuarios que sigues también hay disponibles varias opciones.

Sergio Iglesias

Runner y ciclista de fin de semana. Con ganas de conocer nuevas rutas en Gijón y alrededores.

Dejar de seguir

Igual que en el punto anterior se puede eliminar de los usuarios que se sigue desde la pantalla de detalle de un usuario accediendo tanto desde el buscador, desde un comentario de rutas o eventos o desde el listado de usuarios a los que sigo o desde el propio listado de *usuarios a los que seguimos* aparece un enlace para **Dejar de seguir** al usuario seleccionado.

### 7.2.10. A quién sigo

**Mira a quién sigues**

Estos son los usuarios de Voy en Bici que estás siguiendo

---

 **Alberto Contador**  
Ciclista a nivel profesional desde los 15 años.

---

 **Raquel Rozos**  
Soy maestra en Candás y desde que he llegado allí he descubierto una nueva pasión en la bicicleta por sus cuestas y con su gente.

---

 **Sergio Iglesias**  
Runner y ciclista de fin de semana. Con ganas de conocer

acceder al detalle de un usuario concreto así como eliminar a dichos usuarios de los usuarios que sigues.

Para ver a los usuarios que sigo se puede acceder desde el menú situado en la parte izquierda junto a tu foto de usuario pulsando en el número de personas a las que sigues.

Dicha pantalla es un listado con todos los usuarios que estás siguiendo en la plataforma, desde este punto de la aplicación puedes

### 7.2.11. Mis seguidores

**Mira quién te sigue**

Los usuarios de Voy en Bici que te siguen

---

 **Raquel Rozos**  
Soy maestra en Candás y desde que he llegado allí he descubierto una nueva pasión en la bicicleta por sus cuestas y con su gente.

---

 **David Muñiz**

---

 **Pleayo Otero**  
Me encanta la bicicleta y la escalada, sentirme en plena naturaleza es una de las mejores aficiones que existen.

Si en cambio quieras ver a esos usuarios que te siguen en la aplicación debes acceder en el menú lateral izquierdo justo debajo de tu nombre de usuario al enlace situado en el número de **Quien te sigue**.

Desde dicho listado tenemos acceso a los usuarios que nos están siguiendo en la plataforma de manera que podamos acceder a su perfil de usuario, seguir a dicho usuario o dejar de seguir a dicho usuario según el listado de usuarios que seguimos.

## 7.2.12. Mis conversaciones

Como usuario registrado de Voy en Bici disponemos de un acceso de las conversaciones privadas que estemos manteniendo con otro usuarios de la aplicación.

Mensajes privados

**Alberto Contador** hace 2 minutos  
Muchas gracias Raquel ! Pues la verdad que espero que me vaya muy bien la carrera y pue  
nivel es muy alto !!

**David Muñiz** hace 15 minutos  
Ey Raque !!! No sabía que tú también estabas en Voy en Bici ! ¿Cómo va todo?

**Alejandro Lopez** hace 2 días  
Hola Ale ! ¿viste la última ruta que he compartido? Está genial!!!! !!!

Para ello disponemos de dos accesos, en el menú lateral izquierdo hay un punto denominado **Mensajes**, junto a un número que nos indicará todas las conversaciones que tenemos activas o desde la cabecera en el icono con forma de sobre que puede tener un número asociado que se corresponde al número de mensajes no leídos y que ambos enlaces nos llevarán al listado de todas las conversaciones disponibles.

## 7.2.13. Detalle de una conversación

Tras haber accedido al punto **Mis conversaciones** si pulsamos sobre la imagen o el nombre del usuario seleccionado accederemos al detalle de la conversación que estamos manteniendo con dicho usuario.

Mensajes privados / Raquel Rozos

**Raquel Rozos** hace 3 minutos  
Hola Alberto, soy una gran fan tuya, ¿crees que llegarás a ganar el Giro este año?

**Raquel Rozos** hace 2 minutos  
Espero que sí lo hagas, te lo mereces, que tengas mucha suerte !!

**Alberto Contador** menos de un minuto  
Muchas gracias Raquel ! Pues la verdad que espero que me vaya muy bien la carrera y pueda llegar a ganar aunque el  
nivel es muy alto !!

Escribe tu mensaje

Enviar

En dicho detalle veremos un listado de todos los mensajes escritos con dicho usuario y la posibilidad de escribir un mensaje nuevo.

#### 7.2.14. Escribir un mensaje en una conversación

Desde el detalle de una conversación disponemos de la opción de añadir un nuevo mensaje a la misma. No hay límite de caracteres y al pulsar en el botón enviar se adjuntará el nuevo mensaje a la conversación.

Le llegará una notificación al usuario con el que mantenemos la conversación privada de manera que sepa que tiene un mensaje nuevo sin leer.

#### 7.2.15. Ver rutas de un usuario

Rutas

¿Quieres andar en bici? Busca entre las rutas que disponemos

Nueva ruta

Renfe Oviedo - P.R. AS-183 Ruta Priañes (Ruta de los Meandros del Nora) - Escamplero - Renfe Oviedo 62

★ Desmarcar de favoritos

Gran circular Gijón, Peón, Niévares, Villaviciosa, el Puntal y vuelta por carreteras costeras 56

Mi última ruta de entrenamiento rodador antes de la maratón de los monegros la hi ...

★ Marcar como favorito

Desde nuestro perfil de usuario o desde el perfil de otro usuario se puede acceder a un listado de las rutas que ha creado dicho usuario.

Se visualizará un listado con todas las rutas creadas en orden descendente por fecha de creación en la que veremos una

imagen estática de la ruta creada sobre un mapa, el nombre de la ruta y una breve descripción que nos permitirá acceder al detalle de la ruta seleccionada.

También se dispone de un enlace para crear una nueva ruta o para editarla y eliminarla en caso de que la ruta sea propiedad de nuestro usuario registrado

#### 7.2.16. Dar de alta una ruta

Nueva ruta

Como usuarios registrados de la plataforma disponemos de la posibilidad de dar de alta nuevas rutas en la aplicación.

Para ello disponemos de dos puntos iniciales de acceso a la creación de rutas, desde el listado completo de rutas de la aplicación al que se puede acceder desde el menú superior en el

ícono asociado a las rutas o desde mis rutas en el menú de usuario.

En ambos casos nos aparecerá un botón de *Nueva Ruta* que nos facilitará la creación de una ruta en el sistema.

Una vez pulsado se abrirá una ventana emergente que nos facilitará dos opciones para realizar el alta de una ruta.



Si escogemos la opción de importar desde una URL se solicitará el enlace de una página web que contenga una ruta y la aplicación leerá toda la información disponible de manera que el usuario no tendrá que hacer nada más y automáticamente tendrá la ruta importada.



En caso de escoger la segunda opción se le enviará al formulario de creación de nueva ruta (*al ser el mismo formulario que el que se verá en el próximo punto se explicará a continuación*)

## 7.2.17. Edición de Ruta

Una vez que hemos creado nuestra ruta tenemos la posibilidad de volver a editarla debido a que queremos actualizar la información que se muestra, queremos hacerla privada o pública o cualquier otra opción disponible.

Para ello se puede acceder a la edición



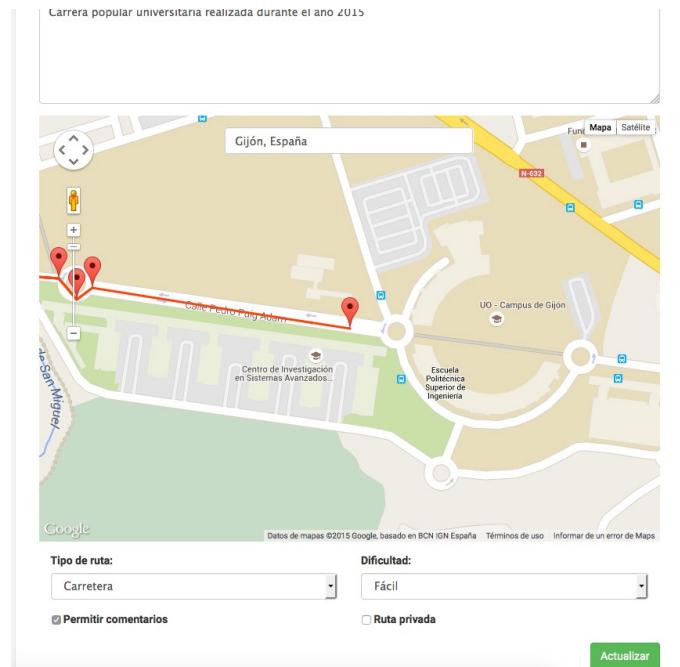
de la ruta desde el listado de todas las rutas disponibles o

desde mis rutas. En ambos casos siempre tiene que coincidir que seamos el propietario de la ruta o tengamos permisos de edición sobre ella.

Dispondremos de un enlace de editar tanto desde el listado como desde el propio detalle de la ruta a editar.

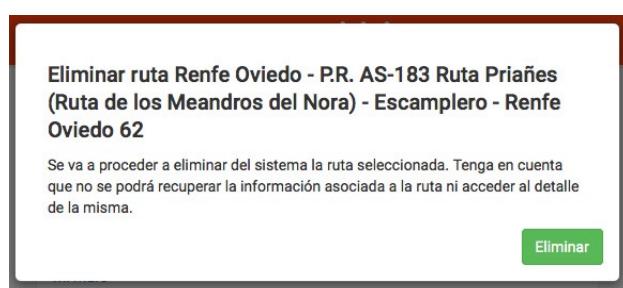
Una vez que pulsamos sobre el botón de editar accederemos al formulario de edición de ruta donde podremos modificar toda la información disponible

de la misma así como los puntos del mapa que editemos, el título, resumen o descripción de la misma.



## 7.2.18. Eliminar ruta

Igualmente y en el mismo caso que para la edición de una ruta disponemos de la posibilidad de eliminar una ruta que hubiésemos compartido previamente en la aplicación.



Para ello y desde el mismo acceso que en el caso anterior podemos eliminar una ruta. Al pulsar en el enlace de borrado se nos pedirá confirmación del paso que vamos a realizar ya que no podremos recuperar dicha información.

### 7.2.19. Buscar rutas

Desde el menú superior se tiene acceso a un completo listado de todas las rutas disponibles en la aplicación.

Dicho punto funciona como un buscador en sí mismo desde el primer momento. Se solicitará al usuario la posibilidad de geolocalizar su posición, en caso de aceptar dicha opción el navegador dispondrá de su posición a partir de las coordenadas de su situación mediante la señal de red disponible en ese momento (ya sea WiFi, cable o 3G) y mostrará el mapa centrado en dicho lugar. Esto a su vez llevará a cabo una búsqueda con todas las rutas disponibles en la latitud y longitud del mapa que se visualiza. Así mismo y pulsando y arrastrando sobre el mapa nos desplazaremos en el mismo y automáticamente el listado de rutas de

la derecha se actualizará con las rutas disponibles.

Además del filtro indicado por la posición en el mapa se puede realizar un filtro por dificultad y por tipo de ruta realizada, bmx, mountain bike, carretera y el resto de opciones disponibles.

Cada una de las rutas visualizada en el listado será un enlace al detalle de la misma.

### 7.2.20. Detalle de una ruta

Desde el listado de rutas global o desde el listado de rutas de un usuario se puede acceder a un detalle de la misma.

En dicho detalle se visualiza un mapa con la ruta sobreimpresionada así como una gráfica de la orografía del terreno de la ruta. También se

mostrará la dificultad asociada a la ruta así como la distancia si disponemos de dicho dato y de la modalidad de la ruta realizada.

También dispondremos de una descripción más detallada de la misma y la posibilidad de añadir comentarios asociadas a la ruta visualizada.

### 7.2.21. Descargar fichero de ruta

Como usuario registrado y desde el detalle de una ruta disponemos de la posibilidad de descargarnos un fichero GPX de la misma. Para ello sólo deberemos pulsar en el enlace habilitado para la descarga del mismo y automáticamente nos aparecerá la opción de descarga del fichero.

### 7.2.22. Comentar una ruta

Como usuario registrado y desde el detalle de una ruta se pueden realizar comentarios en la misma. Para ello en la parte inferior de la misma tras la descripción aparecerá un listado de todos los comentarios escritos hasta el momento así como un formulario para escribir nuestro comentario.

The screenshot shows a user interface for commenting on a route. At the top, there is a text input field containing the message: "Tengo ganas de tener un fin de semana libre para realizar esta ruta que nos has ofrecido, felicidades !". Below the input field, a red text placeholder says "Te quedan 152 caracteres". To the right of the input field is a red "Publicar" button. Below the comment input, there is a list of existing comments. The first comment is by a user named "Alejandro Lopez" (with a small profile picture) and was posted "hace 3 horas". The comment text is: "Menuda descripción de ruta que nos has ofrecido ! así da gusto, sigue así !".

*Nota: Se puede dar el caso de que la ruta no acepte comentarios por lo que este punto está supeditado a que el usuario que ha creado la ruta acepte o no comentarios en la misma.*

### 7.2.23. Ver eventos de un usuario

The screenshot shows a user profile interface titled 'Mis eventos' (My events). At the top right is a green button labeled 'Nuevo evento'. Below it, a sub-header says 'Todos los eventos que has creado' (All events you have created). A small icon of a bicycle and a map pin is shown next to event details. The event listed is 'Presentación de Voy En Bici' dated '22/05/2015' at 'UO - Campus de Gijón, Calle Pedro Puig Adam, 33204 Gijón, Asturias, España'. A note below states 'Estamos preparando la presentación del portal Voy en Bici que se lanzará a finales del verano a disposición de los usuarios.' At the bottom left is a blue star icon with the text 'Desmarcar de favoritos'. To the right are edit and delete icons.

Desde nuestro perfil de usuario o desde el perfil de otro usuario se puede acceder a un listado de los eventos creados por dicho usuario.

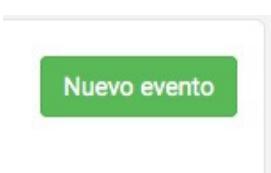
Se visualizará un listado con todos los eventos creados en orden de ejecución del evento

en orden ascendente, los eventos caducados no se mostrarán. Aparecerá el nombre del evento junto a la fecha de ejecución del mismo, un breve resumen junto a su localización y una imagen estática de la posición del evento sobre un mapa. Toda ésta información será un enlace al detalle de cada uno de los eventos.

Se dispone así mismo de un enlace para la creación de un nuevo evento o para editar o eliminar el evento seleccionado.

### 7.2.24. Dar de alta un evento

Como usuarios registrados de la plataforma disponemos de la posibilidad de dar de alta nuevos eventos en la aplicación.



Para ello disponemos de dos puntos iniciales de acceso a la creación de eventos, desde el listado completo de eventos de la aplicación al que se puede acceder desde el menú superior en el ícono asociado a los eventos o desde el listado de mis eventos en el menú de usuario.

En ambos casos nos aparecerá un botón de *Nueva Evento* que nos facilitará la creación del mismo en el sistema. El formulario de creación y edición de evento es el mismo así que pasaremos a explicarlo en el próximo punto.

## 7.2.25. Edición de Evento

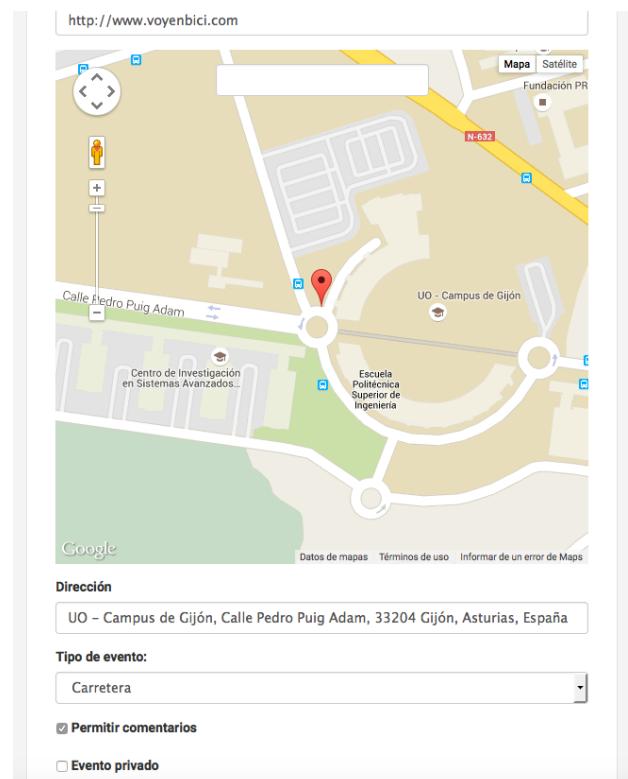
Una vez que hemos creado nuestro evento tenemos la posibilidad de editarlo para realizar los ajustes necesarios en la información que se muestra asociado al evento.



Para ello se puede acceder a la edición del evento desde el listado de todas los eventos disponibles o

desde el menú de usuario *Mis eventos*. En ambos casos siempre tiene que coincidir que seamos el propietario del evento o tengamos permisos de edición sobre el mismo.

Una vez que pulsemos sobre el botón de editar accederemos al formulario de edición de evento donde podremos modificar toda la información disponible de la misma así como el punto del mapa donde se realizará el evento.



## 7.2.26. Eliminar evento

Igualmente y en el mismo caso que para la edición de un evento disponemos de la posibilidad de eliminar un evento compartido previamente en la aplicación.

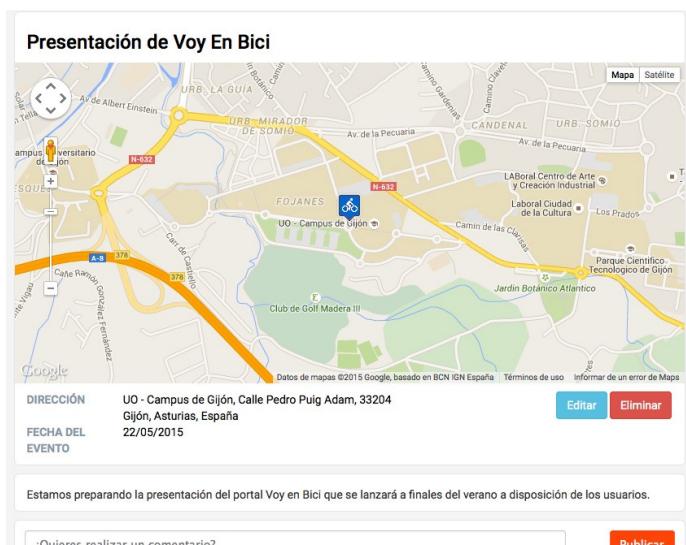


Para ello y desde el mismo acceso que en el caso anterior podemos eliminar el evento seleccionado. Al pulsar en el enlace de borrado se nos pedirá confirmación del paso que vamos a realizar ya que no podremos recuperar dicha información.

## 7.2.27. Ver detalle de un evento

Desde el listado de eventos global o desde el listado de eventos de un usuario se puede acceder a un detalle del mismo.

En dicho detalle se visualiza un mapa con la posición del evento geolocalizado así como otra serie de datos asociados al mismo como son la fecha del evento, título, descripción, dirección y cualquier otro dato de interés.



## 7.2.28. Comentar un evento

Como usuario registrado y desde el detalle de un evento se pueden realizar comentarios en el mismo. Para ello en la parte inferior del evento y tras la descripción aparecerá un listado de todos los comentarios escritos hasta el momento así como un formulario para escribir nuestro comentario.

Gijón, Asturias, España  
FECHA DEL EVENTO 22/05/2015

Estamos preparando la presentación del portal Voy en Bici que se lanzará a finales del verano a disposición de los usuarios.

Que ganas de participar en la presentación !!!!

Te quedan 208 caracteres

Publicar

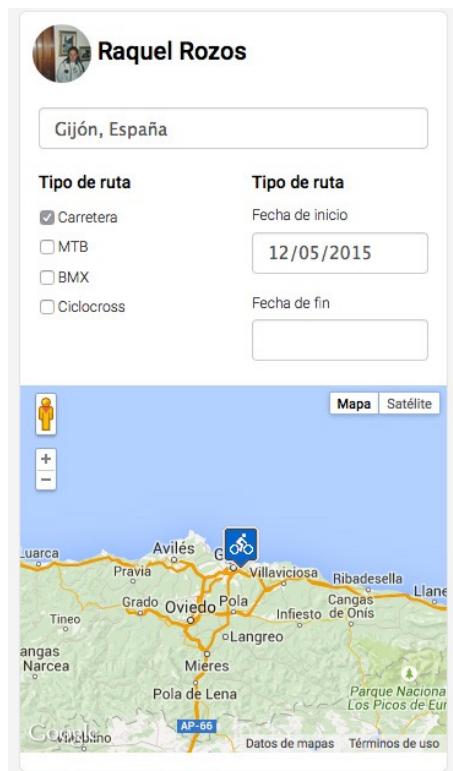
*Nota: Se puede dar el caso de que el evento no acepte comentarios por lo que este punto está supeditado a que el usuario que ha creado el evento acepte o no comentarios.*

## 7.2.29. Buscar eventos

Desde el menú superior indicado en el punto 7.2.1. el usuario dispone de un acceso al listado de todos los eventos disponibles en el sistema.

El usuario puede aceptar que el navegador averigüe su posición y muestre los eventos cercanos a su posición, de todas maneras el usuario puede pulsar sobre el mapa y arrastrarlo de manera que varíe dicha posición y de dicha manera los eventos visualizados.

También podrá realizar un filtro por las fechas de disponibilidad del evento. Todos los eventos mostrados incluyen un enlace al detalle de cada evento.



## 7.2.30. Mis favoritos

Como usuario registrado disponemos la posibilidad de añadir rutas y eventos a un listado de favoritos. Para ello y desde el menú izquierdo de usuario podemos acceder al listado de todos los favoritos que hayamos almacenado.

Nombre del favorito	Tipo	Opciones
Renfe Oviedo - P.R. AS-183 Ruta Priañes (Ruta de los Meandros del Noreste)	Ruta	Desmarcar de favoritos
Gijón - Fontaciera - Varé - Túnel De Conixhu (Boca Sur) - Palacio De Cell Jones	Ruta	Desmarcar de favoritos

En dicho listado tendremos un acceso al detalle de cada evento o ruta que hayamos añadido previamente.

En dicho listado también dispondremos de la posibilidad de eliminar de favorito el elemento seleccionado.

## 7.2.31. Añadir un elemento a favoritos

Desde el listado de elementos o desde el detalle de los mismos un usuario dispone la posibilidad de indicar que el elemento seleccionado se incluya dentro de sus favoritos.

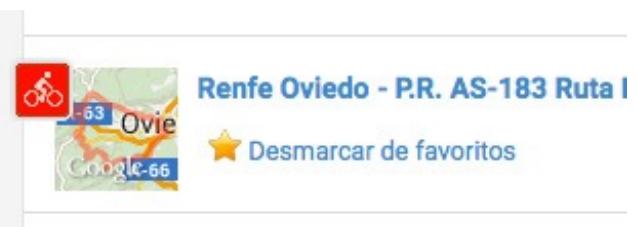
A posteriori aparecerá en nuestro listado de **Mis favoritos**.

Nombre del favorito	Opciones
Gran circular Gijón, Peón, Niévares, Villaviciosa	Marcar como favorito

### 7.2.32. Quitar un elemento de favoritos

Desde el listado de elementos favoritos o desde el listado o detalle de uno de los elementos ya seleccionado previamente como favorito se puede eliminar de dicho listado.

Una vez eliminado de favoritos se eliminará del listado de mis favoritos indicado en el punto **Mis favoritos**.



### 7.2.33. Formulario de contacto

Al pulsar en el enlace de contacto aparecerá un formulario para que se rellene cierta información obligatoria con el comentario a realizar por parte del usuario al departamento administrativo de la plataforma.

Este formulario enviará un correo electrónico al que el departamento técnico dará respuesta lo antes posible.

Datos de la consulta

Si desea ponerte en contacto con nosotros por favor rellene el siguiente formulario y nos pondremos en contacto tan pronto como sea posible.

Nombre

Email

Asunto

Comentarios

Acepto la política de privacidad

Enviar

Recordar contraseña

## 7.3. Acceso como usuario Administrador

Como usuario administrador dispondremos de un acceso completo a todos los elementos disponibles en la plataforma, tanto usuarios, comentarios, rutas y eventos como otros elementos que se incluyan a posteriori en la aplicación.

El acceso a dicha gestión se realizará desde nuestro propio usuario pulsando sobre **Opciones de Administración**.

### 7.3.1. Pantalla inicial usuario administrador

La primera pantalla a la que se accede tras acceder a opción de administrador también se denomina *Escritorio* y consiste en un primer vistazo rápido de los últimos cinco elementos que se han producido de cada caso. Tendremos un listado con los últimos cinco usuarios registrados, los últimos cinco comentarios realizados, eventos y rutas creadas.

Comentario	Usuario	Fecha de creación	Opciones
Hoy he disfrutado como una enana de la última ruta que he hecho, os recomiendo a todos que visitéis los Oscos en primavera :)	Raquel Rozos (raquel.rozos@voyenbici.com)	12 de mayo de 2015 17:40	
Menuda descripción de ruta que nos has ofrecido ! así da gusto, sigue así !	Alejandro Lopez (alejandro.lopez@voyenbici.com)	09 de mayo de 2015 14:30	
Que ganas de que llegue la presentación !	Alejandro Lopez (alejandro.lopez@voyenbici.com)	09 de mayo de 2015 14:28	
Desde luego era bueno que el partido político de turno mejorase las condiciones de acceso a la ruta que comienza en Villaviciosa !! Como se nota que ellos no hacen uso de esas carreteras !!!!	Raquel Rozos (raquel.rozos@voyenbici.com)	19 de abril de 2015 17:46	

Nombre	Correo electrónico	Fecha de creación	Opciones
Pleayo Otero	pelayo.oter@voyenbici.com	19 de abril de 2015 09:20	
David Muríz	david.muniz@voyenbici.com	19 de abril de 2015 09:20	
Jorge Lopez	jorge.lopez@voyenbici.com	19 de abril de 2015 09:19	
Sergio Iglesias	sergio.iglesias@voyenbici.com	19 de abril de 2015 09:19	
Miguel Indurán	miguel.indurain@voyenbici.com	19 de abril de 2015 09:06	

Desde dichos listados tendremos una rápida visualización de lo último ocurrido y podremos acceder a editar o eliminar los mismos desde los enlaces indicados para cada elemento.

### 7.3.2. Acceso a listado completo de usuarios

Desde el menú lateral izquierdo tendremos acceso al listado completo de usuarios de la aplicación. Se obtendrán todos los usuarios ordenados por su nombre y tendremos disponibles de un primer vistazo su nombre, email y fecha en la que han sido creados en la aplicación.

En la parte superior de dicho listado dispondremos de un buscador que nos permitirá filtrar a los usuarios por nombre, email y la fecha de inicio / fin en la que han sido creados. También disponemos de la posibilidad de descargarnos la búsqueda realizada en formato CSV o Excel.

Usuarios				
Buscador de usuarios				
Introduzca nombre		Introduzca email	Introduzca Fecha de in	Introduzca Fecha de fi
Gestión de usuarios				
Nombre	Correo electrónico	Fecha de creación	Opciones	
Alberto Contador	alberto.contador@voyenbici.com	19 de abril de 2015 07:38		
Alejandro Lopez	alejandro.lopez@voyenbici.com	15 de febrero de 2015 20:51		
David Muniz	david.muniz@voyenbici.com	19 de abril de 2015 09:20		
Jorge Lopez	jorge.lopez@voyenbici.com	19 de abril de 2015 09:19		
Miguel Induráin	miguel.indurain@voyenbici.com	19 de abril de 2015 09:06		
Pelayo Otero	pelayo.oter@voyenbici.com	19 de abril de 2015 09:20		
Raquel Rozos	raquel.rozos@voyenbici.com	19 de abril de 2015 08:59		
Sergio Iglesias	sergio.iglesias@voyenbici.com	19 de abril de 2015 09:19		

### 7.3.3. Editar datos de usuario

Desde el listado de usuarios y pulsando sobre el enlace de edición de uno de ellos accedemos a un formulario donde disponemos de todos los datos actuales del usuario seleccionado.

Desde dicho formulario podemos realizar la modificación de los datos permitidos y tras pulsar en el botón de actualizar se almacenarán los nuevos datos seleccionados para el usuario.

Edición de usuario

Edición de usuario Alejandro Lopez

Nombre	Alejandro Lopez	
Correo electrónico	alejandro.lopez@voyenbici.com	
Contraseña	<input type="text"/> Introduzca contraseña	Repetir contraseña
Página web	<a href="http://www.voyenbici.es">http://www.voyenbici.es</a>	
Sobre mí	Preparando el proyecto fin de carrera !	
Foto		
<input type="button"/> Examinar...   No se ha seleccionado ningún archivo.		
<input type="button"/> Guardar		
<a href="#">« Volver</a>		

### 7.3.4. Eliminar usuario

Desde el listado de usuarios disponemos la posibilidad de eliminar a un usuario seleccionado. Tras pulsar en el botón de eliminar se nos solicitará confirmación del borrado, si aceptamos tras la pregunta de confirmación se hará un borrado en cascada de todos los eventos, rutas, comentarios y conversaciones que dispusiese el usuario en la plataforma.

### 7.3.5. Acceso a listado completo de Comentarios

Desde el menú lateral izquierdo tendremos acceso al listado completo de los comentarios que se están realizando en la aplicación. Se obtendrán todos los comentarios ordenados por fecha de creación en orden descendente y se indicará el comentario realizado así como el usuario que lo ha realizado junto a la fecha y hora de creación del mismo.

En la parte superior de dicho listado dispondremos de un buscador que nos permitirá filtrar los comentarios a partir del usuario que los ha realizado. Así mismo disponemos de la posibilidad de descargarnos la búsqueda realizada en formato CSV o Excel.

Comentarios			
Buscador de comentarios		Descargar CSV Descargar Excel	
Introduzca login o ema		<input type="checkbox"/> Marcado como ofensivo	Buscar
Gestión de comentarios			
Comentario	Usuario	Fecha de creación	Opciones
Hoy he disfrutado como una enana de la última ruta que he hecho, os recomiendo a todos que visitéis los Oscos en primavera :)	Raquel Rozos (raquel.rozos@voyenbici.com)	12 de mayo de 2015 17:40	 
Menuda descripción de ruta que nos has ofrecido ! así da gusto, sigue así !	Alejandro Lopez (alelop3z@gmail.com)	09 de mayo de 2015 14:30	 
Que ganas de que llegue la presentación !	Alejandro Lopez (alelop3z@gmail.com)	09 de mayo de 2015 14:28	 
Desde luego era bueno que el partido político de turno mejorase las condiciones de acceso a la ruta que comienza en Villaviciosa !! Como se nota que ellos no hacen uso de esas carreteras !!!	Raquel Rozos (raquel.rozos@voyenbici.com)	19 de abril de 2015 17:46	 

### 7.3.6. Editar comentario

Desde el listado de comentarios y pulsando sobre el enlace de edición de uno de ellos accedemos a un formulario donde disponemos del comentario que ha realizado el usuario.

Desde dicho formulario podemos realizar la modificación de los datos permitidos y tras pulsar en el botón de actualizar se almacenarán los nuevos datos seleccionados para el comentario elegido.

Edición de comentario

Edición de comentario realizado por Raquel Rozos (raquel.rozos@voyenbici.com)

Comentario

Desde luego era bueno que el partido político de turno mejorase las condiciones de acceso a la ruta que comienza en Villaviciosa !! Como se nota que ellos no hacen uso de esas carreteras !!!

¿Comentario ofensivo?

[« Volver](#)

### 7.3.7. Eliminar comentario

Desde el listado de comentarios disponemos la posibilidad de eliminar el comentario realizado por un usuario seleccionado. Tras pulsar en el botón de eliminar se nos solicitará confirmación del borrado, si aceptamos tras la pregunta de confirmación se hará el borrado de dicho comentario y no se podrá recuperar la información del mismo.

### 7.3.8. Acceso a listado completo de Rutas

Desde el menú lateral izquierdo tendremos acceso al listado completo de las rutas que se han creado en la aplicación. Se obtendrán todos las rutas ordenadas por fecha de creación en orden descendente y se indicará el nombre de la ruta, la dificultad de la misma y si es privada o no junto a la fecha de creación.

En la parte superior de dicho listado dispondremos de un buscador que nos permitirá filtrar las rutas a partir del nombre de la misma, localización y la dificultad asociada. Así mismo disponemos de la posibilidad de descargarnos la búsqueda realizada en formato CSV o Excel.

Rutas

Buscador de rutas		Descargar CSV		Descargar Excel	
Introduzca nombre	Introduzca tipo de ruta	Introduzca la dificultad	Introduzca si es privada o no	Buscar	
Gestión de rutas					
Nombre	Route type	Dificultad	Ruta privada	Fecha de creación	Opciones
Gijón - Fontaciera - Varé - Túnel De Conixhu (Boca Sur) - Palacio De Celles - Muncó - Granda - Gijón 62	Carretera	Difícil	No	19 de abr 17:37	
Renfe Oviedo - P.R. AS-183 Ruta Priañes (Ruta de los Meandros del Nora) - Escamplero - Renfe Oviedo 62	BMX	Muy fácil	No	19 de abr 17:23	
Gran circular Gijón, Peón, Niévares, Villaviciosa, el Puntal y vuelta por carreteras costeras 56	BMX	Muy fácil	No	16 de mar 19:59	

### 7.3.9. Editar ruta

Desde el listado de rutas y pulsando sobre el enlace de edición de una de ellas accedemos a un formulario donde disponemos de la posibilidad de realizar ajustes en la ruta seleccionada.

Desde dicho formulario podemos realizar la modificación de los datos permitidos y tras pulsar en el botón de actualizar se almacenarán los nuevos datos seleccionados para la ruta elegida.

Editar ruta

Editor ruta creada por Alejandro Lopez (alelop3z@gmail.com)
Nombre Avenida Finisterre, La Coruña
Descripción Mi última ruta de entrenamiento rodador antes de la maratón de los monegros la hice de la mano de Rober, de Raposoos BTT.

### 7.3.10. Eliminar ruta

Desde el listado de rutas disponemos la posibilidad de eliminar una ruta seleccionada pulsando en el botón eliminar disponible donde se nos solicitará confirmación del borrado, si aceptamos tras la pregunta de confirmación se hará el borrado de dicha ruta y de los comentarios asociados a la misma.

### 7.3.11. Acceso a listado completo de Eventos

Desde el menú lateral izquierdo tendremos acceso al listado completo de los eventos almacenados en la aplicación. Se obtendrán todos los eventos ordenados por fecha de creación en orden

descendente y se indicará el nombre del evento, tipo del mismo, fecha de inicio, localización, privacidad y los botones de administración correspondientes.

En la parte superior de dicho listado dispondremos de un buscador que nos permitirá filtrar los eventos a partir de la localización, nombre, tipo de evento, privacidad y fechas. Así mismo disponemos de la posibilidad de descargarnos la búsqueda realizada en formato CSV o Excel.

Eventos											
<p>Buscador de eventos</p> <input type="text"/> Introduzca título <input type="text"/> Introduzca tipo de evento <input type="text"/> Público <input type="text"/> 21/05/2015 <input type="text"/> Introduzca fecha de fin <input type="button"/> Buscar <input type="button"/> Descargar CSV <input type="button"/> Descargar Excel											
<p>Gestión de eventos</p> <table border="1"><thead><tr><th>Nombre</th><th>Tipo de evento</th><th>Fecha de inicio</th><th>Opciones</th></tr></thead><tbody><tr><td>Presentación de Voy En Bici</td><td>Carretera</td><td>22 de mayo de 2015 12:30</td><td><input checked="" type="checkbox"/> <input type="checkbox"/></td></tr></tbody></table>				Nombre	Tipo de evento	Fecha de inicio	Opciones	Presentación de Voy En Bici	Carretera	22 de mayo de 2015 12:30	<input checked="" type="checkbox"/> <input type="checkbox"/>
Nombre	Tipo de evento	Fecha de inicio	Opciones								
Presentación de Voy En Bici	Carretera	22 de mayo de 2015 12:30	<input checked="" type="checkbox"/> <input type="checkbox"/>								

### 7.3.12. Editar evento

Desde el listado de eventos y pulsando sobre el enlace de edición de uno de ellos accedemos a un formulario donde disponemos de la posibilidad de realizar modificaciones sobre los datos permitidos. Tras pulsar en el botón de actualizar se almacenarán los nuevos datos seleccionados para el evento elegido.

Edición de evento	
<p>Edición de evento creado por Alejandro Lopez (alelop3z@gmail.com)</p> <p>Nombre</p> <input type="text"/> Presentación de Voy En Bici	
<p>Resumen</p> <p>Estamos preparando la presentación del portal Voy en Bici que se lanzará a finales del verano a disposición de los usuarios.</p>	
<p>Descripción</p> <p>Estamos preparando la presentación del portal Voy en Bici que se lanzará a finales del verano a disposición de los usuarios.</p>	
<p>Fecha de inicio</p> <input type="text"/> 22/05/2015	
<p>Introduce una ubicación</p> 	

### 7.3.13. Eliminar evento

Desde el listado de eventos disponemos la posibilidad de eliminar el evento seleccionado. Tras pulsar en el botón de eliminar se nos solicitará confirmación del borrado, si aceptamos tras la pregunta de confirmación se hará el borrado de dicho evento y de los comentarios asociados siendo imposible la recuperación de información del mismo.



# 8. Manual del programador

---

## 8.1. Instalación de la plataforma Voy en Bici

### 8.1.1. Requisitos Hardware

No es necesario ningún requisito hardware específico para que la plataforma Voy en Bici funcione correctamente, sólo los requisitos que tenga el sistema operativo escogido para la instalación.

### 8.1.2. Requisitos Software

Para el correcto funcionamiento de la plataforma Voy en Bici existen una serie de requisitos software que debe cumplir el sistema operativo donde se va a instalar la aplicación. Son totalmente necesarios ya que sin ellos no se podría utilizar la aplicación.

El software que necesitaremos instalar es el aquí indicado:

- Ruby 2.2.2 o superior: Intérprete del lenguaje de programación usado.
- Rails 4.2.0 o superior: Framework web de la aplicación
- MongoDB 3.0.3 o superior: Gestor de base de datos utilizado.
- Nginx
- RVM: Control de versiones Ruby
- Phusion Passenger: Gema que interactúa entre Nginx y Ruby
- Mina: gestor de despliegues automatizados

Podemos descargar el software en caso de máquinas Linux desde los propios repositorios indicados durante la instalación o para cualquier sistema operativo en las páginas aquí indicadas:

- Ruby 2.2.2 o superior: Página oficial de Ruby [1]
- Rails 4.2.0 o superior: Página oficial de Rails [2]
- MongoDB 3.0.3 o superior: Página oficial de MongoDB [3]
- Nginx: Página oficial de Nginx [4]
- RVM: Página oficial de RVM [5]
- Phusion Passenger: Página oficial de Phusion Passenger
- Mina: Página del proyecto en Github

### 8.1.3. Instalación

A continuación se indican los pasos de la instalación de la plataforma Voy en Bici sobre un sistema operativo basado en Debian Wheezy 7 64 bits, siendo muy similar en sus pasos para una máquina con otra distribución Linux (Ubuntu, RedHat, ...) o en ordenadores con otros sistemas operativos como puedan ser OS X o Windows.

Durante toda la instalación consideramos que el usuario que estamos usando es un superusuario u otro distinto con permisos totales en el sistema. El paso inicial se corresponde desde la carpeta inicial del usuario.

*Nota: Todas las instrucciones aquí ejecutadas han sido realizadas con el usuario root, en caso de realizar con otro usuario con permisos sudo será necesario añadir la instrucción sudo por delante de la descrita en este documento.*

En caso de necesitar movernos de carpeta se indicará en las distintas instrucciones aquí mostradas.

## 8.2. Guía de instalación paso a paso

### 8.2.1. Actualización de repositorios de nuestro sistema

El primer paso será realizar una actualización de todos los repositorios disponibles en nuestro sistema para así hacer uso de las librerías más recientes disponibles.

```
apt-get update
```

En máquinas con Windows no será necesario este paso y en sistemas OS X se recomienda la instalación de un sistema de paquetes para facilitar la labor de instalación. Los dos más conocidos o usados por la comunidad son Homebrew (<http://brew.sh/>) y MacPorts (<https://www.macports.org/>) de los cuales sólo deberemos seguir las instrucciones detalladas en sus respectivas páginas web para realizar la instalación en el sistema.

### 8.2.2. Instalar una versión de Ruby

Para la instalación de Ruby vamos a optar por hacer uso de un gestor de versiones de Ruby el cual a posteriori nos permitirá manejar distintas versiones del código en un mismo sistema.

Las dos opciones más conocidas durante la realización de esta documentación son Rbenv y RVM, siendo RVM (<https://rvm.io/>) de la que haremos uso en este caso, aunque si se optase por usar Rbenv (<http://rbenv.org/>) los pasos a seguir son similares ya que el funcionamiento y filosofía usadas son muy parejas.

*NOTA: En caso de no disponer de curl instalado en el sistema deberemos ejecutar previamente*

```
apt-get update curl
```

La instrucción para instalar RVM en nuestro sistema es:

```
curl -L https://get.rvm.io | bash -s stable --rails
```

Una vez terminada la instalación de RVM ya dispondremos de una versión de Ruby instalada en nuestro sistema, en este caso se corresponde con la última versión disponible en el repositorio en el momento de la creación de la documentación (Ruby 2.2.1) pero para poder hacer uso inmediatamente de ella en nuestro sistema deberemos reiniciar nuestro sistema o reiniciar el entorno de nuestro usuario

```
source ~/.rvm/scripts/rvm
```

Si llegado el caso nos interesase instalar una nueva versión de Ruby debido a que necesitemos alojar en el servidor de producción o en nuestra propia máquina varias versiones podremos ver las versiones instaladas con la instrucción

```
rvm list
```

Seleccionar una versión de las instaladas con la instrucción

```
rvm use 2.2.2 # número de versión a usar
```

O instalar y borrar distintas versiones de Ruby

```
rvm install 2.2.2  
rvm remove 2.2.2
```

### 8.2.3. Instalación de MongoDB

El siguiente paso antes de arrancar nuestro proyecto será hacer la instalación de nuestro gestor de Base de Datos. En este caso hemos optado por MongoDB siendo posible una migración sencilla a otros sistemas de bases de datos realizando ciertos ajustes en el código distribuido.

Los pasos que seguiremos están indicados en el propio manual de instalación de MongoDB (<http://docs.mongodb.org/manual/tutorial/install-mongodb-enterprise-on-debian/>) donde vienen indicados además para distintos sistemas operativos, sólo deberemos elegir el nuestro y seguir paso a paso lo allí indicado.

Por facilitar la labor indicaremos los 3 pasos que se deben realizar en nuestro servidor de producción.

Añadir el repositorio de MongoDB en nuestro sistema e instalar MongoDB, en este caso la última versión estable que es la versión 3.0.3.

```
apt-key adv --keyserver keyserver.ubuntu.com --recv 7F0CEB10  
echo 'deb http://downloads-distro.mongodb.org/repo/debian-sysvinit dist  
10gen' | tee /etc/apt/sources.list.d/mongodb.list  
apt-get update  
apt-get install -y mongodb-org
```

Nuestro siguiente punto será comprobar el arranque de nuestro demonio de base de datos. MongoDB almacena toda la información del sistema en una carpeta específica según el sistema operativo donde se instale.

Por defecto dicha carpeta es **/var/lib/mongodb** siendo posible modificarla. Por comodidad y facilidad de uso no haremos ningún tipo de modificación en este caso almacenando todos nuestros datos en dicha carpeta. Será muy importante comprobar que el usuario que arranque el programa tenga permisos de lectura / escritura en dicha carpeta.

```
service mongod start # Arrancar el demonio de MongoDB  
service mongod stop # Detener el demonio de MongoDB
```

En caso de que todo haya ido bien se nos indicará que la base de datos parece estar funcionando con normalidad.

Las instrucciones básicas disponibles tras la instalación de nuestro gestor de base de datos para comprobar que esté funcionando correctamente.

```
mongo
```

Debería salir información relativa a Mongo y una pequeña consola donde poder ejecutar diferentes instrucciones de nuestro sistema de base de datos

Para realizar copias de seguridad o restaurar un backup disponible disponemos de las instrucciones

```
mongodump -d basededatos      # Hacer un backup de la base de datos elegida  
mongorestore basededatos     # Restaurar un backup disponible
```

#### 8.2.4. Arrancando Voy en Bici

Una vez instalado Ruby y MongoDB en nuestro sistema nos desplazaremos a la carpeta donde tenemos nuestro código fuente.

En este caso se ha optado por almacenarlo en la carpeta /var/www/voyenbici siendo este aspecto indiferente, los únicos aspectos a tener en cuenta serán tener permisos suficientes para el usuario que ejecute los procesos en dicha carpeta.

```
cd /var/www/voyenbici
```

Una vez dentro de la carpeta inicial de nuestro proyecto y habiendo elegido previamente la versión de Ruby que queremos ejecutaremos la instrucción

```
bundle install
```

la cual se encarga de instalar todas las gems que nuestro proyecto necesita para funcionar.

Por último ya podremos arrancar el proyecto con la instrucción

```
rails s
```

la cual arranca por defecto nuestro proyecto en el puerto 3000. Podemos acceder con cualquier navegador web a la url <http://localhost:3000> para visualizarlo y empezar a trabajar con él.

*Nota: Aquí terminarían los pasos a ejecutar en caso de que instalemos el proyecto en un equipo local o de desarrollo. Los pasos indicados a continuación habrá que realizarlos para un paso a producción.*

### 8.2.5. Instalación de Nginx

En caso de realizar la instalación en un servidor web en producción y querer que disponga de visibilidad para todos los usuarios debemos instalar un servidor web.

Los dos más conocidos y usados en este momento son Apache (<http://httpd.apache.org/>) y Nginx (<http://nginx.com/>) siendo ambos de ellos gratuitos aunque disponen de productos de pago

Vamos a realizar la instalación y configuración de Nginx el cual dispone de algunas mejoras en cuanto a número de peticiones concurrentes respecto a Apache aunque su configuración a alto nivel es algo más compleja.

El primer punto será la instalación de la **gem passenger**, la cual se encarga de hacer la conversión de la petición web a Ruby on Rails y servirla a través de Nginx / Apache

```
gem install passenger
```

Y ejecutaremos la instrucción que instala Nginx junto a dicho módulo ya configurado y listo para usar

```
passenger-install-nginx-module
```

Durante la instalación pueden ser necesario instalar nuevas dependencias. Todo ello dependerá de las librerías instaladas en el sistema.

Una vez que termine se habrá realizado la instalación de nuestro servidor web. Sólo queda configurar que responda a la carpeta /var/www/voyenbici donde está alojado nuestro código fuente.

Para ello dejaremos los valores incluidos tras la instalación y nos centraremos en la configuración para que el sistema reconozca a qué dominio debe responder y donde está situada nuestra aplicación.

*Nota: La configuración estándar de Nginx está bien para las pruebas que realizaremos pero una configuración acorde a las capacidades de nuestro sistema es un punto crítico en el funcionamiento de una aplicación web. Como esta configuración depende de muchos factores es un estudio que habrá que realizar sobre cada máquina concreta.*

Nuestro primer paso será crear un fichero de VirtualHost de manera que nuestra máquina sepa en qué carpeta se encuentra el código que responde al dominio que dispongamos.

Para crear este fichero nos debemos desplazar a la carpeta sites-available de Nginx donde están todos los VirtualHost que podemos disponer.

```
cd /etc/nginx/sites-available
nano voyenbici
```

Y añadimos el siguiente código

```
server {
    listen IP:80;
    server_name DOMINIO;
    root CARPETA_PROYECTO/public/;
    client_max_body_size 200M;
    access_log  /var/log/nginx/DOMINIO.access.log;
    error_log   /var/log/nginx/DOMINIO.error.log;
    passenger_enabled on;
    passenger_min_instances 4;
    passenger_ruby /usr/local/rvm/gems/RUBY_GEMSET/wrappers/ruby;
    rails_env production;

    location ^~ /assets/ {
        gzip_static on;
        expires max;
        add_header Cache-Control public;
    }
}
```

Indicamos a que se corresponde cada valor a sustituir en el fichero de configuración:

Elemento	Descripción
IP	Se corresponde a la IP de la máquina donde estará alojado nuestro proyecto en la red
DOMINIO	A que URL responderá nuestro servidor
CARPETA_PROYECTO	La carpeta donde está situado nuestro proyecto, en este caso en /var/www/voyenbici
RUBY_GEMSET	La versión de Ruby y Gemset seleccionados en RVM, en este caso ruby-2.2.2@voyenbici

El siguiente paso será activar el VirtualHost que acabamos de crear. Para ello nos desplazaremos a la carpeta donde se encuentran los VirtualHosts activos y lo enlazaremos

```
cd /etc/nginx/sites-enabled  
ln -s /etc/nginx/sites-available/voyenbici voyenbici
```

Por último reiniciamos Nginx y si probasemos ya deberíamos acceder a nuestra página inicial del proyecto

```
/etc/init.d/nginx restart
```

### 8.2.6. Despliegues automatizados

Para este proyecto en concreto se ha añadido una funcionalidad muy útil que son los despliegues automatizados permitiendo que mediante el uso de una o varias instrucciones sencillas nos permitirá realizar un despliegue de la última versión de nuestro código de manera que no tengamos que ir fichero a fichero revisando cual o cuales hemos modificado y subiendo dichos ficheros manualmente por FTP.

En este caso concreto hemos usado la utilidad Mina (<https://github.com/mina-deploy/mina>) existiendo nuevamente varias versiones parecidas como puedan ser Capistrano (<https://github.com/capistrano/capistrano>).

En este documento accederemos sólo a aquellos puntos básicos e iniciales de los despliegues para dar una pequeña pincelada de su funcionamiento.

Por ejemplo un fichero de configuración consta de los siguientes aspectos siendo los aquí indicados los puntos más importantes y sustituidos por los valores que correspondan en cada caso

```
set :domain, DOMINIO
set :deploy_to, RUTA_DESPLIEGUE
set :repository, REPOSITORIO (SVN, GitHub, Mercurial, ...)
set :branch, RAMA_DEL_REPOSITORIO
set :rails_env, ENVIRONMENT_PROYECTO (development, test, production)
set :user, USUARIO_SISTEMA
set :port, PUERTO_ACCESO_SISTEMA
```

Para el despliegue inicial deberemos acceder desde nuestro equipo local donde estamos desarrollando el proyecto.

```
cd /Users/alejandrolopeznunez/Proyectos/Rails/pfc
```

*Nota: la carpeta aquí indicada se corresponde con mi carpeta de desarrollo, habría que sustituirla por la propia de cada usuario*

Dentro de nuestra carpeta de usuario tendremos la opción de hacer el despliegue inicial mediante la instrucción

```
mina setup
```

La cual nos creará una estructura de carpetas dentro de nuestro servidor en la carpeta de despliegue siendo las más interesantes las carpetas releases, shared y current.

La carpeta shared contiene aquellos elementos que serán obviados en cada despliegue automatizado porque se tienen que mantener inalterables en dichos despliegues, se puede considerar por ejemplo las imágenes subidas por los usuarios a nuestro proyecto ya que son elementos que nuestro código no puede eliminar con cada despliegue.

La carpeta releases contiene una serie de carpetas, cada una de ellas se corresponde con cada uno de los despliegues que hayamos realizado.

Cada vez que hacemos un despliegue automatizado se creará una nueva carpeta dentro de releases y una vez terminado se modificará el enlace simbólico current que pasará a apuntar al último despliegue realizado.

Todos estos pasos son automáticos con cada despliegue y nos facilitan el volver a una versión anterior modificando el enlace simbólico o haciendo nosotros mismos una tarea en la propia configuración del fichero deploy.rb asociado a la gema mina.

La instrucción de despliegue es

```
mina deploy
```

Y una vez terminada la ejecución dispondremos de la última versión de nuestro código en el servidor sobre el que se realiza el despliegue, funcionando por tanto la última versión disponible en el repositorio utilizado y visible a nuestros usuarios.

## 8.3. Desarrollo del back-end

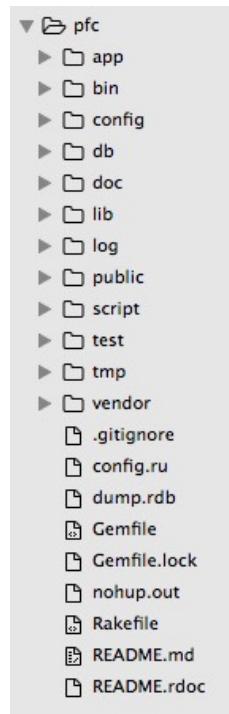
En este punto se explicarán *conceptos básicos de uso del framework Ruby on Rails*, información sobre la *estructura de carpetas proporcionada en el código fuente* y el *ciclo de vida* por el que transcurre una petición realizada.

### 8.3.1. Estructura de carpetas

El primer punto en el que incidiremos será la estructura de carpetas que proporciona Ruby on Rails a la hora de crear un proyecto y que nos obliga a trabajar de una manera ordenada y eficiente.

**app:** Consiste en la carpeta donde se encontrará el código principal de la aplicación. En este punto accederemos a tanto hojas de estilo, javascripts, imágenes, controladores, modelos y vistas de la aplicación. Se divide a su vez en varias subcarpetas:

- **assets:** contiene toda la estructura de hojas de estilo, imágenes y javascript.
- **controllers:** controladores de la aplicación, son los ficheros que reciben el evento y ejecutan una serie de acciones que devuelve la información a la vista que rendiría al usuario.
- **helpers:** funciones usadas a nivel de vista de usuario.
- **mailers:** clases específicas para el envío de correos electrónicos.
- **models:** los modelos de la aplicación, donde tenemos toda la lógica de la clase y donde se conecta con la base de datos.



- **views:** las vistas que se mostrarán al usuario una vez que el controlador y modelo han trabajado juntos para obtener la información a mostrar.
- bin:** ejecutables de rails para ejecutar ciertas acciones, arrancar un servidor local, ejecución de scripts etc
- config:** ficheros de configuración. Los dos más importantes serán explicados en el siguiente punto.
- db:** ficheros relacionados con la base de datos, su cometido principal es la del uso de migraciones, en nuestro caso al trabajar con MongoDB no disponemos de migraciones.
- doc:** para almacenar documentación relacionada con el proyecto.
- lib:** librerías de terceros o nuestras propias librerías.
- log:** ficheros del log de accesos a la aplicación.
- public:** es la carpeta a la que se accede al arrancar un servidor en Rails. Cuando se realiza el despliegue a real todas las hojas de estilo, javascripts e imágenes deben estar dentro de dicha carpeta para que puedan ser visualizados por todos los usuarios.
- script:** comandos propios de rails
- test:** para pruebas de testing unitario u otros tipos de test que queramos realizar
- tmp:** ficheros temporales
- vendor:** plugins de terceros que no se instalen como una gema.
- Gemfile** y **Gemfile.lock** son los ficheros donde se indican las gemas que son dependencias de nuestra aplicación.

### 8.3.2. Ficheros de configuración

Dentro de la carpeta config se encuentran los ficheros de configuración de la aplicación Rails. Sólo vamos a detallar los ficheros más importantes que serán necesarios modificar en algún momento si necesitamos realizar ajustes en el código.

**mongoid.yml:** fichero de configuración de nuestra base de datos mongo. Se divide a su vez en 3 entornos que se corresponden con *development*, *test* y *production*. Esto se produce debido a que Rails se puede *arrancar en distintos entornos*, el habitual y en el que se arranca por defecto una aplicación es el entorno *development* o desarrollo pero llegado el caso puede ser necesario arrancar otro entorno y para ello será conveniente configurar la base de datos del entorno seleccionado.

**routes.rb:** contiene todas las rutas que reconoce una aplicación Rails. Si miramos su distribución veremos diferentes líneas donde se indican las distintas urls que reconocerá la aplicación.

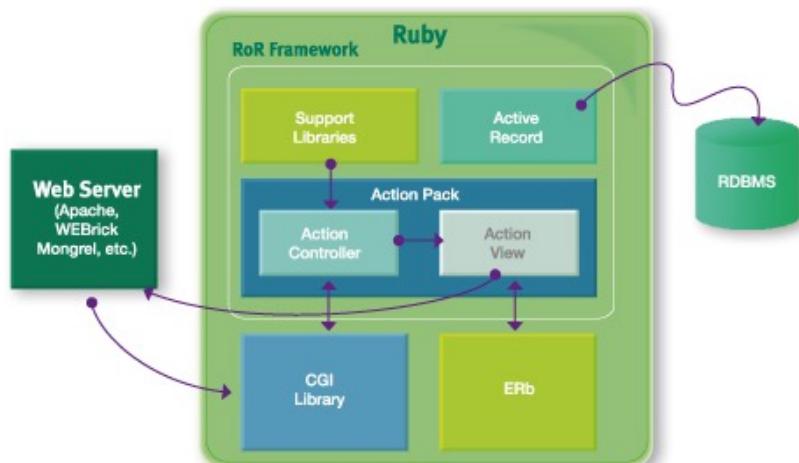
```
resources :comments, only: [:edit, :show] do
  member do
    post 'destroy'
    post 'mark_as_offensive'
    post 'update'
  end
end
```

Se está indicando que se cree una url para el recurso *comments*, el cual creará por defecto únicamente las urls para las acciones *edit* y *show*. Además crearemos urls de tipo POST para las acciones *destroy*, *mark\_as\_offensive* y *update* de tipo **member** con las que le estamos indicando que recibirán un parámetro obligatoriamente.

Dentro de la carpeta **config/locales** nos encontraremos diferentes **archivos en formato .yml** que se corresponden con los **ficheros de traducción de la aplicación**. El único aspecto a tener en cuenta con los ficheros .yml es que es muy importante escribirlos con una *correcta indentación del código incluido*.

### 8.3.3. Ciclo de vida de una petición

Cada aplicación de Rails en un servidor ejecuta su propio proceso. Este permanecerá ejecutándose hasta que el servidor web finalice su proceso o transcurra un tiempo de inactividad de la aplicación.



El ciclo de vida de una aplicación Rails se divide por tanto en una serie de pasos que siguen un orden establecido.

- **El usuario** que accede a la aplicación **realiza una petición al servidor web** usado, ya sea Nginx, Apache u otro cualquiera.
- El servidor web envía la petición a Ruby on Rails, en este punto será **Ruby on Rails el encargado de redirigirlo al controlador** correspondiente según lo que se haya indicado en el fichero de configuración *routes.rb*
- Rails encuentra el controlador solicitado y envía la petición de la acción solicitada por el usuario.
- El controlador y su acción específica comprueban si es necesario acceder al modelo u a otra librería indicada. En caso de tener que acceder al modelo éste se conectará a través de la librería *ActiveRecord* con la base de datos configurada en el fichero *mongoid.yml* o *database.yml* según el sistema elegido.
- Una vez ejecutadas todas las instrucciones indicadas en la acción del controlador **se envía toda la información capturada a las vistas** que a su vez trabajarán con las plantillas ERB para generar el HTML a visualizar.
- **Se devuelve la información en formato HTML** al usuario que había solicitado la ejecución inicial.

#### 8.3.4. Controladores en Ruby on Rails

Un controlador en Ruby on Rails es **una clase que hereda de ApplicationController**. Como indicamos en la estructura de carpetas cualquier controlador que creemos debe ir situado en la carpeta *app/controllers* y ya podremos hacer uso de él al iniciar la aplicación.

Se pone como ejemplo el controlador de comentarios de la aplicación y se explicarán conceptos básicos de la sintaxis empleada.

```
class CommentsController < ApplicationController
  # Acciones previas a borrado de comentario
  before_action :set_comment, only:
    [:destroy, :edit, :mark_as_offensive, :show, :update]
  before_action :owner?, only: [:destroy, :edit, :update]

  # Borrado de un comentario
  def destroy
    @comment.destroy
    respond_to do |format|
      format.html { redirect_to comments_path, notice: 'Comentario eliminado' }
      format.json { head :no_content }
    end
  end
```

```

        format.html { redirect_to :back }
        format.json { head :no_content }
      end
    end

# Edición de un comentario
def edit
  respond_to do |format|
    format.html
    format.js
    format.json { render :json => { :comment => @comment.as_json(json:
'wall') } }
  end
end

# Marcar un comentario como ofensivo
def mark_as_offensive
  @comment.mark_as_offensive
  respond_to do |format|
    format.html
    format.js
    format.json { render :json => { :comment => @comment.as_json(json:
'wall') } }
  end
end

# Visualización de un comentario
def show
  respond_to do |format|
    format.html
    format.js
    format.json { render :json => { :comment => @comment.as_json(json:
'wall') } }
  end
end

# Actualización de un comentario
def update
  if @comment.update_attributes(params[:comment])
    respond_to do |format|
      format.html
      format.json { render :json => { comment: @comment.as_json(json:
'wall') } }
    end
  else
    respond_to do |format|
      format.html { render action: :edit }
      format.json { render :json => { comment: @comment.errors,
status: :unprocessable_entity } }
    end
  end
end

protected

```

```

# Es el propietario del comentario?
def owner?
  redirect_to root_path if @comment.blank? || (@comment &&
@comment.user_id != @current_user._id)
end

# Setter de comentario para acciones indicadas
def set_comment
  @comment = Comment.find(params[:id])
end
end

```

Declaración del controlador CommentsController que como indicamos hereda de ApplicationController. Como se puede ver en este fichero y en los indicados en el código fuente todos los controladores deben terminar en plural.

```
class CommentsController < ApplicationController
```

La instrucción **before\_action** son funciones que se ejecutan antes de las acciones indicadas en el parámetro **only** o en todas las acciones que estén excluidas del parámetro **except**.

```
before_action :owner?, only: [:destroy, :edit, :update]
```

La declaración de un método o de una acción de un controlador se realiza con la palabra clave **def** y con el nombre de la acción o método

```
def owner?
```

Declaración de un método llamado **set\_comment**. Dicho método almacena en una variable llamada **@comment** la información que recupera del modelo **Comment** que le llega en el parámetro de la url [:id].

```

# Setter de comentario para acciones indicadas
def set_comment
  @comment = Comment.find(params[:id])
end

```

**Las variables que tienen @, como @comment, pueden ser usadas en las vistas,** si hacemos referencia en una vista a una variable sin @ nos dará un error de compilación al no existir dicha variable.

### 8.3.5. Modelos en Ruby on Rails

Como se ha indicado los modelos en Ruby on Rails al trabajar con una arquitectura MVC almacenarán toda la lógica de negocio. En este caso continuaremos con el ejemplo de comentario y expondremos el modelo y una explicación de aquellos aspectos más relevantes de explicar.

```
class Comment
  # Librerías de Mongo
  include Mongoid::Document
  include Mongoid::Timestamps

  # Constantes
  PER_PAGE = 25

  # Relaciones
  belongs_to :user

  # Campos
  field :participant_ids, type: Array, default: []
  field :object_id, type: String
  field :object_type, type: String
  # Mayor de 0 es un estado no válido completamente, inicialmente sólo
  # disponemos de 0 (válido) ó 1 (ofensivo)
  field :status, type: Integer, default: 0
  field :text, type: String

  # Callbacks
  before_save :set_participant_ids

  # Indices
  index({ :participant_ids => 1 }, { :name =>
  "participant_ids_index", :background => true })

  # Validaciones
  validates :user_id, presence: {message:
I18n.t("comments.user_id.presence")}
  validates :text, presence: {message: I18n.t("comments.text.presence")}
  validates :text, length: { maximum: 256, message:
I18n.t("comments.text.length")}

  # Methods
  # Search comments by params
  def self.search(_params = {})
    # Implementación del método
  end
end
```

```

    _comments = self.only(:_id, :created_at, :object_type, :participant_ids,
                          :status, :text, :user_id)
    if _params and !_params[:user].blank?
      _users = User.only(:_id, :email, :name).or({:name
=> /.*#{_params[:user]}.*/i}, {:email => /.*#{_params[:user]}.*/i}).collect(&:_id).collect(&:to_s)
      _comments = _comments.where(:participant_ids.in => _users)
    end

    _comments = _comments.where(:status => 1) if _params &&
    _params[:mark_as_offensive]
    _comments.desc(:created_at)
  end

  # Lista de comentarios en formato CSV
  def self.to_csv(_comments = nil, _columns = nil, options = {}, params = {})
    _columns ||= ["_id", "text"]
    _comments ||= Comment.desc(:created_at)

    CSV.generate(options) do |csv|
      csv << _columns
      _comments.each do |_comment|
        csv << _comment.attributes.values_at(*_columns)
      end
    end
  end

  # As_json override
  def as_json(options = {})
    case options[:json]
    when "wall"
      return super(:only => [:created_at, :text], :methods =>
      [:from, :id_to_s, :timestamp, :type_comment, :user_id_to_s])
    end
    super.merge(options)
  end

  # Usuario que ha enviado el comentario
  def from
    _from = User.only(:_id, :avatar_file_name, :avatar_updated_at, :name)
            .find(self.user_id) rescue nil
    if _from
      { :image => _from.avatar.url(:detail), :name => _from.name}
    end
  end

  # Elemento identificativo de la base de datos (JSON)
  def id_to_s
    self._id.to_s
  end

  # Comprueba si un comentario es de tipo comentario
  def is_comment?
    self.object_type.downcase.to_s == "comment" || self.object_type.blank?
  end

```

```

# Comprueba si un comentario es de tipo comentario
def is_event?
  self.object_type.downcase.to_s == "event"
end

# Comprueba si el comentario es ofensivo
def is_offensive?
  self.status == 1
end

# Comprueba si un comentario es de tipo comentario
def is_route?
  self.object_type.downcase.to_s == "route"
end

# Marcar comentario como ofensivo
def mark_as_offensive
  self.update_attribute(:status, 1)
end

# Hora de creación del comentario (formato UNIX)
def timestamp
  self.created_at.to_i
end

# Método para indicar el tipo de comentario
def type_comment
  self.object_type.blank? ? "comment" : self.object_type
end

# Identificador del usuario
def user_id_to_s
  self.user_id.to_s
end

# Comentarios de un objeto en concreto
def self.get_object_comments(_object, _timestamp)
  comments = Comment.where(:object_type => _object.class.to_s, :object_id => _object.id_to_s)
  comments = comments.where(:created_at.gt => "#{Time.at(_timestamp.to_i)}") if _timestamp
  comments = comments.desc(:created_at).limit(25)
end

# Comentarios del muro de un usuario
def self.user_wall(_user, _timestamp = nil)
  comments = Comment.where(:object_type => nil).any_of([{:participant_ids.in => [_user._id.to_s]}, {:participant_ids.in => _user.follow_ids}])
  comments = comments.where(:created_at.lt => "#{Time.at(_timestamp.to_i)}") if _timestamp
  comments = comments.desc(:created_at).limit(10)
end

# Comentarios realizados por el usuario

```

```

def self.user_tweets(_user, _timestamp = nil)
  comments = Comment.where(:object_type => nil, :user_id => _user._id.to_s)
  comments = comments.where(:created_at.lt =>
  "#{Time.at(_timestamp.to_i)}") if _timestamp
  comments = comments.desc(:created_at).limit(10)
end

protected

# Setter de los participantes del comentario
def set_participant_ids
  self.participant_ids = []
  self.participant_ids << user_id.to_s
end
end

```

Como se puede observar el fichero se declara en singular. Todos los modelos en Rails a menos que se cambie mediante ficheros de configuración serán declarados en singular.

```
class Comment
```

Al trabajar con una base de datos de tipo no relacional, en este caso MongoDB, los campos de la misma se declaran a nivel de modelo. Para ello previamente es necesario indicarle al modelo que vamos a trabajar con las librerías de MongoDB y a continuación indicaremos los campos y el tipo de dato que almacenará.

```

# Librerías de Mongo
include Mongoid::Document
include Mongoid::Timestamps

# Campos
field :participant_ids, type: Array, default: []
field :object_id, type: String
field :object_type, type: String
# Mayor de 0 es un estado no válido completamente, inicialmente sólo
# disponemos de 0 (válido) ó 1 (ofensivo)
field :status, type: Integer, default: 0
field :text, type: String

```

Los campos se declaran mediante la instrucción **field**, nombre del campo, tipo de dato a almacenar y cualquier otro parámetro válido como valores por defecto o similar.

También a nivel de modelo se pueden declarar **relaciones** con otros modelos. Una relación indica que un modelo tiene muchos (*has\_many*) de otros modelos, tiene uno (*has\_one*) de otro modelo o pertenece (*belongs\_to*) a otro modelo, esto nos facilita la labor ya que nos crea métodos asociados y nos devolverá los elementos relacionados.

En nuestro caso un comentario pertenece a un usuario por lo que nuestro modelo pertenece al modelo user.

```
# Relaciones  
belongs_to :user
```

Los **callbacks** son ejecuciones que se realizan previo o posterior a ciertas acciones del sistema que pueden ser la creación, actualización, validación o borrado de un elemento.

```
before_save :set_participant_ids
```

El siguiente aspecto a revisar será el de las **validaciones**. A nivel de modelo es el punto del back-end donde se realizarán las comprobaciones de los datos previo al guardado de un elemento.

Consta de la instrucción **validates**, nombre del campo y la validación o validaciones a ejecutar para dicho campo.

```
validates :user_id, presence: {message: I18n.t("comments.user_id.presence")}
```

En este caso se está indicando que el campo user\_id es obligatorio y en caso de que no sea así se indicará por pantalla el mensaje indicado en el fichero de traducciones.

### 8.3.6. Vistas en Ruby on Rails

Las vistas se corresponde con el código HTML que se devolverá al usuario que ha realizado una petición al servidor web.

Las vistas en Ruby on Rails son la **unión de código HTML más las variables que se han declarado a nivel global en la acción del controlador** que se ha ejecutado.

Se pone como ejemplo un listado de comentarios de la parte de administración del sistema:

```
<h4 class="page-header">Comentarios</h4>
<%= render "admin/comments/search" %>
<div class="panel panel-default">
  <div class="panel-heading">Gestión de comentarios</div>
  <div class="panel-body">
    <table class="table table-condensed">
      <tr>
        <th><%= Comment.human_attribute_name(:text) %></th>
        <th><%= Comment.human_attribute_name(:user) %></th>
        <th><%= Comment.human_attribute_name(:created_at) %></th>
        <th class="options"><%= t("main.options") %></th>
      </tr>
      <% @comments.each do |comment| %>
        <tr id="comment_<%= comment._id %>" class="<%= "alert alert-danger" if comment.is_offensive? %>">
          <%= render partial: "admin/comments/comment", locals: {comment: comment} %>
        </tr>
      <% end %>
    </table>
    <%= paginate @comments %>
  </div>
</div>
```

Como se puede ver en este caso sólo se está mostrando una porción de código del HTML final que verá el usuario. El resto saldrá del fichero **app/views/layouts/application.html**. Dicho fichero contiene la estructura básica o esqueleto de la página web, es decir, las cabeceras, peticiones CSS, Javascript y el cuerpo. Tal y como se ha indicado Rails busca sobre todo no repetir código y la mejor opción es usar una plantilla en la que integrar el resultado de la vista.

### 8.3.7. Estructura del código fuente

En este punto se le indicará al usuario programador la **estructura del código fuente adjunto en el CD-ROM**. Para un sencillo recorrido de los distintos ficheros que componen la aplicación comenzaremos detallando los diferentes archivos que disponemos en la configuración del proyecto para a continuación centrarnos en toda la lógica de la aplicación.

### 8.3.7.1. Ficheros de Configuración

Situados en la carpeta config del proyecto disponemos de una serie de ficheros que controlarán la configuración básica de una aplicación en Ruby on Rails, ficheros de traducción, variables que se ejecutan al arrancar el sistema, inicializadores, SMTP de envío de correos electrónicos, etc

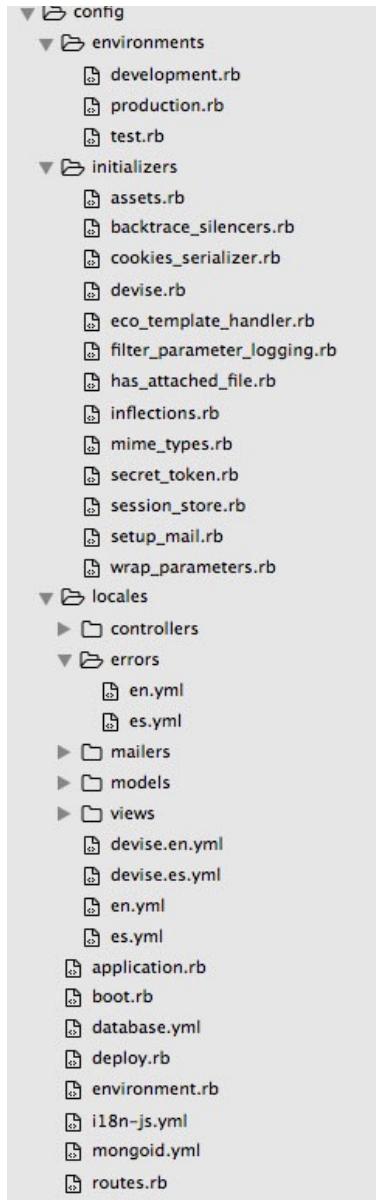
**application.rb:** fichero en el cual arranca la aplicación y configuración de un proyecto en Ruby on Rails. En él tenemos indicadas unas series de dependencias así como la dirección en el sistema de los ficheros de idiomas así como el idioma elegido por defecto.

**database.yml:** en caso de hacer uso de una base de datos de tipo relacional se declarará el conector de la base de datos a utilizar así como su usuario y contraseña.

**deploy.rb:** fichero utilizado para la realización de los despliegues automatizados.

**mongoid.yml:** fichero de conexión de la base de datos para proyectos que hacen uso de MongoDB.

**routes.rb:** fichero con todas las rutas que se generan en nuestra aplicación en Ruby on Rails. En caso de necesitar nuevas acciones o controladores se deberían añadir en dicho fichero para que el sistema las reconozca.



Dentro de la carpeta **environments** disponemos de 3 ficheros que tendrán distintos valores según se arranquen en un entorno *development*, *test* o *production*. Sobre todo será referido a asuntos como la caché, compilación de hojas de estilo y javascript o similares.

Dentro de la carpeta **initializers** disponemos de todos los ficheros de inicialización del proyecto. Los dos únicos ficheros a los cuales se les está dando utilidad en este momento son:

**devise.rb:** fichero de configuración del registro y login de usuarios, se puede configurar el tiempo máximo para el recordatorio de contraseñas, si vamos a almacenar la ip, email remitente de los correos electrónicos, etc

**setup\_mail.rb:** fichero de configuración donde almacenamos el SMTP a usar en la aplicación.

Por último dentro de la carpeta **locales** se encuentran todos los ficheros usados para las traducciones. A su vez **esta carpeta se divide en muchas subcarpetas donde se almacenan los mensajes que se han traducido** asociados a los controladores, modelos, vistas por cada clase de las que disponemos en el sistema.

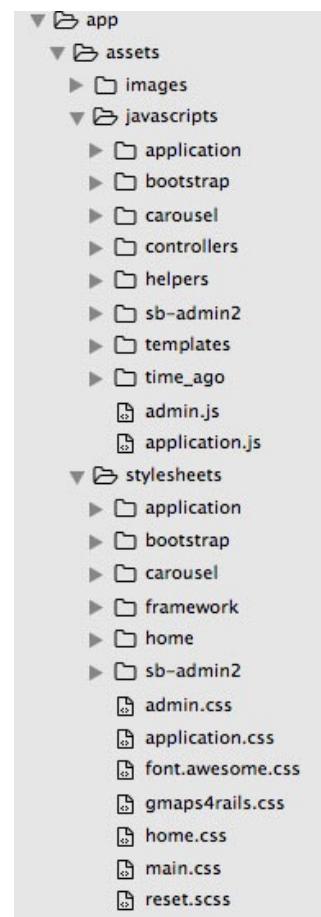
Los ficheros *tienen un formato YAML* que se basa en la **indentación** del texto introducido para reconocer a qué clave pertenece el valor introducido.

#### 8.3.7.2. Hojas de estilo, imágenes y ficheros javascript

En la carpeta **app/assets** disponemos de todos los ficheros de **hojas de estilo, imágenes y javascripts de la aplicación**. Este directorio aunque pueda sorprender al haber explicado anteriormente que un proyecto en Ruby on Rails funciona desde la carpeta public se debe a que durante los despliegues a producción se realizan compilaciones de todos los ficheros a la carpeta **public/assets** y se puede acceder por tanto a dichos ficheros desde el exterior.

Igual que en el punto anterior pasaremos a detallar la diferente estructura de ficheros que disponemos en este punto para facilitar la labor del programador.

La carpeta **images** contiene todas las imágenes distribuidas en el proyecto, no se entrará en detalle en dicha carpeta ya que el acceso a las mismas se realizará indicando la ruta completa en la que estén situadas.



La carpeta **javascripts** contiene los ficheros javascripts a usar en la aplicación. Están escritos bajo lenguaje CoffeeScript, en ella se ven dos ficheros principales:

**admin.js:** fichero de javascript que hace referencia a otros ficheros javascript situados en las carpetas que cuelgan de la carpeta inicial javascripts con instrucciones usadas únicamente para la parte privada de la aplicación.

**application.js:** fichero de javascript que hace referencia a otros ficheros y que se usará en la parte pública de la aplicación.

A continuación se hará un breve resumen de lo que contiene cada carpeta:

**application:** contiene ficheros javascripts que son comunes a la aplicación tanto en la parte pública como la parte privada.

**bootstrap:** ficheros javascripts del framework CSS Bootstrap.

**carousel:** fichero javascript de carruseles (owl-carousel)

**controllers:** ficheros javascript que contienen la lógica de las acciones ejecutadas a nivel javascript durante la ejecución de la SPA (Single Page Application), contienen acciones que solicitan información JSON y que a continuación se visualizarán en pantalla.

**helpers:** javascripts que contienen funciones exclusivas como el contado de caracteres.

**sb-admin2:** ficheros javascript incluidos con el tema SB-ADMIN2 utilizado para la parte privada de administración.

**templates:** contienen las plantillas ECO de las cuales hacen uso los controladores que hemos comentado previamente.

Por último la carpeta **stylesheets** contiene todos los ficheros de hojas de estilo utilizadas en la aplicación, las hojas de estilo principales siguen el patrón SASS. Igual que en el caso de los ficheros javascript existe un fichero **application.css** que hace referencia a una serie de ficheros stylesheet a cargar.

Indicaremos los diferentes ficheros que contienen las carpetas que cuelgan de la carpeta stylesheets:

**application:** contiene ficheros de hojas de estilo divididas por los elementos existentes en la aplicación como pueden ser *event.scss*, *route.scss*, etc.

**bootstrap:** hojas de estilo que contienen el framework utilizado por BootStrap.

**carousel:** hoja de estilo asociada a un plugin jQuery de carruseles (owl-carousel)

**framework:** hoja de estilos con variables para colores, funciones a utilizar, etc.

**home:** hoja de estilos específica para la página de inicio.

**sb-admin2:** hoja de estilos específica para la parte privada de la aplicación.

### 8.3.7.3. Controladores de la aplicación

La siguiente carpeta a explicar será la asociada a los controladores de la aplicación situada en **app/controllers**.

Tal y como se ha indicado anteriormente Ruby on Rails trabaja con una *arquitectura MVC* en la cual el controlador recibe la petición realizada por un usuario y selecciona la acción indicada y devuelve la información a visualizar en pantalla.

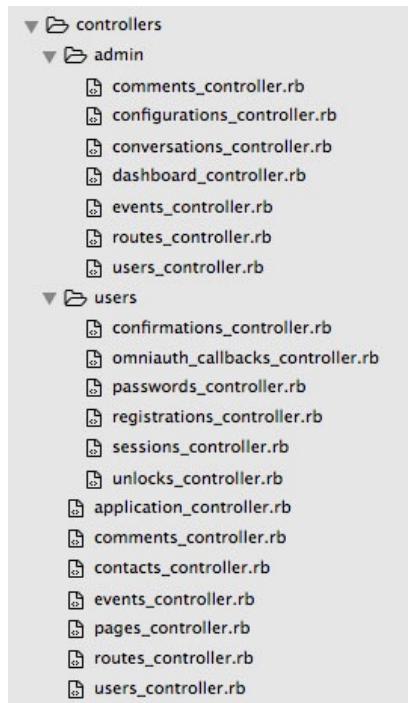
Por tanto cualquier nuevo controlador que queramos crear debería ir dentro de esta carpeta.

A su vez podemos ver que se divide en dos subcarpetas.

La carpeta **admin** contiene los *controladores y las acciones correspondientes a la parte de administración*.

La carpeta **users** contiene todos los controladores *asociados al registro, recordatorio de contraseña, login de usuarios*.

El resto de ficheros se corresponden a los *controladores de la parte pública*.



**application\_controller.rb:** contiene todas las acciones que se utilizarán en todos los controladores que heredan de él.

**comments\_controller.rb:** contiene las acciones asociadas con los comentarios de la aplicación como son la edición, borrado o marcar como ofensivo.

**contacts\_controller.rb:** contiene acciones asociadas al formulario de contacto de la aplicación como puede ser la visualización y el envío del correo electrónico a la administración de la aplicación.

**events\_controller.rb:** controlador de eventos que contiene todas las acciones asociadas a los mismos como puede ser el listado, creación, edición y borrado de los mismos.

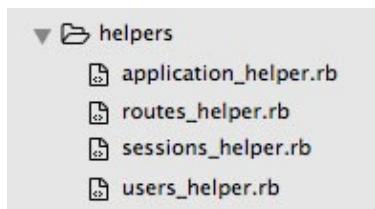
**pages\_controller.rb:** controlador para páginas, en principio sólo hay una acción que se corresponde con la página inicial.

**routes\_controller.rb:** controlador de rutas que contiene todas las acciones asociadas a la misma.

**users\_controller.rb:** controlador de usuarios que contiene acciones como la inserción de favoritos, comentarios a mostrar para el usuario elegido, amistades, etc.

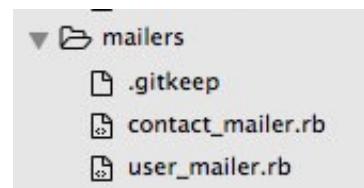
#### 8.3.7.4. Helpers

Los *helpers* son funciones que pueden ser utilizadas exclusivamente en las vistas. En este caso concreto los ficheros que existen contienen funciones específicas para sesiones, usuarios o rutas o en **application\_helper.rb** existen funciones a nivel de proyecto.



#### 8.3.7.5. Mailers

Los ficheros *mailer* se corresponden con controladores que se utilizan para el envío de correos electrónicos. En este caso se disponen de dos ficheros:



**contact\_mailer.rb:** se corresponde con el envío de correos electrónicos asociados al formulario de contacto.

**user\_mailer.rb:** corresponde con el envío de correos electrónicos que se realizan a los usuarios de la aplicación.

#### 8.3.7.6. Models

Los ficheros de modelos almacenan toda la lógica de negocio de la aplicación debido a la arquitectura MVC de la que hace uso Ruby on Rails.

Es una única carpeta que contiene todos los ficheros de modelos de los que disponemos. Todos ellos excepto *contact.rb* e *import.rb* hacen uso de la conectividad con la base de datos y contienen la estructura de campos con la que trabajan, validaciones, callbacks y métodos.

**comment.rb:** modelo asociado a la gestión de comentarios. Relacionado con los usuarios y de manera polimórfica con rutas y eventos y de esta manera con cualquier otro elemento que queremos asociar.

**config.rb:** modelo que almacena la clave y valor de configuración, inicialmente es usada para que el administrador pueda modificar el receptor del correo electrónico enviado por un usuario al hacer uso del formulario de contacto.

**contact.rb:** modelo asociado al formulario de contacto

**conversation.rb:** modelo asociado a las conversaciones, relacionada con usuarios y mensajes.

**event.rb:** modelo asociado a los eventos de la aplicación, relacionado con usuarios y comentarios.

**favorite.rb:** modelo que almacena toda la información de los favoritos almacenados por un usuario.

**import.rb:** modelo que contiene las instrucciones necesarias para realizar la importación de una ruta a partir de una URL dada.

**main.rb:** modelo que contiene métodos de carga básicos para los tipos de eventos y rutas disponibles.

**message.rb:** contiene los mensajes almacenados en una conversación.

**route.rb:** contiene la información y métodos de las rutas almacenadas en la aplicación.

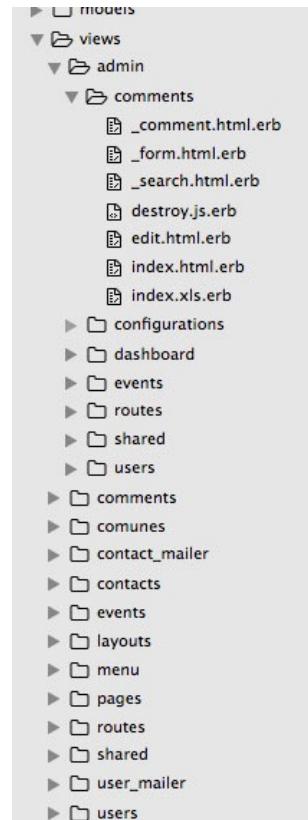
**user.rb:** modelo con todas las instrucciones de acceso de usuarios, encriptación de contraseñas, etc.

### 8.3.7.7. Vistas

Por último se explicará la estructura de las vistas de las que disponemos en la aplicación. Al igual que en el caso de los controladores disponemos de una carpeta **admin** que contiene todas *las vistas asociadas a los controladores de la parte de administración* y también *el resto de carpetas asociados a los controladores de la parte pública*.

*Por norma cada acción de un controlador dispone de una vista asociada* aunque esto no siempre es obligatorio ya que no todas las acciones tienen porque visualizar información en pantalla.

Los ficheros que se denominan `_form.html.erb` (**guión bajo junto a un nombre**) se denominan **partials**. Son ficheros que pueden ser cargados en otra vista y a la que se le puede enviar parámetros a cargar en pantalla, de esta manera si necesitamos el mismo código en varias vistas podemos generar un partial de manera que no repitamos código.





## 9. Anexo

---

### 9.1. Conclusión del Proyecto

Tras la finalización del proyecto y tras muchas horas de trabajo invertidas en el mismo diseñando, estableciendo requisitos y por último con la codificación de la aplicación se puede decir que ha sido una experiencia enriquecedora en la que me ha permitido mejorar mis conocimientos en distintos aspectos que no había ejecutado con anterioridad.

Sobre todo me gustaría agradecer la ayuda recibida en todo momento por parte de mi director de proyecto Fernando Álvarez García por su apoyo, por la libertad que me ha ofrecido tanto en las tecnologías usadas como en la ejecución desarrollada así como por las ideas que me fue aportando durante el proceso, así como a todas aquellas personas que me dieron su feedback de la aplicación y que aspectos se podrían incluir o mejorar de aplicaciones ya existentes.

## 9.2. Futuras ampliaciones

Como red social existen una infinidad de posibles opciones de mejora siendo algunas de las planteadas las siguientes:

- **Integración con otras redes sociales** permitiendo compartir información de nuestra red social en Facebook, Twitter, Pinterest u otras.
- Posibilidad de **creación de álbumes, por usuarios, grupos, eventos y rutas** permitiendo la subida de fotos a partir de los permisos integrados en el desarrollo del portal.
- Posibilidad de **insertar vídeos de YouTube, Vine, Dailymotion u otras páginas web similares** en los propios comentarios de manera que un usuario pueda ver de un vistazo partes de una ruta.
- **Mejora de permisos** dando más posibilidad a los usuarios sobre los mensajes que aparecen en el muro (posibilidad de que lo vean sólo sus amigos, personas de segundo nivel, amigos de amigos, ciertos mensajes privados)
- **Mejora en las rutas** de manera que sea posible realizar una búsqueda sobre la misma para ver como llegar al inicio desde nuestra posición actual.
- Valorar tras el uso y el coste añadido que supone el uso de GoogleMaps la **migración a un conjunto de mapas de libre acceso** como OpenStreetMap.
- **Invitación a tus amigos** a partir de tu correo electrónico (obtener amistades a partir de tu correo electrónico de Gmail, Yahoo, Hotmail u otras a partir de las APIs proporcionadas por cada desarrollador) para facilitar la integración de nuevos usuarios en el portal.
- Posibilidad de añadir **publicidad** ya sea por parte de los usuarios o de compañías de AdServer de manera que se haga un proyecto sostenible económicamente.
- Y muchas más cosas ...

### 9.3. Bibliografía

Libros, libros online y referencias consultadas en internet

A lo largo del desarrollo del proyecto se han consultado numerosos libros, libros online y páginas web útiles durante el desarrollo del mismo. Se detallan a continuación algunas de las más relevantes:

- [1] RUBY, Sam. THOMAS, Dave. HEINEMIER HANSSON, David. *Agile Web Development with Rails (4th edition)*, The Pragmatic Programmers, 2013. 456 p. ISBN: 978-1-93778-556-7
- [2] FERNANDEZ, Obbie. FAUSTINO, Kevin. KUSHNER, Vitaly. *The Rails 4 Way*. Addison Wesley 2014. 880 p. ISBN: 978-0321944276
- [3] SUMMER, Aaron. *Everyday Rails Testing with RSpec*. Disponible en Web: <https://leanpub.com/everydayrailsrspec>
- [4] JIMENEZ VILLAR, Javier. *CoffeeScript: un pequeño gran libro*. Disponible en Web: <https://leanpub.com/coffeescript>
- [5] API de Ruby <http://www.ruby-doc.org/core/>
- [6] API de Ruby on Rails <http://api.rubyonrails.org/>
- [7] AsciiCasts <http://es.asciicasts.com>
- [8] Twitter-Bootstrap <http://www.getbootstrap.com>
- [9] Web oficial de MongoDB <http://www.mongodb.org>
- [10] <https://www.ruby-lang.org>
- [11] <http://rubyonrails.org/>
- [12] <http://www.mongodb.org/>
- [13] <http://nginx.org/>
- [14] <https://rvm.io/>
- [15] <https://www.phusionpassenger.com/>
- [16] <https://github.com/mina-deploy/mina>

