Universidad Zaragoza

Trabajo Fin de Grado
Grado en Ingeniería Informática

Aplicación móvil para la recopilación y seguimiento de esfuerzos en proyectos en equipo con técnicas de gamificación y serious games para fomentar su uso

Mobile application for effort gathering and tracking in team projects with gamification and serious games techniques to encourage its use

Autor

Patricia Lázaro Tello

Director

Rubén Béjar Hernández

Universidad de Zaragoza
Escuela de Ingeniería y Arquitectura

2016

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

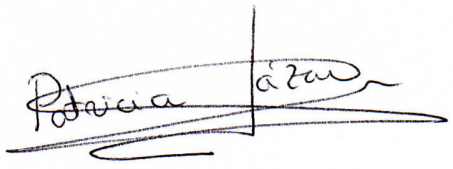
D./Dª. Patricia Lázaro Tello,

con nº de DNI 73025066C en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Grado, (Título del Trabajo) Aplicación móvil para la recopilación y seguimiento de esfuerzos en proyectos en equipo con técnicas de gamificación y serious games para fomentar su uso,

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 19 de septiembre de 2016



Fdo: Patricia Lázaro Tello

Aplicación móvil para la recopilación y seguimiento de esfuerzos en proyectos en equipo con técnicas de gamificación y serious games para fomentar su uso

Resumen

Este proyecto tiene como objetivo la creación de una aplicación para la recopilación y seguimiento de esfuerzos que incite y motive al usuario a utilizarla y rellenar de forma diaria o regular sus horas de esfuerzos en los distintos proyectos en los que participa.

Para motivar al usuario a continuar rellenando sus esfuerzos en la aplicación, se han utilizado técnicas de gamificación como elementos que añaden una componente competitiva a la aplicación y ayudan a los usuarios a permanecer en la aplicación y recopilando sus esfuerzos. Los elementos de gamificación utilizados en este proyecto han sido: un sistema de logros, que recompensa al usuario de forma excepcional, y un sistema de reputación, que recompensa al usuario en sus tareas del día a día.

El proyecto se ha dividido en tres componentes: aplicación cliente, aplicación servidor y base de datos; la arquitectura del sistema es, por tanto, una arquitectura cliente-servidor. La aplicación cliente se ha programado utilizando el framework Ionic, AngularJS, CSS3, HTML5 y JavaScript (tecnologías web). Se ha desplegado en dos plataformas, como aplicación Android creando un APK, y como aplicación web desplegándola en un servidor Express. La aplicación servidor se ha programado en Java íntegramente, siendo desplegada en Tomcat 7, en el PaaS Openshift. La base de datos, desplegada también en Openshift, utiliza la tecnología SQLite.

El desarrollo de este proyecto se ha realizado a lo largo de dos iteraciones o sprints, con una fase inicial de definición de requisitos y estudio de tecnologías y una fase final de compleción de la documentación técnica y redacción de la memoria del proyecto.

El desarrollo de aplicación cliente, aplicación servidor y base de datos ha sido paralelo. El proyecto se ha desarrollado por funcionalidades y no por componentes para poder tener unos primeros prototipos que mostrar y probar.

Pese al trabajo realizado en este proyecto, la aplicación aquí descrita tiene características que se podrían mejorar, como crear una interfaz exclusiva para la aplicación web o agregar más funcionalidades de gamificación (notificaciones personalizables, sistema de niveles...).

Índice

Índice de figuras

Índice de tablas

1. Introducción

Este documento describe el proceso de realización del Trabajo de Fin de Grado titulado *'Aplicación móvil para la recopilación y seguimiento de esfuerzos en proyectos en equipo con técnicas de gamificación y serious games para fomentar su uso'*.

1.1. Contexto

Este proyecto ha sido realizado entre los meses de junio y septiembre de 2016. Todo el trabajo mostrado a lo largo del documento ha sido elaborado en su totalidad por Patricia Lázaro Tello, contando con la ayuda y supervisión de su tutor Rubén Béjar Hernández.

El Trabajo de Fin de Grado está planteado desde su inicio como una aplicación móvil, aunque no se cerró nunca la puerta de desplegar la aplicación en web.

Para el desarrollo de la aplicación cliente, que se desplegó en la plataforma Android y como aplicación web, se utilizó una combinación del framework Ionic [20], AngularJS [21], JavaScript, HTML5 y CSS3. El framework Ionic permite crear aplicaciones para diversas plataformas, entre ellas Android e iOS, utilizando tecnologías web, de forma que, con unos arreglos, se puede además desplegar la aplicación en la web.

Para el desarrollo de la aplicación servidor, desplegada en Openshift [12], se utilizó el lenguaje Java [13], un lenguaje muy conocido y utilizado en el transcurso del grado de Ingeniería Informática. Para la base de datos, alojada igualmente en Openshhift, se utilizó SQLite [15], una tecnología de bases de datos relacionales muy ligera.

1.2. Objetivo

El objetivo de este Trabajo de Fin de Grado es crear una aplicación para la recopilación y seguimiento de esfuerzos que incite y motive al usuario a utilizarla y a rellenar de forma diaria sus horas de esfuerzos.

1.3. Motivación

El principal interés para realizar este Trabajo de Fin de Grado ha sido crear una herramienta de uso personal para el rellenado de esfuerzos, que además funcione en dispositivos móviles.

Esta motivación surge de las experiencias del grado, donde en diferentes asignaturas se pide a los alumnos que lleven un control de las horas de esfuerzo invertidas en sus proyectos (Proyecto Software, Videojuegos, Gestión de Proyecto Software, etcétera).

No obstante, a menudo los estudiantes se olvidan de rellenar estas horas de esfuerzo algunos días o, si no son exigidas, al cabo del tiempo pierden la determinación de continuar rellenándolas.

Con esta aplicación, dirigida más a estudiantes que a una organización o compañía, se pretende motivar a los usuarios para que continúen rellenando sus esfuerzos en los diversos proyectos que acometan, ya sean en solitario, en equipo, de índole personal o laboral.

1.4. Estructura del documento

Este documento consta de seis apartados y los anexos. A continuación se describe de forma general cada una de estas secciones.

La sección 1, **Introducción**, explica cuestiones generales del Trabajo de Fin de Grado, como el contexto, motivación u objetivo del proyecto. La sección 2 describe el problema y los requisitos del proyecto, y realiza un análisis de aplicaciones alternativas.

El apartado 3 describe la arquitectura del sistema, incluyendo un mapa de navegación de la interfaz de la aplicación, y el modelo de datos empleado, además de describir las técnicas de gamificación utilizadas en el proyecto

En la sección 4 se detallan cuestiones como la tecnología empleada, pruebas realizadas, problemas encontrados y sus soluciones, despliegue de los diferentes componentes del proyecto y resultados obtenidos. El contenido del apartado se estructura por componentes: se describen los aspectos anteriormente mencionados para cada componente de la aplicación (cliente, servidor y base de datos).

En el apartado 5 se describe la planificación del proyecto, el control de esfuerzos que se ha llevado y la gestión de configuraciones utilizada. Por último, en la sección 6 se cierra el documento con una valoración personal y posibles líneas de trabajo futuras.

2. Análisis del problema

En esta sección se describe el problema que aborda el Trabajo de Fin de Grado, los requisitos de la aplicación, divididos en requisitos funcionales y no funcionales, y un estudio de mercado para comparar aplicaciones similares a la realizada en este trabajo.

2.1. Descripción del problema

A día de hoy, muchas personas buscan controlar las horas de su tiempo que invierten en distintas tareas, ya sea en el ámbito laboral, el más común, o en el ámbito personal, para planificar mejor el tiempo libre.

Pero el control de horas de esfuerzo no se da solo en el trabajo y en la vida personal, también es una actividad que se les exige a los estudiantes para controlar el tiempo que les cuesta realizar cada trabajo, ya sea individual o en equipo.

Sin embargo, el control de las horas de esfuerzo puede ser una tarea pesada y fácilmente olvidada, a no ser que se haya desarrollado ya un hábito (e incluso entonces, es fácil olvidarse algún día). Este es el problema que se busca resolver en este Trabajo de Fin de Grado aplicando gamificación.

La **gamificación** consiste en el uso de técnicas y conceptos propios de juegos para actividades no recreativas con el fin de potenciar o motivar las actividades y reforzar el aprendizaje de los conceptos que hay detrás.

En el contexto de este Trabajo de Fin de Grado, se utiliza la gamificación para motivar a los usuarios a controlar sus horas de esfuerzo regularmente y crear un hábito a la hora de su rellenado, concretamente se utiliza un **sistema de progresión** y **logros**, del que se hablará más adelante.

2.2. Análisis de alternativas

A continuación se analizarán diversas aplicaciones similares a la que se va a realizar, poniendo de relieve sus puntos positivos y negativos.

- **Toggl** [28]: es una aplicación de control del tiempo orientada principalmente al mundo empresarial, pero que se puede utilizar sin problemas para controlar tareas personales.

  Tiene versión de web, aplicación de escritorio para Windows, Linux y Mac, y aplicación móvil para Android, iOS y Windows Phone. También ofrece integración con otras herramientas de trabajo en grupo/equipo, como Basecamp, Github o Trello.

  Tiene un plan gratis, que permite crear equipos de hasta cinco miembros, ilimitados proyectos y clientes, y crear informes en formato CSV. Los planes Pro y Business añaden otras funcionalidades, como entradas facturables o agregar horas de esfuerzo a otros miembros del equipo.

- **Puntos positivos**: se puede acceder desde cualquier dispositivo, ya sea móvil, ordenador o web, a través de las diferentes aplicaciones. El plan gratis contiene muchas funcionalidades.
- **Puntos negativos**: al estar dirigida al mundo empresarial, tiene planes de pago, detrás de los cuales hay funcionalidades muy interesantes, como agregar horas de esfuerzo a otros miembros del equipo o no poder hacer equipos grandes.

- **TrackingTime** [29]: es una aplicación de control del tiempo orientada a las empresas, más difícil de usar para tareas personales.

  Tiene versión web, aplicación de escritorio para Windows, Linux y Mac, y aplicación para Android y iOS. Como Toogl, también ofrece integración con otras herramientas de trabajo en grupo/equipo, como Basecamp, Github, Bitbucket o Trello.

  Tiene un plan gratis, que permite crear equipo de hasta tres miembros, ilimitados proyectos y tareas, y crear informes en formato CSV. El plan Pro añaden funcionalidades como un calendarios de tareas, importar las horas de esfuerzo desde CSV, crear tareas en masa o establecer permisos avanzados para los usuarios.

  - **Puntos positivos**: se puede acceder desde cualquier dispositivo, ya sea móvil, ordenador o web, a través de las diferentes aplicaciones. El plan gratis contiene muchas funcionalidades. Aunque tiene un plan de pago, al identificarse por primera vez se activa una prueba de 30 días del modo PRO para ayudar al usuario a decidir si desea el plan de pago o no.
  - **Puntos negativos**: al estar dirigida al mundo empresarial, tiene un plan de pago que contiene características interesantes, como ampliar equipos o establecer permisos avanzados. El tamaño de los equipos en el plan gratis es muy pequeño; se puede utilizar como una versión de prueba o para tareas personales.

- **MyMinutes** [30]: es una aplicación de control del tiempo para dispositivos iOS. Está orientada al uso personal, siendo sencilla y fácil de usar, sin términos del mundo empresarial.

  La aplicación utiliza eventos de *'como mucho X tiempo'* y *'como poco Y tiempo'* para controlar el tiempo gastado en cada actividad y construir rachas que motiven al usuario.

  Las horas invertidas en cada tarea solo pueden introducirse mediante un temporizador en la aplicación. Las tareas son repetibles: el usuario señala los días de la semana en los que tiene que realizar la tarea y esos días recibe una notificación. No se puede añadir usuarios a las tareas.

  - **Puntos positivos**: la idea de rachas y eventos que utiliza la aplicación ayuda a fidelizar a los usuarios y mantenerlos motivados para que continúen utilizando la aplicación y controlando su tiempo, sin olvidarse de hacer el seguimiento de esfuerzos.
  - **Puntos negativos**: la aplicación está orientada a actividades personales, no permitiendo crear ningún tipo de asociación con otros usuarios ni introducir las horas de esfuerzo de forma manual.

Las aplicaciones existentes en el mercado a día de hoy están dirigidas al mundo empresarial, con planes de pago y equipos pequeños en los planes gratis, o son de uso personal, con grandes esfuerzos puestos en la motivación del usuario pero enfocadas a tareas personales, sin permitir la interacción con otros usuarios, y sin guardar un seguimientos del tiempo empleado en la tarea a lo largo del tiempo.

Los dos tipos de aplicación tienen características que se desean: por un lado, los proyectos, tareas y equipos de las aplicaciones de control de esfuerzos en el mundo empresarial, y por otro, la ayuda a la motivación del usuario de las aplicaciones de uso personal.

2.3. Requisitos

A continuación se muestran en formato tabla los requisitos funcionales y no funcionales de la aplicación.

2.3.1. Requisitos Funcionales

Cuadro 1: Tabla de Requisitos Funcionales

Código	***Descripción del requisito***
RF1	El usuario podrá identificarse con la cuenta de Google.
RF2	Los proyectos constarán de diferentes actividades, incluyendo una actividad por defecto, a las que los usuarios asignarán sus esfuerzos.
RF3	La aplicación constará de dos roles para cada proyecto: el *administrador*, que corresponde al usuario que crea el proyecto, y los *miembros*, que son todos aquellos usuarios que se unen al proyecto.
RF4	Los esfuerzos constan de horas invertidas, actividad y proyecto en el que se han invertido dichas horas y día de los esfuerzos.
RF5	El usuario podrá introducir los esfuerzos realizados en un proyecto y guardarlos.
RF6	El usuario podrá modificar los esfuerzos realizados en un proyecto.
RF7	El usuario podrá borrar los esfuerzos realizados en un proyecto.
RF8	El usuario podrá visualizar los esfuerzos que ha realizado en un proyecto. La aplicación permitirá distintas formas de visualizar los esfuerzos del usuario dentro del proyecto.
RF9	El usuario podrá crear proyectos nuevos.
RF10	El usuario podrá borrar proyectos.

RF11	El usuario (con rol administrador) podrá crear actividades en el proyecto.
RF12	El usuario (con rol administrador) podrá modificar actividades en el proyecto.
RF13	El usuario (con rol administrador) podrá eliminar actividades en el proyecto.
RF14	El usuario (con rol administrador) podrá agregar usuarios al proyecto.
RF15	El usuario (con rol administrador) podrá quitar usuarios del proyecto.
RF16	El usuario (con rol administrador) podrá obtener y exportar un informe de los esfuerzos de cada usuario en un proyecto en un período de tiempo determinado.
RF17	El usuario podrá visualizar de forma resumida los esfuerzos realizados por todos los miembros en un proyecto, mostrando las actividades en las que los miembros han distribuido sus esfuerzos.
RF18	El usuario podrá buscar a otros usuarios.
RF19	El usuario podrá visualizar los perfiles de otro usuarios y el suyo propio.
RF20	La aplicación contará con un sistema de logros. El usuario podrá visualizar los logros que ha obtenido.
RF21	La aplicación puntuará el desempeño de los usuarios mediante un sistema de reputación. El usuario podrá compararse con otros usuarios comparando su reputación con la de los demás usuarios.

2.3.2. Requisitos No Funcionales

Cuadro 2: Tabla de Requisitos No Funcionales

Código	***Descripción del requisito***
RNF1	La aplicación permitirá la exportación de datos en formato CSV.
RNF2	La aplicación constará de una interfaz en español.
RNF3	La aplicación funcionará en dispositivos Android. La versión mínima de Android en la que la aplicación funcionará será la versión Android 4.1 (Jelly Bean).

RNF4	La aplicación no mostrará información sensible de los usuarios, como contraseñas o direcciones de correo electrónico.
RNF5	La aplicación funcionará a través de navegadores web. Concretamente, funcionará en Mozilla Firefox (versión 48.0.2) y Google Chrome (versión 52.0.2743.116)

3. Diseño de la solución

En esta sección se describe el diseño de la aplicación, incluidas cuestiones de gamificación, un mapa de navegación y la arquitectura del sistema.

3.1. Gamificación

Descrito el problema y la forma de solucionarlo de una forma general, en esta sección se describe cómo, de forma específica, se ha utilizado la gamificación en la aplicación realizada.

En la aplicación existen los usuarios, que son quienes utilizan la aplicación, proyectos y esfuerzos. A partir de estas tres entidades existentes en la aplicación, se ha creado un sistema gamificado que hace uso de ellas. Concretamente, se ha guardado constancia de cada una de las acciones del usuario dentro de la aplicación para premiar su uso a través de logros y un sistema de progresión.

Cada usuario cuenta con un nivel, llamado **reputación**, que sube o baja dependiendo de las acciones del usuario dentro de la aplicación: las acciones buenas, como crear proyectos o subir esfuerzos al servidor, incrementan la reputación del usuario; las acciones malas, como borrar proyectos o eliminar esfuerzos, decrementan la reputación del usuario.

Esta reputación es visible al resto de los usuarios en el sistema para motivar la competición entre usuarios por tener la reputación más alta estimulando el uso de la aplicación y el rellenado de esfuerzos por parte de los usuarios.

Además se utiliza un sistema privado de logros para comenzar a motivar al usuario desde el primer momento. Se ha decidido no mostrar los logros no obtenidos a los usuarios para estimular el uso de la aplicación en los usuarios 'cazadores de logros'.

A continuación se muestra un listado con todos los logros disponibles en el sistema, así como una breve descripción de cada logro:

- **Bienvenido al equipo**: Te has conectado por primera vez.
- **Esforzándote al máximo (I)**: Te has esforzado 10 veces.
- **Esforzándote al máximo** (II): Te has esforzado 25 veces.
- **Esforzándote al máximo (III)**: Te has esforzado 50 veces.
- **Esforzándote al máximo (IV)**: Te has esforzado 100 veces.
- **Esforzándote al máximo (V)**: Te has esforzado 250 veces.
- **Maestro creador de proyectos (I)**: Has creado 1 proyecto.
- **Maestro creador de proyectos (II)**: Has creado 5 proyectos.

- **Maestro creador de proyectos (III)**: Has creado 10 proyectos.
- **Maestro creador de proyectos (IV)**: Has creado 25 proyectos.
- **Maestro creador de proyectos (V)**: Has creado 50 proyectos.
- **Empleado del mes(I)**: Has obtenido 10 puntos de reputación.
- **Empleado del mes(II)**: Has obtenido 25 puntos de reputación.
- **Empleado del mes(III)**: Has obtenido 50 puntos de reputación.
- **Empleado del mes(IV)**: Has obtenido 100 puntos de reputación.
- **Empleado del mes(V)**: Has obtenido 250 puntos de reputación.

3.2. Arquitectura del sistema

El sistema se ha diseñado con una arquitectura cliente/servidor en tres niveles: los distintos clientes se comunican con un único servidor que hace de intermediario entre estos y la base de datos.

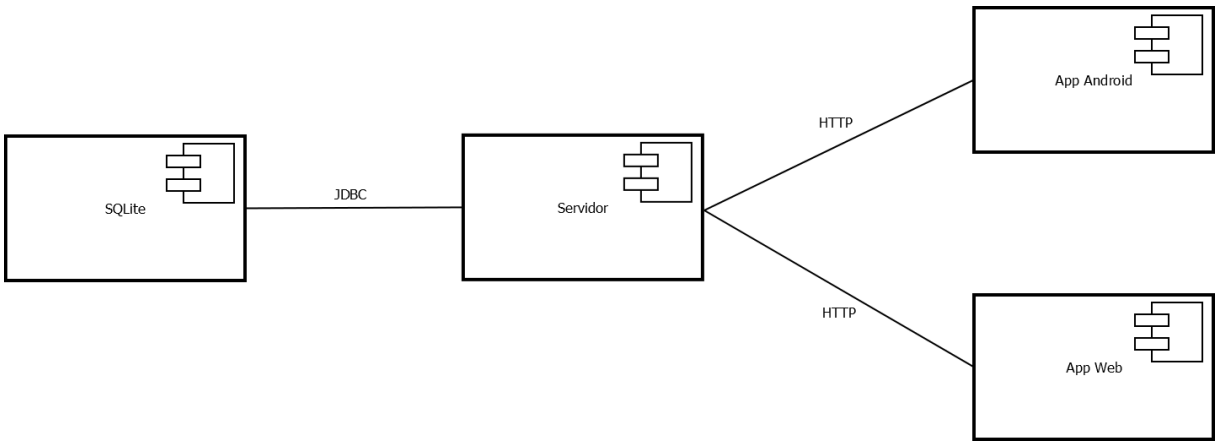


Figura 1: Arquitectura del sistema (diagrama de Componentes y Conectores)

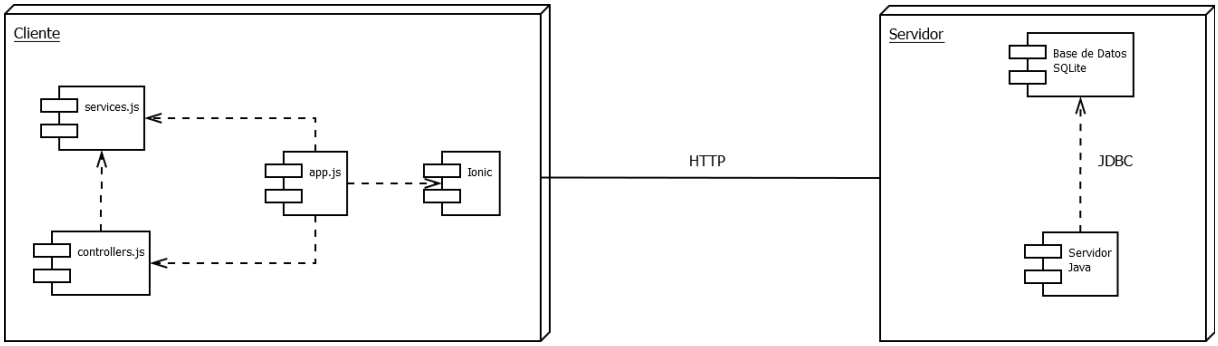


Figura 2: Diagrama de despliegue

3.2.1. Cliente

El cliente web y el cliente Android se corresponden con el mismo código Ionic exportado de diferente manera: para el cliente Android se utilizaron las herramientas que proporcionaba Ionic para crear el APK; para el cliente web se agregó un pequeño código [27] para crear un servidor Node que albergase la aplicación. Las diferencias entre los dos clientes son puramente estéticas.

La aplicación cliente se ha estructurado en una jerarquía de carpetas, agrupando los ficheros por tipo de componente que contienen: controladores, servicios o filtros.

En la raíz de la jerarquía se encuentra el módulo principal, *app.js*, que contiene el enrutamiento de la aplicación, así como un módulo para agrupar los controladores (*controllers.js*) y otro módulo para agrupar los servicios (*services.js*) para simplificar la importación de módulos en otros ficheros.

La aplicación cliente sigue el patrón **Modelo Vista VistaModelo (MVVM)** que proporciona de forma nativa AngularJS e Ionic. Esto significa que cada cambio en los datos, ya sea desde la vista o desde el modelo, se refleja en el otro componente de forma automática.

3.2.2. Servidor

El servidor se ha estructurado en un sistema de paquetes de acuerdo a la funcionalidad que ofrece cada clase. De esta forma, se tiene un paquete *bd*, que corresponde con las funciones de acceso a la base de datos y el paquete *servlets*, que corresponde con todas las clases que interactúan directamente con la aplicación cliente.

Dentro del paquete bd, se puede encontrar el paquete *bd.data*, que contiene la representación de los distintos objetos del servidor y es utilizado por bd y servlets como forma de intercambiar información.

3.3. Mapa de navegación

A continuación se muestra el mapa de navegación de la aplicación. Cuenta con una barra inferior con accesos directos a la pantalla principal de la aplicación, la búsqueda de usuarios y el perfil del usuario actualmente identificado, así como un menú lateral con los mismos accesos, acceso directo a la pantalla de esfuerzos de cada proyecto en el que participa el usuario y a la pantalla de logros.

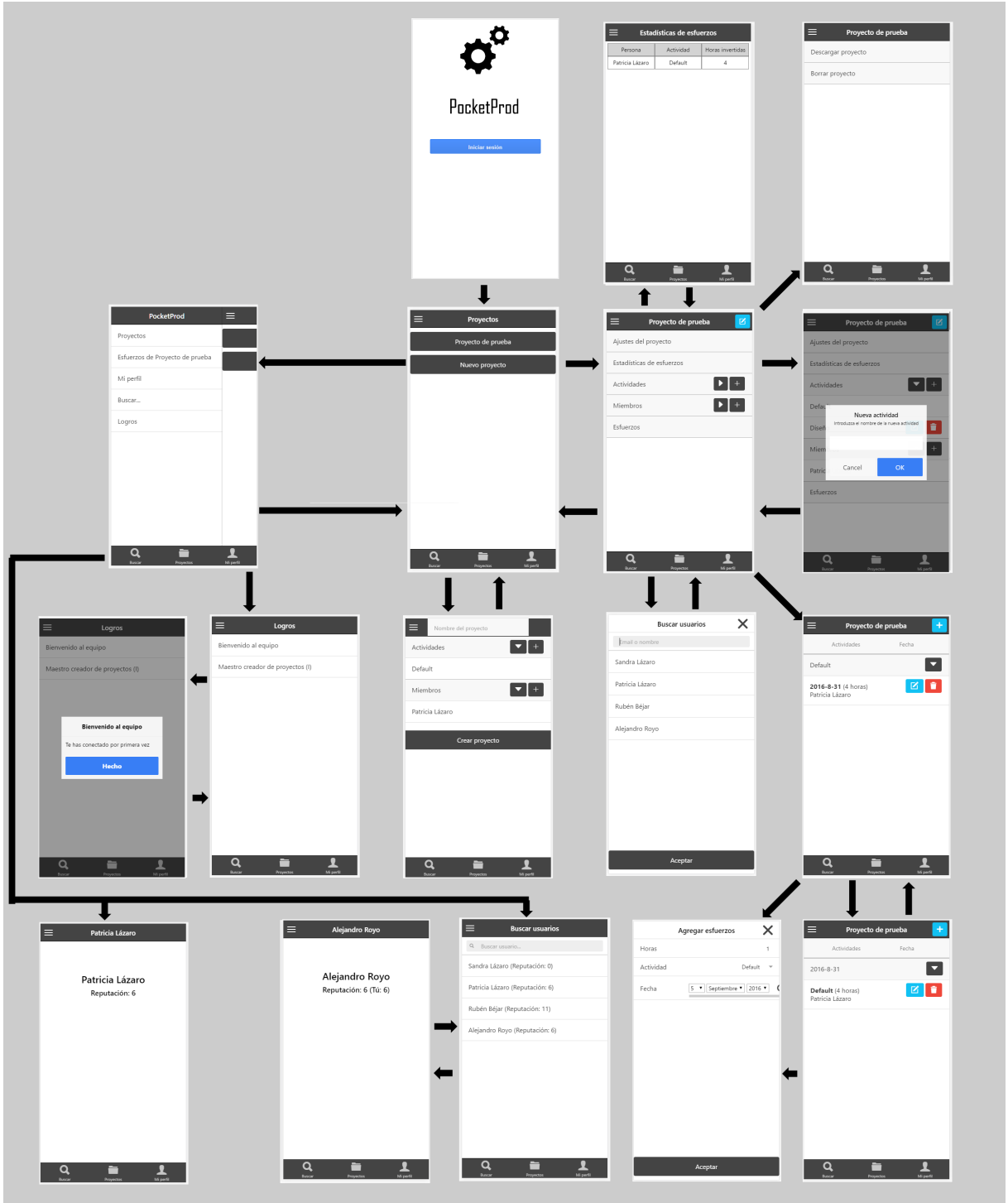


Figura 3: Mapa de navegación de la aplicación

3.4. Modelo de datos

Las principales entidades que maneja la aplicación son los conceptos de usuario, proyecto, esfuerzo y logro. A continuación se describen estas entidades y sus relaciones entre

sí.

- **Usuario**: un usuario consta de un nombre, un correo electrónico que lo identifique y su reputación. Los dos primeros atributos son obtenidos gracias al login con Google+, que además aporta el identificador único del usuario en la plataforma, que también es utilizado en la aplicación para identificar inequívocamente a los usuarios. La reputación del usuario representa el nivel de compromiso y dedicación de dicho usuario a la aplicación y al control y seguimiento de esfuerzos en la misma.

- **Proyecto**: un proyecto consta de un nombre que lo identifique, un conjunto de actividades en las que se desglosan los esfuerzos realizados y un conjunto de usuarios que participan en el proyecto (uno de ellos tendrá el rol de administrador y el resto serán usuarios miembros del proyecto). También consta de un conjunto de esfuerzos, que relacionan a los usuarios del proyecto con las actividades.

- **Esfuerzo**: se define un esfuerzo como el tiempo que un usuario determinado ha invertido en una actividad determinada en un proyecto determinado un día determinado. Por tanto, un esfuerzo consta de un usuario, que realiza los esfuerzos, una actividad en la que los esfuerzos son invertidos, una cantidad de horas que se han consumido, un día en el que se ha realizado y un proyecto al que pertenece el conjunto.

- **Logro**: un logro consta de un nombre que lo identifique y una descripción que provea más información de dicho logro. Los logros se relacionan con los usuarios, siendo los logros poseídos por los usuarios.

Para ilustrar el modelo de datos y las relaciones entre las distintas entidades que lo componen, se anexa un diagrama Entidad-Relación de la base de datos.

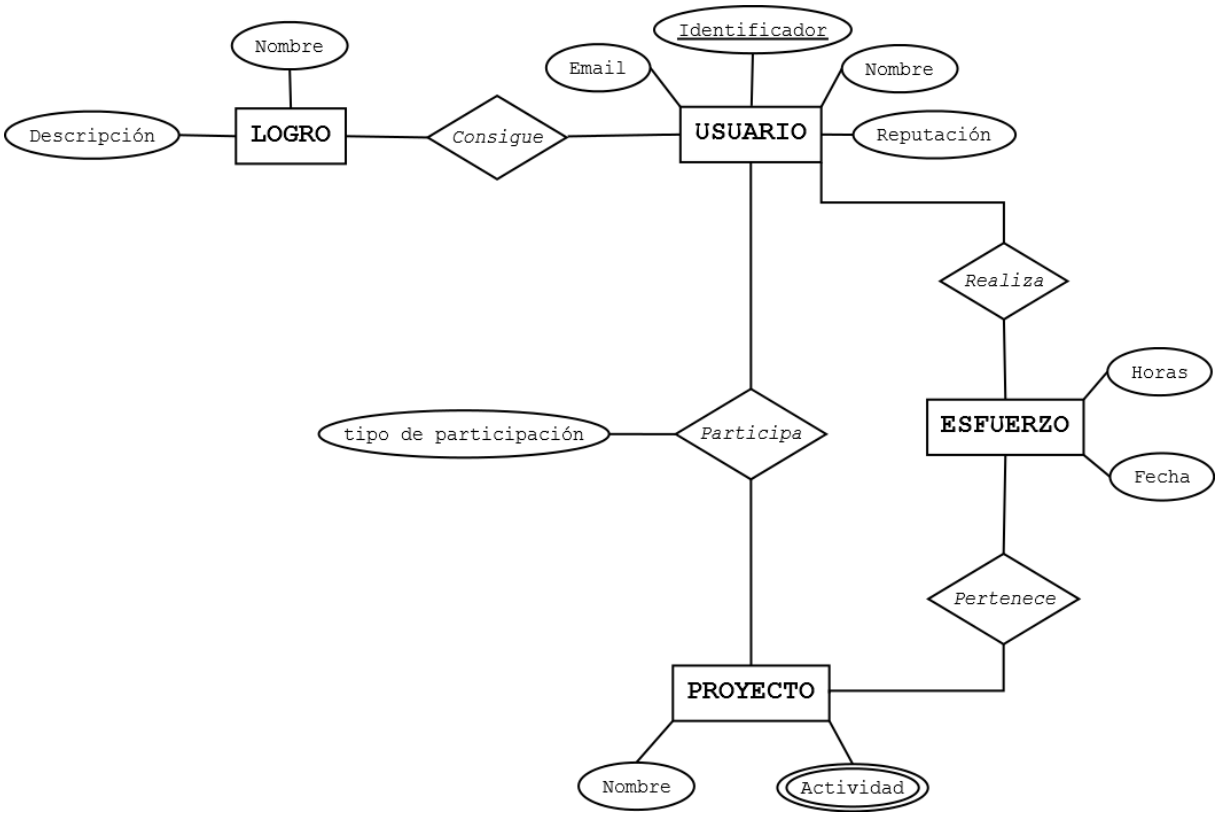


Figura 4: Diagrama Entidad-Relación

4. Implementación de la solución

A continuación se presentan distintas cuestiones referentes a la implementación de la solución, organizadas por los componentes de la misma: cliente, servidor y base de datos; así como las herramientas de desarrollo utilizadas y los resultados obtenidos de la implementación.

4.1. Cliente

En esta sección se describe la tecnología empleada en la implementación de la aplicación cliente, así como pruebas realizadas, problemas encontrados y sus soluciones y cómo se ha llevado a cabo el despliegue de la aplicación. También se nombran las fuentes utilizadas en la implementación del cliente.

4.1.1. Fuentes utilizadas

Para la implementación del datepicker se ha utilizado la implementación como directiva de AngularJS encontrada en `https://github.com/rorymadden/angular-date-dropdowns` (a 19/09/2016), que funciona tanto en el cliente web como en el cliente Android.

También se ha utilizado una función de conversión de cadenas a objetos de tipo Date descrita en `http://stackoverflow.com/a/26676413/5818247` (a 19/09/2016).

Por último, se han seguido varios tutoriales ([26] y [27]), de los que se explicará algo más en otras secciones del cliente.

4.1.2. Tecnología empleada

Para la implementación de la aplicación cliente se ha decidido utilizar el framework de aplicaciones híbridas **Ionic**. Esta tecnología se basa en AngularJS y Apache Cordova para crear aplicaciones exportables a Android, iOS y Windows 10 Universal App. Además, gracias al framework Express, se puede exportar también como aplicación web.

Al ser un framework para aplicaciones híbridas, Ionic no ofrece tan buen rendimiento sobre las plataformas móviles como una aplicación nativa; no obstante, Ionic utiliza diferentes técnicas y optimizaciones para paliar este problema.

Una alternativa a utilizar Ionic para el desarrollo de la aplicación cliente fue utilizar el SDK Android para crear una aplicación nativa, mejor optimizada para la plataforma de despliegue. Se descartó esta opción debido a que las características de la aplicación favorecían que tuviera soporte en una plataforma alternativa, propiedad que el SDK Android no permitía.

Otra alternativa a Ionic fue utilizar otro framework de aplicaciones híbridas. Actualmente existen muchos de estos frameworks, desde los que trabajan con AngularJS, HTML y CSS, como Ionic o Onsen UI, hasta los que trabajan con otros lenguajes de programación, como Xamarin, que utiliza C#.

Se decidió utilizar Ionic en vez de otro framework de aplicaciones híbridas ya que anteriormente ya se había trabajado con él. Otra razón para utilizarlo fue lo popular que es: debido a su popularidad, existe mucha documentación al respecto, así como una comunidad muy activa.

4.1.3. Pruebas

En la parte del cliente, se han realizado dos tipos de tests: los test End-to-End y los tests unitarios. A continuación se comenta en mayor profundidad los dos tipos de tests.

Los **tests E2E** se han llevado a cabo con la herramienta **protractor** [24]. Se ha comprobado que desde la pantalla inicial (la pantalla de login) se puede llegar a cualquier otra pantalla a través de la interfaz, así como sencillas comprobaciones, como crear un nuevo proyecto o comprobar que después de crear un proyecto aparece en la pantalla principal (la pantalla de proyectos).

```
ionic serve
webdriver-manager start
protractor tests/protractor.conf.js  ## considerando que
   estamos en la raiz del directorio
```

Los **tests unitarios** se han realizado con la herramienta **Karma** [22]. Se comprueba que las diversas funciones de controladores y servicios funcionen correctamente, incluidas las llamadas al servidor. Para realizar la cobertura de código se ha utilizado la herramienta **karma-coverage** [25].

```
karma start tests/karma.conf.js  ## considerando que estamos
    en la raíz del directorio
```

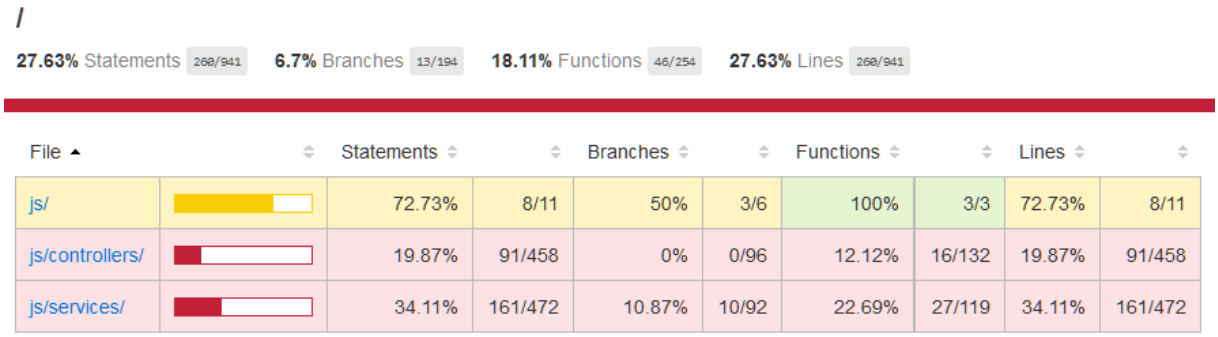


Figura 5: Cobertura de código de la aplicación cliente

4.1.4. Problemas y soluciones

Durante el desarrollo de la aplicación cliente, sobre todo en los pasos iniciales, se encontraron muchos problemas por desconocimiento de la tecnología usada: aunque se había trabajado antes con Ionic y AngularJS, no se había realizado un trabajo de tal magnitud. Estos problemas se solucionaron con el tiempo, aprendiendo más de la tecnología Ionic y AngularJS.

Login con Google+ También se encontraron problemas al implementar el login con Google+, en parte porque no había mucha información concerniente al login con Google+ en Ionic y en parte porque era la primera vez que se realizaba un login contra un tercero, utilizando su API.

Además, que fuera una aplicación híbrida llevó a varios problemas añadidos. A día de hoy, la aplicación cliente identifica al usuario contra Google+ para obtener sus credenciales y enviarlos al servidor, pero lo hace de maneras distintas para el cliente Web y el cliente Android.

Para el cliente web, se utilizan las APIs de Google+ para AngularJS y aplicaciones web. Para el cliente Android se utiliza un plugin específico [26] para hacer la misma llamada, aunque obtiene mucha menos información del usuario.

El motivo por lo que se debe utilizar un plugin en Android es porque las llamadas a las APIS de Google+ utilizadas en el cliente web abren nuevas ventanas en el navegador. La versión vanilla de Ionic no permite abrir nuevas ventanas de navegador en Android.

Para solucionar este problema hay, al menos, tres alternativas: por un lado, utilizar un plugin de identificación contra Google+, el utilizado en este trabajo por ser el más rápido y fácil de implementar; utilizar un plugin de navegador para poder abrir nuevas ventanas de navegador web e identificar al usuario a través de la API de Google+ para AngularJS,

como en el cliente web; o incorporar código nativo, que hiciera la identificación utilizando la API de Google+ para Android.

De las tres alternativas, se ha elegido la primera, utilizar un plugin que identifique contra Google+, por su simplicidad y rapidez a la hora de implementar. Se ha explorado también la posibilidad de utilizar un plugin para abrir nuevas ventanas de navegador en Android, pero los plugins utilizados no han dado resultados satisfactorios.

No obstante, el uso del plugin de identificación contra Google+ ha modificado la visión inicial del sistema, ya que la identificación en Android no devuelve tanta información como la identificación web. Esto ha dado como resultado que ya no se obtenga la imagen del usuario, y por tanto, no se almacene en el servidor.

Tests Otro problema encontrado durante el desarrollo fue la implementación de los tests. En un primer momento, hubo problemas por desconocimiento de la tecnología a utilizar y su manejo, pero con el tiempo se fueron solucionando.

No obstante, un problema que no ha podido ser resuelto satisfactoriamente es probar el login con Google+. Por problemas de tiempo, no se ha podido implementar la interfaz de pruebas en Google+ y se ha tenido que crear un botón (que solo aparece con el flag de test activado) que simula el login con Google+.

Despliegue del cliente web Al desplegar el cliente web en Heroku se encontró un problema con HTTP/HTTPS. Las peticiones al servidor están limitadas a utilizar el protocolo HTTP debido a la tecnología de despliegue que utiliza (Openshift), pero la dirección principal del cliente web en Heroku se encuentra bajo el protocolo HTTPS.

Esto hace que el cliente web no funcione bajo el protocolo HTTPS por realizar llamadas con el protocolo HTTP, que es inseguro. Este problema puede arreglarse de dos maneras: por un lado, cambiando el protocolo del cliente web a HTTP, es decir, cambiando la URL; por otro lado, cambiando las peticiones del servidor para que operen bajo el protocolo HTTPS. Esta última solución implica pagar a Openshift para añadir este protocolo al servidor. Por tanto, se ha decidido no acceder al cliente web desde el protocolo HTTPS.

4.1.5. Despliegue

En esta sección se describen las instrucciones para el despliegue de la aplicación cliente tanto en Android como en web, ya que se realiza de dos formas diferentes.

Android Para desplegar la aplicación en Android es necesario crear la aplicación APK. Para esta tarea, se utilizan los comandos que Ionic provee con su framework:

```
ionic platform add android  # solo es necesario la primera
   vez que se crea la aplicación para generar manifiestos,
   etc.
ionic prepare  # para crear el proyecto en las plataformas (
   Android en este caso).
ionic build android  # crear el APK.
```

Una particularidad es que, para crear el APK con los comandos listados a continuación, es necesario tener la key de debug de Android almacenada en debug.keystore. Para obtener esta key es necesario ejecutar el siguiente comando:

```
keytool -list -v -keystore %USERPROFILE%\.android\debug.
   keystore -alias androiddebugkey -storepass android -
   keypass android
```

De esta forma, el APK podrá conectarse con Google+ sin que la identificación falle.

Cliente web El cliente web se encuentra alojado en Heroku, por lo que es necesario tener instalado en la máquina el toolbelt de Heroku. Además, como el cliente web utiliza **ExpressJS** para crear la aplicación, será necesario tenerlo instalado. Una vez instalado el toolbelt y ExpressJS, es necesario incorporar información adicional al proyecto, concretamente:

- Fichero app.json: este fichero se crea en la raíz del proyecto y contiene información general de la aplicación, como el nombre, donde está alojado el repositorio, una descripción o las palabras clave por las que se identifica.

```
{ "name" : "Pocketprod",
  "description" : "Aplicacion␣para␣la␣recopilacion␣de␣
     esfuerzos",
  "repository" : "https://bitbucket.org/ladypeanut/
     tfg_cliente",
  "keywords" : ["ionic"] }
```

- Fichero server.js: este fichero se encarga de lanzar la aplicación web a través de ExpressJS en el puerto 8080. Se encuentra alojado en la raíz del proyecto.

```
var express = require('express');
app = express();

app.use(express.static('www'));

// CORS (Cross-Origin Resource Sharing) headers to
   support Cross-site HTTP requests
app.all('*', function(req, res, next) {
     res.header("Access-Control-Allow-Origin", "*");
     res.header("Access-Control-Allow-Headers", "X-
        Requested-With");
     next();
 });

app.set('port', process.env.PORT || 8080);
app.listen(app.get('port'), function(){
  console.log('Express server listening on port ' + app
     .get('port'));
});
```

- Fichero Procfile: se encuentra en la raíz del proyecto. Es un fichero que Heroku necesita para saber qué comando ejecutar en el despliegue de la aplicación.

```
web: node server.js
```

4.2. Servidor

En esta sección se describe la tecnología empleada en la implementación del servidor, así como pruebas realizadas, problemas encontrados y sus soluciones y cómo se ha llevado a cabo el despliegue de la aplicación.

4.2.1. Tecnología empleada

La elección de la tecnología empleada en el servidor está fuertemente ligada a la tecnología utilizada en el despliegue del mismo. En este caso, se ha elegido desplegar la aplicación servidor en **Openshift**, que ofrece *plataforma como servicio (PaaS)*.

Un PaaS es un servicio en la nube que proporciona una plataforma y entorno para que los desarrolladores creen sus aplicaciones y servicios web sin tener que preocuparse de la infraestructura que corre por debajo.

Para el despliegue de este servidor se ha utilizado un PaaS, Openshift, pero hay otras alternativas, como Heroku, otro PaaS, o utilizar una máquina de servidor. Se ha elegido Openshift por encima de los demás porque, por un lado, no es necesario preocuparse por la infraestructura, al ser un PaaS, y por otro lado, es una tecnología de despliegue que ya se ha utilizado anteriormente.

Para la implementación del servidor se ha decidido utilizar la tecnología **Java con Tomcat 7 (JBoss EWS 2.0)** [14]. La combinación de Java con Tomcat permite el despliegue de servlets, que es la tecnología de contenido dinámico web de Java.

Otras alternativas dentro de Openshift son Node.js, Perl, Python o PHP; sin embargo, se ha elegido Java (con Tomcat 7) ya que es la tecnología que más se ha utilizado durante el grado y por tanto la que más se conoce, y porque el despliegue de Java y Tomcat 7 sobre Openshift ya se ha realizado en ocasiones anteriores y se tiene más conocimiento de ello.

4.2.2. Pruebas

En la parte del servidor se han realizado tests unitarios, utilizando la tecnología **jUnit 4.1** [16]. Se ha comprobado que tanto las llamadas a la base de datos como las llamadas a los servlets funcionen correctamente, tanto si los parámetros son correctos como si son incorrectos.

Para no contaminar la base de datos de producción con datos de test, se ha creado una base de datos de test, con la que se interactúa cuando el flag de test se encuentra activado.

Los tests unitarios son lanzados cuando se actualiza el código de Openshift. Cuando se sube código a Openshift, se comienza la construcción de la aplicación web (un WAR) con Maven. El comando utilizado para comenzar la construcción del WAR incluye lanzar y pasar los tests. Si algún test falla, el WAR no se genera.

Para la cobertura de código se ha utilizado el plugin de Eclipse EclEmma. A continuación, se muestra la cobertura de código del servidor:

Element	Coverage	Covered Instru...	Missed Instruct...	Total Instructio...
pocketprod	76,9 %	9.014	2.710	11.724
src/main/java	68,6 %	5.102	2.336	7.438
bd	62,4 %	2.560	1.545	4.105
servlets	79,4 %	2.285	594	2.879
bd.data	56,6 %	257	197	454

Figura 6: Cobertura de código de la aplicación servidor

4.2.3. Problemas y soluciones

CORS Durante las pruebas del cliente contra el servidor se encontraron problemas de orígenes no permitidos, es decir, la aplicación cliente no tenía permisos para obtener los recursos del servidor, ni el cliente web ni el cliente Android.

Este problema se solucionó agregando a la respuesta de todas las peticiones que el servidor respondía la cabecera de *Access-Control-Allow-Origin* para permitir todos los orígenes.

No se encontraron más problemas durante el desarrollo de la aplicación servidor. Esto se debe probablemente a que la tecnología de despliegue y desarrollo era conocida y el proceso de creación del WAR y despliegue se habían realizado con anterioridad.

4.2.4. Despliegue

En esta sección se describen las instrucciones y pasos a seguir para el despliegue de la aplicación servidor en Openshift.

Antes de empezar la configuración para el despliegue en Openshift, es necesario configurarlo. Se requiere la instalación y setup de las client tools. Las instrucciones se encuentran en la página oficial de Openshift.

En primer lugar es necesario configurar el fichero **pom.xml**, que es el que le da a Maven la configuración de la aplicación. Es necesario agregar las dependencias de la tecnología que se utiliza, tanto en desarrollo (como el plugin sqlitedbc o la tecnología de los servlets) como en test (como jUnit).

```
<dependencies>
  <dependency>
    <groupId>org.xerial</groupId>
    <artifactId>sqlite-jdbc</artifactId>
    <version>3.8.11.2</version>
  </dependency>
  <dependency>
    <groupId>javax.servlet</groupId>
    <artifactId>javax.servlet-api</artifactId>
    <version>3.0.1</version>
    <scope>provided</scope>
  </dependency>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.1</version>
    <scope>test</scope>
  </dependency>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-test</artifactId>
    <version>4.0.5.RELEASE</version>
  </dependency>
</dependencies>
```

A continuación, es necesario habilitar la opción de test en la compilación con Maven. Por defecto, Openshift la tiene deshabilitada. Esta opción se activa en un hook (**pre_build_jbossews-2.0**). En Openshift, los hooks se encuentran en *.openshift/action_hooks*.

Un hook es un script escrito en shell, Ruby, Python... Que se ejecuta directamente en el sistema. El nombre del hook determina el momento en la compilación en que se ejecuta dicho script.

```
#!/bin/sh

export MAVEN_ARGS="clean␣package␣-Popenshift"
```

Una vez realizados los pasos anteriores, considerando que la aplicación Openshift se encuentra ya creada y el repositorio de Openshift está en la máquina local, solo queda subir los últimos cambios realizados como si se tratara de otro repositorio en Github.

```
git add -A .
git commit -am "Mensaje"
git push origin
```

4.3. Base de Datos

En esta sección se describe la tecnología empleada en la implementación del servidor, así como problemas encontrados y sus soluciones, consideraciones de pruebas y cómo se ha llevado a cabo el despliegue de la base de datos.

4.3.1. Tecnología empleada

Para elegir la tecnología empleada en la base de datos, se decidió primero dónde se alojaría, dado que esa decisión estaba fuertemente ligada a la plataforma de despliegue.

Se decidió desplegar la base de datos en Openshift. Openshift opera bajo un sistema de cartuchos que permite a una aplicación tener hasta tres cartuchos gratis, o más cartuchos si se paga. No obstante, no son necesarios más cartuchos para la aplicación, ya que solo se necesita el cartucho de Tomcat 7 para el servidor y el cartucho de la base de datos.

La decisión de desplegar en Openshift se basó en el rendimiento de las peticiones a la base de datos. Al estar desplegada aplicación y base de datos en el mismo nodo, no es necesario acceder desde el servidor a la base de datos a través de Internet, que ralentiza mucho la velocidad de las peticiones, etcétera.

Otras alternativas al despliegue en Openshift son, por un lado, desplegar la base de datos en una máquina que actúe a modo de servidor, y por el otro, alojarla en un proveedor de hosting de datos, como db4free [17].

Openshift gana por encima de estas alternativas debido a lo fácil de la instalación del cartucho en la aplicación y el mejor rendimiento que provee al acceder a una base de datos local y no remota.

Openshift ofrece varios cartuchos de bases de datos, a saber: MongoDB, MySQL, PostgreSQL y SQLite. En un primer momento se decidió utilizar MySQL, ya que era el cartucho que se había utilizado en asignaturas y aplicaciones Openshift anteriores.

MySQL y SQLite eran las dos opciones favoritas, ya que son las tecnologías que más se han utilizado a lo largo del grado y son muy parecidas en sintaxis entre sí. MongoDB no se había utilizado nunca, además de ser menos apropiada para aplicaciones Java y más apropiada para Node.js. PostgreSQL se había utilizado un poco antes, pero no se exploró la opción de desplegar esta base de datos por haber dos de las que se sabía mucho más.

En un primer momento se decidió utilizar MySQL, pero por problemas explicados en la sección 4.3.3 se tuvo que dejar de lado esa línea y utilizar SQLite.

Mientras que MySQL tiene un cartucho de phpMyAdmin en Openshift que hace mucho más sencillo el acceso y manejo de la base de datos remota, SQLite es mucho más ligera y no necesita de la instalación de un cartucho: se encuentra por defecto en la aplicación Openshift.

Por tanto, se puede concluir que, aunque se tuvo que cambiar de MySQL a SQLite, no se perdió funcionalidad ni rendimiento en la aplicación, e incluso se ganó espacio. Aunque phpMyAdmin ofrece una facilidad de manejo de la base de datos que no se encuentra en SQLite, las tablas creadas no son tan complejas como para que la falta de esa característica sea un problema.

El único problema de SQLite es que no es una base de datos de gran envergadura, es decir, puede funcionar en aplicaciones de tráfico bajo-medio, como esta, pero cuando el tráfico es intenso se pierden prestaciones. Sin embargo, no es realmente un problema a día de hoy (quizás en un futuro sea necesario realizar una migración de datos) ya que la aplicación no maneja volúmenes grandes de tráfico.

4.3.2. Pruebas

Para realizar las pruebas tanto de la aplicación cliente como del servidor, se utilizaron tablas de test en la base de datos. Esto supone que cada tabla de la base de datos de producción se encontraba clonada como tabla de test, con el mismo nombre y la terminación *TEST*. Se decidió implementar la base de datos de test de esta manera para permitir una mejor integración con MySQL si en algún momento futuro se migraban los datos.

Si no se pretendiera migrar en ningún momento la base de datos, la mejor solución habría sido crear una base de datos de test aparte. Esto supone varias ventajas con respecto a las tablas de test: no es necesario clonar las tablas, tan solo ejecutar los mismos comandos de creación de tablas en ambos ficheros; se puede acceder a las dos bases de datos con permisos de modificación a la vez, ya que son completamente independientes la una de la otra; y el código utilizado en la aplicación servidor queda mucho más sencillo y legible.

4.3.3. Problemas y soluciones

MySQL Como se explica en la sección 4.3.1, la tecnología de la base de datos elegida en primer lugar fue MySQL. No obstante, al comprobar la base de datos, se obtuvo que el servidor no podía conectarse a la base de datos.

Desde phpMyAdmin se conectaba con la base de datos, así como lanzando el comando *mysql* desde dentro de la máquina servidor, a través de SSH. No obstante, desde la aplicación Java fue imposible conectarse a la base de datos debido al error *Connection Refused*.

Después de examinar la configuración de MySQL en my.conf y varios cambios que no consiguieron solucionar el error, no se consiguió determinar el problema de conexión, aunque se llegó a la certeza de que había que modificar un valor del fichero de configuración de MySQL no configurable por los usuarios.

4.3.4. Despliegue

El despliegue de la base de datos en Openshift es automático. Una vez creado el fichero que guarda la base de datos, no es necesario realizar más acciones.

Una consideración a tener en cuenta es la localización del fichero de base de datos, ya que parte de la estructura de directorios que ofrece Openshift se borra cada vez que se actualiza el servidor, pero el fichero de base de datos debe permanecer intacto entre despliegues del servidor.

En el caso de Openshift en aplicaciones no escalables, como la que se presenta en este documento, el directorio de almacenamiento persistente se encuentra en *app-root/data*, también representado por la variable de entorno *OPENSHIFT_DATA_DIR*.

4.4. Herramientas de desarrollo

Las herramientas empleadas para el desarrollo de la aplicación fueron:

- Para el desarrollo del servidor Java: IDE Eclipse [6].
- Para el manejo y gestión de la base de datos SQLite: comando *sqlite3* después de entrar al servidor con SSH.
- Para el desarrollo de la aplicación cliente Ionic: IDE Atom [7].
- Dispositivos móviles de pruebas: Samsung Galaxy S4 Mini (Android 6.0.1), Motorola Moto G (Android 5.1), Xiaomi Mi4C (Miui 8, versión Android 5.1.1).
- Elaboración de la memoria: LaTeX[8].
- Elaboración de diagramas: Photoshop 12.1 [4], Dia [3].
- Gestión de tareas pendientes: Trello [5].
- Gestión de esfuerzos: OpenOfficeCalc [9].
- Repositorio de versiones de la aplicación cliente: Bitbucket [1].
- Repositorio de versiones de la aplicación servidor: Bitbucket y Openshift.

4.5. Resultados

El desarrollo del proyecto se realizó en dos iteraciones, cada una de un mes de duración. A continuación se detallan los resultados de la primera y segunda iteraciones del desarrollo del proyecto.

4.5.1. Primera iteración

En la primera iteración se crearon y configuraron las plataformas y tecnologías a utilizar, y se preparó la infraestructura para el desarrollo de la aplicación.

Se abordaron la creación básica de un proyecto, con la funcionalidad de añadir, modificar y borrar actividades y añadir y borrar usuarios al proyecto nuevo. También se implementó el login contra el servicio de Google+, la principal dificultad del sprint.

4.5.2. Segunda iteración

En la segunda iteración el tiempo empleado en el desarrollo del proyecto se centró por completo en el diseño e implementación de la aplicación cliente y del servidor.

Se comenzó la iteración desarrollando la vista de un único proyecto, incluidas las funcionalidades de añadir y quitar usuarios, añadir, modificar y quitar actividades del proyecto, y cambiar el nombre del proyecto.

La siguiente pantalla que se desarrolló fue la vista de esfuerzos, incluidas las distintas formas de organizar los esfuerzos (por actividad y fecha), y las funcionalidades de añadir, modificar y borrar esfuerzos. Se rediseñó la visualización de esfuerzos también por no ofrecer suficiente información al respecto.

Por último, se creó la pantalla de ajustes del proyecto, incluyendo funcionalidades de descarga de esfuerzos y borrado del proyecto, la pantalla de visualización condensada de esfuerzos y estadísticas, y se terminó de implementar la búsqueda de usuarios.

Se modificó la pantalla de visualización de un usuario para incluir la comparación de usuarios por medio de la reputación y se introdujo el sistema de logros, con sus pantallas correspondientes.

También se implementó el sistema de reputación, que hasta el momento tan solo era un número estático. Finalmente se cambiaron algunos detalles de la aplicación, como hacer más visible la reputación, el acceso directo a los esfuerzos de cada proyecto desde el menú lateral o añadir un símbolo de carga en las llamadas al servidor y Google+ para incorporar feedback al usuario.

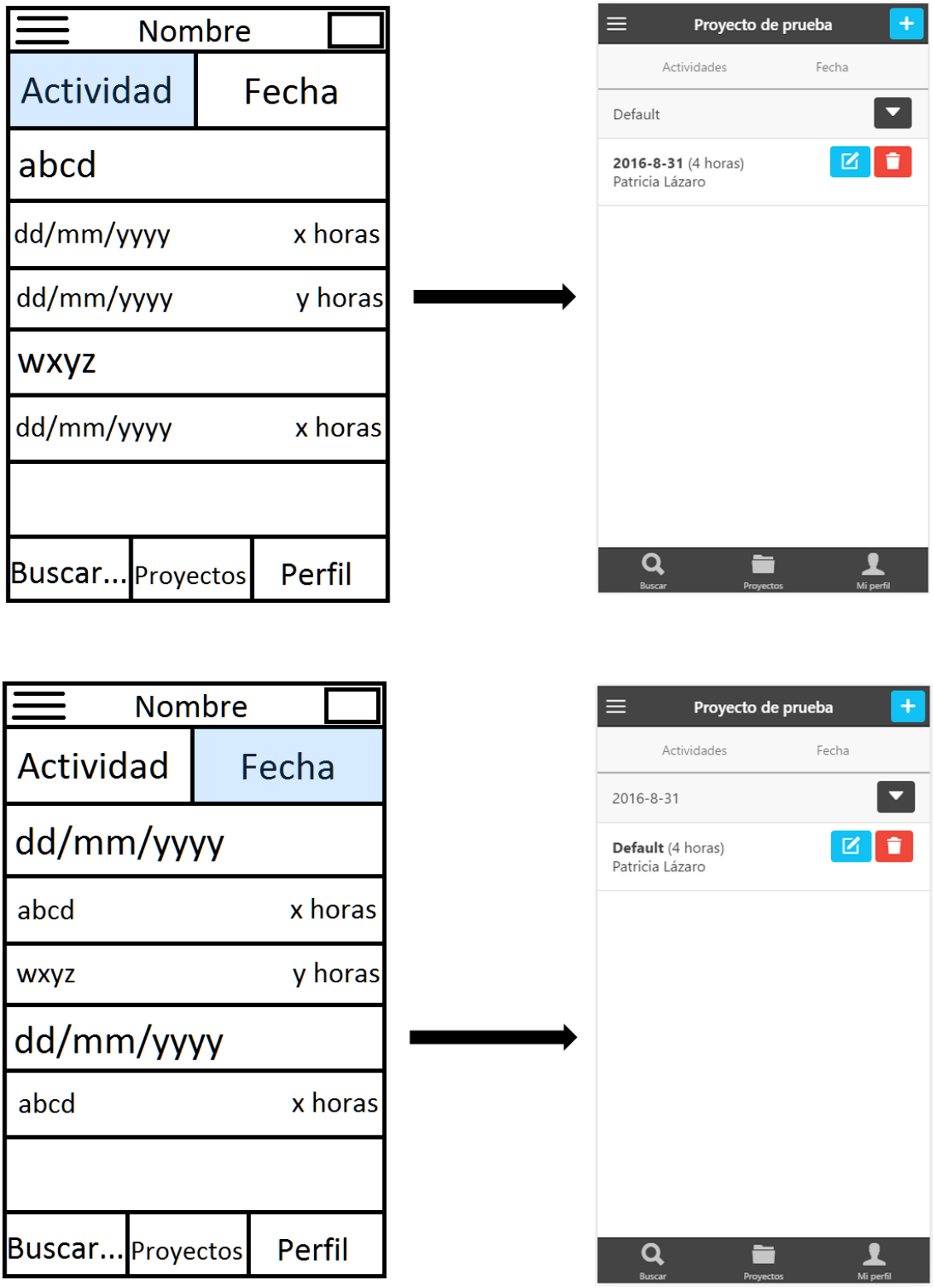


Figura 7: Bocetos de visualización de esfuerzos y pantallas finales

5. Gestión del proyecto

En esta sección se describe cómo se ha gestionado el proyecto durante su desarrollo, así como cuestiones de planificación del desarrollo, control de esfuerzos y gestión de las configuraciones.

5.1. Planificación

A continuación se describe la planificación inicial y los cambios que se tuvieron que realizar en el plan durante el desarrollo del proyecto. La planificación inicial es una versión más detallada de la planificación que se mostró en la propuesta del trabajo.

El proyecto se ha dividido en dos iteraciones, ambas con la misma duración, una fase previa y una fase final a lo largo de tres meses aproximadamente.

5.1.1. Planificación inicial

- **Fase previa** (1 semana):
  - Definición de requisitos
  - Estudio de tecnologías necesarias
- **Primera iteración** (1 mes):
  - Identificación de usuarios con Google+
  - Creación de proyectos
  - Creación de actividades en un proyecto
  - Borrado de actividades en un proyecto
  - Modificación de actividades en un proyecto
  - Inserción de usuarios en un proyecto
  - Borrado de usuarios en un proyecto
  - Visualización de esfuerzos en un proyecto
  - Inserción de esfuerzos en un proyecto
- **Segunda iteración** (1 mes):
  - Renombrado de un proyecto
  - Modificación de esfuerzos en un proyecto
  - Borrado de esfuerzos en un proyecto
  - Ajustes de un proyecto
    - Descarga de los esfuerzos de un proyecto
    - Borrado de un proyecto
  - Búsqueda de usuarios

    - Comparación de usuarios
    - Logros de la aplicación
    - Visualización condensada de los esfuerzos de un proyecto
- **Fase final** (3 semanas):
    - Compleción de la documentación técnica
    - Redacción de la memoria del proyecto

5.1.2. Cambios en la planificación inicial

Durante el desarrollo de la primera iteración, se subestimó el tiempo y esfuerzo que habría que dedicar a la identificación de usuarios con Google+, además del tiempo gastado en aprender la nueva tecnología Ionic y los problemas inesperados con la base de datos MySQL.

Todas estas cuestiones llevaron a un retraso en la iteración y a no poder realizar todas las entradas arriba expuestas. Ante esta situación, se plantearon dos alternativas: alargar el primer sprint o traspasar alguna entrada al segundo sprint.

Se decidió por traspasar las entradas de *Visualización de esfuerzos en un proyecto* e *Inserción de esfuerzos en un proyecto* al segundo sprint para rebajar la carga de trabajo del primer sprint.

El principal motivo para elegir la opción de traspasar entradas por encima de alargar el sprint fue terminar el resto de entradas a un ritmo sostenible. Otros motivos fueron que las entradas traspasadas, relacionadas con la pantalla de esfuerzos, tenían otras entradas relacionadas con esfuerzos en la segunda iteración, o que se eligió mantener constante la duración de los sprints sobre el número de entradas completadas por sprint.

5.2. Control de esfuerzos

Desde el inicio del proyecto se llevó un control de las horas de esfuerzo dedicadas a cada tarea del mismo. Concretamente, se dividió el proyecto en las siguientes grandes tareas:

- Análisis y Diseño del proyecto: engloba las tareas de análisis del proyecto, como identificación de requisitos, y las tareas de diseño, como creación de bocetos de la interfaz, entre otras.
- Implementación del proyecto: engloba la implementación del proyecto, desde el desarrollo de la aplicación cliente y servidor, pasando por la sección de pruebas, hasta la creación de la base de datos, entre otras. Esta tarea está desglosada en las siguientes subtareas:
    - Autoformación en tecnologías: tiempo dedicado al aprendizaje de nuevas tecnologías.

- Implementación del servidor: tiempo dedicado a la implementación del servidor, incluyendo la base de datos en este apartado.
- Implementación del cliente: tiempo dedicado a la implementación del cliente.

- Gestión del proyecto y documentación: engloba la creación de la documentación técnica del proyecto y la creación de esta memoria, entre otras tareas.

A continuación se muestran unas gráficas de tiempo consumido por actividades y por meses.

Figura 8: Tiempo repartido en actividades

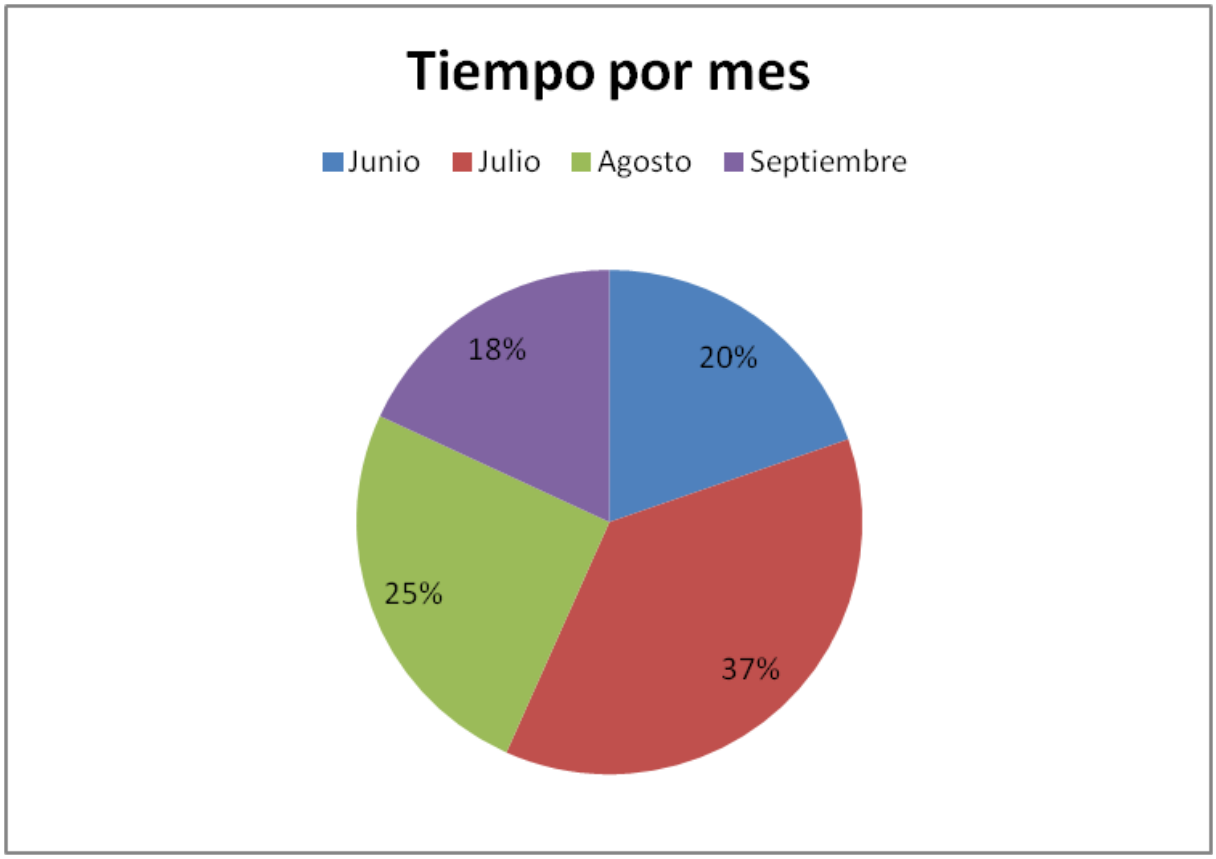


Figura 9: Tiempo repartido en meses de trabajo

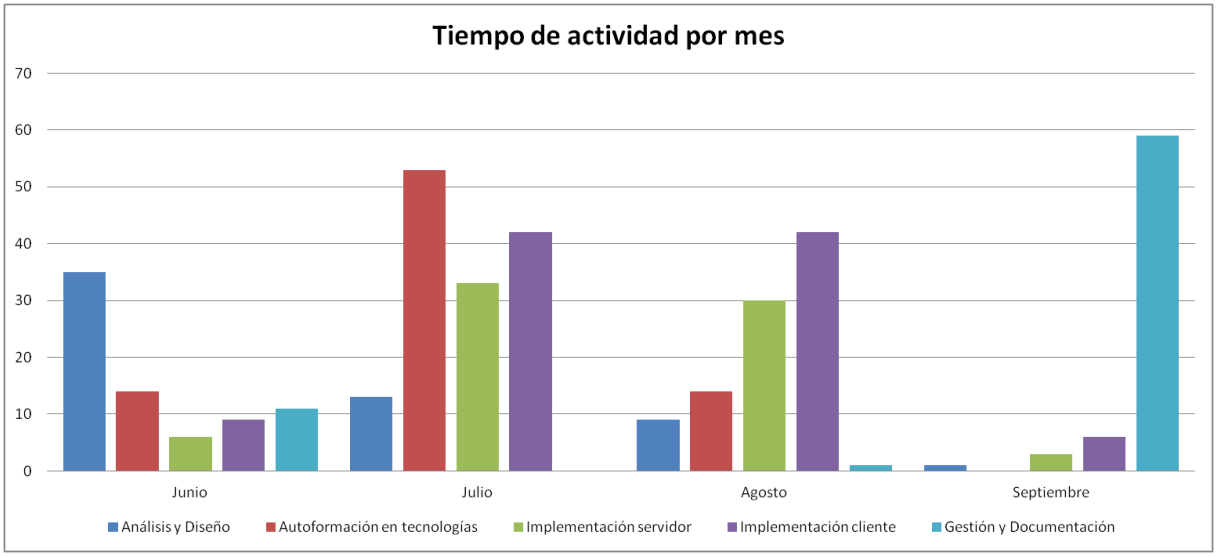


Figura 10: Tiempo desglosado en actividades y meses de trabajo

Se puede observar que julio es el mes donde más tiempo se ha trabajado en el proyecto: julio se corresponde con el primer sprint del proyecto, donde la mayoría de problemas surgieron. De todo el tiempo consumido en este mes, aproximadamente un 90 % se consumió en las tareas de *Implementación del proyecto* debido a los problemas del sprint que se han descrito anteriormente. Cabe destacar que del tiempo dedicado a la implementación del proyecto en julio, más de un 40 % se dedicó a la autoformación en tecnologías.

5.3. Gestión de configuraciones

Desde el comienzo del proyecto se utilizó un control de versiones para dar persistencia a los datos, documentación y código del proyecto.

La documentación del proyecto, como la memoria, la propuesta, la definición de requisitos, los bocetos de la interfaz y los diagramas se almacenaron en **Dropbox** [2] debido a su facilidad de uso y fácil disponibilidad.

El código del proyecto, tanto de la aplicación cliente como del servidor, fue almacenado en **Bitbucket** en sendos repositorios privados. Además del código, se almacenó en la wiki de dichos repositorios información concerniente a la aplicación cliente y servidor, como las características principales del servidor, su API o los problemas encontrados en el desarrollo de cliente y servidor y sus soluciones.

Aplicación cliente Para la aplicación cliente se ha utilizado un repositorio de Bitbucket para su persistencia. Se ha creado una sola rama en este repositorio dado que se ha trabajado en solitario en todo momento. Para recuperar la última versión del código de Bitbucket o actualizarlo se ha utilizado la herramienta GitKraken [11].

Servidor Para el desarrollo del servidor se ha utilizado un repositorio de Bitbucket, sincronizado con el repositorio privado que Openshift proporciona para el desarrollo de proyectos en su plataforma. Solo se ha utilizado una rama, *master*, pues Openshift solo lanza la construcción de una aplicación web sobre esta rama del repositorio.

Para sincronizar ambos repositorios se han realizado los siguientes cambios a la configuración por defecto: en *push origin* se encuentran las URLs del repositorio de Bitbucket y de Openshift, para que al hacer el comando *git push origin* se actualice el código automáticamente en ambos repositorios. En *pull origin* y *fetch origin* solo se encuentra la dirección de Openshift.

La herramienta utilizada para recuperar o actualizar la última versión del código del repositorio ha sido Git Bash, ya que permite tener una traza completa de lo que ha sucedido en Openshift al pasar los tests, característica que no proporciona, por ejemplo, GitKraken o la aplicación de escritorio de Github.

5.3.1. Versión de las tecnologías empleadas

- Aplicación servidor:
  - Java 1.7
  - Maven 4.0.0
  - jUnit 4.1
  - Javax Servlet 3.0.1
  - SQLite JDB 3.8.11.2
  - SQLite 3
  - Tomcat 7
- Aplicación cliente:
  - Ionic 1.3.1
  - AngularJS 1.5.3
  - Cordova 5.1.1

6. Conclusiones

A continuación, se cierra el documento con un análisis de líneas de trabajo futuras del proyecto y una valoración personal del conjunto del Trabajo de Fin de Grado.

6.1. Líneas de trabajo futuras

La principal característica que se podría mejorar de cara al futuro es la **gamificación** a través de notificaciones personalizables y mayores recompensas, como un sistema de niveles o rangos o la posibilidad de añadir rachas.

Otra característica interesante a añadir, que mejoraría la faceta competitiva de la aplicación, es la posibilidad de crear temporadas. Una temporada es un período de tiempo en el que se contabiliza la reputación de todos los usuarios que participan, y al final de la misma, se erige un podio con los usuarios con mejor rendimiento. Los usuarios podrían asociarse en grupos, y estos grupos podrían constituir el alcance de las temporadas.

También se podría personalizar el cliente web: a día de hoy, tiene un menú lateral típico de aplicaciones móviles y los campos de rellenado de texto no se enfocan automáticamente, entre otras cuestiones. Se podría, o bien modificar la interfaz actual para que fuera más usable en la web (por ejemplo, dejar el menúa lateral siempre visible en web) o crear una interfaz específica para web.

Por último, se podría mejorar la usabilidad global de la aplicación, añadiendo funcionalidades como el login automático con Google+.

6.2. Valoración personal

La lección aprendida más valiosa durante este trabajo ha sido el valor de un equipo. El desarrollo en solitario de una aplicación completa, incluyendo cliente, servidor y base de datos, puede ser muy costoso y complejo. Durante el grado de Ingeniería Informática, nunca se había pedido a un único alumno que realizara una aplicación que incluyera cliente, servidor y base de datos, y se entiende por qué.

Crear estos diferentes componentes es una tarea muy costosa donde, si uno se descuida, puede realizar malas decisiones de diseño en seguida. Dedicando el esfuerzo a un solo componente de la aplicación, como es típico en proyectos en equipo, se tiene una visión más global del estado del proyecto y, con la ayuda de los compañeros, es más difícil cometer errores.

A nivel técnico se ha profundizado el conocimiento en el framework Ionic, en el que se había trabajado un poco anteriormente y, en particular, en las tecnologías **protractor** y **karma** para la realización de tests y pruebas en el framework Ionic.

Referencias

[1] Bitbucket https://bitbucket.org (a 19/09/2016).

[2] Dropbox https://www.dropbox.com (a 19/09/2016).

[3] Dia http://dia-installer.de/index.html.es (a 19/09/2016).

[4] Adobe Photoshop http://www.adobe.com/es/products/photoshop.html (a 19/09/2016).

[5] Trello https://trello.com/ (a 19/09/2016).

[6] Eclipse IDE https://eclipse.org/ (a 19/09/2016).

[7] Atom IDE https://atom.io/ (a 19/09/2016).

[8] LaTeXhttps://www.latex-project.org/ (a 19/09/2016).

[9] Apache OpenOffice https://www.openoffice.org/es/ (a 19/09/2016).

[10] Microsoft Office https://products.office.com/es-es/home (a 19/09/2016).

[11] GitKraken www.gitkraken.com (a 19/09/2016).

[12] Openshift https://www.openshift.com/ (a 19/09/2016).

[13] Java https://www.java.com/es (a 19/09/2016).

[14] Tomcat http://tomcat.apache.org/ (a 19/09/2016).

[15] SQLite https://sqlite.org/ (a 19/09/2016).

[16] jUnit http://junit.org/junit4/ (a 19/09/2016).

[17] Db4Free https://www.db4free.net/ (a 19/09/2016).

[18] Heroku https://www.heroku.com/ (a 19/09/2016).

[19] W3CSchool (HTML y CSS) http://www.w3schools.com/ (a 19/09/2016).

[20] Ionic Framework http://ionicframework.com/ (a 19/09/2016).

[21] AngularJS https://angularjs.org/ (a 19/09/2016).

[22] Karma http://karma-runner.github.io/1.0/index.html (a 19/09/2016).

[23] Jasmine http://jasmine.github.io/1.3/introduction.html (a 19/09/2016).

[24] Protractor https://github.com/angular/protractor (a 19/09/2016).

[25] Karma-coverage https://github.com/karma-runner/karma-coverage.

[26] Ionic, tutorial de login con Google+ en una Ionic App. *Add Google+ login to your Ionic App* https://ionicthemes.com/tutorials/about/google-plus-login-with-ionic-framework (a 19/09/2016).

[27] Ionic, tutorial de despliegue de una aplicación Ionic en Heroku. *Deploying your Ionic App to Heroku* https://devdactic.com/deploying-ionic-to-heroku/ (a 19/09/2016).

[28] Toggl https://toggl.com/ (a 19/09/2016).

[29] TrackingTime https://trackingtime.co/es/ (a 19/09/2016).

[30] MyMinutes http://www.myminutesapp.com/ (a 19/09/2016).

[31] StackOverflow http://stackoverflow.com/ (a 19/09/2016).

[32] TexExchange http://tex.stackexchange.com/ (a 19/09/2016).

Anexos

A. API del servidor

A continuación se detallan los diferentes endpoints del servidor. La raíz de la URL de todos los endpoints es *http://pocketprod-554309unizar.rhcloud.com*.

- **Identificar usuario**:

  http://pocketprod-554309unizar.rhcloud.com/identificar

  Métodos: GET y POST

  Parámetros: *id*, *nombre*, *email*, *test*

  - *id*: es obligatorio. Puede contener cualquier cadena de números.
  - *nombre*: es obligatorio. Puede contener cualquier cadena de caracteres, salvo 'undefined' y 'null'.
  - *email*: es obligatorio. Puede contener cualquier cadena de caracteres que cumpla con el patrón *.*@.**.
  - *test*: no es obligatorio. Su valor no es importante, tan solo su presencia o ausencia para utilizar o no la base de datos de test o de producción.

  Devuelve un código 400 (Bad request) si la petición falla, o un código 200 (OK) si la petición es un éxito. Si es la primera vez que el usuario se identifica, se guardan su identificador, nombre y email. Si ya se encontraba en la base de datos, se actualizan su nombre y email. También devuelve un objeto JSON con los nuevos logros obtenidos por la identificación del usuario, que es como sigue:

```
{"logros" : [
  {"nombre" : "cadena",
   "descripcion" : "cadena"},
  ...
]}
```

- **Buscar usuarios por nombre o email**:

  http://pocketprod-554309unizar.rhcloud.com/buscar

  Métodos: GET y POST

  Parámetros: *nombre*, *email*, *test*

  - *nombre*: no es obligatorio. Puede contener cualquier cadena de caracteres.
  - *email*: no es obligatorio. Puede contener cualquier cadena de caracteres.
  - *test*: no es obligatorio. Su valor no es importante, tan solo su presencia o ausencia para utilizar o no la base de datos de test o de producción.

  Devuelve un código 400 (Bad request) si la petición falla, o un código 200 (OK) si es un éxito, junto a un objeto JSON, que es como sigue:

```
{"lista" : [
  {"id" : "cadena",
   "nombre" : "cadena",
   "email" : "cadena",
   "reputacion" : entero},
  ...
]}
```

- **Buscar usuarios por identificador de usuario**:

  http://pocketprod-554309unizar.rhcloud.com/buscarId

  Métodos: GET y POST

  Parámetros: *id*, *test*

  - *id*: es obligatorio. Puede contener cualquier cadena de caracteres.
  - *test*: no es obligatorio. Su valor no es importante, tan solo su presencia o ausencia para utilizar o no la base de datos de test o de producción.

  Devuelve un código 400 (Bad request) si la petición falla, o un código 200 (OK) si es un éxito, junto a un objeto JSON que es como sigue:

```
{"user" : {
  "nombre" : "cadena",
  "email" : "cadena",
  "reputacion" : entero,
  "logros" : [
    {"nombre" : "cadena",
     "descripcion" : "cadena"},
    ...
  ]
}
}
```

- **Crear Proyecto**:

  http://pocketprod-554309unizar.rhcloud.com/crear

  Métodos: GET y POST

  Parámetros: *nombre*, *actividad*, *user*, *test*

  - *nombre*: es obligatorio. Puede contener cualquier cadena de caracteres, salvo 'null' y 'undefined'.
  - *actividad*: es obligatorio, puede tener varios valores. Puede contener cualquier cadena de caracteres.
  - *user*: es obligatorio, puede tener varios valores. Cada usuario se estructura como *identificador_tipo*. Tipo es 0 para usuario y 1 para administrador.
  - *test*: no es obligatorio. Su valor no es importante, tan solo su presencia o ausencia para utilizar o no la base de datos de test o de producción.

Devuelve un código 400 (Bad request) si la petición falla, o un código 200 (OK) si se crea el proyecto correctamente. También devuelve un objeto JSON con los nuevos logros obtenidos por la creación del proyecto, que es como sigue:

```
{"logros" : [
  {"nombre" : "cadena",
   "descripcion" : "cadena"},
  ...
]}
```

- **Renombrar proyecto**:

  http://pocketprod-554309unizar.rhcloud.com/rename

  Métodos: GET y POST

  Parámetros: *id*, *nombre*, *test*

  - *id*: es obligatorio. Puede contener cualquier cadena de números.
  - *nombre*: es obligatorio. Puede contener cualquier cadena de caracteres, salvo 'null' y 'undefined'.
  - *test*: no es obligatorio. Su valor no es importante, tan solo su presencia o ausencia para utilizar o no la base de datos de test o de producción.

  Devuelve un código 400 (Bad request) si la petición falla, o un código 200 (OK) si la petición es exitosa.

- **Eliminar proyecto**:

  http://pocketprod-554309unizar.rhcloud.com/remove

  Métodos: GET y POST

  Parámetros: *id*, *test*

  - *id*: es obligatorio. Puede contener cualquier cadena de números.
  - *test*: no es obligatorio. Su valor no es importante, tan solo su presencia o ausencia para utilizar o no la base de datos de test o de producción.

  Devuelve un código 400 (Bad request) si la petición falla, o un código 200 (OK) si la petición es exitosa.

- **Obtener los proyectos de un usuario**:

  http://pocketprod-554309unizar.rhcloud.com/getProyectos

  Métodos: GET y POST

  Parámetros: *user* y *test*

  - *user*: es obligatorio. Puede contener cualquier cadena de caracteres, salvo 'undefined' y 'null'.
  - *test*: no es obligatorio. Su valor no es importante, tan solo su presencia o ausencia para utilizar o no la base de datos de test o de producción.

Devuelve un código 400 (Bad request) si la petición falla, o un código 200 (OK) si la petición es un éxito junto a un objeto JSON que es como sigue:

```
{"lista" : [
  {"id" : "cadena",
   "nombre" : "cadena"},
  ...
]}
```

- **Obtener la información de un proyecto**:

  http://pocketprod-554309unizar.rhcloud.com/getProjectInfo

  Métodos: GET y POST

  Parámetros: *id*, *test*

  - *id*: es obligatorio. Puede contener una cadena de números.
  - *test*: no es obligatorio. Su valor no es importante, tan solo su presencia o ausencia para utilizar o no la base de datos de test o de producción.

Devuelve un código 400 (Bad request) si la petición falla, o un código 200 (OK) si la petición es exitosa. También devuelve un objeto JSON que es como sigue:

```
{"proyecto" : {
  "id" : "cadena",
  "nombre" : "cadena",
  "activities" : [
    {"nombre" : "cadena",
     "quitar" : true|false},
    ...
  ],
  "members" : [
    {"id" : "cadena",
     "nombre" : "cadena",
     "quitar" : true|false},
    ...
  ]
}}
```

- **Agregar actividades a un proyecto**:

  http://pocketprod-554309unizar.rhcloud.com/add-actividades

  Métodos: GET y POST

  Parámetros: *id*, *actividad*, *test*

  - *id*: es obligatorio. Puede contener cualquier cadena de números.
  - *actividad*: es obligatorio. Puede contener cualquier cadena de caracteres, salvo 'undefined' y 'null'.

- *test*: no es obligatorio. Su valor no es importante, tan solo su presencia o ausencia para utilizar o no la base de datos de test o de producción.

Devuelve un código 400 (Bad request) si la petición falla, o un código 200 (OK) si se ha podido añadir la actividad al proyecto.

- **Eliminar/modificar actividades de un proyecto**:

  http://pocketprod-554309unizar.rhcloud.com/remove-actividades

  Métodos: GET y POST

  Parámetros: *id*, *actividad*, *nueva*, *test*

  - *id*: es obligatorio. Puede contener cualquier cadena de números.
  - *actividad*: es obligatorio. Puede contener cualquier cadena de caracteres, salvo 'undefined' y 'null'.
  - *nueva*: es obligatorio. Puede contener cualquier cadena de caracteres, salvo 'undefined' y 'null'.
  - *test*: no es obligatorio. Su valor no es importante, tan solo su presencia o ausencia para utilizar o no la base de datos de test o de producción.

  Devuelve un código 400 (Bad request) si la petición falla, o un código 200 (OK) si se ha podido cambiar el nombre de la actividad (si había parámetro *nueva*) o se ha podido eliminar.

- **Ver esfuerzos de un proyecto**:

  http://pocketprod-554309unizar.rhcloud.com/getEsfuerzos

  Métodos: GET y POST

  Parámetros: *pid*, *uid*, *test*

  - *pid*: es obligatorio. Puede contener cualquier cadena de números.
  - *uid*: no es obligatorio. Puede contener cualquier cadena de números.
  - *test*: no es obligatorio. Su valor no es importante, tan solo su presencia o ausencia para utilizar o no la base de datos de test o de producción.

  Devuelve un código 400 (Bad request) si la petición falla, o un código 200 (OK) si la petición es exitosa. También devuelve un objeto JSON que es como sigue:

```
{"lista" : [
  {"actividad" : "cadena",
   "fecha" : "yyyy-MM-dd",
   "usuario" : {"id" : "cadena",
                "nombre" : "cadena"},
   "horas" : entero},
  ...
]}
```

- **Agregar esfuerzos de un proyecto**:

  http://pocketprod-554309unizar.rhcloud.com/addEsfuerzos

  Métodos: GET y POST

  Parámetros: *pid*, *uid*, *horas*, *day*, *month*, *year*, *act*, *test*

  - *pid*: es obligatorio. Puede contener cualquier cadena de números.
  - *uid*: es obligatorio. Puede contener cualquier cadena de caracteres, salvo 'undefined' y 'null'.
  - *horas*: es obligatorio. Puede contener cualquier número entero positivo.
  - *day*: es obligatorio. Puede contener cualquier número entero positivo.
  - *month*: es obligatorio. Puede contener cualquier número entero positivo.
  - *year*: es obligatorio. Puede contener cualquier número entero positivo.
  - *act*: es obligatorio. Puede contener cualquier cadena de caracteres, salvo 'undefined' y 'null'.
  - *test*: no es obligatorio. Su valor no es importante, tan solo su presencia o ausencia para utilizar o no la base de datos de test o de producción.

  Devuelve un código 400 (Bad request) si la petición falla, o un código 200 (OK) si la petición es exitosa, es decir, se ha conseguido introducir los esfuerzos del usuario. También devuelve un objeto JSON que es como sigue:

```
{"logros" : [
  {"nombre" : "cadena",
   "descripcion" : "cadena"},
  ...
]}
```

- **Eliminar esfuerzos de un proyecto**:

  http://pocketprod-554309unizar.rhcloud.com/rmvEsfuerzos

  Métodos: GET y POST

  Parámetros: *pid*, *uid*, *day*, *month*, *year*, *act*, *test*

  - *pid*: es obligatorio. Puede contener cualquier cadena de números.
  - *uid*: es obligatorio. Puede contener cualquier cadena de caracteres, salvo 'undefined' y 'null'.
  - *day*: es obligatorio. Puede contener cualquier número entero positivo.
  - *month*: es obligatorio. Puede contener cualquier número entero positivo.
  - *year*: es obligatorio. Puede contener cualquier número entero positivo.
  - *act*: es obligatorio. Puede contener cualquier cadena de caracteres, salvo 'undefined' y 'null'.
  - *test*: no es obligatorio. Su valor no es importante, tan solo su presencia o ausencia para utilizar o no la base de datos de test o de producción.

Devuelve un código 400 (Bad request) si la petición falla, o un código 200 (OK) si la petición es exitosa.

- **Agregar usuarios a un proyecto**:

  http://pocketprod-554309unizar.rhcloud.com/addUser

  Métodos: GET y POST

  Parámetros: *uid*, *pid*, *test*

  - *uid*: es obligatorio. Puede contener cualquier cadena de caracteres, salvo 'null' y 'undefined'.
  - *pid*: es obligatorio. Puede contener cualquier cadena de números.
  - *test*: no es obligatorio. Su valor no es importante, tan solo su presencia o ausencia para utilizar o no la base de datos de test o de producción.

  Devuelve un código 400 (Bad request) si la petición falla, o un código 200 (OK) si la petición es exitosa.

- **Quitar usuarios de un proyecto**:

  http://pocketprod-554309unizar.rhcloud.com/rmUser

  Métodos: GET y POST

  Parámetros: *uid*, *pid*, *test*

  - *uid*: es obligatorio. Puede contener cualquier cadena de caracteres, salvo 'null' y 'undefined'.
  - *pid*: es obligatorio. Puede contener cualquier cadena de números.
  - *test*: no es obligatorio. Su valor no es importante, tan solo su presencia o ausencia para utilizar o no la base de datos de test o de producción.

  Devuelve un código 400 (Bad request) si la petición falla, o un código 200 (OK) si la petición es exitosa.

- **Descargar esfuerzos de un proyecto**:

  http://pocketprod-554309unizar.rhcloud.com/download

  Métodos: GET y POST

  Parámetros: *dayInit*, *monthInit*, *yearInit*, *dayEnd*, *monthEnd*, *yearEnd*, *pid*, *test*

  - *dayInit*: no es obligatorio. Puede contener cualquier entero.
  - *monthInit*: no es obligatorio. Puede contener cualquier entero.
  - *yearInit*: no es obligatorio. Puede contener cualquier entero.
  - *dayEnd*: no es obligatorio. Puede contener cualquier entero.
  - *monthEnd*: no es obligatorio. Puede contener cualquier entero.
  - *yearEnd*: no es obligatorio. Puede contener cualquier entero.

- *pid*: no es obligatorio. Puede contener cualquier cadena de números.
- *test*: no es obligatorio. Su valor no es importante, tan solo su presencia o ausencia para utilizar o no la base de datos de test o de producción.

Devuelve un código 400 (Bad request) si la petición falla, o un código 200 (OK) si la petición es exitosa. También devuelve una respuesta en formato CSV que es como sigue:

```
UID, UNAME, ACTIVITY, DAY, MONTH, YEAR, HOURS
uid, nombre, actividad, dia, mes, anio, horas
...
```

- **Esfuerzos condensados de los usuarios**:

  http://pocketprod-554309unizar.rhcloud.com/estadisticas

  Métodos: GET y POST

  Parámetros: *pid*, *test*

  - *pid*: es obligatorio. Puede contener una cadena de números.
  - *test*: no es obligatorio. Su valor no es importante, tan solo su presencia o ausencia para utilizar o no la base de datos de test o de producción.

  Devuelve un código 400 (Bad request) si la petición falla, o un código 200 (OK) si la petición es exitosa. También devuelve un objeto JSON que es como sigue:

```
{"list" : [
  {"nombre" : "cadena",
   "actividad" : "cadena",
   "horas" : entero},
  ...
]}
```

- **Obtener los logros de un usuario**:

  http://pocketprod-554309unizar.rhcloud.com/getLogros

  Métodos: GET y POST

  Parámetros: *uid*, *test*

  - *uid*: es obligatorio. Puede contener cualquier cadena de números.
  - *test*: no es obligatorio. Su valor no es importante, tan solo su presencia o ausencia para utilizar o no la base de datos de test o de producción.

  Devuelve un código 400 (Bad request) si la petición falla, o un código 200 (OK) si la petición es exitosa. También devuelve un objeto JSON que es como sigue:

```
{"logros" : [
  {"nombre" : "cadena",
   "descripcion" : "cadena"},
  ...
]}
```

B. Manual de usuario

A continuación se detalla el manual de usuario de la aplicación móvil para la recopilación y seguimiento de esfuerzos en proyectos en equipo con técnicas de gamificación y serious games para fomentar su uso. Esta aplicación también tiene un cliente web, pero la interfaz de este cliente web es la misma que la de la aplicación móvil para Android, de forma que se realizará este manual de usuario sobre la versión de móvil.

Las imágenes de este manual, o el proceso que en ellas se ve, puede varias con actualizaciones de la aplicación.

B.1. Pantalla de login

La pantalla de login es la primera pantalla a la que el usuario accede al iniciar la aplicación o acceder a través del cliente web. Consta de una imagen que actúa como logo de la aplicación y un botón (1) para iniciar sesión a través de Google+.

Para comenzar a utilizar la aplicación, el usuario debe proceder a pulsar el botón de acceso a través de Google+ (1). Google+ le pedirá permiso para iniciar sesión en la aplicación y si el usuario acepta, se procederá a identificarlo en la aplicación.

Si la identificación se completa exitosamente, la aplicación continuará a la pantalla principal. Si la identificación no se completa, se volverá a la pantalla de login y aparecerá un mensaje de error.

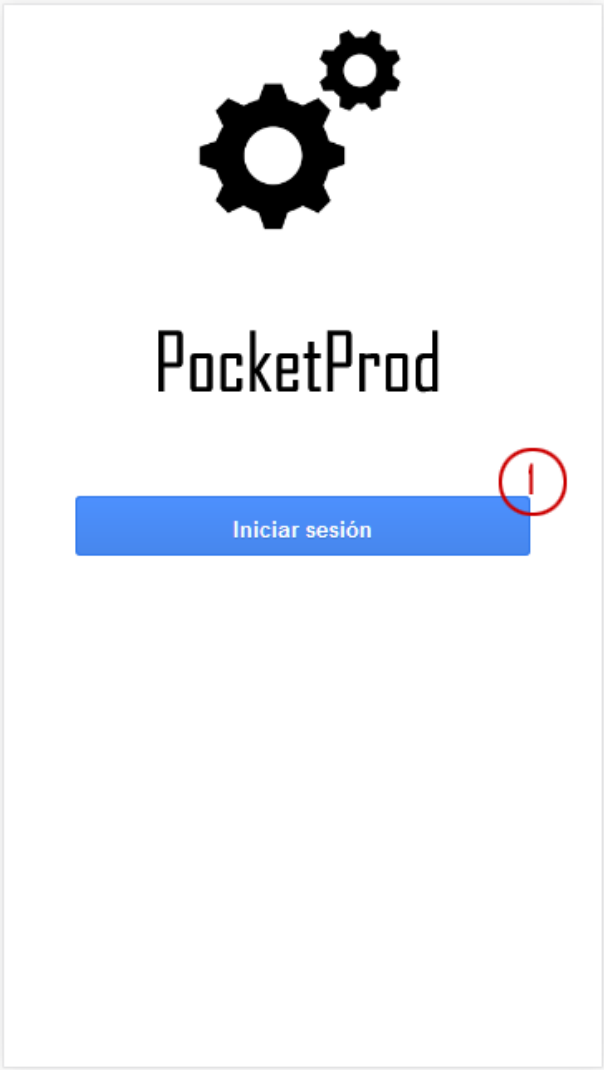


Figura 11: Pantalla de login

B.2. Pantalla principal de la aplicación

La pantalla principal de la aplicación muestra los proyectos en los que el usuario participa, así como la estructura que sigue la aplicación a lo largo de todas sus pantallas.

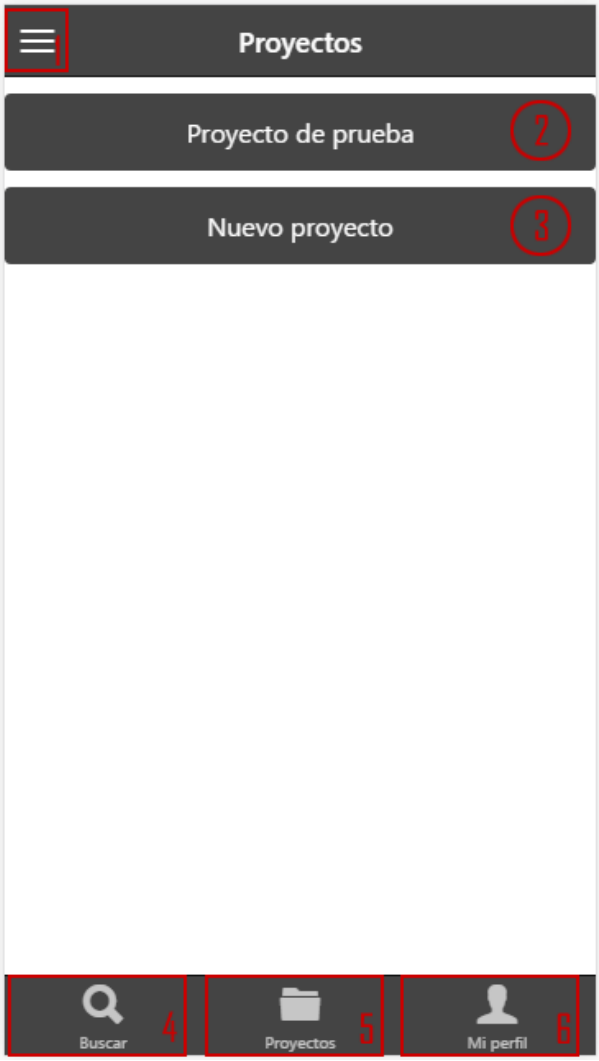


Figura 12: Pantalla principal

El botón para abrir el menú lateral (1) se encuentra en todas las pantallas para poder acceder a los accesos directos más importantes de la aplicación.

A su vez, la aplicación consta de una barra de navegación inferior con tres botones que se encuentra presente en todas las pantallas de la aplicación, y que contiene también accesos directos más generales.

El botón de búsqueda de usuarios (4) abre la pantalla para buscar a usuarios en la aplicación. El botón de visión general de proyectos (5) devuelve al usuario a esta misma pantalla. El botón de visualización del perfil (6) abre el perfil del usuario actualmente identificado en la aplicación.

El área de contenido está conformada por una lista de botones. El último botón de esta lista es el botón de *Nuevo proyecto* (3), que lleva al usuario a la pantalla de creación de nuevo proyecto. El resto de botones (2) se corresponde con los proyectos en los que participa el usuario: pulsar en uno de estos botones enviará al usuario a la pantalla de gestión del proyecto en el que haya pulsado.

B.3. Menú lateral de la aplicación

El menú lateral de la aplicación se puede abrir pulsando en cualquier momento en el botón de apertura del menú lateral (1) o arrastrando con el dedo de izquierda a derecha. En el cliente web, se debe pulsar un punto de la pantalla y arrastrar el ratón hacia la derecha para abrirlo.

Este menú lateral contiene los accesos directos más importantes de la aplicación. También incluye los accesos directos de la barra de navegación inferior, como el botón de visión general de proyectos (2), la visualización del perfil (4) y búsqueda de usuarios (5).

También contiene el botón de acceso a logros (6), que lleva al usuario a la pantalla de visualización de los logros que ha obtenido. Por último, por cada proyecto en el que el usuario participe se muestra un botón de acceso directo a los esfuerzos de estos proyectos (3).

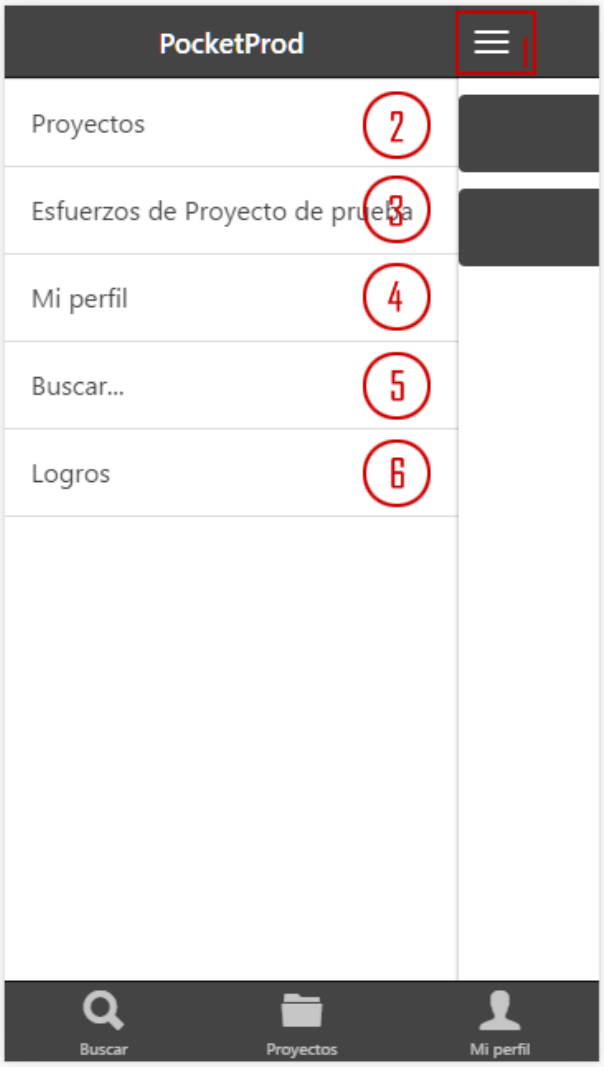


Figura 13: Menú lateral de la aplicación

B.4. Pantalla de visualización de un proyecto

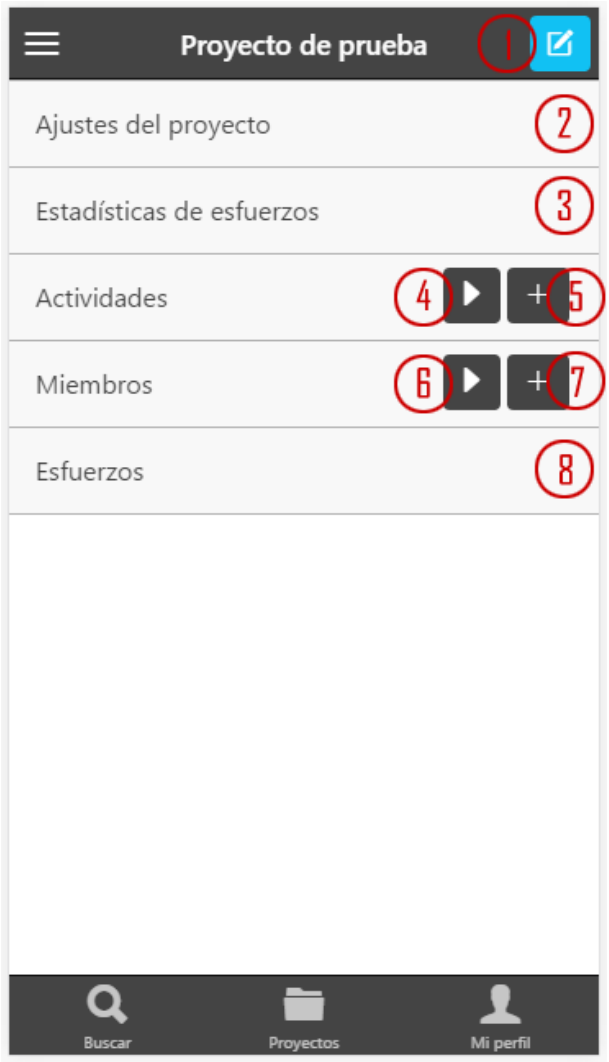


Figura 14: Pantalla de visualización de un proyecto

La pantalla de visualización de un proyecto muestra la información detallada de dicho proyecto. La pantalla cambia si el usuario es administrador o no para reflejar la funcionalidad exclusiva del administrador del proyecto.

La pantalla mostrada en la imagen 14 se corresponde con la visión del proyecto que tiene el administrador. La pantalla del usuario contaría con la misma información que la del administrador, salvo la ausencia del botón de renombrado de proyecto (1), el botón de ajustes del proyecto (2), el botón de añadir actividades (5) y el botón de añadir miembros (7).

Para cambiar el nombre al proyecto (funcionalidad restringida al administrador del proyecto) se debe pulsar sobre el botón de renombrado (1). A continuación aparecerá una ventana emergente pidiendo el nuevo nombre del proyecto y dos botones, uno para cancelar la acción y el otro para confirmarla. Una vez insertado el nuevo nombre del proyecto y pulsado el botón de confirmar, la ventana emergente desaparecerá y se habrá cambiado el nombre del proyecto, o si ha habido un error, aparecerá un mensaje de error.

Para acceder a los ajustes del proyecto se debe pulsar sobre el botón de ajustes del proyecto (2) (funcionalidad restringida al administrador del proyecto).

Para acceder a la visión condensada de esfuerzos, también llamada estadísticas de esfuerzos, es necesario pulsar sobre el botón de estadísticas de esfuerzos (3), que abrirá la pantalla de estadísticas.

Para desplegar las actividades y poder visualizar su totalidad, se debe pulsar el botón de despliegue de actividades (4). Para añadir una nueva actividad al proyecto, es necesario pulsar el botón de añadido de actividades (5). Todos los usuarios, independientemente de su estatus en el proyecto, pueden abandonarlo eliminándose a ellos mismos del proyecto.

Análogamente, para desplegar los usuarios que participan en el proyecto se debe pulsar sobre el botón de despliegue de usuarios (6), y para añadir un nuevo usuario al proyecto, se debe pulsa el botón de añadido de usuarios (7).

Por último, para visualizar los esfuerzos (esfuerzos personales si se es usuario del proyecto, o los esfuerzos de todos los usuarios si se es administrador), así como añadir, actualizar o borrar esfuerzos, es necesario pulsar sobre el botón de esfuerzos (8), que llevará al usuario a la pantalla de visualización de esfuerzos.

B.5. Pantalla de creación de un proyecto

La pantalla de creación de un nuevo proyecto contiene todos los componentes para nombrar un proyecto, incluir actividades y usuarios, y crearlo.

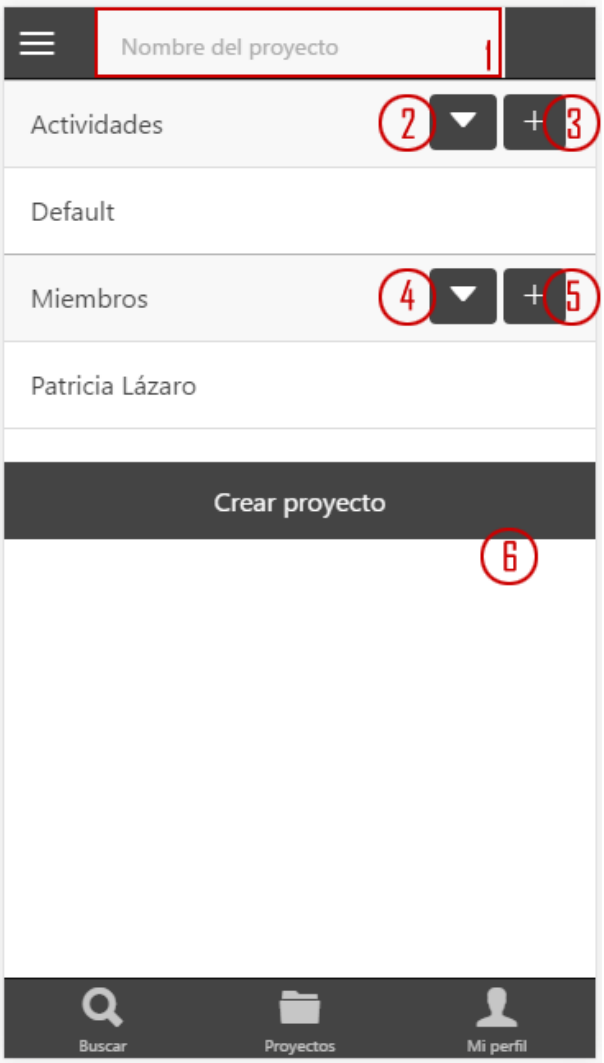


Figura 15: Pantalla de creación de un proyecto

Para introducir el nombre del nuevo proyecto, es necesario rellenar el campo de texto de *Nombre del proyecto* (1). Es un campo obligatorio y si no se rellena aparecerá un mensaje de error cuando se quiera crear el proyecto.

Los botones de despliegue de actividades (2) y despliegue de miembros participantes (4) tienen la función de colapsar o desplegar la lista de actividades y miembros participantes respectivamente.

En la lista de actividades habrá inicialmente una actividad llamada Default, que es la actividad por defecto y no puede eliminarse. Si una actividad creada se elimina en un momento posterior, todos los esfuerzos de esta actividad se asimilarán en la actividad Default.

La lista de miembros participantes contiene inicialmente al miembro que crea el proyecto, que será el administrador de dicho proyecto.

El botón de añadir nueva actividad (3) abre una ventana emergente con un campo de texto para introducir el nombre de la nueva

actividad y sendos botones de confirmar y cancelar para añadir o no una nueva actividad al proyecto.

El botón de añadir nuevo miembro participante (5) abre una ventana emergente con un campo de texto para buscar por email o nombre al usuario a añadir, así como un botón para cancelar la adición y uno para confirmarla. Solo se puede introducir un nuevo miembro participante cada vez.

Por último, cuando se haya terminado de añadir, modificar y borrar actividades, y se tengan en la lista los miembros participantes que se desea tener, el usuario deberá pulsar en el botón de creación del proyecto (6) para crear el nuevo proyecto. Hasta que el proyecto no se haya creado, el proyecto no aparecerá en el tablón de ningún miembro participante.

Si el usuario se equivoca al introducir el nombre del proyecto, actividades o miembros participantes, más adelante lo podrá cambiar entrando a la pantalla de visualización del proyecto, una vez creado.

B.6. Ventana emergente de creación de una nueva actividad en un proyecto

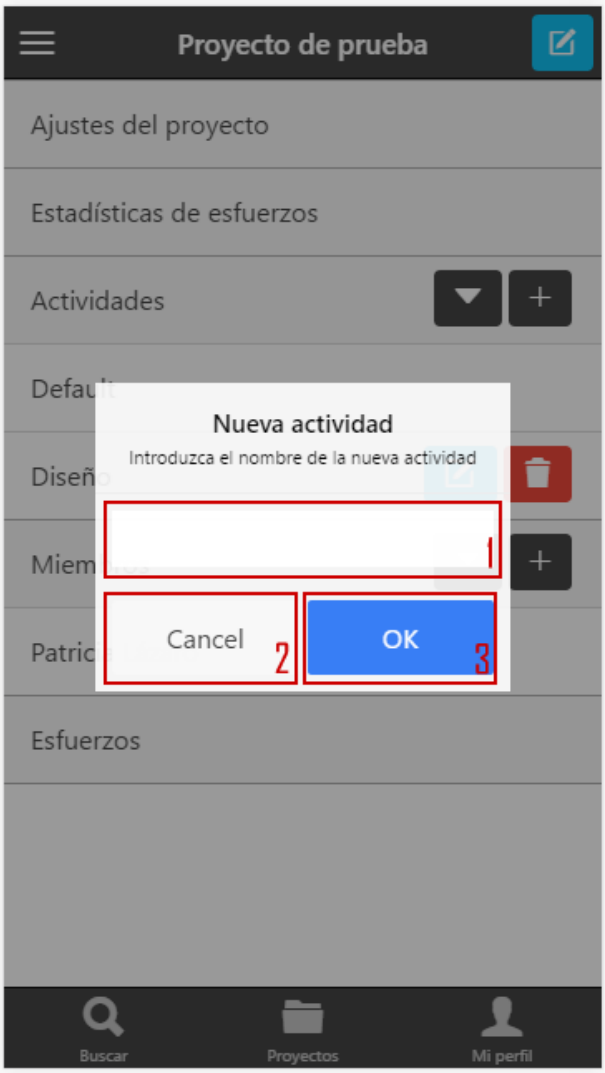


Figura 16: Ventana emergente de creación de una nueva actividad en un proyecto

En la imagen 16 se muestra la ventana emergente de creación de una nueva actividad en un proyecto. Esta ventana emergente es la que aparece cuando el usuario pulsa el botón de añadido de una nueva actividad, ya sea mientras crea el proyecto como cuando lo está visualizando, una vez creado.

El nombre de la nueva actividad se debe introducir en el campo de texto habilitado para ese propósito (1). La ventana emergente cuenta también con un botón para cancelar el agregado de la nueva actividad (2) y un botón para confirmar el añadido de la actividad(3).

Se requiere obligatoriamente de un nombre para la nueva actividad, o no se creará ninguna.

B.7. Ventana emergente de introducción de un nuevo usuario miembro en un proyecto

En la imagen 17 se muestra la ventana emergente de introducción de un nuevo usuario miembro en un proyecto. Esta ventana es la que aparece cuando el usuario pulsa el botón de añadido de un nuevo usuario miembro ya sea mientras crea el proyecto como cuando lo está visualizando, una vez creado.

A través del campo de texto habilitado para tales propósitos (1), se puede buscar a los usuarios del sistema cuyo nombre o email coincida parcial o totalmente con el texto que introduzca el usuario, solo después de haberlo confirmado.

Una vez se realiza una búsqueda, aparece una lista de usuarios que cumplen con los criterios de dicha búsqueda, mostrándose solo su nombre. Si se pulsa en uno de estos usuarios (2) se seleccionará para introducirlos como nuevos usuarios miembros del proyecto cuando se pulse el botón de confirmar.

Solo se puede seleccionar un usuario cada vez, o ninguno.

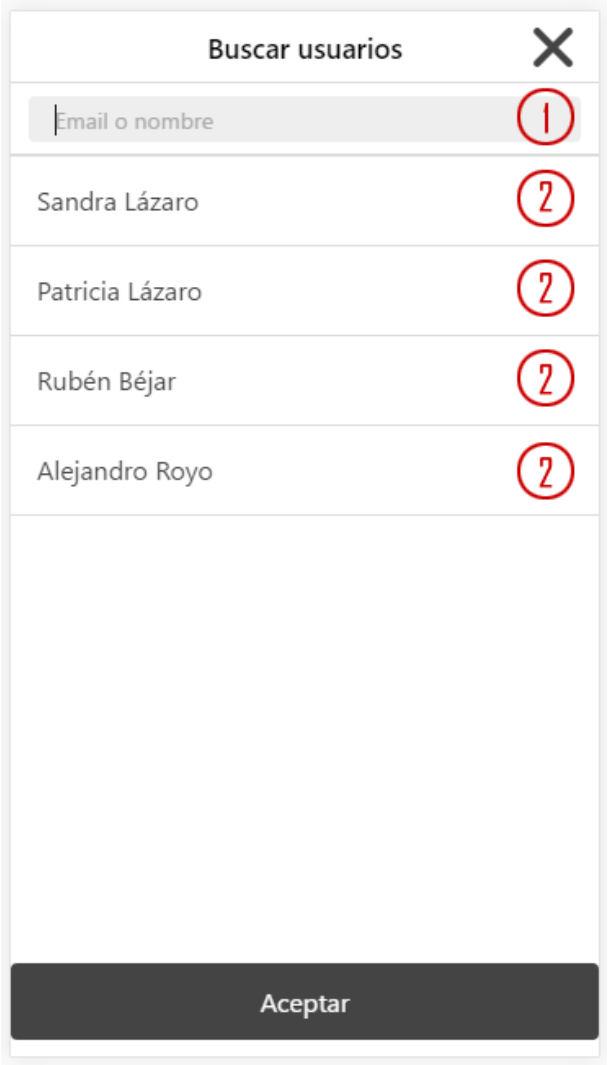


Figura 17: Ventana emergente de introducción de un nuevo usuario miembro en un proyecto

B.8. Pantalla de ajustes de un proyecto

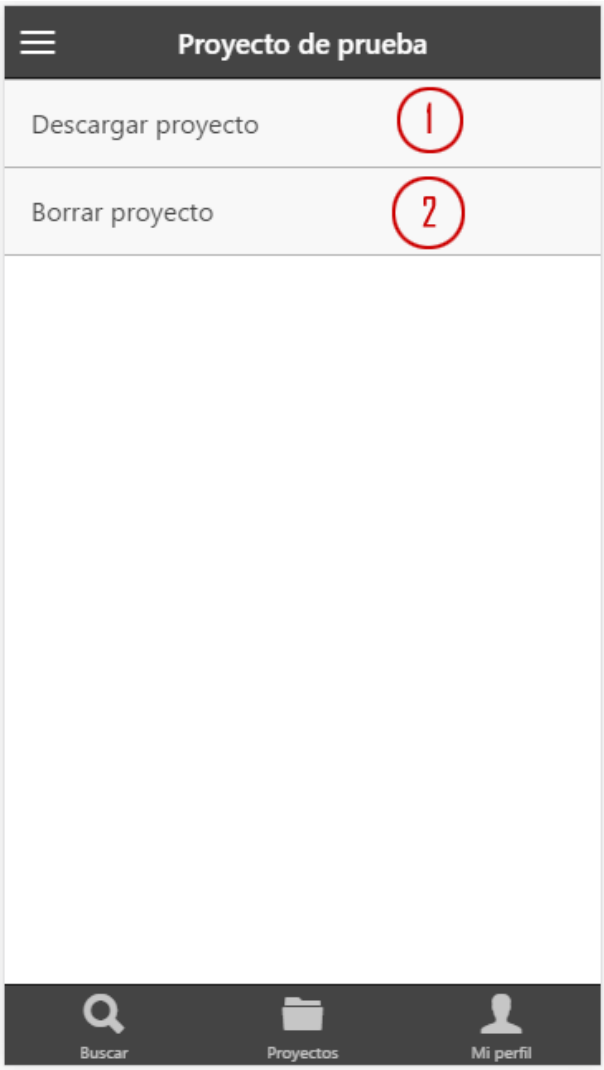


Figura 18: Pantalla de ajustes de un proyecto

En la imagen 18 se muestra la pantalla de ajustes de un proyecto. Los ajustes de un proyecto consisten en descargar los esfuerzos de ese proyecto en un período determinado de tiempo y borrar el proyecto, eliminándolo por completo.

Para descargar los esfuerzos de un proyecto en un determinado período de tiempo, el usuario deberá pulsar el botón de descarga de proyecto (1). Esta acción abrirá una ventana emergente en la que el usuario deberá introducir dos fechas, una que marca a partir de qué día se desea extraer los esfuerzos, y otra que marca hasta que día se desea extraer. Por último, el usuario pulsará sobre el botón de confirmar y se descargará el archivo en formato CSV. En el cliente web, se deberá pulsar sobre el enlace para descargar dicho archivo.

Para borrar el proyecto, se deberá pulsar sobre el botón de borrado de proyecto (2). Aparecerá una ventana emergente para confirmar dicha acción, ya que la eliminación del proyecto conlleva la eliminación de todos los esfuerzos asociados y es permanente. Si el usuario acepta, el proyecto será borrado y el usuario redirigido a la pantalla principal.

B.9. Pantalla de visualización de esfuerzos

En las imágenes 19 y 20 se muestran las dos versiones de la pantalla de visualización de esfuerzos, dependiendo de si se desea agrupar los esfuerzos por fecha o actividad respectivamente.

En ambos casos, se tiene un botón para añadir esfuerzos al proyecto (1), sendos botones para cambiar la forma de visualización (por fecha/por actividad) (2) y una lista de esfuerzos agrupada por fecha o por actividad, dependiendo de la vista.

Cada elemento de la lista agrupada tiene un botón (3) para desplegar los esfuerzos realizados en ese día o en esa actividad. Cada esfuerzo representado, cuenta con un botón (4) para modificar el esfuerzo y un botón para eliminarlo (5).

La visualización de esfuerzos cambia dependiendo de si el usuario es administrador del proyecto o no. Si el usuario es administrador, podrá ver, modificar y eliminar los esfuerzos de los demás miembros participantes del proyecto; si no es administrador, solo podrá ver, modificar y eliminar sus esfuerzos personales.

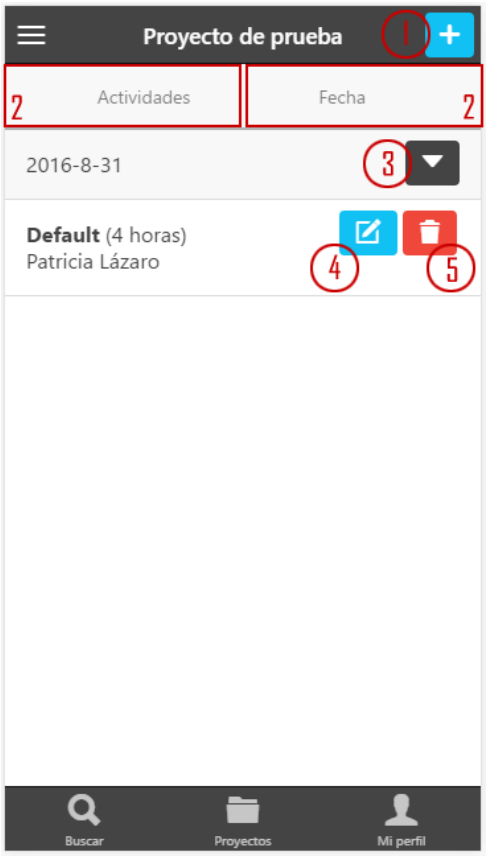


Figura 19: Pantalla de visualización de esfuerzos por fecha

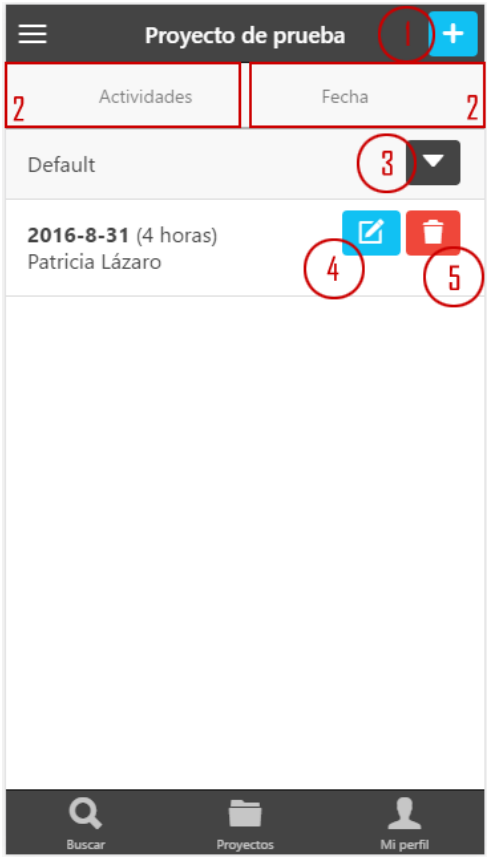


Figura 20: Pantalla de visualización de esfuerzos por actividad

B.10. Ventana emergente de introducción de esfuerzos

En la imagen 21 se muestra la ventana emergente de introducción de esfuerzos. Contiene un campo numérico (2) para la introducción de las horas de esfuerzo, un desplegable (3) para la selección de la actividad en que se han invertido los esfuerzos, y un conjunto de desplegables para seleccionar la fecha de los esfuerzos, junto a un botón para resetear la fecha al día actual (4).

Para cancelar la introducción de esfuerzos se tiene un botón de cancelación (1), y para confirmar la introducción de esfuerzos se tiene un botón de confirmación (5).

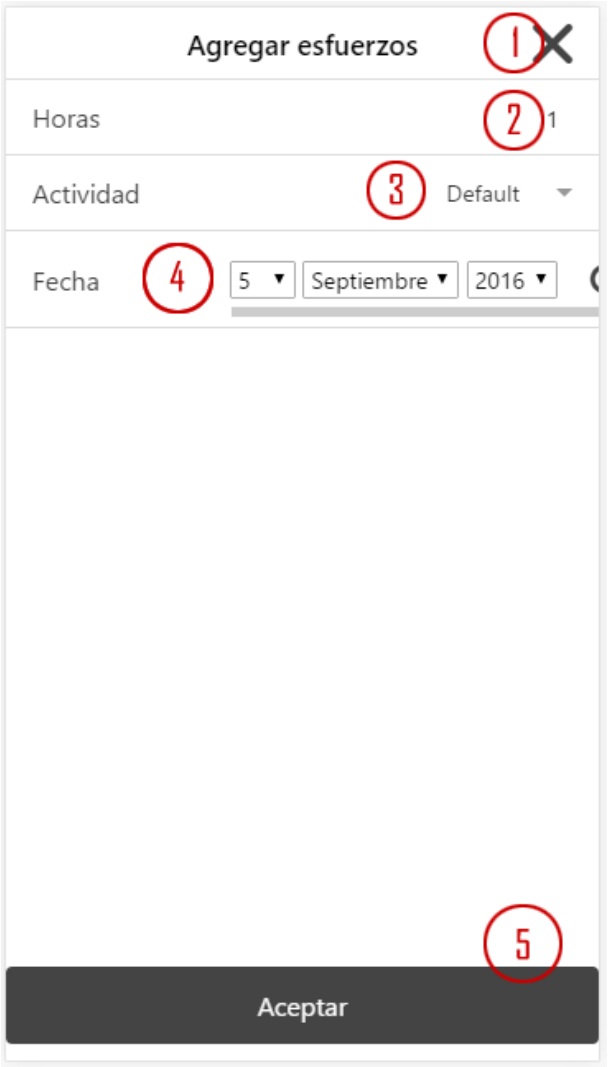


Figura 21: Ventana emergente de introducción de esfuerzos

B.11. Pantalla de visualización condensada de esfuerzos

En la imagen 22 se muestra la pantalla de visualización condensada de esfuerzos. Se muestra en formato tabla, agrupadas por persona y actividad, las horas de esfuerzos dedicadas por dicha persona en dicha actividad a lo largo de todo el proyecto.

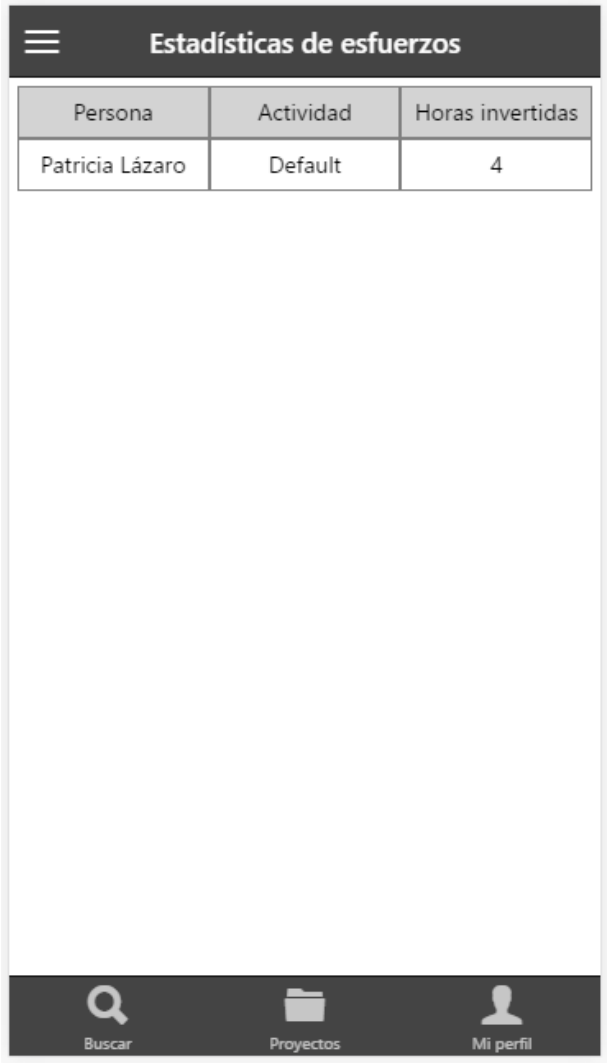


Figura 22: Pantalla de visualización condensada de esfuerzos

B.12. Pantalla de búsqueda de usuarios

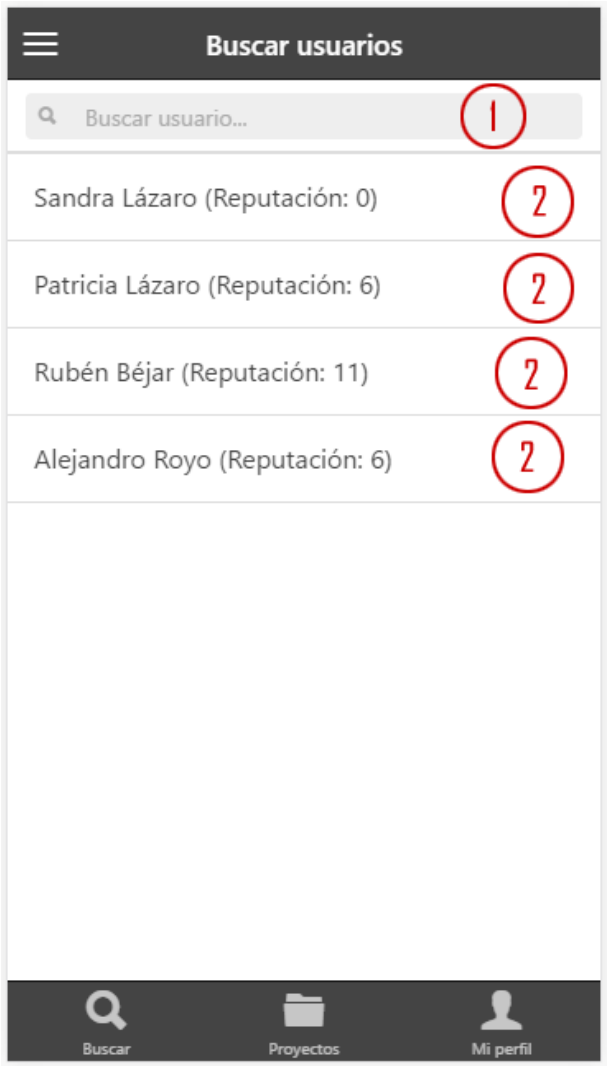


Figura 23: Pantalla de búsqueda de usuarios

En la imagen 23 se muestra la pantalla de búsqueda de usuarios. Contiene una barra de búsqueda de usuarios y una lista de los usuarios encontrados, mostrando su nombre y su reputación.

Para buscar un usuario, se debe introducir un nombre o dirección de correo electrónico en el campo de texto (1). La cadena vacía sirve para buscar a todos los usuarios registrados en el sistema. La aplicación busca a los usuarios cuyo nombre o dirección de correo electrónico contenga el texto introducido por el usuario. Para iniciar la búsqueda, es necesario aceptar el texto introducido.

La lista de usuarios cuyo nombre o email coincide parcial o totalmente con el texto buscado aparece debajo del campo de texto. Presionando cualquiera de los elementos de la lista (2) se visualiza el perfil de dicho usuario.

B.13. Pantalla de perfil

En las imágenes 24 y 25 se muestran las dos versiones de la pantalla de perfil, dependiendo de si el usuario cuyo perfil se visualiza es el que está identificado en la aplicación o no.

En ambos casos, se muestra el nombre del usuario cuyo perfil se está visualizando y la reputación que tiene. Adicionalmente, si el usuario no es el mismo que el identificado en la aplicación, se muestra también la reputación del usuario identificado para facilitar la comparación entre los dos usuarios.

Figura 24: Pantalla de perfil del usuario identificado

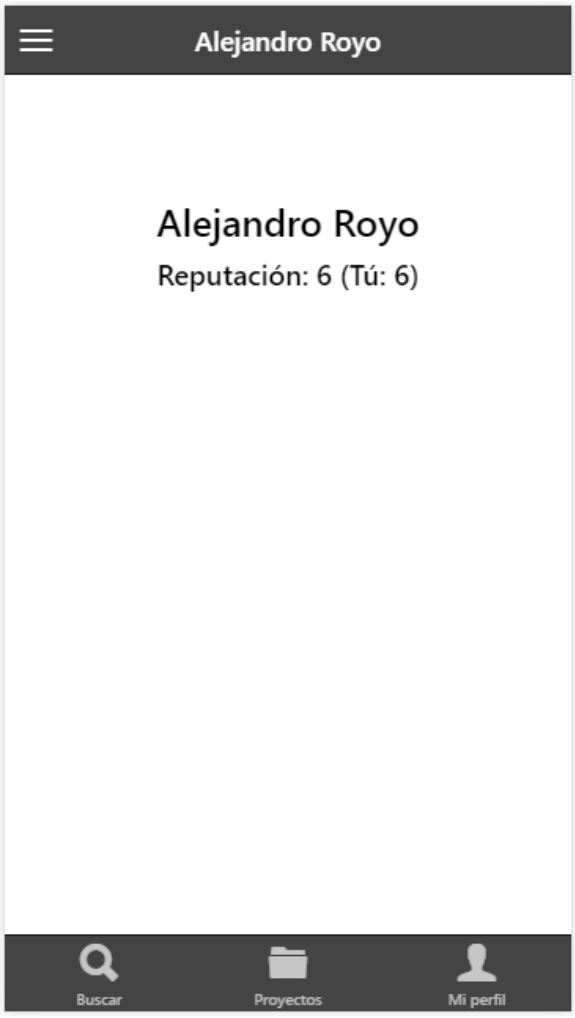


Figura 25: Pantalla de perfil de un usuario distinto al identificado

B.14. Pantalla de visualización de logros

En las imágenes 26 y 27 se muestran las pantallas de visualización de logros de forma general y específicamente de un logro respectivamente.

En la imagen 26 se muestra una lista con los logros obtenidos. Si el usuario pulsa sobre cualquier logro (1), apareceráuna ventana emergente, como se muestra en la imagen 27, mostrando el nombre y descripción de dicho logro, así como un botón para cerrar dicha ventana emergente.

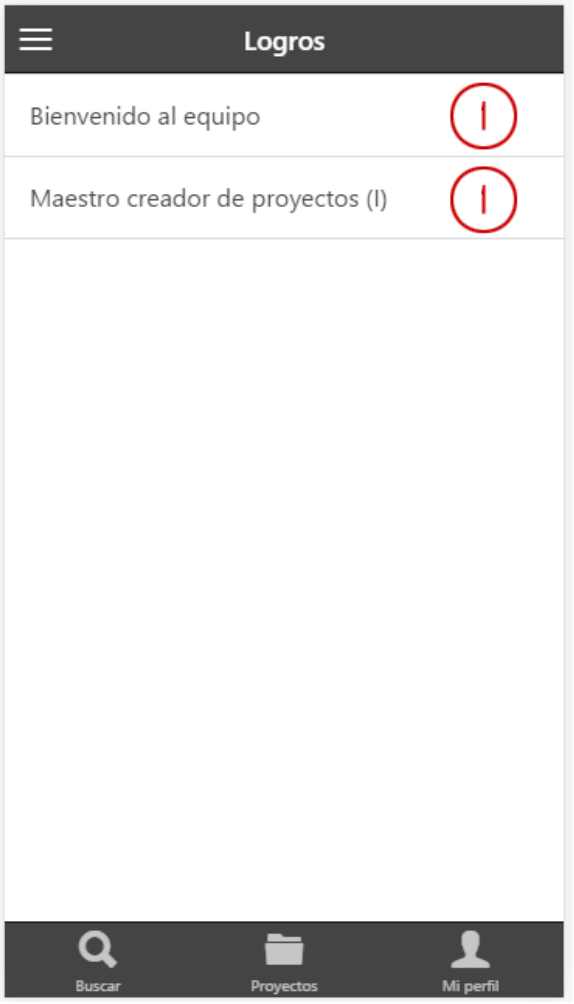


Figura 26: Pantalla de visualización de todos los logros

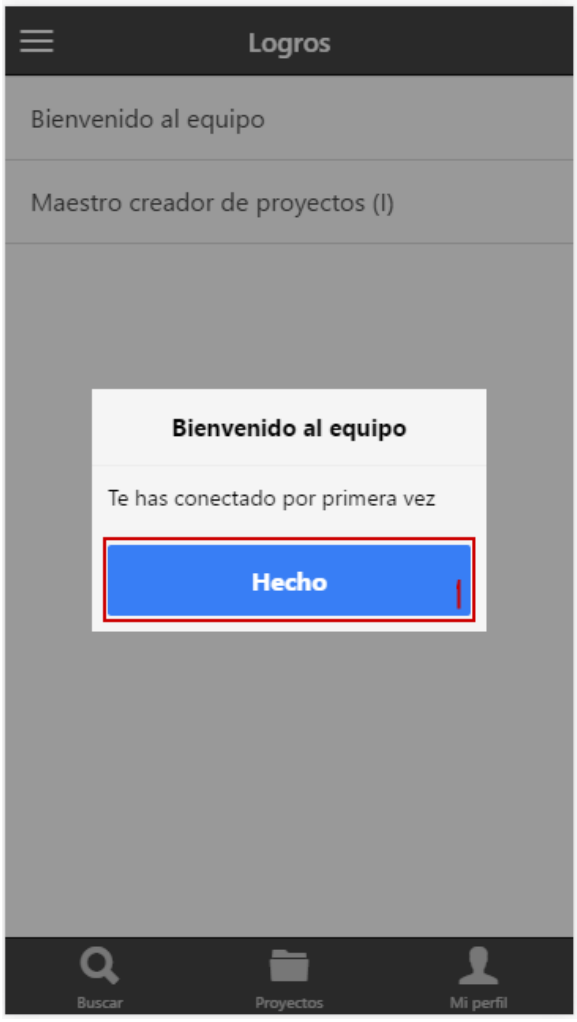


Figura 27: Pantalla de visualización de un logro

C. Exportación de esfuerzos en formato CSV

La aplicación permite exportar los esfuerzos de un proyecto en un determinado período de tiempo en formato CSV. Como se explica en el manual de usuario, para exportar los datos es necesario acceder a la pantalla de ajustes del proyecto, pulsar el botón de descarga de proyecto, introducir las dos fechas que se deseen en la ventana emergente y finalmente confirmar los datos.

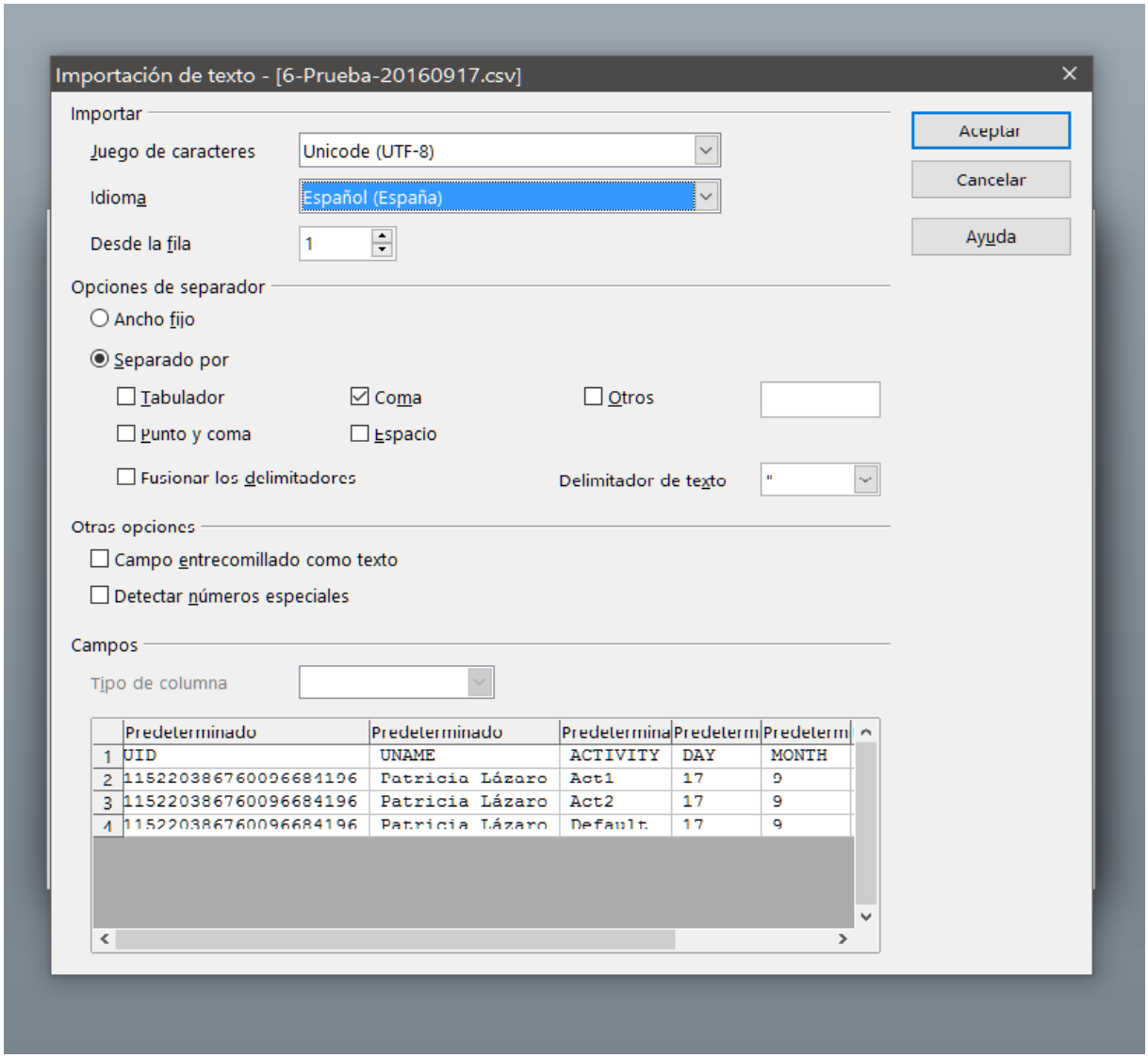


Figura 28: Importación de los datos en OpenOffice

El archivo descargado se encuentra en formato CSV para que pueda visualizarse con cualquier herramienta, desde editores de texto a hojas de cálculo, etcétera. Los datos se encuentran separados por una coma, utilizan el juego de caracteres UTF-8 y la primera línea del fichero siempre se corresponde con las etiquetas de las columnas.

UID	UNAME	ACTIVITY	DAY	MONTH	YEAR	HOURS
115220386760097000000	Patricia Lázaro	Act1	17	9	2016	5
115220386760097000000	Patricia Lázaro	Act2	17	9	2016	12
115220386760097000000	Patricia Lázaro	Default	17	9	2016	2

Figura 29: Visualización de los datos en OpenOffice

La información del proyecto que se descarga del servidor es la que sigue: para cada esfuerzzo, se obtiene el identificador y nombre del usuario que lo realizó, la actividad en la que invirtió dicho esfuerzo, la fecha (día, mes y año) de los esfuerzos y las horas invertidas.

Con estos datos, se pueden realizar diversas tareas de análisis como, por ejemplo, obtener el usuario que más horas ha invertido en el proyecto en total, la media de horas invertidas en una actividad o la variación de horas en total invertidas en el proyecto por días.

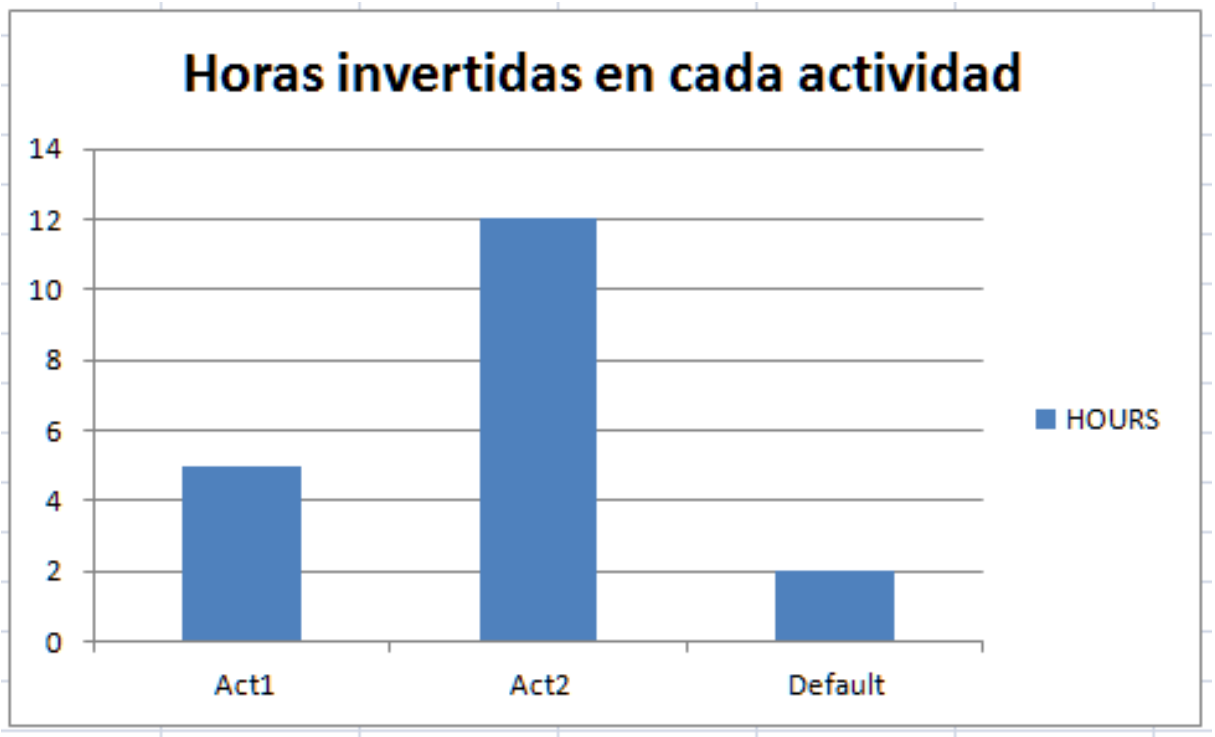


Figura 30: Gráfico de horas por actividad en Microsoft Excel

D. Casos de uso

Cuadro 3: Casos de uso: Identificarse con Google+

Nombre	Identificarse con Google+
Descripción	Se obtienen los datos de la cuenta Google+ del usuario y se concede acceso al sistema.
Actor principal	Usuario en la aplicación móvil o cliente web
Precondición	El usuario ha accedido a la página web o ha iniciado la aplicación. Se encuentra en la primera pantalla, la pantalla de login.
Postcondición	El usuario tiene acceso a sus proyectos, sus logros, etcétera. Se encuentra en la pantalla principal de la aplicación (visualización general de los proyectos en los que el usuario participa).
Escenario de éxito	1. El usuario pulsa el botón de iniciar sesión con Google+. 2. El usuario se autentica contra Google+. 3. Se muestra la pantalla principal de la aplicación, listando los proyectos a los que tiene acceso el usuario.
Extensiones	3.1. Si no se puede conectar con el servidor o se produce un error de autenticación, se muestra un mensaje de error. 3.2. Si es la primera vez que el usuario accede a la aplicación (en cualquiera de las versiones, ya sea Android o cliente web), aparece una notificación con el logro obtenido.

Cuadro 4: Casos de uso: Crear proyecto

Nombre	Crear un proyecto
Descripción	Se crea un proyecto nuevo, siendo el usuario que lo crea su administrador.
Actor principal	Usuario en la aplicación móvil o cliente web.
Precondición	El usuario ha accedido al sistema y se encuentra en la pantalla principal (visualización general de los proyectos en los que participa).

Postcondición	El usuario se encuentra en la pantalla principal. Entre los proyectos que puede visualizar se encuentra el nuevo proyecto que ha creado.
Escenario de éxito	1. El usuario pulsa sobre el botón de *'Nuevo proyecto'*. 2. El usuario rellena el campo de texto de la parte superior de la pantalla, dedicado al nombre del nuevo proyecto. 3. El usuario pulsa sobre el botón de *'Crear proyecto'*. 4. Se muestra la pantalla principal de la aplicación, listando los proyectos que tenía el usuario anteriormente y el nuevo proyecto creado.
Extensiones	2.1. El usuario puede también añadir (caso de uso 5), modificar (caso de uso 6) y eliminar (caso de uso 7) actividades, así como incluir (caso de uso 8) o eliminar (caso de uso 9) miembros participantes del proyecto. 3.1. Si se produce un error al crear el proyecto, se muestra un mensaje de error. 4.1. Si se obtiene uno o varios logros, aparecerán como notificaciones con los logros obtenidos.

Cuadro 5: Casos de uso: Crear actividad

Nombre	Crear una actividad en un proyecto
Descripción	Se crea una nueva actividad en un proyecto. Puede ser una actividad en un proyecto nuevo todavía no creado, o en un proyecto existente en el que el usuario es administrador.
Actor principal	Usuario en la aplicación móvil o cliente web.
Precondición	El usuario se encuentra en la pantalla de creación de un proyecto nuevo o en la pantalla de visualización de un proyecto.
Postcondición	El usuario continúa en la pantalla de creación de un proyecto nuevo o en la pantalla de visualización de un proyecto. El proyecto, creado o no, tiene una nueva actividad.
Escenario de éxito	1. El usuario pulsa sobre el botón de añadir nueva actividad (+).

	2. Se muestra una ventana emergente con un campo de texto para introducir el nombre de la nueva actividad. 3. El usuario escribe el nombre de la nueva actividad en el campo de texto correspodiente de la ventana emergente. 4. El usuario pulsa sobre el botón de *'Confirmar'*. 5. La ventana emergente desaparece y una nueva actividad aparece en el desplegable de actividades.
Extensiones	4.1. Si no hay nada escrito en el campo de texto cuando se pulsa el botón de confirmar, no se creará ninguna actividad. 4.2. Si en vez de pulsar el botón de confirmar pulsa el botón de cancelar, no se creará ninguna actividad. 4.3. Si se produce algún error al crear la actividad (por ejemplo, existe una actividad con el mismo nombre para ese proyecto), se mostrará un mensaje de error y no se creará la actividad. 4.3.1. Si el texto introducido por el usuario es incorrecto (contiene espacios...), se producirá un error y se mostrará un mensaje acorde. No se creará la actividad.

Cuadro 6: Casos de uso: Modificar actividad

Nombre	Modificar una actividad existente en un proyecto
Descripción	Se modifica una actividad existente en un proyecto. Puede ser una actividad en un proyecto nuevo todavía no creado, o en un proyecto existente. La actividad es modificable.
Actor principal	Usuario en aplicación móvil o cliente web.
Precondición	El usuario se encuentra en la pantalla de creación de un proyecto nuevo o en la pantalla de visualización de un proyecto que el usuario administra. Dicho proyecto contiene al menos una actividad modificable por el usuario.
Postcondición	El usuario se encuentra en la pantalla de creación de un proyecto nuevo o en la pantalla de visualización de un proyecto. La actividad modificable elegida ha cambiado su nombre.
Escenario de éxito	1. El usuario pulsa sobre el botón de despliegue de actividades. 2. El usuario pulsa sobre el botón de modificación de una actividad modificable ya creada.

	3. Se muestra una ventana emergente con un campo de texto para introducir el nuevo nombre de la actividad. 4. El usuario escribe el nuevo nombre de la actividad en el campo de texto correspondiente de la ventana emergente. 5. El usuario pulsa sobre el botón de *'Confirmar'*. 6. La ventana emergente desaparece y la actividad modifica su nomnre, cambiándolo al nuevo nombre elegido por el usuario.
Extensiones	4.1. Si no hay nada escrito en el campo de texto cuando se pulsa el botón de confirmar, no se modificará la actividad. 4.2. Si en vez de pulsar el botón de confirmar pulsa el botón de cancelar, no se modificará la actividad. 4.3. Si se produce algún error al modificar la actividad (por ejemplo, existe una actividad con el mismo nombre para ese proyecto), se mostrará un mensaje de error y no se modificará la actividad. 4.3.1. Si el texto introducido por el usuario es incorrecto (contiene espacios...), se producirá un error y se mostrará un mensaje acorde. No se modificará la actividad.

Cuadro 7: Casos de uso: Eliminar actividad

Nombre	Eliminar una actividad existente en un proyecto
Descripción	Se elimina una actividad existente en un proyecto. Puede ser una actividad en un proyecto nuevo todavía no creado, o en un proyecto existente. La actividad es borrable.
Actor principal	Usuario en aplicación móvil o cliente web.
Precondición	El usuario se encuentra en la pantalla de creación de un proyecto nuevo o en la pantalla de visualización de un proyecto que el usuario administra. Dicho proyecto contiene al menos una actividad borrable por el usuario
Postcondición	El usuario se encuentra en la pantalla de creación de un proyecto nuevo o en la pantalla de visualización de un proyecto. La actividad borrable elegida ya no se encuentra entre las actividades del proyecto.
Escenario de éxito	1. El usuario pulsa sobre el botón de despliegue de actividades.

	2. El usuario pulsa sobre el botón de eliminación de una actividad borrable ya creada. 3. La actividad se borra de la lista de actividades.
Extensiones	3.1. Si la actividad no puede borrarse, se muestra un mensaje de error.

Cuadro 8: Casos de uso: Incluir un usuario en un proyecto

Nombre	Incluir un usuario en un proyecto
Descripción	Se incluye un nuevo usuario miembro en un proyecto. Puede ser en un proyecto nuevo todavía no creado, o en un proyecto existente.
Actor principal	Usuario en aplicación móvil o cliente web.
Precondición	El usuario se encuentra en la pantalla de creación de un proyecto nuevo o en la pantalla de visualización de un proyecto que el usuario administra.
Postcondición	El usuario se encuentra en la pantalla de creación de un proyecto nuevo o en la pantalla de visualización de un proyecto. Dicho proyecto tiene un nuevo usuario participante.
Escenario de éxito	1. El usuario pulsa sobre el botón de añadir miembro (*'+'*). 2. Se muestra una ventana emergente con un campo de texto para buscar usuarios por nombre o email. 3. El usuario escribe el nombre o email (completo o parcial) del usuario que desea agregar al proyecto, y confirma dicho texto. 4. Se muestra (debajo del campo de texto de la ventana emergente) una lista con los usuarios encontrados. 5. El usuario pulsa sobre uno de los usuarios para seleccionarlo. 6. El usuario pulsa sobre el botón de *'Aceptar'*. 7. La ventana emergente desaparece y el usuario es agregado al proyecto. El usuario aparece en la lista de usuarios participantes del proyecto.
Extensiones	3.1. Si se produce un error en la conexión con el servidor, aparece un mensaje de error y no se buscan los usuarios.

	7.1. Si se produce un error en la conexión con el servidor, aparece un mensaje de error y no se agrega el usuario a la lista de usuarios participantes del proyecto.

Cuadro 9: Casos de uso: Eliminar un usuario de un proyecto

Nombre	Eliminar un usuario de un proyecto
Descripción	Se elimina un usuario participante de un proyecto del proyecto. Puede ser en un proyecto nuevo todavía no creado o en un proyecto existente.
Actor principal	Usuario en aplicación móvil o cliente web.
Precondición	El usuario se encuentra en la pantalla de creación de un proyecto nuevo o en la pantalla de visualización de un proyecto que el usuario administra. Existe un usuario eliminable del proyecto.
Postcondición	El usuario se encuentra en la pantalla de creación de un proyecto nuevo o en la pantalla de visualización de un proyecto. Dicho proyecto ya no contiene al usuario eliminado.
Escenario de éxito	1. El usuario pulsa sobre el botón de despliegue de usuarios. 2. El usuario pulsa sobre el botón de eliminación de un usuario eliminable ya incluido en el proyecto. 3. El usuario se elimina de la lista de usuarios del proyecto.
Extensiones	3.1. Si el usuario no puede eliminarse, se muestra un mensaje de error.

Cuadro 10: Casos de uso: Descargar esfuerzos de un proyecto

Nombre	Descargar esfuerzos de un proyecto
Descripción	Se descargan los esfuerzos de un proyecto en un período de tiempo determinado en formato CSV.
Actor principal	Usuario en aplicación móvil o cliente web.
Precondición	El usuario se encuentra en la pantalla de ajustes de un proyecto ya creado que él administra.

Postcondición	El usuario se encuentra en la pantalla de ajustes de dicho proyecto. En su máquina local o almacenamiento interno se encuentra un fichero CSV con los esfuerzos del proyecto de un período de tiempo almacenados.
Escenario de éxito	1. El usuario pulsa sobre el botón de *'Descargar proyecto'*. 2. Se muestra una ventana emergente con dos campos de fechas, una de inicio y una de fin. 3. El usuario elige la fecha de inicio y fin de los esfuerzos que desea descargar. 4. El usuario pulsa sobre el botón de *'Aceptar'*. 5. La ventana emergente desaparece. 6.1.1. Si el usuario utiliza el cliente web, aparece una ventana emergente con un enlace. 6.1.2. El usuario pulsa sobre el enlace y descarga los esfuerzos en formato CSV. 6.2. Si el usuario utiliza la aplicación móvil, aparece una ventana emergente informándole de que sus esfuerzos han sido guardados en el almacenamiento interno del dispositivo. 7. El usuario pulsa sobre el botón de *'Confirmar'*. 8. La ventana emergente desaparece.
Extensiones	4.1. Si se produce un error al obtener los esfuerzos del servidor, se muestra un mensaje de error. 6.2.1. Si no se ha podido guardar el archivo en el almacenamiento interno del dispositivo móvil, se muestra un mensaje de error.

Cuadro 11: Casos de uso: Ver esfuerzos de un proyecto

Nombre	Ver esfuerzos de un proyecto
Descripción	Se visualizan los esfuerzos de un proyecto.
Actor principal	Usuario en aplicación móvil o cliente web.

Precondición	El usuario se encuentra en la pantalla de visualización de un proyecto.
Postcondición	El usuario se encuentra en la pantalla de visualización de esfuerzos de ese proyecto.
Escenario de éxito	1. El usuario pulsa sobre el botón *'Esfuerzos'*. 2. Se muestra la pantalla de visualización de esfuerzos.
Extensiones	2.1. Si no se pueden recuperar los esfuerzos, no aparece ningún esfuerzo. 2.2. Por defecto se muestran los esfuerzos agrupados por actividad. Para agruparlos por actividad, el usuario pulsa la pestaña (tab) superior *'Actividades'*. 2.3. Para agrupar los esfuerzos por fecha, el usuario pulsa la pestaña (tab) superior *'Fecha'*.

Cuadro 12: Casos de uso: Añadir esfuerzos a un proyecto

Nombre	Añadir esfuerzos a un proyecto
Descripción	Se añaden esfuerzos a un proyecto. Incluye determinar la cantidad de horas de esfuerzo, el día de los esfuerzos y actividad a la que se ha dedicado el tiempo.
Actor principal	Usuario en la aplicación móvil o cliente web.
Precondición	El usuario se encuentra en la pantalla de visualización de esfuerzos.
Postcondición	El usuario se encuentra en la pantalla de visualización de esfuerzos. Se han añadido esfuerzos al proyecto.
Escenario de éxito	1. El usuario pulsa sobre el botón de añadir esfuerzos (*'+'*) en la barra superior de la pantalla. 2. Se muestra una ventana emergente con un campo numérico para las horas de esfuerzo, un desplegable para la actividad y un grupo de desplegables para la fecha. 3. El usuario introduce un número de horas de esfuerzo en el campo numérico habilitado para tales efectos.

	4. El usuario elige del desplegable la actividad a la que ha dedicado sus esfuerzos. 5. El usuario elige del grupo de desplegables la fecha en la que ha realizado sus esfuerzos. 6. El usuario pulsa sobre el botón de *'Aceptar'*. 7. La ventana emergente desaparece. Se introducen los esfuerzos y se muestran en la pantalla de visualización de esfuerzos.
Extensiones	3.1. Si el usuario introduce un valor no numérico, o un valor numérico no positivo, los esfuerzos no se introducen y aparece un mensaje de error. 7.1. Si no se han podido introducir los esfuerzos en el proyecto, aparece un mensaje de error. 7.2. Si se han obtenido logros por añadir esfuerzos, se muestran en una notificación.

Cuadro 13: Casos de uso: Modificar esfuerzos de un proyecto

Nombre	Modificar esfuerzos de un proyecto
Descripción	Se modifican los esfuerzos de un proyecto.
Actor principal	Usuario en cliente web o aplicación móvil.
Precondición	El usuario se encuentra en la pantalla de visualización de esfuerzos. Existe al menos un esfuerzo.
Postcondición	El usuario se encuentra en la pantalla de visualización de esfuerzos. Un esfuerzo ha sido modificado.
Escenario de éxito	1. El usuario despliega la categoría donde se encuentra el esfuerzo que desea modificar. 2. El usuario pulsa sobre el boton de modificación del esfuerzo que desea modificar. 3. Se muestra una ventana emergente con las horas (campo numérico), actividad (desplegable) y fecha (conjunto de desplegables) del esfuerzo a modificar. 4. El usuario cambia el número de horas del esfuerzo en el campo numérico.

	5. El usuario cambia la actividad en el desplegable. 6. El usuario cambia la fecha en el conjunto de desplegables. 7. El usuario pulsa sobre el botón de *'Aceptar'*. 8. La ventana emergente desaparece. Se modifican los esfuerzos y se muestran en la pantalla de visualización de esfuerzos.
Extensiones	4.1. Si el usuario introduce un valor no numérico, o un valor numérico no positivo, los esfuerzos no se modifican y aparece un mensaje de error. 8.1. Si no se han podido introducir los esfuerzos en el proyecto, aparece un mensaje de error.

Cuadro 14: Casos de uso: Eliminar esfuerzos de un proyecto

Nombre	Eliminar esfuerzos de un proyecto
Descripción	Se eliminan esfuerzos de un proyecto.
Actor principal	Usuario en cliente web o aplicación móvil.
Precondición	El usuario se encuentra en la pantalla de visualización de esfuerzos. Existe al menos un esfuerzo.
Postcondición	El usuario se encuentra en la pantalla de visualización de esfuerzos. Hay un esfuerzo menos que visualizar.
Escenario de éxito	1. El usuario despliega la categoría donde se encuentra el esfuerzo que desea eliminar. 2. El usuario pulsa sobre el botón de eliminar del esfuerzo que desea borrar. 3. La lista de esfuerzos se actualiza y se borra el esfuerzo.
Extensiones	3.1. Si no se puede borrar el esfuerzo aparece un mensaje de error.

Cuadro 15: Casos de uso: Eliminar un proyecto

Nombre	Eliminar un proyecto

Descripción	Se elimina permanentemente un proyecto que el usuario administra.
Actor principal	Usuario en aplicación móvil o cliente web.
Precondición	El usuario se encuentra en la pantalla de ajustes del proyecto (es administrador de dicho proyecto).
Postcondición	El usuario se encuentra en la pantalla principal (pantalla de visualización general de proyectos). El proyecto eliminado no aparece en la lista de proyectos en los que participa el usuario.
Escenario de éxito	1. El usuario pulsa sobre el botón *'Borrar proyecto'*. 2. Aparece una ventana emergente pidiendo confirmación para realizar la acción. La ventana tiene un botón para cancelar y un botón para confirmar el borrado del proyecto. 3. El usuario pulsa sobre el botón de *'Confirmar'*. 4. El proyecto se elimina. Aparece la pantalla principal (visualización general de proyectos). El proyecto eliminado no aparece en la lista de proyectos en los que participa el usuario.
Extensiones	3.1. Si el usuario pulsa sobre el botón *'Cancelar'*, se cierra la ventana emergente sin que se haya borrado el proyecto. 4.1. Si el proyecto no ha podido ser eliminado, aparece un mensaje de error.

Cuadro 16: Casos de uso: Ver logros personales

Nombre	Ver logros personales
Descripción	Se visualizan los logros que ha obtenido el usuario.
Actor principal	Usuario en cliente web o aplicación móvil.
Precondición	El usuario se encuentra en cualquier pantalla de la aplicación (salvo la pantalla de login).
Postcondición	El usuario se encuentra en la pantalla de visualización de logros.
Escenario de éxito	1. El usuario pulsa sobre el botón superior izquierdo de la barra de acción para abrir el menú lateral. 2. Se abre el menú lateral.

	3. El usuario pulsa sobre el ítem *'Logros'*. 4. Se abre la pantalla de visualización de logros.
Extensiones	2.1. Si se produce un error al recuperar la información de los proyectos en los que participa el usuario, no aparecen las entradas asociadas a los esfuerzos de dichos proyectos. 4.1. Si se produce un error al recuperar los logros del usuario, aparece un mensaje de error. 4.2. Si se pulsa en un ítem de la lista de logros, aparece una ventana emergente con información detallada del logro.

Cuadro 17: Casos de uso: Buscar usuarios

Nombre	Buscar usuarios en la aplicación
Descripción	Se buscan usuarios que también utilicen la aplicación, con posibilidad de centrar la búsqueda en los usuarios cuyo nombre o email coincida con un texto introducido por el usuario.
Actor principal	Usuario en cliente web o aplicación móvil.
Precondición	El usuario se encuentra en cualquier pantalla de la aplicación (salvo la pantalla de login).
Postcondición	El usuario se encuentra en la pantalla de búsqueda de usuarios.
Escenario de éxito	1.1. El usuario pulsa el botón de *'Buscar'* de la barra inferior de la aplicación. 1.2.1. El usuario pulsa sobre el botón superior izquierdo de la barra de acción para abrir el menú lateral. 1.2.2. Se abre el menú lateral. 1.2.3. El usuario pulsa sobre el ítem *'Buscar...'*. 2. Se abre la pantalla de búsqueda de usuarios. La pantalla contiene un campo de texto en la parte superior y una lista de todos los usuarios de la aplicación. 3. El usuario introduce el nombre o email (completo o parcial) del usuario que desea buscar y lo confirma.

	4. La pantalla carga una lista de usuarios debajo del campo de texto. Los usuarios mostrados coinciden con los criterios de búsqueda introducidos por el usuario.
Extensiones	1.2.2.1. Si se produce un error al recuperar la información de los proyectos en los que participa el usuario, no aparecen las entradas asociadas a los esfuerzos de dichos proyectos. 2.1. Si se produce un error al obtener la lista de todos los usuarios de la aplicación, aparece un mensaje de error. 4.1. Se si produce un error al realizar la búsqueda de usuarios, aparece un mensaje de error.

Cuadro 18: Casos de uso: Ver el perfil de un usuario (el usuario identificado)

Nombre	Ver el perfil del usuario identificado en la aplicación
Descripción	Se visualiza el perfil de un usuario.
Actor principal	Usuario en el cliente web o aplicación móvil.
Precondición	El usuario se encuentra en cualquier pantalla de la aplicación (salvo la pantalla de login).
Postcondición	El usuario se encuentra en la pantalla de visualización de su perfil.
Escenario de éxito	1. El usuario pulsa sobre el botón inferior derecho de la barra inferior de la aplicación, *'Mi perfil'*. 2. Se abre la pantalla de visualización del perfil de un usuario con la información del usuario identificado.
Extensiones	2.1. Si se produce un error al obtener la información del usuario, aparece un mensaje de error.

Cuadro 19: Casos de uso: Ver el perfil de un usuario

Nombre	Ver el perfil de un usuario
Descripción	Se visualiza el perfil de un usuario.
Actor principal	Usuario en aplicación móvil o cliente web.
Precondición	El usuario se encuentra en la pantalla de búsqueda de usuarios.

Postcondición	El usuario se encuentra en la pantalla de visualización del perfil de un usuario.
Escenario de éxito	1. El usuario introduce el nombre o email (completo o parcial) del usuario que desea buscar y lo confirma. 2. La pantalla carga una lista de usuarios debajo del campo de texto. Los usuarios mostrados coinciden con los criterios de búsqueda introducidos por el usuario. 3. El usuario pulsa sobre un ítem de la lista de usuarios. 4. Se abre la pantalla de visualización del perfil de un usuario con la información del usuario seleccionado.
Extensiones	2.1. Si se produce un error al obtener la lista de todos los usuarios de la aplicación, aparece un mensaje de error. 4.1. Si se produce un error al obtener la información del usuario, aparece un mensaje de error.

E. Análisis de riesgos

E.1. Pérdida de datos (código, documentación...)

- **Riesgo**: el código del cliente o servidor y/o la documentación del proyecto se pierden parcial o totalmente (por borrado accidental, etcétera).

  Probabilidad de que se produzca: 2/5

  Impacto si se produce: 4/5 (se pierde el trabajo realizado)

- **Mitigación**: mantener bajo control de versiones (Dropbox, Bitbucket...) los documentos y el código del proyecto.

E.2. Actualización de la tecnología utilizada

- **Riesgo**: la tecnología utilizada (Ionic, plugins de Ionic utilizados) recibe una actualización por parte de los desarrolladores, haciendo que los componentes no funcionen.

  Probabilidad de que se produzca: 1/5

  Impacto si se produce: 4/5 (pérdida de soporte de la versión antigua, problemas de integración con la versión nueva de la tecnología)

- **Mitigación**: ceñirse a una versión estable de las tecnologías utilizadas. Alternativamente, actualizar solo cuando sea necesario e imprescindible y estudiar a fondo los cambios entre versiones para poder arreglar eficientemente los fallos que puedan aparecer como resultado de la actualización de componentes.

E.3. Desconocimiento de la tecnología a utilizar

- **Riesgo**: se desconoce en gran medida la tecnología a utilizar, no se ha trabajado en demasía con ella y/o no se tiene mucha experiencia.

  Probabilidad de que se produzca: 5/5

  Impacto si se produce: 1/5 (se pierde más tiempo en el desarrollo por no conocer el potencial y las limitaciones de la tecnología que se está utilizando)

- **Mitigación**: tener este riesgo en cuenta en la fase de planificación y reservar más tiempo para el aprendizaje de las tecnologías desconocidas en las primeras iteraciones del desarrollo del proyecto.

E.4. Retrasos en el desarrollo del proyecto

- **Riesgo**: el proyecto puede retrasarse y no llegar a entregarse en la fecha en que se había planeado entregar.

  Probabilidad de que se produzca: 2/5

  Impacto si se produce: 3/5

- **Mitigación**: entregar en otra fecha.