Departamento de Informática

PROYECTO FIN DE CARRERA

INGENIERÍA INFORMÁTICA

DISEÑO E IMPLEMENTACIÓN DE UN CLIENTE DE UNA RED SOCIAL BASADO EN iOS

Autor: Roberto Miranda González

Tutor: Francisco Javier García Blas

Leganés, diciembre de 2011

Título: DISEÑO E IMPLEMENTACIÓN DE UN CLIENTE DE UNA RED SOCIAL BASADO EN iOS

Autor: ROBERTO MIRANDA GONZÁLEZ

Director: FRANCISCO JAVIER GARCÍA BLAS

EL TRIBUNAL

Presidente: _____

Vocal: _____

Secretario: _____

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día ___ de _____ de 20___ en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO PRESIDENTE

Agradecimientos

Todo lo que debemos hacer es avanzar… nunca detenernos ante los problemas sino afrontarlos y superarlos. Esto es algo que he aprendido y me he repetido durante toda la carrera y es la idea que deseo seguir el resto de mi vida.

Mucho ha llovido desde que decidí comenzar la carrera de ingeniería informática. Ha sido un largo trayecto, ha tenido momentos de tristeza, de decepción, pero por supuesto también momentos de alegría y de orgullo.

Por todos esos momentos en los que parecía que todos se derrumbaba y entonces aparecía una persona que te alzara de nuevo, por todas las cosas que no he sabido hacer y una persona me ha enseñado, por todos los ánimos de apoyo que alguien me ha dado cuando los necesitaba, yo le digo… gracias.

Gracias a mi madre Aracely por mantenerse siempre tan atenta y preocupada por mí. Gracias a mi padre Roberto por estar siempre ahí discretamente vigilando que todo fuera bien. Gracias a mi hermano Francisco por comprenderme incluso estando tan lejos. Y gracias a Franzi por ser una de esas personas que hace que la vida sea maravillosa incluso en tiempos de angustia.

Gracias a Mario y a Raúl por ser unos excelentes compañeros y amigos. Gracias a Cristian y a Chema por dar junto a mí mis primeros pasos en Madrid. Gracias a Aitor, Lalla, Martín, Jesús, José y Lillo por compartir tantas experiencias y almuerzos en la 'Ñ'. Recuerdo a Fombellida, Víctor, Quique y todos los demás compañeros que han vivido conmigo esta etapa de mi vida en Madrid.

Gracias también a mi tutor Francisco Javier, quien me ha ayudado y dado consejo tanto a nivel académico como personal. Gracias a todos.

Resumen

A lo largo de los últimos años el uso de los dispositivos móviles con pantalla táctil se ha extendido enormemente. Al mismo tiempo, el uso de redes sociales se ha disparado hasta unos límites inimaginables. La combinación de dispositivos móviles con servicios de datos móviles con tecnologías como Edge o 3G permite que se abra un amplio abanico de posibilidades de acceso a redes sociales a partir de dispositivos móviles.

De manera concreta, de todos los smartphones utilizados, los dispositivos iPhone de Apple son de los más vendidos. Como dato, decir que a finales de 2010, iPhone tenía un share del 17,35% del mercado de Smartphones, es decir, casi 1 de cada 5 smarthpones era un iPhone.

En este proyecto se detalla el análisis, el diseño y la implementación de un cliente móvil para iOS (el sistema operativo de iPhone y iPad de Apple) para una red social basada en el intercambio de valoraciones. La red social se caracteriza por permitir realizar una valoración sobre una determinada imagen. Dicha imagen puede ir acompañada de una localización GPS y de un texto descriptivo. Toda la información de una valoración debe ser enviada a un servidor donde sea almacenada. El objetivo es que cualquier persona pueda utilizar la aplicación para compartir sus fotos (con su respectiva valoración) con otros usuarios y poder ver las valoraciones de otras personas.

Palabras clave: red social, valoración, iPhone, iOS.

Abstract

Over recent years the use of touchscreen mobile devices has expanded enormously. At the same time, the use of social networks has soared to unimaginable limits. The combination of mobile devices with mobile data services such as Edge or 3G technologies allows users open a wide range of possibilities of accessing social networks from mobile devices.

In concrete terms, all smartphones used, Apple iPhone devices are best sellers. It is worth saying that in late 2010, the iPhone had a 17.35% share of the smartphone market; almost 1 in 5 smartphones was an iPhone.

This Project details the analysis, design, and implementation of a mobile client for iOS (the operating system of Apple iPhone and iPad) for a social network based on the rating exchange. The social network is characterized by allowing performing an assessment of a particular image. This image can be attached by a GPS location and a descriptive text. All assessment information should be sent to a server in where it is stored. The goal is that anyone can use the application to share your pictures (with their respective rating) with other users and see the reviews of others.

Keywords: social network, rating, iPhone, iOS.

Índice general

Índice de tablas

Índice de ilustraciones

Capítulo 1

Introducción y objetivos

Este capítulo introductorio iniciará al lector en los objetivos que se pretenden conseguir con este proyecto. Se detallarán las fases de desarrollo del proyecto así como los medios con los que se ha contado para su realización. En último lugar se dará una descripción de la estructura de este documento, aportándose una breve descripción de cada capítulo.

1.1 Introducción

Los teléfonos inteligentes son hoy en día bien conocidos por todos. Son cada vez más los que dan el salto a la tecnología móvil y abandonan su antiguo y obsoleto dispositivo por un teléfono inteligente. Las ventajas que supone disponer de un "ordenador de bolsillo" son claras para muchos. ¿Por qué nos fascina tanto la tecnología móvil? Probablemente sea debido a las innumerables aplicaciones que se le pueden dar a estos dispositivos:,que van desde un simple creador de la lista de la compra hasta un simulador de vuelo online.

Por otra parte, la presencia actual de las redes sociales es innegable. Los datos son abrumadores, sólo la red social Twitter tiene 175 millones de usuarios registrados [1]. Estos datos muestran la tendencia de la sociedad humana a estar comunicada en cualquier lugar y en todo momento.

La idea de esta red social lleva inmediatamente a pensar que debería ser diseñada sobretodo para dispositivos móviles, ya que con ellos se puede tomar fotos de una manera muy sencilla. Debido al auge de los teléfonos inteligentes, un gran número de personas tendría acceso a la red social.

Una red social de este tipo proporcionaría un servicio a muchas personas que buscan evaluar opiniones para comprar o escoger un determinado producto, lo cual implica una gran motivación para su desarrollo.

Este sistema se ha pensado por tres compañeros y el objetivo es dividirlo en tres proyectos independientes pero ligados en cierto modo. Un proyecto sería el desarrollo de la parte servidor. Un segundo proyecto sería el desarrollo para un cliente para el sistema operativo Android. Y el último proyecto, que corresponde al presente documento, sería el de desarrollar un cliente para el sistema operativo iOS de Apple. La creación de un sistema de esta magnitud con vistas a una posible entrada a mercado supone una excelente motivación para su desarrollo.

Motivado por los argumentos presentados anteriormente, el objetivo de este proyecto es el de crear una red social para valorar fotos de elementos que podemos encontrar en nuestra vida cotidiana. El funcionamiento básico es simple: se toma una foto a un elemento, se describe y se le da una puntuación. De manera que unas personas puedan ver los que otras han publicado y viceversa. El hecho de que podamos valorar objetos de manera global abre grandes posibilidades: podríamos saber si un modelo de coche es mejor que otro leyendo las críticas de los usuarios., podríamos tener una idea de a qué restaurante ir conociendo los comentarios del resto de usuarios. El sistema es autogenerado, es decir, son los propios usuarios quienes construyen la gran base de datos.

Cabe destacar que debido a que el uso de dispositivos móviles está en un momento culminante, el hecho de aprender a desarrollar una aplicación para un dispositivo móvil iPhone, que se ejecuta sobre el sistema iOS, es otra fuente de motivación.

1.2 Principales objetivos

El objetivo fundamental del proyecto es el de diseñar e implementar un cliente de una red social de valoración de imágenes sobre la plataforma iOS, que permita añadir valoraciones sobre una imagen tomada desde el dispositivo iPhone y agregar comentarios, localización geográfica y precio a dicha imagen. Se debe permitir subir esta valoración a un servidor remoto de manera que se comparta con otros usuarios de la red social.

En base dicho objetivo principal, se proponen los siguientes objetivos segundarios:

- Aprendizaje del lenguaje, API y entorno de programación de iOS.

Es necesario familiarizarse con el lenguaje Objective-C, un superconjunto de C, orientado a objetos. Adicionalmente para poder trabajar sobre iOS es necesario comprender su API de desarrollo.

Además es preciso aprender a trabajar usando la herramienta de desarrollo Xcode, ya que es la única soportada oficialmente por Apple que está disponible para desarrollo de iOS.

- Diseño de interfaces para la aplicación

Se debe diseñar cómo serán las pantallas de la aplicación, de manera que el uso de la misma sea sencillo e intuitivo. Se debe considerar que la información mostrada en cada pantalla debe ser coherente. Además se debe tener en cuenta que la interfaz debe inspirar seguridad en el usuario ya que en algunos casos como en el registro, se proporcionan datos personales.

Se deberá diseñar la interfaz minimizando el tiempo que el usuario tarda navegando por la interfaz para realizar una tarea.

- Creación de un sistema de registro de usuarios

El registro debe ser sencillo a la vez que seguro para el usuario. Se debe hacer uso de CAPTCHA para evitar la automatización de registros por parte de usuarios malintencionados.

- Creación de un sistema de usuarios, seguidos y seguidores.

Se debe almacenar y mostrar la información de cada usuario. Los usuarios pueden tener seguidos y seguidores. Los usuarios deben poder definir quién puede acceder a su perfil, pudiendo este ser público, privado o solo seguidores.

- Creación de un sistema de valoraciones.

Se debe permitir a un usuario subir una valoración, es decir, una imagen con un comentario y un número que indique la puntuación que se le otorga. Una valoración podrá adicionalmente contener un precio.

- Diseñar un sistema de búsqueda de artículos.

El sistema deberá permitir realizar búsquedas por medio de etiquetas o nombres del artículo.

- Etiquetado de artículos

Categorizar los artículos según una determinada etiqueta de manera que se pueda acceder a artículos similares a partir de un artículo dado.

- Creación de un ranking de artículos

Permitir al usuario acceder a un ranking de los artículos mejor valorados, pudiendo acceder a rankings para distintas categorías.

- Implementar una memoria caché local para almacenar datos de usuario.

Usar un caché local para disponer localmente de aquella información ya recibida y así evitar hacer peticiones al servidor de información que no ha sido actualizada. Se deberá utilizar una política de reemplazo para gestionar la saturación de la información local.

1.3 Medios empleados

Para la realización de este proyecto se ha dispuesto de una serie de herramientas hardware como software. El hardware necesario ha incluido:

- Ordenador portátil Apple Macbook Pro 13”
- Teléfono móvil Apple iPhone 3G 8GB

En cuanto a las herramientas software se ha usado:

- Mac OS X 10.6.8 (el S.O. del Macbook)
- iOS 4.2 (S.O. del iPhone)
- Xcode 4.0.2
- Inkscape 0.48
- Adobe Photoshop CS5

Se debe mencionar que ha sido utilizado un servidor web en todo momento donde se ejecuta la aplicación servidora de la red social. Pero queda excluido de este documento como material empleado puesto que formaría parte de otro proyecto.

1.4 Esquema de la memoria

Para facilitar la lectura de la memoria, se incluye a continuación un breve resumen de cada capítulo.

En el primer capítulo: “*Introducción*”, se describen los motivos que han llevado a la elaboración de este proyecto, así como los objetivos que se desean alcanzar con la realización del proyecto.

En el segundo capítulo: “*Estudio de viabilidad del sistema*” se realiza un estudio previo de las tecnologías existentes similares al objetivo del proyecto y a continuación se proporciona un razonamiento detallado del porqué se ha utilizado una determinada tecnología para la realización del proyecto.

El tercer capítulo: *"Análisis"*, se hace un estudio detallado de los elementos necesarios para cumplir con los objetivos marcados en el primer capítulo. Además se realiza un análisis de requisitos que van a ser referencia obligada para llevar a cabo cada punto del desarrollo del proyecto.

El cuarto capítulo: *"Diseño"*, se explica los pasos que se han llevado a cabo hace una exposición general de la estructura que va a seguir la arquitectura clúster del proyecto, haciendo énfasis en los aspectos más importantes que deben ser desarrollados.

En el quinto capítulo: "*Implementación y pruebas*", se exponen los resultados que se han obtenido al ejecutar diversas pruebas en el clúster y se realizan comparaciones con los resultados que se obtienen al ejecutar las mismas pruebas en los superordenadores más potentes del mundo.

En el sexto y último capítulo, con título "*Conclusión y trabajos futuros*" se exponen las conclusiones finales a las que se ha llegado después de la realización del proyecto, haciendo una revisión de los objetivos iniciales y estudiando si se han cumplido debidamente cada uno de ellos. Además se indican una serie de futuros trabajos que se pueden realizar como ampliación al desarrollo de este proyecto. Por último se exponen los recursos necesarios, tanto tecnológicos como de personales, para el desarrollo del mismo.

En "*Acrónimos y abreviaturas*" se pueden encontrar las definiciones de los acrónimos más importantes que aparecen a lo largo de este documento.

En "*Referencias*" se indican las páginas y los documentos a las que se ha hecho algún tipo de referencia a lo largo del presente documento.

En el primer y único anexo llamado "*Manual de usuario*" se explica cómo se hace uso de la aplicación con imágenes descriptivas de la interfaz y secuencias de pasos para realizar un uso típico de la aplicación.

Capítulo 2

Estudio de la viabilidad del sistema

En esta sección se presentará un estudio de las diferentes tecnologías existentes que presentan similitudes con el sistema a construir. De esta manera, se tendrá conocimiento de las tecnologías involucradas en el desarrollo, consiguiendo una mejoría en la eficiencia y eficacia del producto final.También en este capítulo se estudiará el estado del arte, analizando varios productos del mercado actual con similares características. Asimismo se aportará una comparativa con las ventajas que aporta cada producto en relación con el sistema a desarrollar en este proyecto.

Con este estudio se pretende adquirir una mayor comprensión del entorno tecnológico y valorar otros productos similares con el fin de evaluar si la construcción del sistema planteado es viable en el mercado actual.

2.1 Entorno tecnológico

En esta sección se exhiben cada una de las tecnologías empleadas en el presente proyecto, tanto hardware como software. Además se aportan las razones por las que se han escogido.

2.1.1 iPhone

iPhone es una familia de teléfonos inteligentes multimedia con conexión a Internet, pantalla táctil capacitiva y escasos botones físicos diseñado por la compañía Apple.

En junio de 2007, el iPhone original fue sacado al público por primera vez en Estados Unidos. Desde entonces, otras versiones mejoradas han salido a un ratio de aproximadamente una por año hasta llegar al último modelo, el iPhone 4, el cuál incorpora nuevas características innovadoras como la pantalla "Retina", que cuadruplica el número de píxeles de sus predecesores [2].

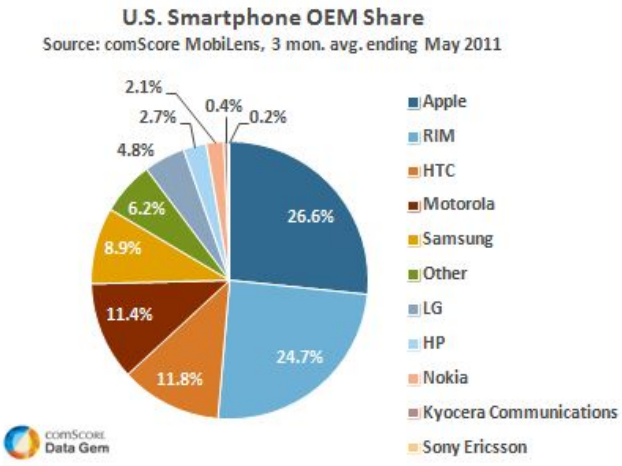


Ilustración 1. Comparativa del mercado smartphones por compañías

Desde su aparición el iPhone se ha convertido en uno de los teléfonos más vendidos del mundo, habiendo llegado a la cifra de 108.624.000 unidades vendidas a nivel mundial hasta el 24 de marzo de 2011. Los datos indican que los teléfonos de Apple son los más vendidos de los teléfonos inteligentes en estados unidos, dato recogido en mayo de 2011 [2].

En el segundo cuatrimestre de 2011 (2Q2011) el iPhone tenía un 4.6% del mercado global de teléfonos móviles y un 18.2% del mercado global de teléfonos inteligentes (smartphones) [3].

Una de las ventajas más destacadas que ofrece el dispositivo es la llamada App Store, un mercado de aplicaciones online para dispositivos Apple. La introducción del mercado App Store de Apple para el iPhone en julio de 2007 popularizó la distribución online de aplicaciones de terceros centradas en la plataforma. Siendo la App Store la que más ingresos genera en 2010 [5].

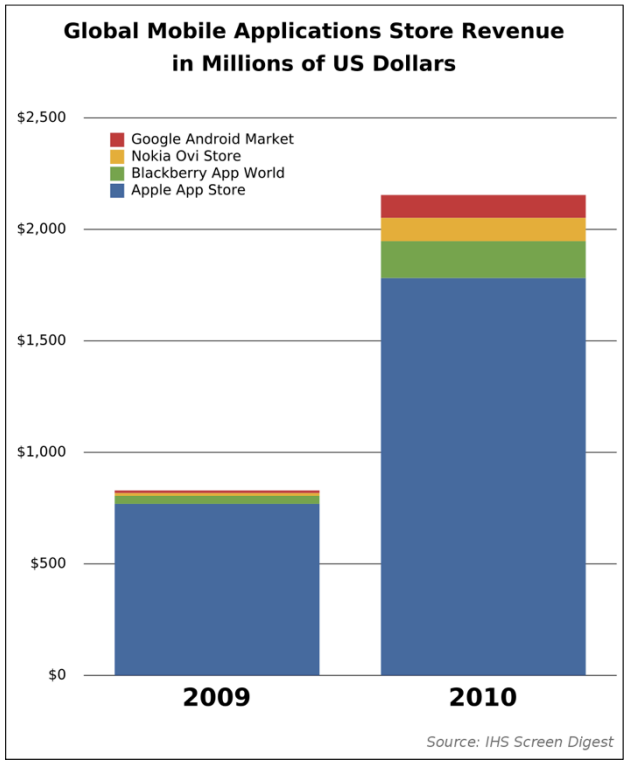


Ilustración 2. Comparativa de ventas de smartphones

Estos datos demuestran que el iPhone es uno de los dispositivos más activos en el mercado, teniendo casi 20% de las ventas de teléfonos inteligentes. Además la App Store ofrece un medio de ventas exitoso hasta la fecha, que es beneficioso para la distribución de la aplicación final de este proyecto.

En este proyecto se hará uso del iPhone 3G como terminal de pruebas ya que recoge todas las capacidades necesarias para el funcionamiento. Además se tendrá en cuenta la posible extensión a nuevos dispositivos de Apple durante el desarrollo.

2.1.2 iOS

iOS, anteriormente conocido como iPhone OS, es el sistema operativo móvil de Apple. Por el 4 de Octubre de 2011 la App Store de Apple contiene más de 500,000 aplicaciones iOS, las cuales han sido colectivamente descargadas más de 18 billones de veces [6].

Ilustración 3. Logotipo de iOS

Sus características más relevantes son:

- Programado en C, C++ y Objective-C.
- Multitáctil.
- Detección de movimientos del dispositivo: giroscopio y acelerómetro
- Multitarea (desde iOS 4.0).
- Pantalla de inicio simple: una única pantalla de inicio desde la que acceder a todas las aplicaciones.
- Organización en Ficheros.
- Centro de notificación: informa al usuario de los eventos que se producen en el sistema.
- Centro de juegos: plataforma online dónde los jugadores pueden compartir estadísticas y progresos.

El concepto iOS como sistema operativo es distinto respecto a otros sistemas operativos de escritorio. La entrada puede ser por medio de gestos táctiles como deslizamientos, pellizcos, giros, …

El sistema operativo es ampliamente utilizado por desarrolladores independientes y tiene una serie de librerías desarrolladas por Apple basadas en su antiguo NextStep, por lo que están creadas sobre una sólida base.

En el último cuatrimestre de 2010, iOS tenía un 26% de la cuota de teléfonos inteligentes en términos de unidades vendidas, detrás de Android de Google y Symbian de Nokia . La creación de la App Store, y su sencillo método

de distribución, es una de las razones por las que el uso de este sistema operativo se ha extendido tan ampliamente entre los desarrolladores.

iOS es un sistema operativo con un sólido trasfondo y un amplio catálogo de librerías y documentación, elementos muy beneficiosos a la hora de afrontar la programación de un dispositivo iPhone.

2.1.3 Cocoa Touch

Cocoa Touch es una API para construir programas software que se ejecuten sobre el sistema operativo iOS de Apple [7].

Esta API está mayoritariamente construida en Objective-C, un lenguaje orientado a objetos que es compilado para ejecutar a una gran velocidad. Objective-C es un superconjunto de C, por lo que es posible introducir C e incluso C++ en aplicaciones Cocoa Touch . Es la primera de las 4 capas que componen iOS.

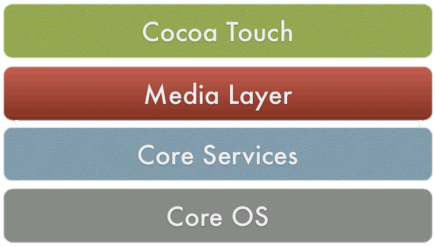


Ilustración 4. Capas de iOS

Cocoa Touch tiene múltiples frameworks para acceder a las distintas funcionalidades del dispositivo. Tiene las funciones fundamentales para realizar la mayoría de las aplicaciones [8]. Algunos de estos frameworks son:

- UIKit Framework
- Game Kit Framework
- iAd Framework
- Map Kit Framework
- Message UI Framework

El uso de este conjunto de frameworks es fundamental durante el posterior desarrollo del proyecto por lo que un conocimiento profundo de las capacidades que ofrece es un paso casi imprescindible.

2.1.4 JSON

JSON, acrónimo de JavaScript Object Notation, es un formato ligero para el intercambio de datos. JSON es un subconjunto de la notación literal de objetos de JavaScript que no requiere el uso de XML [9]. Es un formato de texto que es completamente independiente del lenguaje pero utiliza convenciones que son ampliamente conocidos por los programadores de la familia de lenguajes C, incluyendo C, C++, C#, Java, JavaScript, Perl, Python, y muchos otros. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos [9].

Ilustración 5. Logotipo de JSON

JSON es un formato de datos muy simple. Contiene dos elementos básicos: los objetos, que no son más que pares clave valor; y los arrays, que son listas ordenadas de valores. Virtualmente todos los lenguajes soportan estos dos elementos. Es razonable que un formato de datos independiente del lenguaje esté basado en estas estructuras elementales [10].

JSON tiene similitudes con el formato XML. Ambos están adaptados a la internacionalización al soportar Unicode. Poseen las mismas características de interoperabilidad y extensibilidad. Pero la clara ventaja sobre XML reside en que JSON es mucho más simple y por tanto más fácilmente entendible tanto por seres humanos como por las máquinas [11].

Existen multitud de librerías de tratamiento de datos en formato JSON para el lenguaje Objective-C, lenguaje usado en este proyecto. Entre ellas están:

- *YAJL*
- *JSONKit*
- *TouchJSON*

- *JSON Framework*

Según un análisis realizado por *"cocoanetics"* [12] la librería *JSONKit* es la más eficiente, ya que analiza y genera textos en formato JSON más rápidamente que el resto de librerías.

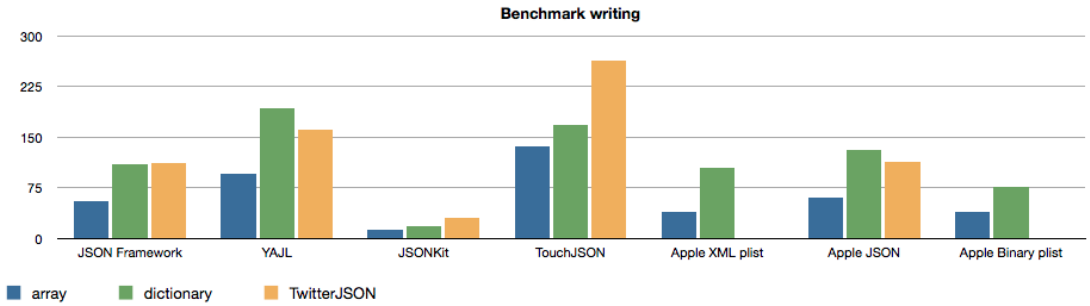


Ilustración 6. Comparativa de velocidad de escritura de distintos parsers JSON

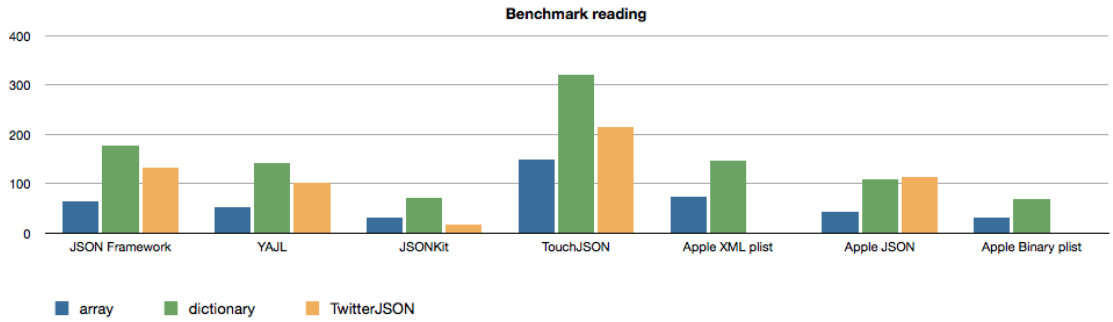


Ilustración 7. Comparativa de velocidad de lectura de distintos parsers JSON

A partir del estudio de este gráfico y de otras fuentes comparativas se ha concluido que en este proyecto se escogerá la librería JSONKit para obtener los datos a partir de las peticiones al servidor.

2.1.5 Xcode

Xcode es una suite de herramientas desarrollado por Apple, para desarrollar software para Mac OS X y iOS. Xcode 4.2, la última versión, está disponible en la App Store para Mac de manera gratuita para desarrolladores registrados [14].

Ilustración 8. Icono de Xcode

Entre las características de Xcode se encuentra la tecnología para distribuir la construcción del código fuente en varios ordenadores. Esta tecnología usa el protocolo Bonjour para descubrir automáticamente ordenadores que proveen servicios de compilador.

Gracias al formato ejecutable Mach-O, que permite binarios más grandes conteniendo el código para múltiples arquitecturas, Xcode puede construir binarios universales que permiten al software ejecutar tanto en PowerPC como en plataformas basadas en Intel. Además los compiladores proporcionados con Xcode pueden compilar aplicaciones de 32 y 64 bits para ambas arquitecturas. Xcode también puede compilar aplicaciones para iOS que ejecutan sobre el procesador ARM.

Xcode además incluye el Interface Builder, un diseñador de interfaces que relaciona los objetos creados visualmente con el código fuente de una manera simple y rápida. De esta manera se evita tener que usar terceros programas para el diseño de interfaces ganando así tiempo en el desarrollo.

Xcode ofrece además herramientas de depuración y simuladores para iPhone e iPad, convirtiéndose por tanto en un IDE indispensable en el desarrollo de aplicaciones para iOS.

2.2 Estado de la cuestión

En este apartado se estudiarán los sistemas actualmente existentes que tienen similitudes con el proyecto a desarrollar. Se aportará una descripción de dichos sistemas y se destacarán las características que tiene el presente proyecto que innoven de alguna manera sobre los sistemas existentes.

En primer lugar se debe destacar que el presente proyecto se ha categorizado como una red social.

Una red social es una estructura social compuesta por grupos de personas, las cuales están conectadas por uno o varios tipos de relaciones, tales como amistad, parentesco, intereses comunes o que comparten conocimientos [15].

Las redes sociales virtuales son aquellas creadas como una aplicación o servicio informático permitiendo a las personas conectarse entre ellas a través de internet. Una de las primeras redes sociales virtuales se llamó Six Degrees (sixdegrees.com) que surgió en 1997 y existió hasta el 2001. Este sitio brindaba la posibilidad de generar perfiles de usuarios, lista de amigos y de los amigos de estos, si se encontraban dentro del sitio como si se encontraban fuera de él. Su nombre se basa en la teoría del "Mundo pequeño", según la que cualquier ser humano está conectado a otro del planeta, por un máximo de 6 conocidos [16]. Desde su nacimiento hasta ahora han surgido muchas redes sociales, Facebook, Twitter, Tuenti, son sólo algunas de ellas.

Las razones de la creación de estas redes sociales virtuales son varios, fundamentalmente, crear un lugar virtual de fácil acceso, donde poder conectar con gente de cualquier parte del mundo sin la necesidad de desplazarse y dónde se pueda intercambiar información sobre intereses comunes.

Debido a la extensa cantidad de redes sociales, es necesario conocer y valorar la competitividad existente entre ellas ya que es un factor influyente en el éxito del presente proyecto. Dicho proyecto está enfocado a valoraciones de artículos basadas en las fotos de los mismos. Es un tipo de red social que ayuda al usuario a tomar decisiones en su vida diaria, pudiendo ver incrementada su utilidad debido a que estará enfocado al uso con los dispositivos móviles. Se trata de crear un espacio de consulta de información sobre distintos artículos y que la comunidad autogenere la base de datos de artículos y valoraciones.

A continuación se estudiarán las redes sociales que más cerca están con el objetivo de este proyecto. Se han elegido cuatro redes sociales cuyas

características en conjunto conformarían una funcionalidad similar a la de este proyecto. Se ha elegido Twitter, Twitpic, Instagram y Ciao. El estudio incluye una descripción y las similitudes y diferencias respecto al presente proyecto de cada una de ellas.

2.2.1 Twitter - Twitpic

Twitter es una red social basada en microblogging. Fue fundada en 2006 para uso exclusivamente internet y después de tener una acogida importante se abrió al publico llegando a tener actualmente cerca de 200 millones de usuarios [17].

La red es usada para mandar mensajes de texto de 140 caracteres que se muestran en la pagina principal del usuario. La clave está en que los usuarios pueden seguirse entre ellos para estar informados de lo que sus seguidos suban a la red. Así se creó una red de seguimiento que rápidamente funcionó, imitándose este estilo en muchas otras aplicaciones. Otra de las claves del éxito fue la creación de los hashtags, palabras que se utilizan para determinar el tema del mensaje.

Ilustración 9. Logotipos de Twitter y Twitpic

Internamente Twitter está escrita en Ruby on Rails y se mantiene a partir de un servidor con un software programado en Scala.

En el caso de Twitpic, es una aplicación web que permite a los usuarios de Twitter publicar imágenes en tiempo real. Se ha incluido esta última aplicación debido a que las fotos son una de las claves del proyecto y juntándolo con Twitter establecen lo que sería una primera parte de este.

Entre los puntos en común de estas dos aplicaciones, en cuanto a la estructura, se encuentran, las cuentas de usuario, los seguidores y seguidos, y la

posibilidad de compartir fácilmente con ellos las fotos de los artículos. En cuanto a la tecnología interna, el punto donde Twitter se acerca más a este proyecto es en que ofrece un API abierta para todo tipo de desarrolladores. Esto nos da la idea de que las grandes compañías de este sector se preocupan por dar una difusión rápida y ofrecer flexibilidad a los desarrolladores de clientes móviles. Al mismo tiempo hacen uso de la tecnología Memcached, para acelerar las descargas y liberar de trabajo a la base de datos. Como se comenta en su página web de ingeniería [18], esto es gran utilidad, aunque requiere mantenimiento, debido a que cuando se produce un error que colapsa al servidor Memcached, las peticiones a este servidor seguirán realizándose, elevando el número de errores hasta que vuelva a activarse el sistema.

Ilustración 10. Captura de pantalla sobre los seguidores de un usuario de Twitter

En cuanto a la posible competencia, son dos sólidas aplicaciones que se han enfocado en la comunicación interpersonal y no en la valoración de artículos, por lo que no existe un peligro hacia el objetivo de este proyecto.

2.2.2 Ciao!

Ciao es un portal europeo de compra online fundado en 1999. Su principal función es la creación de un foro de opinión sobre productos con el objetivo de ayudar a los consumidores al decidirse entre varios productos [19].

Todas las opiniones que se recogen en Ciao son públicas, pero en el caso de las valoraciones, queda reservado para usuarios registrados. Los artículos pueden ser comparados según las críticas y valoraciones de los usuarios.

Ilustración 11. Logotipo de Ciao

En cuanto a las similitudes con este proyecto, a simple vista pueden ser varias, debido a que además de disponer de los artículos con un sistema de usuarios para valoraciones y críticas, también dispone de una navegación por categorías que se acerca bastante a la de este proyecto.

Los puntos a destacar de este proyecto respecto a *Ciao!* son: la integración como red social, tan de moda últimamente y sobre todo la orientación a dispositivos móviles. También la inclusión de un sistema de seguidores y seguidos hace que el usuario esté más involucrado y la posibilidad de subir fotos sobre artículos propios que otras personas valoren es lo que marca finalmente la diferencia.

2.2.3 Instagram

Instagram es una aplicación de compartición de fotos diseñado para su uso en Apple iOS. Esta aplicación fue lanzada en 2010, y desde entonces ha evolucionado hasta convertirse en una especie de red social donde poder compartir con tus seguidores las fotos que tomas.

Ilustración 12. Logotipo de Instagram

En cuanto a esta aplicación tendría en común con el proyecto la estructura de la red social en la cual se encuentran seguidores y seguidos. Pero se separaría en la temática de la red social. Mientras que el tema de Instagram es compartir tu vida diaria y la visión artística de la foto, este proyecto se basa más en los artículos de los que hablábamos previamente con Ciao!

En la Ilustración 13 puede verse una captura de pantalla de la aplicación, mostrando los puntos similares coincidentes entre ambas aplicaciones.

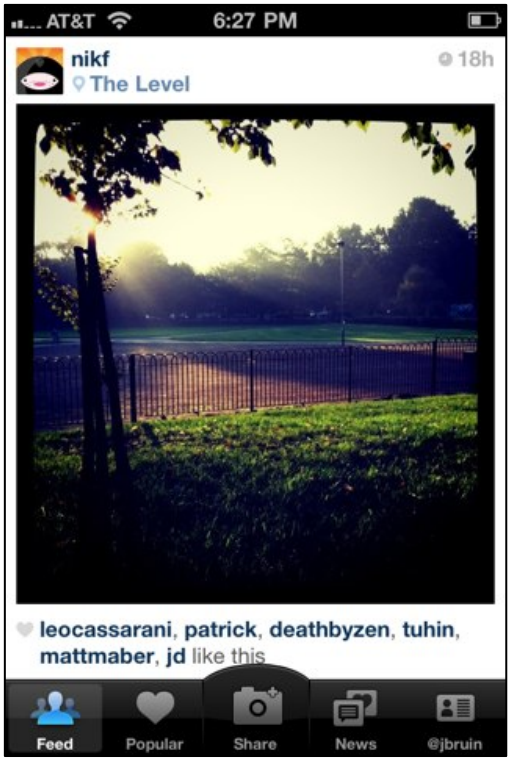


Ilustración 13. Captura de Instagram

La gran cuestión en cuanto a los competidores y las posibles ideas similares es la fidelidad de la gente a una aplicación. Independientemente del nombre que tenga, si surge una web mejor que satisface las necesidades, la gente puede migrar desde otra red social de manera fulminante como pasó con MySpace en 2009 [20].

Con este último se puede decir que hemos acabado con el estudio de mercado, ya tenemos las bases generales para situar este proyecto dentro de un dominio que se encontraría en un punto intermedio de estas tres aplicaciones.

Capítulo 3

Análisis del sistema

Una vez conocido el entorno tecnológico y el estado de la cuestión, se definirá clara y detalladamente lo que se espera del sistema. Es decir, las funcionalidades requeridas. Para ello se dará una descripción general del sistema como idea aproximada. Se concretará luego una lista de requisitos de usuario. A partir de los requisitos se generarán los casos de uso. Por último se establecerá, en base a lo anterior, la lista de requisitos software del sistema.

3.1 Descripción general

Como se ha comentado, el objetivo de este sistema es implementar un cliente para una red social de valoraciones sobre el sistema iOS. Para implementar este cliente, será necesario disponer del API documentado del servidor de la red social, para poder realizar las peticiones necesarias. Dicho servidor ha sido previamente creado y se dispone de la documentación.

La red social se basa en valorar fotos, por ello en el cliente se debe incluir todas las interfaces necesarias para que un usuario pueda hacer uso de todas las

funcionalidades de esta red social. Las funcionalidades que el cliente debe permitir son las siguientes:

- Registro de usuarios.
- Inicio y cierre de sesión.
- Vista del perfil.
- Modificación del perfil.
- Creación de nuevo artículo (foto, descripción y valoración).
- Creación de crítica de un artículo.
- Vista de lista de artículos.
- Vista de lista de críticas.
- Búsqueda de usuarios.
- Búsqueda de artículos.

Además, el cliente debe mostrar al usuario cualquier error que ocurra durante el uso de la aplicación.

3.2 Requisitos de usuario

Los requisitos de usuario son los primeros a definir en el análisis y deben ser descritos por el cliente, que en este caso concreto es el propio alumno desarrollador de la aplicación. Patrón de especificación de requisitos:

Identificador	**Nombre**		
Fuente	De qué persona/s proviene el requisito.	Estabilidad	Define si este requisito es definitivo o está en proceso de cambio aún.
Necesidad	Si es necesario incluir esta funcionalidad en el sistema final.	Prioridad	La importancia que le da el cliente a la implementación de este requisito.
Descripción	Descripción textual del requisito.		

Tabla 1. Plantilla de especificación de requisitos

3.2.1 Requisitos de capacidad

RUC-01	Crear artículo		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá capturar un objeto con la cámara, puntuarlo, agregar un título, un texto descriptivo y subir el nuevo artículo al servidor.		

Tabla 2. RUC-03

RUC-02	Valorar artículos		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá dar una valoración a un objeto pudiendo además aportar una descripción. El usuario está limitado a una valoración por artículo.		

Tabla 3. RUC-04

RUC-03	Asignar categorías a artículos		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá asignar un objeto a una categoría (Ejemplo: el artículo “coche deportivo” es asignado al grupo "Vehículos").		

Tabla 4. RUC-07

RUC-04	Navegar por artículos		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Conveniente	Prioridad	Baja
Descripción	Un usuario podrá consultar los objetos según su categoría o por medio de una lista ordenada según su puntuación.		

Tabla 5. RUC-04

RUC-05	Ver información del perfil		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá ver sus datos de perfil.		

Tabla 6. RUC-05

RUC-06	Modificar información de privacidad		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá restringir el acceso de los artículos que publica, pudiendo hacerlo privado, sólo a amigos o público.		

Tabla 7. RUC-05

RUC-07	Buscar otros usuarios		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá realizar una búsqueda de usuarios partiendo del nombre u otros datos del usuario a buscar.		

Tabla 8. RUC-05

RUC-08	Agregar seguidos		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá agregar a otros usuarios como seguidos y estos últimos podrán aceptar ser seguidos por un determinado usuario, en este caso ambos usuarios pasan a ser amigos.		

Tabla 9. RUC-05

RUC-09	Ver información de otros usuarios		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá visualizar el perfil de otros usuarios y acceder a la información que estos publican, pero únicamente si ambos usuarios son amigos.		

Tabla 10. RUC-07

RUC-10	Buscar un artículo		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá buscar un objeto por medio de su nombre.		

Tabla 11. RUC-06

RUC-11	Agregar artículo a favoritos		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Conveniente	Prioridad	Media
Descripción	Un usuario dispondrá de una lista de artículos favoritos donde podrá añadir y eliminar artículos.		

Tabla 12. RUC-06

3.2.2 Requisitos de restricción

RUR-01	Limitar uso a personas registradas		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Únicamente las personas registradas podrán hacer uso del sistema.		

Tabla 13. RUC-05

RUR-02	Interfaz en castellano		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	El sistema tendrá la interfaz en castellano.		

Tabla 14. RUC-05

RUR-03	Sencilla extensibilidad de lenguajes		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Opcional	Prioridad	Baja
Descripción	El sistema será fácilmente extensible a otros idiomas.		

Tabla 15. RUC-05

RUR-04	Protección de datos personales		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Opcional	Prioridad	Baja
Descripción	El sistema impedirá que terceros puedan acceder a información confidencial de los usuarios registrados.		

Tabla 16. RUC-05

RUR-05	Minimizar tiempos de espera		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Conveniente	Prioridad	Alta
Descripción	El sistema responderá en menos de 10 segundos a todas las peticiones realizadas desde el terminal móvil.		

Tabla 17. RUC-05

RUR-06	Permitir funcionamiento en iPhone 3G		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	La aplicación funcionará en dispositivos iPhone 3G.		

Tabla 18. RUC-05

RUR-07	Extensibilidad a otros dispositivos Apple		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	El sistema será fácilmente extensible a otros dispositivos como iPhone 4 e iPad.		

Tabla 19. RUC-05

RUR-08	Evitar registro automático		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Se impedirá que un sistema automatizado pueda crear cuentas de usuario.		

Tabla 20. RUC-05

RUR-09	Almacenar credenciales		
Fuente	Cliente	Estabilidad	Estable
Necesidad	Conveniente	Prioridad	Alta
Descripción	Los datos introducidos de inicio de sesión podrán ser almacenados en el dispositivo con el fin de evitar al usuario introducirlos cada vez que quiera acceder.		

Tabla 21. RUC-05

3.3 Casos de uso

Los casos de uso especifican qué hace el sistema desde el punto de vista del usuario. Es decir, describen un uso del sistema y cómo este interactúa con el usuario. Con esta técnica podemos describir de una manera más clara lo que va a hacer el sistema. En este apartado se mostrará una descripción tanto visual como textual de los casos de uso del sistema.

3.3.1 Diagrama

En estos diagramas se representan a los actores posibles y las interacciones que pueden realizar con la aplicación:

Ilustración 14. Diagrama de Casos de uso (I)

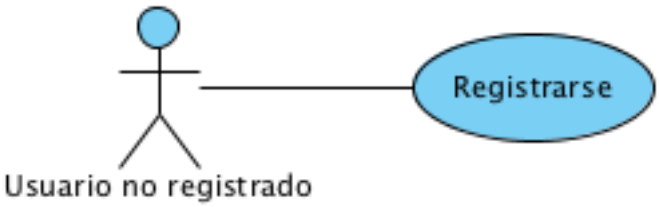


Ilustración 15. Diagrama de Casos de uso (II)

3.3.2 Descripción textual

La plantilla que se utilizará para detallar textualmente los casos de uso es la que aparece en la siguiente tabla:

Identificador	Nombre del caso de uso
Actores	*Lista de los actores involucrados en el caso de uso*
Descripción	*Meta a ser conseguida con el caso de uso y fuentes del requisito*
Suposiciones	*Condiciones que deben ser verdaderas para que el caso de uso concluya con éxito*
Flujo principal	*Interacciones entre actores y sistema que son necesarias para lograr el objetivo. El flujo básico es cuando todos los eventos ocurren en un entorno ideal en el que todo sale bien.*
Flujos alternativos	*Las excepciones al flujo básico serán consideradas en esta sección.*

Tabla 22. Plantilla de especificación de casos de uso

CU-01	Registrarse
Actores	Usuario no registrado
Descripción	Crear una cuenta de un determinado usuario
Suposiciones	El usuario no ha sido previamente registrado. El usuario creador de la cuenta no es un sistema automático.
Pasos	1. El usuario toca el botón de Registro. 2. El usuario rellena los campos necesarios con su información. Además rellena el CAPTCHA. 3. El usuario toca el botón de Confirmar. 4. Se comprueba si los datos son correctos. 5. Se da de alta la nueva cuenta.
Flujos alternativos	4.1. Si no son correctos, se muestra la pantalla de registro los campos que no son válidos.

Tabla 23. CU-01

CU-02	Darse de baja
Actores	Usuario registrado
Descripción	Eliminar la cuenta de un determinado usuario
Suposiciones	El usuario ha iniciado sesión.
Flujo principal	1. El usuario toca el botón de Dar de Baja 2. Se muestra un cuadro de diálogo al usuario para confirmar. 3. El usuario toca el botón de Confirmar. 4. Se cierra la sesión del usuario. 5. Se informa al usuario de que su cuenta ha sido eliminada.
Flujos alternativos	

Tabla 24. CU-01

CU-03	Iniciar sesión
Actores	Usuario registrado
Descripción	Iniciar una sesión dentro del sistema.
Suposiciones	El usuario ha entrado en la aplicación y se sitúa en la pantalla de inicio de sesión.
Flujo principal	1. El usuario introduce su nombre de usuario y su contraseña. 2. El usuario pulsa el botón de iniciar sesión. 3. Se envía la petición y se comprueba el resultado: 4. Si son correctos, se inicia sesión.
Flujos alternativos	3.1. Si no son correctos, se informa al usuario de que los campos no coinciden.

Tabla 25. CU-01

CU-04	Cerrar sesión
Actores	Usuario registrado
Descripción	Cerrar la sesión activa.
Suposiciones	El usuario ha iniciado sesión.
Flujo principal	1. El usuario pulsa el botón de Cerrar sesión. 2. Se envía la petición de cierre de sesión. 3. Se muestra al usuario la pantalla de Inicio de sesión.
Flujos alternativos	

Tabla 26. CU-01

CU-05	Subir artículo
Actores	Usuario registrado
Descripción	Subir un nuevo artículo al servidor.
Suposiciones	El usuario ha iniciado sesión. El usuario está en el menú principal de la aplicación.
Flujo principal	1. El usuario pulsa el botón de Nuevo Artículo. 2. Se muestra la pantalla de selección si desde la cámara o desde la biblioteca. 3. El usuario toma una foto. 4. El usuario rellena los campos necesarios de título y define una puntuación. Opcionalmente añade una descripción y una/s etiqueta/s. 5. El usuario pulsa el botón Subir. 6. Se envía la petición al servidor y se procesa el resultado: 7. Si la subida fue correcta, se informa al usuario y se vuelve al menú principal.
Flujos alternativos	6.1. Si la subida no es correcta, se muestra un mensaje de error.

Tabla 27. CU-01

CU-06	Buscar artículo
Actores	Usuario registrado
Descripción	Subir un nuevo artículo al servidor.
Suposiciones	El usuario ha iniciado sesión. El usuario está en el menú principal de la aplicación.
Flujo principal	1. El usuario pulsa el botón de Buscar. 2. Se muestra la pantalla de búsqueda. 3. El usuario introduce su búsqueda y pulsa Buscar 4. Se envía la petición al servidor y este responde con los resultados. 5. Se muestra una lista de resultados al usuario.
Flujos alternativos	

Tabla 28. CU-01

CU-07	Añadir artículo cómo favorito
Actores	Usuario registrado
Descripción	Añadir un artículo a favoritos.
Suposiciones	El usuario ha iniciado sesión. El usuario está en la vista de artículo.
Flujo principal	1. El usuario pulsa el botón de Añadir artículo a favoritos. 2. Se envía la petición al servidor. 3. Se informa al usuario.
Flujos alternativos	

Tabla 29. CU-01

CU-08	Editar artículo
Actores	Usuario registrado
Descripción	Editar un artículo
Suposiciones	El usuario ha iniciado sesión. El usuario está en la vista de artículo.
Flujo principal	1. El usuario pulsa el botón de Editar. 2. Se muestra la pantalla de edición del artículo. 3. El usuario modifica los campos que desea y pulsa el botón Aceptar. 4. Se envía la petición al servidor. 5. Se muestra la vista de artículo nuevamente.
Flujos alternativos	4.1. Si los campos son correctos, se informa al usuario de que no se puede modificar con esos nuevos valores.

Tabla 30. CU-01

CU-09	Borrar artículo
Actores	Usuario registrado
Descripción	Borrar un artículo
Suposiciones	El usuario ha iniciado sesión. El usuario está en la vista de artículo.
Flujo principal	1. El usuario pulsa el botón de Borrar. 2. Se realiza la solicitud al servidor. 3. Se borra el artículo.
Flujos alternativos	2.1. Si el usuario no es que subió el artículo, el servidor devuelve un error y no se borra el artículo.

Tabla 31. CU-01

CU-10	Criticar artículo
Actores	Usuario registrado
Descripción	Realizar una crítica de un artículo
Suposiciones	El usuario ha iniciado sesión. El usuario está en la vista de artículo.
Flujo principal	1. El usuario pulsa el botón de Añadir crítica. 2. Se muestra la vista de añadir crítica. 3. El usuario introduce los datos: texto y puntuación.
Flujos alternativos	1.1. Si el artículo no permite añadir críticas, se impide la pulsación del botón.

Tabla 32. CU-01

CU-11	Buscar usuario
Actores	Usuario registrado
Descripción	Buscar un usuario por nombre o por email.
Suposiciones	El usuario ha iniciado sesión. El usuario está en la de menú principal.
Flujo principal	1. El usuario pulsa el botón de Buscar. 2. El usuario selecciona buscar por nombre o por mail. 3. El usuario rellena el campo de búsqueda. 4. Se envía la petición al servidor. 5. Se muestra la lista de usuarios que coinciden con la consulta.
Flujos alternativos	

Tabla 33. CU-01

CU-12	Borrar contacto
Actores	Usuario registrado
Descripción	Borrar un contacto.
Suposiciones	El usuario ha iniciado sesión. El usuario está en la pantalla de perfil de algún usuario.
Flujo principal	1. El usuario pulsa el botón de Eliminar. 2. Se envía la petición al servidor. 3. Se elimina el contacto y se regresa al menú principal.
Flujos alternativos	2.1. Si el usuario no pertenece a la lista de contactos, no se realiza ninguna acción.

Tabla 34. CU-01

CU-13	Seguir usuario
Actores	Usuario registrado
Descripción	Añadir un usuario a la lista de seguidos.
Suposiciones	El usuario ha iniciado sesión. El usuario está en la pantalla de perfil de algún usuario.
Flujo principal	1. El usuario pulsa el botón de Seguir. 2. Se envía la petición al servidor. 3. El usuario que se quiere seguir acepta la petición.
Flujos alternativos	3.1. Si el usuario al que se quiere seguir rechaza la petición, no se muestra a dicho contacto como seguido.

Tabla 35. CU-01

CU-14	Editar opciones
Actores	Usuario registrado
Descripción	Editar las opciones de un determinado usuario.
Suposiciones	El usuario ha iniciado sesión. El usuario está en la pantalla de menú principal.
Flujo principal	1. El usuario pulsa el botón de Opciones. 2. El usuario modifica las opciones que desee. 3. El usuario pulsa el botón Aceptar 4. Se envían las modificaciones al servidor.
Flujos alternativos	

Tabla 36. CU-01

3.4 Requisitos Software

A partir de los requisitos de usuario y los casos de uso podemos obtener la lista de los requisitos software que serán la base fundamental de lo que va a ser finalmente la aplicación.

3.4.1 Requisitos funcionales

RSF-01	Registrar a un nuevo usuario		
Fuente	RUR-01	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Dar de alta en el sistema a un nuevo usuario partiendo de sus datos personales.		

Tabla 37. RUC-01

RSF-02	Dar de baja a un usuario		
Fuente	RUR-01	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá dar de baja su cuenta.		

Tabla 38. RUC-02

RSF-03	Abrir sesión		
Fuente	RUR-01	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá abrir una sesión en el sistema proporcionando sus datos de cuenta.		

Tabla 39. RUC-02

RSF-04	Cerrar sesión		
Fuente	RUR-01	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá cerrar su sesión en cualquier momento.		

Tabla 40. RUC-07

RSF-05	Crear nuevo artículo		
Fuente	RUC-01	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá añadir un nuevo artículo, formado por una imagen, un título y una puntuación. Opcionalmente puede contener una descripción y unas etiquetas.		

Tabla 41. RUC-03

RSF-06	Ver información del perfil		
Fuente	RUC-07	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá visualizar la información de su perfil.		

Tabla 42. RUC-04

RSF-07	Buscar otros usuarios		
Fuente	RUC-09	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá realizar una búsqueda de usuarios partiendo del nombre u otros datos del usuario a buscar.		

Tabla 43. RUC-05

RSF-08	Agregar seguidos		
Fuente	RUC-10	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá agregar a otros usuarios como seguidos y estos últimos podrán aceptar ser seguidos por un determinado usuario.		

Tabla 44. RUC-05

RSF-09	Mostrar seguidos		
Fuente	RUC-07	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá ver una lista con todos los usuarios seguidos.		

Tabla 45. RUC-05

RSF-10	Ver perfil de otros usuarios		
Fuente	RUC-11	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá visualizar el perfil de otros usuarios. La información disponible para un determinado usuario dependerá de la relación entre ambos usuarios y de las opciones de privacidad de ambos.		

Tabla 46. RUC-07

RSF-11	Establecer opciones de privacidad		
Fuente	RUC-11	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Los usuarios podrán decidir si sus datos son públicos, mostrados sólo a amigos o privados.		

Tabla 47. RUC-07

RSF-12	Buscar un artículo		
Fuente	RUC-05	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Se debe permitir la búsqueda de un artículo por medio de su nombre.		

Tabla 48. RUC-06

RSF-13	Agregar artículo a favoritos		
Fuente	RUC-06	Estabilidad	Estable
Necesidad	Conveniente	Prioridad	Media
Descripción	Un usuario podrá agregar un artículo a su lista de artículos favoritos.		

Tabla 49. RUC-06

RSF-14	Puntuar un artículo		
Fuente	RUC-02	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá hacer una crítica o una valoración de un artículo, aportando un texto y una puntuación.		

Tabla 50. RUC-07

RSF-15	Mostrar valoraciones		
Fuente	RUC-07	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Un usuario podrá ver las valoraciones de un determinado artículo.		

Tabla 51. RUC-07

RSF-16	Mostrar categorías		
Fuente	RUC-03, RUC-04	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Media
Descripción	Los artículos se encontrarán clasificados según categorías definidas en el servidor y el cliente podrá acceder y mostrar estas categorías.		

Tabla 52. RUC-07

RSF-17	Top de artículos		
Fuente	RUC-04	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Los usuarios podrán visualizar el top de los artículos mejor puntuados.		

Tabla 53. RUC-07

3.4.2 Requisitos No Funcionales:

Requisitos de interfaz

RSNF-01	Interfaz en castellano		
Fuente	RUR-08	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	La interfaz del sistema estará en castellano.		

Tabla 54. RUC-05

Requisitos de rendimiento

RSNF-02	Minimizar tiempos de espera		
Fuente	RUR-05	Estabilidad	Estable
Necesidad	Conveniente	Prioridad	Alta
Descripción	El sistema responderá en menos de 10 segundos a todas las peticiones realizadas desde el terminal móvil.		

Tabla 55. RUC-05

Requisitos de operación

RSNF-03	Funcionamiento en iPhone 3G		
Fuente	RUR-06	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	La aplicación funcionará en un iPhone 3G de Apple.		

Tabla 56. RUC-05

RSNF-04	Implementación en iOS		
Fuente	RUR-06	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	La aplicación funcionará en el sistema operativo iOS versión 4.0 de Apple®.		

Tabla 57. RUC-05

Requisitos de seguridad

RSNF-05	Uso de CAPTCHA en el registro		
Fuente	RUR-08	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	Durante el registro se requerirá un reconocimiento de CAPTCHA para permitir crear un nuevo usuario.		

Tabla 58. RUC-05

Requisitos de extensibilidad

RSNF-06	Sistema fácilmente traducible		
Fuente	RUR-03	Estabilidad	Estable
Necesidad	Esencial	Prioridad	Alta
Descripción	El sistema deberá ser fácilmente extensible a otros idiomas.		

Tabla 59. RUC-05

Capítulo 4

Diseño

En este capítulo se presentará tanto el diseño arquitectónico como el diseño detallado del sistema. Se dará una explicación de cada una de las decisiones de diseño tomadas. Estos pasos son de una gran importancia ya que son el punto de partida de la implementación.

4.1 Arquitectura del sistema

Debido a la magnitud del proyecto el sistema global de la red social se ha separado en servidor y cliente. La parte de servidor será implementada por separado mientras que el presente proyecto tiene como objetivo la implementación del cliente para el sistema iOS.

Dicho esto, para el sistema se ha seleccionado un patrón arquitectónico Modelo Vista Controlador (MVC) que se comporta a su vez como un Cliente-Servidor. A continuación se explicará brevemente en qué consisten ambos modelos arquitectónicos.

4.1.1 MVC

El patrón MVC separa los datos de una aplicación (modelo), la interfaz de usuario (vista), y la lógica de negocio (controlador) en tres roles distintos. El patrón define no sólo los roles que desempeñan los objetos en la aplicación sino también la manera en la que los dichos objetos se comunican. Cada uno de los tres tipos de objetos está separado de los otros por límites abstractos y se comunica con objetos de otros tipos cruzando esos límites [21]. A continuación se detallan los tres componentes:

Modelo: Esta es la representación específica de la información con la cual el sistema opera. En resumen, el modelo se limita a lo relativo de la vista y su controlador facilitando las presentaciones visuales complejas. El sistema también puede operar con más datos no relativos a la presentación, haciendo uso integrado de otras lógicas de negocio y de datos afines con el sistema modelado.

Vista: Este presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario.

Controlador: Este responde a eventos, usualmente acciones del usuario, e invoca peticiones al modelo y, probablemente, a la vista.

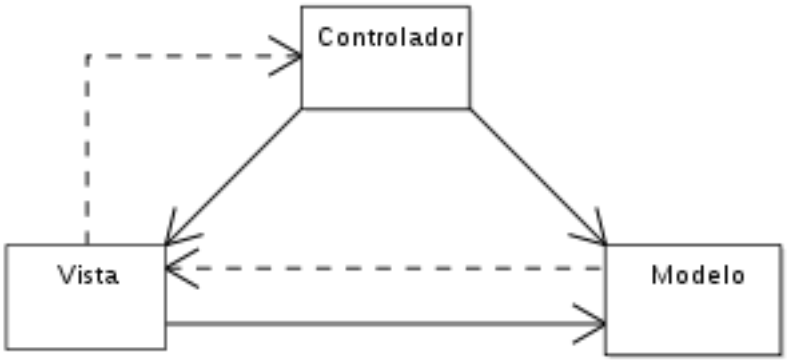


Ilustración 16. Diagrama de un MVC

4.1.2 Cliente – Servidor

El modelo cliente-servidor es un modelo de aplicación distribuida en la que un prestador de servicio, llamado servidor, responde a las peticiones de los

clientes. Una máquina servidora es una máquina que está ejecutando uno o más programas servidores que comparten sus recursos con los clientes. Un cliente no comparte sus recursos sino que solicita al servidor su contenido o servicio. Por tanto son los clientes quienes inician el proceso de comunicación.

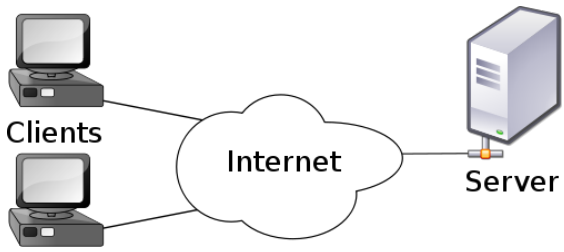


Ilustración 17. Diagrama de un modelo Cliente-Servidor

El sistema se comporta como un cliente-servidor ya que es un sistema distribuido en el que son los clientes los que realizan las peticiones a la API. El servidor se limita a responder estas peticiones aportando los datos solicitados.

El sistema completo de la red social se compone de un servidor y clientes móviles para distintas plataformas. La arquitectura establece que el servidor contiene tanto el modelo, es decir, la base de datos como el controlador, o la API a la que tienen acceso los clientes. La aplicación para el dispositivo móvil, en este caso, el cliente iPhone es el componente de vista del sistema.

La arquitectura del sistema es un Modelo Vista Controlador modificado en el que el cliente no tiene permitido acceder al Modelo directamente sino siempre a través del Controlador. Se ha tomado esta decisión porque esta arquitectura ofrece una mayor modularización pudiendo crearse un conjunto de funciones a través de la cuál se puede acceder al modelo y que son independientes de la plataforma de la Vista.

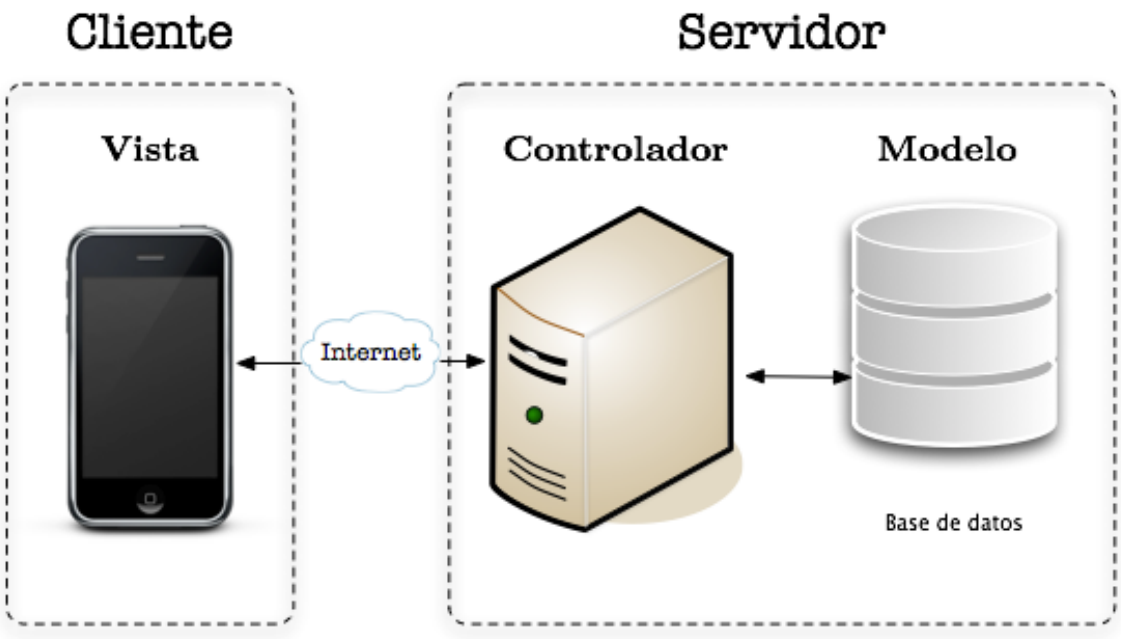


Ilustración 18. Diagrama del la arquitectura del sistema: MVC y Cliente-Servidor

Los pasos a seguir para obtener o enviar datos al servidor son los siguientes.:

1. El usuario ejecuta una acción en la aplicación cliente (Vista).
2. La aplicación crea una solicitud HTTP y la envía a la URL donde reside el servidor.
3. El servidor realiza la lógica de control (Controlador) y hace la consulta a la base de datos (Modelo)
4. El servidor crea una paquete de respuesta HTTP, lo codifica en formato JSON y lo envía a dirección desde la que recibió la petición, es decir, al cliente.
5. La aplicación cliente recibe la respuesta del servidor, la decodifica, procesa los datos y los muestra en pantalla al usuario.

Como se ha indicado con anterioridad, el presente proyecto se dedicará a diseñar e implementar la parte de la Vista.

4.1.3 Arquitectura del cliente iOS

En la aplicación cliente para iOS se va a emplear otro MVC. Dicho patrón es intrínsecamente usado por la aplicación cliente debido a que *Cocoa Framework*, el conjunto de herramientas nativas para la programación en iOS, está basado en el patrón de diseño MVC.

En *Cocoa Framework* el patrón MVC tiene roles específicos [22] que serán explicados a continuación:

El **Modelo** está compuesto por objetos del modelo. Estos objetos encapsulan los datos específicos de una aplicación y definen la lógica que manipula y procesa esos datos. Por ejemplo, un objeto del modelo podría representar un personaje en un juego o un contacto en una libreta de direcciones. Debido a que los objetos del modelo representan conocimiento o saber relacionados a un problema de un dominio específico, pueden ser reusados en problemas de dominios similares. Esta es una ventaja de la modularidad. Idealmente, un objeto del modelo debería no tener conexión explícita a los objetos de la vista, cuya función es presentar los datos.

Los objetos de la **Vista** son los objetos de la aplicación que el usuario puede ver. Un objeto de vista sabe como dibujarse y puede responder a las acciones del usuario. El principal objetivo de los objetos de vista es mostrar los datos de los objetos del modelo.

Cada objeto **Controlador** actúa como un intermediario entre uno o más de los objetos de la vista y uno o más de los objetos del modelo. Los objetos controladores son por ello un conducto a través del cual los objetos de la vista son conscientes de los cambios en el modelo y viceversa.

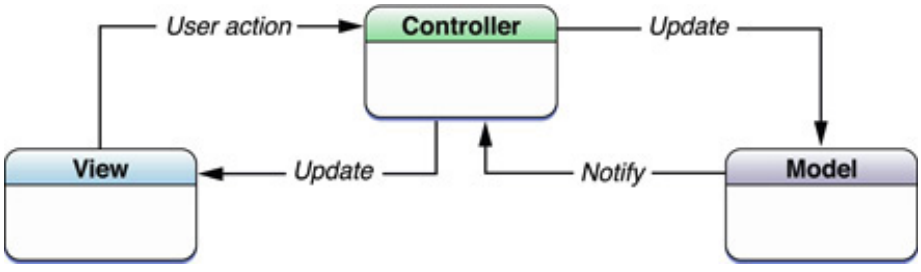


Ilustración 19. Diagrama del patrón MVC en el dominio de Cocoa Framework

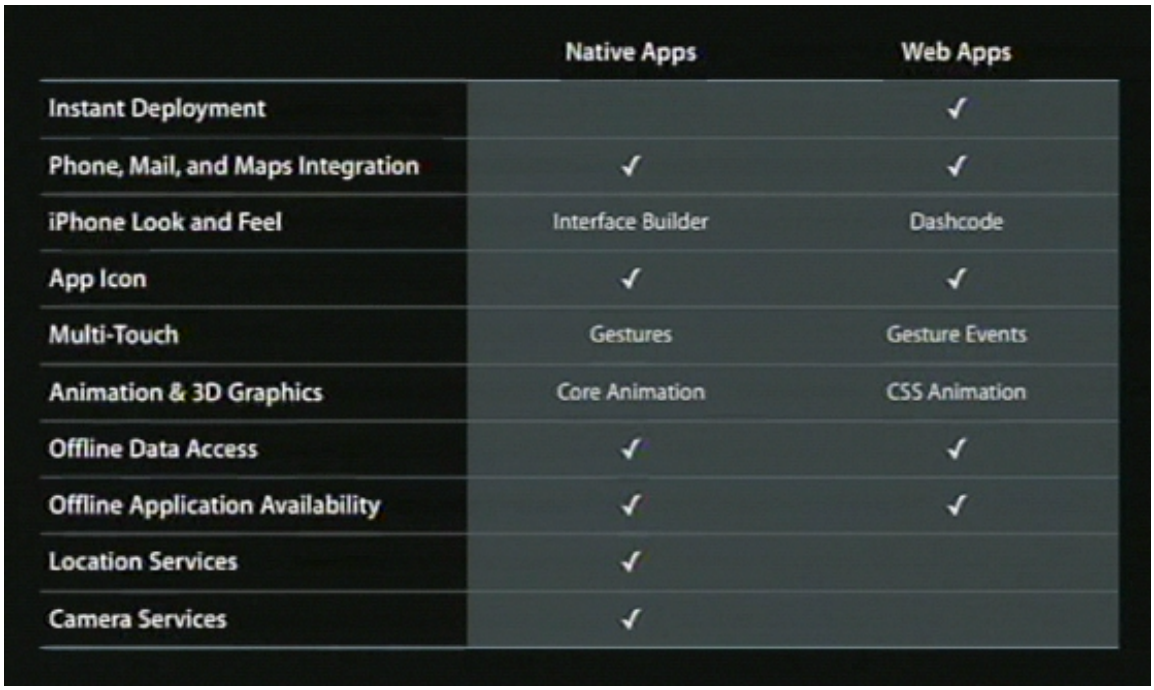
4.2 Diseño detallado

El diseño detallado mostrará el diseño de las interfaces de la aplicación y a continuación el diagrama de clases para la aplicación cliente iOS. Este diseño de clases corresponde a la arquitectura MVC como se explicó en el apartado anterior.

4.2.1 Decisiones de diseño

Aplicación nativa o aplicación web

Antes de empezar a diseñar las interfaces fue necesario plantearse la cuestión de si se iba a realizar una aplicación nativa sobre iOS o una aplicación web que ejecutase en Safari, el navegador del iPhone. Ambas decisiones tienen sus ventajas y sus inconvenientes.



	Native Apps	Web Apps
Instant Deployment		✓
Phone, Mail, and Maps Integration	✓	✓
iPhone Look and Feel	Interface Builder	Dashcode
App Icon	✓	✓
Multi-Touch	Gestures	Gesture Events
Animation & 3D Graphics	Core Animation	CSS Animation
Offline Data Access	✓	✓
Offline Application Availability	✓	✓
Location Services	✓	
Camera Services	✓	

Ilustración 20. Tabla comparativa entre aplicaciones nativas y aplicaciones web

Para desarrollar de forma nativa sobre iOS es necesario aprender el lenguaje objetive-C y las librerías involucradas. De esta forma se crea una aplicación que no puede ser reusada en otros dispositivos (excepto los basados en iOS como iPod Touch o iPad) sino que es específica del iPhone.

Programar una aplicación web requiere conocer HTML, CSS y muy probablemente JavaScript. Estos últimos lenguajes son muy populares y con una extensa documentación. Sin embargo, programar de esta manera impide tener acceso a ciertos elementos del núcleo de iOS tales como el micrófono o la cámara.

Tampoco se puede obtener acceso a los servicios de localización GPS desde una aplicación web. La falta del servicio de localización y la imposibilidad del uso de la cámara impiden la construcción de una característica fundamental del sistema, que es la creación de artículos a partir de una foto estando opcionalmente geo localizados. Por consiguiente y a pesar de tener que aprender un nuevo lenguaje de programación se ha tomado la decisión de implementar una aplicación nativa.

4.2.2 Diseño de interfaces

Esta proyecto consiste fundamentalmente en un conjunto de interfaces para proporcionar al usuario el acceso y los servicios de la red social de valoraciones. Por consiguiente el proyecto está orientado directamente a la interacción con el usuario y el diseño interfaces es un factor clave. Así pues, el primer apartado del diseño detallado es el conjunto de interfaces diseñadas ya que a partir de este conjunto se obtendrán las clases necesarias para la implementación.

Se debe decir que el diseño de clases implica una comprensión profunda de cómo el usuario va a desear interactuar con el sistema. Se necesita saber cuáles son las acciones típicas, los datos que el usuario necesita saber, etc. El objetivo de esto es hacer una interfaz sencilla y cómoda para los usuarios.

El diseño de las interfaces parte de la base del conjunto de requisitos software. Se ha procurado crear una serie de interfaces que satisfagan completamente dichos requisitos.

Antes de comenzar al dibujo de las interfaces se ha considerado la lectura de los *Human Interface Guídeles*. Son una serie de directrices y recomendaciones de programación para dispositivos móviles con pantalla táctil de Apple [23].

Estas recomendaciones sirven para evitar cometer errores comunes y integrar la capacidad táctil con el desarrollo de las interfaces.

Para la creación de las interfaces para el dispositivo iPhone se ha comenzado por el dibujo sobre papel de las interfaces. Se ha usado una plantilla para facilitar la tarea.

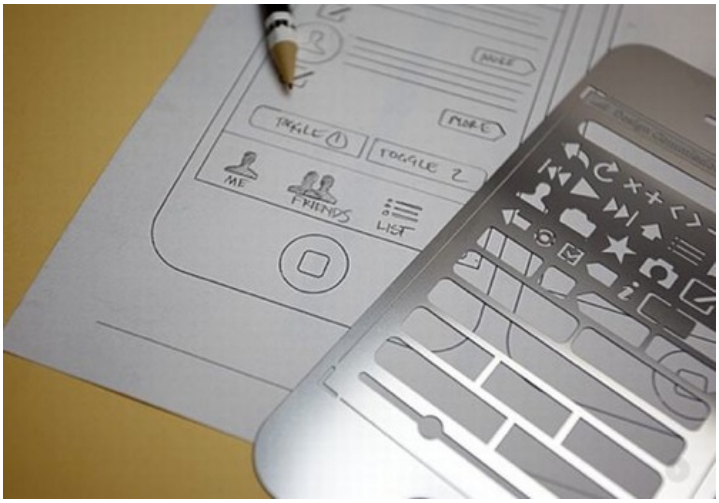


Ilustración 21. Plantilla para creación de interfaces en papel

Una vez en papel las interfaces, se han dibujado usando un editor gráfico para digitalizar las interfaces, en este proyecto la herramienta empleada ha sido Adobe Photoshop CS5. A continuación se exponen una serie de imágenes representativas que ilustran algunas de las interfaces creadas.

Ilustración 22. Interfaz de ejemplo 1 de creación de interfaces

El tiempo invertido en la creación de estas representaciones gráficas – no funcionales – de las interfaces ha sido imprescindible. Este método tiene la ventaja de que se puede tener una visión general, hasta el nivel de píxel, de la pantalla antes de comenzar a programarla por lo que todos los parámetros (posiciones, colores, etc.) ya han sido calculados, y la programación se hace mucho más ágil.

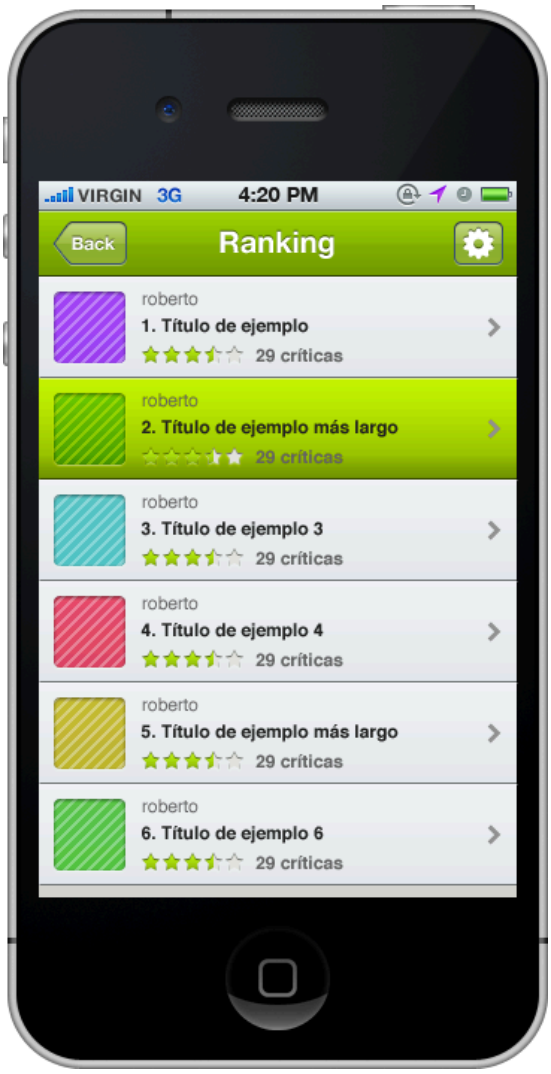


Ilustración 23. Interfaz de ejemplo 2 de creación de interfaces

Debido a que cada elemento de la interfaz se debe programar después, una interfaz aunque sencilla de dibujar en el editor gráfico puede ser bastante tediosa de programar. A la hora de considerar cuánto tiempo se debería invertir en la programación de las interfaces se tomó el criterio de hacer prevalecer la sencillez ya que en muchas aplicaciones del mercado predomina la simplicidad de la interfaz [24]. Una interfaz con muchos botones puede resultar compleja para los usuarios.

Ilustración 24. Interfaz de ejemplo 3 de creación de interfaces

4.2.3 Diagrama de clases

El diagrama de clases parte de la base del conjunto de interfaces y de los requisitos software. Para diseñarlo fue necesario documentación del funcionamiento del sistema iOS.

A continuación se muestra el diagrama de clases:

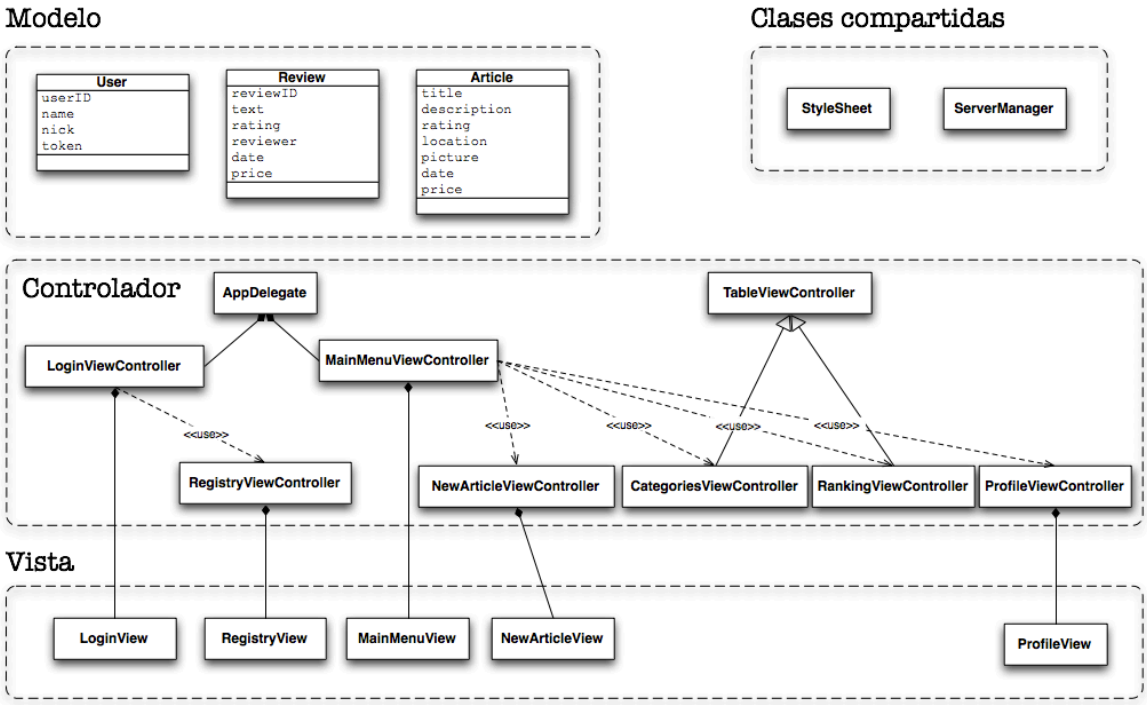


Ilustración 25. Diagrama de clases

Como se aprecia en el diagrama anterior, el diseño de clases está guiado por el patrón MVC. Como se ha visto, el *framework Cocoa* está basado en tal patrón y por ello es tenido en cuenta en el diseño de clases.

Cada pantalla es representada por una vista, por tanto se han hecho necesarias tantas vistas como pantallas (diferentes) tenga nuestra aplicación. Algunas pantallas son reutilizables. Como se aprecia en el diagrama de clases, *CategoriesViewController* y *RankingViewController*, son ambas vistas en forma de lista de elementos. Por ello ambas clases heredan de *TableViewControler* que es una clase con métodos para manipulación de tablas.

Los controladores existentes manipulan la información recibida por las vistas y actualizan tanto el modelos como dichas vistas. *AppDelegate* es el controlador general de la aplicación y da paso al resto de controladores. Tienes instancias de *LoginViewController*, el controlador de la pantalla de acceso y con *MainMenuViewController*, el controlador del menú principal, ya que da paso al inicio de sesión si no hay guardado ningún dato de inicio de sesión o al menú principal en otro caso.

Desde el menú se puede acceder a crear un nuevo artículo, a ver el ranking, a ver las categorías de artículos o al perfil por lo que tiene una instancia de cada uno de esos controladores tal y como muestra el diagrama.

Las clases del modelo son empleadas por los controladores durante toda la ejecución. Como ejemplo, la clase *LoginViewController* actualiza la información del usuario, la clase *User*, y almacena esta información en memoria no volátil, para permitir que accesos posteriores sin necesidad de introducir los credenciales.

Se ha considerado en el diseño una serie de clases compartidas que contienen elementos usados por todas o muchas de las clases del programa. Estas clases son:

StyleSheet: clase que contiene los estilos visuales de los elementos que se usarán. Elementos tales como los colores del degradado de los botones o el tipo fuente de la barra de navegación entre otros.

ServerManager: esta clase alberga todas las funciones de red relacionadas con el servidor de la red social. Provee las funciones necesarias para realizar cómodamente peticiones HTTP usando la notación JSON y otras funciones que requieren de comunicaciones de red. De esta manera se gestiona el tráfico de red en un solo fichero y se reduce el código necesario en otros ficheros de la aplicación. Esto ve sus beneficios ya que se trata de una aplicación que se mantiene conectada a un servidor, y son las comunicaciones necesarias en muchas ocasiones.

Capítulo 5

Implementación y pruebas

En este capítulo se explica el procedimiento y los detalles de la implementación. Adicionalmente se incluye el conjunto de pruebas realizadas para la verificación del correcto funcionamiento del sistema. Por último se incluye la matriz de trazabilidad entre pruebas y requisitos de software.

5.1 Implementación

En esta sección se presentan los pasos realizados en la implementación de la aplicación así como todos los problemas que han ido apareciendo a lo largo de la misma y las soluciones planteadas para resolver dichos problemas.

A continuación se muestran las distintas partes implementadas según su orden de aparición el proyecto.

1. **Gestión de los recursos de red:** Para ejercer cualquier comunicación con el servidor web es necesaria la creación de un conjunto de funciones que permitan tanto realizar solicitudes a partir de una URL como gestionar el resultado de las mismas. En este primer paso se implementará

un conjunto de clases que permitan establecer comunicaciones de red con el servidor.

2. **Implementación de las pantallas:** Este es el segundo y más importante paso, se trata de implementar la vista, el controlador y el modelo de las distintas pantallas de la aplicación.

3. **Creación de ficheros de traducción:** Este último paso consiste en permitir la traducción de la aplicación a varios idiomas de manera sencilla.

Seguidamente se exponen los detalles de la implementación de cada uno de los pasos expuestos anteriormente.

5.1.1 Gestión de los recursos de red

Como se ha mencionado con anterioridad, el sistema a construir sigue el patrón Cliente-Servidor y dado que se trata de un sistema distribuido, es habitual que el cliente tenga que realizar frecuentes solicitudes al servidor. Dichas solicitudes son dependientes del servidor y no siempre son iguales. Una gestión de todas estas solicitudes de forma genérica requeriría una gran cantidad de código. Por lo tanto es muy conveniente crear una serie de funciones comunes que permitan la comunicación con el servidor en pocas llamadas. Esto tiene dos ventajas:

- Evitar redundancia de código.

- Crear un lugar único de gestión de todos los errores de red.

Una de las virtudes del lenguaje Objective-C es que dispone de muchas librerías de ayuda a la programación. Una de ellas es *ASIHTTPRequest*, una librería que implementa las solicitudes de red de una manera sencilla. Esta librería ha sido empleada como base para las comunicaciones de red.

El servidor dispone de una API en la que se muestran todas las llamadas al servidor, sus parámetros y la forma de sus respuestas. Conocer todas esas llamadas y posibles errores es un paso básico para la implementación de este gestor de red.

Se ha implementado una clase denominada *ServerManager* que dispone de varios métodos. Uno de ellos sirve para la creación de las URLs a partir de los parámetros necesarios de la llamada. Otra función, que complementa a la

primera, permite realizar una solicitud a partir de una URL e invocar una determinada función al completar la solicitud.

El gestor también procesa el resultado de la solicitud. Como se ha dicho el servidor web devuelve una cadena codificada en notación JSON en cada una de sus respuestas por lo que se requiere un decodificador JSON que traduzca la cadena obtenida a un objeto entendible por la aplicación.

Para facilitar el trabajo de gestión de errores del servidor, se ha implementado una jerarquía de clases que se van especializando en el tipo de solicitud. La raíz de esta jerarquía es una clase que permite hacer solicitudes HTTP a partir de una URL. Una subclase de esta es `JSONFetcher`, una clase que decodifica la respuesta obtenida con un decodificador JSON. Adicionalmente y con el objetivo de unificar el lugar de gestión de los posibles errores en las respuestas del servidor, se ha añadido una última clase, que hereda de la segunda, que comprueba si ha habido error y muestra un mensaje de alerta en dicho caso. El diagrama de clases para esta jerarquía es el siguiente:

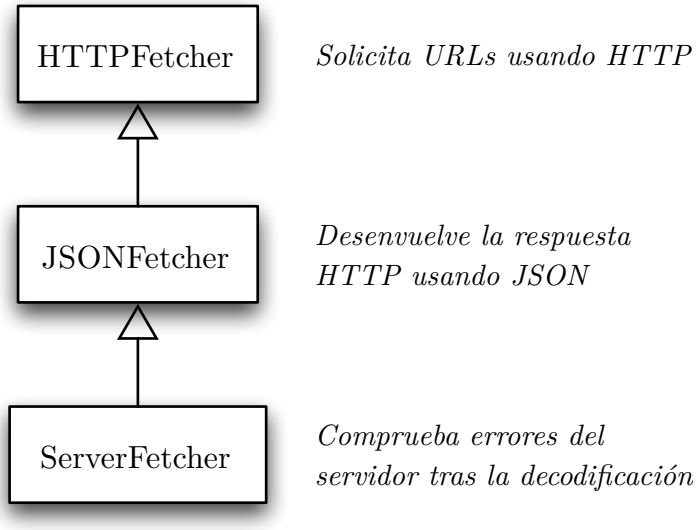


Ilustración 26. Diagrama de clases para las solicitudes de red

Para decodificar la respuesta HTTP se ha empleado una librería para traducir cadenas en notación JSON denominada JSONKit [13]. Se ha hablado de ella en el capítulo 2. Su objetivo es básicamente convertir cadenas JSON a objetos Objective-C, ya sean arrays, diccionarios, etc.

5.1.2 Implementación de las pantallas

Una vez creadas las funciones de comunicación con el servidor, se procede a implementar las distintas pantallas de la aplicación. La creación de las pantallas supone la carga más importante de la parte de implementación de este proyecto. Por ello se explicará con detalle cada una de las pantallas por separado.

Para la implementación de las pantallas ha sido necesario crear clases separadas para la vista, el modelo y el controlador. Es posible implementar la vista dentro de la clase del controlador pero en la práctica sólo se hace si la vista es muy sencilla

Pantalla de inicio de sesión

La primera pantalla es la pantalla de acceso o *login.* Esta pantalla tiene la funcionalidad de permitir a los usuarios acceder al sistema al introducir sus datos o bien ofrece al usuario la posibilidad de registrarse a través de un enlace que lleva a la pantalla de registro. Es una pantalla muy simple que usa dos campos de texto para introducir nombre de usuario y contraseña y un botón para acceder.

Esta pantalla tiene la peculiaridad de que permite almacenar los credenciales de un usuario permitiendo que no sea necesario introducir los datos de inicio de sesión la segunda vez que acceda a la aplicación. Para permitir la gestión de credenciales se hace uso del elemento `NSUserDefaults`. Esta clase provee métodos para leer y escribir variables de forma no volátil, o lo que es lo mismo, que no son eliminadas al cerrar la aplicación.

Pantalla de registro

Esta pantalla permite el registro de los usuarios. Los usuarios introducen los campos nombre, nombre de usuario, correo electrónico y contraseña.

Cada campo es validado a la vez que se escribe. Para ello se hace uso de expresiones regulares, para verificar que el email sigue la forma *nombre@dominio.tipo*, y también se hace uso de llamadas al servidor para saber si el nombre de usuario o el correo electrónico no ha sido registrado previamente.

Para el campo contraseña se ha implementado una barra de seguridad que proporciona una idea visual de lo segura que es la contraseña.

Como medida de seguridad y para impedir la creación automática de cuentas se emplea un CAPTCHA. El usuario tiene la posibilidad de descargar una nueva imagen CAPTCHA. La petición del *captcha* se realiza al servidor web y este devuelve un reto. A partir de este reto se hace una petición al servidor del servicio *Recaptcha* de Google y este devuelve la imagen necesaria.

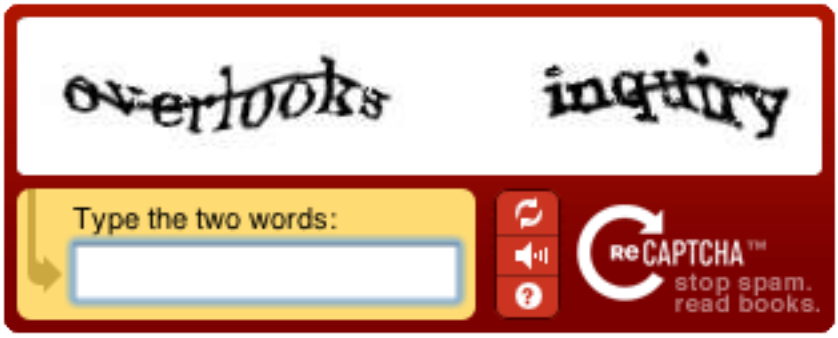


Ilustración 27. Ejemplo de CAPTCHA

Pantalla de menú principal

El menú principal es la pantalla raíz de una ejecución típica de la aplicación. Desde el menú se puede acceder a todos los servicios del cliente. Permite entre otras cosas acceder al perfil, acceder a la sección de categorías, acceder al los usuarios seguidos, realizar búsquedas de usuarios o de artículos… En la barra de navegación se han implementado la posibilidad de volver a la pantalla de login, para el caso de que se quiera cambiar de usuario.

Pantalla de perfil

La pantalla de perfil incluye toda la información de un determinado usuario. Desde su nombre de usuario hasta su fecha de nacimiento. Desde el perfil se puede acceder a las críticas del usuario, a sus favoritos, a los seguidores y a sus seguidos.

Se ha implementado botón para permitir seguir o dejar de seguir a un determinado usuario desde el perfil. Para ello se realiza una consulta al servidor de los seguidos al cargar un determinado usuario y se fija el texto del botón en cada caso.

Pantalla de nuevo artículo

Desde esta pantalla se puede crear y subir al servidor un artículo. Un artículo está compuesto por tres campos obligatorios: título, imagen y valoración; y cuatro campos opcionales: descripción, precio, etiquetas, localización.

La obtención de la imagen se hace mediante el uso de `UIImagePickerController`, un controlador que muestra una vista modal de un la cámara preparada para la toma de imágenes. Se ha implementado tanto la posibilidad de seleccionar la imagen desde la cámara cómo desde la biblioteca de imágenes.

Otro aspecto a destacar en la implementación de esta pantalla ha sido la obtención de la localización por GPS. El *Cocoa Framework* provee ciertas clases para tal efecto, la clase empleada ha sido `CLLocationManager`. Para obtener una posición GPS es necesario solicitar el comienzo de las actualizaciones y permanecer a la escucha de las nuevas posiciones obtenidas. Una vez fijado el punto de coordenadas es almacenado a la espera de que el usuario solicite la subida del artículo.

Como se detalla en la API del servidor, un artículos requiere de una solicitud HTTP de tipo POST. Por ello cuando el usuario presiona el botón de subir artículo, se crea una solicitud POST con todos los campos introducidos y se muestra un cuadro de diálogo con una barra de progreso que indica el porcentaje del nuevo artículo que ha sido cargado al servidor.

Pantallas de listas

Las pantallas de listas son tablas que albergan celdas de un determinado tipo. Más concretamente, sus controladores heredan de `UITableViewController,` la clase base de *Cocoa Framework* que ejerce de controlador de vistas de tablas.

Cada pantalla de lista tiene celdas de un cierto tipo y con un determinado diseño. Concretamente para la pantalla de lista de usuarios se ha implementado un tipo de celda, llamada `UserCellView`, que dispone los datos de un usuario: su foto, su nombre de usuario y su nombre real.

Muchas pantallas de la aplicación tienen forma de lista: los seguidos de un usuario, los artículos favoritos o las críticas de un artículos entre otros. Todos ellos son vistas de lista y todas descargan su modelo de datos desde el servidor. Sólo se diferencian en el tipo de celda. Por ello es razonable pensar que conviene

diseñar una clase genérica que controle a las vistas de lista con un modelo de datos a partir de una URL y con un tipo de celda variable. Con tal objetivo se ha implementado una clase que permite hacer lo descrito anteriormente. Y de esta clase heredan las clases controlador de las pantallas de lista usadas en la aplicación.

Los objetos que son mostrados en la aplicación en forma de lista son tres:

- Usuarios
- Artículos
- Críticas

Por lo tanto se requiere crear una clase controlador para las pantallas de listas de cada uno de estos objetos.

Para la implementar las pantallas de lista de cada tipo sólo es necesario crear una clase controlador que herede de la clase genérica creada como se menciona anteriormente, crear la vista de la celda para el objeto que se desea mostrar y por último asignar los datos obtenidos desde la red a los datos de la celda.

Una característica que se ha implementado en las pantallas de listas es la opción de *"arrastrar y refrescar"* (*drag-and-refresh*). Esta característica permite que si el usuario intenta ver lo que hay más allá del inicio de la lista, una nota le dirá que puede soltar para actualizar. Esta popular característica presente en muchas aplicaciones actuales permite que una lista ordenada de manera cronológica descendente pueda cargar las últimas actualizaciones de una manera muy sencilla e intuitiva. Esta característica se ha implementado haciendo uso de la clase `TTTableViewController` de la librería *Three20*.

Pantalla de nueva crítica

Esta pantalla muestra los campos necesarios para construir una crítica y permite enviar dicha crítica al servidor. Esta pantalla se muestra como una vista modal, es decir, se muestra como una pantalla única en forma de cuadro de diálogo que aparece para solicitar datos, sin entrar en la jerarquía de navegación. Para implementar este comportamiento modal, la clase hereda de `TTMessageController`, una clase de la librería *Three20* que proporciona el código necesario para que la vista se muestre de esta manera.

Los campos de una crítica son tres: descripción, precio y valoración. De los cuales solo el último es opcional. Para implementar la vista de las estrellas, donde se señala la valoración que se le quiere dar al artículo se ha usado la clase `SCRRatingView` en la que se implementa una vista que recibe una serie de imágenes para los distintos estados de las estrellas y almacena la valoración actual que el usuario ha seleccionado.

Pantalla de búsqueda

La pantalla de búsqueda permite tanto las búsquedas de usuarios como las búsquedas de artículos. La búsqueda contiene una vista de tabla que muestra los resultados de la consulta.

Se ha implementado búsqueda instantánea, a medida que el usuario va introduciendo caracteres, se van realizando las consultas. No es necesario esperar a que el usuario pulse *"Buscar"*. De esta manera se reduce le tiempo de interacción, lo cual es beneficioso para el usuario final.

Con el objetivo de minimizar los tiempos de espera, en cada consulta sólo se solicitan al servidor los K primeros resultados de la búsqueda, siendo K una constante "pequeña" definida en el código. Si el usuario desea ver más resultados tan sólo tiene que desplazarse hasta el final de la lista de resultados y presionar *"Cargar más"*.

Pantalla de Opciones

Esta pantalla es muy simple. Permite tanto que el usuario seleccione la privacidad de su cuenta como la cancelación de la cuenta. Para implementar la selección de opciones de privacidad se hizo uso de una vista de tabla, de clase `TTTableViewController`, cuyas celdas son las distintas opciones.

Para dar de baja una cuenta se implementó una verificación de la identidad del usuario a través de la contraseña. Dicha contraseña se solicita al usuario a través de un cuadro de diálogo personalizado con un campo de texto.

5.1.3 Creación de los ficheros de traducción

Uno de los requisitos es hacer un programa fácilmente traducible a otros idiomas, esto es una premisa que se ha tenido en cuenta a la hora de la implementación. En todos los lugares donde aparece una cadena de texto que se vaya a mostrar en la interfaz en vez de poner directamente una cadena, se ha hecho una llamada a una función `NSLocalizedString(NSString* cadena, NSString* descripción)`. Esta llamada permite que cualquier cadena pasada sea tratada en primer lugar como una clave de un diccionario. Dicho diccionario es dependiente del lenguaje establecido en las opciones de la máquina donde el programa se ejecute. El diccionario en cuestión no es más que un fichero de texto donde cada entrada está formada de la siguiente forma:

```
"Clave" = "Valor de la clave";
```

De esta simple manera se consigue que todos los elementos de la interfaz puedan ser fácilmente traducibles. Si quisiéramos traducir la cadena `"Esto es un ejemplo"` tan solo habría que ir al fichero del lenguaje correspondiente y añadir una línea. En el caso de la traducción al inglés sería de la siguiente manera:

```
"Esto es un ejemplo" = "This is an example";
```

Habría que hacer esto con cada lenguaje que se deseara traducir. En este sistema se han creado los ficheros de traducción tanto para el lenguaje castellano como para el inglés. Además todos los errores del servidor, los cuales tienen un número asociado han sido traducidos a ambos idiomas usando como clave el número de error.

5.2 Pruebas

Esta sección describe el conjunto de pruebas realizadas. Se debe notar que las pruebas aquí descritas no son las únicas llevadas a cabo puesto que durante el desarrollo de la aplicación se han realizado incontables pruebas de funcionamiento que no podrían ser detalladas en este documento por razones de tiempo.

Las pruebas realizadas demuestran que todos los requisitos de software se han cumplido. Esta afirmación queda reflejada en la matriz de trazabilidad al final de esta sección.

Todas las pruebas aquí expuestas han sido superadas exitosamente. A continuación se muestra la plantilla de definición de pruebas.

Identificador	Nombre
Descripción	*Descripción textual de la prueba*

Tabla 60. Plantilla para la definición de una prueba

5.2.1 Pruebas de aceptación

PA-01	Comprobación de registro a través de login
Descripción	Inserción de los datos de registro de un nuevo usuario y comprobar mediante el login que efectivamente el usuario ha sido creado.

Tabla 61. P-01

PA-02	Comprobación de cancelación de cuenta
Descripción	Borrar de un usuario existente a través de la interfaz e intentar acceder otra vez a ese usuario por medio de la interfaz de login.

Tabla 62. P-02

PA-03	Comprobación de login
Descripción	Introducir datos válidos de un usuario registrado en el login y comprobar que se inicia sesión correctamente.

Tabla 63. P-02

PA-04	Cierre de sesión
Descripción	Una vez iniciada sesión, en el menú principal presionar el botón "*Cuenta*" y luego *"Salir"* . Verificar que se accede a la pantalla de inicio.

Tabla 64. P-02

PA-05	Crear un nuevo artículo
Descripción	Una vez iniciada sesión, en el menú principal presionar el botón "*Nuevo*" e introducir los datos necesarios para un nuevo artículo: aportar una foto desde la cámara, un título y una valoración. Finalmente pulsar el botón de nuevo artículo. Verificar que la subida al servidor es correcta accediendo a la pantalla de artículos del usuario y viendo que el artículo creado aparece en el primer lugar de la lista.

Tabla 65. P-02

PA-06	Acceder a la información del perfil
Descripción	Una vez iniciada sesión, acceder a la pantalla de perfil a través del botón con dicho nombre y verificar que los datos son los del usuario que ha iniciado sesión.

Tabla 66. P-02

PA-07	Comprobación de la búsqueda de usuarios
Descripción	Una vez iniciada sesión, acceder a la pantalla de búsqueda e introducir el nombre del usuario que ha iniciado sesión en el campo de búsqueda. Verificar que la lista de usuarios mostrada coincide con el criterio de búsqueda.

Tabla 67. P-02

PA-08	Comprobación de la función seguir usuario
Descripción	Una vez iniciada sesión, realizar la búsqueda de un usuario al que no se esté siguiendo. Entrar a su perfil y presionar el botón *Seguir*. El resultado esperado es que la pantalla de información de seguidos se actualice y diga que se ha realizado una petición de seguimiento hacia ese usuario.

Tabla 68. P-02

PA-09	Comprobación de la función mostrar seguidos
Descripción	Tras realización de la prueba *Comprobación de la función seguir usuario*, acceder a la pantalla de perfil y desde ahí presionar el botón *"Seguidos"*. Verificar que se muestra la lista de seguidos correspondiente incluyendo el usuario recientemente añadido.

Tabla 69. P-02

PA-10	Modificación de opciones de privacidad
Descripción	Se debe partir de un determinado usuario A y un determinado usuario B ambos sin ser contactos. Acceder al sistema con el usuario B. Comprobar que al acceder al perfil de A, se puede ver su información. Luego acceder al sistema con A, modificar las opciones de perfil a *Privado* y realizar la operación anterior. Verificar que los datos de A (todos excepto su nick) no son mostrados en el perfil.

Tabla 70. P-02

PA-11	Comprobación de la búsqueda de artículos
Descripción	Tras la realización de la prueba *Crear un nuevo artículo* ir a la pantalla de búsqueda y seleccionar búsqueda de artículos. Introducir todo o parte del título del artículo creado y verificar que se muestra en pantalla en la lista de resultados.

Tabla 71. P-02

PA-12	Comprobación de la función de favoritos
Descripción	Tras la realización de la prueba *Crear un nuevo artículo* ir a la pantalla de detalles de un artículo y presionar el botón *"Añadir a Favoritos*". Hecho esto ir a la pantalla de perfil y presionar el botón de *"Favoritos"*. Verificar que el artículo previamente añadido se muestra en la lista de favoritos.

Tabla 72. P-02

PA-13	Valoración de un artículo
Descripción	Realizar una búsqueda de un determinado artículo, entrar en la pantalla de detalles de artículo y presionar el botón *Críticas*. Desde la pantalla de críticas, presionar el botón *"+"* que aparece en la esquina superior derecha. En esa pantalla elaborar una valoración y presionar enviar. Verificar que la valoración es mostrada en la pantalla de valoraciones.

Tabla 73. P-02

PA-14	Comprobación de la función de categorías
Descripción	Tras acceder al sistema, desde el menú principal acceder a la pantalla de categorías, seleccionar una determinada categoría. Verificar que se muestra una lista de categorías y se puede acceder a los artículos de cada una de ellas.

Tabla 74. P-02

PA-15	Comprobación de la función de ranking
Descripción	Tras acceder al sistema, desde el menú principal acceder a la pantalla de *Ranking*. Verificar que se muestran el top de los 10 artículos más votados.

Tabla 75. P-02

PA-16	Comprobación del idioma del sistema
Descripción	Disponiendo de un dispositivo iPhone (o simulador) cuyas opciones de lenguaje estén fijadas en Castellano, realizar accesos a todas las interfaces del sistema y comprobar que los textos están en castellano.

Tabla 76. P-02

PA-17	Comprobación de la velocidad de respuesta
Descripción	Realizar búsquedas de artículos con una letra o una frase corta y asegurarse de que el tiempo de respuesta es inferior a 10 segundos en al menos el 90% de los casos.

Tabla 77. P-02

PA-18	Comprobación de la velocidad de respuesta
Descripción	Realizar búsquedas de artículos con una letra o una frase corta y asegurarse de que el tiempo de respuesta es inferior a 10 segundos en al menos el 90% de los casos.

Tabla 78. P-02

PA-19	Comprobación del funcionamiento en un iPhone3G
Descripción	Instalar la aplicación en un dispositivo iPhone3G y realizar un uso típico de la aplicación. Verificar que la aplicación funciona correctamente.

Tabla 79. P-02

PA-20	Comprobación del funcionamiento en un iPhone3G
Descripción	Instalar la aplicación en un dispositivo iPhone3G y realizar un uso típico de la aplicación. Verificar que la aplicación funciona correctamente.

Tabla 80. P-02

PA-21	Comprobación del funcionamiento del sistema de traducción
Descripción	Localizar los ficheros de traducción en el código fuente y añadir un nuevo fichero para un lenguaje no traducido. Evaluar el tiempo que toma añadir nuevos idiomas al sistema. Dicho tiempo debe ser del orden de minutos.

Tabla 81. P-02

5.3 Matriz de trazabilidad

En esta sección se muestra la matriz de trazabilidad de requisitos de software frente a las pruebas realizadas.

	P-01	P-02	P-03	P-04	P-05	P-06	P-07	P-08	P-09	P-10	P-11	P-12	P-13	P-14	P-15	P-16	P-17	P-18	P-19	P-20	P-21
RSF-01	●																				
RSF-02		●																			
RSF-03	●	●	●																		
RSF-04				●																	
RSF-05					●																
RSF-06						●															
RSF-07							●														
RSF-08								●													
RSF-09									●												
RSF-10								●													
RSF-11										●											
RSF-12											●										
RSF-13												●									
RSF-14													●								
RSF-15													●								
RSF-16														●							
RSF-17															●						
RSNF-01																●					
RSNF-02																	●				
RSNF-03																		●			
RSNF-04																			●		
RSNF-05																				●	
RSNF-06																					●

Tabla 82. Matriz de trazabilidad entre Requisitos Software y Pruebas

Como se puede apreciar en la matriz, todos los requisitos de software propuestos en el análisis quedan cubiertos por las pruebas realizadas. Se debe notar que se han realizado pruebas para constatar que la funcionalidad es completa pero es necesario un posterior testeo mucho más profundo cuando se vaya a sacar el sistema al mercado.

Capítulo 6

Conclusión y líneas futuras

En este capítulo se explicarán las conclusiones que se ha sacado en claro del desarrollo del presente proyecto tanto positivas como negativas. Además se realizará un desglose del presupuesto. Por último se destacará una serie de aspectos a mejorar con objeto de una futura implementación.

6.1 Conclusión

La presencia de las redes sociales en nuestra vida cotidiana han modificado la forma en que nos comunicamos. Al combinar las tecnologías de los dispositivos móviles con las redes sociales se consigue permanecer comunicado en cualquier momento y lugar. En el presente proyecto se ha intentado usar ambas tecnologías para crear una nueva tendencia en el mercado de las redes sociales.

Debido al crecimiento exponencial de los teléfonos inteligentes, el aprendizaje del desarrollo de aplicaciones para dispositivos móviles puede suponer una ventaja laboral en el futuro. Además, el popular iPad está basado en

iOS por lo que los conocimientos adquiridos pueden emplearse para el desarrollo en este dispositivo.

El desarrollo de este Proyecto de Fin de Carrera ha supuesto dos problemas destacados. El primero de ellos es el diseño de interfaces sencillas y atractivas, que incrementen el número de clientes de la aplicación. El segundo de ellos es puesta en marcha del proyecto sobre un entorno real.

El primer problema ha supuesto marcar nuevos objetivos como informáticos. A lo largo de la carrera en escasas ocasiones vemos cómo se diseña una interfaz. En ningún caso nos enseñan cómo crear una interfaz atractiva para el usuario. El diseño de las interfaces ha requerido la comprensión y el manejo de las herramientas de diseño gráfico como el *Adobe Photoshop*. Además el diseño ha supuesto el estudio de manuales y ejemplos de interfaces. Entre los manuales se recalca el *Human Interface Guidelines*, un manual que provee de cierta información del diseño al que el usuario típico está acostumbrado.

El segundo problema implica la preparación de la aplicación para su despliegue a nivel global, es decir, para ser vendida y usada por infinidad de usuarios. Es siempre necesario controlar todos los posibles errores informando al usuario de los problemas pero sin saturarlo con excesiva información técnica. Para controlar todos los posibles casos se ha hecho hincapié en pruebas de aceptación con muchos usuarios, con registros y inicios de sesión con las más variadas combinaciones de caracteres, etc. Todo ello para intentar quebrar el correcto funcionamiento de la aplicación.

La planificación inicial se ha visto postergada debido a factores que han aparecido a lo largo del desarrollo. A medida que se iba aprendiendo el lenguaje de programación se iba cogiendo confianza y descubriendo nuevas librerías y métodos de hacer las cosas de una manera mejor. Se ha aprendido a desarrollar en un entorno nuevo y desconocido, a investigar mejores maneras de programar, a reciclar código… en definitiva, se han adquirido las habilidades necesarias para ser capaz de desarrollar un proyecto de manera eficaz.

Se ha partido desde cero y se ha pasado por todas las fases de desarrollo para concluir la creación del proyecto. Muchos de los conocimientos adquiridos a lo largo de la carrera se han puesto de manifiesto al afrontar su desarrollo. Por tanto cabe decir que es una gran satisfacción ver cómo una idea creada por un grupo de amigos puede llevarse a cabo si se trabaja con dedicación y con empeño.

6.2 Presupuesto

En esta sección se hará un desglose del coste de los elementos necesarios en este proyecto. Dichos elementos incluyen costes de personal, de equipo y costes de tecnologías.

6.2.1 Costes de personal

Para el cálculo de los costes de personal se ha evaluado el tiempo dedicado al proyecto. Para ello se tienen en cuenta la siguiente información:

- El día de comienzo del proyecto se ha fijado en el día 20 de junio de 2011.
- El día de fin del proyecto se ha estimado como el día 10 de noviembre de 2011.
- Ha habido un descanso de un 10 días por vacaciones de verano desde el 21 de agosto hasta el 31 de agosto de 2011.
- El horario laboral ha sido establecido en 8 horas.

Por lo tanto vemos que el número total de días trabajados es de 132 días. Y por consiguiente el número de horas totales trabajadas es de **1.056 horas**. A continuación y debido a que en las distintas fases del proyecto se ha realizado una función diferente, se muestran los periodos de las distintas funciones realizadas:

Fase	Días	Horas
Análisis	10	80
Diseño	40	320
Implementación	60	480
Pruebas	10	80
Documentación	12	96
Total	**132**	**1056**

Tabla 83. Tiempo dedicado por fase

Los costes de personal se han calculado en base a una tabla dependiendo de la función realizada. A continuación se expone la tabla de coste por función:

Fase	Coste/hora	Horas	Coste total
Análisis	25	80	2.000
Diseño	30	320	9.600
Implementación	20	480	9.600
Pruebas	20	80	1.600
Documentación	22	96	2.112
Total	**132**	**1056**	**24.912**

Tabla 84. Coste por fase

Por lo tanto el coste de personal asciende a 24.912 euros.

6.2.2 Costes de hardware y software

Los costes de equipo comprenden tanto el hardware como el software empleado en el proyecto. A continuación se exponen los costes de hardware empleados:

Elemento	Cantidad	Coste (€)	Coste Total (€)
Macbook Pro 13”	1	1.290	1.290
iPhone 3G	1	349	349
Total			**1.639**

Tabla 85. Costes de hardware

Seguidamente se exhibe el conjunto de elementos software y sus costes:

Elemento	No. Licencias	Coste (€)	Coste Total (€)
Xcode 4	1	0[1]	0
Photoshop CS5	1	1.001,82	1.001,82
Inkscape	1	0[2]	0
Microsoft Project	1	0[3]	0
Microsoft Office	1	119	119
Total			**1.120,82**

Tabla 86. Costes de software

(1) Xcode 4 es gratis con la suscripción al Apple Developer Program. Véase Otros costes.

(2) Inkscape es un programa de código abierto

(3) Microsoft Project está disponible de manera gratuita para los estudiantes de la UC3M a través de la plataforma MSDNAA.

6.2.3 Otros costes

En esta sección se incluyen aquellos costes que no forman parte de equipo ni de personal.

Elemento	Cantidad	Coste (€)	Coste Total (€)
Apple Developer Program	1 (12 meses)	73,96	73,96
Total			**73,96**

Tabla 87. Costes de software

6.2.4 Total de costes

Una vez se dispone de todos los costes parciales se calcula el total de costes del proyecto.

Concepto	Coste
Coste de personal	24.912
Coste de hardware	1.639
Coste de software	1.120,82
Otros costes	73,96
Total	**27.745,78**
Beneficio 120%	**61.040,71**

Tabla 88. Coste total del proyecto

El coste total es de VEINTISIETE MIL SETECIENTOS CUARENTAYCINCO EUROS Y SETENTAYOCHO CÉNTIMOS (27.745,78 €)

6.3 Líneas futuras

En esta sección se detalla todas aquellas características que han sido planteadas como futuras inclusiones. Estas características mejorarían las prestaciones de la aplicación para una mejor aceptación en el mercado.

6.3.1 Uso del protocolo HTTPS

HTTPS es un protocolo de la capa de aplicación que permite transferencias de datos seguras a diferencia de HTTP, que no las permite. HTTPS utiliza el protocolo SSL para cifrar sus datos, provee de una comunicación confidencial y permite la autenticación y la integridad de los datos [25].

Ilustración 28. Ejemplo de web que usa HTTPS

Se hubiera querido incluir HTTPS como protocolo para las comunicaciones donde la seguridad fuera un factor crítico. Mensajes donde se envíe información sensible como la contraseña actualmente van codificados con una función hash SHA1. Para una primera fase beta es suficiente pero con vistas a integración en el mercado nacional o global, tal nivel de seguridad se quedaría corto.

Los motivos que llevaron a la inicial desestimación de la idea de usar HTTPS tuvieron que ver con el coste de un mantenimiento de un servidor con servicios HTTPS y que al tratarse de un proyecto que no se sabía si se iba finalmente a llevar a mercado, no merecía la pena pagar.

6.3.2 Buscar artículos cercanos

Una de las características actuales del sistema es que permita que un artículo sea localizado cuando se sube. Por lo tanto una característica deseable para una próxima versión es que se permita una búsqueda según la posición del artículo. Esta característica permitiría, por ejemplo, buscar los bares que te rodean y ver su valoración y/o sus comentarios.

Para implementarlo habría que usar el conjunto de clases `MapKit` para representar un mapa como se muestra en la imagen y en él definir los lugares donde se sitúan los artículos tras haberlos solicitado al servidor.

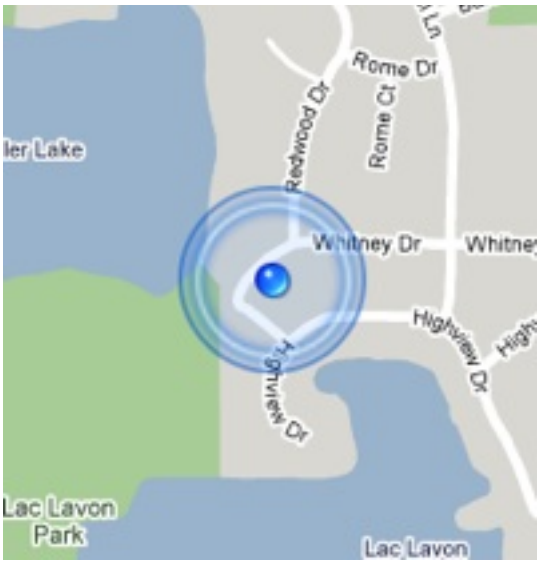


Ilustración 29. Ejemplo de uso de MapKit

6.3.3 Extensión al iPad

Esta Tablet de Apple tiene cada vez más usuarios, un objetivo a largo plazo sería la adaptación del código de iPhone al iPad. El iPad ofrece una pantalla mucho mayor por lo que la interfaz gráfica se vería considerablemente modificada.

Para implementar la extensión al iPad sería necesario rediseñar algunas interfaces ya que en la pantalla del iPad, las interfaces existentes quedarían

demasiado holgadas, sobraría demasiado espacio. También habría que cambiar el tipo de vista de algunas clases para adaptarla al tipo de vistas en el iPad, que suele ser una vista de la clase `SplitView` con una parte de la pantalla que hace de maestro y otra más grande que hace de esclavo. Este tipo de pantalla se muestra en la siguiente ilustración:

Ilustración 30. SplitView de iPad y ScrollView de iPhone e iPod Touch

6.3.4 Deslizar para opciones

Una característica que sería interesante implementar es la posibilidad de deslizar el dedo sobre un artículo para mostrar las opciones de éste. Con esto se consigue dar más posibilidades a la lista de artículos permitiendo mostrar los comentarios, valorar, o añadir a favoritos. Todo ello desde la lista de artículos. La idea se basa en lo que hace la aplicación Tweetie para mostrar las opciones de un *tweet*.

Ilustración 31. Vista de Tweets de Tweetie

Se pensó hacer esto desde el principio pero fue descartado debido a la complicación que suponía al no tener nociones del lenguaje ni de las librerías. Pero esta característica es cada vez más empleada en las aplicaciones para iPhone ya que permite aprovechar las características que este ofrece a la vez de eliminar la necesidad de elementos como botones de la pantalla, maximizando así la eficiencia de espacio.

Ahora cuando se ha comprendido en profundidad el lenguaje Objective-C y el conjunto de librerías y frameworks que componen iOS, se puede pensar en realizar esta característica ya que se implementaría mucho más rápidamente que al principio.

Acrónimos y abreviaturas

API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
CAPTCHA	Completely Automatic Public Turing Test to Tell Computers and Humans Apart
CSS	Cascading Style Sheet
GPS	Global Positioning System
HTTP	HyperText Transfer Protocol
HTTPS	HTTP Secure
IDE	Integrated Development Environment
IOS	IPhone Operative System
JSON	JavaScript Object Notation
MSDNAA	MicroSoft Developer Network Academic Alliance
MVC	Model-View-Controller
OS	Operative System
SSL	Secure Socket Layer

UC3M	Universidad Carlos Tercero de Madrid
URL	Uniform Resource Locator
XML	Extensible Markup Language
YAJL	Yet Another JSON Library

Referencias

[1] Número real de usuarios de Twitter. Ticbeat. Disponible [Internet]: http://www.ticbeat.com/socialmedia/numero-real-usuarios-twitter/ [12 de noviembre de 2011]

[2] iPhone. Wikipedia. 2011. Disponible [Internet]: <http://www.es.wikipedia.org/wiki/IPhone> [11 de noviembre de 2011]

[3] Top Smartphone OEMs. ComScore. 2011. Disponible [Internet]: <http://www.comscoredatamine.com/2011/07/top-smartphone-oems/> [11 de noviembre de 2011]

[4] Smartphone. Wikipedia. 2011. Disponible [Internet]: <http://en.wikipedia.org/wiki/Smartphone> [11 de noviembre de 2011]

[5] Smartphone Market Review. Gartner. 2011. Disponible [Internet]: <http://www.gartner.com/it/page.jsp?id=1764714> [11 de noviembre de 2011]

[6] iOS. Wikipedia. 2011. Disponible [Internet]: <http://en.wikipedia.org/wiki/iOS> [11 de noviembre de 2011]

[7] Cocoa Touch. Wikipedia. 2011. Disponible [Internet]: <http://en.wikipedia.org/wiki/Cocoa_Touch> [11 de noviembre de 2011]

[8] Cocoa Touch. Apple Developers. 2011. Disponible [Internet]: <http://developer.apple.com/technologies/ios/cocoa-touch.html> [11 de noviembre de 2011]

[9] JSON. Wikipedia. 2011. Disponible [Internet]: <http://es.wikipedia.org/wiki/JSON> [11 de noviembre de 2011]

[10] Introducción a JSON. 2011. Disponible [Internet]: <http://www.json.org/json-es.html> [24 de octubre de 2011]

[11] The fat-free alternative to XML. 2011. Disponible [Internet]: <http://www.json.org/xml.html> [24 de octubre de 2011]

[12] JSON Parser comparison. Cocoannetics. 2011. Disponible [Internet]; <http://www.cocoanetics.com/2011/03/json-versus-plist-the-ultimate-showdown/> [25 de octubre de 2011]

[13] JSONKit. Github. 2011. Disponible [Internet]: <https://github.com/johnezang/JSONKit> [24 de octubre de 2011]

[14] Xcode. Wikipedia. 2011. Disponible [Internet]: <http://en.wikipedia.org/wiki/Xcode> [25 de octubre de 2011]

[15] Red Social. Wikipedia. 2011. Disponible [Internet]: <http://es.wikipedia.org/wiki/Red_social> [26 de octubre de 2011]

[16] Seis grados de separación. Wikipedia. 2011. Disponible [Internet]: <http://es.wikipedia.org/wiki/Seis_grados_de_separaci%C3%B3n> [26 de octubre de 2011]

[17] Twitter. Wikipedia. 2011. Disponible [Internet]: <http://es.wikipedia.org/wiki/Twitter> [27 de octubre de 2011]

[18] Memcached. Twitter Blog. 2011. Disponible [Internet]: <http://engineering.twitter.com/2010/04/Memcached-spof-mystery.html> [27 de octubre de 2011]

[19] Ciao. Wikipedia. 2011. Disponible [Internet]: <http://es.wikipedia.org/wiki/Ciao_(Portal_de_internet)> [27 de octubre de 2011]

[20] Batalla entre redes sociales. El País. 2011. Disponible [Internet]: <http://www.elpais.com/articulo/reportajes/Batalla/redes/sociales/elpepuso cdmg/20110710elpdmgrep_6/Tes> [27 de octubre de 2011]

[21] MVC. Wikipedia. 2011. Disponible [Internet]: <http://es.wikipedia.org/wiki/Modelo_Vista_Controlador> [16 de noviembre de 2011]

[22] Cocoa Core Competencies: Model-View-Controller. Disponible [Internet]: <http://developer.apple.com/library/mac/#documentation/General/Conceptual/DevPedia-CocoaCore/MVC.html> [14 de noviembre de 2011]

[23] Human Interface Guidelines. Apple Developer. Disponible [Internet]: <http://developer.apple.com/library/ios/#documentation/userexperience

/conceptual/mobilehig/Introduction/Introduction.html> [24 de noviembre de 2011]

[24] User Interface Design Tips. Ambysoft. 2011. Disponible [Internet]: http://www.ambysoft.com/essays/userInterfaceDesign.html [24 de noviembre de 2011]

[25] HTTPS. Wikipedia. 2011. Disponible [Internet]: <http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol_Secure> [22 de noviembre de 2011]

[26] Conway, J., and Hillegass, A.: 'iPhone Programming' (Big Nerd Ranch, 2010, 1st edn.) ISBN-13 978-0321706249.

Anexo I

Manual de usuario

Este manual contiene toda la información necesaria para conocer y utilizar cada una de las funcionalidades y servicios que dispone la aplicación. Se ha organizado el manual en apartados en orden de uso para facilitar la lectura:

- Registro
- Inicio de sesión
- Menú principal
- Perfil
- Seguidos y seguidores
- Nuevo artículo
- Críticas y favoritos
- Detalles de artículo
- Comentarios
- Búsqueda de usuarios o artículos
- Opciones

Registro

La pantalla de registro muestra los campos necesarios para completar un nuevo registro y las condiciones de uso del sistema.

Los campos a rellenar son cuatro. Cada vez que se introduzca un valor en alguno de los campos se comprobará si es válido y se mostrará una 'V' en el caso de ser válido o una 'X' en otro caso tal y como se aprecia en la pantalla de la derecha. El campo de contraseña debe dispone de una barra de seguridad de la contraseña. La contraseña es más segura cuanto mayor sea la longitud de la barra.

Ilustración 32. Pantallas de registro sin y con datos de ejemplo

Si el usuario no entiende la imagen del CAPTCHA tiene la posibilidad de presionar el botón de recarga situado en la parte inferior para obtener un nueva imagen del servidor. Cuando los datos estén rellenos incluido el campo CAPTCHA, se deberá presionar el botón “Crear mi cuenta” y un mensaje de éxito o de fracaso en la creación de la cuenta será mostrado en pantalla.

Inicio de sesión

El inicio de sesión permite a un usuario registrado entrar en el sistema partiendo de sus datos.

Desde la pantalla de inicio de sesión se puede acceder a la pantalla de registro pulsando en la palabra *"Regístrate"*.

Cuando se haya rellenado tanto el nombre de usuario como la contraseña se debe pulsar "Acceder" para proceder a iniciar sesión. En caso de error en los credenciales introducidos se mostrará inmediatamente un cuadro de diálogo en pantalla.

Ilustración 33. Pantalla de inicio de sesión

Menú

La pantalla de menú principal es la raíz de las interfaces. Desde ella se accede al resto de pantallas del sistema. Es muy sencilla. Tiene seis botones, cada uno con un nombre y un icono explicativo de su utilidad.

En el menú también se incluye la posibilidad de hacer búsquedas pulsando en el botón en forma de lupa en la parte derecha de la barra de navegación. Además si se desea cerrar sesión y volver a la pantalla de inicio de sesión para iniciar sesión con otro usuario se tiene que presionar el botón "Cuenta" en la parte izquierda de la barra de navegación y luego "Cerrar sesión" en la lista desplegable.

Ilustración 34. Pantallas del menú

Perfil

Cada usuario tiene un perfil. Desde la pantalla de perfil se puede ver los datos de un usuario, así como la información de lo que ha publicado, a qué usuarios sigue y quienes le siguen.

La pantalla del perfil muestra la imagen del usuario, su nombre completo, su nombre de usuario, la ciudad dónde reside y su género. Además se muestran cuatro botones que indican el número de críticas realizadas, el número de artículos favoritos, los seguidos y los seguidores de el usuario. Al pulsar cualquiera de esos botones se abren nuevas pantallas donde se muestra lo que indica cada botón. Existe un botón en la pantalla y una barra de información que permiten seguir o dejar de seguir al usuario del cual se está viendo el perfil.

Ilustración 35. Pantallas del perfil

Seguidos y seguidores

La lista de seguidos y seguidores son listas que permiten acceder a los perfiles de los usuarios. Para acceder a la lista de seguidos hay un botón indicado en el menú con tal efecto. Además desde el perfil de un usuario se puede acceder tanto a la lista de seguidos como a la de seguidores. Al pulsar en un usuario de cualquier lista se abrirá su perfil.

Ilustración 36. Pantallas de seguidos y seguidores

Nuevo artículo

Para crear un nuevo artículo es necesario introducir rellenar como mínimo tres campos:

Hasta que estos campos no sean introducidos no se habilitará el botón de "Puntuar" y por tanto no se podrá cargar el artículo al servidor.

Ilustración 37. Pantallas de nuevo artículo (I)

Para introducir una imagen se debe presionar el botón con aspecto de cámara y seleccionar si se desea tomar la imagen desde la cámara o seleccionarla de entre las existentes en el dispositivo.

Para incluir la localización GPS en el artículo se debe pulsar el botón con el logotipo de un *geotag* situado en la parte inferior izquierda de la pantalla.

Ilustración 38. Pantallas de nuevo artículo (II)

Una vez se tienen todos los campos deseados se debe pulsar el botón *"Puntuar"* para subir el artículo al servidor. Cuando se esté subiendo el artículo se mostrará una barra de progreso indicando que el artículo está siendo subido. Se mostrará un mensaje al finalizar la subida o un mensaje de alerta en caso de error de conexión.

Críticas y favoritos

Cada usuario tiene una lista de críticas hechas sobre un artículo y de artículos favoritos. Sendas listas son accesibles desde el perfil. Cada elemento de la lista puede se presionado para ver su vista detallada.

Ilustración 39. Pantallas de críticas y favoritos

Además las listas de críticas y favoritos pueden ser actualizadas al intentar arrastrar para ver los artículos más recientes. En la imagen siguiente se muestra un ejemplo más claro:

Ilustración 40. Ejemplo de arrastrar y refrescar

Vista detallada del artículo

Al presionar un artículo en cualquier lista de artículos se pasará a la pantalla de vista detallada del artículo. En dicha vista se muestra información detallada del artículo así como una imagen ampliada y la posibilidad de marcar el artículo como favorito.

Ilustración 41. Pantallas de vista detallada de artículos

Estas pantallas muestran el precio del artículo, el usuario subió por primera vez el artículo, la fecha en la que lo hizo y el número de críticas que tiene el artículo.

Desde esta pantalla se puede acceder a las críticas realizadas sobre el artículo pulsando el botón *"Críticas"*.

Lista de críticas

Desde la pantalla comentarios se puede ver, modificar y añadir una crítica a un determinado artículo. Cada crítica tiene tres campos: valoración (de 1 a 5 estrellas), descripción (opcional) y precio (opcional).

Para añadir una nueva crítica basta con pulsar el botón en forma de '+' en la parte izquierda de la barra de navegación. Al pulsarlo se mostrará una vista con los tres campos mencionados. Una vez rellenados los campos es necesario pulsar el botón *"Enviar"* para que la nueva crítica sea cargada en el servidor.

Ilustración 42. Pantallas de lista de críticas y de nueva crítica

Búsqueda

Para acceder a la búsqueda, ya sea de usuarios o de artículos, debe hacerse desde el menú de la aplicación. En la pantalla de búsqueda se permite seleccionar si se desea buscar usuarios o artículos.

Para realizar una consulta solo es necesario empezar a escribir en el campo de búsqueda.

Ilustración 43. Pantallas de búsqueda de usuarios y de artículos

Opciones

En la pantalla de opciones se puede cambiar la privacidad de la cuenta del usuario así como darse de baja del sistema. Para cambiar la privacidad se debe seleccionar uno de:

- **"Público"**: todos los usuarios pueden ver su información del perfil.
- **"Seguidos"**: tanto seguidos como seguidores puede ver su perfil.
- **"Privado"**: sólo los usuario seguidos por usted pueden ver su perfil.

Como medida de seguridad, para dar de baja una cuenta es necesario verificar la contraseña del solicitante. Un diálogo será mostrado para introducir la contraseña del usuario y verificar que es él quien está solicitando la baja.

Ilustración 44. Pantallas de opciones