UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Diseño e implementación de una aplicación de gestión de incidencias para dispositivos móviles orientada a corporaciones municipales

TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Autor: Alejandro de los Ríos Real

Tutor: Xavier Molero Prieto

Curso 2016-2017

Resum

Mitjançant aquest projecte es planteja i es realitza el procés complet de disseny i implementació d'una aplicació mòbil per incidències a la via publica. També s'hi inclou el disseny de la part del servidor i un petit gestor d'incidències com a aplicació web per al control per part dels operaris.

Amb aquesta aplicació els ciutadans podran sentir-se més partícips en el procés de millorar, arreglar o difondre l'estat de la seua ciutat. Mitjançant un sistema de geolocalització es podran obrir incidències de desperfectes o deterioraments en la via pública, incloent-hi tant equipament com infraestructures urbanes, d'una forma ràpida i fàcil per als usuaris.

En aquesta memòria s'exposen les diferents fases de desenvolupament, que abasten des de l'estat de l'art fins a la seua implementació en Android, a més d'un conjunt de proves realitzades una vegada instal·lada en el terminal mòbil.

Paraules clau: Aplicació mòbil, incidències, via publica, Android

Resumen

Mediante este proyecto se plantea y se realiza el proceso completo de diseño e implementación de una aplicación móvil para incidencias en la vía pública. También se incluye el diseño de la parte del servidor y un pequeño gestor de incidencias como aplicación web para el control por parte de los operarios.

Con dicha aplicación los ciudadanos podrán sentirse más participes en el proceso de mejorar, arreglar o difundir el estado de su ciudad. Mediante un sistema de geolocalización se podrán abrir incidencias de desperfectos o deterioros en la vía pública, incluyendo tanto equipamiento como infraestructuras urbanas, de una forma rápida y fácil para los usuarios.

En esta memoria se exponen las diferentes fases de desarrollo, que incluyen desde el estado del arte hasta la implementación en Android de la misma, además de un conjunto de pruebas realizadas una vez instalada en el terminal móvil.

Palabras clave: Aplicación, incidencias, Android, vía pública, móvil

Abstract

In this project the complete process for the design and implementation of a mobile application for incidents on the street is proposed and developed. The design of the server part is also presented together with an incident management web application for user/operator control.

With this application citizens will feel more involved in the process of improving, arranging or spreading information about the state of their city. By means of a geolocation system it will be possible to open incidents of damages or deteriorations on the street, including those related to both equipment and urban infrastructures, in a fast and easy way for users.

In this report, the different stages of the development of the application, from the state of the art to its implementation in Android, are presented. A set of tests performed once the application had been installed in the mobile terminal are also included.

Key words: Mobile app, application, Android, public road, mobile, smartphone, issue

Índice general

Índice de figuras

Índice de tablas

CAPÍTULO 1
Introducción

A modo de definiciones previas se intenta aclarar primero que nada qué se sabe o entiende por un gestor de incidencias. Para ello se muestran a continuación unas citas [6] que reflejan una primera definición:

> La Gestión de Incidentes tiene como objetivo resolver cualquier incidente que cause una interrupción en el servicio de la manera más rápida y eficaz posible.
>
> (*Osiatis*, *ITIL – Gestión de Servicio TI, Visión General*)

Además de esa definición, también se puede leer la siguiente aclaración para obtener una definición más precisa aún:

> La Gestión de Incidentes no debe confundirse con la Gestión de Problemas, pues a diferencia de esta última, no se preocupa de encontrar y analizar las causas subyacentes a un determinado incidente sino exclusivamente a restaurar el servicio.
>
> (*Osiatis*, *ITIL – Gestión de Servicio TI, Visión General*)

En definitiva, los gestores de incidencias son los programas o aplicaciones que se encargan de administrar todos los errores, correcciones o desperfectos de un determinado sistema. En el caso de este proyecto, en vez de sistema, será la ciudad, en la cual las incidencias generan una reducción de la calidad de la vía pública.

La memoria del trabajo final de grado comienza exponiendo tanto la motivación generada para poder realizarlo como los objetivos esenciales que se quieren cumplir con ella. A continuación se describe como está estructurada la memoria y finalmente se explica brevemente el uso que se ha dado a la bibliografía.

1.1 Motivación

Tras un paseo por la ciudad y un hermano trabajando en el equipo de urbanismo de la ciudad de Cádiz, surge la idea de realizar una aplicación móvil. Con dicha aplicación se quiere conseguir un acercamiento del ciudadano a las instituciones de la ciudad, hacerlo participe y ayudante tanto del mantenimiento como de arreglos de desperfectos en la vía pública. No solo siendo parte de ella sino para hacer entre todos una ciudad mejor donde vivir.

Es sabido que hoy en día gran parte de los ayuntamientos están implementando y ofreciendo un nuevo servicio de transparencia y datos en abierto. De esta forma podría decirse que es el cambio de era, no una era de cambio, donde ahora la participación ciudadana cuenta más que nunca. Cada vez la ciudades son más grandes, mejores infraestructuras, mayor uso de las mismas y gracias a este tipo de aplicaciones se puede organizar un mantenimiento más eficiente y colaborativo.

1.2 Objetivos

Los objetivos principales de este proyecto son aprender a desarrollar una aplicación móvil capaz de comunicarse con un servidor mediante el servicio web REST y comprender todo el proceso de desarrollo desde el diseño hasta la implementación.

Más concretamente, los objetivos específicos del proyecto son:

- Crear una aplicación donde los usuarios puedan abrir incidencias.
- Desarrollar el servicio web encargado de gestionar las incidencias.
- Crear un pequeño cuadro de mandos para los operarios.
- Hacer una aplicación fácil de usar y que cumpla los principios de usabilidad.

Además de varios objetivos secundarios, como es la posibilidad de configurar el idioma de la aplicación, tener un perfil de usuario donde ver las incidencias creadas o la posibilidad de añadir datos de contacto.

1.3 Estructura de la memoria

En esta sección encontrarás una breve descripción de como ha sido estructurado la memoria del proyecto. Dicha memoria está dividida en seis capítulos, los cuales se describen a continuación:

* Capítulo 1, Introducción: Comienza la memoria con la descripción de las motivaciones por las que se ha realizado este proyecto, además de exponer los objetivos marcados que deben ser abarcados en el transcurso de la misma y una breve descripción de como esta estructurada.

* Capítulo 2, Situación actual: En este capítulo se realiza un estudio de las herramientas disponibles para llevar a cabo el desarrollo de la aplicación móvil. Posteriormente se muestra un análisis de las aplicaciones que se encuentran actualmente en el mercado, cuál es su fuerte y de que aspectos carece. Gracias a este análisis se puede saber por donde empezar el diseño de la aplicación.

* Capítulo 3, Diseño y arquitectura: Aquí se expone el diseño completo mediante capas de la aplicación desde el diseño conceptual hasta el diseño físico, pasando por el diseño lógico.

* Capítulo 4, Implementación: En esta parte se realiza la implementación de toda la aplicación hasta obtener un paquete APK funcional. Además del desarrollo de la parte servidor, la cual hace posible todo el funcionamiento de la aplicación.

* Capítulo 5, Pruebas y despliegue: Para finalizar la implementación se realizan las pruebas con usuarios reales, se redacta el presupuesto de todo el proyecto, se describe la documentación de la aplicación y se muestra una breve guía de instalación.

* Capítulo 7, Conclusiones: En esta última sección se exponen las conclusiones más relevantes que se han obtenido en la elaboración de este proyecto y posibles mejoras futuras.

Se adjuntan al final de la memoria varios anexos donde se podrá ver el código completo de la implementación junto con otras informaciones de interés.

1.4 Notas bibliográficas

En un primer estudio del estado del arte se han consultado fuentes oficiales de ayuntamientos, entre ellos el de Valencia, Guadalajara y Cádiz. También se ha buscando sobre empresas privadas que hayan desarrollado aplicaciones de este estilo. Se puede ver reflejado en las refencias [16, 15, 20].

Para el desarrollo de la aplicación móvil la mayor parte de la documentación consultada la componen las definiciones y formas de uso de las API [6, 7, 11]. Las fuentes oficiales de las mismas son, generalmente, documentos en línea. Por este motivo, una gran parte de la bibliografía esta compuesta por documentos web.

Incluyendo también las páginas oficiales de las herramientas utilizadas [17, 18] y de proyectos de código abierto como es Apache Cordova [19]. Todo ello queda reflejado en la bibliografía adjunta a esta memoria.

1.5 Siglas y acrónimos

OS: (*Operating System*) Sistema operativo, según la RAE [9] es un conjunto de órdenes y programas de un sistema informático que controlan todos los procesos básicos de una computadora y permiten el funcionamiento de los programas de aplicación de software.

IDE: (*Integrated Development Environment*) Un entorno de desarrollo integrado o entorno de desarrollo interactivo [5] es un programa informático diseñado para hacer más fácil y rápida las tareas de un programador o desarrollador de aplicaciones.

HTML: (*HyperText Markup Language*) Lenguaje de marcas de hipertexto [7], considerado el elemento de construcción más básico de una página web. Usado para crear y representar visualmente una página web determinando el contenido de la misma, pero no su funcionalidad.

CSS: (*Cascading Style Sheets*) Hojas de Estilo en Cascada [10, 12], es un simple mecanismo o lenguaje utilizado para describir cómo debe ser mostrado un documento en la pantalla, a la hora de imprimir, o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura. URL: (*Uniform Resource Locator*) Localizador de recursos uniforme, es un identificador recursos dentro de una red, como Internet, los cuales pueden cambiar con el tiempo mientras que la dirección es la misma. Muy utilizado en la *World Wide Web*.

JAR: (*Java ARchive*) Archivo java, es un paquete de archivos comprimido que contiene clases Java y recursos para ejecutar aplicaciones software o librerías de *Java platform*.

APK: (*Android Application Package*) Paquete de aplicación Android, es un paquete de archivos variante del formato JAR de Java y se usa para distribuir e instalar componentes empaquetados para la plataforma Android.

UML: (*Unified Modeling Language*) Lenguaje unificado de modelado, es un lenguaje de modelado para construir y documentar un sistema.

REST: (*REpresentational State Transfer*) Transferencia de Estado Representacional, según la wikipedia [8] es un estilo de arquitectura software para sistemas hipermedia distribuidos como la World Wide Web.

JSON: (*JavaScript Object Notation*) Notación de objetos JavaScript, es un formato de texto ligero utilizado para el intercambio de mensajes y datos. Es el formato de mensajes más utilizado, por encima de XML, llegando a considerarse como un lenguaje independiente.

XML: (*eXtensible Markup Language*) Lenguaje de marcas extensible, permite almacenar datos de forma legible lo que le hace ser muy útil en el intercambio de datos entre aplicaciones.

API: (*Application Programming Interface*) Interfaz de programación de aplicaciones, es una biblioteca que ofrece un conjunto de métodos o funciones capaces de ser llamados o utilizados por otra aplicación como una capa de abstracción.

CAPÍTULO 2
Situación actual

Antes de comenzar el proyecto se ha realizado un estudio amplio sobre el material que existe actualmente. Se ha analizado tanto herramientas para realizar el proyecto como aplicaciones que ya existen en el mercado.

2.1 Herramientas

Las herramientas, las cuales se han utilizado para la realización de este proyecto, se pueden encontrar tanto en versiones de pago como en su versión de prueba gratuita. En todo momento se ha utilizado las versiones de pruebas gratuitas ya que han sido suficiente para el uso requerido. Se detalla el motivo de su elección frente a otras que no han sido seleccionadas.

2.1.1. Android Studio

Android Studio [14] es el IDE oficial para el desarrollo de aplicaciones en la plataforma Android. Partiendo de la herramienta IntelliJ IDEA, Android Studio ofrece aún más características que mejoran su productividad en la construcción de aplicaciones Android, tales como:

- Un sistema de construcción basado en Gradle flexibles.
- Construir variantes y generación de archivos APK múltiples.
- Plantillas de código para ayudar a desarrollar características comunes.
- Un rico editor de diseño con soporte para la edición de arrastrar y soltar.
- Herramientas para capturar excepciones, medir rendimiento, facilidad de uso, compatibilidad de versiones, y otros problemas.
- Reducción de código con ProGuard y menores recursos con Gradle.

Considerada la mejor herramienta para desarrollar aplicaciones móviles nativas, pero a su vez, también requiere una gran curva de aprendizaje y grandes conocimientos de Java. Esto último ha sido la razón que nos ha llevado a descartar esta gran herramienta. Además al ser una herramienta de desarrollo de aplicaciones nativas se pierde la oportunidad de tener la característica multiplataforma, muy útil en estos tiempos donde hay variedad de sistemas operativos móviles en el mercado.

Figura 2.1: Logotipo de Android Studio

2.1.2. Apache Cordova

Apache Cordova, como lo describe la página web oficial [19], es un marco de desarrollo móvil de código abierto con el cual podemos utilizar las tecnologías web estándar (HTML5, CSS3 y JavaScript) para desarrollo multiplataforma. Una de las grandes ventajas de este entorno de trabajo es que no es necesario programar en lenguajes nativos de cada OS sino que nos da la posibilidad de desarrollar, como si fuera una página web, aplicaciones móviles para las siguientes plataformas:

- Amazon Fire OS.
- Android.
- BlackBerry 10.
- Firefox OS.
- iOS.
- Ubuntu.
- Windows Phone.
- Windows 8.
- Tizen.

Figura 2.2: Logotipo de Apache Cordova

Teniendo en cuenta su facilidad de uso y la posibilidad de aplicaciones multiplataforma, esta herramienta ha sido la seleccionada para desarrollar la aplicación móvil.

2.1.3. Visual Studio 2015

Visual Studio 2015 [21] es un IDE desarrollado por Microsoft para sistemas operativos Windows. Soporta múltiples lenguajes de programación tales como Visual Basic, Java, C++, PHP, .NET, Python, Ruby. Desde 2014, Microsoft anuncia el soporte para Apache Cordova dentro de Visual Studio y distribuye el paquete denominado «Tools for Apache Cordova».

Figura 2.3: Logotipo de Visual Studio 2015

Uno de los puntos fuertes de este entrono de desarrollo es la depuración avanzada de código JavaScript dentro del mismo, donde podemos inspeccionar los problemas de rendimiento, establecer puntos de interrupción, realizar un análisis de uso de memoria y otras tareas de depuración para Windows y Android. Además, incluye el simulador creado por el proyecto de código abierto Apache Ripple, podemos probar aplicaciones directamente en él o en un dispositivo físico conectado al equipo de desarrollo.

2.1.4. Apache Ripple

El simulador Apache Ripple se ejecuta como una aplicación web en el explorador Google Chrome. En Cordova, este simulador puede servir para simular la aplicación en una serie de dispositivos iOS y Android, y proporciona compatibilidad básica con los complementos principales de Cordova, como *Geolocation* y *Device Orientation*.

Figura 2.4: Logotipo de Apache Ripple y Cordova

Este simulador es especialmente útil para validar el diseño y el código CSS. Para lograr una emulación más fiel, la cual sea específica de una plataforma, se debe usar un dispositivo real.

2.1.5. JetBrains PhpStorm 2016

JetBrains PhpStorm 2016 es un IDE de programación desarrollado por JetBrains. Es uno de los entornos de programación más completos de la actualidad, permite editar código no sólo del lenguaje de programación php como lo indica su nombre.

Figura 2.5: Logotipo de PhpStorm 2016

Entre sus características principales se puede ver que permite la gestión de proyectos rápidamente, proporciona un fácil autocompletado de código, soporta el trabajo con PHP 5.5 y una sintaxis abreviada.

2.1.6. Visual Paradigm for UML

Visual Paradigm para UML es una herramienta para desarrollo de aplicaciones utilizando modelado UML como su nombre indica. Según su definición [17] y comentarios aportados por los usuarios es ideal para Arquitectos de sistemas, Analistas de Sistemas e Ingenieros de Software que estén interesados en la construcción de sistemas a gran escala. Dispone de un entorno gráfico para visualizar, especificar, construir y documentar el sistema.

Figura 2.6: Logotipo de Visual Paradigm for UML

Se ha elegido esta herramienta por su facilidad de uso. Aunque es una herramienta de pago, su versión de prueba es suficiente para el uso deseado en este proyecto.

2.1.7. WireframeSketcher

WireframeSketcher [18] es una herramienta que ayuda a crear rápidamente diagramas, maquetas y prototipos para aplicaciones de escritorio, web y móviles. Podemos encontrarla tanto como un *plug-in* para cualquier IDE basado en Eclipse como en su versión independiente. Con WireframeSketcher se puede reunir rápidamente la información útil de las partes interesadas, mostrar sus propuestas a los clientes, compartir sus ideas con otros desarrolladores y al final construir un software mejor.

Figura 2.7: Logotipo de WireframeSketcher

Se ha optado por esta herramienta por haber sido utilizada anteriormente en Desarrollo centrado en el usuario, además de ser simple y fácil de usar, su versión de prueba es totalmente funcional para un proyecto.

2.2 Aplicaciones en el mercado

Se ha llevado a cabo un estudio de las aplicaciones que ya existen en funcionamiento con características similares a la nuestra. Además de probarlas se indican posibles mejoras y errores que se han detectado.

2.2.1. Appvalencia

El ayuntamiento de Valencia pone a disposición de sus ciudadanos una aplicación móvil denominada Appvalencia [16]. Es una aplicación muy fluida pero sobrecargada, entre un sin fin de opciones y herramientas terminamos encontrando el apartado de «Incidencias en la vía pública» el cual podemos ver en la figura 2.8.

(a) Bienvenida **(b)** Menú incidencias

Figura 2.8: AppValencia

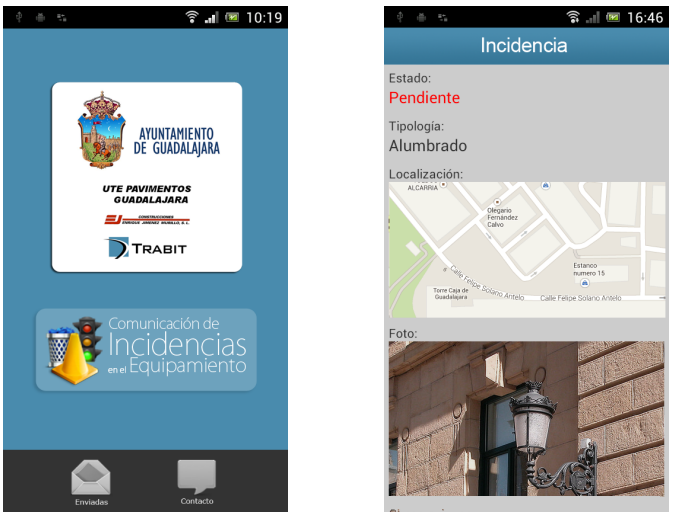
La conclusión principal es que no es una aplicación para abrir incidencias sino una aplicación general para ciudadano en la cual se encuentra escondida el apartado de incidencias en la vía pública. Citamos a continuación la descripción oficial encontrada en la página web del ayuntamiento de Valencia:

> El usuario podrá comunicar al Ayuntamiento incidencias en la vía pública (baldosas en mal estado, contenedores rotos, etc.), incluyendo fotografías georeferenciadas para que éste proceda a su reparación.
>
> (*Ajuntament de València*, *www.valencia.es*)

2.2.2. Haciendo ciudad Guadalajara

En esta ocasión es el Ayuntamiento de Guadalajara el que dispone de una aplicación íntegra para el mantenimiento urbano, denominada Haciendo ciudad Guadalajara [15]. Esta aplicación, mucho más simple y directa, se acerca más a la idea principal de nuestro proyecto. A su favor destacamos una interfaz simple y la posibilidad de obtener información puntual de las distintas fases del proceso de reparación.

Se detecta una falta de publicidad o promoción por parte del ayuntamiento, esto provoca que no sea muy conocida entre los ciudadanos. Citamos a continuación la descripción oficial de la aplicación dada por el ayuntamiento en Google Play:



(a) Bienvenida **(b)** Menú incidencias

Figura 2.9: Haciendo ciudad Guadalajara

La comunicación de incidencias en la vía pública a través de esta App es una iniciativa de la Concejalía de Servicios Municipales del Ayuntamiento de Guadalajara para hacer más rápida y eficaz la comunicación con los ciudadanos y de esta forma agilizar las labores de mantenimiento y reparación del equipamiento urbano.
Con esta aplicación, se puede comunicar cualquier incidencia detectada en la vía pública. Sólo hay que seleccionar una categoría entre las establecidas. Por ejemplo: acera deteriorada. A continuación, si se desea, se puede adjuntar una fotografía sobre el desperfecto detectado.
El dispositivo detectará las coordenadas exactas de la situación de la incidencia. También puede añadir comentarios sobre la misma.
Gracias a esta nueva aplicación, el Ayuntamiento localiza y repara los desperfectos de una forma más ágil. El ciudadano recibirá información puntual de las distintas fases del proceso de reparación.

(*Ayuntamiento de Guadalajara*, *www.guadalajara.es*)

2.2.3. ReparaCiudad

También existe una aplicación [20] llamada ReparaCiudad que no pertenece a ningún ayuntamiento ni institución pública. Esta aplicación es obra de Open Data Cities SL la cual se define como una empresa comprometida con los ciudadanos e intenta mejorar las administraciones públicas para conseguir una administración comprometida y participativa.

Citamos a continuación la descripción oficial de la aplicación dada por la empresa en Google Play:

(a) Bienvenida **(b)** Menú incidencias

Figura 2.10: ReparaCiudad de ODC

ReparaCiudad es una aplicación donde los ciudadanos pueden, de forma muy fácil y sencilla, a través del móvil o la web, reportar incidencias en la vía pública de manera que los ayuntamientos puedan reaccionar más rápidamente, focalizar en la incidencias que afectan y preocupan principalmente a los ciudadanos e informarlos de las soluciones adoptadas.

(*Open Data Cities SL*, *www.reparaciudad.com*)

CAPÍTULO 3
Análisis y diseño

Procedemos entonces a realizar un primer análisis de la aplicación que se desea llevar a cabo, entre ello los requerimientos funcionales que deben tener la aplicación móvil y web. Seguido del diseño de los diagramas y de las capas necesarias. Para finalizar se muestra un primer boceto de las interfaces y su análisis de usabilidad.

3.1 Diseño conceptual y lógico

En este apartado se considera que el diseño es independiente del teléfono móvil usado, del modelo y sistema gestor de bases de datos, etc. Simplemente se estudia el problema u objetivo a subsanar y se detallan las soluciones escogidas.

3.1.1. Requerimientos funcionales

Aplicación móvil

1. La aplicación debe permitir dar de alta una nueva incidencia así como consultar incidencias ya generadas.
2. Todas las incidencias tienen los siguientes posibles estados: pendiente de asignación, pendiente de resolución, iniciada, resuelta y reabierta.
3. Los usuarios solo podrán ver los estados: pendiente, iniciada o resuelta.
4. Al crear una incidencia se deberá rellenar siempre el campo de fotografía, teléfono y coordenadas GPS.
5. Los campos teléfono, coordenadas GPS y fecha de creación lo establecerá automáticamente la aplicación en el momento de crear la incidencia.
6. La aplicación registrará la incidencia mediante el *Web Services* alojado en el servidor.
7. Opcionalmente antes de crear la incidencia se podrá rellenar el campo de descripción.
8. La fotografía se subirá automáticamente al servidor renombrada con el Id de la incidencia.
9. Dispondrá de un perfil de usuario donde los usuarios pueden indicar sus datos de contacto.

10. Debe incluir la posibilidad de cambiar de idioma (Inglés, Valenciano y Español) en un menú de configuración.

Aplicación web para operarios

1. La aplicación debe mostrar todas las incidencias registradas.
2. Mostrará un mapa interactivo de incidencias pendientes de resolver.
3. Actualización automática de los puntos en el mapa.
4. Debe ser simple y fácil de entender, sin estar sobrecargada.
5. Posibilidad de comprimir todas las incidencias del día.

3.1.2. Diagrama de casos de uso

Todo el sistema se base en tres actores: usuario, operario y jefe de mantenimiento urbano. Los usuarios tendrán la posibilidad de generar nuevas incidencias y consultar el estado de sus incidencias comunicadas, mientras que los operarios serán los que modifiquen las incidencias según su estado. El jefe de mantenimiento urbano se encargará de supervisar a los operarios y tiene la posibilidad de reabrir una incidencia mal cerrada.

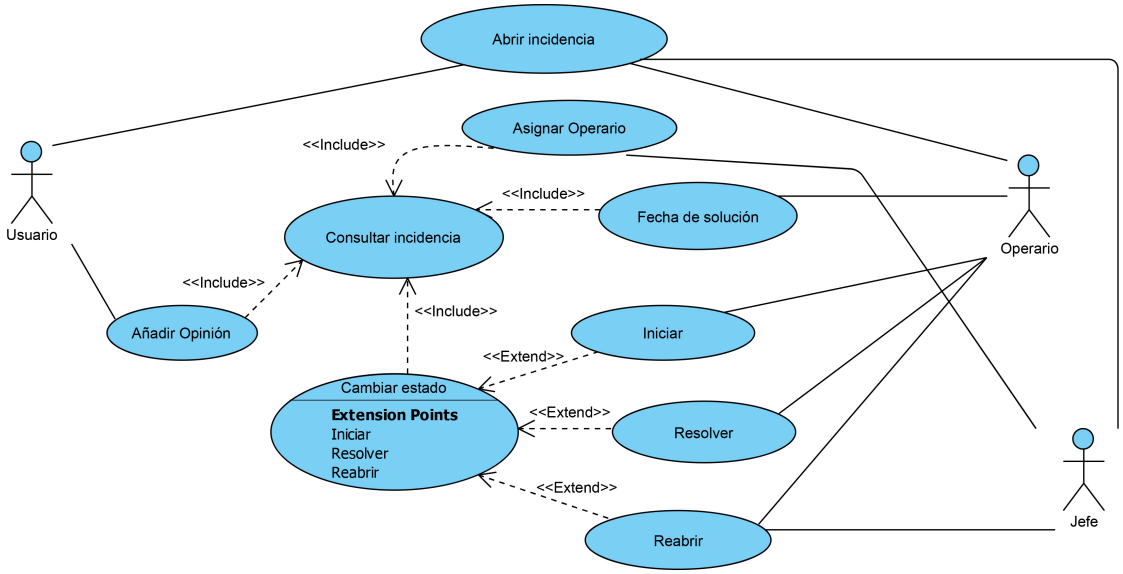


Figura 3.1: Casos de uso

3.1.3. Diagrama de estados

Una incidencia siempre deberá estar asociada a un estado y por consiguiente ese estado irá variando durante el ciclo de vida de la incidencia. En la figura 3.2 se observan los estados por los que pasará la incidencia además de las acciones que provocan la transición.

Una incidencia abierta pasa directamente a estar pendiente de asignación. Es el Jefe quien debe asignar a un operario en dicha incidencia, entonces es cuando pasará al estado pendiente de resolución. El operario será el encargado de modificar los siguientes estados (iniciada, resuelta) según los vaya alcanzando.

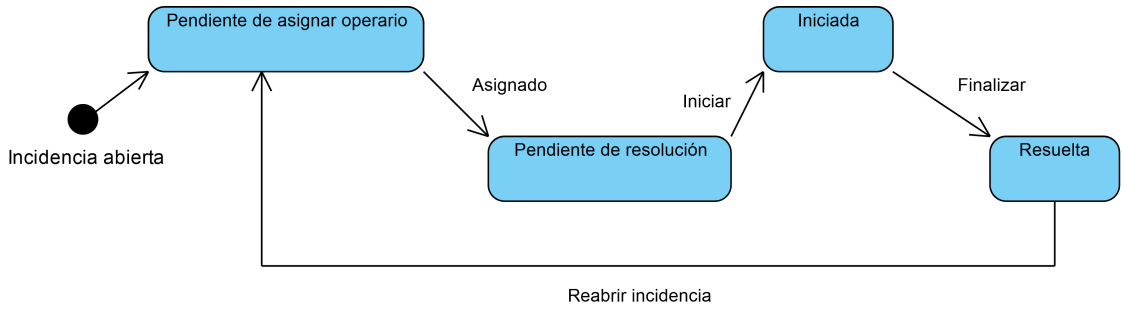


Figura 3.2: Diagrama de estados

3.1.4. Diagrama de capas

Para la implementación de la aplicación se utilizará una arquitectura por capas, de esta forma el desarrollo se puede llevar a cabo de forma independiente por niveles. Si es necesario añadir una modificación o una actualización de las aplicaciones será mucho más eficiente cambiando solo el código necesario de la capa oportuna.

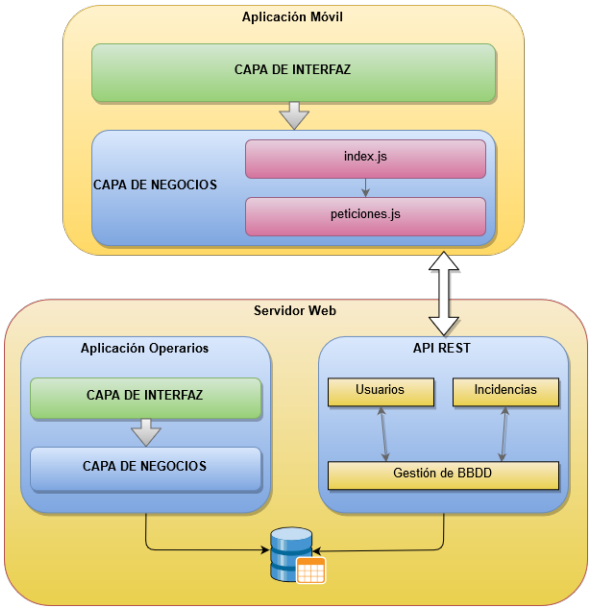


Figura 3.3: Diagrama de capas

Como se ha indicado la aplicación móvil será implementada como una página web, de este modo tenemos los archivos principales HTML y CSS en la capa de interfaz y por debajo se encuentran los archivos javascript que conforman la capa de negocios.

3.1.5. Base de datos: Modelo Entidad - Relación

Se ha diseño mediante UML el modelo Entidad-Relación, figura 3.4, de la base de datos, el cual representa de forma visual una abstracción de las tablas que formarán nuestra base de datos.

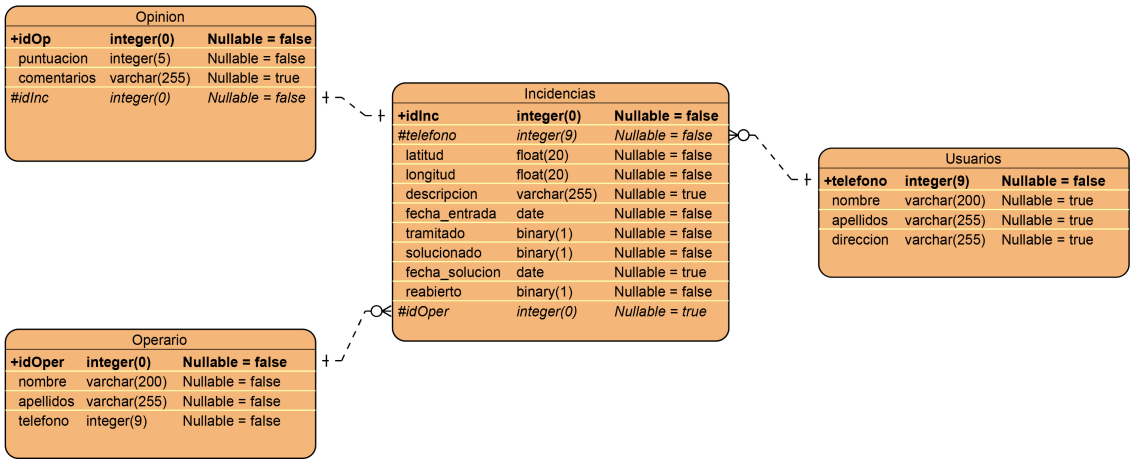


Figura 3.4: Modelo E-R

3.2 Interfaces

El contenido de esta sección muestra los esquemas de página o planos de pantalla confeccionados con la herramienta WireframeSketcher. Se ha dibujado el esqueleto o estructura visual de las pantallas y la posición del contenido que presentan al usuario, no se ha entrado en detalle en aspectos de estilos.

3.2.1. Interfaz móvil

Comenzando el diseño con la interfaz principal de la aplicación, figura 3.5. Esta pantalla se muestra nada más abrir la aplicación en el terminal móvil y permite directamente crear una incidencia, sin necesidad de navegar por submenús ni elegir categorías.

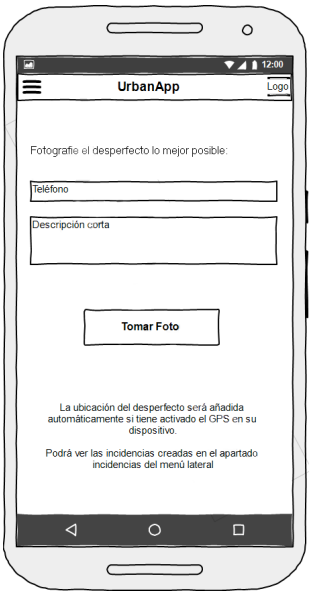


Figura 3.5: Pantalla de inicio

Se ha pensado este diseño teniendo en cuenta la facilidad de uso y la rapidez, de esta forma se puede considerar que será usada tanto por personas avanzadas como por personas con pocos conocimientos en el uso de los teléfonos inteligentes.

Dispone de un primer cuadro de texto que se rellena automáticamente con el número de teléfono, es además el identificador único que posee el usuario en la base de datos. Seguidamente muestra una segunda caja de texto, opcional, en la cual se podría escribir una breve descripción de la incidencia. Solo quedaría pulsar el botón «Tomar Foto» para completar la creación de una nueva incidencia en el sistema, automáticamente se abrirá la cámara del dispositivo para tomar la fotografía y comienza el proceso de registro y tramitación de la incidencia.

Pulsando en el botón de la esquina superior izquierda se abrirá un menú, figura 3.6, mediante un efecto de deslizado que muestra el logotipo y nombre corporativo con todas las opciones de la aplicación.

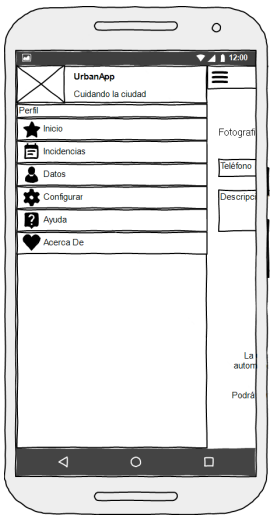


Figura 3.6: Pantalla «Menú lateral»

Desde este menú se tiene la posibilidad de navegar por las diferentes categorías o submenús que componen la aplicación. Cada botón incluye, además, un icono representativo de su contenido, así por ejemplo el apartado configuración muestra un icono de engranaje y el apartado de usuario muestra el icono de una persona.

En la pantalla incidencias, figura 3.7, se muestra una tabla actualizada según las incidencias creadas por ese usuario.

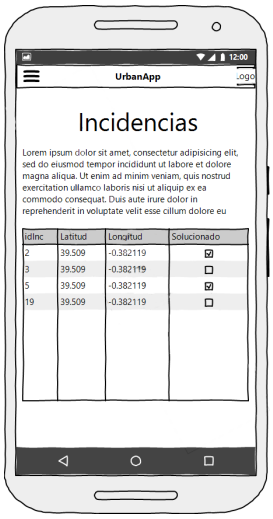


Figura 3.7: Pantalla «Incidencias»

Se ha reducido la información de la tabla para hacerla más llevadera y fácil de entender por el usuario, entre otros elementos se han eliminado la descripción, fecha de solución y variables de gestión de los operarios (tramitado, reabierto, etc).

La pantalla de datos de usuario, figura 3.8, recopila la información del perfil y la muestra en su correspondiente cuadro de texto. Desde esta ventana se tiene la posibilidad de modificar o agregar datos al perfil, confirmándolos al pulsar el botón guardar.

Figura 3.8: Pantalla «Datos de usuario»

Por último la pantalla de configuración, figura 3.9, en la cual se da la posibilidad de cambiar el idioma completo de la aplicación.

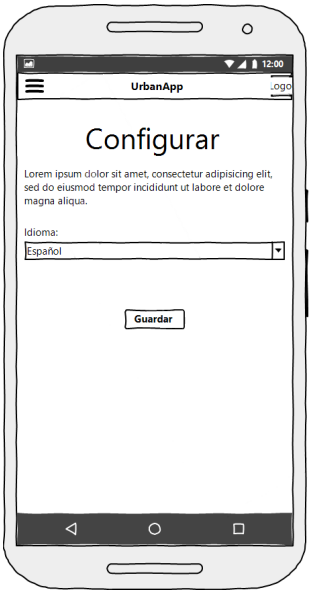


Figura 3.9: Pantalla «Configurar»

3.2.2. Interfaz WEB

La interfaz WEB, figura 3.10, muestra la aplicación que usarán los operarios para gestionar las incidencias creadas, esta pantalla será un tablero de instrumento o *dashboard* donde se puede administrar la información.

Dado que las incidencias están geolocalizadas se aprovecha esta información para dibujar un mapa donde se localizan todos los puntos registrados. De esta forma el operario puede tener una primera visión de que zonas de la ciudad están en peor estado o son deterioradas más habitualmente.

También muestra varias tablas que reflejan las incidencias registradas en la base de datos. Para mayor comodidad se separan las solucionadas de las que están en proceso de reparación quedando dos tablas claramente diferenciadas por colores.

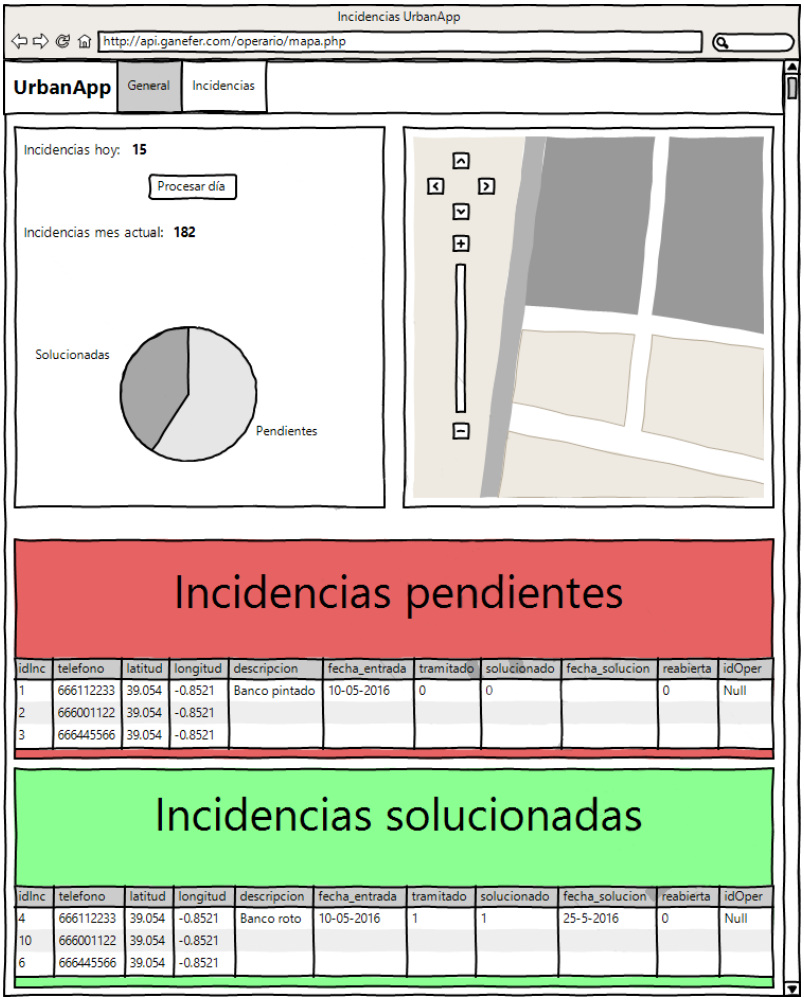


Figura 3.10: Pantalla de operarios

Con una barra de navegación en la parte superior se consigue una mayor velocidad al desplazarse por las secciones de la página. Incluye también el logotipo de la aplicación y su nombre.

3.3 Pruebas de usabilidad

Se entiende por usabilidad la cualidad que posee la página web o la aplicación móvil que son sencillos de usar porque facilitan la lectura de los textos y presentan funciones y

menús sencillos, por lo que el usuario encuentra satisfechas sus consultas y cómodo su uso.

Para analizar la usabilidad de los diseños mostrados anteriormente, se ha analizado si cumplen con los principios de usabilidad de Jakob Nielsen[1].

3.3.1. Visibilidad del estado del sistema

Este principio de usabilidad web indica que siempre se debe tener informado al usuario de lo que está pasando en la aplicación o página web y ofrecerle una respuesta en el menor tiempo posible.

Para cumplir este principio se ha añadido un mensaje tras la creación satisfactoria de una incidencia y procesos de carga con imágenes en movimiento en los apartados que suponen un mayor tiempo de espera.

3.3.2. Relación entre el mundo real y el sistema

La información tiene que mostrarse con un orden lógico y las imágenes o iconos usados tienen que ser claros, sin darle la posibilidad al usuario de equivocarse.

En este caso se han colocado iconos representativos en cada apartado del menú, seguido del nombre identificativo.

3.3.3. Control y libertad del usuario

Darle al usuario la posibilidad de subsanar un error y no sentirse frustrado por no poder realizar algo.

Se implementará en la aplicación un apartado para dar la posibilidad de editar un perfil personal.

3.3.4. Consistencia y estándares

Presentar un diseño de manera consistente, es decir, que la información que es similar aparezca siempre de la misma manera (mismas palabras, iconos y posición en la pantalla) y que la información que es diferente se exprese siempre diferente.

Se ha colocado el botón de menú de navegación en la esquina superior izquierda y se mantiene en esa posición en toda la aplicación, además siempre se mantiene el mismo estilo y forma de menú y cabecera.

3.3.5. Prevención de errores

Prevenir cualquier error que pueda cometer el usuario. Y dado el caso de que este cometa uno, tenemos que poner a su alcance todas las opciones posibles para poder corregirlo.

Dado esto se realizarán comprobaciones de campos de formularios en tiempo real, de esta forma se evita el error al escribir el número de teléfono.

[1]Jakob Nielsen es una de las personas más respetadas en el ámbito mundial sobre usabilidad en la web.

3.3.6. Reconocer antes que recordar

Siempre es mejor reconocer que obligar al usuario a memorizar acciones u objetos para que pueda cumplir su objetivo.

Todos los botones y menús de la aplicación cumplen los estándares de iconos y textos para que sean familiares a los usuarios.

3.3.7. Flexibilidad y eficiencia de uso

Disponer de una aplicación o sitio web preparado para todo tipo de usuarios, desde los más experimentados hasta los más novatos. Si cualquiera navega por la aplicación se consigue flexibilidad y si además dispone de opciones avanzadas se obtiene eficiencia.

Cualquier usuario puede crear una incidencia con tan solo hacer una fotografía, sin necesidad de descripción ni escribir dirección. Si el usuario busca algo más avanzado también dispone de un apartado para realizar el seguimiento de sus incidencias.

3.3.8. Diseño estético y minimalista

Las aplicaciones o páginas web no deben contener información innecesaria, distrae al usuario y puede llegar a molestar en la navegación. Elimina todo lo que consideres innecesario y que no aporta nada a lo que quieres decir.

El diseño utilizado para la aplicación es minimalista y se muestra solo y exclusivamente la información necesaria.

3.3.9. Ayuda a los usuarios a corregir los errores

Intentar que todos los errores que puedan ocurrir en la aplicación o página web estén expresados en un lenguaje entendible por todos, no solo por códigos. De esta forma se indica al usuario qué es lo que pasa en ese momento y que tiene que hacer para salir de ahí.

Cada error que se produzca en la aplicación mostrará una alerta con la descripción del problema acompañado del número de error.

3.3.10. Ayuda y documentación

Con estos principios se intenta siempre que el usuario no tenga que usar documentos de ayuda para poder navegar o utilizar una aplicación. Aun así, siempre tenemos que dar al usuario la posibilidad de tener un pequeño manual de funcionamiento.

Para cumplir con este principio se añade una categoría en la aplicación de ayuda para el usuario. Esta ayuda debe ser fácil de localizar, definir los pasos claramente y no ser muy extensa.

CAPÍTULO 4
Implementación

En este capítulo se describe el proceso completo de implementación tanto de la aplicación móvil como del servidor y su interfaz de operarios. Se añade a modo de ejemplo partes del código necesarios para la explicación.

4.1 Base de datos

El sistema de gestión de bases de datos utilizado para el proyecto es MariaDB, se ha utilizado este sistema debido a que viene impuesto por el servidor contratado. MariaDB, bajo licencia GPL, posee las mismas órdenes, interfaces, API y bibliotecas que MySQL lo que le otorga una alta compatibilidad. La administración de la base de datos se lleva a cabo mediante phpMyAdmin, también instalado en el servidor.

A continuación se muestran las sentencias MySQL utilizadas para la creación de las tablas y sus relaciones.

```
CREATE TABLE incidencias(
    idInc int NOT NULL AUTO_INCREMENT,
    telefono int(9) NOT NULL,
    latitud float(20) NOT NULL,
    longitud float(20) NOT NULL,
    descripcion varchar(250),
    fecha_entrada date NOT NULL,
    tramitado int NOT NULL DEFAULT '0',
    solucionado int NOT NULL DEFAULT '0',
    fecha_solucion date,
    reabierta int NOT NULL DEFAULT '0',
    idOper int,
    PRIMARY KEY(id)
)

CREATE TABLE usuarios(
    telefono int(9) NOT NULL,
    nombre varchar(200),
    apellidos varchar(250),
    direccion varchar(250),
    correo varchar(250),
    claveApi int,
    PRIMARY KEY(telefono)
    FOREIGN KEY (telefono) REFERENCES incidencias(telefono)
)

ALTER TABLE usuarios ADD CONSTRAINT ref_inc FOREIGN KEY (telefono) REFERENCES
    incidencias(telefono);
```

Listing 4.1: Instrucciones MySQL

4.2 Aplicación móvil

Las aplicaciones creadas con Apache Cordova tienen un archivo común denominado «config.xml» el cual proporciona información acerca de la aplicación y especifica los parámetros que afectan a cómo funciona.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <widget xml:lang="es" xmlns:gap="http://phonegap.com/ns/1.0" xmlns:cdv="http://
       cordova.apache.org/ns/1.0" xmlns:vs="http://schemas.microsoft.com/appx
       /2014/htmlapps" id="io.cordova.myappf0a025" version="1.0.0" xmlns="http://
       www.w3.org/ns/widgets" defaultlocale="es-ES">
3    <name>UrbanApp</name>
4    <description>Con UrbanApp podrás ayudar a mejorar tu ciudad, se un operario m
         ás y ayuda a informar de desperfectos en la vía pública</description>
5    <author email="aldelos@inf.upv.es">Alejandro de los Ríos Real</author>
6    <content src="index.html" />
7    <access origin="*" />
8    <vs:features />
9    [...]
10   <plugin name="cordova-plugin-camera" version="2.2.0" />
11   <plugin name="cordova-plugin-geolocation" version="2.2.0" />
12   <plugin name="cordova-plugin-compat" version="1.0.0" />
13   <plugin name="TelephoneNumber" value="com.simonmacdonald.cordova.plugins.
         TelephoneNumber" />
14   <plugin name="com.simonmacdonald.telephonenumber" version="1.0.0" />
15   <plugin name="cordova-plugin-file-transfer" version="1.6.0" />
16   <plugin name="cordova-plugin-file" version="4.3.0" />
17   <plugin name="cordova-plugin-device" version="1.1.3" />
18   <plugin name="cordova-plugin-dialogs" version="1.3.0" />
19 </widget>
```

Listing 4.2: Fragmento del archivo «config.xml»

Como cualquier XML comienza con la versión y el tipo de documento, destacamos sobre todo los datos importantes como son el nombre de la aplicación, su descripción y los datos del autor (Nombre, apellidos y correo electrónico). La linea 6 indica donde se encuentra el contenido de nuestra aplicación, en este caso «index.html ». Es decir, esta aplicación se implementa como una página web que hace referencia a archivos del tipo CSS, JavaScript, imágenes, multimedia u otros recursos que sean necesarios para que se ejecute de forma predeterminada.

Además se ha utilizado el *framework* denominado JQuery Mobile el cual proporciona un sistema de interfaz de usuario valido para todas las plataformas disponibles. Con esto se consigue una solución fácil, extensible y multiplataforma para diseñar la interfaz interactiva.

Al final se puede observar que se han ido añadiendo los *plug-ins* necesarios para que la aplicación funcione como se desea, entre ellos destacan los siguientes:

4.2.1. Cordova plugin compat

Este *plug-in* permite la compatibilidad con versiones anteriores de los complementos que debían actualizarse a los nuevos modelos de permisos de Android 6.0.0

4.2.2. Cordova plugin geolocation

Cordova dispone de un *plug-in* para obtener la localización mediante GPS del teléfono móvil indicándonos la latitud y longitud en coordenadas. A continuación se muestra el fragmento del código «index.js » encargado de realizar la llamada al GPS:

```
var geoID = null;
var options = { enableHighAccuracy: true };
geoID = navigator.geolocation.getCurrentPosition(function (position) {
    $("#geo").html(
        'Latitud: ' + position.coords.latitude + '<br />' +
        'Longitud: ' + position.coords.longitude
    );
    latitud = position.coords.latitude;
    longitud = position.coords.longitude;
}, onFail, options);
```

Listing 4.3: Fragmento localización de «index.js»

Con estos datos se genera la incidencia geolocalizada sin necesidad de que el usuario escriba una dirección. Todo de forma automática aporta una mayor rapidez y comodidad al usuario, lo que se traduce en mayor satisfacción con la aplicación.

4.2.3. Simon MacDonald plugin telephonenumber

Apache Cordova no facilita un *plug-in* para obtener el número de teléfono del terminal, debido a esto se ha obtenido un *plug-in* de Simon MacDonald[1] el cual facilita la tarea.

Para instalar el *plug-in* en el proyecto que lo requiera se deben seguir los siguientes pasos:

1. Copiar el archivo «www/telephonenumber.js» en la carpeta «www» del proyecto e incluir la siguiente referencia en el HTML depués de «cordova.js»:

```
<script type="text/javascript" charset="utf-8" src="cordova.js"></script>
<script type="text/javascript" charset="utf-8" src="telephonenumber.js">
    </script>
```

2. Crear un directorio dentro del proyecto llamado «src/com/simonmacdonald/cordova/plugins» y copiar el archivo «TelephoneNumber.java» dentro de él.

3. En el archivo «config.xml» se añade la siguiente línea como referencia:

```
<plugin name="TelephoneNumber"
    value="com.simonmacdonald.cordova.plugins.TelephoneNumber"/>
```

Una vez instalado, solo se tiene que llamar a las funciones que trae implementadas de la siguiente forma:

```
var telephoneNumber = cordova.require("cordova/plugin/telephonenumber");
        telephoneNumber.get(function (result) {
            if (result) {
                tlf = result;
                document.getElementById("tlfono").value = result;
            } else {
                showConfirm();
```

[1] Desarrollador de plugins para Apache Cordova y fundador del blog simonmacdonald.blogspot.com.es

```
            }
        }, function(error) {
            document.getElementById("test").innerHTML = "ERROR";
        });
```

Listing 4.4: Fragmento teléfono de "index.js"

Obteniendo así el número de teléfono en la variable «result» y comprobando que no es nula, si fuera el caso se llama al *plug-in* que activa el diálogo.

4.2.4. Cordova plugin dialogs

Una primera versión de la aplicación hacia uso de un *plug-in* para obtener el número de teléfono de la tarjeta SIM, de esta forma se registraría automáticamente todas las incidencias a ese número. Con las primeras pruebas empezaron a surgir los fallos, no obtenía ningún número y devolvía un valor *Null*, tras varios *test* y *debug* se pudo ver que no todos los teléfonos permiten el acceso a ese dato de la tarjeta SIM.

Para solventar este problema se decide hacer uso del *plug-in Dialogs* el cual permite crear un simple cuadro de diálogo solicitando el número de teléfono. Además se ha incluido una verificación de los caracteres introducidos para que no sea texto sin sentido.

```
function showConfirm() {
    navigator.notification.prompt(
        'Por favor, indica su número',
        onPrompt,
        'Número de teléfono',
        ['Enviar', 'Salir'],
        ''
    );
}

function onPrompt(results) {
    if (results.buttonIndex == 2) {
        navigator.app.exitApp();
    } else if (!(/^\d{9}$/.test(results.input1)) || results.input1 == 0) {
        alert("[ERROR] Teléfono mal escrito, vuelva a intentarlo.");
        showConfirm();
    } else {
        tlf = results.input1;
        document.getElementById("tlfono").value = results.input1;
    }
}
```

Listing 4.5: Fragmento diálogo de «index.js»

La función showConfirm() es la encargada de generar el diálogo y quedar a la espera del resultado. Con el resultado se realiza la llamada a onPrompt() que será la función encargada de evaluarlo y proseguir por el camino correcto.

4.2.5. Cordova plugin camera

Gracias a este *plug-in* se consigue acceso a la cámara del teléfono móvil. Solo debemos realizar las llamadas a los métodos para realizar las fotografías, a continuación se muestra el fragmento del código «peticiones.js» encargado de realizar la llamada a la cámara:

```
function capturePhoto() {
    navigator.camera.getPicture(onPhotoURISuccess, onFail, {
        quality: 50,
        destinationType: destinationType.FILE_URI
```

```
    });
}

function onPhotoURISuccess(imageURI) {
    var pImagen = document.getElementById('pintarImagen');
    pImagen.style.display = 'block';
    pImagen.src = imageURI;
    subirImagen(imageURI)
}

function onFail(message) {
    alert('ERROR: ' + message);
}
```

Listing 4.6: Fragmento cámara de «peticiones.js»

Mediante la función capturePhoto() se realiza la llamada a la cámara del dispositivo y se mantiene a la espera del resultado. Si el resultado es satisfactorio pasa a la función onPhotoURISuccess() donde se dibuja en la interfaz y se pasa como parámetro a la función de subir al servidor. En el caso de fallo se muestra un mensaje de alerta con el mensaje de error.

4.2.6. Cordova plugin file transfer

Todas las incidencias que se crean en la aplicación van acompañadas de una fotografía del desperfecto. Los datos que componen la imagen son pasados como parámetro a la función subirImagen() encargada de nómbrala con el ID de la incidencia y subirla al servidor.

```
function subirImagen(fileURL) {
    var opciones = new FileUploadOptions();
    opciones.fileKey = "imagen";
    opciones.fileName = idFoto + ".jpg";
    var ft = new FileTransfer();
    ft.upload(fileURL, encodeURI("http://api.ganefer.com/upload/upload.php"),
        uploadSuccess, uploadFail, opciones);
}

function uploadSuccess(r) {
    alert("Codigo = " + r.responseCode + " Respuesta = " + r.response + "
        Enviado = " + r.bytesSent);
}

function uploadFail(error) {
    alert("Ha ocurrido un error: Codigo = " + error.code + " upload error
        source " + error.source + " upload error target " + error.target);
}
```

Listing 4.7: Fragmento transferencia de «peticiones.js»

4.2.7. Cordova plugin globalization

Dado que la aplicación está destinada a las administraciones públicas se ha añadido la posibilidad de cambiar el idioma entre Español, Valenciano e Inglés. Además se facilita las posibles mejoras añadiendo más idiomas sin necesidad de modificar gran parte del código.

Para llevar a cabo el proceso de detección de idioma se ha utilizado el *plug-in Globalization* el cual nos permite saber el idioma del terminal e incluye una forma fácil de etiquetar los textos que deben ser traducidos.

```
navigator.globalization.getPreferredLanguage(
    function (locale) {
        cLANGUAGE = locale.value;
        languageControls(cLANGUAGE);
    },function () { cLANGUAGE = "es"; }
);

function languageControls(language) {
    if ((language.toString() == "en") || (language.toString() == "English") ||
        (language.toString().indexOf("en") != -1)) {
        languageSpecificURL = englishLanguageSpecificURL;
    } else if ((language.toString() == "va") || (language.toString() == "
        Valencia") || (language.toString().indexOf("va") != -1)) {
        languageSpecificURL = valenciaLanguageSpecificURL;
    } else {
        languageSpecificURL = spanishLanguageSpecificURL;
    }

    onNetworkCall(languageSpecificURL, function (msg) {
        languageSpecificObject = JSON.parse(msg);
        $(".languagespecificHTML").each(function () {
            $(this).html(languageSpecificObject.languageSpecifications[0][$(
                this).data("text")]);
        });
        $(".languageSpecificPlaceholder").each(function () {
            $(this).attr("placeholder", languageSpecificObject.
                languageSpecifications[0][$(this).data("text")]);
        });
        $(".languageSpecificValue").each(function () {
            $(this).attr("value", languageSpecificObject.languageSpecifications
                [0][$(this).data("text")]);
        });
    });
};

function onNetworkCall(urlToHit, successCallback) {
    $.ajax({
        type: "GET",
        url: urlToHit,
        dataType: "html",
        timeout: 30000,
    }).done(function (msg) {
        successCallback(msg);
    }).fail(function (jqXHR, textStatus, errorThrown) {
        alert("Internal Server Error: " + errorThrown);
    });
};
```

Listing 4.8: Fragmento globalización de «index.js»

Se puede observar en el código que se realiza una petición GET a un archivo de la aplicación. Estos archivos están en formato JSON y contienen los literales en los idiomas disponibles.

```
{
  "languageSpecifications": [
    {
      "acerca": "Acerca de",
      "apellidos": "Apellidos",
      "ayuda": "Ayuda",
```

```
7       "carInci": "Cargando incidencias...",
8       "configurar": "Configurar",
9       "correo": "eMail",
10      "datoUsu": "Datos de usuario",
11      "descripcion": "Descripción corta, máximo 60 caracteres",
12      "direccion": "Dirección",
13      "espere": "Espere, por favor...",
14      "foto": "Tomar foto",
15      "guardar": "Guardar",
16      "idioma": "Idioma:",
17      "incidencias": "Incidencias",
18      "inicio": "Inicio",
19      "nombre": "Nombre",
20      "pagAcerca": "Aplicación diseñada y desarrollada por Alejandro de los Rí
            os Real como Trabajo final de grado en la Universidad Politécnica de
            Valencia.",
21      "pagAyuda": "Bienvenido a la pagina de ayuda de UrbanApp.",
22      "pagConfig": "Configuración de la aplicación",
23      "pagInci": "A continuación se muestran las incidencias creadas desde su
            telefono:",
24      "pagUsu": "A continuación se muestra la información de perfil:",
25      "perfil": "Perfil",
26      "slogan": "Cuidando la ciudad",
27      "telefono": "Teléfono móvil"
28    }
29  ]
30 }
```

Listing 4.9: Archivo de idioma Español

Se dispone de tres archivos, uno para cada lenguaje, todos en formato JSON. De estos archivos se obtienen los textos que se deben mostrar en la aplicación, dependiendo del idioma seleccionado se procesa uno u otro JSON.

4.3 Servidor

La parte del servidor quedará en la nube disponible para resolver todas las peticiones que se soliciten, de esta forma además de obtener un sistema separado en Cliente-Servidor también se obtiene una alta escalabilidad del sistema. Para conseguir esta disponibilidad se ha desarrollado un servicio web RESTful capaz de procesar distintas peticiones.

4.3.1. Servicio Web RESTful

Un servicio web RESTful es una aplicación web que implementa la arquitectura REST o se basa en sus principios. Dio luz por primera vez en la Universidad de California para explicar un nuevo enfoque de la transferencia de recursos entre clientes y servidores a través de HTTP. Estandariza de forma amigable la transferencia de recursos entre aplicaciones enfocando el recurso como unidad fundamental del servicio web.

Todo lo relacionado con datos que interesen comunicar o enviar es considerado un recurso. Por ejemplo, puede referirse a un registro de la base de datos como lo sería cada usuario. De forma extendida se les llama colecciones a un conjunto de recursos del mismo tipo. Por lo que al obtener un conjunto de incidencias estaríamos hablando de una colección.

REST permite navegar a través de los recursos de forma intuitiva gracias a la utilización de formatos de URL. De esta forma permite navegar a través de ellos de forma intuitiva y más amigable a la vista de quienes deseen acceder a la API.

Los servicios web RESTful ofrecen distintos formatos para la comunicación de datos. Para este proyecto se ha decidido utilizar los más frecuentes como son JSON y XML, sin embargo pueden usarse distintas variedades como HTML, CSV, PDF, etc.

La cabecera «Content-Type» se basa en un registro estándar de tipos llamados MIME TYPES. Simplemente son valores de texto asociados a un significado predefinido por las regulaciones de la web para usarse como la manera correcta de representar datos.

Para JSON la cabecera se define con el siguiente tipo:

```
1 Content-Type: application/json
```

Y para XML con:

```
1 Content-Type: text/xml
```

Normalmente el formato se especifica en la cabecera de tipo de contenido, pero también es posible usarlo a través de parámetros en la URL.

En REST cada uno de los verbos que se usan en las peticiones equivale a una acción sobre un registro o colección. Las acciones más frecuentes se representan en la sigla CRUD (create, read, update y delete). Estas cuatro operaciones básicas tienen asignados los siguientes métodos (4.1):

Método	Acción	Seguro	Idempotente
GET	Obtiene un recurso o una colección	Si	Si
POST	Crea un nuevo recurso	No	No
PUT	Actualiza un recurso específico	No	Si
DELETE	Elimina un recurso específico	No	Si
PATCH	Actualiza parcialmente un recurso específico	No	No

Tabla 4.1: Métodos REST.

La columna «Seguro»indica si el método no es propenso a la alteración de datos, donde el único que cumple esta condición es GET debido a que solo obtiene recursos sin ningún cambio. Por otro lado, la creación, modificación y eliminación se basan en cambiar los recursos, así que son propensos a generar inconsistencias en la base de datos.

En la columna «Idempotente»se especifica la capacidad de un método para no repetir una misma acción. Por ejemplo, DELETE es idempotente ya que al eliminarse un recurso no es posible volverlo a hacer. En contraste, POST puede crear un nuevo recurso y si es llamado de nuevo, entonces creará otro, por lo que al utilizarlo sucesivamente siempre habrá un efecto.

REST utiliza los códigos de estado de HTTP para determinar si una respuesta fue exitosa o fallida dependiendo del inconveniente sucedido. Aunque existen gran cantidad de códigos de estados, estos pueden ser clasificados dependiendo del componente que generó el error. Los de la familia 2xx indican que la respuesta tuvo éxito, los de la familia 3xx indican que es necesaria una acción adicional para que el servidor complete la respuesta.

También existen los del conjunto 4xx para representar un error por parte del cliente y los 5xx para errores del servidor. La siguiente tabla (4.2) muestra algunos de los métodos más populares en un servicio web RESTful.

Código	Significado	Utilidad
200	OK	Especificar que un recurso o colección existe.
201	Created	Especificar que se creó un recurso. Se puede complementar con el *Location header* para especificar la URI hacia el nuevo recurso.
204	No Content	Resultado exitoso pero sin retorno de algún dato.
304	No Modified	Indica que un recurso o colección no ha cambiado.
401	Unauthorized	El cliente debe estar autorizado primero antes de realizar operaciones con los recursos.
404	Not Found	El recurso buscado no existe.
405	Method Not Allowed	Método relacionado a la URL no está permitido en la API.
422	Unprocessable Entity	Parámetros de la petición no están completos o su sintaxis es incorrecta para procesar la petición.
429	Too Many Requests	Excedido su número de peticiones si es que existe una política de límites.
500	Internal Server Error	Existe un error interno del servidor.
503	Service Unavailable	Servidor está temporalmente fuera de servicio.

Tabla 4.2: Códigos de estado REST.

4.3.2. Mensajes de comunicación

La comunicación entre la aplicación móvil y el servidor se realiza mediante mensajes JSON debido a su amplia adopción como alternativa a XML. La aplicación debe comunicarse con el servidor en ocasiones exactas lo que nos facilita la definición de mensajes concretos:

Crear incidencia:

```
{
    "telefono":"666112233",
    "latitud":"3152222",
    "longitud":"-0.999995",
    "descripcion":"Baldosa levantada"
}
```

Listing 4.10: Mensaje crear incidencia

Leer incidencia: Para recibir la información de una incidencia creada se ha tenido en cuenta que puede ser solicitada tanto por el usuario como por los operarios, de esta forma y para una mayor comodidad, la petición será mediante GET a una URL designada a cada incidencia.

```
api.ganefer.com/incidencias/[TELEFONO]
```

Listing 4.11: URL para leer incidencias

Crear, *loguear* y actualizar usuario: Se ha optado por realizar las tres acciones con un único mensaje, esto conlleva que un único método del servidor RestFul será el encargado de procesar el mensaje y decidir que debe hacer con él.

```
{
    "telefono":"666112233",
    "nombre":"prueba",
    "apellidos":"probado",
```

```
    "direccion":"calle alegria",
    "correo":"prueba@mail.com"
}
```

Listing 4.12: Mensaje crear loguear y actualizar usuario

Leer usuario: Leer la información almacenada es tan simple como indicar el número de teléfono.

```
{
    "telefono":"666112233"
}
```

Listing 4.13: Mensaje leer usuario

4.3.3. Seguridad y optimización

Cada incidencia generada lleva consigo una imagen identificativa que es subida al servidor. Estas imágenes se suben al directorio renombradas con el ID de la incidencia, de esta forma podemos acceder a ellas fácilmente sin necesidad de realizar una consulta en base de datos.

Dada la carga que debe soportar el servidor, con todas estas fotografías diarias, se ha desarrollado una función en PHP que comprime todas las imágenes de un día completo en un archivo ZIP. Este proceso se puede realizar de dos formas: el operario realiza el procesamiento de las imágenes desde el botón habilitado para ello en su cuadro de mandos o el *cron* del servidor realiza la tarea automáticamente todos los días a las 00:00 horas.

```
<?php

$fecha = date("dmy");  // Fecha del sistema
$zip = new ZipArchive(); // Creamos objeto ZIP

/**
 * Función encargada de recorrer el directorio y añadir los archivos al ZIP.
 * Si existen subdirectorios los leerá de forma recursiva.
 * @param string $directorio nombre del directorio
 * @param object $zip objeto zip
 */
function agregar_zip($directorio, $zip) {
    if (is_dir($directorio)) {  // Comprueba si es directorio
        if ($dir_abierto = opendir($directorio)) { // Abre el directorio
            while (($archivo = readdir($dir_abierto)) !== false) { // Bucle que recorreo el directorio
                if (is_dir($directorio . $archivo) && $archivo != "." && $archivo != "..") { // Llamada recursiva
                    agregar_zip($directorio . $archivo . "/", $zip);
                } elseif (is_file($directorio . $archivo) && $archivo != "." && $archivo != "..") { // Añade el archivo al ZIP
                    $zip->addFile($directorio . $archivo, $directorio . $archivo);
                }
            }
            closedir($dir_abierto); // Cierra el directorio
        }
    }
}

$dir = $fecha . "/"; // Directorio a comprimir
$rutaFinal = "procesados"; // Directorio final

```

```php
if(!file_exists($rutaFinal)){ // Comprueba si existe
    mkdir($rutaFinal); // sino lo crea
}

$archivoZip = $fecha . ".zip"; // Nombre del ZIP

if ($zip->open($archivoZip, ZIPARCHIVE::CREATE) === true) { // Abre el ZIP
    agregar_zip($dir, $zip); // Llamada a funcion
    $zip->close(); // Cierra el ZIP
    rename($archivoZip, "$rutaFinal/$archivoZip"); // Renombra y mueve el ZIP
    if (file_exists($rutaFinal. "/" . $archivoZip)) { // Comprobacion
        echo "Proceso Finalizado!";
    } else {
        echo "Error, archivo zip no ha sido creado!!";
    }
}
```

Listing 4.14: Función para comprimir imágenes

CAPÍTULO 5
Pruebas y despliegue

Por último, se ha calculado un presupuesto estimado del coste total y una vez desarrollada la versión final de la aplicación se ha sometido a un proceso de pruebas de funcionamiento y se han redactado la documentación y una breve guía de instalación.

5.1 Presupuesto

Para la creación del presupuesto, tabla 5.1, se han divido los costes en dos grupos o categorías distintas. Estas categorías son los costes del personal involucrado en el proyecto y los costes de las herramientas software y hardware.

Los costes indicados a continuación se expresan en euros y se tomarán dos decimales para las cantidades económicas, redondeando al alza cuando sea necesario.

Concepto	Mes 1	Mes 2	Mes 3	Cantidad	Precio	Total
Mano de obra						
Director	100 h	80 h	40 h	220 h	7.50 €	1,650.00 €
Analista desarrollador de Software	80 h	80 h	0	160 h	5.00 €	800.00 €
Hardware						
Depreciación de la computadora del desarrollador	80 h	80 h	40 h	200 h	0.50 €	100 €
Adquisición del servidor web				1	45 €	45 €
Software						
Apache Cordova	0	0	0	1	0	0
Visual Paradigm for UML	0	0	0	1	0	0
WireframeSketcher	0	0	0	1	0	0
PhpStorm IDE	0	0	0	1	0	0
					Sub Total	2,595.00 €
				% IVA	21	544.95 €
				Total Presupuesto		3,139.95 €

Tabla 5.1: Presupuesto del proyecto

5.2 Informe de errores

Se ha distribuido la aplicación entre familiares y amigos para darle un uso más parecido al de un usuario final, de esta forma también obtenemos, entre otras cosas, errores o faltas que hayan encontrado. Se detalla a continuación una lista con los datos obtenidos de las críticas constructivas:

1. Errores a la hora de obtener el número de teléfono automáticamente.
2. El menú izquierdo se abre correctamente desplazándose hasta su posición pero no se cierra entero automáticamente al arrastrar el dedo.
3. Fallos en la traducción de los botones de la aplicación.

5.3 Documentación

5.3.1. Instalación de la aplicación

Antes de comenzar con la instalación en el dispositivo se debe habilitar la opción de aceptar instalaciones de paquetes sin firmar por Google Play o también llamado orígenes desconocidos. Para ello, abrimos el menú de ajustes de nuestro dispositivo y nos dirigimos al apartado seguridad. En este nuevo menú que aparece se debe buscar y activar la opción denominada «Orígenes desconocidos», el sistema mostrará una alerta en pantalla advirtiendo de los peligros que conlleva instalar aplicaciones que no proceden de Google Play.

Una vez realizados los pasos anteriores ya tenemos el sistema preparado para la instalación. Descargamos o transferimos desde el ordenador el APK a nuestro dispositivo Android, nos dirigimos a la carpeta descargas y pulsamos sobre el APK descargado. Empezará el proceso de instalación mostrando un menú, figura 5.1, que contiene una descripción resumida de los permisos que requiere la aplicación UrbanApp. Aceptando estos permisos se instalará la aplicación y ya está disponible para su uso desde el menú principal.

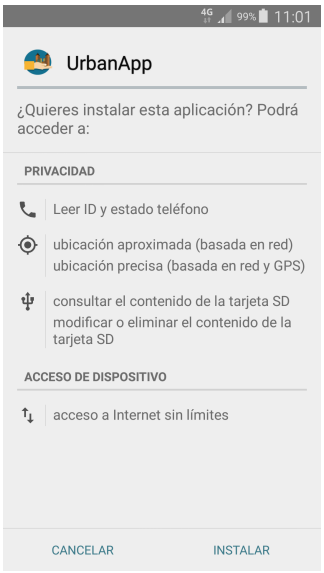


Figura 5.1: Permisos de la aplicación

5.3.2. Primeros pasos

Una vez instalada la aplicación se puede ejecutar pulsando el icono de la misma desde el menú principal del terminal. De primera imagen se mostrará el logotipo de la Universidad Politécnica de Valencia, figura 5.2, mientras realiza el cargando del arranque.

Figura 5.2: Logotipo de la Universidad

Acto seguido ya estará iniciada la aplicación y automáticamente intentará conseguir el número de teléfono de la tarjeta SIM insertada. Si este proceso falla o no consigue permisos para acceder a ese dato, mostrará un cuadro de diálogo, figura 5.3, solicitando al usuario que introduzca el número.

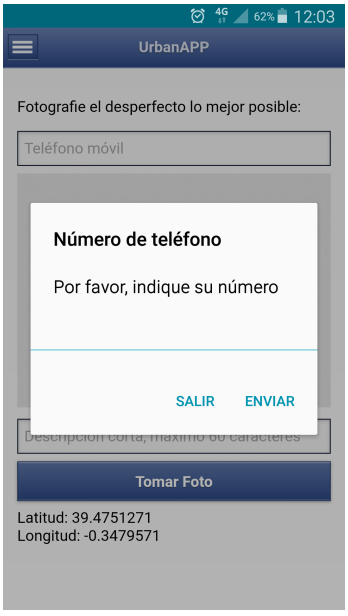


Figura 5.3: Solicitud del número de teléfono

En este momento ya se puede crear una nueva incidencia directamente, figura 5.4. El usuario tiene la opción de añadir además una descripción corta que acompañará a la

incidencia. Pulsando el botón «Tomar Foto» abrirá la cámara del dispositivo y una vez tomada la foto quedará registrada la incidencia automáticamente.

Figura 5.4: Menú principal

Se tiene acceso a un menú lateral, figura 5.5, mediante el cual se puede navegar por las distintas opciones de la aplicación. Entre ellas encontramos las opciones de seguimiento de incidencias creadas o la configuración de los datos de usuario o aplicación.

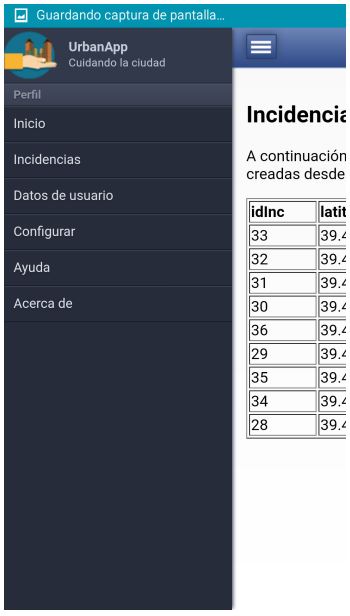


Figura 5.5: Menú lateral

En el menú «Incidencias» se puede obtener información relacionada con las incidencias creadas desde ese número de teléfono. Muestra una tabla indicando el identificador de la misma, sus coordenadas para localizarla y si ha sido solucionada o no, figura 5.6.

Dado el hecho de que la aplicación está orientada a administraciones públicas, se puede cambiar el idioma completo de la aplicación. Para ello debe abrirse el menú «Configurar» y una vez ahí cambiar el idioma por el que se desee, figura 5.7.

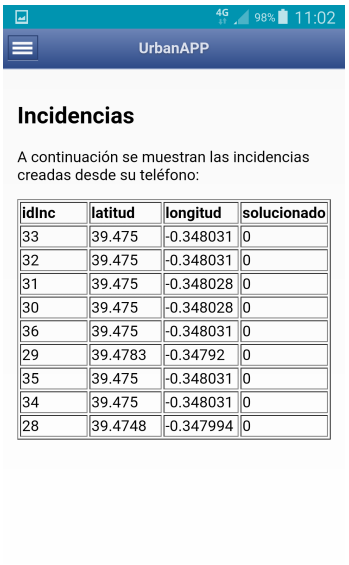


Figura 5.6: Menú incidencias

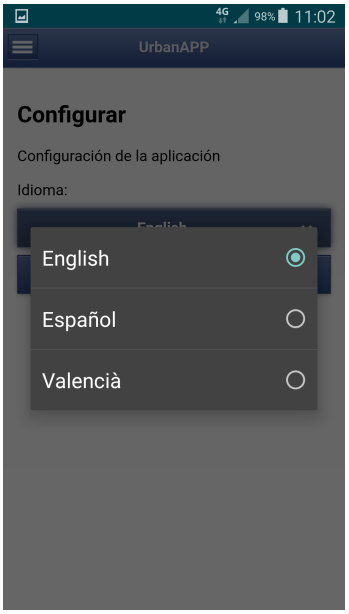


Figura 5.7: Configuración de idioma

CAPÍTULO 6
Conclusiones

El producto obtenido ha sido una aplicación móvil básica para la creación de incidencias por parte de los ciudadanos. Dicha aplicación reúne todos los requisitos de funcionalidad mínima necesaria que se requiere de este tipo de sistemas añadiendo una serie de características interesantes como son la asignación automática de las coordenadas GPS o la automatización del registro de la incidencia.

6.1 Análisis de los resultados

El trabajo desarrollado se ha cerrado con la creación de aproximadamente 900 líneas de código para la aplicación móvil y 1045 líneas de código para la parte del servidor, incluyendo API y aplicación para operarios. Sin contar las pruebas y tutoriales realizados para obtener experiencia con Apache Cordova, que dan forma al prototipo final.

En cuanto a los objetivos planteados al comienzo del proyecto, se puede afirmar que se han cumplido la mayoría e incluso se ha avanzado un poco más en el desarrollo, dejando así el camino libre para futuros proyectos o mejoras. Además se han completado también los objetivos secundarios como son el cuadro de mandos para los operarios y la comprobación del cumplimiento de los principios de usabilidad.

Cabe destacar que no eran muy extensos los conocimientos sobre programación en Android, de ahí que se haya utilizado una herramienta como Apache Cordova para facilitar el proceso. A lo largo de todo el proceso se ha ganado habilidad y soltura en cuanto a programación JavaScript, PHP, HTML y CSS.

También han surgido inconvenientes de gran envergadura que ha retrasado todo el proceso, entre los que destacan los tres más problemáticos. El primer problema encontrado fue la incompatibilidad de algunos teléfonos a la hora de conseguir el número de la tarjeta SIM, en un principio no se vio dicho problema hasta que no sucedió el segundo inconveniente: el terminal de las pruebas dejó de funcionar de la noche a la mañana, dejando todo el proyecto sin móvil físico con el que estudiar. Aumentaron los retrasos y se continuó mediante un emulador denominado «Ripple».

Cuando parecían que estaban solucionados los problemas, los cuales retrasaban el proyecto, surgió el mayor problema de todos. El ordenador portátil donde se estaba desarrollando el proyecto dejó de responder. Se salvaron los datos en una copia de seguridad y se procedió al formateo del equipo, perdiendo de nuevo varios días de trabajo.

6.2 Posibles mejoras

Partiendo de la base que se ha creado con este proyecto se pueden realizar nuevas mejoras para ampliar la funcionalidad, obteniendo un producto mucho más completo, con la finalidad de que pueda cubrir las necesidades de casi cualquier administración pública.

Gracias a la multiplataforma y la arquitectura software en capas empleada se ha conseguido un aumento sobre las plataformas en las que puede trabajar el producto en caso de que así se requiera y añadir funcionalidades a la capa de negocios sin necesidad de modificar todo el código. Por ejemplo, se ha mostrado la aplicación para sistemas Android pero no conlleva ninguna complicación mayor realizar la compilación para sistemas iOS.

Además se han estudiado las posibilidades de añadir nuevas mejoras tanto a la aplicación móvil como al servicio web. Unos ejemplos de estas mejoras serían:

- Gestión de usuarios y perfiles. Añadir la posibilidad de administrar los usuarios que se encuentran registrados o que han usado la aplicación. Esta mejora sería muy útil para estudios de estadísticas o para seguimiento del uso por parte de los ciudadanos.
- Gestión de operarios con distintos roles. En todos los equipos de trabajo existen diferentes roles para cada persona involucrada. Con esta mejora se añadiría la posibilidad de restringir el acceso a operarios que sean de menos nivel que otros.
- Diseño de la aplicación móvil. Mejorar todo el diseño visual de la aplicación móvil.
- Diseño de la aplicación web. Mejorar todo el diseño visual de la aplicación web para operarios.

Bibliografía

[1] Andy Harris. *Programación con PHP 6 y MySQL*. Anaya multimedia, Madrid, primera edición, 2009.

[2] Mario Rubiales Gómez. *HTML5, CSS3 y Javascript*. Anaya multimedia, Madrid, primera edición, 2013.

[3] Christopher Schmitt. *Curso de CSS*. ANAYA multimedia, Madrid, tercera edición, 2010.

[4] John M. Wargo. *Apache Cordova 4 Programming*. Addison Wesley Professional, English, primera edición, 2015.

[5] ¿Qué es un IDE? - Global Mentoring. Consultado en www.globalmentoring.com.mx/cursos-java/java-fundamentos/que-es-un-ide/.

[6] Definición de gestión de incidentes. Consultado en http://itil.osiatis.es.

[7] Definición de HTML - Qué es, Significado y Concepto. Consultado en http://definicion.de/html/.

[8] Descripción de REST en la Wikipedia. Consultado en https://es.wikipedia.org/wiki/Representational_State_Transfer.

[9] Diccionario de la lengua española por la RAE. Consultado en http://dle.rae.es.

[10] Documentación de CSS. Consultado en https://developer.mozilla.org/es/docs/Web/CSS.

[11] Documentación y API de PhoneGap. Consultado en http://docs.phonegap.com.

[12] Guía breve de CSS. Consultado en http://www.w3c.es/Divulgacion/GuiasBreves/HojasEstilo.

[13] Página web oficial del sistema Android. Consultado en https://www.android.com.

[14] Página web oficial de Android Studio. Consultado en https://developer.android.com/studio/index.html.

[15] Página web oficial del Ayuntamiento de Guadalajara. Consultado en http://www.guadalajara.es.

[16] Página web oficial del Ayuntamiento de Valencia, apartado AppValencia. Consultado en http://www.valencia.es.

[17] Página web oficial del programa Visual Paradigm for UML. Consultado en https://www.visual-paradigm.com.

[18] Página web oficial del programa WireframeSketcher. Consultado en http://www.wireframesketcher.com.

[19] Página web oficial del proyecto Apache Cordova. Consultado en https://cordova.apache.org.

[20] Página web oficial del proyecto ReparaCiudad de Open Data Cities SL. Consultado en http://www.reparaciudad.com.

[21] Página web oficial de Visual Studio. Consultado en https://www.visualstudio.com/es/.

APÉNDICE A
El emulador Ripple

Antes de comenzar a usar el emulador Ripple debemos instalar el navegador Google Chrome en nuestro ordenador. El instalador de Visual Studio no instala Chrome cuando instala Visual Studio Tools para Apache Cordova.

Elija Android o iOS de la lista Plataformas de solución, figura A.1. Si no ve la lista, seleccione Plataformas de solución de la lista Agregar o quitar botones para mostrarla.

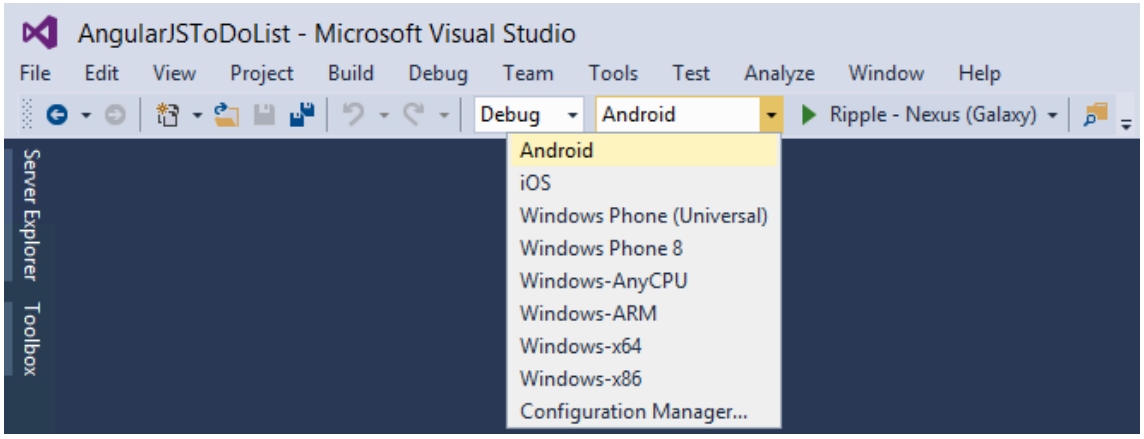


Figura A.1: Plataformas de solución

En la lista de dispositivos de destino, elija uno de los simuladores Ripple, figura A.2. Presione F5 para iniciar la depuración o Mayús+F5 para iniciar la aplicación sin depurar.

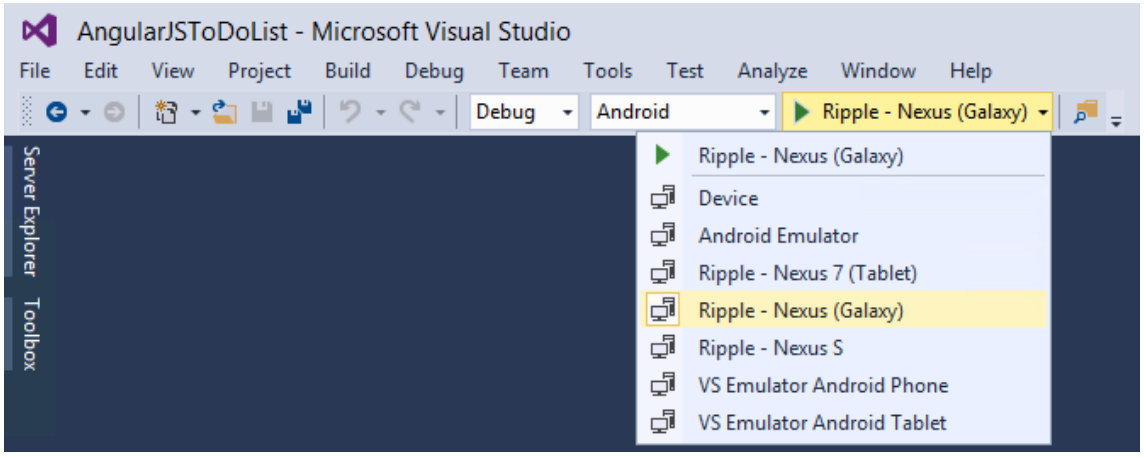


Figura A.2: Dispositivos de destino

Una vez iniciado el emulador se puede cambiar el dispositivo de destino sobre la marcha mientras la aplicación se ejecuta en Chrome. De esta forma se puede evitar tener que reiniciar la aplicación en Visual Studio.

Mientras la aplicación se ejecuta en Chrome, elija el botón de flecha a la izquierda y, a continuación, elija la sección de dispositivos. Seleccione el botón con el nombre del dispositivo actual y, a continuación, elija otro dispositivo, figura A.3.

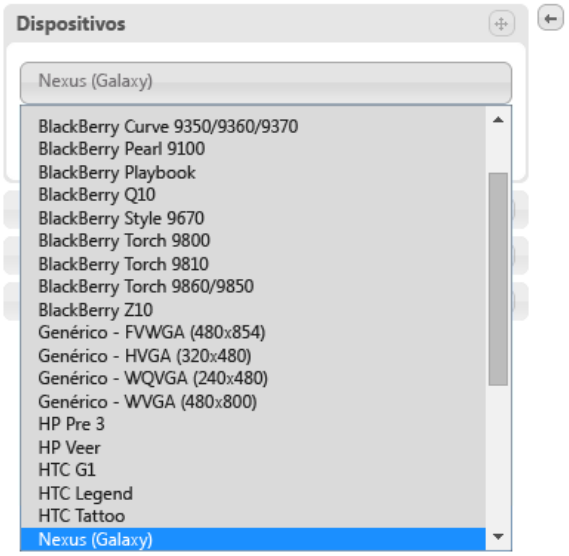


Figura A.3: Cambiar dispositivo

APÉNDICE B

Árbol de directorios

```
/
├── merges
│   ├── android
│   └── ..
├── plugins
│   ├── com.simonmacdonald.telephonenumber
│   │   └── ...
│   ├── cordova-plugin-camera
│   │   └── ...
│   ├── cordova-plugin-file
│   │   └── ...
│   ├── cordova-plugin-file-transfer
│   │   └── ...
│   ├── cordova-plugin-geolocation
│   │   └── ...
│   ├── cordova-plugin-whitelist
│   │   └── ...
│   └── fetch.json
├── res
│   ├── icons
│   └── ...
└── www
    ├── css
    │   ├── images
    │   │   └── ..
    │   ├── themes
    │   │   └── ..
    │   ├── index.css
    │   ├── jqm.slidemenu.css
    │   └── ...
    ├── i18n
    │   ├── en
    │   │   └── strings.json
    │   ├── es
    │   │   └── strings.json
    │   └── va
    │       └── strings.json
    └── images
```

```
        └── ..
    ├── scripts
    │   ├── index.js
    │   ├── jqm.globals.js
    │   ├── jqm.slidemenu.js
    │   ├── peticiones.js
    │   ├── platformOverrides.js
    │   └── jquery[...].js
    ├── src
    ├── index.html
    └── telephonenumber.js
├── bower.json
├── build.json
├── config.xml
├── package.json
├── packages.config
└── taco.json
```

APÉNDICE C

Códigos de la aplicación móvil

C.1 Archivo «index.html»

```
<!doctype html>
<html lang="es">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
    <meta name="HandheldFriendly" content="True">
    <meta name="MobileOptimized" content="320" />
    <meta name="apple-mobile-web-app-capable" content="yes">
    <meta name="apple-mobile-web-app-status-bar-style" content="black">
    <meta http-equiv="cleartype" content="on">
    <meta name="viewport" content="width=device-width, initial-scale=1.0,
        maximum-scale=1.0, minimum-scale=1.0, user-scalable=no">
    <title>UrbanApp</title>

    <!--Incluir JQM y JQ-->
    <link rel="stylesheet" href="css/themes/jqmfb.min.css" />
    <link rel="stylesheet" href="http://code.jquery.com/mobile/latest/jquery.
        mobile.structure.min.css" />
    <script src="scripts/jquery-2.2.3.min.js"></script>
    <script src="scripts/jquery.animate-enhanced.min.js"></script>

    <!--JQM globals you can edit or remove file entirely... note it needs to be
         loaded before jquerymobile js -->
    <script src="scripts/jqm.globals.js"></script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js"></
        script>

    <!--JQM SlideMenu-->
    <link rel="stylesheet" href="css/jqm.slidemenu.css" />
    <script src="scripts/jqm.slidemenu.js"></script>
</head>
<body>
    <div id="slidemenu">
        <div id="profile">
            <img src="images/tegan.png">
            <div class="profile_info"><strong>UrbanApp</strong><br><small class
                ="languagespecificHTML" data-text="slogan">Cuidando la ciudad</
                small></div>
        </div>
        <h3 class="languagespecificHTML" data-text="perfil">Perfil</h3>
        <ul>
            <li><a href="#main_page" class="languagespecificHTML" data-text="
                inicio"><img src="images/smico3.png">Inicio</a></li>
```

```
37              <li><a href="#incidencias_page" onclick="cargarIncidencias()" class
                    ="languagespecificHTML" data-text="incidencias"><img src="
                    images/smico2.png">Incidencias</a></li>
38              <li><a href="#data_page" onClick='conectarUsu(document.
                    getElementById("tlfono").value)' class="languagespecificHTML"
                    data-text="datoUsu"><img src="images/smico5.png">Datos</a></li>
39              <li><a href="#config_page" class="languagespecificHTML" data-text="
                    configurar"><img src="images/smico6.png">Configurar</a></li>
40              <li><a href="#help_page" rel="external" class="languagespecificHTML
                    " data-text="ayuda"><img src="images/smico4.png">Ayuda</a></li>
41              <li><a href="#about_page" rel="external" class="
                    languagespecificHTML" data-text="acerca"><img src="images/
                    smico4.png">Acerca De</a></li>
42          </ul>
43      </div>
44      <div data-role="page" id="main_page" data-theme="a">
45          <div data-role="header" data-position="fixed" data-tap-toggle="false"
                data-update-page-padding="false">
46              <a href="#" data-slidemenu="#slidemenu" data-slideopen="false" data
                    -icon="smico" data-corners="false" data-iconpos="notext">Menu</
                    a>
47              <h1>UrbanAPP</h1>
48          </div>
49          <div data-role="content">
50              <div id="pre" style="display:block;">
51                  <p class="languagespecificHTML" data-text="espere">Espere, por
                        favor...</p>
52              </div>
53              <div id="deviceready" style="display:none;">
54                  <p class="languagespecificHTML" data-text="pagIni">Fotografie
                        el desperfecto lo mejor posible:</p>
55              </div>
56              <div>
57                  <input type="text" size="15" maxlength="60" placeholder="
                        Telefono móvil" id="tlfono" name="tlfono" class="
                        languageSpecificPlaceholder" data-text="telefono" required>
58                  <img style="width:100%" id="pintarImagen" src="./images/foto.
                        jpg" />
59                  <input type="text" size="15" maxlength="60" placeholder="
                        Descripción corta, max 60 caracteres" id="descrip" name="
                        descrip" class="languageSpecificPlaceholder" data-text="
                        descripcion">
60                  <button onclick="crearIncidencia()" class="
                        languageSpecificValue" data-text="foto" value="Tomar Foto">
                        </button>
61              </div>
62              <div id="geo"></div>
63          </div>
64      </div>
65
66      <div data-role="page" id="incidencias_page" data-theme="a">
67          <div data-role="header" data-position="fixed" data-tap-toggle="false"
                data-update-page-padding="false">
68              <a href="#" data-slidemenu="#slidemenu" data-slideopen="false" data
                    -icon="smico" data-corners="false" data-iconpos="notext">Menu</
                    a>
69              <h1>UrbanAPP</h1>
70          </div>
71          <div data-role="content">
72              <h2 class="languagespecificHTML" data-text="incidencias">
                    Incidencias</h2>
73              <p class="languagespecificHTML" data-text="pagInci">Aquí se mostrar
                    án todas las incidencias creadas por este usuario. El usuario
```

```
                será asociado a un número de teléfono automáticamente en la
                base de datos.</p>
74              <div name="tablaInc" id="tablaInc">
75                  <center>
76                      <p class="languagespecificHTML" data-text="carInci">
                            Cargando incidencias...</p>
77                      <img style="width:50%" id="loading" src="./images/loading.
                            gif" />
78                  </center>
79              </div>
80          </div>
81      </div>
82
83      <div data-role="page" id="data_page" data-theme="a">
84          <div data-role="header" data-position="fixed" data-tap-toggle="false"
                data-update-page-padding="false">
85              <a href="#" data-slidemenu="#slidemenu" data-slideopen="false" data
                    -icon="smico" data-corners="false" data-iconpos="notext">Menu</
                    a>
86              <h1>UrbanAPP</h1>
87          </div>
88          <div data-role="content">
89              <h2 class="languagespecificHTML" data-text="datoUsu">Datos de
                    Usuario</h2>
90              <p class="languagespecificHTML" data-text="pagUsu">Aquí se mostrará
                     los datos del perfil de usuario, datos personales y demás.</p>
91              <div>
92                  <input type="text" size="15" placeholder="Nombre" value="" id="
                        data_name" name="data_name" class="
                        languageSpecificPlaceholder" data-text="nombre">
93                  <input type="text" size="15" placeholder="Apellidos" value=""
                        id="data_last" name="data_last" class="
                        languageSpecificPlaceholder" data-text="apellidos">
94                  <input type="text" size="15" placeholder="Teléfono" value="" id
                        ="data_tlf" name="data_tlf" class="
                        languageSpecificPlaceholder" data-text="telefono">
95                  <input type="text" size="15" placeholder="Dirección" value=""
                        id="data_address" name="data_address" class="
                        languageSpecificPlaceholder" data-text="direccion">
96                  <input type="text" size="15" placeholder="Correo" value="" id="
                        data_mail" name="data_mail" class="
                        languageSpecificPlaceholder" data-text="correo">
97                  <button onclick="actUsu()">Guardar</button>
98              </div>
99          </div>
100     </div>
101
102     <div data-role="page" id="config_page" data-theme="a">
103         <div data-role="header" data-position="fixed" data-tap-toggle="false"
                data-update-page-padding="false">
104             <a href="#" data-slidemenu="#slidemenu" data-slideopen="false" data
                    -icon="smico" data-corners="false" data-iconpos="notext">Menu</
                    a>
105             <h1>UrbanAPP</h1>
106         </div>
107         <div data-role="content">
108
109             <h2 class="languagespecificHTML" data-text="configurar">Configurar<
                    /h2>
110             <p class="languagespecificHTML" data-text="pagConfig">Aquí se
                    mostrará la página de configuración con las opciones posibles
                    para cambiar.</p>
111             <div>
```

```
112                 <label for="idiomas" class="languagespecificHTML" data-text="
                        idioma">Idioma:</label>
113                 <select name="idiomas" id="idiomas">
114                     <option value="en">English</option>
115                     <option value="es">Español</option>
116                     <option value="va">Valencià</option>
117                 </select>
118                 <button onclick="cambiarIdioma()">Guardar</button>
119             </div>
120         </div>
121     </div>
122
123     <div data-role="page" id="help_page" data-theme="a">
124         <div data-role="header" data-position="fixed" data-tap-toggle="false"
                data-update-page-padding="false">
125             <a href="#" data-slidemenu="#slidemenu" data-slideopen="false" data
                    -icon="smico" data-corners="false" data-iconpos="notext">Menu</
                    a>
126             <h1>UrbanAPP</h1>
127         </div>
128         <div data-role="content">
129
130             <h2 class="languagespecificHTML" data-text="ayuda">Ayuda</h2>
131             <p class="languagespecificHTML" data-text="pagAyuda">Aquí se
                    mostrará la ayuda de la aplicación, como usar y demás
                    parafernalia.</p>
132         </div>
133     </div>
134
135     <div data-role="page" id="about_page" data-theme="a">
136         <div data-role="header" data-position="fixed" data-tap-toggle="false"
                data-update-page-padding="false">
137             <a href="#" data-slidemenu="#slidemenu" data-slideopen="false" data
                    -icon="smico" data-corners="false" data-iconpos="notext">Menu</
                    a>
138             <h1>UrbanAPP</h1>
139         </div>
140         <div data-role="content">
141
142             <h2 class="languagespecificHTML" data-text="acerca">Acerca De</h2>
143             <p class="languagespecificHTML" data-text="pagAcerca">Aplicación
                    diseñada y desarrollada por Alejandro de los Ríos Real como
                    Trabajo de Fin de Grado en la Universidad Politécnica de
                    Valencia.</p>
144         </div>
145     </div>
146
147     <script type="text/javascript" src="cordova.js"></script>
148     <script type="text/javascript" src="scripts/platformOverrides.js"></script>
149     <script type="text/javascript" src="telephonenumber.js"></script>
150     <script type="text/javascript" src="scripts/index.js"></script>
151     <script type="text/javascript" src="scripts/peticiones.js"></script>
152 </body>
153 </html>
```

C.2 Archivo «index.js»

```
1 $.getScript("/scripts/peticiones.js");
2
3 var tlf;
4 var latitud, longitud;
5 var pictureSource;
```

```
var destinationType;
var cLANGUAGE = null;
var languageSpecificObject = null;
var languageSpecificURL = "";
var englishLanguageSpecificURL = "i18n/en/strings.json";
var spanishLanguageSpecificURL = "i18n/es/strings.json";
var valenciaLanguageSpecificURL = "i18n/va/strings.json";

(function () {
    "use strict";

    document.addEventListener( 'deviceready', onDeviceReady.bind( this ), false );

    function onDeviceReady() {
        document.addEventListener('backbutton', onBackKeyDown, false);
        document.addEventListener('pause', onPause.bind( this ), false );
        document.addEventListener('resume', onResume.bind( this ), false );

        navigator.globalization.getPreferredLanguage(
            function (locale) {
                cLANGUAGE = locale.value;
                languageControls(cLANGUAGE);
            },
            function () { cLANGUAGE = "es"; }
        );

        document.getElementById('pre').style.display = 'none';
        document.getElementById('deviceready').style.display = 'block';

        var telephoneNumber = cordova.require("cordova/plugin/telephonenumber");
        telephoneNumber.get(function (result) {
            if (result) {
                tlf = result;
                document.getElementById("tlfono").value = result;
            } else {
                showConfirm();
            }
        }, function(error) {
            alert("Error consultando teléfono: " + error);
        });

        pictureSource = navigator.camera.PictureSourceType;
        destinationType = navigator.camera.DestinationType;

        var geoID = null;
        var options = { enableHighAccuracy: true };
        geoID = navigator.geolocation.getCurrentPosition(function (position) {
            $("#geo").html(
                'Latitud: ' + position.coords.latitude + '<br />' +
                'Longitud: ' + position.coords.longitude
            );
            latitud = position.coords.latitude;
            longitud = position.coords.longitude;
        }
        , onFail, options);
    };

    function onBackKeyDown() {
        navigator.app.exitApp();
    }

    function onPause() {
```

```
68      };
69
70      function onResume() {
71      };
72  })();
73
74  function onPrompt(results) {
75      if (results.buttonIndex == 2) {
76          navigator.app.exitApp();
77      } else if (!(/^\d{9}$/.test(results.input1)) || results.input1 == 0) {
78          alert("[ERROR] Teléfono mal escrito, vuelva a intentarlo.");
79          showConfirm();
80      } else {
81          tlf = results.input1;
82          document.getElementById("tlfono").value = results.input1;
83      }
84  }
85
86  function showConfirm() {
87      navigator.notification.prompt(
88          'Por favor, indica su número',
89          onPrompt,
90          'Número de teléfono',
91          ['Enviar', 'Salir'],
92          ''
93      );
94  }
95
96  function languageControls(language) {
97      if ((language.toString() == "en") || (language.toString() == "English") ||
            (language.toString().indexOf("en") != -1)) {
98          languageSpecificURL = englishLanguageSpecificURL;
99      } else if ((language.toString() == "va") || (language.toString() == "
            Valencia") || (language.toString().indexOf("va") != -1)) {
100         languageSpecificURL = valenciaLanguageSpecificURL;
101     } else {
102         languageSpecificURL = spanishLanguageSpecificURL;  //Por defecto Españ
                ol
103     }
104
105     onNetworkCall(languageSpecificURL, function (msg) {
106         languageSpecificObject = JSON.parse(msg);
107         $(".languagespecificHTML").each(function () {
108             $(this).html(languageSpecificObject.languageSpecifications[0][$(
                    this).data("text")]);
109         });
110         $(".languageSpecificPlaceholder").each(function () {
111             $(this).attr("placeholder", languageSpecificObject.
                    languageSpecifications[0][$(this).data("text")]);
112         });
113         $(".languageSpecificValue").each(function () {
114             $(this).attr("value", languageSpecificObject.languageSpecifications
                    [0][$(this).data("text")]);
115         });
116     });
117 };
118
119 function getLanguageValue(key) {
120     value = languageSpecificObject.languageSpecifications[0][key];
121     return value;
122 };
123
124 function onNetworkCall(urlToHit, successCallback) {
125     $.ajax({
```

```
126         type: "GET",
127         url: urlToHit,
128         dataType: "html",
129         timeout: 30000,
130     }).done(function (msg) {
131         successCallback(msg);
132     }).fail(function (jqXHR, textStatus, errorThrown) {
133         alert("Internal Server Error: " + errorThrown);
134     });
135 };
136
137 function crearIncidencia() {
138     conectarUsu(tlf);
139     crearInc(tlf, latitud, longitud, document.getElementById("descrip").value);
140 }
141
142 function cargarIncidencias() {
143     obtenerInc(document.getElementById("tlfono").value);
144 }
145
146 function actUsu() {
147     actualizarUsu(document.getElementById("data_name").value,
148         document.getElementById("data_last").value,
149         document.getElementById("data_tlf").value,
150         document.getElementById("data_address").value,
151         document.getElementById("data_mail").value);
152 }
153
154 function cambiarIdioma() {
155     cLANGUAGE = document.getElementById("idiomas").value;
156     languageControls(cLANGUAGE);
157 }
```

C.3 Archivo «peticiones.js»

```
1  var idFoto;
2
3  function conectarUsu(tlf) {
4      $.ajax({
5          url: 'http://api.ganefer.com/usuarios/conectar',
6          type: 'post',
7          dataType: 'json',
8          contentType: 'application/json',
9          data: JSON.stringify({ "telefono": tlf, "nombre": "", "apellidos": "",
               "direccion": "", "correo": "" }),
10         timeout: 50000
11     })
12     .done(function (data, textStatus, jqXHR) {
13         if (console && console.log) {
14             console.log("Solicitud usuario creado o logueado.");
15             mostrarUsu(data);
16         }
17     })
18     .fail(function (jqXHR, textStatus, errorThrown) {
19         if (console && console.log) {
20             console.log(jqXHR);
21             console.log("La solicitud a fallado: " + textStatus);
22         }
23     });
24 }
25
26 function mostrarUsu(datos) {
```

```
27      document.getElementById("data_tlf").value = datos.usuario.telefono;
28      document.getElementById("data_name").value = datos.usuario.nombre;
29      document.getElementById("data_last").value = datos.usuario.apellidos;
30      document.getElementById("data_address").value = datos.usuario.direccion;
31      document.getElementById("data_mail").value = datos.usuario.correo;
32  }
33
34  function actualizarUsu(nombre, apellidos, telefono, direccion, correo) {
35      $.ajax({
36          url: 'http://api.ganefer.com/usuarios/conectar',
37          type: 'put',
38          dataType: 'json',
39          contentType: 'application/json',
40          data: JSON.stringify({ "telefono": telefono, "nombre": nombre, "
                apellidos": apellidos, "direccion": direccion, "correo": correo }),
41          timeout: 50000
42      })
43      .done(function (data, textStatus, jqXHR) {
44          if (console && console.log) {
45              console.log("Solicitud usuario actualizado correcta.");
46              alert("Datos actualizados correctamente.");
47          }
48      })
49      .fail(function (jqXHR, textStatus, errorThrown) {
50          if (console && console.log) {
51              console.log("La solicitud a fallado: " + textStatus);
52          }
53      });
54  }
55
56  function crearInc(tlf, lat, long, desc) {
57      $.ajax({
58          url: 'http://api.ganefer.com/incidencias',
59          type: 'post',
60          dataType: 'json',
61          contentType: 'application/json',
62          data: JSON.stringify({ "telefono": tlf, "latitud": lat, "longitud":
                long, "descripcion": desc }),
63          timeout: 50000
64      })
65      .done(function (data, textStatus, jqXHR) {
66          if (console && console.log) {
67              console.log("Solicitud incidencia creada.");
68              idFoto = data.id;
69              capturePhoto();
70          }
71      })
72      .fail(function (jqXHR, textStatus, errorThrown) {
73          if (console && console.log) {
74              console.log(jqXHR);
75              console.log("La solicitud a fallado: " + textStatus);
76          }
77      });
78  }
79
80  function obtenerInc(tlf) {
81      $.ajax({
82          url: 'http://api.ganefer.com/incidencias/'+ tlf,
83          type: 'get',
84          timeout: 50000
85      })
86      .done(function (data, textStatus, jqXHR) {
87          if (console && console.log) {
88              console.log("Solicitud incidencia obtenida.");
```

```
89              dibujarTabla(data);
90          }
91      })
92      .fail(function (jqXHR, textStatus, errorThrown) {
93          if (console && console.log) {
94              console.log(jqXHR);
95              console.log("La solicitud a fallado: " + textStatus);
96          }
97      });
98  }
99
100 function dibujarTabla(data) {
101     var columnas = ["idInc", "latitud", "longitud", "solucionado"];
102
103     var divTablaInc = document.getElementById("tablaInc");
104     while (divTablaInc.hasChildNodes())
105         divTablaInc.removeChild(divTablaInc.firstChild);
106
107     var tabla = document.createElement("table");
108     var tblBody = document.createElement("tbody");
109
110     var cabecera = document.createElement("tr");
111     columnas.forEach(function (item, array) {
112         var celda = document.createElement("td");
113         var negra = document.createElement("strong");
114         var textoCelda = document.createTextNode(item);
115         negra.appendChild(textoCelda);
116         celda.appendChild(negra);
117         cabecera.appendChild(celda);
118     });
119
120     tblBody.appendChild(cabecera);
121
122     for (var i = 0; i < $(data.datos).size() ; i++) {
123
124         var hilera = document.createElement("tr");
125
126         columnas.forEach(function (item, array) {
127             var atributo = data.datos[i];
128             var celda = document.createElement("td");
129             celda.setAttribute("width", "25%");
130             var textoCelda = document.createTextNode(atributo[item]);
131             celda.appendChild(textoCelda);
132             hilera.appendChild(celda);
133         });
134
135         tblBody.appendChild(hilera);
136     }
137
138     tabla.appendChild(tblBody);
139     divTablaInc.appendChild(tabla);
140     tabla.setAttribute("border", "1");
141 }
142
143 function capturePhoto() {
144     navigator.camera.getPicture(onPhotoURISuccess, onFail, {
145         quality: 50,
146         destinationType: destinationType.FILE_URI
147     });
148 }
149
150 function onPhotoURISuccess(imageURI) {
151     var pImagen = document.getElementById('pintarImagen');
152     pImagen.style.display = 'block';
```

```
153     pImagen.src = imageURI;
154     subirImagen(imageURI)
155 }
156
157 function onFail(message) {
158     alert('ERROR: ' + message);
159 }
160
161 function subirImagen(fileURL) {
162     var opciones = new FileUploadOptions();
163     opciones.fileKey = "imagen";
164     opciones.fileName = idFoto + ".jpg";
165
166     var ft = new FileTransfer();
167     ft.upload(fileURL, encodeURI("http://api.ganefer.com/upload/upload.php"),
            uploadSuccess, uploadFail, opciones);
168 }
169
170 function uploadSuccess(r) {
171     alert("Codigo = " + r.responseCode + " Respuesta = " + r.response + "
            Enviado = " + r.bytesSent);
172 }
173
174 function uploadFail(error) {
175     alert("Ha ocurrido un error: Codigo = " + error.code + " upload error
            source " + error.source + " upload error target " + error.target);
176 }
```

APÉNDICE D
Códigos del servidor

D.1 Archivo «index.php»

```php
<?php

require 'controladores/usuarios.php';
require 'controladores/incidencias.php';
require 'vistas/VistaJson.php';
require 'utilidades/ExcepcionApi.php';

// Constantes de estado
const ESTADO_URL_INCORRECTA = 2;
const ESTADO_EXISTENCIA_RECURSO = 3;
const ESTADO_METODO_NO_PERMITIDO = 4;

// Preparar manejo de excepciones
$formato = isset($_GET['formato']) ? $_GET['formato'] : 'json';

switch ($formato) {
    case 'xml':
        $vista = new VistaXML();
        break;
    case 'json':
    $vista = new VistaJson();
    break;
    default:
        $vista = new VistaJson();
}

set_exception_handler(function ($exception) use ($vista) {
    $cuerpo = array(
        "estado" => $exception->estado,
        "mensaje" => $exception->getMessage()
    );
    if ($exception->getCode()) {
        $vista->estado = $exception->getCode();
    } else {
        $vista->estado = 500;
    }

    $vista->imprimir($cuerpo);
}
);

// Extraer segmento de la url
if (isset($_GET['PATH_INFO']))
    $peticion = explode('/', $_GET['PATH_INFO']);
```

```php
else
    throw new ExcepcionApi(ESTADO_URL_INCORRECTA, utf8_encode("No se reconoce la peticion"));

// Obtener recurso
$recurso = array_shift($peticion);
$recursos_existentes = array('incidencias', 'usuarios');

// Comprobar si existe el recurso
if (!in_array($recurso, $recursos_existentes)) {
    throw new ExcepcionApi(ESTADO_EXISTENCIA_RECURSO, "No se reconoce el recurso al que intentas acceder");
}

$metodo = strtolower($_SERVER['REQUEST_METHOD']);

// Filtrar método
switch ($metodo) {
    case 'get':
        if (method_exists($recurso, $metodo)) {
            $respuesta = call_user_func(array($recurso, $metodo), $peticion);
            $vista->imprimir($respuesta);
            break;
        }
    break;
    case 'post':
        if (method_exists($recurso, $metodo)) {
            $respuesta = call_user_func(array($recurso, $metodo), $peticion);
            $vista->imprimir($respuesta);
            break;
        }
        break;
    case 'put':
        if (method_exists($recurso, $metodo)) {
            $respuesta = call_user_func(array($recurso, $metodo), $peticion);
            $vista->imprimir($respuesta);
            break;
        }
    break;
    case 'delete':
        if (method_exists($recurso, $metodo)) {
            $respuesta = call_user_func(array($recurso, $metodo), $peticion);
            $vista->imprimir($respuesta);
            break;
        }
        break;
    default:
        // Metodo no aceptado
        $vista->estado = 405;
        $cuerpo = [
            "estado" => ESTADO_METODO_NO_PERMITIDO,
            "mensaje" => utf8_encode("Metodo no permitido")
        ];
        $vista->imprimir($cuerpo);
}
```

D.2 Archivo «VistaApi.php»

```php
<?php

/**
 * Clase base para la representacion de las vistas
```

```
 */
abstract class VistaApi{

    // Codigo de error
    public $estado;

    public abstract function imprimir($cuerpo);
}
```

D.3 Archivo «VistaJSON.php»

```
<?php

require_once "VistaApi.php";

/**
 * Clase para imprimir en la salida respuestas con formato JSON
 */
class VistaJson extends VistaApi
{

    /**
     * Imprime el cuerpo de la respuesta y setea el codigo de respuesta
     * @param mixed $cuerpo de la respuesta a enviar
     */
    public function imprimir($cuerpo)
    {
        if ($this->estado) {
            http_response_code($this->estado);
        }
        header('Content-Type: application/json; charset=utf8');
        echo json_encode($cuerpo, JSON_PRETTY_PRINT);
        exit;
    }
}
```

D.4 Archivo «ExcepcionApi.php»

```
<?php

/**
 * Excepcion personalizada para el envio del estado
 */
class ExcepcionApi extends Exception
{
    public $estado;

    public function __construct($estado, $mensaje, $codigo = 400)
    {
        $this->estado = $estado;
        $this->message = $mensaje;
        $this->code = $codigo;
    }
}
```

D.5 Archivo «ConexionBD.php»

```php
<?php
/**
 * Clase que envuelve una instancia de la clase PDO
 * para el manejo de la base de controladores
 */

require_once 'login_mysql.php';

class ConexionBD
{
    /**
     * Única instancia de la clase
     */
    private static $db = null;

    /**
     * Instancia de PDO
     */
    private static $pdo;

    final private function __construct()
    {
        try {
            // Crear nueva conexión PDO
            self::obtenerBD();
        } catch (PDOException $e) {
            // Manejo de excepciones
        }
    }

    /**
     * Retorna en la única instancia de la clase
     * @return ConexionBD|null
     */
    public static function obtenerInstancia()
    {
        if (self::$db === null) {
            self::$db = new self();
        }
        return self::$db;
    }

    /**
     * Crear una nueva conexión PDO basada
     * en las constantes de conexión
     * @return PDO Objeto PDO
     */
    public function obtenerBD()
    {
        if (self::$pdo == null) {
            self::$pdo = new PDO(
                'mysql:dbname=' . BASE_DE_DATOS .
                ';host=' . NOMBRE_HOST . ";",
                USUARIO,
                CONTRASENA,
                array(PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8")
            );

            // Habilitar excepciones
            self::$pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION)
                ;
        }
```

```
62          return self::$pdo;
63      }
64
65      /**
66       * Evita la clonación del objeto
67       */
68      final protected function __clone()
69      {
70      }
71
72      function _destructor()
73      {
74          self::$pdo = null;
75      }
76  }
```

D.6 Archivo «login_mysql.php»

```
1  <?php
2  /**
3   * Provee las constantes para conectarse a la base de datos
4   * Mysql.
5   */
6  define("NOMBRE_HOST", "ganefer.com.mysql"); // Nombre del host
7  define("BASE_DE_DATOS", "ganefer_com_tfgurbanapp"); // Nombre de la base de
       controladores
8  define("USUARIO", "ganefer_com_tfgurbanapp"); // Nombre del usuario
9  define("CONTRASENA", "tfgurbanapp"); // Constraseña
```

D.7 Archivo «incidencias.php»

```
1  <?php
2
3  class incidencias
4  {
5
6      // Datos de la tabla "incidencias" en la base de datos
7      const NOMBRE_TABLA = "incidencias";
8      const ID_INCIDENCIA = "idInc";
9      const TELEFONO = "telefono";
10     const LATITUD = "latitud";
11     const LONGITUD = "longitud";
12     const DESCRIPCION = "descripcion";
13     const FECHA_ENTRADA = "fecha_entrada";
14     const TRAMITADO = "tramitado";
15     const SOLUCIONADO = "solucionado";
16     const FECHA_SOLUCION = "fecha_solucion";
17     const REABIERTA = "reabierta";
18     const ID_OPERARIO = "idOper";
19
20     const CODIGO_EXITO = 1;
21     const ESTADO_EXITO = 1;
22     const ESTADO_ERROR = 2;
23     const ESTADO_ERROR_BD = 3;
24     const ESTADO_ERROR_PARAMETROS = 4;
25     const ESTADO_NO_ENCONTRADO = 5;
26     const ESTADO_URL_INCORRECTA = 6;
27     const ESTADO_FALLA_DESCONOCIDA = 8;
28
```

```php
    /**
     * @param $peticion
     * @return array
     * @throws ExcepcionApi
     */
    public static function get($peticion)
    {
        if (sizeof($peticion)==1){
            return self::obtenerIncidencias($peticion[0]);
        }
        else{
            return self::obtenerIncidencias($peticion[0], $peticion[1]);
        }
    }

    /**
     * @param $peticion
     * @return array
     * @throws ExcepcionApi
     */
    public static function post($peticion)
    {
        if(empty($peticion[0])){
            $cuerpo = file_get_contents('php://input');
            $incidencia = json_decode($cuerpo);

            $idInc = self::crearInc($incidencia);

            http_response_code(201);
            return [
                "estado" => self::CODIGO_EXITO,
                "mensaje" => "Incidencia creada con exito",
                "id" => $idInc
            ];
        } else {
            throw new ExcepcionApi(self::ESTADO_URL_INCORRECTA, "ERROR url mal
                formada", 400);
        }
    }

    /**
     * @param $peticion
     * @return array
     * @throws ExcepcionApi
     */
    public static function put($peticion)
    {
        if (!empty($peticion[0])) {
            $body = file_get_contents('php://input');
            $incidencia = json_decode($body);

            if (self::actualizar($incidencia, $peticion[0]) > 0) {
                http_response_code(200);
                return [
                    "estado" => self::CODIGO_EXITO,
                    "mensaje" => "Registro actualizado correctamente"
                ];
            } else {
                throw new ExcepcionApi(self::ESTADO_NO_ENCONTRADO,
                    "La incidencia que intentas acceder no existe", 404);
            }
        } else {
            throw new ExcepcionApi(self::ESTADO_ERROR_PARAMETROS, "Falta Id de
                incidencia", 422);
```

```
91          }
92      }
93
94      /**
95       * @param $peticion
96       * @return array
97       * @throws ExcepcionApi
98       */
99      public static function delete($peticion)
100     {
101         if (!empty($peticion[0])) {
102             if (self::eliminar($peticion[0]) > 0) {
103                 http_response_code(200);
104                 return [
105                     "estado" => self::CODIGO_EXITO,
106                     "mensaje" => "Registro eliminado correctamente"
107                 ];
108             } else {
109                 throw new ExcepcionApi(self::ESTADO_NO_ENCONTRADO,
110                     "La incidencia que intentas acceder no existe", 404);
111             }
112         } else {
113             throw new ExcepcionApi(self::ESTADO_ERROR_PARAMETROS, "Falta
                identificador de incidencia", 422);
114         }
115
116     }
117
118     /**
119      * Obtiene la coleccion de incidencias o un solo incidencia indicado por el
           identificador
120      * @param int $telefono
121      * @param null $idInc identificador de la incidencia (Opcional)
122      * @return array registros de la tabla incidencias
123      * @throws ExcepcionApi
124      */
125     private function obtenerIncidencias($telefono, $idInc = NULL)
126     {
127         try {
128             if (!$idInc) {
129                 $incidenciaBD = self::obtenerIncPorTelefono($telefono);
130             } else {
131                 $incidenciaBD = self::obtenerIncPorId($idInc, $telefono);
132             }
133             if ($incidenciaBD != NULL) {
134                 http_response_code(200);
135                 return
136                     [
137                         "estado" => self::ESTADO_EXITO,
138                         "datos" => $incidenciaBD
139                     ];
140             } else {
141                 throw new ExcepcionApi(self::ESTADO_ERROR, "Se ha producido un
                    error");
142             }
143         } catch (PDOException $e) {
144             throw new ExcepcionApi(self::ESTADO_ERROR_BD, $e->getMessage());
145         }
146     }
147
148     private function obtenerIncPorTelefono($telefono){
149         $comando = "SELECT * FROM " . self::NOMBRE_TABLA .
150             " WHERE " . self::TELEFONO . "=?";
151
```

```
152         // Preparar sentencia
153         $sentencia = ConexionBD::obtenerInstancia()->obtenerBD()->prepare(
                $comando);
154         $sentencia->bindParam(1, $telefono);
155 
156         if ($sentencia->execute())
157             return $sentencia->fetchAll(PDO::FETCH_ASSOC);
158         else
159             return null;
160     }
161 
162     private function obtenerIncPorId($idInc, $telefono){
163         $comando = "SELECT * FROM " . self::NOMBRE_TABLA .
164             " WHERE " . self::ID_INCIDENCIA . "=? AND " .
165             self::TELEFONO . "=?";
166 
167         // Preparar sentencia
168         $sentencia = ConexionBD::obtenerInstancia()->obtenerBD()->prepare(
                $comando);
169         $sentencia->bindParam(1, $idInc);
170         $sentencia->bindParam(2, $telefono);
171 
172         if ($sentencia->execute())
173             return $sentencia->fetchAll(PDO::FETCH_ASSOC);
174         else
175             return null;
176     }
177 
178     /**
179      * Añade una nueva incidencia asociada a un usuario
180      * @param $inc
181      * @return string identificador de la incidencia
182      * @throws ExcepcionApi
183      */
184     private function crearInc($inc)
185     {
186         if ($inc) {
187             try {
188                 $telefono = $inc->telefono;
189                 $latitud = $inc->latitud;
190                 $longitud = $inc->longitud;
191                 $descripcion = $inc->descripcion;
192                 $fecha_entrada = date('Y/m/d');
193 
194                 $pdo = ConexionBD::obtenerInstancia()->obtenerBD();
195 
196                 // Sentencia INSERT
197                 $comando = "INSERT INTO " . self::NOMBRE_TABLA . " ( " .
198                     self::TELEFONO . "," .
199                     self::LATITUD . "," .
200                     self::LONGITUD . "," .
201                     self::DESCRIPCION . "," .
202                     self::FECHA_ENTRADA . ")" .
203                     " VALUES(?,?,?,?,?)";
204 
205                 // Preparar la sentencia
206                 $sentencia = $pdo->prepare($comando);
207 
208                 $sentencia->bindParam(1, $telefono);
209                 $sentencia->bindParam(2, $latitud);
210                 $sentencia->bindParam(3, $longitud);
211                 $sentencia->bindParam(4, $descripcion);
212                 $sentencia->bindParam(5, $fecha_entrada);
213 
```

```
214                 $sentencia->execute();
215
216                 // Retornar en el ultimo idInc insertado
217                 return $pdo->lastInsertId();
218
219             } catch (PDOException $e) {
220                 throw new ExcepcionApi(self::ESTADO_ERROR_BD, $e->getMessage())
                        ;
221             }
222         } else {
223             throw new ExcepcionApi(
224                 self::ESTADO_ERROR_PARAMETROS,
225                 utf8_encode("Error en existencia o sintaxis de parametros
                        INCIDENCIAS"));
226         }
227
228     }
229
230     /**
231      * Actualiza la incidencia especificada por idUsuario
232      * @param object $incidencia objeto con los valores nuevos de la incidencia
233      * @param int $idInc
234      * @return PDOStatement
235      * @throws Exception
236      */
237     private function actualizar($incidencia, $idInc)
238     {
239         try {
240             // Creando consulta UPDATE
241             $consulta = "UPDATE " . self::NOMBRE_TABLA .
242                 " SET " . self::TELEFONO . "=?," .
243                 self::LATITUD . "=?," .
244                 self::LONGITUD . "=?," .
245                 self::DESCRIPCION . "=? " .
246                 " WHERE " . self::ID_INCIDENCIA . "=? AND " . self::TELEFONO .
                        "=?";
247
248             // Preparar la sentencia
249             $sentencia = ConexionBD::obtenerInstancia()->obtenerBD()->prepare(
                    $consulta);
250
251             $telefono = $incidencia->telefono;
252             $latitud = $incidencia->latitud;
253             $longitud = $incidencia->longitud;
254             $descripcion = $incidencia->descripcion;
255
256             $sentencia->bindParam(1, $telefono);
257             $sentencia->bindParam(2, $latitud);
258             $sentencia->bindParam(3, $longitud);
259             $sentencia->bindParam(4, $descripcion);
260             $sentencia->bindParam(5, $idInc);
261             $sentencia->bindParam(6, $telefono);
262
263             // Ejecutar la sentencia
264             $sentencia->execute();
265
266             return $sentencia->rowCount();
267
268         } catch (PDOException $e) {
269             throw new ExcepcionApi(self::ESTADO_ERROR_BD, $e->getMessage());
270         }
271     }
272
273
```

```
274     /**
275      * Elimina una incidencia segun su ID
276      * @param int $idInc identificador de la incidencia
277      * @return bool true si la eliminacion se pudo realizar, en caso contrario
             false
278      * @throws ExcepcionApi
279      */
280     private function eliminar($idInc)
281     {
282         try {
283             // Sentencia DELETE
284             $comando = "DELETE FROM " . self::NOMBRE_TABLA .
285                 " WHERE " . self::ID_INCIDENCIA . "=?";
286 
287             // Preparar la sentencia
288             $sentencia = ConexionBD::obtenerInstancia()->obtenerBD()->prepare(
                    $comando);
289 
290             $sentencia->bindParam(1, $idInc);
291 
292             $sentencia->execute();
293 
294             return $sentencia->rowCount();
295 
296         } catch (PDOException $e) {
297             throw new ExcepcionApi(self::ESTADO_ERROR_BD, $e->getMessage());
298         }
299     }
300 }
```

D.8 Archivo «usuarios.php»

```
1  <?php
2  
3  require './datos/ConexionBD.php';
4  
5  class usuarios
6  {
7      // Datos de la tabla "usuarios" en la base de datos
8      const NOMBRE_TABLA = "usuarios";
9      const TELEFONO = "telefono";
10     const NOMBRE = "nombre";
11   const APELLIDOS = "apellidos";
12   const DIRECCION = "direccion";
13     const CORREO = "correo";
14     const CLAVE_API = "claveApi";
15 
16     const ESTADO_CREACION_EXITOSA = 1;
17     const ESTADO_CREACION_FALLIDA = 2;
18     const ESTADO_ACTUALIZA_EXITO = 3;
19     const ESTADO_ACTUALIZA_NO_ENCONTRADO = 4;
20     const ESTADO_ERROR_BD = 5;
21     const ESTADO_AUSENCIA_CLAVE_API = 6;
22     const ESTADO_CLAVE_NO_AUTORIZADA = 7;
23     const ESTADO_URL_INCORRECTA = 8;
24     const ESTADO_FALLA_DESCONOCIDA = 9;
25     const ESTADO_PARAMETROS_INCORRECTOS = 10;
26 
27     /**
28      * @param $peticion
29      * @return array
30      * @throws ExcepcionApi
```

```
31      */
32     public static function get($peticion)
33     {
34     }
35
36     /**
37      * @param $peticion
38      * @return array|null
39      * @throws ExcepcionApi
40      */
41     public static function post($peticion)
42     {
43         if ($peticion[0] == 'conectar') {
44             $cuerpo = file_get_contents('php://input');
45             $usuario = json_decode($cuerpo);
46
47             $resultado = self::comprobarRegistro($usuario);
48
49             if ($resultado) {
50                 return self::loguear($usuario);
51             } else {
52                 return self::registrar($usuario);
53             }
54         } else {
55             throw new ExcepcionApi(self::ESTADO_URL_INCORRECTA, "Url mal
                formada", 400);
56         }
57     }
58
59     /**
60      * @param $peticion
61      * @return array
62      * @throws ExcepcionApi
63      */
64     public static function put($peticion)
65     {
66         if (!empty($peticion[0])) {
67             if (self::actualizar() > 0) {
68                 http_response_code(200);
69                 return [
70                     "estado" => self::ESTADO_ACTUALIZA_EXITO,
71                     "mensaje" => "Registro actualizado correctamente"
72                 ];
73             } else {
74                 throw new ExcepcionApi(self::ESTADO_ACTUALIZA_NO_ENCONTRADO,
75                     "Usuario no registrado, registre el numero", 404);
76             }
77         } else {
78             throw new ExcepcionApi(self::ESTADO_URL_INCORRECTA, "Url mal
                formada", 422);
79         }
80     }
81
82     /**
83      * Comprueba si el usuario ya esta registrado o no
84      * @param mixed $usuario a comprobar
85      * @return mixed resultado de la consulta en BD
86      */
87     private function comprobarRegistro($usuario){
88         $telefono = $usuario->telefono;
89
90         $comando = "SELECT telefono FROM " . self::NOMBRE_TABLA .
91             " WHERE " . self::TELEFONO . "=?";
92
```

```
93          $sentencia = ConexionBD::obtenerInstancia()->obtenerBD()->prepare(
                $comando);
94
95          $sentencia->bindParam(1, $telefono);
96
97          $sentencia->execute();
98
99          return $sentencia->fetch();
100     }
101
102     /**
103      * Crea un nuevo usuario en la base de datos
104      * @param mixed $usuario para crear
105      * @return array resultado devuelto
106      * @throws ExcepcionApi
107      */
108     private function registrar($usuario)
109     {
110         $resultado = self::crear($usuario);
111
112         switch ($resultado) {
113             case self::ESTADO_CREACION_EXITOSA:
114                 http_response_code(200);
115                 return
116                     [
117                         "estado" => self::ESTADO_CREACION_EXITOSA,
118                         "mensaje" => utf8_encode("Registro con exito!")
119                     ];
120                 break;
121             case self::ESTADO_CREACION_FALLIDA:
122                 throw new ExcepcionApi(self::ESTADO_CREACION_FALLIDA, "Ha
                        ocurrido un error");
123                 break;
124             default:
125                 throw new ExcepcionApi(self::ESTADO_FALLA_DESCONOCIDA, "Falla
                        desconocida", 400);
126         }
127     }
128
129     /**
130      * Crea un nuevo usuario en la tabla "usuarios"
131      * @param mixed $datosUsuario columnas del registro
132      * @return int codigo para determinar si la insercion fue correcta
133      * @throws ExcepcionApi
134      */
135     private function crear($datosUsuario)
136     {
137     $telefono = $datosUsuario->telefono;
138         $nombre = $datosUsuario->nombre;
139     $apellidos = $datosUsuario->apellidos;
140     $direccion = $datosUsuario->direccion;
141
142         $correo = $datosUsuario->correo;
143
144         $claveApi = self::generarClaveApi();
145
146         try {
147
148             $pdo = ConexionBD::obtenerInstancia()->obtenerBD();
149
150             // Sentencia INSERT
151             $comando = "INSERT INTO " . self::NOMBRE_TABLA . " ( " .
152         self::TELEFONO . "," .
153                 self::NOMBRE . "," .
```

```
154                 self::APELLIDOS . "," .
155         self::DIRECCION . "," .
156                 self::CORREO . "," .
157                 self::CLAVE_API . ")" .
158                 " VALUES(?,?,?,?,?,?)";
159
160             $sentencia = $pdo->prepare($comando);
161
162       $sentencia->bindParam(1, $telefono);
163             $sentencia->bindParam(2, $nombre);
164             $sentencia->bindParam(3, $apellidos);
165       $sentencia->bindParam(4, $direccion);
166             $sentencia->bindParam(5, $correo);
167             $sentencia->bindParam(6, $claveApi);
168
169             $resultado = $sentencia->execute();
170
171             if ($resultado) {
172                 return self::ESTADO_CREACION_EXITOSA;
173             } else {
174                 return self::ESTADO_CREACION_FALLIDA;
175             }
176         } catch (PDOException $e) {
177             throw new ExcepcionApi(self::ESTADO_ERROR_BD, $e->getMessage());
178         }
179     }
180
181     /**
182      * Genera una clave de acceso a la API
183      * @return null|string
184      */
185     private function generarClaveApi()
186     {
187         return md5(microtime() . rand());
188     }
189
190     /**
191      * @param $usuario
192      * @return array
193      * @throws ExcepcionApi
194      */
195     private function loguear($usuario)
196     {
197         $respuesta = array();
198
199         $telefono = $usuario->telefono;
200
201         if (self::autenticar($telefono)) {
202             $usuarioBD = self::obtenerUsuarioPorTelefono($telefono);
203
204             if ($usuarioBD != NULL) {
205                 $respuesta["telefono"] = $usuarioBD["telefono"];
206                 $respuesta["nombre"] = $usuarioBD["nombre"];
207         $respuesta["apellidos"] = $usuarioBD["apellidos"];
208                 $respuesta["direccion"] = $usuarioBD["direccion"];
209                 $respuesta["correo"] = $usuarioBD["correo"];
210                 $respuesta["claveApi"] = $usuarioBD["claveApi"];
211
212                 http_response_code(200);
213                 return
214                     [
215                         "estado" => 1,
216                         "usuario" => $respuesta
217                     ];
```

```
218             } else {
219                 throw new ExcepcionApi(self::ESTADO_FALLA_DESCONOCIDA, "Ha
                        ocurrido un error");
220             }
221         } else {
222             throw new ExcepcionApi(self::ESTADO_PARAMETROS_INCORRECTOS,
                    utf8_encode("Telefono no valido"));
223         }
224     }
225 
226     /**
227      * @param $telefono
228      * @return bool si existe o no en la BD
229      * @throws ExcepcionApi
230      */
231     private function autenticar($telefono)
232     {
233         $comando = "SELECT telefono FROM " . self::NOMBRE_TABLA .
234             " WHERE " . self::TELEFONO . "=?";
235 
236         try {
237 
238             $sentencia = ConexionBD::obtenerInstancia()->obtenerBD()->prepare(
                    $comando);
239 
240             $sentencia->bindParam(1, $telefono);
241 
242             $sentencia->execute();
243 
244             if ($sentencia) {
245                 return true;
246             } else {
247                 return false;
248             }
249         } catch (PDOException $e) {
250             throw new ExcepcionApi(self::ESTADO_ERROR_BD, $e->getMessage());
251         }
252     }
253 
254     /**
255      * Devuelve un usuario dado su telefono
256      * @param mixed $telefono del usuario a obtener
257      * @return mixed|null
258      */
259     private function obtenerUsuarioPorTelefono($telefono)
260     {
261         $comando = "SELECT * FROM " . self::NOMBRE_TABLA .
262             " WHERE " . self::TELEFONO . "=?";
263 
264         $sentencia = ConexionBD::obtenerInstancia()->obtenerBD()->prepare(
                $comando);
265 
266         $sentencia->bindParam(1, $telefono);
267 
268         if ($sentencia->execute())
269             return $sentencia->fetch(PDO::FETCH_ASSOC);
270         else
271             return null;
272     }
273 
274     /**
275      * @return null
276      * @throws ExcepcionApi
277      */
```

```
278     private function actualizar(){
279         $cuerpo = file_get_contents('php://input');
280         $usuario = json_decode($cuerpo);
281 
282         $telefono = $usuario->telefono;
283         $nombre = $usuario->nombre;
284         $apellidos = $usuario->apellidos;
285         $direccion = $usuario->direccion;
286         $correo = $usuario->correo;
287 
288         try {
289             $consulta = "UPDATE " . self::NOMBRE_TABLA .
290                 " SET " . self::NOMBRE . "=?," .
291                 self::APELLIDOS . "=?," .
292                 self::DIRECCION . "=?," .
293                 self::CORREO . "=? " .
294                 " WHERE " . self::TELEFONO . "=?";
295 
296             // Preparar la sentencia
297             $sentencia = ConexionBD::obtenerInstancia()->obtenerBD()->prepare(
                    $consulta);
298 
299             $sentencia->bindParam(1, $nombre);
300             $sentencia->bindParam(2, $apellidos);
301             $sentencia->bindParam(3, $direccion);
302             $sentencia->bindParam(4, $correo);
303             $sentencia->bindParam(5, $telefono);
304 
305             // Ejecutar la sentencia
306             $sentencia->execute();
307 
308             return $sentencia->rowCount();
309         } catch (PDOException $e) {
310             throw new ExcepcionApi(self::ESTADO_ERROR_BD, $e->getMessage());
311         }
312     }
313 }
```

D.9 Archivo «compresor.php»

```
1  <?php
2 
3  $fecha = date("dmy");   // Fecha del sistema
4 
5  $zip = new ZipArchive();    // Creamos objeto ZIP
6 
7  /**
8   * Función encargada de recorrer el directorio y añadir los archivos al ZIP.
9   * Si existen subdirectorios los leerá de forma recursiva.
10  * @param string $directorio nombre del directorio
11  * @param object $zip objeto zip
12  */
13 function agregar_zip($directorio, $zip) {
14     if (is_dir($directorio)) {  // Comprueba si es directorio
15         if ($dir_abierto = opendir($directorio)) {  // Abre el directorio
16             while (($archivo = readdir($dir_abierto)) !== false) {  // Bucle
                    que recorreo el directorio
17                 if (is_dir($directorio . $archivo) && $archivo != "." &&
                        $archivo != "..") { // Llamada recursiva
18                     agregar_zip($directorio . $archivo . "/", $zip);
19                 } elseif (is_file($directorio . $archivo) && $archivo != "." &&
                        $archivo != "..") { // Añade el archivo al ZIP
```

```
                    $zip->addFile($directorio . $archivo, $directorio .
                        $archivo);
                }
            }
            closedir($dir_abierto); // Cierra el directorio
        }
    }
}

$dir = $fecha . "/";      // Directorio a comprimir

$rutaFinal = "procesados";  // Directorio final

if(!file_exists($rutaFinal)){   // Comprueba si existe
    mkdir($rutaFinal);  // sino lo crea
}

$archivoZip = $fecha . ".zip";  // Nombre del ZIP

if ($zip->open($archivoZip, ZIPARCHIVE::CREATE) === true) { // Abre el ZIP
    agregar_zip($dir, $zip);    // Llamada a función
    $zip->close();  // Cierra el ZIP

    rename($archivoZip, "$rutaFinal/$archivoZip");  // Renombra y mueve el ZIP

    if (file_exists($rutaFinal. "/" . $archivoZip)) {   // Comprobación
        echo "Proceso Finalizado!";
    } else {
        echo "Error, archivo zip no ha sido creado!!";
    }
}
```

D.10 Archivo «upload.php»

```
<?php
    $date = date("dmy") . "/";
    if (!file_exists($date)) {
        mkdir($date, 0777, true);
    }
    $target_file = $date . basename($_FILES["imagen"]["name"]);
    move_uploaded_file($_FILES["imagen"]["tmp_name"], "./" . $date . $_FILES["
        imagen"]["name"]);
    echo "http://" . $_SERVER['SERVER_NAME'] . "/" . $target_file;
```

D.11 Archivo «mapa.php»

```
<html lang="es">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <script type="text/javascript" src="http://maps.google.com/maps/api/js?
            sensor=false"></script>

        <!-- Latest compiled and minified CSS -->
        <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap
            /3.3.7/css/bootstrap.min.css" integrity="sha384-
            BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u"
            crossorigin="anonymous">

```

```
10          <!-- Optional theme -->
11          <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap
                /3.3.7/css/bootstrap-theme.min.css" integrity="sha384-
                rHyoN1iRsVXV4nD0JutlnGaslCJuC7uwjduW9SVrLvRYooPp2bWYgmgJQIXwl/Sp"
                crossorigin="anonymous">
12
13          <!-- Latest compiled and minified JavaScript -->
14          <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/
                bootstrap.min.js" integrity="sha384-
                Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa"
                crossorigin="anonymous"></script>
15
16          <link href="css/styles.css" rel="stylesheet">
17
18          <script type="text/javascript">
19              function load() {
20                  var map = new google.maps.Map(document.getElementById("map"), {
21                      center: new google.maps.LatLng
                            (39.475621076408196,-0.37142488479616986),
22                      zoom: 13,
23                      mapTypeId: 'roadmap'
24                  });
25                  var infoWindow = new google.maps.InfoWindow;
26                  downloadUrl("markers.php", function(data) {
27                      var xml = data.responseXML;
28                      var markers = xml.documentElement.getElementsByTagName("
                            marker");
29                      for (var i = 0; i < markers.length; i++) {
30                          var point = new google.maps.LatLng(
31                              parseFloat(markers[i].getAttribute("lat")),
32                              parseFloat(markers[i].getAttribute("lng")));
33                          var icon = 'img/marker.png';
34                          var marker = new google.maps.Marker({
35                              map: map,
36                              position: point,
37                              icon: icon
38                          });
39                      }
40                  });
41              }
42              function downloadUrl(url, callback) {
43                  var request = window.ActiveXObject ?
44                      new ActiveXObject('Microsoft.XMLHTTP') :
45                      new XMLHttpRequest;
46                  request.onreadystatechange = function() {
47                      if (request.readyState == 4) {
48                          request.onreadystatechange = doNothing;
49                          callback(request, request.status);
50                      }
51                  };
52                  request.open('GET', url, true);
53                  request.send(null);
54              }
55              function doNothing() {}
56          </script>
57      </head>
58      <body onload="load()">
59      <nav class="navbar navbar-inverse">
60          <div class="container-fluid">
61              <div class="navbar-header">
62                  <button type="button" class="navbar-toggle collapsed" data-
                        toggle="collapse" data-target="#bs-example-navbar-collapse
                        -2">
63                      <span class="sr-only">Toggle navigation</span>
```

```
64                  <span class="icon-bar"></span>
65                  <span class="icon-bar"></span>
66                  <span class="icon-bar"></span>
67              </button>
68              <a class="navbar-brand" href="#">UrbanApp</a>
69          </div>
70
71          <div class="collapse navbar-collapse" id="bs-example-navbar-
                collapse-2">
72              <ul class="nav navbar-nav">
73                  <li class="active"><a href="#">General <span class="sr-only
                        ">(current)</span></a></li>
74                  <li><a href="#">Incidencias</a></li>
75              </ul>
76          </div>
77      </div>
78  </nav>
79  <div class="container">
80      <div class="row">
81          <div class="col-lg-6">
82              <div style="background: #ffffff; border: 2px solid #666666;
                    border-radius: 6px">
83                  Procesar día: <a href='../upload/compresor.php'>Procesar</a
                        >
84              </div>
85          </div>
86          <div class="col-lg-6">
87              <div id="map"></div>
88          </div>
89      </div>
90      <br />
91      <div class="row" style="background: #ff938e; border: 2px solid #666666;
             border-radius: 6px">
92          <div class="page-header">
93              <center><h1 id="incidencias">Incidencias pendientes</h1></
                    center>
94          </div>
95          <div class="col-lg-12">
96              <?php
97              include ("conexion.php");
98              $sql=mysqli_query($con,"select * from incidencias where
                    solucionado=0");
99              $rawdata = array();
100             //guardamos en un array multidimensional todos los datos de la
                    consulta
101             $i=0;
102             while($row=mysqli_fetch_array($sql))
103             {
104                 $rawdata[$i] = $row;
105                 $i++;
106             }
107             echo '<table class="table table-striped table-hover ">';
108             $columnas = count($rawdata[0])/2;
109             $filas = count($rawdata);
110
111             //Añadimos los titulos
112             for($i=1;$i<count($rawdata[0]);$i=$i+2){
113                 next($rawdata[0]);
114                 echo "<th><b>".key($rawdata[0])."</b></th>";
115                 next($rawdata[0]);
116             }
117
118             for($i=0;$i<$filas;$i++){
119                 echo "<tr>";
```

```
120                     for($j=0;$j<$columnas;$j++){
121                         echo "<td>".$rawdata[$i][$j]."</td>";
122                     }
123                     echo "</tr>";
124                 }
125                 echo '</table>';
126                 ?>
127             </div>
128         </div>
129         <div class="row" style="background: #9dff8d; border: 2px solid #666666;
                border-radius: 6px">
130             <div class="page-header">
131                 <center><h1 id="incidencias">Incidencias solucionadas</h1></
                        center>
132             </div>
133             <div class="col-lg-12">
134                 <?php
135                 include ("conexion.php");
136                 $sql=mysqli_query($con,"select * from incidencias where
                        solucionado=1");
137                 $rawdata = array();
138                 //guardamos en un array multidimensional todos los datos de la
                        consulta
139                 $i=0;
140                 while($row=mysqli_fetch_array($sql))
141                 {
142                     $rawdata[$i] = $row;
143                     $i++;
144                 }
145                 echo '<table class="table table-striped table-hover ">';
146                 $columnas = count($rawdata[0])/2;
147                 $filas = count($rawdata);
148
149                 //Añadimos los titulos
150                 for($i=1;$i<count($rawdata[0]);$i=$i+2){
151                     next($rawdata[0]);
152                     echo "<th><b>".key($rawdata[0])."</b></th>";
153                     next($rawdata[0]);
154                 }
155
156                 for($i=0;$i<$filas;$i++){
157                     echo "<tr>";
158                     for($j=0;$j<$columnas;$j++){
159                         echo "<td>".$rawdata[$i][$j]."</td>";
160                     }
161                     echo "</tr>";
162                 }
163                 echo '</table>';
164                 ?>
165             </div>
166         </div>
167         <footer>
168             <div class="row">
169                 <div class="col-lg-12">
170                     <ul class="list-unstyled">
171                         <li class="pull-right"><a href="#">Volver arriba</a></
                                li>
172                     </ul>
173                     <p>Creado por Alejandro de los Rios Real.</p>
174                     <p>Basado en <a href="http://getbootstrap.com" rel="
                            nofollow">Bootstrap</a>. Iconos de <a href="http://
                            fortawesome.github.io/Font-Awesome/" rel="nofollow">
                            Font Awesome</a>. Web fonts de <a href="http://www.
                            google.com/webfonts" rel="nofollow">Google</a>.</p>
```

```
                </div>
            </div>
        </footer>
    </div>
    </body>
</html>
```

D.12 Archivo «markers.php»

```
<?php
    header("Content-Type: text/xml");
    echo "<?xml version=\"1.0\"?>\n";
    echo "<markers>\n";
    include ("conexion.php");
    $sql=mysqli_query($con,"select * from incidencias");
    while($row=mysqli_fetch_array($sql))
    {
      echo "<marker id ='" . $row['idInc'] . "' lat='" . $row['latitud'] . "'
          lng='" . $row['longitud'] . "'>\n";
      echo "</marker>\n";
    }
    echo "</markers>";
?>
```

D.13 Archivo «conexion.php»

```
<?php
$con=mysqli_connect("ganefer.com.mysql","ganefer_com_tfgurbanapp","tfgurbanapp"
    ,"ganefer_com_tfgurbanapp");
if (mysqli_connect_errno())
{
  echo "Error en la conexión: " . mysqli_connect_error();
}
mysqli_set_charset($con,"utf8");
?>
```

D.14 Archivo «.htaccess»

```
RewriteEngine on
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ index.php?PATH_INFO=$1 [L,QSA]
```