Universidad Zaragoza
1542

Trabajo Fin de Grado
Grado en Ingeniería Informática

Desarrollo de una aplicación para la gestión de inventario con smart glasses.

Autor

Sergio Josa López

Director/es

Alfonso Puértolas Marcén

Universidad de Zaragoza / Escuela de Ingeniería y Arquitectura

2015

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

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

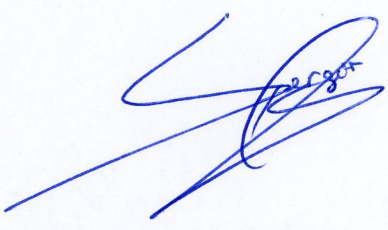
D./Dª. Sergio Josa López,

con nº de DNI 76924382P en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Grado en Ingeniería Informática, (Título del Trabajo) Desarrollo de una aplicación para la gestión de inventario con smart glasses.

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

Zaragoza, a 24 de Septiembre de 2015



Fdo: Sergio Josa López

Desarrollo de una aplicación para la gestión de inventario con smart glasses.

Resumen ejecutivo

Este trabajo consiste en el desarrollo de una aplicación móvil que permite, mediante el uso de smart glasses, el control de inventario del almacén de una empresa. La aplicación se integrará con el sistema de gestión de la misma (ERP), de forma que mediante el escaneo con las gafas de los códigos QR de cada producto, se conecte automáticamente con la sección del ERP que gestiona el almacén a través de un servicio Web desplegado en un servidor alojado en una máquina de la propia empresa, para ello se necesita que el dispositivo, en este caso las gafas, accedan a la red mediante una conexión virtual privada (VPN), de tal forma que vaya realizando un inventario exhaustivo del stock del que se dispone para cada uno de esos productos.

Contenido

1. Introducción.

Este capítulo contiene una introducción con el contexto y los objetivos principales del proyecto, la colaboración con la empresa y la motivación que conlleva, así como se detalla la estructura del documento.

1.1 Contexto del proyecto.

1.1.1 Colaboración con tap.

He realizado este proyecto en colaboración con la empresa Tap con dos sedes principales: una de ellas en el polígono Walqa (Huesca) y otra en el Centro Europeo de Empresas e Innovación (CEEI) de la Universidad de Zaragoza. Se define como una consultoría tecnológica especializada en la realización de proyectos que permiten mejorar los procesos de las empresas.

1.1.2 Contexto tecnológico .

Para la construcción del sistema objetivo de este proyecto es importante entender ciertas tecnologías actuales que están necesariamente presentes.

En primer lugar la aplicación se desarrolla para un sistema operativo en auge, Android, el cual ha tenido un crecimiento exponencial desde que entrara Google en acción en 2005.

En cuanto a las gafas de realidad aumentada, o más correctamente, smart glasses se puede decir que de cara al usuario no han tenido todavía gran aceptación debido a su precio tan elevado, pero si hablamos en un entorno empresarial, éstas han aportado agilidad y rapidez en los procesos de negocio de las empresas; ya que muchas de ellas han apostado por una nueva manera de realizar las tareas tanto complejas como simples y han obtenido mejoras de rendimiento.

También es necesario hablar sobre la forma en que las empresas gestionaban sus recursos hace algunos años y cómo lo hacen ahora. Los sistemas ERP (Enterprise Resource Management) aparecen a inicios del 2000 y prometen un incremento en las ventas frente a sus predecesores. En un ERP se integran una suite de productos corriendo bajo una arquitectura común: la gestión de la cadena de suministros (SCM), gestión de los clientes (CRM), gestión de inventarios, contabilidad, etc.

Ha habido un avance tremendo, mediante el cual, gracias a la tecnología, las empresas son capaces de encontrar información acerca de todos sus recursos en períodos muy cortos de tiempo. Todo ello gracias a la integración de un ERP en casi todas las empresas que mueven una cantidad de información considerable. En este caso, se ha trabajado con "eon.bs", el cual tiene funciones similares a Epicor, que es

un referente internacional en este campo, han dado un nuevo enfoque a la forma en la que se diseña, construye y utiliza la planeación de los recursos de la empresa.

Para hacerse una idea de la magnitud del ERP eon.bs, esta imagen muestra de forma resumida todos los módulos con los que cuenta el sistema.

Imagen. Módulos ERP eon.bs

Además, como el sistema no se limita solamente a una aplicación en local, sino que debe acceder en remoto a una base de datos externa, se debe poner en contexto dicha forma de almacenar la información. La base de datos que utiliza "eon.bs" en su capa inferior es Oracle, el cual es proveedor mundial líder de software para la administración de información. Otra de las opciones que existen para gestionar la información es una base de datos de código abierto como es MySQL, la cual es gratuita pero las prestaciones son más reducidas.

En estos últimos años se ha popularizado una nueva forma de almacenar de forma encriptada y sencilla información corta como palabras clave, URLs, caracteres especiales, códigos numéricos. Esto se denomina códigos QR y almacena la información en una matriz de puntos, es lo más utilizado en gestión de inventario en las industrias sobre todo por su rapidez de lectura, lo cual encaja perfectamente en el proyecto.

Finalmente, se debe destacar la funcionalidad de los servicios Web, que se está utilizando mucho en la red para acceder a información relevante directamente con una simple petición. Existen dos formas de realizar un servicio de estas características: SOAP o RESTful. En este caso para realizar la petición desde el dispositivo android lo más conveniente es usar un patrón de diseño REST ya que

agiliza mucho el proceso y tiene una construcción mucho más amigable del propio servicio. Hoy en día casi nadie usa SOAP a menos que sea estrictamente necesario o la propia empresa se resigne al cambio.

Todas las tecnologías descritas en este apartado se integrarán para llevar a cabo la funcionalidad final del sistema descrito anteriormente.

1.2 Motivación.

Una de las principales razones por las que he elegido realizar este proyecto es el hecho de que se base en tecnologías como android, ya que es una tecnología actual y la cual me parece interesante manejar y aprender desde cero.

Además era una gran oportunidad de conocer el ámbito empresarial, la forma de proceder en una empresa, relacionarse con personas ajenas a la universidad, trabajar en equipo con ellas y además ser un nuevo reto profesional.

En general, no sólo he mejorado mis capacidades como programador, sino que he aprendido como se trabaja en una empresa "de verdad", considero que he aprendido más de lo que esperaba, cosas interesantes y muy útiles, he estado en contacto con términos empresariales, logísticos y he trabajado en equipo con un grupo de personas muy diferentes a mis compañeros de universidad.

1.3 Estructura del documento.

El documento está estructurado en las distintas secciones que corresponden a la documentación fundamental de un proyecto, es decir, la documentación del problema, del diseño de la solución, la implementación con sus distintas etapas y las correspondientes conclusiones. Además, al final del documento se encuentra la bibliografía, los anexos, el manual de uso de la aplicación y el glosario.

Primero se explica en la sección del Problema (ver pág. 10) la descripción del problema, la documentación con los requisitos funcionales y no funcionales del sistema y los casos de uso del sistema.

Seguidamente, en la sección Diseño de la aplicación (ver pág. 15) se incluye la documentación del primer prototipo a papel que se realizó, diseño arquitectónico de la infraestructura web, el diseño de la capa de abstracción del ERP con respecto a la base de datos (esquema entidad-relación), diseño con tecnologías android y diseño final con departamento de diseño de la empresa, de esta forma mejorando las sucesivas iteraciones del desarrollo del proyecto.

La sección de Implementación (ver pág. 17) se va explicando todos los detalles de los componentes del sistema, como se ha ido aprendiendo y acoplando al proyecto, junto con el software y las tecnologías usadas para la implementación de todos ellos.

Posteriormente, se procede a la sección de Verificación y Validación (ver pág. 33) donde en cada hito del proyecto se ha ido verificando y validando cada una de las partes por el director del proyecto así como de las personas involucradas en esa etapa del proyecto.

En la sección Gestión del proyecto (ver pág. 35) se muestra la documentación referente a la gestión de todo el proyecto, desde la documentación acerca de los tiempos invertidos, así como la organización, planificación, distribución y gestión del trabajo.

En Conclusiones (ver pág. 40) se exponen las conclusiones obtenidas una vez terminado el proyecto.

Por último, se muestra la Bibliografía (ver pág. 42).

A partir de la página 43 se muestran todos los anexos de la memoria, los cuales contienen detalles adicionales al proyecto, tales como el prototipo final del sistema, el esquema con la integración de la base de datos del ERP, el manual de usuario, el manual de instalación del sistema, la documentación referente a la infraestructura del Web Service y el manual de conexión al servidor por VPN.

Finalmente se muestra un Glosario (ver pág 59) donde se encuentran las definiciones de los términos más importantes utilizados en el documento.

2. Problema.

En esta sección se va a presentar la descripción del problema, la documentación de los requisitos funcionales y no funcionales, y los casos de uso del sistema.

En el Anexo II, Primer prototipo a papel, se puede consultar el prototipo de la interfaz gráfica que se propuso en una de las primeras fases del proyecto.

2.1 Descripción.

Este trabajo consiste en el desarrollo de una aplicación móvil para las gafas de realidad aumentada que permita el control del inventario de todos los productos de un almacén de cierta empresa, a partir de la interacción del usuario con dicha aplicación.

En cuanto a la funcionalidad que debe ofrecer el sistema final, por una parte deberá acceder remotamente a la base de datos del ERP para obtener la información sobre el stock del producto y para ello, hacer la petición al servicio en Rest lo más eficiente posible. Por otra parte, el servicio alojado en una máquina que actúa como servidor dentro de la propia empresa, realiza la petición y consulta a la base de datos Oracle. De esta forma, obtiene toda la información necesaria acerca del producto además del stock y la muestra en la pantalla durante unos segundos, para posteriormente, seguir con el siguiente producto sin necesidad de utilizar ningún botón adicional y pudiendo realizar cualquier otra función con las manos.

Además, toda la funcionalidad descrita anteriormente se deberá implementar de manera que el sistema final se pueda usar perfectamente sin la necesidad de instalar ningún complemento adicional, simplemente se deberá conectar el dispositivo a la red de la empresa mediante una conexión privada virtual (VPN).

La funcionalidad descrita es, especialmente para un control exhaustivo de los productos que se encuentran en el almacén, de tal forma que un operario o cualquier personal de la empresa pueda examinarlos a la vez que se va recorriendo el mismo o, simplemente, desde cualquier lugar con conexión a internet disponiendo de los qr de los productos, sin la necesidad de desplazarse físicamente a dicho lugar o contando uno a uno los productos disponibles, además por supuesto, de los credenciales para la conexión a la red privada.

2.2 Requisitos de la aplicación.

El análisis global de los requisitos de una aplicación es un proceso de conceptualización y formulación de los conceptos que involucra de forma concreta. Es una parte fundamental del proceso de desarrollo de una aplicación, la mayor parte de los defectos encontrados en el software entregado se originan en la fase de análisis de requisitos, y además son los más caros de reparar.

Siempre se ha discutido quién es el dueño de los requisitos: el cliente o el desarrollador. Por lo tanto, para este proyecto, hubo una reunión con el cliente, en este caso, director de proyectos de Tap Consultoría, para definir ciertos requisitos funcionales y no funcionales.
El análisis inicial de un sistema debe tratar de descubrir los requerimientos del producto final que se desarrolla en detalle.
Unos de los principales objetivos de UML es hacer que este análisis sea lo suficientemente intuitivo para que los clientes y expertos en el dominio que solicitan el producto puedan comprenderlo, y lo suficientemente formal y riguroso para que se establezca una formulación no ambigua que pueda ser utilizada por el desarrollador.
Así pues, en el Anexo I se exponen los requisitos funcionales y no funcionales que el sistema debe satisfacer. En la tabla 1 se encuentran documentados todos los requisitos funcionales de la aplicación y en la tabla 2 están los requisitos no funcionales.

2.3 Casos de uso.

Un caso de uso es una descripción de los pasos o las actividades que deberán realizarse para llevar a cabo algún proceso.
Los diagramas de casos de uso sirven para especificar la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios y/u otros sistemas. O lo que es igual, un diagrama que muestra la relación entre los actores y los casos de uso en un sistema. Los diagramas de casos de uso se utilizan para ilustrar los requerimientos del sistema al mostrar cómo reacciona a eventos que se producen en su ámbito o en él mismo.
En este apartado se encuentra el modelado de los requisitos del sistema en cuanto a casos de uso. Los casos de uso describen las posibles interacciones que puede realizar, en este caso, el usuario autorizado con el sistema. La figura 1 muestra el diagrama de casos de uso de la aplicación, que muestra las interacciones anteriormente descritas de una manera gráfica e intuitiva.

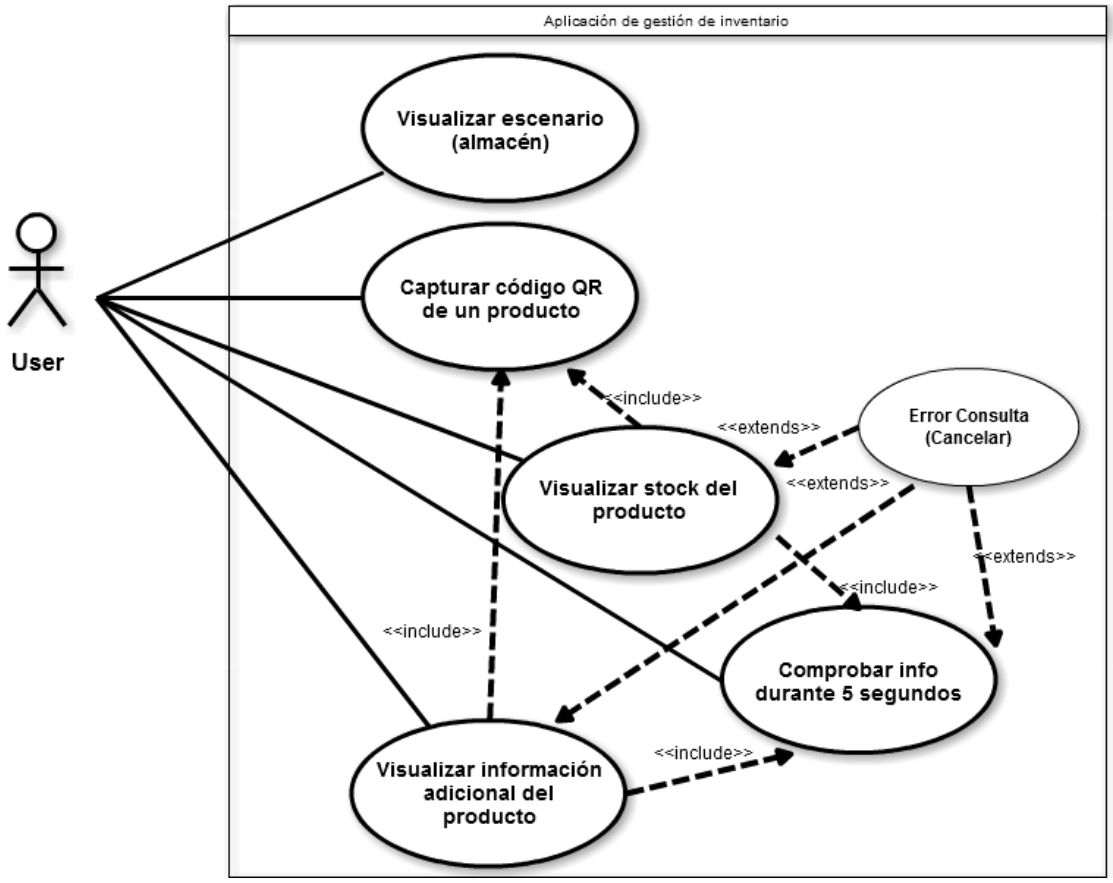


Figura 1. Diagrama de casos de uso del sistema

Análisis.

El modelo de casos de uso contiene 1 actor principal en el sistema: el usuario, es decir la persona encargada de la gestión de inventario en el almacén. Este actor usuario puede interactuar con los casos de uso "Visualizar escenario (almacén)", "Capturar código QR de un producto", "Visualizar stock del producto", "Comprobar info durante 5 segundos", "Visualizar información adicional del producto".

El caso de uso "Visualizar escenario (almacén)" permite al usuario visualizar el entorno simplemente activando automáticamente la cámara de las gafas, lo que permite al usuario desplazarse por todo el almacén.

El caso de uso "Capturar código QR de un producto" le permite al usuario, mediante el uso de las gafas, capturar un código QR de un producto del almacén.

El caso de uso "Visualizar stock del producto" le permite al usuario ver información acerca del stock del que se dispone en el almacén para así tomar decisiones en cuanto a reponer o no más stock del producto; este caso incluye al caso de uso "Capturar código QR de un producto" porque es una acción que se ha debido realizar para ello; a su vez incluye también "Comprobar info durante 5 segundos".

El último caso de uso del actor usuario, "Visualizar info adicional del producto" permite al usuario ver información general del producto, así como su descripción detallada, la familia y la subfamilia a la que pertenece; este caso de uso también incluye haber escaneado el código QR de cierto producto antes; a su vez incluye también "Comprobar info durante 5 segundos".

El caso de uso "Comprobar info durante 5 segundos" permite al usuario leer toda la información del producto disponible durante 5 segundos, una vez transcurrido este tiempo la aplicación volverá a su caso de uso "Visualizar escenario" para seguir escaneando otros códigos de otros productos.

Para realizar un completo análisis de casos de uso se va a especificar en detalle los casos de uso más importantes.

Caso de uso: Visualizar escenario	
Actor	Sistema
1. El usuario enciende las gafas y pone en funcionamiento la aplicación.	
	2. La aplicación muestra un mensaje de bienvenida.
	3. El sistema muestra el visor de la cámara, para el posterior escaneo.
4. El usuario ya dispone de la visión del escenario a través de la cámara del dispositivo.	
5. El usuario ya puede avanzar por el almacén para realizar las pertinentes operaciones.	

Tabla 3. Caso de uso detallado "Visualizar escenario"

Caso de uso: Visualizar stock de un producto	
1. El usuario realiza lo descrito para el caso de uso anterior.	
2. El usuario camina por el almacén hasta que desee realizar la consulta sobre un producto.	
3. Escanea dicho código QR con las gafas.	
	4. El sistema decodifica el código y lo transforma en texto.
	5. El sistema realiza la petición al servicio web sobre ese código de producto.
	6. El sistema accede a la base de datos

	para realizar la consulta.
	7. El sistema consigue la información acerca del stock y la muestra.
8. El usuario ya puede visualizar dicho stock y tomar las medidas oportunas.	

Tabla 4. Caso de uso detallado "Visualizar stock del producto"

3. Diseño.

En esta sección se incluyen todos los diseños arquitectónicos del sistema que se han realizado y mejorado a lo largo del proyecto en las sucesivas etapas del desarrollo. Cabe destacar las sucesivas iteraciones de diseño que se han realizado para cada una de las etapas del proyecto.

A la hora de realizar el diseño de la interfaz, se han identificado dos alternativas importantes. Se tuvo que elegir entre disponer de un botón en la pantalla donde se visualiza la información del producto o simplemente que la pantalla se mostrara durante unos segundos y volviera a la captura. Se decidió implementar la segunda opción finalmente, ya que en las gafas es incómodo tener la obligación de pulsar a un botón, ya que el operario que utilice la aplicación no podría tener total disponibilidad con las manos que, aparte de la velocidad y agilidad como objetivo de la aplicación, también se considera uno de los puntos más importantes el no tener que usar las manos con las gafas, simplemente para iniciar la aplicación.

En la arquitectura final, se ha elaborado el esquema entidad-relación de la base de datos del ERP con la que se va a integrar la aplicación para realizar las consultas correctamente.

Además, con respecto a la arquitectura final, se muestra el esquema que se va a seguir con la tecnología del Web Service, donde toda la responsabilidad se encuentra en el lado del servidor que es el que interacciona con la base de datos Oracle, el cliente solo realiza la petición "get" para consumir el servicio en Rest alojado en el servidor. En definitiva, el estilo de la arquitectura del sistema es cliente-servidor.

3.1 Primer prototipo a papel.

En el Anexo II se puede observar este primer diseño a papel, en el cual tras una pantalla inicial de bienvenida se accede automáticamente a la visualización del entorno mediante la cámara. De esta forma cuando nos quedamos mirando a un código QR, la aplicación intentará acceder al ERP para obtener la información correspondiente.

En esta versión inicial, se propuso un botón como vuelta a la cámara para seguir visualizando.

3.2 Esquema entidad/relación sección ERP.

Una vez se obtuvo el diseño a papel, otra de las iteraciones del diseño fue la integración con la base de datos del ERP. Para ello, la empresa proporcionó una visión global del conjunto que se explicará posteriormente en una de las etapas de implementación y, a su vez, proporcionó unos esquemas en XML sobre las tablas necesarias para la posterior integración con la aplicación. Con dichos ficheros XML se procedió al estudio de la base de datos y al diseño de un esquema entidad-relación basado en ingeniería inversa.

En el Anexo V se puede comprobar dicho esquema entidad-relación.

3.3 Primer diseño mediante las tecnologías android.

En este apartado se diseñó, en el entorno utilizado para android, la interfaz que se quería dado el prototipo a papel inicial.

En el Anexo III, se puede comprobar dicho diseño con un botón que se denominó "Finalizar" en la últimas figuras. Tras las diversas verificaciones este botón se decidió suprimir ya que reducía completamente la disponibilidad del usuario a realizar otras tareas, sustituyéndolo cartel y volviendo automáticamente a la pantalla anterior tras 4 segundos, ver figura en el mismo anexo.

3.4 Diseño de la infraestructura del Web Service.

En este punto se desarrolló un esquema como se puede ver en el Anexo VI donde se muestra un diseño de la infraestructura de un Web Service. Este esquema está dividido en dos tareas fundamentales: la creación del servicio con su publicación correspondiente, y la forma en la que se puede consumir el servicio mediante navegador y su posterior versión mediante la aplicación android.

Todo esto se explicará más en detalle en el punto de implementación.

3.5 Diseño app final con departamento de diseño gráfico tap.

Uno de los puntos más importantes de este proyecto ha sido el trabajo en el entorno empresarial, además de interaccionar y trabajar en equipo en algunos puntos del proyecto.

Este apartado fue uno de esos momentos donde el trabajo en equipo estuvo presente. Una vez diseñado una posible versión de la aplicación, y en una de las últimas etapas de verificación y validación se procedió al cambio de estilos de la interfaz. Esto se explicará más en detalle en el punto de verificación y validación, pero el resultado final que se pautó para su desarrollo se puede observar en el Anexo IV del documento.

4. Implementación.

La implementación no se ha realizado de golpe, sino que siguió un proceso incremental, un proceso en el que se han ido adquiriendo conocimientos para posteriormente, aplicarlos al proyecto. Todo ello hasta que la funcionalidad de la aplicación cumpliera todos los requisitos establecidos en la fase de análisis.

4.1 Etapas del proyecto.

4.1.1 Aprendizaje SO Android

En primer lugar, y para entrar en contexto, se preparó el entorno de programación. Para ello, el proyecto se ha desarrollado en el entorno IDE Eclipse, con el plug-in que proporciona todas las herramientas de android. Una vez preparado todo el entorno para trabajar, se decidió dividir el proyecto en distintas etapas con varios objetivos. Primero de todo y más importante aprender como programar en android, sus funcionalidades y como crear una aplicación desde cero. Para después, ir realizando hitos cada cierto periodo de tiempo como se muestra en cada uno de los puntos siguientes.

La tecnología android que se ha utilizado ha sido totalmente actual, desarrollando la aplicación en la versión 4.4 de android pero disponible para cualquier versión que disponga de dicha tecnología. Al principio, sobretodo, se ha seguido un tutorial [1] de android para un comienzo correcto y desde cero.

4.1.2 Aprendizaje utilización smart glasses.

Una vez que se dispuso de las gafas en mano, se pueden observar en la *Imagen 1*, se procedió a investigar el uso de las mismas. Para ello, lo primero de todo fue leerse toda la documentación referente a las mismas, así como la instalación del sdk para desarrollar con ellas.

Imagen 1. Gafas optinvent

Se investigó acerca de una aplicación basada en el modelo cliente-servidor para las gafas, se modificó y se probó. Dicha aplicación consistía en poder manejar mediante la pantalla táctil del móvil, el cursor de las gafas; de tal forma que, las gafas actuaban como servidor y el móvil o smartphone como cliente.
La conexión entre ambos dispositivos se realizaba mediante bluetooth pero también cabe la posibilidad mediante Wi-Fi.

De esta forma también se aprendió a transferir archivos a las gafas, a instalar nuevas aplicaciones en las mismas y a un completo manejo de las aplicaciones disponibles para ellas.

4.1.2 Escaneo e interacción con base de datos SQLite.

En esta fase crucial, se desarrolló parte de un primer prototipo en el cual, mediante el escaneo de un código QR y su posterior lectura, se accedía a la base de datos SQLite y se comprobaba a que usuario pertenecía dicho código.

Para realizar este hito se creó un nuevo proyecto, en el cual, primero se desarrolló una base de datos SQLite de Usuarios, únicamente identificados por un código numérico, nombre, edad, peso y altura. Se rellenó dicha tabla con cinco valores aleatorios.

Ahora bien, para la captura de los códigos QR mediante la cámara de las gafas, se eligió la librería "zxing".

Zxing es una librería procesadora de imágenes multi-formato de 1D/2D y de código abierto. Es capaz de reconocer prácticamente todo tipo de formatos, especialmente, los populares QR muy usados desde que la tecnología móvil está presente. Además se puede decir que Zxing tiene su propio generador online.

Lo primero de todo es, configurarlo como librería en Eclipse para poder usarlo en el proyecto de la app y solucionar los pocos errores que aparecen en el código.

Es importante, en la app, activar la posibilidad de uso de la cámara en el documento AndroidManifest.xml donde se añade la siguiente línea:

```
<uses-permission android:name="android.permission.CAMERA"/>
```

Además se debe registrar en la propia librería el nombre del paquete donde se va a utilizar, por lo tanto se modifica también el AndroidManifest.xml de la librería:

```
<intent-filter>
        <action android:name="nombrepaquete.SCAN"/>
</intent-filter>
```

A parte de todo esto, se deben deshabilitar las funciones de la librería, ya que es ajena a la aplicación y cambiar el nombre de la aplicación.

Una vez todo instalado, se pueden leer los códigos QR mediante intentos, cuando se detecta un código el programa va directamente al método "onActivityResult" donde se realizan las operaciones correspondientes al código.

En esta etapa, una vez leído el código, se realizaba consulta a la base de datos para comprobar a que usuario identificaba el código y se mostraba por pantalla en otra actividad.

4.1.3 Estudio base de datos ERP(secciones, tablas)

Como se ha explicado anteriormente, un ERP es un sistema que gestiona todos los recursos de una empresa.

Para eon.bs, el ERP con el que se ha trabajado, los distintos recursos se encuentran claramente diferenciados y se pueden observar en la Figura 1.

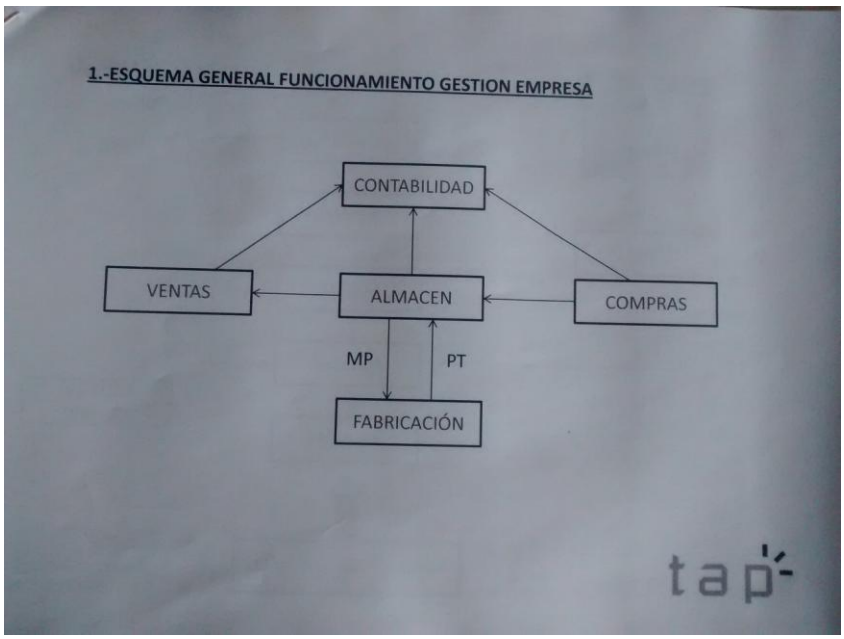


Figura 1. Esquema general ERP

La distribución de los recursos está presente y se divide en: Ventas, Contabilidad, Compras, Fabricación y Almacén.

Para este proyecto, la única parte del ERP con la que se ha trabajado ha sido el Almacén, que es donde se almacena toda la información acerca del stock de cada uno de los productos. Por lo tanto, se han estudiado los distintos procesos que se realizan en dicha sección, mostrados en la Figura 2.

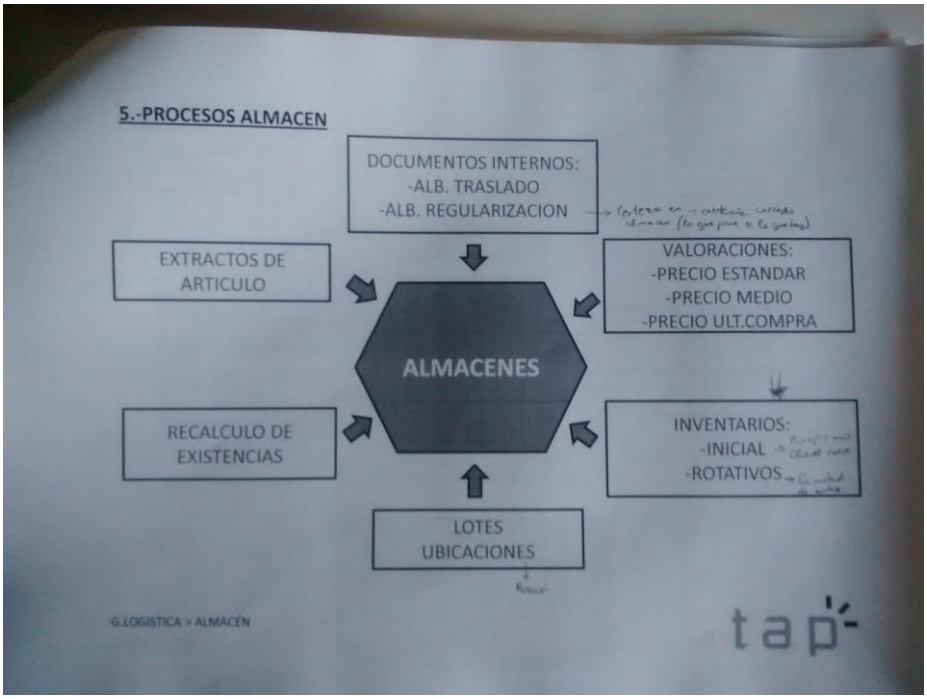


Figura 2. Modelado de procesos

En esta sección, ha habido que estudiar en profundidad algunos procesos como el recálculo de existencias y los extractos de los artículos. Por lo tanto, se ha pedido información acerca de las tablas correspondientes a estos procesos.

Las tablas principales que se han estudiado y con las que se han trabajado han sido:

- A_EXISTENCIAS: En esta tabla se muestra el stock para cada uno de los productos en el almacén correspondiente.
- E2_ARTICULOS: Aquí se encuentran cada uno de los artículos que distribuye la empresa en cuestión.
- E2_FAMILIAS y E2_SUBFAMILIAS: Donde se muestra para cada producto la familia y la subfamilia a la que pertenece, respectivamente.
- A_ALMACENES: En esta tabla se encuentran cada uno de los almacenes de los que dispone la empresa.

Todas estas tablas estaban especificadas en documentos XML provistos por la empresa para su estudio y utilización.

Una vez estudiado los atributos de las tablas, se ha diseñado el diagrama entidad relación con el que se va a integrar la aplicación para poder realizar la consulta que requiere la misma. Este esquema entidad-relación se puede observar en el Anexo V.

4.1.4 Alternativas de conexión con la BD.

En esta sección se explica las alternativas que se contemplaron para realizar la conexión con la base de datos del ERP.

Lo primero de todo y más directo, fue intentar acceder a través del ojdbc.jar desde android directamente a la base de datos, suponiendo que nos encontramos en la red local. Tras algunos días de buscar información acerca de ello y ver que era imposible hacer funcionarlo de esta forma debido a temas legales entre Oracle y Android (que a su vez es de Google), además de considerarlo una alternativa muy poco segura, se decidió descartar esta opción.

La siguiente opción ya fue implementar la conexión a través de un Web Service, pero como apenas se disponía de información acerca de ello y no se tenía conocimiento de su funcionamiento, se dedicó mucho tiempo a la formación. Una vez entendido el concepto de desplegar servicio y, posteriormente, consumirlo; se consideraron dos alternativas: el protocolo de acceso SOAP y el patrón de diseño RESTful.

Debido a que la empresa no estaba acostumbrada a utilizar Servicios Web y sabiendo las ventajas y desventajas de cada uno de ellos, la alternativa más favorable con mucha diferencia fue un diseño Restful, sobre todo por la complejidad de SOAP y la nula utilización que le dan las nuevas empresas a dicho protocolo.

Así pues, se decidió por una mezcla entre las dos alternativas contempladas. Por un lado un Servicio Web desplegado en Java que conecte con la base de datos Oracle mediante "ojdbc" y una aplicación android, ya diseñada en parte anteriormente, que consuma dicho servicio para acceder a la base de datos y consultar la información relevante.

4.1.6 Estudio lenguaje Oracle interaccionar con la BD.

Lo primero que se realizó en esta etapa fue establecer una conexión segura con la base de datos, por medio de la ip, puerto y nombre de usuario con contraseña en la sentencia para obtener la conexión. De tal forma que se entró a la base de datos de Copiso, empresa con la que se realizaron algunas pruebas.

En esta sección se aprendió el lenguaje Oracle para consultar una base de datos. Esto llevó un tiempo debido a que la consulta que se quería realizar era algo compleja, aunque cabía la posibilidad de desarrollarla en varias consultas agregadas. Por lo que, dado el esquema entidad-relación que describe como está construida la BD, se creó dicha consulta, habiendo una buena optimización al realizarlo en una única consulta.

La consulta que se ha desarrollado en una conexión mediante Java a la base de datos ha sido la siguiente:

SELECT a_existencias.stock_unidades, e2_articulos.articulo, e2_articulos.descripcion, e2_familias.clave_familia, e2_familias.descripcion, e2_subfamilias.clave_subfamilia, e2_subfamilias.descripcion

FROM a_existencias inner join a_almacenes on a_almacenes.codigo = a_existencias.fk_almacen and a_almacenes.almacen = 1 inner join e2_articulos on e2_articulos.codigo = a_existencias.fk_articulo and e2_articulos.articulo LIKE ?

inner join e2_familias on e2_familias.codigo = e2_articulos.EF1_CODIGO inner join e2_subfamilias on e2_subfamilias.codigo = e2_articulos.es_codigo;

Esta consulta devuelve, dado el código que se introduzca como id y sustituyendo el interrogante por dicho id en un "preparedStatement", la descripción, el stock del producto en el almacen, la familia y la subfamilia a la que pertenece dicho producto.

Se puede observar en la claúsula del SELECT cada uno de los parámetros que se requieren, además de los códigos o claves artificiales de cada uno de los atributos. Hay que diferenciarlo de la clave natural de dichos artículos, como por ejemplo es el identificador introducido para realizar la búsqueda.

Al no haber sentencia WHERE toda la consulta se realiza en el FROM, esto es muy utilizado en el lenguaje Oracle. Por lo que se puede observar se realizan varios joins entre tablas con la sentencia "inner join" en Oracle, el primero de ellos entre la tabla existencias y la tabla almacenes, el segundo entre el resultado de dicha consulta con la tabla artículos y, seguidamente, con las tablas familia y subfamilia.

De esta forma, solo existirá un único resultado, ya que en la tabla artículo el código o clave natural es único por producto y al realizar join sobre él sólo existirá una fila que coincida por cada join que se realice.

Ya que no se permite la conexión directa mediante "ojdbc" a la base de datos Oracle, debido a que android es de Google, y éste emprendió acciones legales con Oracle porque son competencia directa; a la hora de conectar a la base de datos Oracle desde una aplicación android directamente, muestra error de conexión. Además esta forma directa de conexión es muy poco segura y nada tolerante a fallos, incluso cabe decir que pocas aplicaciones online hoy por hoy conectan directamente a su base de datos. Por todo esto se llegó a la conclusión de implementar un Servicio Web que hiciera dicha labor.

4.1.5 Estudio y diseño simple de un Web Service desplegado en Java y consumido con Android.

En este apartado se explica cómo se creó un Web Service simple consumido en una app android. Para ello, es importante seguir el esquema de la infraestructura mostrado en el anexo VI.

Lo primero de todo se comprobó que se estaba usando Java 8 en la máquina donde se iba a alojar el servidor web, y posteriormente se instaló Apache Tomcat 8, totalmente compatible con dicha versión de Java, modificando las variables de entorno correspondientes, lo cual no es un proceso trivial.

El módulo *javax.ws.rs* es un módulo externo que ofrece un interfaz de alto nivel para la creación de servicios web. Concretamente, se usa Jersey [7], una implementación de la especificación JAX-RS [8].

Una vez instalado Tomcat, se procedió a la instalación de Jersey en Eclipse además de todo el paquete de J2EE para tener todas las funcionalidades disponibles para adaptar el entorno web. Al comprobar que todo se ha instalado correctamente, se tiene la posibilidad de crear un "Dynamic Web Project", el cual permitirá crear y desplegar el servicio web.

En la carpeta "lib" del proyecto que se acaba de crear se deben copiar todos los archivos descargados de Jersey. Pero no es suficiente con ello, también es necesario añadirlos en el Java Build Path del proyecto, con lo que se perdió mucho tiempo ya que en ningún tutorial se explicaba y el servicio no funcionaba.

Después, se creó la clase Java donde se crea el servicio, simplemente con un método "GET" que devuelve un "Hello Jersey"; a su vez se debe crear un Servlet Dispatcher, para registrar el url-pattern para su posterior acceso, en este caso, se dispuso mediante "/rest/*". Por lo que la dirección del servicio quedó así:

"http://ip:puerto/com.vogella.jersey.first/rest/helloJersey"

Ahora bien ya se tiene el servicio creado, lo cual se puede comprobar mediante el navegador web. Sin embargo, se necesita consumir dicho servicio mediante una app android, tal y como se muestra en el anexo.

Así pues, se debe crear un cliente en Java para consumir el servicio. Por lo que teniendo la URL a la que se quiere pedir el servicio, la estructura de petición es la siguiente mostrada en la *Imagen 2*:

```
protected String doInBackground(String... params) {
    result="";
    HttpClient httpclient = new DefaultHttpClient();
    HttpResponse response;
    String responseString = "Información del producto:\n";
    try {
        response = httpclient.execute(new HttpGet(apiURL+params[0]));
        StatusLine statusLine = response.getStatusLine();
        if(statusLine.getStatusCode() == HttpStatus.SC_OK){
            ByteArrayOutputStream out = new ByteArrayOutputStream();
            response.getEntity().writeTo(out);
            responseString += out.toString();
            out.close();
        } else{
            //Closes the connection.
            response.getEntity().getContent().close();
            throw new IOException(statusLine.getReasonPhrase());
        }
    } catch (ClientProtocolException e) {
        result+="BadProtocol";
    } catch (IOException e) {
        result+="ProblemIO";
    }
    result+=responseString;
    return result;

}
```

Imagen 2. Petición al Web Service desde Android

En un hilo asíncrono se crea la petición, con un cliente http y una respuesta en la cual se introducirá la petición "Get" con la URL explicada anteriormente. Si el resultado es OK, se introduce la respuesta en un buffer byte a byte que, posteriormente será volcado a un *String* y será mostrado en la pantalla de la aplicación.

4.1.7 Integración consulta Oracle con el Web Service.

En este hito se tuvo que integrar la conexión y consulta a la base de datos realizada en etapas anteriores con Java en el Servidor Web que se acaba de explicar.

Para ello, lo primero de todo se cambió el final de la URL de "/helloJersey" por "/ConsultaBd/{id}", de esta forma cuando se introdujera el id en dicha URL, ya se sabrá sobre que producto se ha requerido el servicio. Obviamente esto se realizó tanto en el lado del cliente como del servidor web, ya que en el cliente es donde se tiene el id del producto que se quiere consultar.

Cabe destacar que es lo único que cambia en el lado del cliente, la URL mediante la cual se consume el servicio, el resto de la conexión con dicho servicio es prácticamente igual a la explicada en el apartado anterior.

Lo siguiente fue realizar la conexión a la base de datos igual que en la etapa donde se realiza la consulta Oracle, es decir, mediante el ojdbc se obtiene la conexión y se crea

la consulta. Habiendo hecho esta misma a la BD, ya se tendrá la tupla correspondiente al producto, por lo tanto, se lee y se guarda en una variable, la cual será devuelta al terminar la ejecución del servicio web, para la espera de nuevas peticiones.

El funcionamiento del servicio es similar al explicado en el apartado anterior, con una funcionalidad diferente, a diferencia del anterior, cuando se requiera el servicio web, se realizará la consulta y automáticamente se devolverá el resultado.

4.1.8 Aplicación móvil operativa y funcional.

Una vez integrada la consulta y desarrollado el Web Service completo con el consiguiente acceso a la base de datos mediante la app, la aplicación ya tiene su total funcionalidad por lo que se dispone a realizar una reunión o auditoría interna con el director.

Al resultarse de un desarrollo iterativo incremental, esto se consideró como una etapa de verificación y validación, por lo tanto se explica también en ese punto. En esta auditoría se decidió cambiar el estilo de la aplicación para que fuera acorde al logo de la empresa.

4.1.9 Implementación y desarrollo de la interfaz final.

En este apartado se muestra las configuraciones de la pantalla con los estilos correspondientes a lo que se acordó en la auditoría.

En la figura 3 se muestra dicho documento.

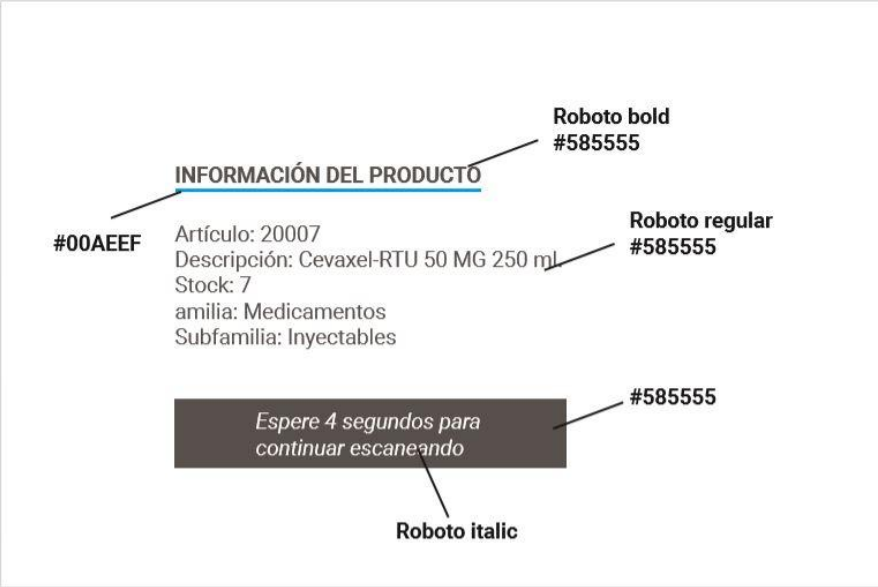


Figura 3. Estilos definitivos app.

Para implementar dichos cambios, lo primero de todo se descargaron los tipos de letra indicados y se cambiaron los documentos XML para cada uno de los recuadros de la pantalla, indicando para cada uno de ellos su color y tipo de letra correspondiente.

Una vez realizado esto, se comprobó que la aplicación seguía funcionando correctamente y se dispuso para la validación y verificación final, la cual se explica más adelante en el punto 5.

4.1.10 Conexión remota con el servidor.

En esta sección se indica como se ha realizado la conexión a la red local de la empresa mediante una conexión privada virtual (VPN).

Una vez desarrollada la aplicación, totalmente operativa y funcional, es interesante saber cómo conectarse al servidor web alojado físicamente en la oficina que tiene acceso a la red.

Este apartado se trabajó con el departamento de administración de sistemas, para saber las posibilidades que existían, ya que el firewall impedía el acceso a la base de datos sin una conexión con dirección IP permitida, es decir, la red local. Además de ello, el encargado del departamento de sistemas es el único que posee los credenciales de conexión a la red mediante VPN.

En el anexo VIII se explica cómo realizar esta conexión privada en remoto.

Así comprobamos como la aplicación funcionaba correctamente desde cualquier punto con acceso a internet.

4.2 Tecnologías involucradas.

4.2.1 Android (documentos XML). Estado del arte.

En este proyecto se ha desarrollado una aplicación para el sistema operativo android, en particular, desarrollada en Android 4.4 y apta para cualquier dispositivo con este SO.

Android fue diseñada para permitir a los usuarios crear distintas aplicaciones aprovechando las características propias de cada teléfono. De esta forma y, gracias a su especificación de API, mediante XML, se activan o desactivan dichas propiedades y se crea la aplicación.

Dispone de bases de datos liviana llamada SQLite usada para propósitos de almacenamiento.

Gracias a su conectividad también se ha podido llevar a cabo la aplicación, por su disposición de las tecnologías Wi-Fi.

La mayoría de las aplicaciones se escriben en Java, como ha sido el caso, pero sin embargo, no hay una máquina virtual Java en la plataforma. La que hace el trabajo de compilar es la máquina virtual Dalvik y, en la que posteriormente se ejecuta.

Finalmente, cabe destacar en cuanto al estado del arte de Android que es un sistema operativo pionero en desarrollo de aplicaciones y que se encuentra en un punto álgido hoy en día.

4.2.2 Gafas optinvent.

A diferencia de las Google Glass, las gafas Ora no presentan un pequeño espacio para el visor, sino que ofrecen un campo de visión mucho mayor. El campo de visión es unas tres veces mayor que Google Glass y ofrece 24º en 4:3, lo que equivale a una pantalla de 85", con un brillo máximo de 3.000 nits.

Además el visor sólo tiene 4mm de espesor y está hecho en plástico inyectado, lo que supone un importante ahorro a la hora de fabricarlo de forma masiva. El sistema tiene una apariencia bastante más tosca que las Google Glass, asemejándolas con unas gafas deportivas, en las que además podemos cambiar la posición del visor, de manera que podemos tenerlo justo delante o abajo, por si queremos que no nos distraiga respecto a lo que tenemos delante en el mundo real y podamos consultarlo en cualquier momento con mirar hacia abajo.

Las gafas Ora cuentan también con una cámara, conexión Wi-Fi y Bluetooth, un IMU de 9 grados, Android 4.1.2 y una batería de 800mAh que le da una autonomía de 8 horas, 4 horas de uso constante.

4.2.3 Códigos QR.

Módulo para almacenar información en una matriz de puntos o en un código de barras bidimensional. Fue creado en 1994 por la compañía japonesa Denso Wave, subsidiaria de Toyota. Presenta tres cuadrados en las esquinas que permiten detectar la posición del código al lector. La sigla «QR» viene de la frase inglesa «Quick Response» (respuesta rápida).

Este código se utiliza tanto para almacenar URLs como para guardar cualquier código de información. Ha sustituido al código de barras convencional en muchos usos diarios y, poco a poco, se está instaurando en los supermercados y almacenes.

4.2.4 Base de datos Oracle del ERP.

El sistema de gestión de la empresa (Enterprise Resource Management) "eonbs", que es con el ERP que se ha trabajado, tiene por debajo como forma de almacenamiento una base de datos Oracle 11g, la cual ofrece fiabilidad, alto rendimiento, buena seguridad y una escalabilidad sencilla.

Desde android no se permite la conexión directa mediante jdbc a la base de datos Oracle, además de que es muy poco seguro y nadie lo realiza de esta forma. Por lo tanto se desarrolló una conexión mediante Servicio Web.

4.2.5 Servicio Web Restful

Es un estilo de arquitectura software para sistemas hipermedia distribuidos como la World Wide Web. El término se originó en el año 2000, en una tesis doctoral sobre la web escrita por Roy Fielding, uno de los principales autores de la especificación del protocolo HTTP y ha pasado a ser ampliamente utilizado por la comunidad de desarrollo.

En la actualidad se usa en el sentido más amplio para describir cualquier interfaz entre sistemas que utilice directamente HTTP para obtener datos o indicar la ejecución de operaciones sobre los datos, en cualquier formato (texto plano, XML, JSON, etc) sin las abstracciones adicionales de los protocolos basados en patrones de intercambio de mensajes, como por ejemplo SOAP.

4.2.5.1 Modelo cliente-servidor

Este modelo se apoya en terminales(clientes) conectadas a un ordenador que las provee de un recurso (servidor).

Debido al uso de Rest, es claro que existe una infraestructura cliente-servidor sin estado, cada mensaje contiene toda la información necesaria para comprender la petición.

Como resultado, ni el cliente ni el servidor necesitan recordar ningún estado de las comunicaciones entre mensajes. Sin embargo, en la práctica, muchas aplicaciones basadas en HTTP utilizan cookies y otros mecanismos para mantener el estado de la sesión, lo cual no ha sido necesario para esta aplicación.

4.2.5.2 Herramientas Servidor Web.

Las herramientas utilizadas para desplegar el servicio web han sido:

- Java 1.8
- Jersey 2.17 que implementa JAX-RS 2.0
- J2EE (Java Platform, Enterprise Edition) para Eclipse
- ojdbc14.jar como plug-in para la conexión directa con la base de datos
- Apache Tomcat 8 como servidor de aplicaciones
- HTTP 1.1 como protocolo de comunicación

4.2.5.3 Servicio Web consumido en app android.

Para consumir dicho servicio Web, se utiliza la aplicación que se había desarrollado anteriormente, con las mismas tecnologías android.

El protocolo Http (Hypertext Transfer Protocol) es un protocolo sin estado capaz de transferir información, por lo tanto, en un hilo asíncrono se crea un cliente mediante Httpclient, una petición get "HttpGet" y una respuesta "HttpResponse" que permite una forma de poder utilizar el protocolo HTTP para realizar la petición al servidor.

Esta tecnología es la más utilizada cuando se habla de Servicios Web, el protocolo HTTP está instaurado y es el líder de protocolos de comunicación en internet.

4.2.6 Conexión VPN.

En este proyecto se utiliza VPN (Red Privada Virtual) para conectarnos al servidor web alojado en la oficina de la empresa desde cualquier punto con acceso a internet.

Por lo tanto, se usa VPN de acceso remoto, el cual es el modelo más usado actualmente en las empresas. Mediante autenticación se tiene un acceso a la red muy similar a una conexión en local.

4.3 Decisiones y/o restricciones de implementación.

La implementación se decidió realizar en IDE Eclipse ya que fue una de las decisiones de la propia empresa. Aunque Android Studio tiene grandes funcionalidades y puede que sea más sencillo de usar, Eclipse está muy consolidado en el ámbito de desarrollo de aplicaciones android.

En cuanto al servicio web, se ha desarrollado utilizando Apache Tomcat 8 ya que es el único servidor de aplicaciones que funciona totalmente con Java 8, el cual ha sido utilizado para desarrollar este proyecto.

De acuerdo a la aplicación, se puede considerar que han existido varias restricciones de diseño. La primera de ellas es que android limita mucho los estilos de la aplicación, basados en XML, por lo que la pantalla final se tuvo que amoldar a lo que las herramientas ofrecen.

Otra restricción que no modificó los requisitos de la aplicación pero fue de relevante importancia es, que al capturar los códigos QR de los productos mostrara la información en la propia imagen de la cámara, lo cual no se pudo realizar y se optó por trasladarlo a una tercera pantalla que mostrara la información y, a los pocos segundos, volviera automáticamente a la visión de la cámara. Como se puede observar, la funcionalidad es totalmente la misma.

4.4 Control de versiones y sistema de backup.

Con un control de versiones se puede recuperar un código que antes funcionaba bien, y que se ha estropeado; a lo cual se llama una copia de seguridad.
Por ello, se ha utilizado e instalado *Subversion* en un servidor Apache, lo que ha permitido un control de versiones. Se creó un repositorio donde se iba actualizando

y guardando la nueva aplicación en su versión Beta en un archivo binario, por lo cual cuando se realizaba algo nuevo, solo se guardaba el conjunto de las modificaciones, de esta forma se optimiza mucho el espacio de disco si lo comparas a guardar una copia íntegra por cada modificación.

Realizar la copia de seguridad protege los datos ante las pérdidas producidas por virus, ataque de hackers, borrados involuntarios, caídas de tensión, etc., además que los soportes donde se realizan las copias de seguridad también se deben proteger.

Además de Apache Subversion se ha utilizado Dropbox como herramienta de backup, donde cada día se guardaba el progreso realizado con el software actualizado a cada fecha. Principalmente se realizaba copia de seguridad cada día de modificación ya que los ficheros de android o, incluso Eclipse, es fácil que exista fallo en el código o cree cambios inesperados en dichos ficheros XML. De esta forma, se podía volver a la versión anterior recuperando la información de cualquiera de los dos repositorios, tanto el de Subversion como el de Dropbox y así, el periodo de tiempo perdido se reducía al mínimo.

4.5 Diagrama de paquetes

En esta sección se encuentra la documentación de la vista de despliegue del sistema. La Figura 4 muestra su presentación primaria, en notación UML.

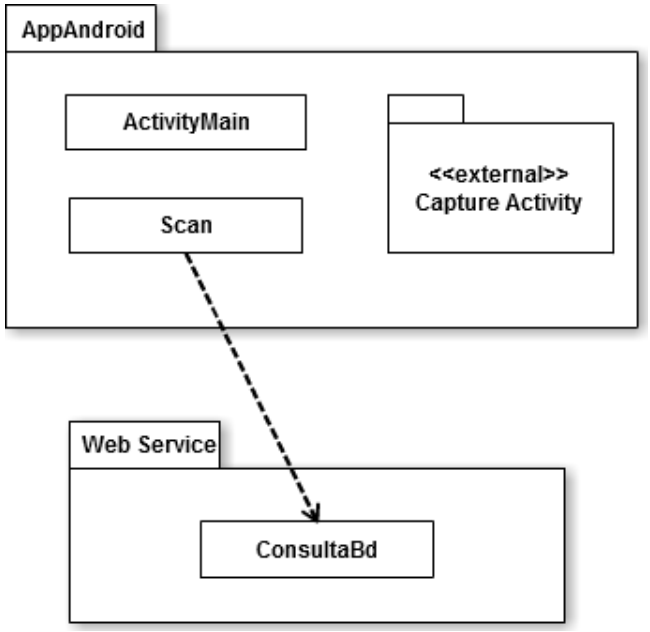


Figura 4. Diagrama de paquetes del sistema

Hay dos paquetes principales claramente diferenciados en el sistema, "AppAndroid" y "WebService".

En el paquete "AppAndroid" se encuentran las dos clases desarrolladas "ActivityMain" que es la clase que lanza la aplicación y "Scan" que se encarga de realizar la petición al WebService, como se muestra en la figura anterior. Estas clases

están relacionadas con sus respectivos ficheros XML y son dos actividades de la aplicación claramente diferenciadas.

También en dicho paquete encontramos una librería externa que se llama "CaptureActivity", esta se encarga de invocar a la cámara del dispositivo, capturar y escanear el código del producto.

En el paquete "WebService" se encuentra el servicio desplegado en Java, cuya clase se denomida "ConsultaBd", la cual se encargará de realizar la conexión con la base de datos Oracle.

4.6 Diagrama de despliegue.

En esta sección se encuentra la documentación de la vista de despliegue del sistema. La Figura 5 muestra su presentación primaria, en notación UML.

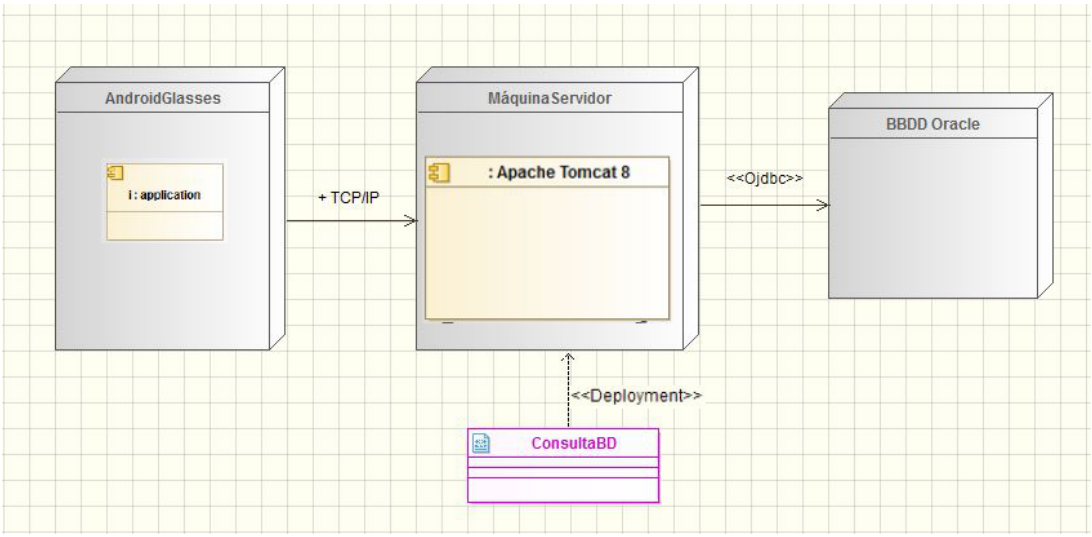


Figura 5. Diagrama de despliegue del sistema

Los componentes del sistema están desplegados sobre 3 nodos: AndroidGlasses como máquina cliente, "Máquina servidor" y el servidor donde se encuentra alojada la base de datos Oracle.

En el nodo "Android Glasses" o dispositivo cliente, se despliega el componente "application" que se corresponde al componente instalado en dicha máquina.

En cambio en el nodo "Máquina Servidor" se despliega el componente "Apache Tomcat 8" que se corresponde con el servidor de aplicaciones web Apache Tomcat 8 [9], además del componente o artefacto "ConsultaBD" que es el servicio web desplegado en dicho servidor de aplicaciones.

Por último, se encuentra el nodo BBDD Oracle, que se identifica como el servidor donde está alojada la base de datos del "ERP del eon.bs".

4.7 Elementos multimedia utilizados.

En cuanto a formatos de imágenes, se han usado varios. Para el logotipo de la aplicación, se ha usado el formato JPEG Progresivo, ya que debido a que puede guardar más colores, la imagen se mostrará con más calidad. Aunque la comprensión es con pérdidas, al ser una imagen tan pequeña no se encuentra ningún tipo de problema.

Por último, también se ha considerado el tamaño de las imágenes. Se han ajustado a las dimensiones de visualización, para minimizar todo lo posible el sobredimensionamiento, esto es, que la imagen sea muy grande y luego la visualización sea en un lugar muy pequeño.

Las imágenes tanto del logo de la aplicación como del logo de bienvenida se han redimensionado. Para el logo de la aplicación, se ha dispuesto en los tamaños drawable-hdpi (150x150), drawable-mdpi (100x100) y drawable-ldpi (75x75). Sin embargo para el logo de bienvenida en la aplicación, el tamaño de la imagen ha sido predeterminado acorde al tamaño de la pantalla del dispositivo en el que se muestre.

Para todas estas redimensiones y modificaciones se ha utilizado la herramienta GIMP, así como para hacer transparente el fondo de la imagen de bienvenida.

Todas las imágenes utilizadas han sido proporcionadas por la empresa, son originales y de alta calidad.

5. Verificación y validación

En esta sección se explica la forma en la que se ha trabajado para verificar y validar la aplicación, y el cómo se ha ido avanzando en cada uno de los hitos o etapas del proyecto.

5.1 Etapas.

A la hora de realizar un proyecto de esta envergadura es importante seguir una metodología de pruebas del sistema.

Para cada una de las etapas en las que se ha dividido el proyecto, ha existido una verificación y una validación de lo que se llevaba hecho hasta el momento. En algunos casos ha sido necesaria una reunión y otros simplemente se ha enviado el progreso por correo electrónico.

En cuanto a la usabilidad de la aplicación, se ha ido variando conforme se iban completando los hitos. Al principio se decidió poner un botón para volver a la cámara después de haber leído la información, pero tras diversas reuniones se decidió suprimirlo, la usabilidad era prácticamente la misma, pero la accesibilidad se reducía completamente para lo que era su propósito final.

Gracias a ello, se ha diseñado una aplicación totalmente accesible, que mediante el uso de threads se encarga de realizar todas las operaciones sin necesidad de tocar ningún botón. Así, cualquier persona con discapacidades motoras tiene la oportunidad de trabajar con la aplicación para las gafas, ya que no requiere de ningún botón para su constante funcionamiento.

5.2 Validación final.

Para la validación final, se realizó una reunión en la que se dispuso la aplicación en el dispositivo. Además, se comprobó que todos los requisitos habían sido cumplidos y la aplicación funcionaba correctamente.

Cabe destacar que se siguieron unas reglas de codificación estándar para el desarrollo de la aplicación. Esto es, se tuvo especial cuidado con la nomenclatura de paquetes, clases y atributos, lo cual facilitó la legibilidad del código a la hora de mostrarlo en las reuniones.

Una vez comprobados todo, se procedió a una validación final con el director y el departamento de diseño de tap. En esta reunión se decidió como iba a ser la apariencia final de la interfaz, se definieron cada uno de los estilos en cuanto al tipo de letra, y finalmente, los colores correspondientes y acordes al logo de la empresa, todo esto se puede observar en el punto 4 la figura 4.

Finalmente, al implementar todos los colores y fuentes propuestos, se dio el vistazo final a la aplicación y se comprobó que todo era correcto. En el anexo III se puede observar la aplicación final.

6. Gestión del proyecto

En esta sección se encuentra la documentación referente a la gestión del proyecto. Se compone de la distribución del tiempo en un diagrama de Gantt, planificación y metodologías utilizadas, y una especificación en cuanto al número de horas invertidas en el proyecto.

6.1 Distribución del tiempo, planificación.

A continuación se presenta la planificación del trabajo. El proyecto ha contando con 10 fases:

I. Aprendizaje sistema operativo Android.
II. Formación acerca de las gafas ORA de optinvent.
III. Escanear QR y creación base de datos SQLite.
IV. Estudio del ERP y tablas.
V. Interacción con la base de datos desde Java.
VI. Formación Web Services.
VII. Desarrollo Web Service e interacción con la app.
VIII. Implementación interfaz final con estilos.
IX. Verificación y validación final.
X. Desarrollo de la memoria.

La figura 5 muestra el diagrama de Gantt del proyecto.

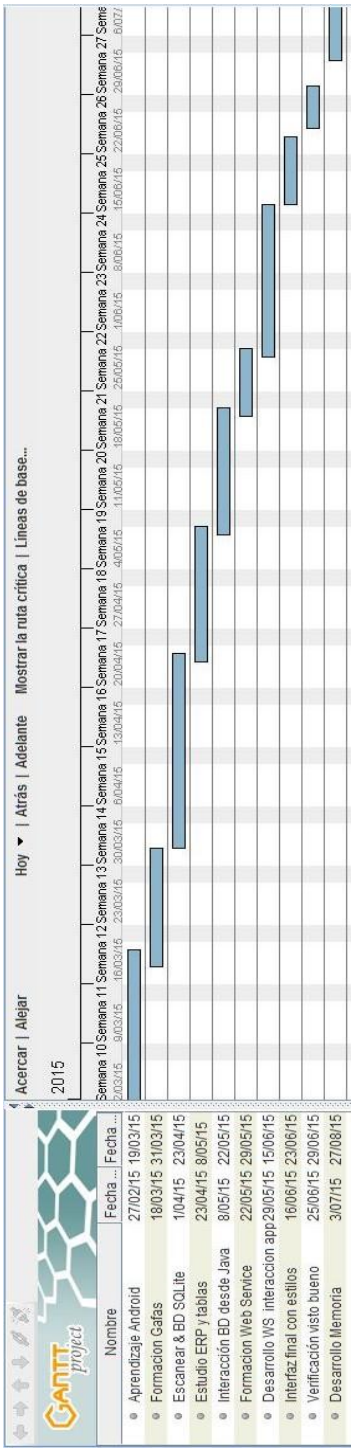


Figura 5. Diagrama de Gantt

Como se puede observar en el diagrama mostrado anteriormente, las tareas del proyecto han sido bastante equitativas comparando unas con otras. Cabe destacar sobre las demás, tanto "la construcción de una primera aplicación con escaneo y base de datos", como la formación sobre las posibles formas de realizar el Web Service. Además no se muestra bien en la imagen, pero el desarrollo de la memoria también ha llevado buena parte del tiempo.

En este proyecto ha habido una constante formación, para cada hito eran cosas nuevas y actuales, con las que una empresa trabaja a diario. Muchas tecnologías nuevas, un nuevo dispositivo y una novedosa aplicación han marcado el desarrollo del proyecto.

El proceso de "formación y creación del Web Service" se ha prologado tanto debido a las múltiples alternativas que existen para conectarse a una base de datos, en este caso Oracle, de un ERP. No se conseguían los resultados de búsqueda esperados y había que cambiar la tecnología utilizada, por lo que esto hizo que se dedicaran más horas. Además coincidió con la época de entregas y exámenes por lo que se tardó más en realizar este hito. También cabe destacar que los dos meses de verano no se han dedicado plenamente a la memoria.

6.2 Metodologías.

La metodología de trabajo ha seguido la filosofía de las metodologías "ágiles". Todo ello fue consensuado con el director de proyecto y se decidió por un proceso iterativo e incremental, en el cual, se consideró dividir el proyecto en hitos para hacer el aprendizaje de las nuevas tecnologías más sencillo e ir consolidándolo poco a poco. Debido a las ventajas de este modelo de proceso se han obtenido resultados en fases tempranas al proyecto, de modo que los fallos se podían detectar pronto así como los aspectos que había que retocar, también ha aportado facilidad para adaptarse a las circunstancias, flexibilidad y el hecho de que son resultados fruto de las iteraciones completamente funcionales, que mediante herramientas como Dropbox, se podían guardar y pasar por un proceso de pruebas.

Además, he formado parte del equipo de Tap Consultoría, especialmente del equipo de desarrollo del "eon.bs", por lo que he contado con la ayuda y soporte de algunos programadores, de la guía de consultores, departamento de diseño y departamento de sistemas para distintas fases del proyecto.

Cada hito del proyecto comprendía en su mayoría análisis diseño e implementación, aunque en algunos de ellos fueran cortas alguna de las partes. Al final de cada hito, se ha procedido a mostrar el resultado, verificarlo y validarlo si cumple con el prototipo y los requerimientos predeterminados.

Debido a tantas tecnologías nuevas para aprender se han realizado numerosas etapas como ya hemos explicado anteriormente, cada una de ellas considerada como hito del proyecto.

Con todo esto podemos decir que las metodologías ágiles en el desarrollo de software han estado constantemente presentes.

6.3 Métricas de trabajo.

En cuanto a las horas de trabajo dedicadas al proyecto, se han ido guardando en una hoja de cálculo Excel, de esta forma la métrica del trabajo ha sido exhaustiva y simplemente ha habido que recopilar dichas hojas de esfuerzos.
En la tabla 3 se muestra las horas dedicadas para cada una de los hitos descritos en el diagrama de Gantt anterior.

Actividad	Coste
Aprendizaje Android	31h
Formación Gafas	25h
Preparación entorno programación	5h
Escanear QR & Base Datos SQLite	55h
Estudio ERP y tablas	20h
Interacción BD desde Java	30h
Formación acerca Web Service	31h
Desarrollo del WS e interacción con app	36h
Interfaz final	16h
Verificación final	12h
Desarrollo Memoria	72h
Total	333h

Tabla 5. Coste total del proyecto, desglosado en los diferentes hitos.

Como se puede observar, los periodos de formación han sido muy largos, debido a que todo era novedad y ha sido necesario aprenderlo desde cero. En los últimos hitos de interfaz final y verificación se incluyen las reuniones en la empresa, el coste de modificar los estilos acordes al logo y colores de la misma y, también cabe destacar que llevó un tiempo conseguir conectarse mediante VPN a la red local de la empresa y, de esta forma, poder acceder al Web Service que se conecta a la base de datos del ERP.

6.4 Herramientas y tecnologías

Las herramientas y tecnologías empleadas en el desarrollo del proyecto, aparte de las identificadas en el apartado 4.3, han sido las mismas que utiliza el equipo de Tap Consultoría. Además, debido a la naturaleza del proyecto ha sido necesario añadir tecnoogías como JavaEE junto con las librerías y frameworks necesarios para desarrollar la aplicación en su plenitud.
El listado de herramientas utilizadas es:

- Windows como Sistema Operativo.
- Eclipse como IDE de desarrollo.
- sdk de Android kit de desarrollo software de aplicaciones en Eclipse.
- Ficheros XML, para desarrollo de aplicación Android.

- "Capture Activity" como escaneo de códigos QR.
- Java 1.8 como lenguaje de programación.
- JavaEE: Tecnología Java orientada a grandes aplicaciones empresariales.
- Jersey 1.2 como framework para desarrollo de servicios web.
- JAX-RS como API para la creación de servicios web en Java.
- Apache Tomcat 8 [9] como servidor de aplicaciones.
- Oracle como gestor de bases de datos del ERP eon.bs.
- Ofimática: Microsoft Office tanto Word como Excel.
- Creación de documentación UML: Cacoo (Herramienta online), Modelio.
- Control de versiones: TortoiseSVN
- Email y Skype como herramienta de comunicación con los miembros del equipo.

7. Conclusiones

Una vez terminado el proyecto se pueden sacar muchas conclusiones. La primera de ellas ha sido la correcta decisión de elegir un modelo de proceso basado en hitos, es decir, un proceso iterativo e incremental, basado en prototipos. Este modelo ha permitido amoldarse a los requisitos de la empresa, adaptarse a las circunstancias con facilidad, tener resultados presentables ya en fases tempranas al proyecto y obtener prototipos mejorados.

El sistema ha cumplido todos los requisitos que se habían impuesto al principio del proyecto, ya que se pueden realizar todas las funcionalidades pensadas para la aplicación, con buena usabilidad y, sobretodo, buena accesibilidad.

La ayuda recibida por parte del director del proyecto ha sido importantísima, principalmente para desplegar una vista global en la primera reunión del proyecto y, seguidamente, para dividir el proyecto y llevar el seguimiento en los diferentes hitos; ya que a primera vista parecía un mundo todo lo que había que aprender.

La experiencia obtenida en el desarrollo del proyecto ha sido muy satisfactoria. Por un lado, se ha aprendido a trabajar en un entorno empresarial, con compañeros ajenos a la universidad que te pueden echar una mano en cualquier momento, se ha trabajado en equipo en alguna fases del proyecto, tanto en la fase de explicación del ERP, como en la fase de diseño con estilos acordes a la empresa, como con el departamento de sistema para conectarse a la red de forma remota. Por otro lado, se han aprendido nuevas tecnologías en profundidad, como es android, trabajar con un ERP, despliegue de un Web Service en Java y consumido en android, nuevos dispositivos como las gafas de realidad aumentada, que poca gente tiene el gusto de poder trabajar con esta tecnología tan actual. Y finalmente, el hecho de crear una aplicación para una empresa real que se utilizará en muchas empresas que colaboran con tap.

Al fin y al cabo, el principal objetivo del proyecto era entrar en el ámbito empresarial y aprender. Ambas cosas se han cumplido con creces y sobretodo, se ha aprendido cosas muy útiles para el futuro profesional. Por lo tanto, creo que mi elección ha sido muy acertada.

En cuanto a posibles versiones beta de la aplicación hay varios aspectos con los que se podría trabajar para mejorarla. Uno de ellos podría ser, introducir el geoposicionamiento en la aplicación, ya que sería interesante localizar al usuario en la zona del almacén en la que se encuentra, de tal forma que muestre al usuario su posición vista desde arriba en un mapa bidimensional, así el usuario sabrá dónde está en cada momento. Otro aspecto que se podría mejorar sería introducir reconocimiento de patrones en lugar de reconocer los productos por código QR, de esta forma se podría identificar a un objeto solo por su apariencia física. También se podría añadir la información de los productos para que salieran directamente en la propia cámara, sin necesidad de saltar a una pantalla auxiliar. Esto último resulta

bastante complicado, y muchas horas de estudio ya que requiere modificar código de la propia librería de escaneo, la cual debería estar muy bien documentada.

8. Bibliografía

[1] Android. https://www.android.com/

[2] Android Tutorial. http://www.javaya.com.ar/androidya/index.php?inicio=0

[3] Gafas. http://optinvent.com/

[4] Zxing. http://zomwi.blogspot.com.es/2012/09/zxing.html

[5] Bases de datos SQLite. http://www.sgoliver.net/blog/bases-de-datos-en-android-iii-consultarrecuperar-registros/

[6] Conexión Oracle. http://elbauldelprogramador.com/conectar-base-de-datos-oracle/

[7] Jersey. https://jersey.java.net/.

[8] JAX-RS. *Java API for RESTful Services.* https://jax-rs-spec.java.net/.

[9] Apache Tomcat. http://tomcat.apache.org/.

[10]. Java. https://www.java.com/.

[11] Apache Maven. http://maven.apache.org/.

[12] UML. *Unified Modeling Language.* http://www.uml.org/.

[13] Oracle. http://www.oracle.com/index.html

[14] Dropbox. https://www.dropbox.com/

Anexo I. Requisitos de la aplicación

En el siguiente anexo se muestran los requisitos funcionales y no funcionales de la aplicación, los cuales se desarrollaron en la fase de análisis y se explican en el punto 2.2 de la memoria.

Requisitos funcionales

RF-1	La aplicación debe permitir la visualización del entorno a través de las propias gafas.
RF-2	La aplicación debe ser capaz de leer códigos qr a través de la cámara.
RF-3	La aplicación debe ser capaz de encontrar un producto en la base de datos del ERP de la empresa.
RF-4	La aplicación deberá ser capaz de acceder al servicio Web para realizar la petición.
RF-5	La aplicación debe mostrar la información acerca del stock de dicho producto.
RF-6	La aplicación deberá mostrar además información relativa al producto como la descripción, familia y subfamilia del producto.

Tabla 1. Requisitos funcionales de la aplicación

Requisitos no funcionales

RNF-1	La aplicación deberá poderse ejecutar en las gafas de realidad aumentada, así como en cualquier dispositivo android.
RNF-2	La aplicación deberá tener acceso a internet para su correcto funcionamiento.
RNF-3	La aplicación deberá ser totalmente intuitiva.
RNF-4	La aplicación no requerirá de ningún botón adicional para ofrecer una mayor accesibilidad.
RNF-5	La aplicación medirá el tiempo que se muestra la información de un producto para proceder al siguiente escaneo sin necesidad de realizar ninguna acción.
RNF-6	La aplicación estará construida de manera que se pueda integrar fácilmente con otros componentes Web, como puede ser el servicio en Rest.

Tabla 2. Requisitos no funcionales de la aplicación

Anexo II. Primer prototipo a papel.

En fases tempranas al proyecto, se ha elaborado un prototipo a papel de la interfaz gráfica del usuario para la aplicación.
La figura 6 muestra el prototipo de las diversas pantallas de la aplicación.

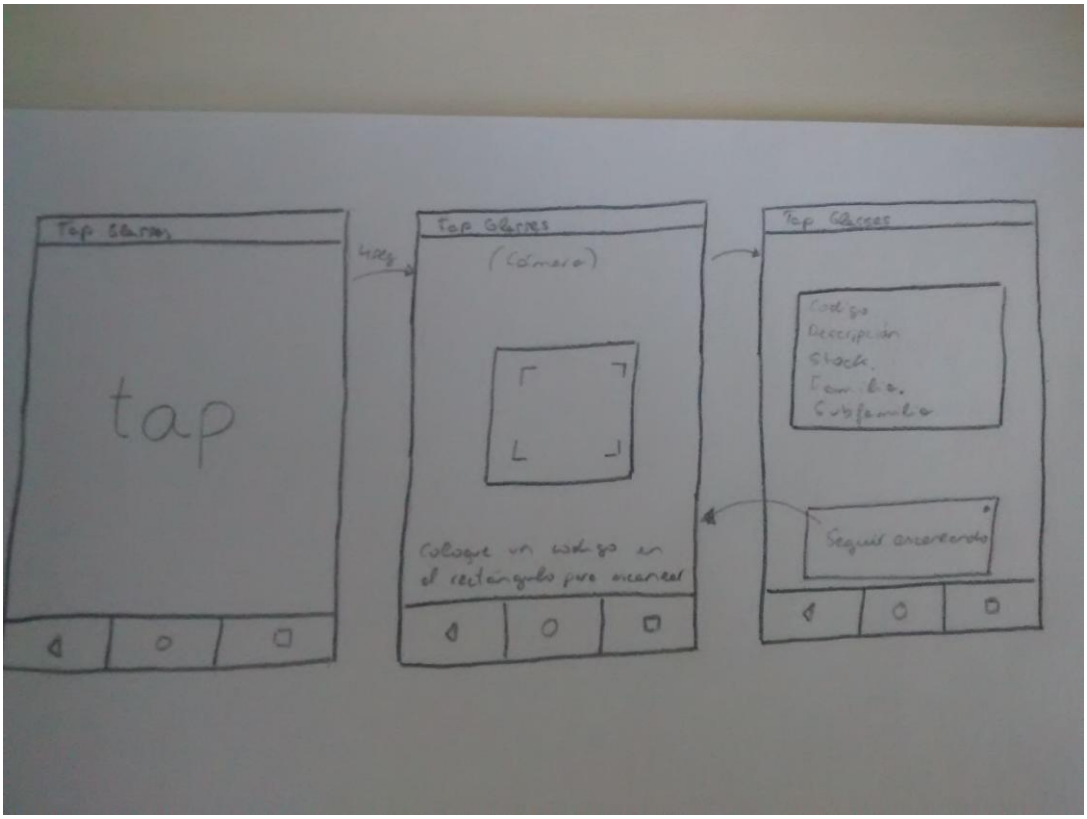


Figura 6. Primer prototipo a papel

Como se puede observar, el documento está dividido en tres posibles pantallas. En la primera de ellas se muestra una cabecera con el nombre de la aplicación y una pantalla de bienvenida, donde aparecerá el logo de la empresa.

En la segunda pantalla, a la que se pasa al transcurrir unos segundos, se muestra la visualización de la cámara con un mensaje en la parte inferior que indica: "Coloque un código para escanear". La cabecera continua igual que en la primera pantalla.

Para finalizar, la tercera pantalla se mostrará al escanear un código QR, mostrando toda la información acerca del producto en un recuadro en el medio de la pantalla, y un botón para retornar a la visualización de la cámara llamado "Seguir escaneando".

Anexo III. Primer diseño con tecnologías android.

En este anexo se muestra el primer desarrollo e implementación con las tecnologías android, un primer prototipo de la interfaz del sistema.

La figura 7 muestra la pantalla inicial de la aplicación, es decir, la bienvenida. Para la implementación de dicha pantalla se creó un fondo estándar azul claro que sirviera para el resto de pantallas y, encima de ella, el logo de Tap. Además esta pantalla se desarrolló con una imagen de tap no oficial.

Figura 7. Pantalla inical de la aplicación

La figura 8 muestra la visualización de la cámara vista desde la aplicación. Esta pantalla se muestra al invocar la librería Zxing explicada en la memoria, esta invocación se puede observar en la Imagen 4 que se muestra también a continuación.

```
Intent captura = new Intent("AppAndroid.SCAN");
captura.putExtra("SCAN_MODE", "QR_CODE_MODE");//Solo capturará códigos QR
startActivityForResult(captura, 1);
```

Imagen 4. Lanzamiento cámara.

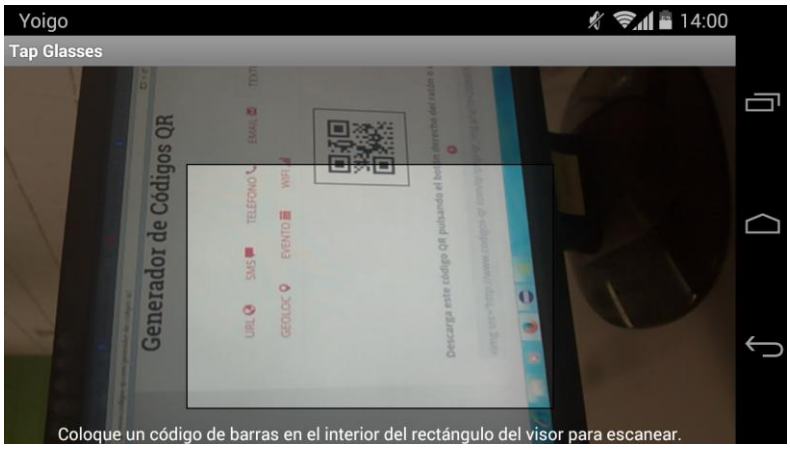


Figura 8. Visualización cámara.

La figura 9 muestra como la aplicación captura un código QR, mostrando un mensaje de "Texto encontrado".

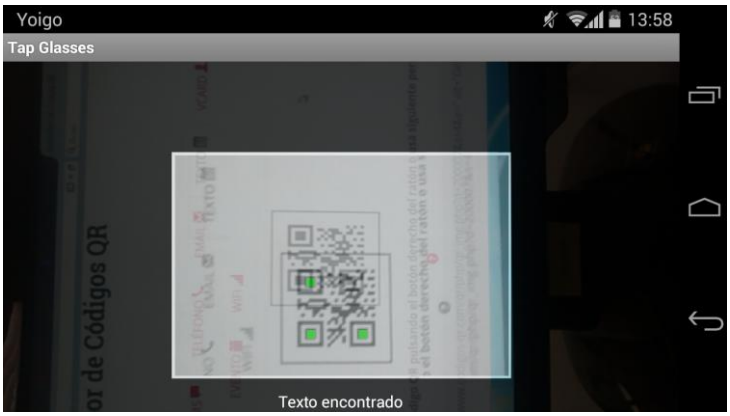


Figura 9. Texto encontrado

Una vez escaneado el código QR, la aplicación android accede directamente al método "onActivityResult" que se encarga de tratar dicho texto, tal y como se muestra en la Imagen 5.

```
public void onActivityResult(int requestCode, int resultCode, Intent intent) {
       super.onActivityResult(requestCode, resultCode, intent);
       if (requestCode == 1) {
           if (resultCode == RESULT_OK) {
               String contenido = intent.getStringExtra("SCAN_RESULT");
               String formato = intent.getStringExtra("SCAN_RESULT_FORMAT");

               tv1=(TextView)findViewById(R.id.tv1);
               tv1.setText("Cargando consulta");
               new Tarea().execute(contenido);
           } else if (resultCode == RESULT_CANCELED) {
               // Si se cancelo la captura.
               Toast toast = Toast.makeText(this, "El escaneo ha sido cancelado", Toast.LENGTH_SHORT);
               toast.show();
           }
       }
   }
```

Imagen 5. Código que trata el código QR.

Como se puede observar en la imagen, el codigo se guarda en la variable contenido, y a su vez en dicho método, lanzamos un hilo asíncrono "Tarea" que se encargará de realizar la petición de la consulta a la base de datos.

La figura 10 muestra la pantalla donde aparece toda la información acerca del producto con un botón "Finalizar", el cual se pulsará para seguir escaneando otro código. El código que define los estilos de dicha pantalla se muestra en el XML de la Imagen 6.

```
android:layout_height="match_parent"
android:clickable="false"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context="com.app.MainActivity" >

<Button
    android:id="@+id/button1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    android:layout_centerHorizontal="true"
    android:layout_marginBottom="84dp"
    android:background="@drawable/borde"
    android:clickable="false"
    android:text="@string/esperar"
    android:textSize="17sp"
    android:textStyle="bold|normal"
    android:typeface="monospace" />

<TextView
    android:id="@+id/tv1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentTop="true"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="113dp"
    android:background="@drawable/bordetextview"
    android:textStyle="bold"
    android:text="TextView" />
```

Imagen 6. XML que define pantalla Figura 10.

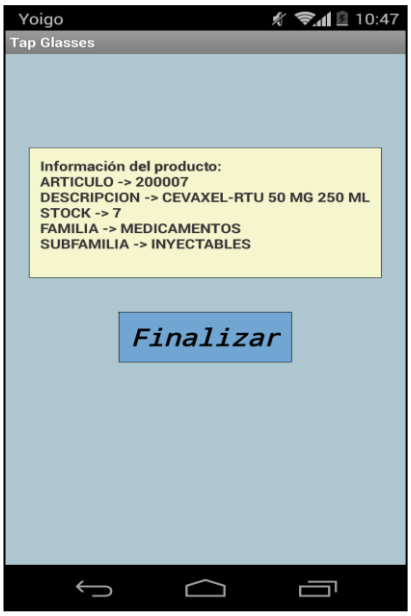


Figura 10. Pantalla con botón

La figura 11 muestra un ligero cambio en la interfaz de la pantalla final. Tras la verificación en una de las fases del proyecto, en una reunión se decidió eliminar el botón para hacer de la aplicación un nivel máximo de accesibilidad, dejando dicho cuadro de texto como un mensaje de información y haciendo que la aplicación vuelva al visor de la cámara tras unos segundos.

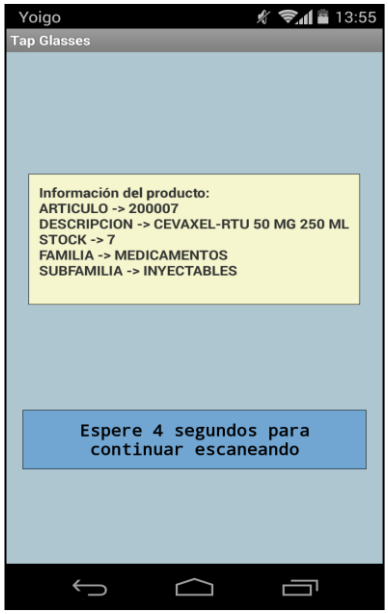


Figura 11. Interfaz sin botón

Anexo IV. Pantalla final con estilos y tipos de fuentes personalizados.

En este documento se muestran las pantallas finales de la aplicación con los estilos correspondientes y acordados con el departamento de diseño de Tap.
La figura 12 muestra la pantalla inicial con la imagen oficial proporcionada por la empresa, además se ha eliminado el fondo que se mostró en el anexo anterior, ya que no era oportuno para la aplicación.

Figura 12. Bienvenida con imagen oficial

Al mostrar esta imagen durante varios segundos, se procede a la visualización de la cámara, la cual ya se ha explicado en el anexo III y, es una pantalla que no se ha modificado con respecto al diseño final.

Una vez se escanea un código QR, la figura 13 muestra el comportamiento de la aplicación mientras se está realizando la consulta a la base de datos a través del servicio web.

Figura 13. Cargando consulta.

La figura 14 muestra la pantalla final con los estilos y tipos de letra correspondientes y acordados en la última reunión.

Figura 14. Pantalla final.

La figura 15 muestra la misma pantalla que la figura anterior, con una diferencia en el comportamiento, ya que si el sotck del producto es cero, se muestra una alerta visual para avisar al usuario de que no hay existencias de ese producto y, por lo tanto, es necesario reponerlo si el encargado lo considera oportuno.

Figura 15. Pantalla final. Stock 0

Anexo V. Diseño esquema entidad-relación.

En este anexo se muestra el esquema entidad-relación para integrar la aplicación con la base de datos de la parte del ERP que se necesita.

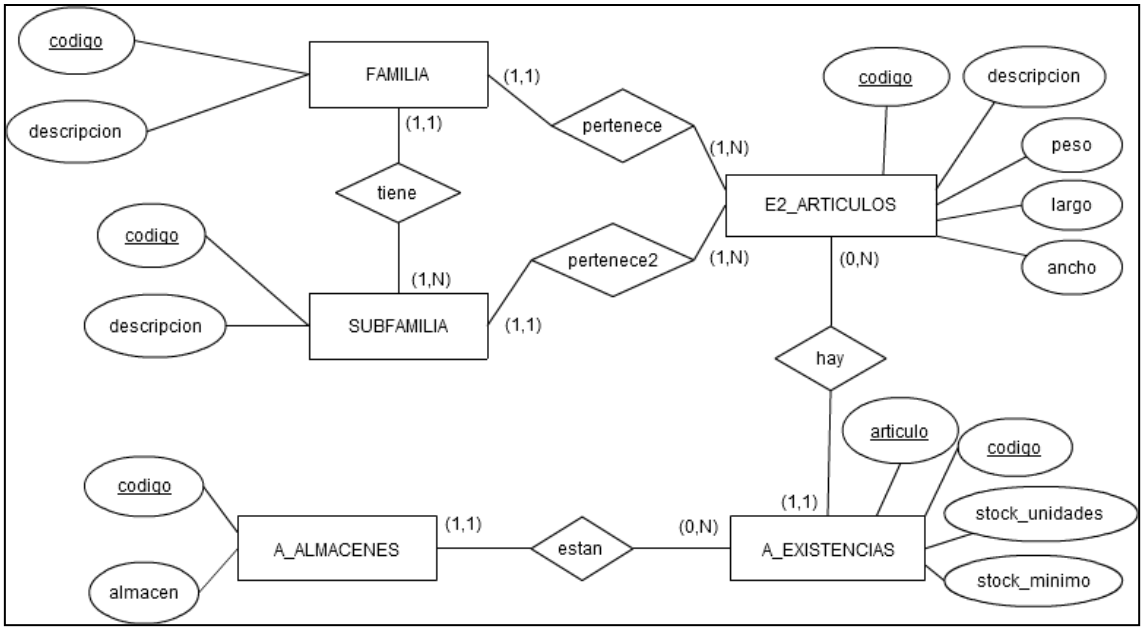


Figura 16. Esquema entidad-relación sección del ERP

Este esquema entidad-relación se ha desarrollado a partir de los documentos XML aportados por el departamento de programación del ERP EON.BS. En particular, se proporcionaron los ficheros correspondientes a dichas entidades o tablas.

En la figura 17 se muestra un ejemplo de este tipo de documentos con algunos atributos importantes.

```
<?xml version="1.0" encoding="UTF-8"?>
<Tabla>
  <NombreTabla>E2_ARTICULOS</NombreTabla>
  <FicheroDatos>E2_ARTICULOS.dat</FicheroDatos>
  <Separador>|</Separador>
  <Campos>
    <Campo>
      <Nombre>CODIGO</Nombre>
      <Tipo>NUMBER</Tipo>
      <PermiteNulo>false</PermiteNulo>
      <TamanoCampo>10</TamanoCampo>
      <TamanoDecimales>0</TamanoDecimales>
      <Posicion>1</Posicion>
    </Campo>
    <Campo>
      <Nombre>ARTICULO</Nombre>
      <Tipo>VARCHAR2</Tipo>
      <PermiteNulo>false</PermiteNulo>
      <TamanoCampo>40</TamanoCampo>
      <Posicion>2</Posicion>
    </Campo>
    <Campo>
      <Nombre>DESCRIPCION</Nombre>
      <Tipo>VARCHAR2</Tipo>
      <PermiteNulo>false</PermiteNulo>
      <TamanoCampo>70</TamanoCampo>
      <Posicion>3</Posicion>
    </Campo>
```

Figura 17. XML tabla E2_ARTICULOS

En este documento XML fue proporcionado con otros cuatro documentos más, uno para cada una de las tablas a estudiar. Con este documento XML, se pudo obtener una idea de cómo estaba estructurada la tabla de artículos. Se observan los atributos "codigo", "articulo" y "descripción". Bien pues, cabe destacar en este almacén de datos que es el ERP la diferencia entre una clave natural y una clave artificial. Para este caso en particular el código hace de clave artificial, el cual no es el código por el que se busca en la base de datos. Este atributo "codigo" se utiliza en muchas tablas de los almacenes de datos para agilizar las consultas y no tienen ningún valor informativo acerca del producto. El verdadero código del producto, el cual escanearemos con las gafas está referenciado por el atributo "articulo", el cual sí aporta información al producto, así como su descripción correspondiente.

Para el resto de tabla se realizó el mismo estudio, y se comprobó que las claves artificiales y naturales estaban presentes en todas ellas, por lo que se tuvo especial cuidado para entender y utilizar bien los atributos correspondientes.

Anexo VI. Infraestructura Web Service.

A continuación se detalla la documentación referente a la infraestructura del Web Service, ya sea en cuanto a la publicación del servicio (crearlo) como a como consumir dicho servicio (conectar).

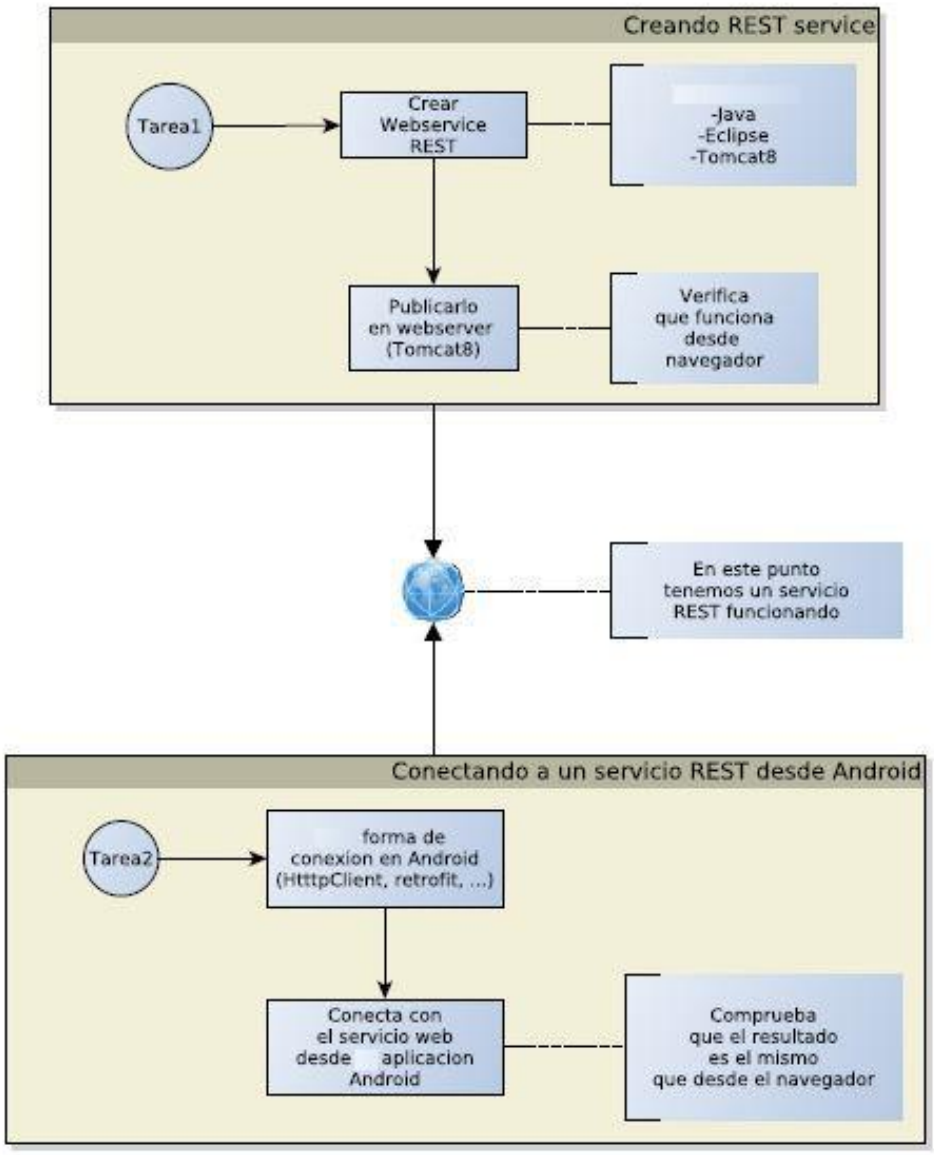


Figura 17. Esquema Web Service. Desplegar y consumir.

Anexo VII. Manual de instalación app.

En este anexo se indica como instalar la aplicación para su correcto funcionamiento. Para empezar debemos diferenciar entre la aplicación en las gafas, la cual actúa como cliente y el Servicio Web alojado en un ordenador físico de la oficina que actúa como servidor.

1. Servidor

En el caso del servidor es sencillo, una vez que se tiene el servicio web con la constancia de que funciona correctamente y desplegado en un servidor Tomcat, lo único que se debe hacer es iniciar dicho servidor. La finalidad de este servidor es que esté recibiendo peticiones constantemente y devolviendo la información correspondiente.

2. Cliente

En cuanto al cliente, una vez desarrollada completamente la aplicación, debemos extraer el fichero .apk de los archivos del proyecto de android. Mediante un gestor de archivos o compartiéndolo utilizando la nube, se debe transferir dicho archivo con terminación ".apk" a las gafas para proceder a su instalación. Por supuesto, dicho dispositivo tiene que tener conexión a internet, sino la aplicación no funcionará. Como se trata de un sistema operativo android, la instalación se realiza automática, por lo tanto no es necesario realizar nada más para que la aplicación quede instalada.

La aplicación ya está instalada y el servidor corriendo. Por lo tanto, se procede a su utilización. Se puede observar que la aplicación funciona correctamente si el dispositivo con el que nos estamos conectando está conectado a la red local de la empresa, ya que el firewall del sistema permite dichas conexiones internas.

Sin embargo, si se quiere conectar desde cualquier sitio remoto se deberá realizar una conexión vía VPN. Lo cual se explica en el siguiente anexo.

Anexo VIII. Manual de conexión al servidor con VPN.

En este anexo se explica la forma en la que un usuario se debe conectar a la red local de la empresa mediante una conexión privada virtual.

De esta forma sabemos que cualquier dispositivo android dispone de conexión VPN en Ajustes>Más>VPN. Aquí se crea una nueva conexión donde se debe introducir la dirección IP pública a la que se quiere conectar. En este caso, la 62.151.210.129, con los credenciales de un usuario que tiene acceso a la red.

Si la conexión ha sido correcta aparecerá un símbolo de una llave en la parte superior derecha de la pantalla y, por lo tanto, ya se podrá utilizar la aplicación en cualquier punto.

Bien es sabido que este tipo de conexión se realiza para un periodo de tiempo determinado, así que en el caso de que la llave deje de aparecer, simplemente con autenticarse de nuevo será suficiente para mantener la conexión.

Anexo IX. Manual de usuario.

En este anexo se explica el funcionamiento de la aplicación y la forma en la que se debe usar.

Una vez que se ha instalada la aplicación y puesto en marcha el servidor donde está desplegado el servicio web el usuario se procede a su utilización.

Como bien se puede observar en anexos anteriores disponemos de todas las herramientas y todas las pantallas con las que nos podemos encontrar. Siguiendo lo pasos de dichos anexos se supone que la conexión es correcta y el servidor funciona correctamente.

Así pues, se inicia la aplicación en las gafas donde aparece el logo de la empresa durante unos segundos y, directamente, se muestra la cámara con la que se dispone a capturar un código.

El usuario tendrá la posibilidad de tener las manos libres en todo momento para realizar lo que considere oportuno, como reponer los productos o conducir un toro mecánico por el almacén.

En el momento que el usuario mira a un código QR, la aplicación lo procesa y muestra la información acerca de dicho producto, donde el usuario tomará al decisión que cree oportuna.

En el siguiente diagrama de secuencia de la figura 18 se puede observar más claramente como interacciona la aplicación con el servidor web.

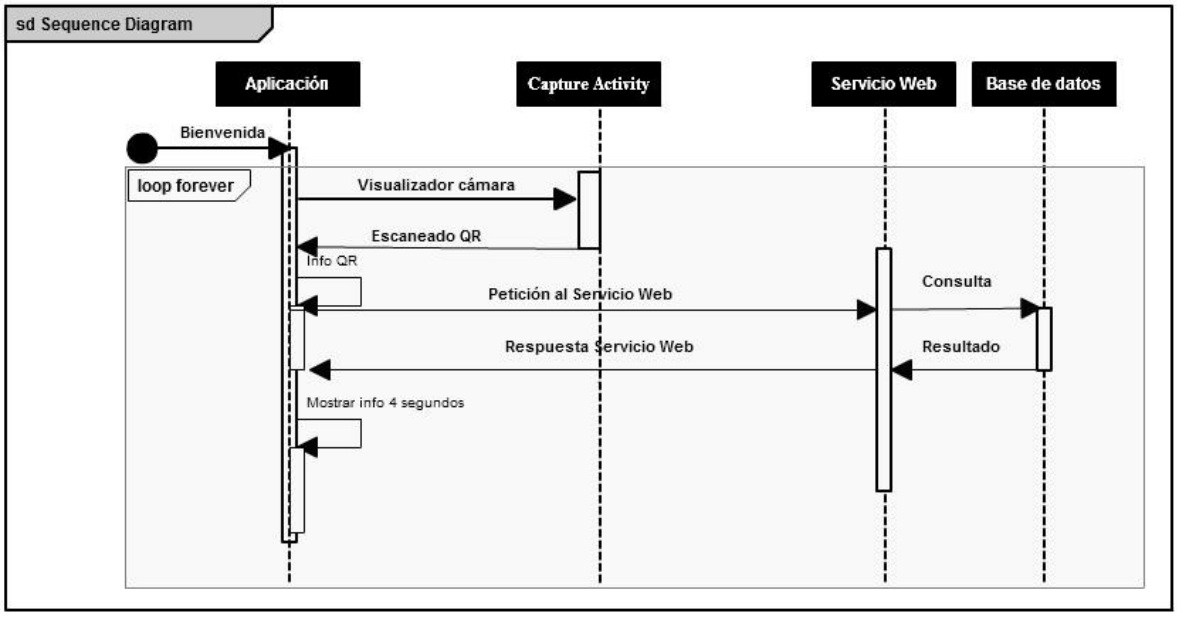


Figura 18. Diagrama de secuencia aplicación

Glosario.

HTTP: Hypertext Transfer Protocol es un protocolo de red, en concreto, del nivel de aplicación. Es usado ampliamente para las peticiones en la web. Se caracteriza por ser un protocolo sin estado.

API: Application Programming Interface, es el conjunto de funciones que ofrece un sistema para ser utilizado por otro software como una caja negra, abstrayendo los detalles de su implementación.

RESTful: Representational State Transfer, es una técnica de arquitectura de software para los sistemas web que se caracteriza por ser un protocolo sin estado y con un conjunto de operaciones bien definido, entre las cuales las más importantes son GET, POST, PUT y DELETE.

Android: Android es un sistema operativo basado en el núcleo Linux.

OJDBC: Oracle Java Database Connectivity es una API que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java, independientemente del sistema operativo donde se ejecute.