Universidad Argentina John F. Kennedy

Escuela de Sistemas

Licenciatura de Sistemas

TFI: Producto Gestión de Alimentos mediante QR.

Juan Manuel Gómez Varela

Tutor: Prof. Pablo Pandolfo

Argentina, Buenos Aires - 2014

Índice

Abstract

Mundialmente por año se desperdician gran parte de los alimentos que se producen para el consumo humano.

La insatisfacción en los parámetros estéticos del alimento, los errores logísticos en la anticipación de la demanda, los productos no consumidos anteriormente a su respectiva fecha de vencimiento y las malas prácticas en almacenamiento, constituyen las principales razones que hacen que una gran parte del desperdicio total se dé en los restaurantes, hoteles, y hogares. Además de un impacto económico la problemática supone desaprovechar importantes recursos, incluyendo agua, tierras, energía y mano de obra.

Es posible minimizar el desperdicio de alimento mediante el desarrollo de un producto de gestión que se apoye en la utilización de códigos QR en los refrigeradores, secciones de los mismos y/o ziplocs para de esa manera llevar un inventario de los alimentos y también alertar al usuario cuando estén próximos a su vencimiento.

De esta forma, es posible hacer un uso racional de los alimentos conociendo la fecha de todos los productos almacenados con solo leer un código QR con nuestro teléfono móvil o tablet.

Introducción

En todo el mundo por año se desperdician cerca de un tercio de los alimentos que se producen para el consumo humano, aproximadamente 1300 millones de toneladas[1]. Las pérdidas de alimentos afectan a la seguridad alimentaria de las personas, a la calidad y la inocuidad alimentarias, al desarrollo económico y al medioambiente. Las pérdidas de alimentos deberían mantenerse al mínimo en cualquier país, independientemente de su nivel de desarrollo económico y de la madurez de sus sistemas.

Las pérdidas de alimentos conllevan el desperdicio de recursos utilizados en la producción, como tierra, agua, energía e insumos. Producir comida que no va a consumirse supone emisiones innecesarias de CO2 además de pérdidas en el valor añadido de los alimentos producidos.

En el Área Metropolitana de Buenos Aires, Argentina, según un estudio realizado por la Facultad de Ingeniería de la UBA, los desechos alimenticios son el primer componente en el flujo de residuos sólidos (representan el 41,55 por ciento para la Ciudad Autónoma de Buenos Aires y el 37,65 para los alrededores).

Los barrios porteños que lideran el derroche son Palermo, Recoleta y Caballito (118,26; 107,55 y 90,12 toneladas por día, respectivamente), mientras que en los municipios con mayor desperdicios alimenticios son Avellaneda, Lanús y Quilmes (con el 45,61; 43,08 y 40,75 por ciento de los residuos)[2].

La reducción del despilfarro de alimentos es necesaria como una de las primeras medidas para luchar contra el hambre a la vez que contribuye a preservar los recursos naturales. Así lo asegura el equipo técnico de Nutrición y Educación Alimentaria del Ministerio de Agricultura, Ganadería y Pesca de la Nación[3].

[1] Información extraída de: Estudio “Global food losses and food waste” encargado por la FAO (Organización de las Naciones Unidas para la Alimentación y la Agricultura) al instituto sueco de alimentos y biotecnología año 2012.

2 Información extraída de Perdida y desperdicio de alimentos en America Latina y el Caribe, año 2014 FAO.

3 Banco Mundial 2014. Food Price Watch. Febrero 2014.

Desperdicio de alimentos por segmento: 28% en consumo, 28% en producción, 22% en manejo y almacenamiento, 17% en mercado y distribución, 5% durante el procesamiento.

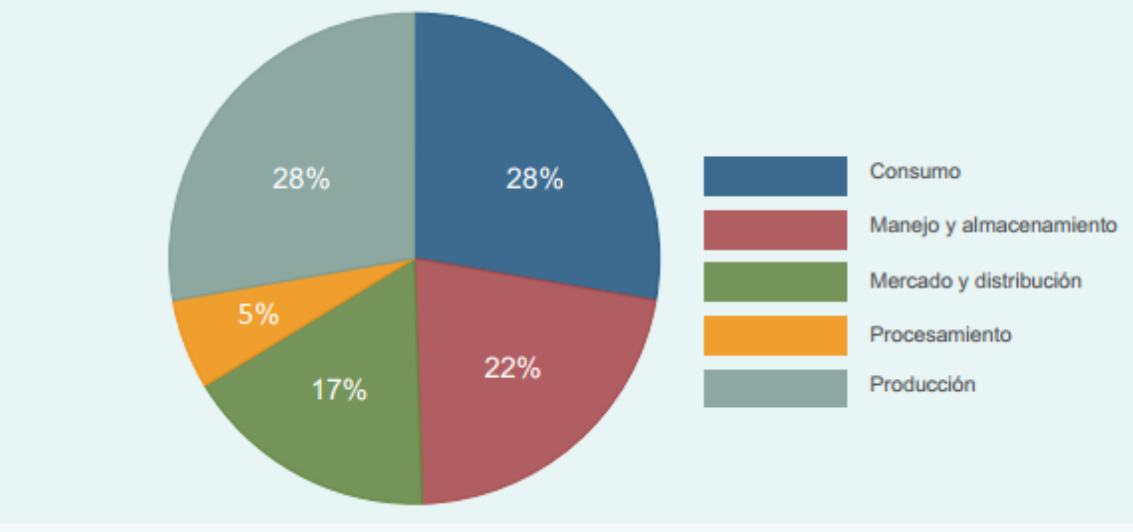


Figura 1 Pérdidas y desperdicios de alimentos por segmento[4].

4 FAO con base en banco mundial año 2014.

Objetivos

Objetivos Generales

De acuerdo con la FAO, el desperdicio que se da en la fase de consumo (28%) sucede porque los individuos no planifican sus compras, lo hacen en exceso estimulados por el marketing y la publicidad. Este trabajo propone y desarrollar un producto de software con objeto de reducir los desperdicios en este segmento de la cadena de consumo de alimentos.

El objetivo de este trabajo es demostrar que se puede reducir el desperdicio en hogares, hoteles, comercios, etc por el no consumo de productos alimenticios anteriormente a su fecha de caducidad mediante el desarrollo de un producto de gestión de alimentos que permite alertar al consumidor sobre los alimentos que se encuentran cercanos a su fecha de vencimiento a través de la utilización de tecnologías Android y códigos QR.

En este trabajo además se incluirá el análisis, diseño, desarrollo, implementación y testing del producto de gestión que permitirá a los usuarios con su teléfono móvil o tablet con la simple actividad de leer un código QR conocer las fechas de vencimiento de sus alimentos.

Objetivos Específicos

La construcción de este producto de gestión de alimentos brindara a los consumidores la posibilidad de:

- Consultar fecha de vencimiento de sus productos mediante QR.
- Consultar fecha de vencimiento de sus productos mediante aplicación mobile.
- Realizar alta de productos.
- Realizar modificación de fecha en los productos.
- Eliminar productos en la aplicación por ya haber sido consumidos o encontrarse caducos.
- Recibir alertas y mensajes de precaución vía mail.
- Configurar idioma de la aplicación.
- Configurar nivel de alertas.

Esto beneficiara al consumidor ya que podrá planificar sus compras dependiendo del estado de sus productos al igual que evitar consumir productos ya vencidos así como obtener información para saber que productos están próximos a su fecha de vencimiento, implicando esto un ahorro económico para el consumidor y mas globalmente aportando

Implicando de esta manera un ahorro económico para el consumidor y un beneficio ecológico para la sociedad.

Mapa Conceptual

1 Marco Teórico

1.1 Códigos QR (Quick Response)

Un código QR (*quick response code*, código de respuesta rápida) es un módulo útil para almacenar información en una matriz de puntos o un código de barras bidimensional creado en 1994 por la compañía japonesa Denso Wave, subsidiaria de Toyota. Se caracteriza por los tres cuadrados que se encuentran en las esquinas y que permiten detectar la posición del código al lector. La sigla «QR» viene de la frase inglesa «Quick Response» («Respuesta Rápida» en español), pues los creadores (un equipo de dos personas en Denso Wave, dirigido por Masahiro Hara)[5] tenían como objetivo que el código permitiera que su contenido se leyera a alta velocidad. Los códigos QR son muy comunes en Japón y de hecho son el código bidimensional más popular en ese país.

Figura 2 Ejemplo de código QR especificación ISO/IEC 18004:2006 [6]

1.1.1 Características

Aunque inicialmente se usó para registrar repuestos en el área de la fabricación de vehículos, hoy los códigos QR se usan para administración de inventarios en una gran variedad de industrias. La inclusión de software que lee códigos QR en teléfonos móviles, ha permitido nuevos usos orientados al consumidor, que se manifiestan en comodidades como el dejar de tener que introducir datos de forma manual en los teléfonos. Las direcciones y los URLs se están volviendo cada vez más comunes en revistas y anuncios. El agregado de códigos QR en tarjetas de presentación también se está haciendo común, simplificando en gran medida la tarea de introducir detalles individuales de un nuevo cliente en la agenda de un teléfono móvil.

Los códigos QR también pueden leerse desde PC, smartphone o tableta mediante dispositivos de captura de imagen, como puede ser un escáner o la cámara de fotos, programas que lean los datos QR y una conexión a Internet para las direcciones web.

El estándar japonés para códigos QR (JIS X 0510) fue publicado en enero de 1998 y su correspondiente estándar internacional ISO (ISO/IEC18004) fue aprobado en junio de 2000.

[5] Información extraída de pagina official página oficial de Denso Wave, http://www.qrcode.com/en/

[6] **¡Error! No se encuentra el origen de la referencia.**

Un detalle importante sobre el código QR es que, a diferencia de otros formatos de códigos de barras bidimensionales como el BIDI, su código es abierto y sus derechos de patente (propiedad de Denso Wave) no son ejercidos, es libre utilizar y generar códigos QR, la única condición privativa es sobre las palabras “QR Code” donde para mencionarlas hay que citar “QR Code is registered trademark of DENSO WAVE INCORPORATED”.

1.1.1.1 Corrección de errores de los códigos QR

Incluso si la imagen de un código QR es parcialmente borrada o dañada, la información podrá ser recuperada. Esa recuperación dependerá de la cantidad de daño o deterioro, hay cuatro niveles de corrección de errores disponibles:

- Nivel L 7%
- Nivel M 15%
- Nivel Q 25%
- Nivel H 30%

1.1.1.2 Independencia de la orientación del código QR

Los códigos QR pueden ser leídos desde cualquier dirección, soportan tanto la rotación como la reflexión. Los patrones de detección de posición a las tres esquinas de los códigos QR garantizan una alta velocidad de lectura independientemente de en que orientación se encuentre el código evitando además los efectos negativos de la interferencia de fondo.

1.1.2 Estructura

Cada código QR es un cuadrado regular constituido por varios módulos cuadrados incluyendo una región encriptada, información de la versión, información del formato, una sección de corrección de errores y datos. Los patrones requeridos son: el de posición, el de alineamiento y el de sincronización.

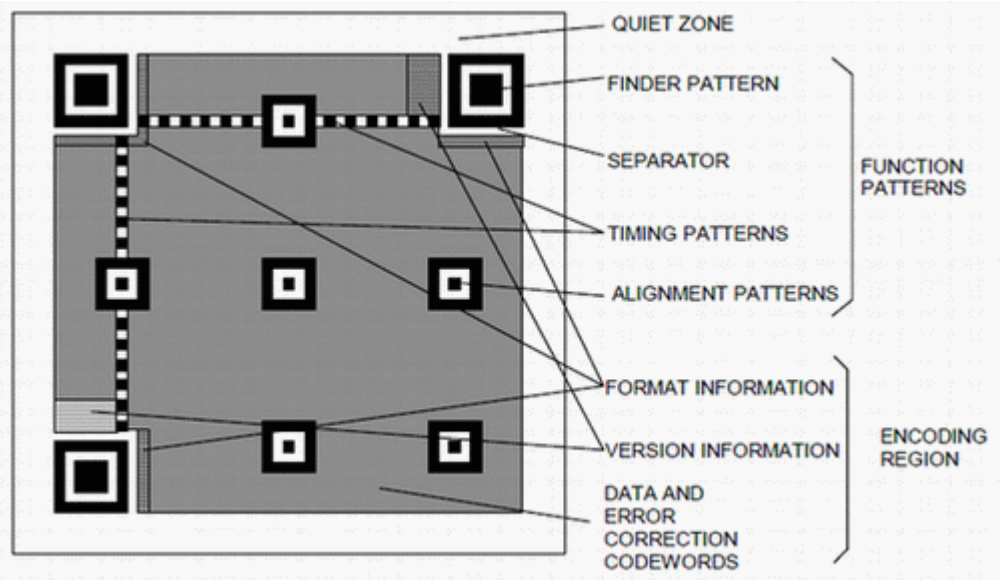


Figura 3 QR code ISO/IEC 18004 2D Barcode structure.

1.1.3 Tipos de código QR

1.1.3.1 Modelo 1

Es el código QR original, un código capaz de contener 1.167 numerales, su mayor versión es la número 14 (73 x 73 módulos).

QR Code Model 1

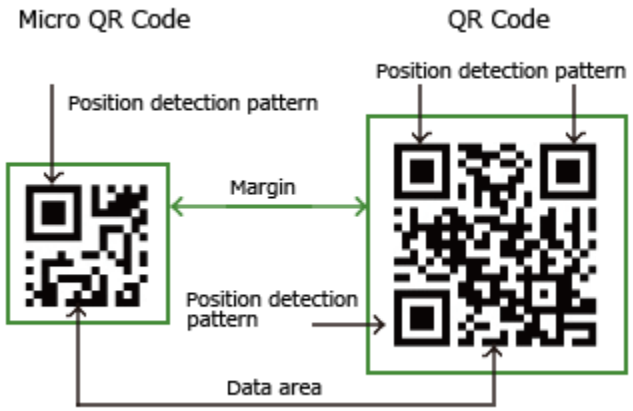
1.1.3.2 Modelo 2

Creado a partir del modelo 1, este tipo de código QR puede ser leído incluso si es dañado o distorsionado de alguna manera. Este tipo de código QR puede contener arriba de 7.089 numerales, su máxima versión es la 40 (177 x 177 modulos).

QR Code Model 2

1.1.3.3 Micro QR

El micro código QR es una versión más pequeña del estándar del código QR y está diseñado para aplicaciones que tengan una habilidad menor en el manejo de escaneos grandes. Hay diferentes versiones de micro código QR. La más grande de ellas puede contener hasta 35 caracteres.



1.1.3.4iQR Code

iQR Code es una matrix de dos dimensiones que permite una lectura fácil de su posición y tamaño. Este código permite una amplia gama de tamaños desde incluso más pequeños que el micro QR hasta incluso más grandes que el código QR del modelo 1 y 2

El código iQR puede contener una mayor cantidad de información que el código QR tradicional. Un código iQR del mismo tamaño que un QR existente puede contener 80% más de información que la segunda.

El tamaño mínimo se compone de 11 por 11 módulos para el código QR regular. Por otro lado, es 9 por 9 módulos para el código iQR. El área requerida para este código se puede reducir hasta en un 60% aproximadamente en comparación con el código regular.

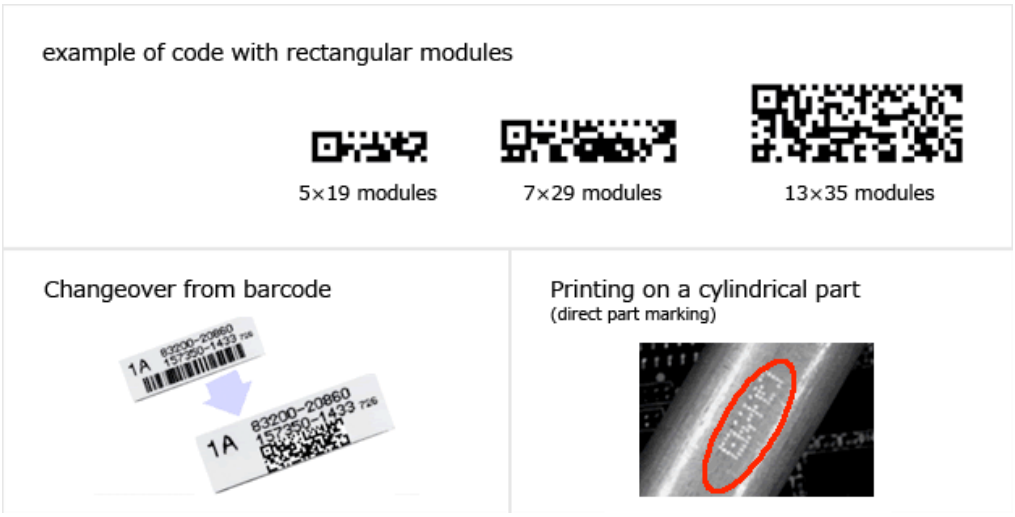


Figura 4 extraída de pagina web http://www.qrcode.com de su creador Denso Wave

1.1.3.5 SQRC Code

Es un código QR que posee funciones de restricción de lectura. Puede ser usada para almacenar información privada o manejar información interna empresarial. En apariencia es similar a un código QR regular.

Los códigos SQRC pueden ser leídos solamente por tipos específicos de scanners, es decir que se configura el par código QR/Scanner para que una partida de códigos sean leídos solamente por una partida particular de scanners.

Figura 5 extraída de pagina web http://www.qrcode.com de su creador Denso Wave

1.1.3.6 LogoQ

Es un código QR que incorpora altos niveles de características de diseño como ser ilustraciones, letras o logos sin que sus propiedades de lectura se vean comprometidas.

Figura 6 extraída de pagina web http://www.qrcode.com de su creador Denso Wave

1.2 Android

1.2.1 El sistema Operativo Android

Android es un sistema operativo móvil basado en el kernel de Linux y actualmente desarrollado por Google. Diseñado principalmente para dispositivos móviles con pantalla táctil, como teléfonos inteligentes o tabletas, y también para relojes inteligentes, televisores y automóviles. El sistema operativo utiliza entradas táctiles que corresponden en términos generales a las acciones del mundo real, como deslizar, tocar, arrastrar para manipular objetos en la pantalla y un teclado virtual. A pesar de estar diseñado principalmente para la entrada de la pantalla táctil, también se ha utilizado en las consolas de juegos, cámaras digitales y otros aparatos electrónicos.

Android es el sistema operativo para moviles más popular desde el año 2013. Los dispositivos con Android han sido más vendidos que los que poseen sistemas operativos Windows, iOs y Mac OS juntos.

El código fuente de Android es lanzado por Google bajo licencia open source.

1.2.2 Características

1.2.2.1 Diseño de dispositivo

La plataforma es adaptable a pantallas de mayor resolución, VGA, biblioteca de gráficos 2D, biblioteca de gráficos 3D basada en las especificaciones de la OpenGL ES 2.0 y diseño de teléfonos tradicionales

1.2.2.2 Almacenamiento

SQLite, una base de datos liviana, que es usada para propósitos de almacenamiento de datos.

1.2.2.3 Conectividad

Android soporta las siguientes tecnologías de conectividad: GSM/EDGE,IDEN, CDMA, EV-DO, UMTS, Bluetooth,Wi-Fi, LTE, HSDPA, HSPA+, NFC yWiMAX.GPRS,UMTS y HSDPA+.

1.2.2.4 Mensajería

SMS y MMS son formas de mensajería, incluyendo mensajería de texto y ahora la Android Cloud to Device Messaging Framework (C2DM) es parte del servicio de Push Messaging de Android.

1.2.2.5 Navegador web

El navegador web incluido en Android está basado en el motor de renderizado de código abierto WebKit, emparejado con el motor JavaScript V8 de Google Chrome. El navegador por defecto de Ice Cream Sandwich obtiene una puntuación de 100/100 en el test Acid3.

1.2.2.6 Soporte de Java

Aunque la mayoría de las aplicaciones están escritas en Java, no hay una máquina virtual Java en la plataforma. Elbytecode Java no es ejecutado, sino que primero se compila en un ejecutable Dalvik y corre en la Máquina Virtual Dalvik. Dalvik es una máquina virtual especializada, diseñada específicamente para Android y optimizada para dispositivos móviles que funcionan con batería y que tienen memoria y procesador limitados. El soporte para J2ME puede ser agregado mediante aplicaciones de terceros como el J2ME MIDP Runner.

1.2.2.7 Soporte multimedia

Android soporta los siguientes formatos
multimedia: WebM, H.263, H.264 (en3GP o MP4), MPEG-4 SP, AMR, AMR-WB (en un contenedor 3GP), AAC, HE-AAC (en contenedores MP4 o 3GP),MP3, MIDI, Ogg
Vorbis, WAV, JPEG,PNG, GIF y BMP.

1.2.2.8 Soporte para streaming

Streaming RTP/RTSP (3GPP PSS, ISMA), descarga progresiva de HTML (HTML5 <video> tag). Adobe Flash Streaming (RTMP) es soportado mediante el Adobe Flash Player. Se planea el soporte de Microsoft Smooth Streaming con el port de Silverlight a Android. Adobe Flash HTTP Dynamic Streaming estará disponible mediante una actualización de Adobe Flash Player.

1.2.2.9 Soporte para hardware adicional

Android soporta cámaras de fotos, de vídeo, pantallas táctiles, GPS, acelerómetros, giroscopios, magnetómetros, sensores de proximidad y de presión, sensores de luz, gamepad, termómetro, aceleración por GPU 2D y 3D.

1.2.2.10 Entorno de desarrollo

Incluye un emulador de dispositivos, herramientas para depuración de memoria y análisis del rendimiento del software. El entorno de desarrollo integrado es Eclipse (actualmente 3.4, 3.5 o 3.6) usando el plugin de Herramientas de Desarrollo de Android.

1.2.2.11 Google Play

Google Play es un catálogo de aplicaciones gratuitas o de pago en el que pueden ser descargadas e instaladas en dispositivos Android sin la necesidad de un PC.

1.2.2.12 Multi-táctil

Android tiene soporte nativo para pantallas capacitivas con soporte multi-táctil que inicialmente hicieron su aparición en dispositivos como el HTC Hero. La funcionalidad fue originalmente desactivada a nivel de kernel (posiblemente para evitar infringir patentes de otras compañías). Más tarde, Google publicó una actualización para el Nexus One y el Motorola Droidque activa el soporte multi-táctil de forma nativa.

1.2.2.13 Bluetooth

El soporte para A2DF y AVRCP fue agregado en la versión 1.5;49 el envío de archivos (OPP) y la exploración del directorio telefónico fueron agregados en la versión 2.0; y el marcado por voz junto con el envío de contactos entre teléfonos lo fueron en la versión 2.2.</ref> Los cambios incluyeron.

1.2.2.14 Multitarea

Multitarea real de aplicaciones está disponible, es decir, las aplicaciones que no estén ejecutándose en primer plano reciben ciclos de reloj.

1.2.3 Arquitectura

1.2.3.1 Aplicaciones

Las aplicaciones base incluyen un cliente de correo electrónico, programa de SMS, calendario, mapas, navegador, contactos y otros. Todas las aplicaciones están escritas en lenguaje de programación Java.

1.2.3.2 Marco de trabajo de aplicaciones

Los desarrolladores tienen acceso completo a los mismos APIs del framework usados por las aplicaciones base. La arquitectura está diseñada para simplificar la reutilización de componentes; cualquier aplicación puede publicar sus capacidades y cualquier otra aplicación puede luego hacer uso de esas capacidades (sujeto a reglas de seguridad del framework). Este mismo mecanismo permite que los componentes sean reemplazados por el usuario.

1.2.3.3Bibliotecas

Android incluye un conjunto de bibliotecas de C/C++ usadas por varios componentes del sistema. Estas características se exponen a los desarrolladores a través del marco de trabajo de aplicaciones de Android; algunas son: System C library (implementación biblioteca C estándar), bibliotecas de medios, bibliotecas de gráficos, 3D y SQLite, entre otras.

1.2.3.4 Runtime de Android

Android incluye un set de bibliotecas base que proporcionan la mayor parte de las funciones disponibles en las bibliotecas base del lenguaje Java. Cada aplicación Android corre su propio proceso, con su propia instancia de la máquina virtual Dalvik. Dalvik ha sido escrito de forma que un dispositivo puede correr múltiples máquinas virtuales de forma eficiente. Dalvik ejecuta archivos en el formato Dalvik Executable (.dex), el cual está optimizado para memoria mínima. La Máquina Virtual está basada en registros y corre clases compiladas por el compilador de Java que han sido transformadas al formato.dex por la herramienta incluida "dx".

1.2.3.5Núcleo Linux

Android depende de Linux para los servicios base del sistema como seguridad, gestión de memoria, gestión de procesos, pila de red y modelo de controladores. El núcleo también actúa como una capa de abstracción entre el hardware y el resto de la pila de software.

1.3 Android SDK

Android Software Develpment es el proceso por el cual las nuevas aplicaciones son creadas para el sistema operativo Android. Las aplicaciones para Android son usualmente desarrolladas en el leguaje de programación Java usandio el SDK de Android.

Hasta Julio del 2013 mas de 1 millon de aplicaciones han sido desarrolladas para Android las cuales han obtenido 25 billones de descargas.

El kit de desarrollo de software para Android incluye un conjunto completo de herramientas de desarrollo. Este kit incluye un depurador, bibliotecas, una consola o terminal, un emulador de sistema operativo Android, emulador de dispositivos móviles, ejemplos de aplicaciones, documentación, tutoriales.

Las aplicaciones de Android son empaquetadas en formato .apk y almacenadas bajo la carpeta /data/app. Estos paquetes son instalados a través del puente de depuración de Android o ADB (Android Debug Bridge).

1.4 Java

1.4.1 Concepto

Java es un lenguaje de programación de propósito general, concurrente, orientado a objetos y basado en clases que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Su intención es permitir que los desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo (conocido en inglés como WORA, o "write once, run anywhere"), lo que quiere decir que el código que es ejecutado en una plataforma no tiene que ser recompilado para correr en otra. Java es, a partir de 2012, uno de los lenguajes de programación más populares en uso, particularmente para aplicaciones de cliente-servidor de web, con unos 10 millones de usuarios reportados.

El lenguaje de programación Java fue originalmente desarrollado por James Gosling de Sun Microsystems (la cual fue adquirida por la compañía Oracle) y publicado en 1995 como un componente fundamental de la plataforma Java de Sun Microsystems. Su sintaxis deriva en gran medida de C y C++, pero tiene menos utilidades de bajo nivel que cualquiera de ellos. Las aplicaciones de Java son generalmente compiladas a bytecode (clase Java) que puede

ejecutarse en cualquier máquina virtual Java (JVM) sin importar la arquitectura de la computadora subyacente.

La compañía Sun desarrolló la implementación de referencia original para los compiladores de Java, máquinas virtuales, y librerías de clases en 1991 y las publicó por primera vez en 1995. A partir de mayo de 2007, en cumplimiento con las especificaciones del Proceso de la Comunidad Java, Sun volvió a licenciar la mayoría de sus tecnologías de Java bajo la Licencia Pública General de GNU. Otros también han desarrollado implementaciones alternas a estas tecnologías de Sun, tales como el Compilador de Java de GNU y el GNU Classpath.

1.4.2 Características

1.4.2.1 Lenguaje totalmente orientado a Objetos

Todos los conceptos en los que se apoya esta técnica, encapsulación, herencia, polimorfismo, etc., están presentes en Java.

1.4.2.2 Disponibilidad de un amplio conjunto de bibliotecas

Como ya se mencionó anteriormente, Java es algo más que un lenguaje. La programación de aplicaciones con Java se basa no solo en el empleo del juego de instrucciones que componen el lenguaje, sino, fundamentalmente, en la posibilidad de utilizar el amplísimo conjunto de clases que Sun pone a disposición del programador y con las cuales es posible realizar prácticamente cualquier tipo de aplicación.

1.4.2.3 Lenguaje simple

Java posee una curva de aprendizaje muy rápida. Resulta relativamente sencillo escribir applets interesantes desde el principio. Todos aquellos familiarizados con C++ encontrarán que Java es más sencillo, ya que se han eliminado ciertas características, como los punteros. Debido a su semejanza con C y C++, y dado que la mayoría de la gente los conoce aunque sea de forma elemental, resulta muy fácil aprender Java. Los programadores experimentados en C++ pueden migrar muy rápidamente a Java y ser productivos en poco tiempo.

1.4.2.4 Distribuido

Java proporciona una colección de clases para su uso en aplicaciones de red, que permiten abrir sockets y establecer y aceptar conexiones con servidores o clientes remotos, facilitando así la creación de aplicaciones distribuidas.

1.4.2.5 Interpretado y compilado a la vez

Java es compilado, en la medida en que su código fuente se transforma en una especie de código máquina, los bytecodes, semejantes a las instrucciones de ensamblador. Por otra parte,

es interpretado, ya que los bytecodes se pueden ejecutar directamente sobre cualquier máquina a la cual se hayan portado el intérprete y el sistema de ejecución en tiempo real (run-time).

1.4.2.6 Robusto

Java fue diseñado para crear software altamente fiable. Para ello proporciona numerosas comprobaciones en compilación y en tiempo de ejecución. Sus características de memoria liberan a los programadores de una familia entera de errores (la aritmética de punteros), ya que se ha prescindido por completo de los punteros, y la recolección de basura elimina la necesidad de liberación explícita de memoria.

1.4.2.7 Seguro

Dada la naturaleza distribuida de Java, donde las applets se bajan desde cualquier punto de la Red, la seguridad se impuso como una necesidad de vital importancia. A nadie le gustaría ejecutar en su ordenador programas con acceso total a su sistema, procedentes de fuentes desconocidas. Así que se implementaron barreras de seguridad en el lenguaje y en el sistema de ejecución en tiempo real.

1.4.2.8 Indiferente a la arquitectura

Java está diseñado para soportar aplicaciones que serán ejecutadas en los más variados entornos de red, desde Unix a Windows Nt, pasando por Mac y estaciones de trabajo, sobre arquitecturas distintas y con sistemas operativos diversos. Para acomodar requisitos de ejecución tan diversos o variopintos, el compilador de Java genera bytecodes: un formato intermedio indiferente a la arquitectura diseñado para transportar el código eficientemente a múltiples plataformas hardware y software. El resto de problemas los soluciona el intérprete de Java.

1.4.2.9 Portable

La indiferencia a la arquitectura representa sólo una parte de su portabilidad. Además, Java especifica los tamaños de sus tipos de datos básicos y el comportamiento de sus operadores aritméticos, de manera que los programas son iguales en todas las plataformas. Estas dos últimas características se conocen como la Máquina Virtual Java (JVM).

1.4.2.10 Multihebra

Hoy en día ya se ven como terriblemente limitadas las aplicaciones que sólo pueden ejecutar una acción a la vez. Java soporta sincronización de múltiples hilos de ejecución (multithreading) a nivel de lenguaje, especialmente útiles en la creación de aplicaciones de red distribuidas. Así, mientras un hilo se encarga de la comunicación, otro puede interactuar con el usuario mientras otro presenta una animación en pantalla y otro realiza cálculos.

1.4.2.11 Dinámico

El lenguaje Java y su sistema de ejecución en tiempo real son dinámicos en la fase de enlazado. Las clases sólo se enlazan a medida que son necesitadas. Se pueden enlazar nuevos módulos de código bajo demanda, procedente de fuentes muy variadas, incluso desde la Red.

1.5 Patrón de arquitectura de software modelo-vista-controlador

1.5.1 Introducción

El patrón de arquitectura MVC (Modelo Vista Controlador) es un patrón que define la organización independiente del Modelo (Objetos de Negocio), la Vista (interfaz con el usuario u otro sistema) y el Controlador (controlador del workflow de la aplicación).

De esta forma, dividimos el sistema en tres capas donde, como explicaremos más adelante, tenemos la encapsulación de los datos, la interfaz o vista por otro y por último la lógica interna o controlador.

1.5.2 Estructura

El patrón de arquitectura "modelo vista controlador", es una filosofía de diseño de aplicaciones, compuesta por:

Modelo:

- Contiene el núcleo de la funcionalidad (dominio) de la aplicación.
- Encapsula el estado de la aplicación.
- No sabe nada / independiente del Controlador y la Vista.

Vista

- Es la presentación del Modelo.
- Puede acceder al Modelo pero nunca cambiar su estado.
- Puede ser notificada cuando hay un cambio de estado en el Modelo.

Controlador

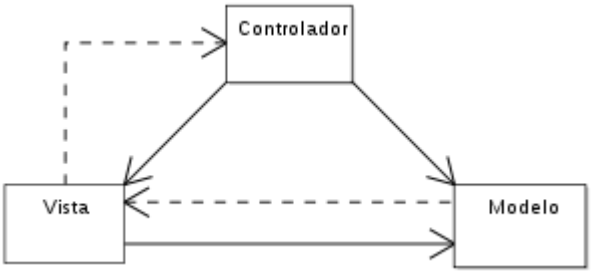
- Reacciona a la petición del Cliente, ejecutando la acción adecuada y creando el modelo pertinente

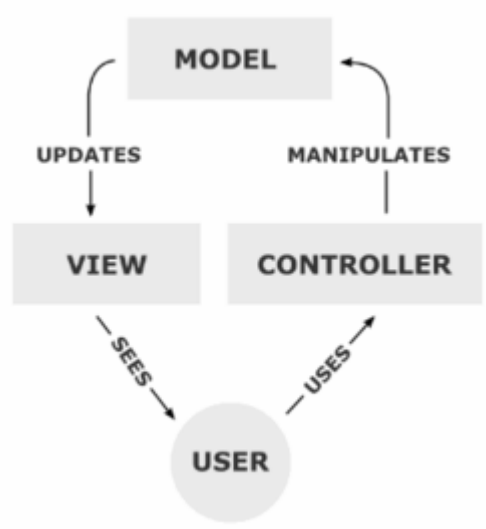
Para entender cómo funciona nuestro patrón Modelo vista controlador, se debe entender la división a través del conjunto de estos tres elementos y como estos componentes se comunican unos con los otros y con otras vistas y controladores externos a el modelo principal. Para ello, es importante saber que el controlador interpreta las entradas del usuario (tanto teclado como el ratón), enviado el mensaje de acción al modelo y a la vista para que se proceda con los cambios que se consideren adecuados

1.5.3 Comunicación

El modelo, la vista y el controlador deben comunicarse de una manera estable los unos con los otros, de manera que sea coherente con las iteraciones que el usuario realizara. Como es lógico la comunicación entre la vista y el controlador es bastante básica pues están diseñados para operar juntos, pero los modelos se comunican de una manera diferente, un poco más sutil

1.5.4 Diagrama





1.6 MySQL

MySQL es un sistema de gestión de bases de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones. MySQL AB —desde enero de 2008 una subsidiaria de Sun Microsystems y ésta a su vez de Oracle Corporation desde abril de 2009— desarrolla MySQL como software libre en un esquema de licenciamiento dual.

Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso. Está desarrollado en su mayor parte en ANSI C.

Al contrario de proyectos como Apache, donde el software es desarrollado por una comunidad pública y los derechos de autor del código están en poder del autor individual, MySQL es patrocinado por una empresa privada, que posee el copyright de la mayor parte del código. Esto es lo que posibilita el esquema de licenciamiento anteriormente mencionado. Además de la venta de licencias privativas, la compañía ofrece soporte y servicios. Para sus operaciones contratan trabajadores alrededor del mundo que colaboran vía Internet.

MySQL es usado por muchos sitios web grandes y populares, como Wikipedia, Google (aunque no para búsquedas), Facebook, Twitter, Flickr y YouTube.

1.6.1 Aplicaciones

MySQL es muy utilizado en aplicaciones web, como Drupal o phpBB, en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece en combinación con MySQL.

MySQL es una base de datos muy rápida en la lectura cuando utiliza el motor no transaccional MyISAM, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones. Sea cual sea el entorno en el que va a utilizar MySQL, es importante monitorizar de antemano el rendimiento para detectar y corregir errores tanto de SQL como de programación

1.6.2 Características

Inicialmente, MySQL carecía de elementos considerados esenciales en las bases de datos relacionales, tales como integridad referencial y transacciones. A pesar de ello, atrajo a los desarrolladores de páginas web con contenido dinámico, justamente por su simplicidad.

Poco a poco los elementos de los que carecía MySQL están siendo incorporados tanto por desarrollos internos, como por desarrolladores de software libre. Entre las características disponibles en las últimas versiones se puede destacar:

- Amplio subconjunto del lenguaje SQL. Algunas extensiones son incluidas igualmente.
- Disponibilidad en gran cantidad de plataformas y sistemas.
- Posibilidad de selección de mecanismos de almacenamiento que ofrecen diferentes velocidades de operación, soporte físico, capacidad, distribución geográfica, transacciones.
- Transacciones y claves foráneas.
- Conectividad segura.
- Replicación.
- Búsqueda e indexación de campos de texto.

2 Estado del Arte

La tecnología podría ser la solución para evitar que se pierda entre el 30 y 50 por ciento de la producción mundial de alimentos a consecuencia de las deficiencias en los métodos de cosecha, almacenamiento, distribución y consumo.

Las propuestas tecnológicas van desde programas informáticos para monitorear datos de los cultivos y el clima hasta dispositivos que recogen información en el estomago de las vacas.

Entre todo este rango de productos tecnológicos se destacan los siguientes.

2.1 Envases microbioticos

Investigadores del Instituto Tecnológico de Sonora (ITSON) desarrollan envases activos que permitirán conservar los alimentos hasta en un 50 por ciento más, debido a su acción antioxidante.

El envase activo es una denominación que obtienen algunos envolventes de alimentos que cumplen con alguna función adicional al empaquetado. Los productos diseñados en el ITSON fungen como agente antioxidante, y permiten un mayor almacenamiento de productos susceptibles a la oxidación.

Mediante esta tecnología, los alimentos que contienen alto índice de lípidos (grasas) quedan protegidos ante factores ambientales que provocan su descomposición, como la luz, el oxígeno o la temperatura.

Figura 7 extraída de pagina web http://www.mtycic.org/ de su creador el Instituto de innovación y transferencia de tecnología.

2.2 Aplicación The climate corporation

Una de las soluciones al problema de pérdida de alimentos es la plataforma de tecnología 'The Climate Corporation' que combina el monitoreo hiperlocal del clima con datos agronómicos

para ayudar a los agricultores a evitar desperdicios mediante la toma de decisiones operativas y financieras.

Los cálculos que realiza la compañía permiten a los agricultores evaluar cuándo sembrar y cosechar sus cultivos de modo que se logre mejorar el rendimiento agrícola. La empresa de propiedad de la compañía agrícola Monsanto tiene un banco de datos de mediciones del clima en miles de lugares en el mundo.

Asimismo, cuenta con alrededor de 150.000 millones de observaciones de suelos y simulaciones de clima con lo que ofrece a los agricultores pronósticos de temperatura, lluvia y viento en áreas específicas.

Figura 8 extraída de pagina web http://www.climate.com de su creador Monsanto.

2.2.1 Características de la aplicación:

- Aporta los datos meteorológicos, de suelo y de cultivos a nivel de campo utilizando estaciones de clima, radares y satélites.
- Mejor seguimiento en línea y herramientas de exploración.
- Mejora sus decisiones de producción.
- El campo y la información meteorológica actualizada a medida que pasa.
- Disponible en cualquier dispositivo.

2.3 Aplicación Food waste diary

Food waste diary ayuda a mantener un registro de los alimentos que son desperdiciados, se pueden especificar las razones por las cuales los productos han sido desperdiciados y catalogar que tipo de comida es. Tambien se puede opcionalmente indicar el precio que se ha pagado por un producto o escribir un comentario. La aplicación ayuda a entender y ver que es lo que se ha tirado mes a mes.

Figura 9 extraída de pagina web https://play.google.com/ publicada por sus autores Christoph Fischer, Eva Ganglbauer y Georg Molzer .

2.3.1 Caracteristicas

- Posibilidad de registrar la comida que es desperdiciada.
- Seguimiento del desperdicio de alimentos según su tipo y motivo o razón, opcionalmente se puede añadir su costo e historia.
- Calculo de cuanto dinero se va desperdiciando.
- Historial de productos desperdiciados en forma de lista y también con graficos de torta y de barras

3 Desarrollo

A continuación se presentara el análisis, diseño, desarrollo, implementación y testing de un producto de gestión de alimentos mediante QR cuyos objetivos generales y específicos se encuentran detallados a partir de la página 7 de este trabajo.
Sobre las tecnologías utilizadas para el desarrollo un detalle importante es que los códigos QR son de código abierto y sus derechos de patente propiedad de Denso Wave no son ejercidos. En cuanto a Android se manifiesta bajo Licencia Android libre y Java bajo licencia GNU GPL / Java Community Process[7].

3.1 Análisis

Cabe aclarar que los diagramas de secuencia corresponden a la sección Diseño pero para una fácil lectura se han ubicado en la sección análisis junto a los casos de Uso.

3.1.1 Casos de Uso

3.1.1.1 Alta de producto

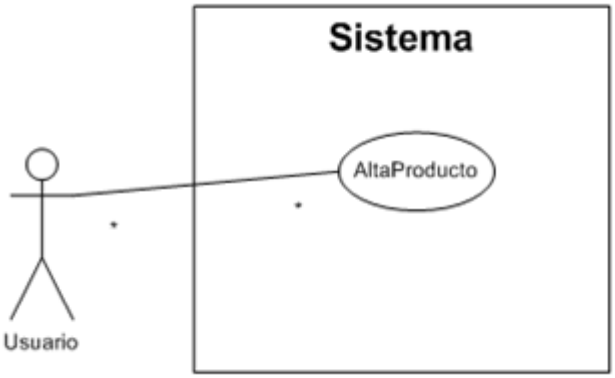
Actores: Usuario.

Descripción: El Usuario ingresa los datos correspondientes para un alta de producto, los cuales son el nombre de producto (String, obligatorio), imagen (Blob, opcional) y fecha de vencimiento (Date, obligatorio), la cual se ingresará mediante un Calendar.

No se permitirán el ingreso de datos vacíos ni solo numéricos al igual que fechas de caducidad menores a la actual, en tales casos el sistema anunciará mediante mensajes el incumplimiento de estas características. En caso de que el producto sea ingresado el sistema mostrara un mensaje informando que el producto se ha dado de alta correctamente.

Precondición: Ubicación en Pestaña de Alta de producto.
Postcondición: Ninguna.

[7] Licencia Pública General de GNU o más conocida por su nombre en inglés GNU General Public License



Flujo principal.

Paso 1 – Usuario: Ingresa, nombre de producto, imagen y fecha de caducidad y presiona el botón “Guardar”.

Paso 2 – Sistema: Inserta el producto en la tabla de Productos, limpia los campos y retorna mensaje “Se ha ingresado el producto [nombreProducto] correctamente”.

Flujo alternativo 1.

Nombre producto faltante.

Paso 1 – El sistema: Presenta mensaje “Deberá ingresar un producto”.

Flujo alternativo 2.

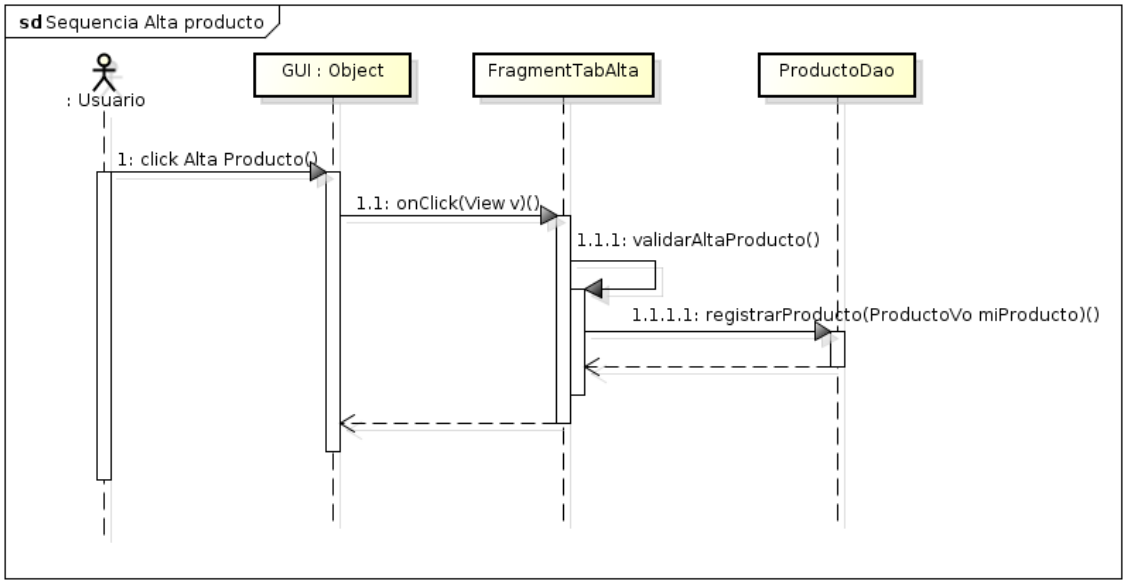
Nombre producto Incorrecto.

Paso 1 – El sistema: Presenta mensaje “No se aceptan producto íntegramente numéricos”.

Flujo alternativo 3.

Fecha Incorrecta.

Paso 1 – El sistema: Presenta mensaje “Deberá Seleccionar una fecha mayor a la actual”.



3.1.1.2 Borrador Pantalla alta

3.1.1.3 Lectura de código QR

Actores: Usuario.

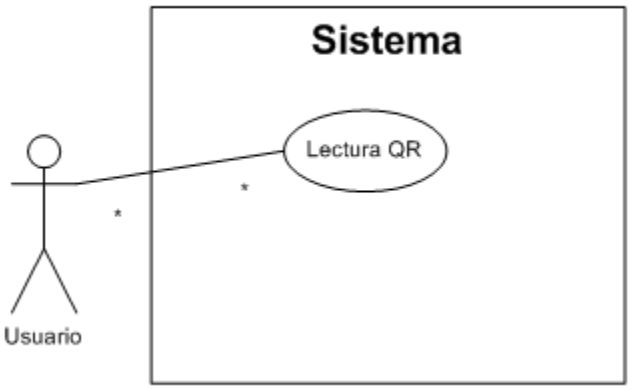
Descripción: El usuario hace click en botón “Lectura QR” con lo que el sistema enciende la cámara del dispositivo, ya sea notebook, PC, celular o tablet y se prepara para la lectura de códigos QR para cuando el usuario la enfoque en su campo visual. Una vez leído el código QR

el sistema automáticamente abrirá la aplicación y desplegara el listado de productos correspondiente a la heladera y usuario que correspondan al código QR.

Precondición:

- Código QR de más de 2cm cuadrados.
- Dispositivo con cámara.

Postcondición: Ninguna.



Flujo principal.

Paso 1 – Usuario: Presiona el botón “Lectura QR”.

Paso 2 – Sistema: Prende la cámara del dispositivo y se prepara para la lectura de códigos QR.

Paso 3 – Usuario: Presenta el código QR delante de la cámara del dispositivo.

Paso 4 – Sistema: Lee el código QR y automáticamente despliega el listado de productos que se corresponden para el usuario y heladera identificadas en el código QR.

Flujo alternativo 1.

Código QR no generado según las especificaciones de la aplicación.

Paso 1 – Usuario: Presiona el botón “Lectura QR”.

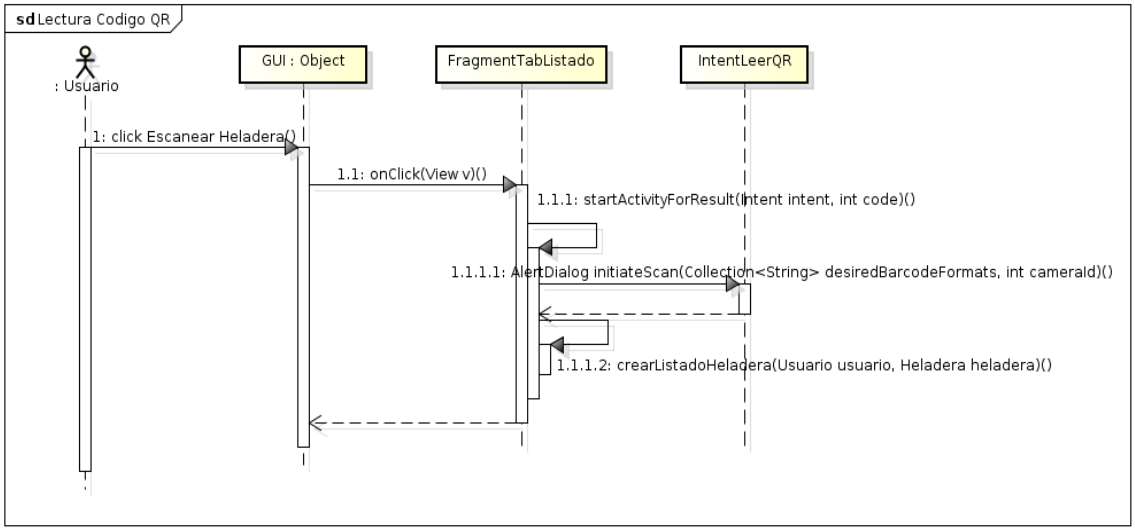
Paso 2 – Sistema: Se despliega mensaje indicando que los parámetros recibidos no son correctos.

Flujo alternativo 2.

Código QR no generado según las especificaciones de la aplicación.

Paso 1 – Usuario: Presiona el botón “Lectura QR”.

Paso 2 – Sistema: Se despliega mensaje indicando que los parámetros recibidos no son correctos.



3.1.1.4 Borrador Pantalla para lectura de QR

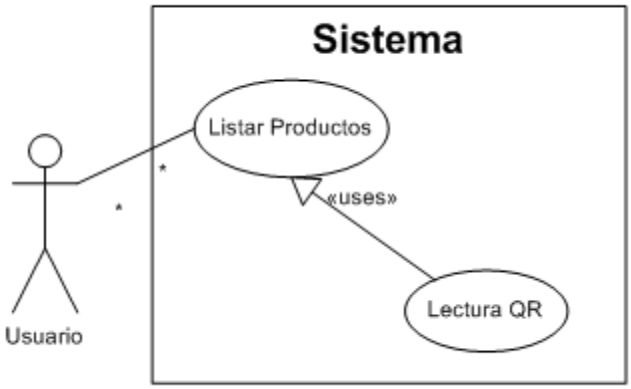
N/A Puede realizarse con cualquier lector de QR mediante la cámara del dispositivo, ya sea teléfono o tablet.

3.1.1.5 Listado de productos

Actores: Usuario.

Descripción: El usuario da lectura de un código QR de más de 2cm cuadrados o selecciona la sección “Listado”. Ante estos estímulos el Sistema deberá listar el total de productos correspondientes al id de la Heladera leída en el QR, los productos se ordenaran por fecha de vencimiento y por cada uno se indicara su nombre, estado y días para su vencimiento.

Precondición: Lectura QR o Selección de sección “Listado”.
Postcondición: Ninguna.



Flujo principal.

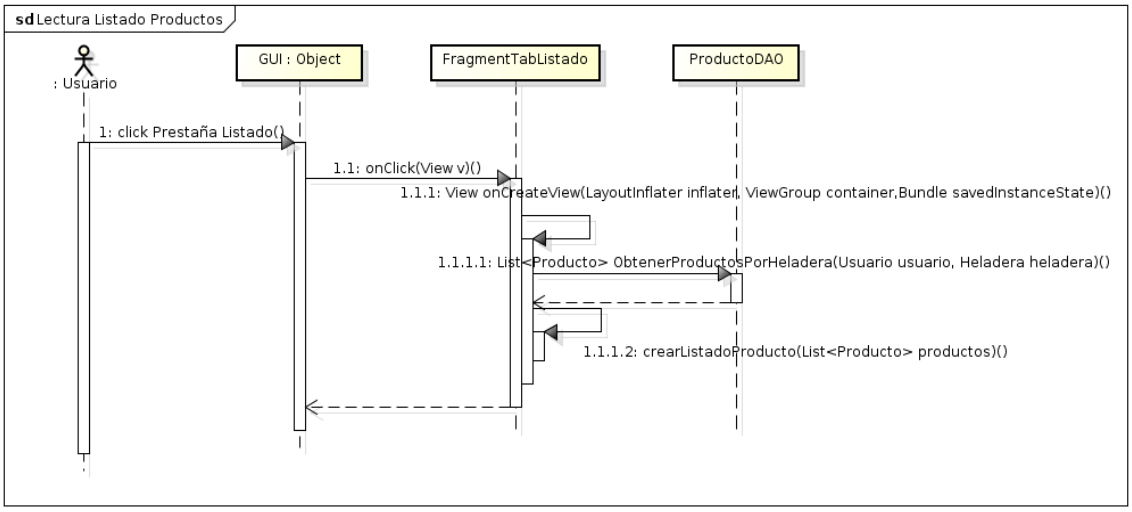
Paso 1 – Usuario: Lectura de QR o selección de sección “Listado”.

Paso 2 – Sistema: Lista todos los productos pertenecientes a la heladera indicada en el código QR. Los productos se ordenaran de forma descendente con el criterio de días por vencer de cada producto, si el producto se encuentra vencido, es decir con fecha de vencimiento, menos a la actual se deberá mostrar una leyenda indicando que el mismo se encuentra vencido. Para cada fila se despliega también un checkBox para poder seleccionar los productos y un botón eliminar debajo del listado el cual permitirá realizar eliminación de múltiples productos simultáneamente.

Flujo alternativo 1.

No existen productos.

Paso 1 – El sistema: Presenta mensaje “No hay productos cargados en esta heladera, podrá cargarlos desde la sección de Alta de productos”.



3.1.1.6 Borrador Pantalla Listado de productos

3.1.1.7 Modificación de producto

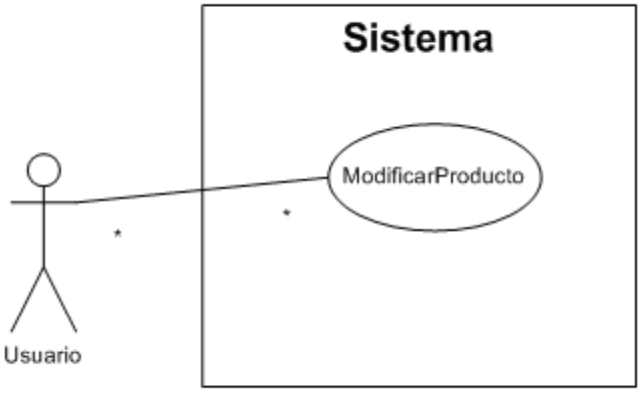
Actores: Usuario.

Descripción: En la sección de “Listado de productos” al realizar click sobre uno de los productos el sistema le deberá dar aparición a una pantalla con los detalles del mismo, nombre, fecha de caducidad y su imagen, cualquiera de estos 3 datos podrá ser modificado por el usuario.

Precondición:

- Ubicación en Pestaña de Listado de producto.
- Al menos un producto cargado en el sistema.

Postcondición: Ninguna.



Flujo principal.

Paso 1 – Usuario: Lectura de QR o selección de sección “Listado”.

Paso 2 – Sistema: Lista todos los productos pertenecientes a la heladera indicada en el código QR.

Paso 3 – Usuario: El usuario realiza un click sobre un producto listado, ante este estimulo se requiere que el sistema despliegue una nueva pantalla con el detalle de los datos de este producto como ser, nombre, fecha de vencimiento e imagen, el usuario puede modificar cualquiera de estos valores.

Paso 4 – Usuario: El usuario realiza un click fuera de la pantalla de modificaciones o cierra la misma.

Paso 5 – Sistema: El sistema almacena la nueva información registrada.

Flujo alternativo 1.

No existen productos.

Paso 1 – El sistema: Presenta mensaje “No hay productos almacenados en esta heladera, podrá almacenarlos desde la sección de |Alta de productos|”.

Flujo alternativo 2.

Nombre producto faltante.

Paso 1 – El sistema: Presenta mensaje “Deberá ingresar un producto”.

Flujo alternativo 3.

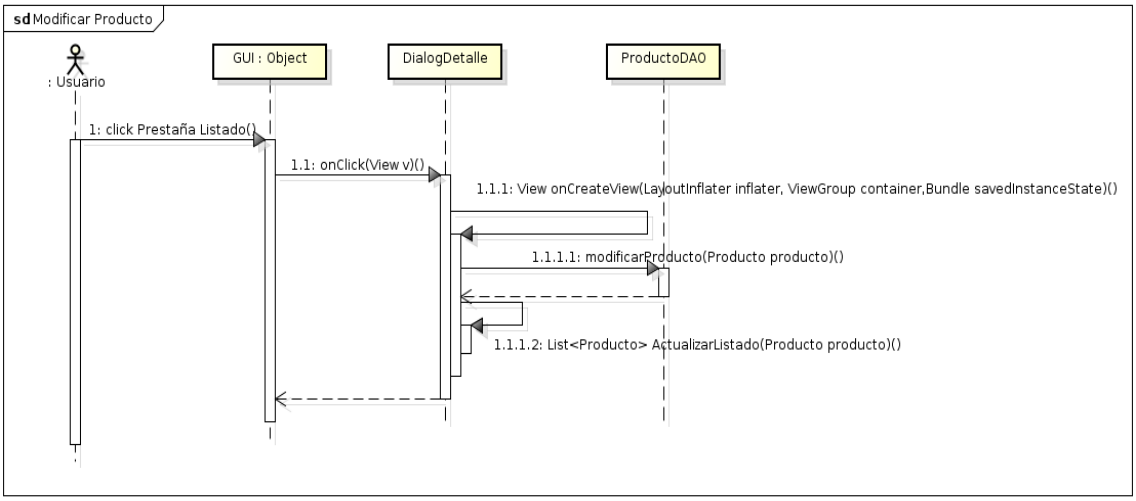
Nombre producto Incorrecto.

Paso 1 – El sistema: Presenta mensaje “No se aceptan producto íntegramente numéricos”.

Flujo alternativo 4.

Fecha Incorrecta.

Paso 1 – El sistema: Presenta mensaje “Deberá Seleccionar una fecha mayor a la actual”.



3.1.1.8 Borrador Pantalla modificación

3.1.1.9 Eliminación de productos

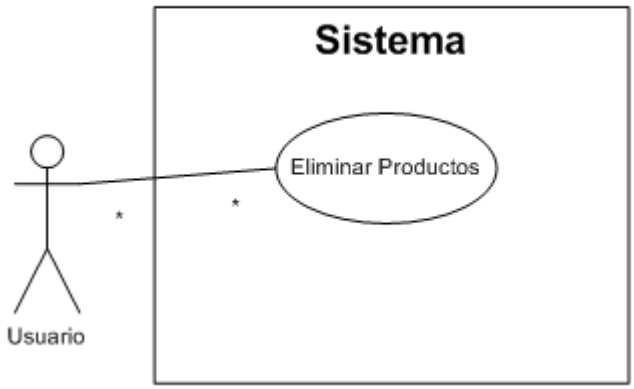
Actores: Usuario.

Descripción: El usuario da lectura de un código QR de más de 2cm cuadrados o selecciona la sección “Listado”. Ante estos estímulos el Sistema deberá listar el total de productos correspondientes al id de la Heladera leída en el QR, los productos se ordenaran por fecha de vencimiento y por cada uno se indicara su nombre, estado y días para su vencimiento. Junto a estos datos se añadirá un checkbox para cada producto listado el cual puede ser tildado por el Usuario y un botón eliminar debajo del listado total de productos para de esta manera dar opción de la eliminación múltiple de productos en el sistema. Al seleccionar uno o varios productos y presionar el botón eliminar el sistema deberá inmediatamente borrarlos de la lista de productos.

Precondición:

- Ubicación en Pestaña de Listado de producto.
- Al menos un producto cargado en el sistema.

Postcondición: Ninguna.



Flujo principal.

Paso 1 – Usuario: Lectura de QR o selección de sección “Listado”.

Paso 2 – Sistema: Lista todos los productos pertenecientes a la heladera indicada en el código QR.

Paso 3 – Usuario: Selecciona mediante los checkbox los productos a eliminar y presiona el botón “Borrar”.

Paso 4 – Sistema: Elimina los productos seleccionados por el usuario.

Flujo alternativo 1.

No existen productos.

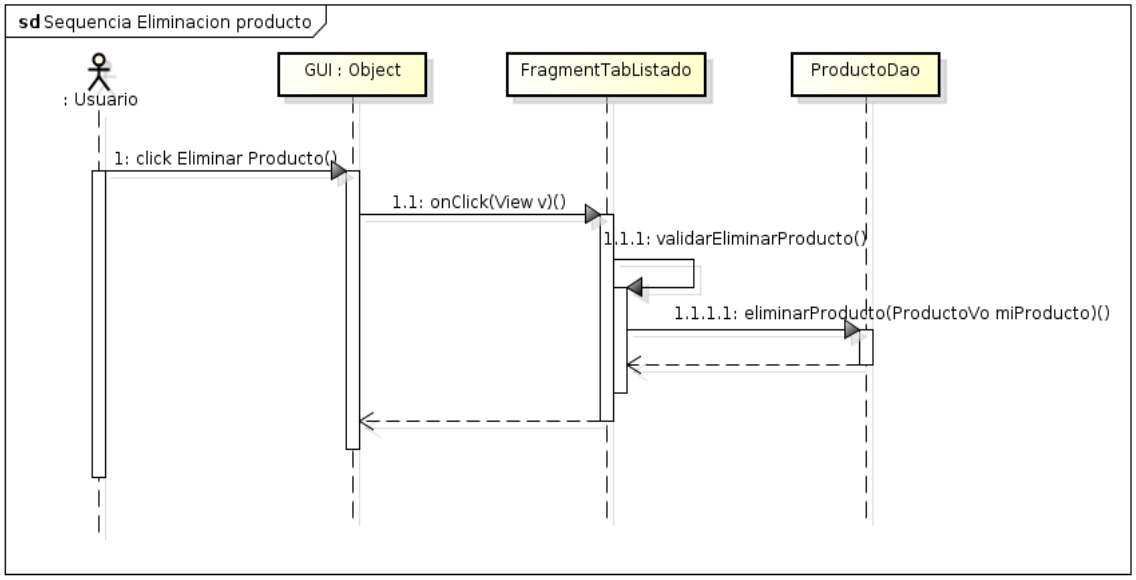
Paso 1 – El sistema: Presenta mensaje “No hay productos almacenados en esta heladera, podrá almacenarlos desde la sección de |Alta de productos|”.

Flujo alternativo 2.

No se han seleccionado productos.

Paso 1 – El Usuario: Presiona el botón eliminar sin haber seleccionado productos.

Pasó 2 – El sistema: Presenta mensaje “Deberá seleccionar al menos un producto a eliminar”.



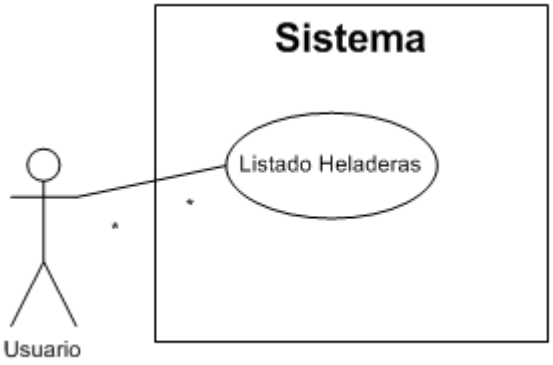
3.1.1.10 Borrador pantalla eliminar producto

3.1.1.11 Listado de heladeras

Actores: Usuario.

Descripción: El usuario selecciona la sección “Listado”. Ante esta acción el Sistema deberá listar el total de heladeras correspondientes al id del usuario, las heladeras se ordenaran por orden alfabético y por cada una se indicara la cantidad de productos vencidos por heladera.

Precondición: Al menos una heladera dada de alta.
Postcondición: Ninguna.



Flujo principal.

Paso 1 – Usuario: Selección de sección “Listado”.

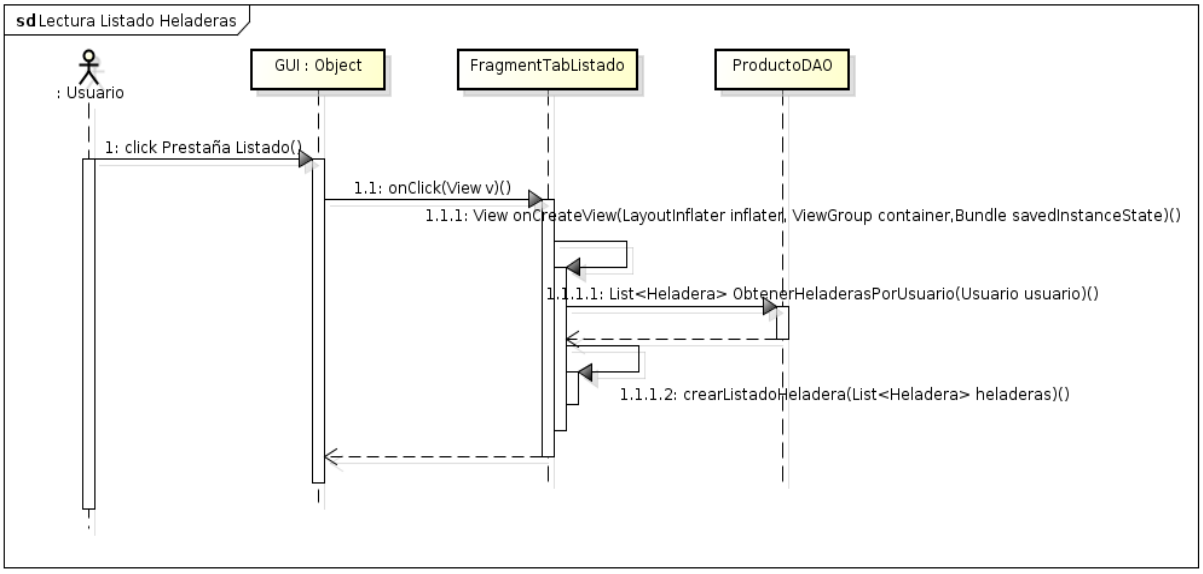
Paso 2 – Sistema: Lista todas las heladeras pertenecientes al usuario utilizado. Las heladeras se ordenaran alfabéticamente, el sistema también indicara la cantidad de productos vencidos por heladera.

Flujo alternativo 1.

No existen heladeras.

Paso 1 – El sistema: Presenta mensaje “No hay heladeras registradas para el usuario seleccionado”.

Diagrama de secuencia.



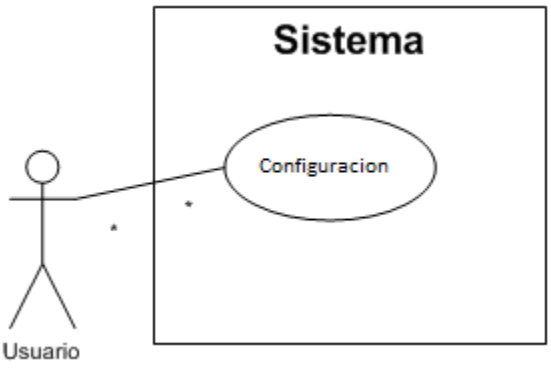
3.1.1.12 Borrador pantalla listado de heladeras

3.1.1.13 Configuración

Actores: Usuario.

Descripción: El usuario selecciona la sección “Configuración”. Ante esta acción el Sistema deberá desplegar los componentes visuales que den opción al usuario de modificar el idioma de la aplicación, a ser estos Español, Inglés y Portugués, asimismo también se desplegaran los componentes visuales para que el cliente pueda optar por el nivel de alertas que desea obtener del sistema, pudiendo fijar la cantidad de días para considerar un producto en estado de precaución y la cantidad de días para considerar un producto en estado de alerta.

Precondición: Ninguna.
Postcondición: Ninguna.



Flujo principal.

Paso 1 – Usuario: Selección de sección “Configuración”.

Paso 2 – Sistema: Despliega componentes visuales (ver figura correspondiente de borrador de la pantalla) que permitirán seleccionar el idioma de la aplicación y el nivel de alertas sobre los productos almacenados. Los idiomas desplegados serán Español, Inglés y Portugués, por defecto la aplicación tomara el idioma seleccionado en el sistema operativo del dispositivo.

Paso 3 – Usuario: Selecciona un idioma distinto al actual.

Paso 4 – Sistema: Modifica el idioma de todos los literales de la aplicación.

Paso 5 – Usuario: Fija los días de notificación de alertas de productos almacenados.

Paso 6 – Sistema: Fija la notificación de alertas de productos almacenados en el numero seleccionado por el usuario.

Paso 7 – Usuario: Fija los días para notificación de precaución de productos almacenados.

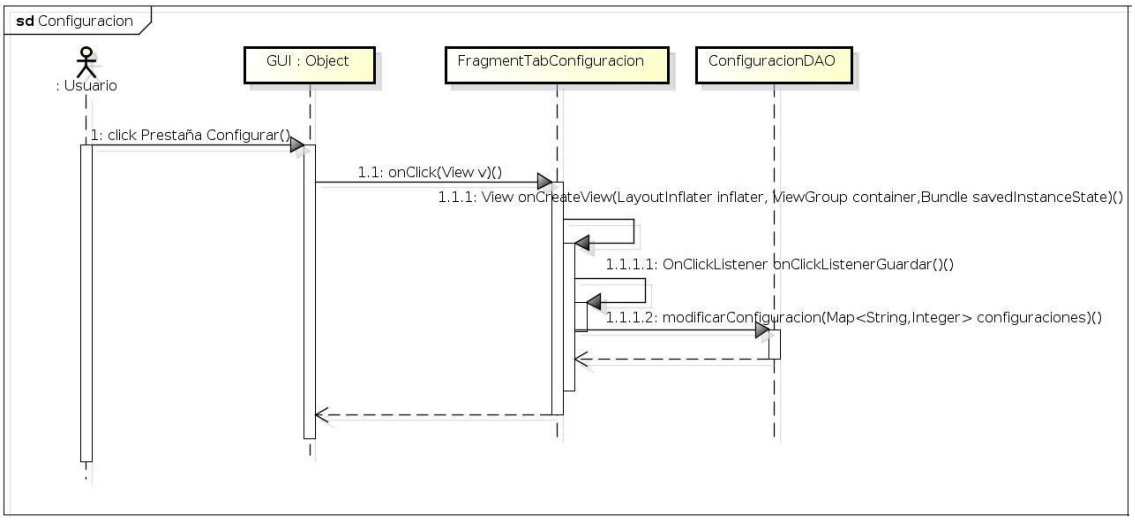
Paso 8 – Sistema: Fija la notificación de precaución de productos en el numero seleccionado por el usuario.

Flujo alternativo 1.

El número de días fijado para notificación de precaución es menor que el de notificación de alerta.

Paso 1 – El sistema: No fija los días seleccionados por el usuario y presenta mensaje “El número de días para notificaciones de precaución debe ser mayor que el número de días para alerta”.

Diagrama de secuencia.

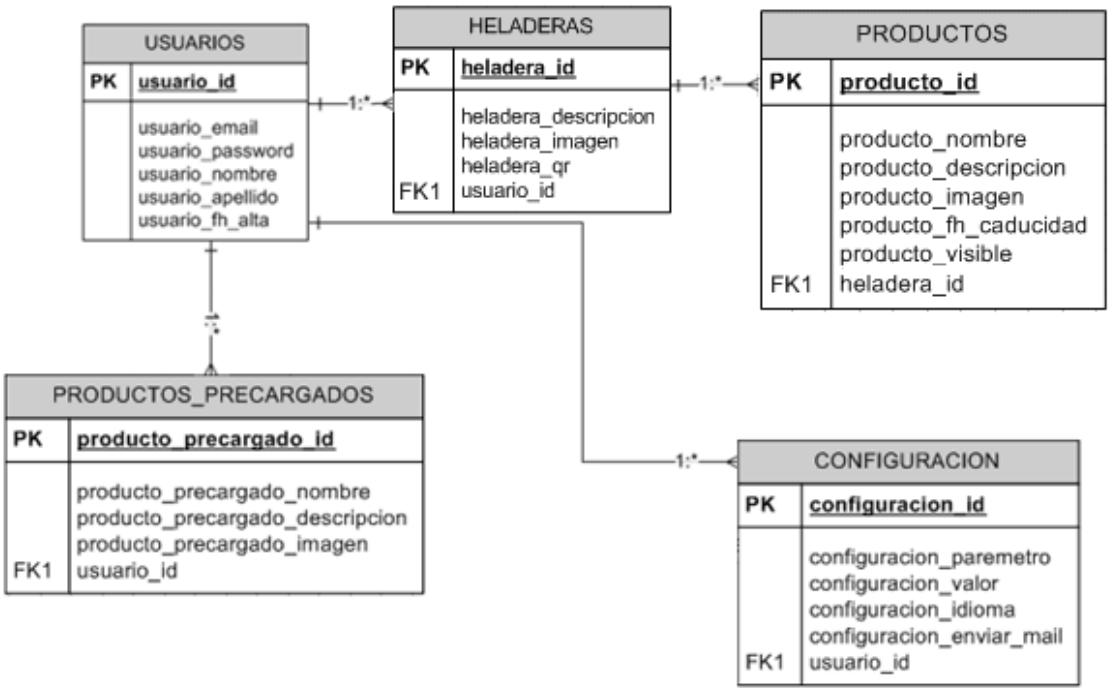


3.1.1.14 Borrador Pantalla configurar

3.2 Diseño

Los diagramas de secuencia presentados en la sección Análisis corresponden al diseño pero se han presentado en la anterior sección para dar una lectura cómoda de estos diagramas junto a los casos de uso.

3.2.1 Diagrama de Entidad Relación Base de Datos

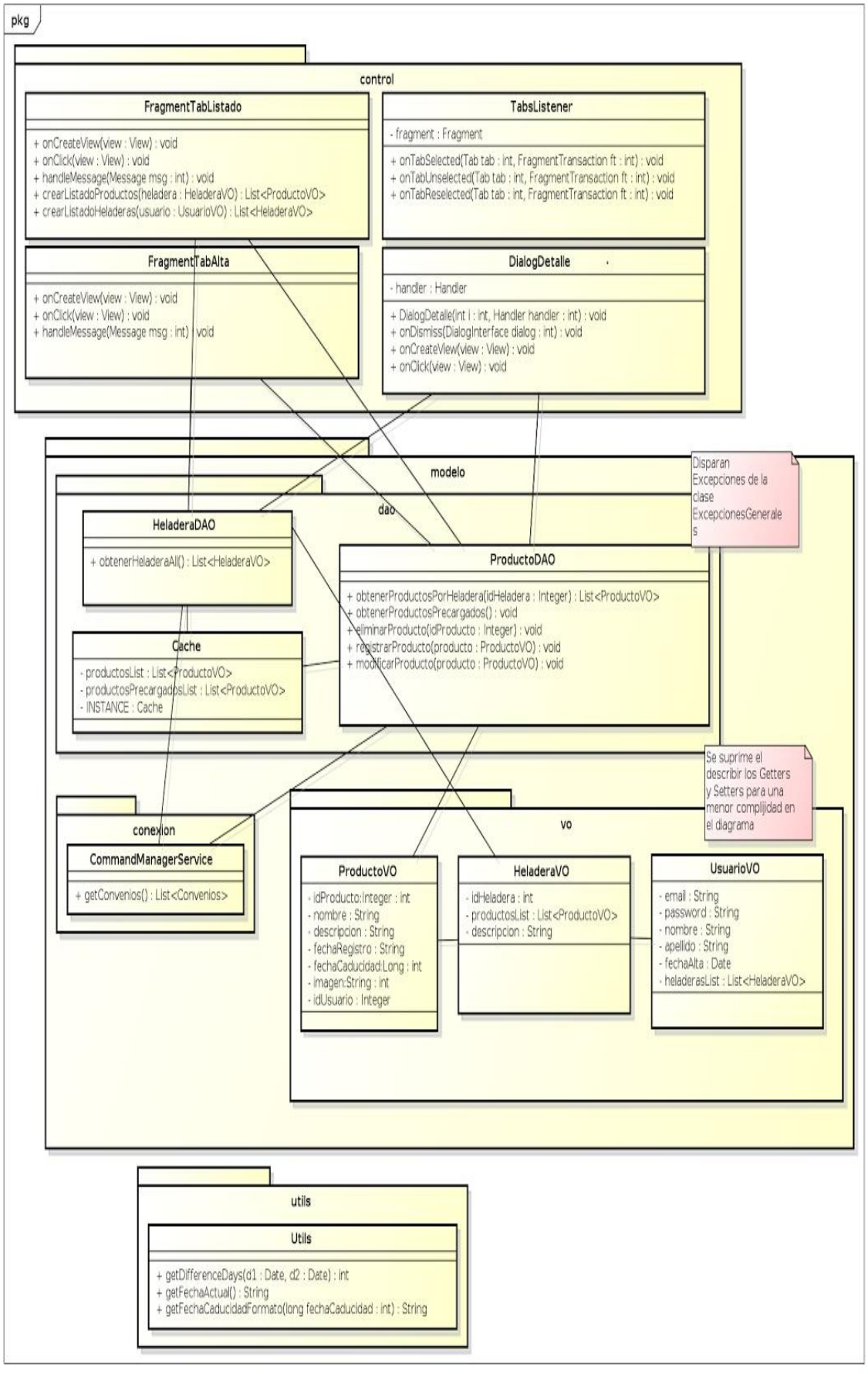


3.2.2 Diccionario de datos

ENTIDAD	ATRIBUTOS	TIPO	LARGO	RESTRICCIONES	DESCRIPCION
USUARIOS	usuario_id	int	16	clave primaria, not null	identificador de la tabla usuarios
	usuario_email	varchar	128	not null	correo electrónico del usuario
	usuario_password	varchar	128	not null	password del usuario
	usuario_nombre	varchar	128	not null	nombre del usuario
	usuario_apellido	varchar	128	not null	apellido del usuario
	usuario_fh_alta	date		not null	fecha de alta del usuario en la aplicación
PRODUCTOS	producto_id	date		clave primaria, not null	identificador de la tabla productos
	producto_nombre	date		not null	nombre del producto
	producto_descripcion	varchar	600		descripción del producto

	producto_imagen	blob			imagen del producto
	producto_fh_caducidad	datetime		not null	fecha de caducidad seleccionada para el producto
	producto_visible	tinyint	1	not null	indica si el producto se mostrara en la aplicación, 1 el producto se visualizara 0 no se visualizara
	heladera_id	int	16	clave foránea, not null	identificador foráneo de la tabla heladeras
PRODUCTOS_PRECARGADOS	producto_precargado_id	int	6	clave primaria, not null	identificador de la tabla de productos precargados
	producto_precargado_nombre	varchar	512	not null	nombre del producto precargado
	producto_precargado_descripcion	varchar	1024		descripción del producto precargado
	producto_precargado_imagen	longblob			imagen del producto precargado
	usuario_id	int		clave foránea, not null	identificador foráneo de la tabla usuarios
HELADERAS	heladera_id	int	16	clave primaria, not null	identificador de la tabla heladera
	heladera_descripcion	varchar	256	not null	descripción de la heladera
	heladera_imagen	blob			imagen de la heladera
	heladera_qr	blob			qr correspondiente a la heladera
	usuario_id	int	16	clave foránea, not null	identificador foráneo de la tabla de usuarios
CONFIGURACION	configuracion_id	int	16	clave primaria, not null	identificador de la tabla configuración
	configuracion_parametro	varchar	64	not null	parámetro perteneciente a la configuración
	configuracion_valor	int	3	not null	valor dado al parámetro perteneciente a la configuración
	configuracion_idioma	varchar	24	not null	valor que contiene el idioma seleccionado ya sea es_es, es_ar, en_en, en_us, pt_pt, pt_br.
	configuracion_enviar_mail	tinyint	1	not null	valor que fija si se desea o no recibir mails de la aplicación, 1 si, 0 no.
	usuario_id	int	16	clave foránea, not null	identificador foráneo de la tabla de usuarios

3.2.3 Diagrama de clases

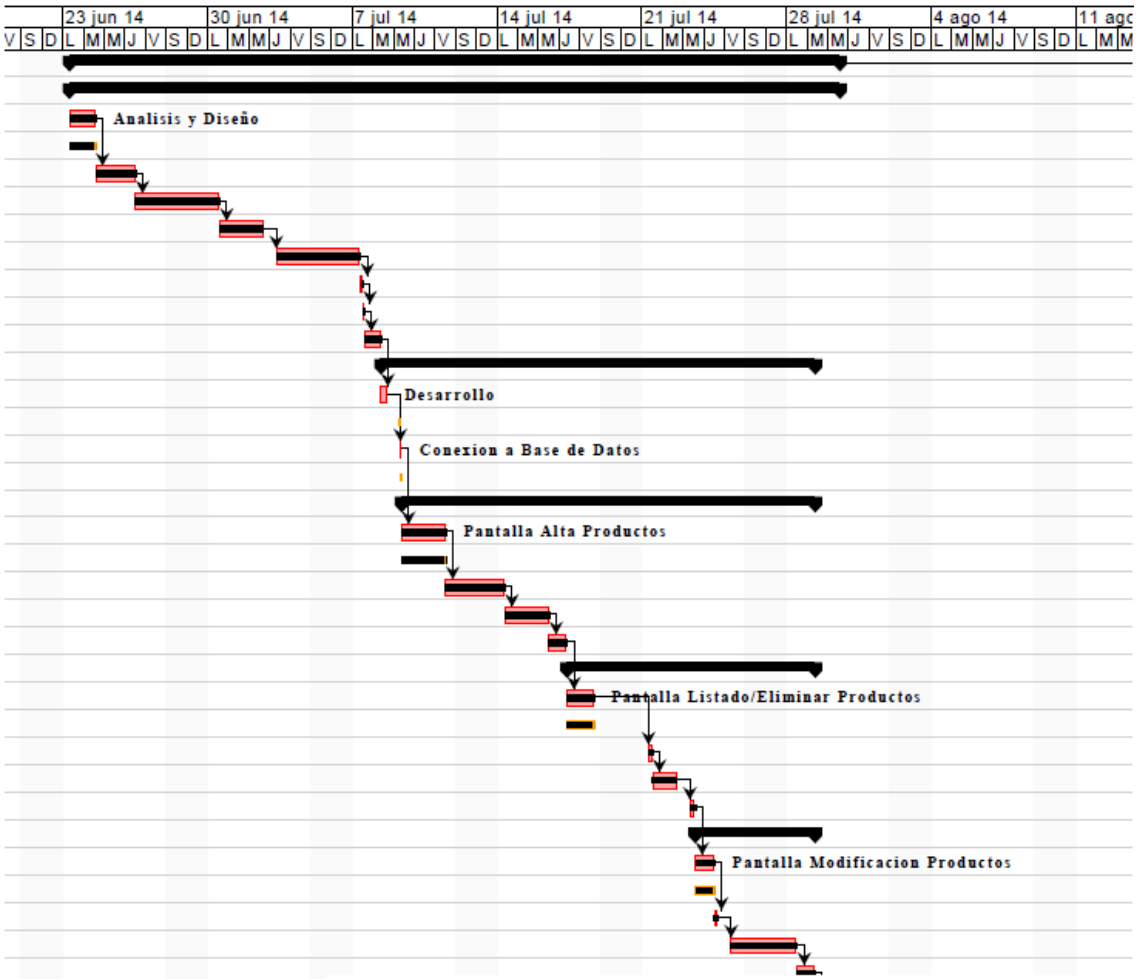


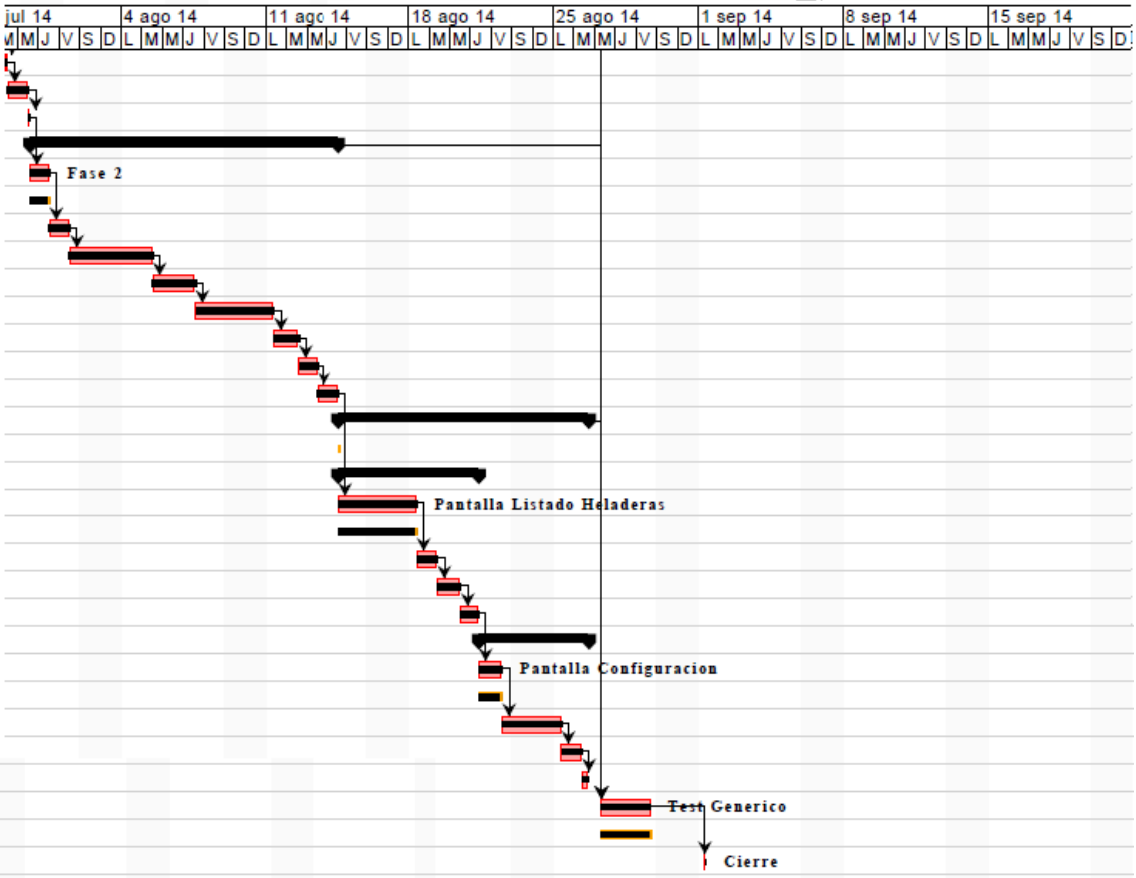
3.3 Desarrollo de la solucion

3.3.1 Gantt

		Nombre	Duracion	Inicio	Terminado	Predecesores
1	✓	**Fase 1**	**111 days**	**23/06/14 08:00**	**30/07/14 15:00**	
2	✓	**Analisis y Diseño**	**111 days**	**23/06/14 08:00**	**30/07/14 15:00**	
3	✓	Analisis	7 days	23/06/14 08:00	24/06/14 15:00	
4	✓	Especificaciones del producto	7 days	24/06/14 15:00	26/06/14 13:00	3
5	✓	Casos de uso	9 days	26/06/14 12:00	30/06/14 15:00	4
6	✓	Diagrama de clases	9 days	30/06/14 15:00	02/07/14 17:00	5
7	✓	Diagrama de secuencia	9 days	03/07/14 08:00	07/07/14 10:00	6
8	✓	Diagrama Entidad Relacion	1 day	07/07/14 10:00	07/07/14 13:00	7
9	✓	Estimacion/Gantt	1 day	07/07/14 12:00	07/07/14 15:00	8
10	✓	Requerimientos minimos del producto	2 days	07/07/14 15:00	08/07/14 10:00	9
11	✓	**Desarrollo**	**60 days**	**08/07/14 10:00**	**29/07/14 10:00**	
12	✓	Conexion Base de datos/Uso de Threads	3 days	08/07/14 10:00	08/07/14 17:00	10
13	✓	Creacion de Values Objects	1 day	09/07/14 08:00	09/07/14 10:00	12
14	✓	**Pantalla Alta producto**	**56 days**	**09/07/14 10:00**	**29/07/14 10:00**	
15	✓	Vista Pantalla Alta producto	9 days	09/07/14 10:00	11/07/14 13:00	13
16	✓	Creacion DAO de Alta	3 days	11/07/14 12:00	14/07/14 10:00	15
17	✓	Transaccionalidad y validaciones Pantala Alta	9 days	14/07/14 10:00	16/07/14 13:00	16
18	✓	Test unitario	3 days	16/07/14 12:00	17/07/14 10:00	17
19	✓	**Pantalla Listado Productos**	**32 days**	**17/07/14 10:00**	**29/07/14 10:00**	
20	✓	Vista pantalla Listado	7 days	17/07/14 10:00	18/07/14 17:00	18
21	✓	Creacion Dao de Listado	2 days	21/07/14 08:00	21/07/14 13:00	20
22	✓	Transaccionalidad y validaciones Pantalla Listado	6 days	21/07/14 12:00	22/07/14 17:00	21
23	✓	Test unitario	3 days	23/07/14 08:00	23/07/14 15:00	22
24	✓	**Pantalla Detalle/Modificar**	**14 days**	**23/07/14 15:00**	**29/07/14 10:00**	
25	✓	Vista pantalla Detalle/Modificar	3 days	23/07/14 15:00	24/07/14 13:00	23
26	✓	Creacion DAO de Modificar	2 days	24/07/14 12:00	24/07/14 17:00	25
27	✓	Transaccionalidad y validaciones Pantalla Detalle/Modifi...	6 days	25/07/14 08:00	28/07/14 13:00	26
28	✓	Test Unitario	3 days	28/07/14 12:00	29/07/14 10:00	27
29	✓	Vista Prestañas	1 day	29/07/14 10:00	29/07/14 13:00	28
30	✓	Transaccionalidad y validaciones Lectura QR	4 days	29/07/14 12:00	30/07/14 13:00	29
31	✓	Alta listado de productos precargados en Base de Datos	1 day	30/07/14 12:00	30/07/14 15:00	30
32	✓	**Fase 2**	**44 days**	**30/07/14 15:00**	**14/08/14 15:00**	
33	✓	Correccion bug camara pick image	3 days	30/07/14 15:00	31/07/14 13:00	31
34	✓	Correccion bug eliminar productos	4 days	31/07/14 12:00	01/08/14 13:00	33
35	✓	Listado de Heladeras/Secciones	8 days	01/08/14 12:00	05/08/14 13:00	34
36	✓	Captura QR desde el producto de Software	9 days	05/08/14 12:00	07/08/14 15:00	35
37	✓	Alta de imagenes de productos precargados	6 days	07/08/14 15:00	11/08/14 10:00	36
38	✓	Transaccionalidad obtener imagen	6 days	11/08/14 10:00	12/08/14 15:00	37
39	✓	Modificacion de queries SQL para multiples heladeras	3 days	12/08/14 15:00	13/08/14 13:00	38
40	✓	Test Integracion	5 days	13/08/14 12:00	14/08/14 15:00	39
41	✓	**Fase 3**	**33 days**	**14/08/14 15:00**	**26/08/14 17:00**	
42	✓	**Pantalla Listado Heladeras**	**18 days**	**14/08/14 15:00**	**21/08/14 10:00**	
43	✓	Vista pantalla Listado Heladeras	6 days	14/08/14 15:00	18/08/14 10:00	40
44	✓	Creacion DAO de Heladeras	4 days	18/08/14 10:00	19/08/14 10:00	43
45	✓	Transaccionalidad y validaciones Pantalla Listado Heladeras	5 days	19/08/14 10:00	20/08/14 13:00	44
46	✓	Test Unitario	3 days	20/08/14 12:00	21/08/14 10:00	45
47	✓	**Pantalla Configuracion**	**15 days**	**21/08/14 10:00**	**26/08/14 17:00**	
48	✓	Vista pantalla Configuracion	5 days	21/08/14 10:00	22/08/14 13:00	46
49	✓	Creacion DAO de Configuracion	3 days	22/08/14 12:00	25/08/14 10:00	48
50	✓	Transaccionalidad y validaciones Pantalla Configuracion	4 days	25/08/14 10:00	26/08/14 10:00	49
51	✓	Test Unitario	3 days	26/08/14 10:00	26/08/14 17:00	50
52	✓	Test General	12 days	27/08/14 08:00	29/08/14 17:00	41;32;1
53	✓	Cierre	1 day	01/09/14 08:00	01/09/14 10:00	52

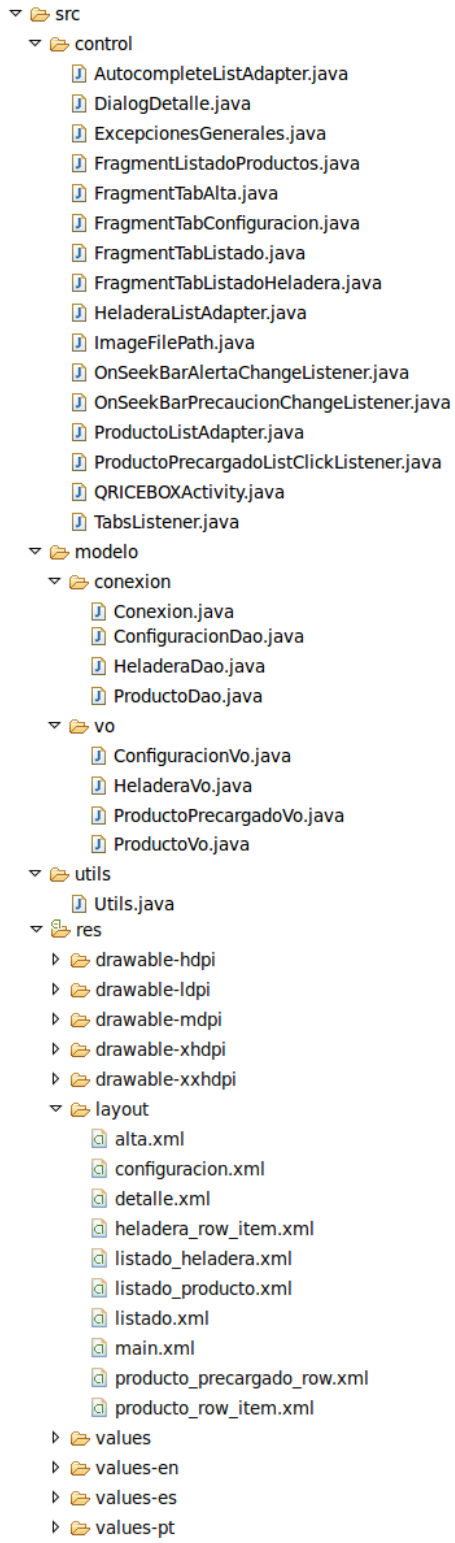
Figura 10: Se ha considerado para la planificación 3 hs diarias de trabajo.


23 jun 14
30 jun 14
7 jul 14
14 jul 14
21 jul 14
28 jul 14
4 ago 14
11 ago
Analisis y Diseño
Desarrollo
Conexion a Base de Datos
Pantalla Alta Productos
Pantalla Listado/Eliminar Productos
Pantalla Modificacion Productos


jul 14
4 ago 14
11 ago 14
18 ago 14
25 ago 14
1 sep 14
8 sep 14
15 sep 14
Fase 2
Pantalla Listado Heladeras
Pantalla Configuracion
Test Generico
Cierre

3.3.2 Código Fuente

Se detalla a continuación el código fuente correspondiente a la solución, para facilitar su lectura se han de presentar en orden: paquete Utils, paquete Modelo (contiene a su vez paquete Conexión y Value objects), paquete Control, paquete Vista Aplicando el patrón de arquitectura de software Modelo-Vista-Control explicado en la sección marco teorico de este trabajo.



Otra convención adoptada para facilitar la lectura del código es una diferenciación por colores donde:

- **Verde**: comentarios sobre el código.
- **Violeta**: palabras reservadas propias del lenguaje.
- **Azul**: miembros heredados y palabras reservadas.
- **Gris**: cadenas de caracteres.
- **Naranja**: numéricos.

3.2.2.1 Paquete Utils

3.2.2.1.1 Clase Utils

```
package utils;
import java.util.Date;

/**
 * Clase responsable de las utilidades utilizadas en toda la
aplicacion
 * @author jgomezva
 *
 */
public class Utils {
	Date fechaHoy = new Date();

	// El constructor debe ser Protected para evitar su acceso
desde fuera.
	protected Utils() {
	}

	private static class UtilsHolder {
	  private final static Utils INSTANCE = new Utils();
	}

	// Método para obtener la instancia de la clase
	public static Utils getInstance() {
	  return UtilsHolder.INSTANCE;
	}

	public String getFechaCaducidadFormato(long fechaCaducidad){
		java.text.SimpleDateFormat sdf = new
java.text.SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
		String fechacaducidadResultado =
sdf.format(fechaCaducidad);
		return fechacaducidadResultado;
	    }

	public String getFechaActual(){
		java.util.Date dt = new java.util.Date();
		java.text.SimpleDateFormat sdf = new
java.text.SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
		String currentTime = sdf.format(dt);
		return currentTime;
```

```
		}

	public int getDifferenceDays(Date d1, Date d2){
		int daysdiff = 0;
		long diff = d2.getTime() - d1.getTime();
		long diffDays = diff / (24 * 60 * 60 * 1000)+1;
		daysdiff = (int) diffDays;
		return daysdiff;
	}

	public Date getFechaHoy() {
		return fechaHoy;
	}

	public void setFechaHoy(Date fechaHoy) {
		this.fechaHoy = fechaHoy;
	}
}
```

3.2.2.2 Paquete model.vo

3.2.2.2.1 clase ProductoVo

```
package modelo.vo;
/**
 * Clase representativa del value object de Productos
 * @author jgomezva
 *
 */
public class ProductoVo {

	private Integer idProducto;
	private String nombre;
	private String descripcion;
	private String fechaRegistro;
	private long fechaCaducidad;
	private byte[] imagen;
	private Integer idUsuario;

	public Integer getIdProducto() {
		return idProducto;
	}
	public void setIdProducto(Integer idProducto) {
		this.idProducto = idProducto;
	}
	public String getNombre() {
		return nombre;
	}
	public void setNombre(String nombre) {
		this.nombre = nombre;
	}
	public String getDescripcion() {
		return descripcion;
	}
	public void setDescripcion(String descripcion) {
		this.descripcion = descripcion;
	}
	public String getFechaRegistro() {
```

```
		return fechaRegistro;
	}
	public void setFechaRegistro(String fechaRegistro) {
		this.fechaRegistro = fechaRegistro;
	}
	public long getFechaCaducidad() {
		return fechaCaducidad;
	}
	public void setFechaCaducidad(long fechaCaducidad) {
		this.fechaCaducidad = fechaCaducidad;
	}
	public Integer getIdUsuario() {
		return idUsuario;
	}
	public void setIdUsuario(Integer idUsuario) {
		this.idUsuario = idUsuario;
	}
	public byte[] getImagen() {
		return imagen;
	}
	public void setImagen(byte[] imagen) {
		this.imagen = imagen;
	}
}
```

3.2.2.2.2 clase ProductoPrecargadoVo

```
package modelo.vo;

import java.sql.Blob;
/**
 * Clase representativa del value object de Productos precargados
 * @author jgomezva
 *
 */
public class ProductoPrecargadoVo {
	private String nombre;
	private Blob imagen;
	public String getNombre() {
		return nombre;
	}
	public void setNombre(String nombre) {
		this.nombre = nombre;
	}
	public Blob getImagen() {
		return imagen;
	}
	public void setImagen(Blob imagen) {
		this.imagen = imagen;
	}
}
```

3.2.2.2.3 clase HeladeraVo

```
package modelo.vo;

import java.sql.Blob;
```

```
/**
 * Clase representativa del value object de Heladeras
 * @author jgomezva
 *
 */
public class HeladeraVo {
	private Integer idHeladera;
	private String descripcion;
	private Blob imagen;
	private Integer cantidadProductos;
	private Integer cantidaVencidos;
	private Integer cantidadEnRiesgo;
	private Integer cantidadConPrecaucion;

	public Integer getIdHeladera() {
		return idHeladera;
	}
	public void setIdHeladera(Integer idHeladera) {
		this.idHeladera = idHeladera;
	}
	public String getDescripcion() {
		return descripcion;
	}
	public void setDescripcion(String descripcion) {
		this.descripcion = descripcion;
	}
	public Blob getImagen() {
		return imagen;
	}
	public void setImagen(Blob imagen) {
		this.imagen = imagen;
	}
	public Integer getCantidadProductos() {
		return cantidadProductos;
	}
	public void setCantidadProductos(Integer cantidadProductos) {
		this.cantidadProductos = cantidadProductos;
	}
	public Integer getCantidaVencidos() {
		return cantidaVencidos;
	}
	public void setCantidaVencidos(Integer cantidaVencidos) {
		this.cantidaVencidos = cantidaVencidos;
	}
	public Integer getCantidadEnRiesgo() {
		return cantidadEnRiesgo;
	}
	public void setCantidadEnRiesgo(Integer cantidadEnRiesgo) {
		this.cantidadEnRiesgo = cantidadEnRiesgo;
	}
	public Integer getCantidadConPrecaucion() {
		return cantidadConPrecaucion;
	}
	public void setCantidadConPrecaucion(Integer
cantidadConPrecaucion) {
		this.cantidadConPrecaucion = cantidadConPrecaucion;
	}
}
```

3.2.2.2.4 clase ConfiguracionVo

```
package modelo.vo;
/**
 * Clase representativa del value object de Configuracion
 * @author jgomezva
 *
 */
public class ConfiguracionVo {
	private String parametro;
	private Integer valor;
	public String getParametro() {
		return parametro;
	}
	public void setParametro(String parametro) {
		this.parametro = parametro;
	}
	public Integer getValor() {
		return valor;
	}
	public void setValor(Integer valor) {
		this.valor = valor;
	}
}
```

3.2.2.3 Paquete model.conexion

3.2.2.3.1 Clase Conexión

```
package modelo.conexion;

/**
 * Clase que permite conectar con la base de datos
 * @author jgomezva
 *
 */
public class Conexion {
   private static final String bd = "QRIceBox";
   private static final String login = "root";
   private static final String password = "**********";
   private static final String url = "jdbc:mysql://10.0.2.2/"+bd;

   public static String getBd() {
      return bd;
   }

   public static String getLogin() {
      return login;
   }

   public static String getPassword() {
      return password;
   }

   public static String getUrl() {
      return url;
   }
}
```

3.2.2.3 Paquete model.dao

3.2.2.3.1 Clase ProductoDao

```
package modelo.dao;

import java.sql.Blob;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import modelo.conexion.Conexion;
import modelo.vo.ProductoPrecargadoVo;
import modelo.vo.ProductoVo;
import utils.Utils;
import control.ExcepcionesGenerales;
/**
 * Clase que oficia de Data Access Object para entidades PRODUCTOS
 * @author jgomezva
 *
 */
public class ProductoDao extends Thread
{
	private ProductoVo producto;
	private List<ProductoVo> listaProducto;
	private List<ProductoPrecargadoVo> listaProductosPrecargados;
	private static String operacion;
	private List<Integer> identificadores;

	public void registrarProducto(ProductoVo miProducto) throws
ExcepcionesGenerales{
		operacion = "alta";
		if(miProducto == null){
			throw new ExcepcionesGenerales("Se deben ingresar
datos");
		}
		Date fechaVencimiento = new
Date(miProducto.getFechaCaducidad());
		int dias = Utils.getInstance().getDifferenceDays(new
Date(),fechaVencimiento);
		if(miProducto.getNombre().length()<1 ||
miProducto.getNombre()==null){
			throw new ExcepcionesGenerales("No se ha ingresado
nombre");
		}else if(dias < 1){
			throw new ExcepcionesGenerales("Fecha vencimiento
menor a la actual");
		}
		this.setProducto(miProducto);
		this.start();
    }
```

```
	public void modificarProducto(ProductoVo miProducto) throws
ExcepcionesGenerales{
		operacion = "modificacion";
		if(miProducto == null){
			throw new ExcepcionesGenerales("Se deben ingresar
datos");
		}
		Date fechaVencimiento = new
Date(miProducto.getFechaCaducidad());
		int dias = Utils.getInstance().getDifferenceDays(new
Date(),fechaVencimiento);
		if(miProducto.getNombre().length()<1 ||
miProducto.getNombre()==null){
			throw new ExcepcionesGenerales("No se ha ingresado
nombre");
		}else if(dias < 1){
			throw new ExcepcionesGenerales("Fecha vencimiento
menor a la actual");
		}
		this.setProducto(miProducto);
		this.start();
	}

	public List<ProductoVo> obtenerProductosAll(){
		operacion = "selectAll";
		this.start();
		return listaProducto;
	}

	public List<ProductoPrecargadoVo> obtenerProductosPrecargados(){
		operacion = "selectProductosPrecargados";
		this.start();
		return listaProductosPrecargados;
	}

	public void eliminarProducto(List<Integer> idEliminarList){
		operacion = "eliminar";
		this.identificadores = idEliminarList;
		this.start();
	}

	public ProductoVo getProducto() {
		return producto;
	}

	public void setProducto(ProductoVo producto) {
		this.producto = producto;
	}

	public String getFechaCaducidadFormato(long fechaCaducidad){
		java.text.SimpleDateFormat sdf = new
java.text.SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
		String fechacaducidadResultado =
sdf.format(fechaCaducidad);
		return fechacaducidadResultado;
	}

	public String getFechaActual(){
		java.util.Date dt = new java.util.Date();
		java.text.SimpleDateFormat sdf = new
java.text.SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
```

```
            String currentTime = sdf.format(dt);
            return currentTime;
      }

      public void resultSetToArrayList(ResultSet rs) throws
SQLException{
            listaProducto = new ArrayList<ProductoVo>();
            while (rs.next()) {
                  ProductoVo producto = new ProductoVo();
                  producto.setIdProducto(rs.getInt("idProducto"));
                producto.setNombre(rs.getString("nombre"));
                Blob b = rs.getBlob("imagen");
                if(b!=null){
                  producto.setImagen(b.getBytes(1, (int) b.length()));
                }

producto.setFechaCaducidad(rs.getDate("fechaCaducidad").getTime());

                listaProducto.add(producto);
        }
      }

      public void resultSetToArrayPrecargadosList(ResultSet rs) throws
SQLException{
              listaProductosPrecargados = new
ArrayList<ProductoPrecargadoVo>();
              while (rs.next()) {
                    ProductoPrecargadoVo productoPrecargado = new
ProductoPrecargadoVo();

productoPrecargado.setNombre(rs.getString("pp_nombre"));

productoPrecargado.setImagen(rs.getBlob("pp_imagen"));
                    listaProductosPrecargados.add(productoPrecargado);
              }
      }

      @Override
      public void run() {
            //SelectAll
             if (operacion =="selectAll"){
                   try{
               //obtenemos el driver de para mysql
               Class.forName("com.mysql.jdbc.Driver");
               //obtenemos la conexion
               Connection conn =
DriverManager.getConnection(Conexion.getUrl(),Conexion.getLogin(),Cone
xion.getPassword());
                String stsql = "SELECT * FROM  `Productos` where
visible = 1 and heladera_id = "+
Cache.getInstance().getHeladera().getIdHeladera() + " ORDER BY
`fechaCaducidad` ";
              Statement st = conn.createStatement();
              resultSetToArrayList(st.executeQuery(stsql));
              conn.close();
                if (conn!=null){
                   conn=null;
                }
            }
            catch(SQLException e){
               System.out.println(e);
```

```
            }catch(ClassNotFoundException e){
               System.out.println(e);
            }catch(Exception e){
               System.out.println(e);
            }
             }
            //SelectProductosPrecargados
             if (operacion =="selectProductosPrecargados"){
                   try{
               //obtenemos el driver de para mysql
               Class.forName("com.mysql.jdbc.Driver");
               //obtenemos la conexion
               Connection conn =
DriverManager.getConnection(Conexion.getUrl(),Conexion.getLogin(),Cone
xion.getPassword());
             String stsql = "SELECT DISTINCT pp_nombre, pp_imagen FROM
((SELECT DISTINCT pp_nombre, pp_imagen FROM `PRODUCTOS_PRECARGADOS`)
UNION (SELECT DISTINCT nombre, imagen FROM `Productos`)) AS a group by
a.pp_nombre ORDER BY a.pp_nombre";
             Statement st = conn.createStatement();
             resultSetToArrayPrecargadosList(st.executeQuery(stsql));
             conn.close();
               if (conn!=null){
                  conn=null;
               }
            }
            catch(SQLException e){
               System.out.println(e);
            }catch(ClassNotFoundException e){
               System.out.println(e);
            }catch(Exception e){
               System.out.println(e);
            }
             }
            //Alta
             if (operacion == "alta"){
             try{
               //obtenemos el driver de para mysql
               Class.forName("com.mysql.jdbc.Driver");
               //obtenemos la conexion
               Connection conn =
DriverManager.getConnection(Conexion.getUrl(),Conexion.getLogin(),Cone
xion.getPassword());
             String stsql = "INSERT INTO `Productos`(`idProducto`,
`nombre`, `descripcion`, `fechaRegistro`, `fechaCaducidad`, `imagen`,
`heladera_id`) VALUES (NULL,?,?,?,?,?,?)";

             PreparedStatement pstmt = conn.prepareStatement(stsql);
             pstmt.setString(1, producto.getNombre());
             pstmt.setString(2, producto.getDescripcion());
             pstmt.setString(3, getFechaActual());
             pstmt.setString(4,
getFechaCaducidadFormato(producto.getFechaCaducidad()));
             pstmt.setBytes(5, producto.getImagen());
             pstmt.setInt(6,
Cache.getInstance().getHeladera().getIdHeladera());

             pstmt.executeUpdate();
             conn.close();
               if (conn!=null){
                  System.out.println("Coneccion a base de datos OK");
```

```
                conn=null;
            }
        }
        catch(SQLException e){
            System.out.println(e);
        }catch(ClassNotFoundException e){
            System.out.println(e);
        }catch(Exception e){
            System.out.println(e);
        }
         }
        //Modificacion
         if (operacion == "modificacion"){
         try{
            //obtenemos el driver de para mysql
            Class.forName("com.mysql.jdbc.Driver");
            //obtenemos la conexion
            Connection conn =
DriverManager.getConnection(Conexion.getUrl(),Conexion.getLogin(),Cone
xion.getPassword());
             String stsql = "UPDATE `Productos` SET `nombre` =
'"+producto.getNombre()+"', `fechaCaducidad` =
'"+getFechaCaducidadFormato(producto.getFechaCaducidad())+ "' ,
`imagen` = ? Where idProducto = " + producto.getIdProducto();
             PreparedStatement pstmt = conn.prepareStatement(stsql);
             pstmt.setBytes(1, producto.getImagen());
             pstmt.executeUpdate();
             conn.close();
            if (conn!=null){
               conn=null;
            }
        }
        catch(SQLException e){
            System.out.println(e);
        }catch(ClassNotFoundException e){
            System.out.println(e);
        }catch(Exception e){
            System.out.println(e);
        }
         }
         //Eliminar
         if(operacion == "eliminar"){
               try{
                 //obtenemos el driver de para mysql
                 Class.forName("com.mysql.jdbc.Driver");
                 //obtenemos la conexion
                 Connection conn =
DriverManager.getConnection(Conexion.getUrl(),Conexion.getLogin(),Cone
xion.getPassword());

                 for(Integer identificador: this.identificadores){
                     String stsql = "UPDATE `Productos` SET visible
= 0 WHERE idProducto = '"+identificador+"'";
                     Statement st = conn.createStatement();
                     st.executeUpdate(stsql);
                 }

               conn.close();
                 if (conn!=null){
                    conn=null;
                 }
```

```
                }
                catch(SQLException e){
                   System.out.println(e);
                }catch(ClassNotFoundException e){
                   System.out.println(e);
                }catch(Exception e){
                   System.out.println(e);
                }
             }
      }
      public List<ProductoVo> getListaProducto() {
            return listaProducto;
      }

      public void setListaProducto(List<ProductoVo> listaProducto) {
            this.listaProducto = listaProducto;
      }

      public List<ProductoPrecargadoVo> getListaProductosPrecargados()
{
            return listaProductosPrecargados;
      }

      public void setListaProductosPrecargados(
                  List<ProductoPrecargadoVo> listaProductosPrecargados)
{
            this.listaProductosPrecargados = listaProductosPrecargados;
      }
}
```

3.2.2.3.2 Clase HeladeraDao

```
package modelo.dao;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;

import modelo.conexion.Conexion;
import modelo.vo.HeladeraVo;
/**
 * Clase que oficia de Data Access Object para entidades HELADERAS
 * @author jgomezva
 *
 */
public class HeladeraDao extends Thread {

      private static String operacion;
      private List<HeladeraVo> listaHeladera;

      public List<HeladeraVo> obtenerHeladerasAll(){
            operacion = "selectAll";
            this.start();
            return listaHeladera;
    }

      @Override
```

```
	public void run() {
		//SelectAll
		 if (operacion =="selectAll"){
			 try{
		   //obtenemos el driver de para mysql
		   Class.forName("com.mysql.jdbc.Driver");
		   //obtenemos la conexion
		   Connection conn =
DriverManager.getConnection(Conexion.getUrl(),Conexion.getLogin(),Cone
xion.getPassword());
		   String stsql = "SELECT * FROM  `HELADERAS` ORDER BY
`heladera_id`";
		   Statement st = conn.createStatement();
		   resultSetToArrayList(st.executeQuery(stsql));
		   conn.close();
		     if (conn!=null){
		        conn=null;
		     }
		  }
		  catch(SQLException e){
		     System.out.println(e);
		  }catch(ClassNotFoundException e){
		     System.out.println(e);
		  }catch(Exception e){
		     System.out.println(e);
		  }
		   }
		   }
		   public void resultSetToArrayList(ResultSet rs) throws
SQLException{
		                listaHeladera = new ArrayList<HeladeraVo>();
		                while (rs.next()) {
		                       HeladeraVo heladera = new HeladeraVo();

heladera.setIdHeladera(rs.getInt("heladera_id"));

heladera.setDescripcion(rs.getString("heladera_descripcion"));

heladera.setImagen(rs.getBlob("heladera_imagen"));
		                       listaHeladera.add(heladera);
		                }
		             }

		  public List<HeladeraVo> getListaHeladera() {
		        return listaHeladera;
		  }

		  public void setListaHeladera(List<HeladeraVo>
listaHeladera) {
		        this.listaHeladera = listaHeladera;
		  }

}
```

3.2.2.3.3 Clase HeladeraDao

```
package modelo.dao;

import java.util.HashMap;
import java.util.Map;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import control.ExcepcionesGenerales;
import modelo.conexion.Conexion;
import modelo.vo.ConfiguracionVo;
/**
 * Clase que oficia de Data Access Object para entidades
CONFIGURACIONES
 * @author jgomezva
 *
 */
public class ConfiguracionDao extends Thread {

	private static String operacion;
	Map<String,Integer> configuraciones;

	public Map<String,Integer> obtenerParametrosAll(){
		operacion = "selectAll";
		this.start();
		return configuraciones;
    }

	public void modificarConfiguracion(Map<String,Integer>
configuraciones) throws ExcepcionesGenerales{
		operacion = "modificacion";
		if(configuraciones == null){
			throw new ExcepcionesGenerales("no_data");
		}

	if(configuraciones.get("dias_precaucion")<configuraciones.get("d
ias_alerta")){
			throw new
ExcepcionesGenerales("configuracion_error_dias");
		}
		Cache.getInstance().setConfiguraciones(configuraciones);
		this.setConfiguraciones(configuraciones);
		this.start();
    }

	@Override
	public void run() {
		//SelectAll
		 if (operacion =="selectAll"){
			 try{
			//obtenemos el driver de para mysql
			Class.forName("com.mysql.jdbc.Driver");
			//obtenemos la conexion
			Connection conn =
DriverManager.getConnection(Conexion.getUrl(),Conexion.getLogin(),Cone
xion.getPassword());
			String stsql = "SELECT * FROM  `CONFIGURACION`";
```

```
                Statement st = conn.createStatement();
                resultSetToArrayList(st.executeQuery(stsql));
                conn.close();
                  if (conn!=null){
                     conn=null;
                  }
               }
               catch(SQLException e){
                  System.out.println(e);
               }catch(ClassNotFoundException e){
                  System.out.println(e);
               }catch(Exception e){
                  System.out.println(e);
               }
                }
               //Modificacion
                if (operacion == "modificacion"){
                try{
                  //obtenemos el driver de para mysql
                  Class.forName("com.mysql.jdbc.Driver");
                  //obtenemos la conexion
                  Connection conn =
DriverManager.getConnection(Conexion.getUrl(),Conexion.getLogin(),Cone
xion.getPassword());
                // modificar dias precaucion
                  String stsql = "UPDATE `CONFIGURACION` SET
`configuracion_valor` = ? Where configuracion_parametro =
'dias_precaucion'";
                PreparedStatement pstmt = conn.prepareStatement(stsql);
                pstmt.setInt(1,
this.getConfiguraciones().get("dias_precaucion"));
                pstmt.executeUpdate();
                // modificar dias alerta
                stsql = "UPDATE `CONFIGURACION` SET `configuracion_valor`
= ? Where configuracion_parametro = 'dias_alerta'";
                pstmt = conn.prepareStatement(stsql);
                pstmt.setInt(1,
this.getConfiguraciones().get("dias_alerta"));
                pstmt.executeUpdate();
                conn.close();
                  if (conn!=null){
                     conn=null;
                  }
               }
               catch(SQLException e){
                  System.out.println(e);
               }catch(ClassNotFoundException e){
                  System.out.println(e);
               }catch(Exception e){
                  System.out.println(e);
               }
                }
                }
                public void resultSetToArrayList(ResultSet rs) throws
SQLException{
                        configuraciones= new HashMap<String,Integer>();
                          while (rs.next()) {
                                ConfiguracionVo configuracion = new
ConfiguracionVo();
```

```
      configuracion.setParametro(rs.getString("configuracion_parametro
"));

      configuracion.setValor(rs.getInt("configuracion_valor"));

configuraciones.put(configuracion.getParametro(),configuracion.getValo
r());
                  }
            }
            public static String getOperacion() {
                  return operacion;
            }
            public static void setOperacion(String operacion) {
                  ConfiguracionDao.operacion = operacion;
            }

           public Map<String, Integer> getConfiguraciones() {
                  return configuraciones;
           }

           public void setConfiguraciones(Map<String, Integer>
configuraciones) {
                  this.configuraciones = configuraciones;
           }
}
```

3.2.2.3.4 Clase Cache

```
package modelo.dao;

import java.util.List;
import java.util.Map;

import modelo.vo.HeladeraVo;
import modelo.vo.ProductoPrecargadoVo;
import modelo.vo.ProductoVo;
/**
 * Implementacion de memoria cache para toda la aplicación.
 * @author jgomezva
 *
 */
public class Cache {
      private List<ProductoVo> productosList;
      private List<HeladeraVo> heladerasList;
      private List<ProductoPrecargadoVo> productosPrecargadosList;
      private Map<String,Integer> configuraciones;
      private int indice;
      private HeladeraVo heladera;


         // El constructor debe ser Protected para evitar su acceso
desde fuera.
         protected Cache() {
             //Se carga lista de productos precargados desde la BD.
               cargarProductosPrecargados();

              //Se carga Map con parametros de configuracion para la
aplicacion desde la BD.
              ConfiguracionDao configuracionDao = new
ConfiguracionDao();
```

```
                configuracionDao.obtenerParametrosAll();
                try {
                    configuracionDao.join();
                  } catch (InterruptedException e1) {
                        e1.printStackTrace();
                  }
                configuraciones =
configuracionDao.getConfiguraciones();
          }
        public void cargarProductosPrecargados(){
        //Se carga lista de productos precargados desde la BD.
         ProductoDao productoDao = new ProductoDao();
        productoDao.obtenerProductosPrecargados();
        try {
                    productoDao.join();
             } catch (InterruptedException e1) {
                    e1.printStackTrace();
             }
        productosPrecargadosList =
productoDao.getListaProductosPrecargados();
        }
        // Clase estatica oculta. Tan solo se instanciara el singleton
una vez
          private static class CacheHolder {
         // El constructor de Singleton puede ser llamado desde aquí al
ser protected
            private final static Cache INSTANCE = new Cache();
          }
          // Método para obtener la instancia de nuestra clase
          public static Cache getInstance() {
            return CacheHolder.INSTANCE;
          }
          public List<ProductoVo> getProductosList() {
                return productosList;
          }
          public void setProductosList(List<ProductoVo> productosList)
{
                this.productosList = productosList;
          }
          public int getIndice() {
                return indice;
          }
          public void setIndice(int indice) {
                this.indice = indice;
          }
          public List<ProductoPrecargadoVo>
getProductosPrecargadosList() {
                return productosPrecargadosList;
          }
          public void
setProductosPrecargadosList(List<ProductoPrecargadoVo>
productosPrecargadosList) {
                this.productosPrecargadosList =
productosPrecargadosList;
          }
          public List<HeladeraVo> getHeladerasList() {
                return heladerasList;
          }
          public void setHeladerasList(List<HeladeraVo> heladerasList)
{
                this.heladerasList = heladerasList;
```

```
	}
	public HeladeraVo getHeladera() {
		return heladera;
	}
	public void setHeladera(HeladeraVo heladera) {
		this.heladera = heladera;
	}
	public Map<String, Integer> getConfiguraciones() {
		return configuraciones;
	}
	public void setConfiguraciones(Map<String, Integer>
configuraciones) {
		this.configuraciones = configuraciones;
	}
}
```

3.2.2.5 Paquete control

3.2.2.5.1 Clase FragmentListadoProducto

```
package control;

import java.util.ArrayList;
import java.util.List;

import modelo.dao.Cache;
import modelo.dao.ProductoDao;
import modelo.vo.ProductoVo;
import tesina.qr.R;
import android.app.Fragment;
import android.app.FragmentManager;
import android.app.FragmentTransaction;
import android.content.Intent;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.view.LayoutInflater;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.ViewGroup;
import android.widget.CheckBox;
import android.widget.ImageButton;
import android.widget.ListView;
import android.widget.Toast;
/**
 * Clase responsable de contol de pantalla Listado Productos.
 * @author jgomezva
 *
 */
public class FragmentListadoProductos extends Fragment {


	public FragmentListadoProductos(){
	}

	private class DetalleDialogFragmentDismissHandler extends
Handler {
		@Override
		public void handleMessage(Message msg) {
		    super.handleMessage(msg);
		 // Refrescar Fragment
```

```
                        Fragment newFragment = new
FragmentTabListado();
                    FragmentTransaction transaction =
getFragmentManager().beginTransaction();
                    transaction.replace(R.id.fragment_container,
newFragment);
                    transaction.addToBackStack(null);
                    transaction.commit();
                }
            }

      @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
container,
            Bundle savedInstanceState) {
        View rootView = inflater.inflate(R.layout.listado_producto,
container, false);
        ListView listViewProductos = (ListView)
rootView.findViewById(R.id.ProductosList);
        ImageButton botonBorrar = (ImageButton)
rootView.findViewById(R.id.buttonListadoBorrar);
        botonBorrar.setOnClickListener(onClickListenerProducto);
        ImageButton botonVer = (ImageButton)
rootView.findViewById(R.id.buttonVer);
        botonVer.setOnClickListener(onClickListenerVerProducto);
        ImageButton botonVolver = (ImageButton)
rootView.findViewById(R.id.botonVolver);
        botonVolver.setOnClickListener(onClickListenerVolver);

        ProductoDao productoDao = new ProductoDao();
        productoDao.obtenerProductosAll();
        try {
                productoDao.join();
            } catch (InterruptedException e1) {
                e1.printStackTrace();
            }

Cache.getInstance().setProductosList(productoDao.getListaProducto());

        if (Cache.getInstance().getProductosList() != null &&
!Cache.getInstance().getProductosList().isEmpty()){
            listViewProductos = ( ListView )
rootView.findViewById(R.id.ProductosList);
            listViewProductos.setAdapter(new
ProductoListAdapter(this.getActivity().getApplicationContext(),
R.layout.producto_row_item, Cache.getInstance().getProductosList()));
      }else{
        Toast.makeText(rootView.getContext(),
getResources().getString(R.string.alta_no_heladera_seleccionada) ,
Toast.LENGTH_LONG).show();
      }
      return rootView;
    }

       private OnClickListener onClickListenerVerProducto = new
OnClickListener() {
           @Override
           public void onClick(View v) {
             if(v.getId()==R.id.buttonVer){
                   List<Integer> idsList = new ArrayList<Integer>();
                   ProductoVo productoSeleccionado = new ProductoVo();
```

```
                        for(ProductoVo producto:
Cache.getInstance().getProductosList()){
                            CheckBox checkBox = (CheckBox)
getActivity().findViewById(producto.getIdProducto());
                            if (checkBox!= null &&
checkBox.isChecked()){

idsList.add(producto.getIdProducto());
                                productoSeleccionado = producto;
                            }
                        }
                        if(idsList.isEmpty()){
                            Toast.makeText(v.getContext(),
getResources().getString(R.string.alta_error_detalle) ,
Toast.LENGTH_LONG).show();
                        }
                        if(idsList.size()>1){
                            Toast.makeText(v.getContext(),
getResources().getString(R.string.alta_ha_seleccionado) +" " +
idsList.size() + " "+
getResources().getString(R.string.alta_seleccionar_detalle) ,
Toast.LENGTH_LONG).show();
                        }
                        if(idsList.size()==1){
                            FragmentManager fm =
getFragmentManager();
                            DialogDetalle dialogDetalle = new
DialogDetalle(productoSeleccionado, new
DetalleDialogFragmentDismissHandler());

      dialogDetalle.setRetainInstance(true);
                            dialogDetalle.show(fm,
"fragment_detalle");
                        }
                    }
            });

      private OnClickListener onClickListenerVolver = new
OnClickListener() {
                @Override
                public void onClick(View v) {
            if(v.getId()==R.id.botonVolver){
                Cache.getInstance().setHeladera(null);
                Cache.getInstance().setHeladerasList(null);
                Intent i = v.getContext().getPackageManager()
                        .getLaunchIntentForPackage(
v.getContext().getPackageName() );
                i.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                startActivity(i);
            }
            });

    private OnClickListener onClickListenerProducto = new
OnClickListener() {
            @Override
            public void onClick(View v) {
                List<Integer> idsList = new ArrayList<Integer>();
                ProductoDao productoDao = new ProductoDao();
                if(v.getId()==R.id.buttonListadoBorrar){
                        for(ProductoVo producto:
Cache.getInstance().getProductosList()){
```

```
                                CheckBox checkBox = (CheckBox)
getActivity().findViewById(producto.getIdProducto());
                                if (checkBox!= null &&
checkBox.isChecked()){

      idsList.add(producto.getIdProducto());
                                }
                            }
                            productoDao.eliminarProducto(idsList);
                        try {
                                    productoDao.join();
                                } catch (InterruptedException e) {

                                    e.printStackTrace();
                                }
                        // Recargar listado de productos precargados
                        Cache.getInstance().cargarProductosPrecargados();

                            // Refrescar Fragment
                            Fragment newFragment = new
FragmentTabListado();
                        FragmentTransaction transaction =
getFragmentManager().beginTransaction();
                        transaction.replace(R.id.fragment_container,
newFragment);
                        transaction.addToBackStack(null);
                        transaction.commit();
                    }}
                };
}
```

3.2.2.5.2 Clase FragmentTabAlta

```
package control;

import java.io.ByteArrayOutputStream;
import java.io.File;
import java.util.ArrayList;
import java.util.List;

import modelo.dao.Cache;
import modelo.dao.ProductoDao;
import modelo.vo.ProductoPrecargadoVo;
import modelo.vo.ProductoVo;
import tesina.qr.R;
import android.app.Fragment;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.Bitmap.CompressFormat;
import android.graphics.BitmapFactory;
import android.graphics.drawable.BitmapDrawable;
import android.net.Uri;
import android.os.Bundle;
import android.os.Environment;
import android.provider.MediaStore;
import android.view.LayoutInflater;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.AutoCompleteTextView;
```

```
import android.widget.CalendarView;
import android.widget.ImageButton;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;
/**
 * Clase responsable de contol de pantalla Alta de Productos.
 * @author jgomezva
 *
 */
public class FragmentTabAlta extends Fragment {
      private ImageView imagenProducto;
      Bitmap bitmap = null;
      private Uri mImageCaptureUri;
      private static final int PICK_FROM_CAMERA = 100;
      private static final int PICK_FROM_FILE = 2;
    private String ubicacionImagen;
    private TextView noImagen;

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
container,
            Bundle savedInstanceState) {
        View rootView = inflater.inflate(R.layout.alta, container,
false);
        TextView titulo = (TextView)
rootView.findViewById(R.id.textViewAltaProductoTitulo);
        noImagen = (TextView)
rootView.findViewById(R.id.textNoImagen);
        imagenProducto = ((ImageView)
rootView.findViewById(R.id.imagenProducto));
        ImageButton buttonAceptar= (ImageButton)
rootView.findViewById(R.id.buttonAceptar);
      ImageButton buttonCamara= (ImageButton)
rootView.findViewById(R.id.buttonCamara);
      ImageButton buttonMemoria= (ImageButton)
rootView.findViewById(R.id.buttonMemoria);
      AutoCompleteTextView textView = (AutoCompleteTextView)
rootView.findViewById(R.id.autoNombre);
      textView.setOnItemClickListener(new
ProductoPrecargadoListClickListener(noImagen, imagenProducto,
Cache.getInstance().getProductosPrecargadosList()));
      CalendarView fechaCaducidad = (CalendarView)
rootView.findViewById(R.id.calendarioFechaCaducidad);
      ImageView imageFlecha =
(ImageView)rootView.findViewById(R.id.imageViewFlecha);
      if(Cache.getInstance().getHeladera()!=null){
            imageFlecha.setVisibility(ImageView.INVISIBLE);

      titulo.setText(getResources().getString(R.string.alta_titulo)+"
"+Cache.getInstance().getHeladera().getDescripcion());
            List<String> nombresPrecargados = new ArrayList<String>();
            for(ProductoPrecargadoVo precargado:
Cache.getInstance().getProductosPrecargadosList()){
                  nombresPrecargados.add(precargado.getNombre());
            }
            ArrayAdapter<String> autoCompleteAdapter = new
AutocompleteListAdapter(rootView.getContext(),
            R.layout.producto_precargado_row,nombresPrecargados);
            textView.setAdapter(autoCompleteAdapter);
            //Acciones Botones
```

```
            buttonCamara.setOnClickListener(onClickListener);
            buttonAceptar.setOnClickListener(onClickListener);
            buttonMemoria.setOnClickListener(onClickListener);
      }else{
            TextView textNombre = (TextView)
rootView.findViewById(R.id.textNombre);
            TextView textImagen = (TextView)
rootView.findViewById(R.id.textViewImagenAlta);

            textNombre.setVisibility(TextView.INVISIBLE);
            textImagen.setVisibility(TextView.INVISIBLE);
            titulo.setText("
"+getResources().getString(R.string.heladera_no_seleccionada));
            textView.setVisibility(TextView.INVISIBLE);
            noImagen.setVisibility(TextView.INVISIBLE);
            buttonAceptar.setVisibility(ImageButton.INVISIBLE);
            buttonCamara.setVisibility(ImageButton.INVISIBLE);
            buttonMemoria.setVisibility(ImageButton.INVISIBLE);
            imagenProducto.setVisibility(ImageView.INVISIBLE);
            fechaCaducidad.setVisibility(CalendarView.INVISIBLE);
            Toast.makeText(rootView.getContext(),
getResources().getString(R.string.alta_error_seleccion) ,
Toast.LENGTH_LONG).show();
      }
        return rootView;
    }


    private OnClickListener onClickListener = new OnClickListener() {
           @Override
           public void onClick(View v) {
      switch(v.getId()) {
      //Funcionamiento para Boton Aceptar
   case R.id.buttonAceptar:
         AutoCompleteTextView nombre = (AutoCompleteTextView)
getView().findViewById(R.id.autoNombre);
         CalendarView fechaCaducidad = (CalendarView)
getView().findViewById(R.id.calendarioFechaCaducidad);
         imagenProducto = ((ImageView)
getView().findViewById(R.id.imagenProducto));

       ProductoVo producto = new ProductoVo();
       producto.setNombre(nombre.getText().toString());
       producto.setFechaCaducidad(fechaCaducidad.getDate());

       Bitmap bmp =
((BitmapDrawable)imagenProducto.getDrawable()).getBitmap();
       ByteArrayOutputStream stream = new ByteArrayOutputStream();
       Bitmap resizedbitmap = Bitmap.createScaledBitmap(bmp, 100, 100,
true);
       resizedbitmap.compress(CompressFormat.PNG, 50, stream);
       producto.setImagen(stream.toByteArray());
       ProductoDao productoDao = new ProductoDao();
         try {
                  productoDao.registrarProducto(producto);
                  Toast.makeText(v.getContext(),
getResources().getString(R.string.alta_ha_registrado) +" "+
producto.getNombre() +
getResources().getString(R.string.correctamente)  ,
Toast.LENGTH_SHORT).show();
                  producto = new ProductoVo();
```

```
                producto.setNombre("");
                producto.setImagen(null);
                imagenProducto.setImageBitmap(null);
                nombre.setText("");
            } catch (ExcepcionesGenerales e1) {
                Toast.makeText(v.getContext(), e1.getMessage(),
Toast.LENGTH_LONG).show();
            }

   break;
   case R.id.buttonCamara:

         Intent intent    = new
Intent(MediaStore.ACTION_IMAGE_CAPTURE);
       File file        = new
File(Environment.getExternalStorageDirectory(),
                           "tmp_qricebox_" +
String.valueOf(System.currentTimeMillis()) + ".png");
       mImageCaptureUri = Uri.fromFile(file);
       ubicacionImagen = mImageCaptureUri.getPath();
       try {
           intent.putExtra(android.provider.MediaStore.EXTRA_OUTPUT,
mImageCaptureUri);
           intent.putExtra("return-data", true);

           startActivityForResult(intent, PICK_FROM_CAMERA);
       } catch (Exception e) {
           e.printStackTrace();
       }
   break;
   case R.id.buttonMemoria:
         intent = new Intent();

       intent.setType("image/*");
       intent.setAction(Intent.ACTION_GET_CONTENT);

       startActivityForResult(Intent.createChooser(intent, "Complete
action using"), PICK_FROM_FILE);
   break;
 }
}
};

@Override
public void onActivityResult(int requestCode, int resultCode, Intent
data) {
    if (resultCode != -1) return;
    Bitmap bitmap   = null;
    this.ubicacionImagen     = "";

    if (requestCode == PICK_FROM_FILE) {
          this.ubicacionImagen     = "";
          Uri selectedImageUri = data.getData();
          ubicacionImagen =
ImageFilePath.getPath(this.getActivity().getApplicationContext(),
selectedImageUri);
    } else {
      ubicacionImagen    = mImageCaptureUri.getPath();
    }
    bitmap  = BitmapFactory.decodeFile(ubicacionImagen);
    imagenProducto.setImageBitmap(bitmap);
```

```
    noImagen.setVisibility(TextView.INVISIBLE);
}
}
```

3.2.2.5.3 Clase FragmentTabConfiguracion

```
package control;

import java.util.HashMap;
import java.util.Locale;
import java.util.Map;

import modelo.dao.Cache;
import modelo.dao.ConfiguracionDao;
import tesina.qr.R;
import android.app.Fragment;
import android.app.FragmentTransaction;
import android.content.res.Configuration;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageButton;
import android.widget.SeekBar;
import android.widget.TextView;
import android.widget.Toast;
/**
 * Clase responsable de contol de pantalla Configuracion.
 * @author jgomezva
 *
 */
public class FragmentTabConfiguracion extends Fragment {
      private SeekBar seekBarPrecaucion = null;
      private SeekBar seekBarAlerta = null;
      private TextView textViewAmarillo = null;
      private TextView textViewRojo = null;
      private int diasPrecaucion;
      private int diasAlerta;

      public FragmentTabConfiguracion(){
      }
      @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
container,
            Bundle savedInstanceState) {
        View rootView = inflater.inflate(R.layout.configuracion,
container, false);
        ImageButton botonEspanol = (ImageButton)
rootView.findViewById(R.id.imageButtonEspanol);
        ImageButton botonIngles = (ImageButton)
rootView.findViewById(R.id.imageButtonIngles);
        ImageButton botonPortugues = (ImageButton)
rootView.findViewById(R.id.imageButtonPortugues);
        botonEspanol.setOnClickListener(onClickListenerGuardar);
        botonIngles.setOnClickListener(onClickListenerGuardar);
        botonPortugues.setOnClickListener(onClickListenerGuardar);

        ArrayAdapter<CharSequence> adapter =
ArrayAdapter.createFromResource(rootView.getContext(),
```

```
            R.array.idiomas_array,
android.R.layout.simple_spinner_item);

adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdo
wn_item);

        diasPrecaucion =
Cache.getInstance().getConfiguraciones().get("dias_precaucion");
        diasAlerta =
Cache.getInstance().getConfiguraciones().get("dias_alerta");

        ImageButton buttonGuardar = (ImageButton)
rootView.findViewById(R.id.imageButtonGuardarConfiguracion);
        buttonGuardar.setOnClickListener(onClickListenerGuardar);
        seekBarPrecaucion = (SeekBar)
rootView.findViewById(R.id.seekBarAmarillo);
        textViewAmarillo = (TextView)
rootView.findViewById(R.id.textViewAmarillo);

textViewAmarillo.setText(getResources().getString(R.string.configuraci
on_texto_precaucion)+" "+diasPrecaucion +"
"+getResources().getString(R.string.dias)+".");
        seekBarAlerta = (SeekBar)
rootView.findViewById(R.id.seekBarRojo);
        textViewRojo = (TextView)
rootView.findViewById(R.id.textViewRojo);

textViewRojo.setText(getResources().getString(R.string.configuracion_t
exto_alerta)+" "+diasAlerta +" "+
getResources().getString(R.string.dias)+".");
        seekBarPrecaucion.setProgress(diasPrecaucion);
        seekBarAlerta.setProgress(diasAlerta);

            seekBarPrecaucion.setOnSeekBarChangeListener(new
OnSeekBarPrecaucionChangeListener(textViewAmarillo,getResources().getS
tring(R.string.configuracion_texto_precaucion),getResources().getStrin
g(R.string.dias)));
            seekBarAlerta.setOnSeekBarChangeListener(new
OnSeekBarAlertaChangeListener(textViewRojo,getResources().getString(R.
string.configuracion_texto_alerta),getResources().getString(R.string.d
ias)));
        return rootView;
      }

      @Override
      public void onConfigurationChanged(Configuration newConfig) {
      super.onConfigurationChanged(newConfig);
      // Refrescar Fragment
      Fragment newFragment = new FragmentTabConfiguracion();
          FragmentTransaction transaction =
getFragmentManager().beginTransaction();
          transaction.replace(R.id.fragment_container, newFragment);
          transaction.addToBackStack(null);
          transaction.commit();
      }

      private OnClickListener onClickListenerGuardar = new
OnClickListener() {
           private String idiomaSeleccionado;
           @Override
           public void onClick(View v) {
```

```
            if(v.getId()==R.id.imageButtonIngles){
                  idiomaSeleccionado = "en_us";
            }
            if(v.getId()==R.id.imageButtonEspanol){
                  idiomaSeleccionado = "es_es";
            }
            if(v.getId()==R.id.imageButtonPortugues){
                  idiomaSeleccionado = "pt_br";
            }
            if(idiomaSeleccionado!=null){

if((!idiomaSeleccionado.equalsIgnoreCase(Locale.getDefault().toString(
).toLowerCase()))){
                        Locale locale = new
Locale(idiomaSeleccionado);
                        Locale.setDefault(locale);

                        Configuration config = new Configuration();
                        config.locale = locale;

v.getContext().getApplicationContext().getResources().updateConfigurat
ion(config, null);
                        onConfigurationChanged(config);
                  }
            }
            if(v.getId()==R.id.imageButtonGuardarConfiguracion){
                  Map<String,Integer> configuraciones = new
HashMap<String,Integer>();
                  configuraciones.put("dias_precaucion",
seekBarPrecaucion.getProgress());

configuraciones.put("dias_alerta",seekBarAlerta.getProgress());
                  ConfiguracionDao configuracionDao = new
ConfiguracionDao();

                  try {

configuracionDao.modificarConfiguracion(configuraciones);

Cache.getInstance().setConfiguraciones(configuraciones);
                        Toast.makeText(v.getContext(),
getResources().getString(R.string.frase_nivel_precaucion_alerta) +"
"+seekBarPrecaucion.getProgress()+" "+
getResources().getString(R.string.dias_alerta) +"
"+seekBarAlerta.getProgress()+"
"+getResources().getString(R.string.dias), Toast.LENGTH_LONG).show();
                  } catch (ExcepcionesGenerales e) {
                        String mensaje="";
                        if (e.getMessage()=="nodata"){
                              mensaje =
getResources().getString(R.string.no_data);
                        }
                        if
(e.getMessage()=="configuracion_error_dias"){
                              mensaje =
getResources().getString(R.string.configuracion_error_dias);
                        }
                        Toast.makeText(v.getContext(), mensaje,
Toast.LENGTH_LONG).show();
                     seekBarPrecaucion.setProgress(diasPrecaucion);
```

```
                        seekBarAlerta.setProgress(diasAlerta);
                    }
                }
            }};
}
```

3.2.2.5.4 Clase FragmentTabConfiguracion

```
package control;

import java.util.Date;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import modelo.dao.Cache;
import modelo.dao.HeladeraDao;
import modelo.dao.ProductoDao;
import modelo.vo.HeladeraVo;
import modelo.vo.ProductoVo;
import tesina.qr.R;
import utils.Utils;
import android.app.Fragment;
import android.app.FragmentTransaction;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ListView;
import android.widget.Toast;
/**
 * Clase responsable de contol de pantalla Listado de Heladeras.
 * @author jgomezva
 *
 */
public class FragmentTabListadoHeladera extends Fragment {
      public FragmentTabListadoHeladera(){
      }

      @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
container,
            Bundle savedInstanceState) {
          View rootView = inflater.inflate(R.layout.listado_heladera,
container, false);


            HeladeraDao heladeraDao = new HeladeraDao();
             if(Cache.getInstance().getHeladerasList()==null){
                  heladeraDao.obtenerHeladerasAll();
                  try {
                        heladeraDao.join();
                  } catch (InterruptedException e1) {
                        e1.printStackTrace();
                  }

      Cache.getInstance().setHeladerasList(heladeraDao.getListaHelader
a());
             }
```

```
            if (Cache.getInstance().getHeladerasList() != null &&
!Cache.getInstance().getHeladerasList().isEmpty()){
                    ListView listViewHeladeras = (ListView)
rootView.findViewById(R.id.HeladerasList);
                    listViewHeladeras.setAdapter(new
HeladeraListAdapter(this.getActivity().getApplicationContext(),
R.layout.heladera_row_item, Cache.getInstance().getHeladerasList()));

informacionAdicionalHeladeras(Cache.getInstance().getHeladerasList());
                    Cache.getInstance().setHeladera(null);
                    listViewHeladeras.setOnItemClickListener(new
OnItemClickListener() {
                        @Override
                        public void onItemClick(AdapterView<?> parent,
View view, int position, long id) {
                            HeladeraVo o = (HeladeraVo)
parent.getItemAtPosition(position);
                             Cache.getInstance().setHeladera(o);
                         //reemplazar Fragment de Heladera por el de
Productos.
                         FragmentTransaction trans =
getFragmentManager().beginTransaction();
                            trans.replace(R.id.layoutRaizListado, new
FragmentListadoProductos());

trans.setTransition(FragmentTransaction.TRANSIT_FRAGMENT_OPEN);
                            trans.addToBackStack(null);
                            trans.commit();
                        }
                    });
             }else{
              Toast.makeText(rootView.getContext(),
getResources().getString(R.string.listado_heladera_no_ingresada) ,
Toast.LENGTH_LONG).show();
            }
            return rootView;
      }
   public String validarLink(String link){
         String[] params = link.split("&");
           Map<String, String> map = new HashMap<String, String>();
           for (String param : params)
           {
               String name = param.split("=")[0];
               String value = param.split("=")[1];
               map.put(name, value);
           }
           if(map.get("hel").isEmpty()){
             return null;
           }else{
             return map.get("hel");
           }
   }

   public void informacionAdicionalHeladeras(List<HeladeraVo>
heladerasList){
        Integer diasPrecaucion =
Cache.getInstance().getConfiguraciones().get("dias_precaucion");
      Integer  diasAlerta =
Cache.getInstance().getConfiguraciones().get("dias_alerta");
        for(HeladeraVo heladera: heladerasList){
              ProductoDao productoDao = new ProductoDao();
```

```
            Cache.getInstance().setHeladera(heladera);
              productoDao.obtenerProductosAll();
              try {
                        productoDao.join();
                    } catch (InterruptedException e1) {
                        e1.printStackTrace();
                    }
              List<ProductoVo> productosList =
productoDao.getListaProducto();
              if(productosList!=null){
                  //cantidad de productos por heladera
                  heladera.setCantidadProductos(productosList.size());
                  int dias;
                  int cantidadProductosVencidos=0;
                  int cantidadProductosEnRiesgo=0;
                  int cantidadProductosPrecaucion=0;
                  for(ProductoVo producto: productosList){
                        Date fechaVencimiento = new
Date(producto.getFechaCaducidad());
                     dias =
Utils.getInstance().getDifferenceDays(Utils.getInstance().getFechaHoy(
),fechaVencimiento);
                        //cantidad de productos vencidos
                        if(dias<0){
                              cantidadProductosVencidos++;
                    }else if(dias<=diasAlerta){
                              cantidadProductosEnRiesgo++;
                    }else if(dias>diasAlerta && dias <diasPrecaucion){
                              cantidadProductosPrecaucion++;
                    }
                  }

      heladera.setCantidaVencidos(cantidadProductosVencidos);

      heladera.setCantidadEnRiesgo(cantidadProductosEnRiesgo);

      heladera.setCantidadConPrecaucion(cantidadProductosPrecaucion);
                  }else{
                    heladera.setCantidadProductos(0);
                  }
        }
      }
}
```

3.2.2.5.5 Clase DialogDetalle

```
package control;

import java.io.ByteArrayOutputStream;
import java.io.File;
import java.util.ArrayList;
import java.util.List;

import modelo.dao.Cache;
import modelo.dao.ProductoDao;
import modelo.vo.ProductoPrecargadoVo;
import modelo.vo.ProductoVo;
import tesina.qr.R;
import android.app.DialogFragment;
import android.content.DialogInterface;
import android.content.Intent;
```

```
import android.graphics.Bitmap;
import android.graphics.Bitmap.CompressFormat;
import android.graphics.BitmapFactory;
import android.graphics.drawable.BitmapDrawable;
import android.net.Uri;
import android.os.Bundle;
import android.os.Environment;
import android.os.Handler;
import android.provider.MediaStore;
import android.view.LayoutInflater;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.AutoCompleteTextView;
import android.widget.CalendarView;
import android.widget.ImageButton;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;
/**
* Clase responsable de contol de pantalla Detalle de Producto.
* @author jgomezva
*
*/
public class DialogDetalle extends DialogFragment {
      private ProductoVo productoSeleccionado;
      Bitmap bitmap = null;
      private Uri mImageCaptureUri;
      private static final int PICK_FROM_CAMERA = 100;
      private static final int PICK_FROM_FILE = 2;
    private String ubicacionImagen;
    private ImageView imagenProducto;
      Handler handler;

      public DialogDetalle() {
      }

      public DialogDetalle(ProductoVo o, Handler handler) {
            productoSeleccionado = o;
            this.handler = handler;
      }

    @Override
    public void onDismiss(DialogInterface dialog) {
        super.onDismiss(dialog);
        handler.sendEmptyMessage(0);
    }

      @Override
      public View onCreateView(LayoutInflater inflater, ViewGroup
container,
                  Bundle savedInstanceState) {
            View view = inflater.inflate(R.layout.detalle, container);

      getDialog().setTitle(getResources().getString(R.string.producto_
detalle_titulo));
            imagenProducto = (ImageView)
view.findViewById(R.id.imageViewProducto);
            ImageButton buttonCamara= (ImageButton)
view.findViewById(R.id.buttonCamaraDetalle);
```

```
	ImageButton buttonMemoria= (ImageButton)
view.findViewById(R.id.buttonMemoriaDetalle);
		List<String> nombresPrecargados = new ArrayList<String>();
		for(ProductoPrecargadoVo precargado:
Cache.getInstance().getProductosPrecargadosList()){
			nombresPrecargados.add(precargado.getNombre());
		}
	ArrayAdapter<String> autoCompleteAdapter = new
AutocompleteListAdapter(view.getContext(),R.layout.producto_precargado
_row,nombresPrecargados);
	  AutoCompleteTextView textView = (AutoCompleteTextView)
view.findViewById(R.id.editNombre);
		textView.setOnItemClickListener(new
ProductoPrecargadoListClickListener(null, imagenProducto,
Cache.getInstance().getProductosPrecargadosList()));
		textView.setAdapter(autoCompleteAdapter);


		CalendarView fechaCaducidad = (CalendarView) view
				.findViewById(R.id.calendarVencimiento);
		ImageButton buttonGuardar = (ImageButton)
view.findViewById(R.id.buttonGuardar);

		textView.setText(productoSeleccionado.getNombre());

	fechaCaducidad.setDate(productoSeleccionado.getFechaCaducidad())
;
		if(productoSeleccionado.getImagen()!=null){
			Bitmap bm =
BitmapFactory.decodeByteArray(productoSeleccionado.getImagen(), 0,
productoSeleccionado.getImagen().length);
			imagenProducto.setImageBitmap(bm);
		}
		// Acciones Botones
	  buttonCamara.setOnClickListener(onClickListener);
	buttonMemoria.setOnClickListener(onClickListener);
		buttonGuardar.setOnClickListener(onClickListener);
		return view;
	}

	private OnClickListener onClickListener = new OnClickListener()
{
		@Override
		public void onClick(View view) {
			switch (view.getId()) {
			// Funcionamiento para Boton Aceptar
			case R.id.buttonGuardar:
				TextView nombre = (TextView)
getView().findViewById(
						R.id.editNombre);
				ImageView imagenProducto = (ImageView)
getView().findViewById(
						R.id.imageViewProducto);
				CalendarView fechaCaducidad = (CalendarView)
getView()

	.findViewById(R.id.calendarVencimiento);

				ProductoVo producto = new ProductoVo();

	producto.setNombre(nombre.getText().toString());
```

```
      producto.setFechaCaducidad(fechaCaducidad.getDate());

      producto.setImagen(productoSeleccionado.getImagen());

      producto.setIdProducto(productoSeleccionado.getIdProducto());
                        Bitmap bmp =
((BitmapDrawable)imagenProducto.getDrawable()).getBitmap();
                        ByteArrayOutputStream stream = new
ByteArrayOutputStream();
                        Bitmap resizedbitmap =
Bitmap.createScaledBitmap(bmp, 50, 50, true);
                        resizedbitmap.compress(CompressFormat.PNG, 50,
stream);
                        producto.setImagen(stream.toByteArray());

                        ProductoDao productoDao = new ProductoDao();
                          try {
                                productoDao.modificarProducto(producto);
                                Toast.makeText(
                                            view.getContext(),

      getResources().getString(R.string.modificado) +
producto.getNombre()
                                                              +" "+
getResources().getString(R.string.correctamente), Toast.LENGTH_SHORT)
                                            .show();

                          } catch (ExcepcionesGenerales e1) {
                                Toast.makeText(view.getContext(),
e1.getMessage(),
                                            Toast.LENGTH_LONG).show();
                          }

                         break;
                    case R.id.buttonCamaraDetalle:

                             Intent intent    = new
Intent(MediaStore.ACTION_IMAGE_CAPTURE);
                          File file        = new
File(Environment.getExternalStorageDirectory(),
                                                  "tmp_qricebox_" +
String.valueOf(System.currentTimeMillis()) + ".png");
                         mImageCaptureUri = Uri.fromFile(file);
                         ubicacionImagen = mImageCaptureUri.getPath();
                         try {

intent.putExtra(android.provider.MediaStore.EXTRA_OUTPUT,
mImageCaptureUri);
                              intent.putExtra("return-data", true);

                              startActivityForResult(intent,
PICK_FROM_CAMERA);
                         } catch (Exception e) {
                              e.printStackTrace();
                         }
                       break;
                       case R.id.buttonMemoriaDetalle:
                             intent = new Intent();

                         intent.setType("image/*");
```

```
                        intent.setAction(Intent.ACTION_GET_CONTENT);

startActivityForResult(Intent.createChooser(intent, "Complete action
using"), PICK_FROM_FILE);
                        break;
                    }
                }
        };
        @Override
        public void onActivityResult(int requestCode, int resultCode,
Intent data) {
            if (resultCode != -1) return;
            Bitmap bitmap   = null;
            this.ubicacionImagen     = "";

            if (requestCode == PICK_FROM_FILE) {
                  this.ubicacionImagen     = "";
                  Uri selectedImageUri = data.getData();
                  ubicacionImagen =
ImageFilePath.getPath(this.getActivity().getApplicationContext(),
selectedImageUri);
            } else {
              ubicacionImagen    = mImageCaptureUri.getPath();
            }
            bitmap  = BitmapFactory.decodeFile(ubicacionImagen);
            imagenProducto.setImageBitmap(bitmap);
        }
}
```

3.2.2.6 Paquete Layout o Vista

3.2.2.6.1 Vista de pantalla Alta

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <GridLayout
        android:id="@+id/grid"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:columnCount="34"
        android:background="#000000"
        android:orientation="horizontal" >

        <TextView
            android:id="@+id/textViewAltaProductoTitulo"
            android:layout_column="1"
            android:layout_columnSpan="8"
            android:layout_gravity="left"
            android:layout_row="1"
            android:textColor="#2eb3e3"
            android:textSize="18sp" />

        <TextView
            android:id="@+id/textNombre"
            android:layout_column="1"
            android:layout_columnSpan="3"
```

```
        android:layout_marginLeft="5dp"
        android:layout_marginTop="8dp"
        android:layout_row="3"
        android:text="@string/alta_nombre" />

    <TextView
        android:id="@+id/textViewImagenAlta"
        android:layout_column="1"
        android:layout_columnSpan="3"
        android:layout_marginLeft="5dp"
        android:layout_row="5"
        android:layout_rowSpan="2"
        android:text="@string/alta_imagen" />

    <Space
        android:layout_width="21dp"
        android:layout_height="1dp"
        android:layout_column="1"
        android:layout_gravity="fill_horizontal"
        android:layout_row="1" />

    <Space
        android:layout_width="59dp"
        android:layout_height="1dp"
        android:layout_column="3"
        android:layout_row="1" />

    <Space
        android:layout_width="128dp"
        android:layout_height="1dp"
        android:layout_column="5"
        android:layout_row="1" />

    <Space
        android:layout_width="305dp"
        android:layout_height="1dp"
        android:layout_column="6"
        android:layout_row="1" />

    <Space
        android:layout_width="1dp"
        android:layout_height="82dp"
        android:layout_column="1"
        android:layout_row="6" />

    <Space
        android:layout_width="1dp"
        android:layout_height="61dp"
        android:layout_column="1"
        android:layout_row="7" />

    <GridLayout
        android:layout_column="5"
        android:layout_gravity="left|top"
        android:layout_row="7" >
    </GridLayout>

    <CalendarView
        android:id="@+id/calendarioFechaCaducidad"
        android:layout_width="283dp"
        android:layout_height="174dp"
```

```
        android:layout_column="3"
        android:layout_gravity="left|center_vertical"
        android:layout_marginRight="5dp"
        android:layout_row="7"
        android:layout_weight="2.38" />

    <AutoCompleteTextView
        android:id="@+id/autoNombre"
        android:layout_width="196dp"
        android:layout_height="wrap_content"
        android:layout_column="3"
        android:layout_gravity="center_horizontal|top"
        android:layout_row="3"
        android:ems="10" />

    <ImageView
        android:id="@+id/imagenProducto"
        android:layout_width="81dp"
        android:layout_height="101dp"
        android:layout_column="3"
        android:layout_gravity="center_horizontal|top"
        android:layout_marginBottom="5dp"
        android:layout_row="6"
        android:src="@drawable/sinimagen" />

    <GridLayout
        android:layout_width="102dp"
        android:layout_height="wrap_content"
        android:layout_column="3"
        android:layout_gravity="center_horizontal|top"
        android:layout_row="7"
        android:layout_weight="1.13"
        android:columnCount="5" >

        <ImageButton
            android:id="@+id/buttonCamara"
            android:layout_width="50dp"
            android:layout_height="37dp"
            android:layout_column="4"
            android:layout_gravity="left|top"
            android:layout_row="1"
            android:background="#000000"
            android:src="@drawable/ic_boton_camara" />

        <ImageButton
            android:id="@+id/buttonMemoria"
            android:layout_width="50dp"
            android:layout_height="37dp"
            android:layout_column="4"
            android:layout_gravity="right|top"
            android:layout_row="1"
            android:background="#000000"
            android:src="@drawable/ic_boton_galeria" />
    </GridLayout>

    <ImageButton
        android:id="@+id/buttonAceptar"
        android:layout_width="50dp"
        android:layout_height="37dp"
        android:layout_column="3"
        android:layout_gravity="right|bottom"
```

```
            android:layout_marginBottom="10dp"
            android:layout_row="7"
            android:background="#000000"
            android:src="@drawable/ic_boton_guardar" />

        <ImageView
            android:id="@+id/imageViewFlecha"
            android:layout_width="24dp"
            android:layout_height="24dp"
            android:layout_column="3"
            android:layout_gravity="left|top"
            android:layout_marginLeft="35dp"
            android:layout_row="1"
            android:src="@drawable/f_azul" />

        <TextView
            android:id="@+id/textNoImagen"
            android:layout_column="3"
            android:layout_gravity="center_horizontal|bottom"
            android:layout_row="6"
            android:text="@string/sin_imagen"
            android:textColor="#33b5e5" />

    </GridLayout>

</RelativeLayout>
```

3.2.2.6.2 Vista de pantalla Listado Heladeras

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_margin="3dp"
        android:gravity="center"
        android:text="@string/listado_producto_titulo"
        android:textColor="#FFFFFF"
        android:textSize="25dp" />

    <ListView
        android:id="@+id/ProductosList"
        android:layout_width="fill_parent"
        android:layout_height="200dp"
        android:layout_weight="0.54"
        android:paddingTop="5dp" >
    </ListView>

    <GridLayout
        android:layout_width="match_parent"
        android:layout_height="31dp"
        android:layout_weight="0.02"
        android:columnCount="1" >
```

```
        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="left|top"
            android:layout_row="1" >

            <ImageButton
                android:id="@+id/buttonListadoBorrar"
                android:layout_width="33dp"
                android:layout_height="40dp"
                android:layout_column="0"
                android:layout_gravity="left|top"
                android:layout_marginLeft="5dp"
                android:layout_row="1"
                android:layout_weight="0.22"
                android:background="#000000"
                android:src="@drawable/boton_eliminar_domi" />

            <ImageButton
                android:id="@+id/buttonVer"
                android:layout_width="40dp"
                android:layout_height="40dp"
                android:layout_column="0"

android:layout_gravity="center_horizontal|fill_vertical"
                android:layout_marginLeft="5dp"
                android:layout_row="1"
                android:layout_rowSpan="2"
                android:background="#000000"
                android:src="@drawable/boton_ver" />

            <ImageButton
                android:id="@+id/botonVolver"
                android:layout_width="40dp"
                android:layout_height="40dp"
                android:layout_column="0"
                android:layout_gravity="right|top"
                android:layout_marginLeft="5dp"
                android:layout_row="1"
                android:background="#000000"
                android:src="@drawable/buttonvolver" />

        </LinearLayout>

    </GridLayout>

</LinearLayout>
```

3.2.2.6.3 Vista de pantalla Listado Productos

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:id = "@+id/layoutListadoHeladeras">

    <TextView
```

```
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_margin="3dp"
        android:gravity="center"
        android:text="@string/listado_heladeras_titulo"
        android:textColor="#FFFFFF"
        android:textSize="25dp" />

    <ListView
        android:id="@+id/HeladerasList"
        android:layout_width="fill_parent"
        android:layout_height="250dp"
        android:layout_weight="0.30"
        android:paddingTop="5dp" >
    </ListView>
</LinearLayout>
```

3.2.2.6.4 Vista de pantalla Configuracion

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <LinearLayout
        android:id="@+id/linearLayout1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentTop="true"
        android:layout_marginTop="85dp"
        android:orientation="vertical" >

        <ImageView
            android:id="@+id/imageView2"
            android:layout_width="match_parent"
            android:layout_height="1dp"
            android:background="#33b5e5" />

        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" >

            <ImageView
                android:id="@+id/imageView1"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginLeft="5dp"
                android:layout_marginTop="45dp"
                android:src="@drawable/semasforo_amarillo" />

            <TextView
                android:id="@+id/textViewAmarillo"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginLeft="5dp"
```

```
                android:layout_marginTop="58dp"
                android:text="@string/configuracion_texto_precaucion"
/>
        </LinearLayout>

        <SeekBar
            android:id="@+id/seekBarAmarillo"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:max="60"
            android:progress="1" />

        <LinearLayout
            android:id="@+id/linearLayout2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" >

            <ImageView
                android:id="@+id/ImageView02"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginLeft="5dp"
                android:layout_marginTop="45dp"
                android:src="@drawable/semasforo_rojo" />

            <TextView
                android:id="@+id/textViewRojo"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginLeft="5dp"
                android:layout_marginTop="58dp"
                android:text="@string/configuracion_texto_alerta" />
        </LinearLayout>

        <SeekBar
            android:id="@+id/seekBarRojo"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:max="60"
            android:progress="1" />

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="260dp"
            android:layout_marginTop="60dp" >

            <ImageButton
                android:id="@+id/imageButtonGuardarConfiguracion"
                android:layout_width="44dp"
                android:layout_height="40dp"
                android:src="@drawable/ic_boton_guardar" />
        </LinearLayout>
    </LinearLayout>

    <TextView
        android:id="@+id/textViewNombre"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
```

```
        android:layout_margin="3dp"
        android:gravity="center"
        android:text="@string/configuracion_titulo"
        android:textColor="#FFFFFF"
        android:textSize="25dp" />

    <LinearLayout
        android:id="@+id/linearLayout3"
        android:layout_width="250dp"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/textViewNombre" >

        <ImageButton
            android:id="@+id/imageButtonIngles"
            android:layout_width="30dp"
            android:layout_height="20dp"
            android:layout_marginLeft="5dp"
            android:layout_marginTop="55dp"
            android:src="@drawable/ic_idioma_ingles" />

        <ImageButton
            android:id="@+id/imageButtonEspanol"
            android:layout_width="30dp"
            android:layout_height="20dp"
            android:layout_marginLeft="10dp"
            android:layout_marginTop="55dp"
            android:src="@drawable/ic_idioma_espanol" />

        <ImageButton
            android:id="@+id/imageButtonPortugues"
            android:layout_width="30dp"
            android:layout_height="20dp"
            android:layout_marginLeft="8dp"
            android:layout_marginTop="55dp"
            android:src="@drawable/ic_idioma_portugues" />

    </LinearLayout>

</RelativeLayout>
```

3.2.2.6.6 Vista de filas para producto precargado

```
<?xml version="1.0" encoding="utf-8"?>
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/autoNombre"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:background="#000000"
    android:gravity="center_vertical"
    android:padding="5dp"
    android:textColor="#FFFFFF" />
```

3.2.2.6.7 Vista de filas para

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
```

```
    android:layout_height="wrap_content"
    android:background="@drawable/list_selector"
    android:orientation="horizontal"
    android:padding="5dip" >

    <LinearLayout android:id="@+id/thumbnailHeladeras"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:padding="3dip"
        android:layout_alignParentLeft="true"
        android:layout_marginRight="5dip">

        <ImageView
            android:id="@+id/heladeraImagen"
            android:layout_width="50dip"
            android:layout_height="50dip" />
    </LinearLayout>

    <TextView
        android:id="@+id/heladeraNombre"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignTop="@+id/thumbnailHeladeras"
        android:layout_toRightOf="@+id/thumbnailHeladeras"
        android:text=""
        android:textColor="#040404"
        android:typeface="sans"
        android:textSize="12dip"
        android:textStyle="bold"/>

    <TextView
        android:id="@+id/heladeraDetalle"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/heladeraNombre"
        android:layout_below="@id/heladeraNombre"
        android:layout_marginTop="1dip"
        android:textColor="#343434"
        android:textSize="10dip" />
</RelativeLayout>
```

3.3.3 Producto Resultante

3.3.3.1 Flujo Principal

3.3.3.1.1 Escaneo de código QR

Actores: Usuario.

Descripción: El usuario hace click en botón “Lectura QR” con lo que el sistema enciende la cámara del dispositivo, ya sea notebook, PC, celular o tablet y se prepara para la lectura de códigos QR para cuando el usuario la enfoque en su campo visual. Una vez leído el código QR el sistema automáticamente abrirá la aplicación y desplegara el listado de productos correspondiente a la heladera y usuario que correspondan al código QR.

Precondición:

- Código QR de más de 2cm cuadrados.
- Dispositivo con cámara.

Postcondición: Ninguna.

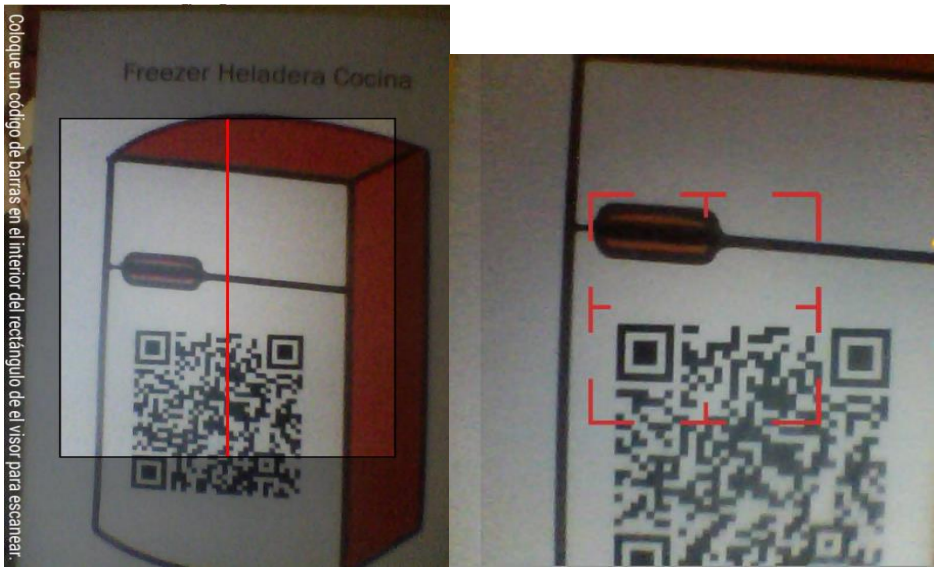


Figura 11: Escaneo de Código QR, La aplicación soporta el escaneo con cualquier aplicación, el único requisito es que el código QR sea mayor de dos centímetros cuadrados.

3.3.3.1.2 Pantalla listado de Productos

Actores: Usuario.

Descripción: El usuario da lectura de un código QR de más de 2cm cuadrados o selecciona la sección “Listado”. Ante estos estímulos el Sistema deberá listar el total de productos correspondientes al id de la Heladera leída en el QR, los productos se ordenaran por fecha de vencimiento y por cada uno se indicara su nombre, estado y días para su vencimiento.

Precondición: Lectura QR o Selección de sección “Listado”.
Postcondición: Ninguna.

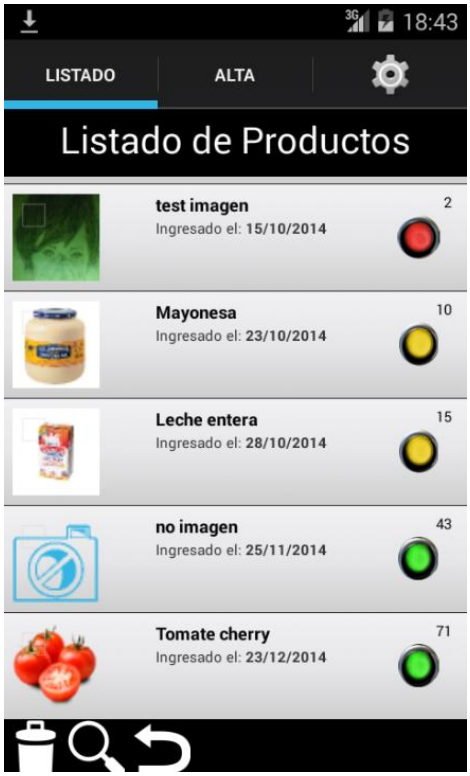


Figura 12: Listado de los productos de una heladera tras escaneo de código QR, se muestra Imagen del producto, descripción, fecha de vencimiento, estado y días faltantes para su vencimiento.

3.3.3.1.3 Pantalla Alta de Productos

Actores: Usuario.
Descripción: El Usuario ingresa los datos correspondientes para un alta de producto, los cuales son el nombre de producto (String, obligatorio), imagen (Blob, opcional) y fecha de vencimiento (Date, obligatorio), la cual se ingresará mediante un Calendar.

No se permitirán el ingreso de datos vacíos ni solo numéricos al igual que fechas de caducidad menores a la actual, en tales casos el sistema anunciará mediante mensajes el incumplimiento de estas características. En caso de que el producto sea ingresado el sistema mostrara un mensaje informando que el producto se ha dado de alta correctamente.

Precondición: Ubicación en Pestaña de Alta de producto.
Postcondición: Ninguna.

Figura 13: Alta de un producto, se ingresa el nombre de mismo, optativamente una imagen desde la memoria del dispositivo o utilizando la cámara y su fecha de vencimiento. Ingresando las primeras dos letras de un producto si este se ha ingresado alguna vez aparecerá en forma de listado para facilitar su carga, ídem para su imagen.

3.3.3.1.4 Pantalla Detalle y Modificación de Producto

Actores: Usuario.

Descripción: En la sección de “Listado de productos” al realizar click sobre uno de los productos el sistema le deberá dar aparición a una pantalla con los detalles del mismo, nombre, fecha de caducidad y su imagen, cualquiera de estos 3 datos podrá ser modificado por el usuario.

Precondición:

- Ubicación en Pestaña de Listado de producto.
- Al menos un producto cargado en el sistema.

Postcondición: Ninguna.

Figura 14: Modificación de un producto, se modifica optativamente el nombre de mismo, imagen desde la memoria del dispositivo o utilizando la cámara y su fecha de vencimiento.

3.3.3.1.5 Pantalla Eliminación de Producto

Actores: Usuario.

Descripción: El usuario da lectura de un código QR de más de 2cm cuadrados o selecciona la sección “Listado”. Ante estos estímulos el Sistema deberá listar el total de productos correspondientes al id de la Heladera leída en el QR, los productos se ordenaran por fecha de vencimiento y por cada uno se indicara su nombre, estado y días para su vencimiento. Junto a estos datos se añadirá un checkbox para cada producto listado el cual puede ser tildado por el Usuario y un botón eliminar debajo del listado total de productos para de esta manera dar opción de la eliminación múltiple de productos en el sistema. Al seleccionar uno o varios productos y presionar el botón eliminar el sistema deberá inmediatamente borrarlos de la lista de productos.

Precondición:

- Ubicación en Pestaña de Listado de producto.
- Al menos un producto cargado en el sistema.

Postcondición: Ninguna.

Figura 15: Se muestra la eliminación de dos productos a partir de seleccionarlos y luego realizar un click en botón de eliminar.

3.3.3.1.6 Pantalla Listado de Heladeras

Actores: Usuario.

Descripción: El usuario selecciona la sección “Listado”. Ante esta acción el Sistema deberá listar el total de heladeras correspondientes al id del usuario, las heladeras se ordenaran por orden alfabético y por cada una se indicara la cantidad de productos vencidos por heladera.

Precondición: Al menos una heladera dada de alta.
Postcondición: Ninguna.

Figura 16: En caso que el usuario en vez de escanear un código QR acceda a la aplicación mediante click en el icono de la misma en su dispositivo, lo que sucederá será que ingresara a la pantalla de listado de heladeras para de esa forma seleccionar una en particular.

3.3.3.1.7 Pantalla de Configuraciones

Actores: Usuario.

Descripción: El usuario selecciona la sección “Configuración”. Ante esta acción el Sistema deberá desplegar los componentes visuales que den opción al usuario de modificar el idioma de la aplicación, a ser estos Español, Inglés y Portugués, asimismo también se desplegaran los componentes visuales para que el cliente pueda optar por el nivel de alertas que desea obtener del sistema, pudiendo fijar la cantidad de días para considerar un producto en estado de precaución y la cantidad de días para considerar un producto en estado de alerta.

Precondición: Ninguna.
Postcondición: Ninguna.

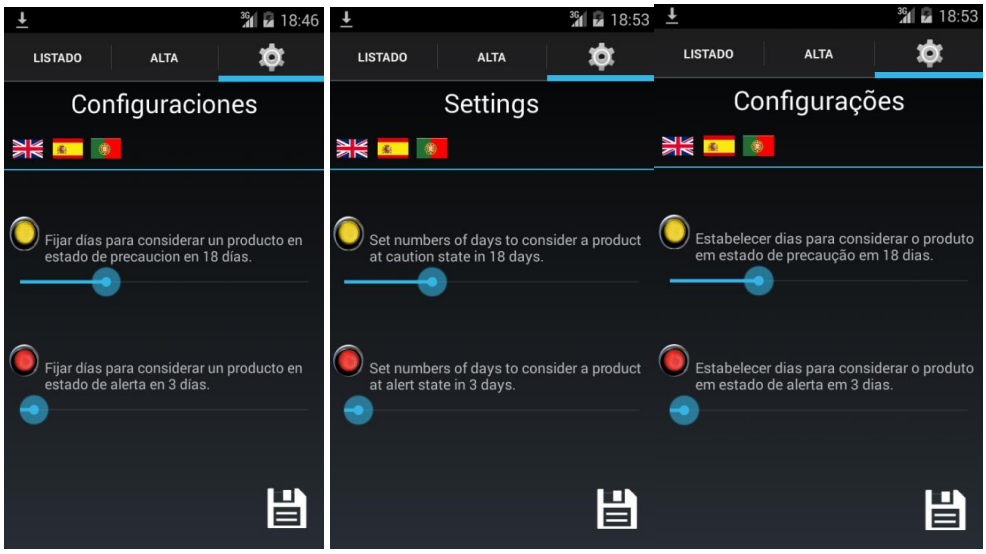


Figura 17: Pantalla de configuraciones donde se puede fijar el idioma ya sea Español, Ingles o Portugués, por defecto la aplicación tomara el configurado en el sistema operativo Android, también puede fijarse en esta pantalla los niveles de alerta y de prevención.

3.3.3.2 Flujos Alternativos

3.3.3.2.1 .Pantalla de Alta sin haber seleccionado Heladera

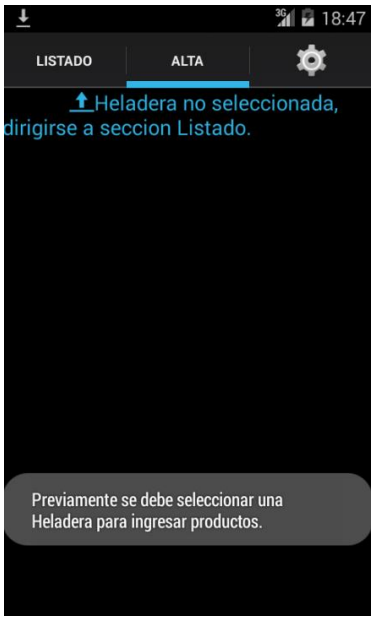


Figura 18: La aplicación muestra mensaje de error ya que previamente debe de indicarse sobre que heladera trabajar.

3.3.3.2.2 .Errores en Pantalla de Alta

Figura 19: Distintos errores en pantalla de alta por no haber ingresado el dato obligatorio de nombre y por haber ingresado una fecha menor a la actual como fecha de vencimiento de un producto.

3.3.3.2.3 .Error en Pantalla de Listado de producto

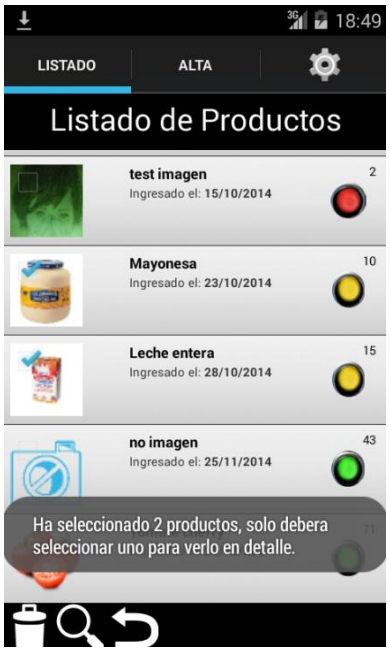


Figura 20: Para poder ver el detalle de un producto solo debe seleccionarse un único producto, si se selecciona más de uno o ningún producto y luego se realiza un click en el icono de detalle la aplicación disparara el mensaje correspondiente.

3.3.3.2.4 .Errores en Pantalla de Configuraciones

Figura 21: No es posible fijar la cantidad de días del nivel de precaución menor a la cantidad de días del nivel de alerta.

4 Conclusión

Se han tomado 7 sujetos de los cuales 2 han sido tomados como sujetos de control.

Fase 1: A todos los sujetos se les ha pedido a partir del 01/08/2014 que contabilicen los productos alimenticios que han desperdiciado hasta el día 15/08/2014.

Fase 2: El día 15/08/2014 se ha instalado el producto de gestión de alimentos resultante en este trabajo en los dispositivos móviles de los sujetos salvo a dos de ellos tomados como sujetos de control. A todos los sujetos se les ha pedido al igual que en la quincena anterior continuar contabilizando los productos alimenticios que desperdicien hasta el 30/08/2014.

De la recolección de datos se ha obtenido la siguiente tabla:

Sujeto/Experimento	Productos desperdiciados Fase 1	productos desperdiciados Fase 2
sujeto 1	9	0
sujeto 2	3	0
sujeto 3	1	0
sujeto 4	5	1
sujeto 5	3	1
sujeto 6 (control)	4	3
sujeto 7 (control)	1	3
Total	26	8

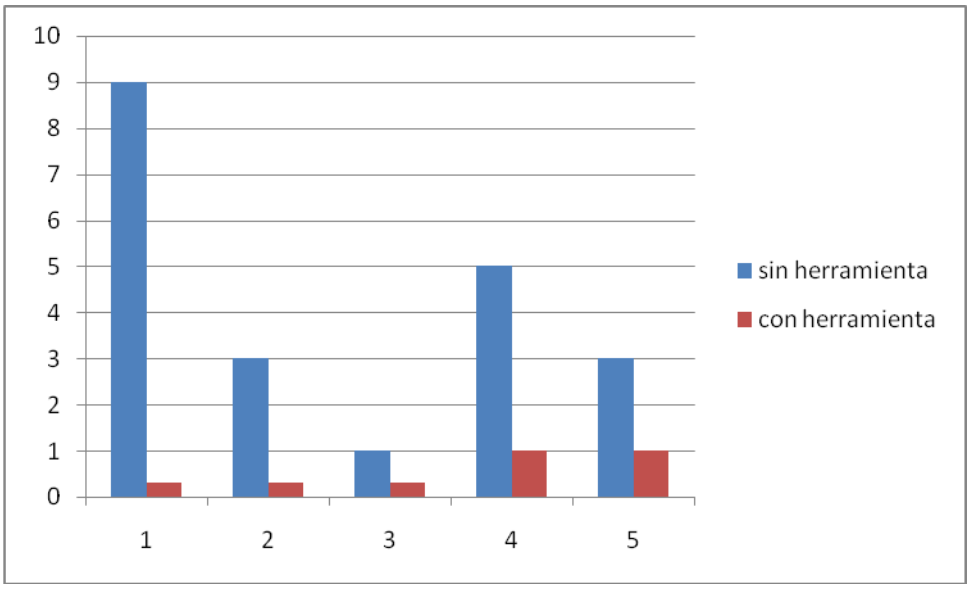


Figura 22: Se muestran gráficamente los valores dados sin y con la herramienta de gestión para evitar el desperdicio alimenticio para los sujetos 1, 2, 3, 4 y 5.

Dado los datos recolectados para su análisis añadimos las filas de subtotales para los sujetos a los que se les ha instalado en el dispositivo móvil la herramienta como a los de control, asimismo se añade una columna para identificar el porcentaje de ahorro en cada caso.

Sujeto/Experimento	Productos desperdiciados quincenalmente	productos desperdiciados quincenal aplicación	% ahorro
sujeto 1	9	0	**100%**
sujeto 2	3	0	**100%**
sujeto 3	1	0	**100%**
sujeto 4	5	1	**80%**
sujeto 5	3	1	**67%**
Subtotal	**21**	**2**	**89%**
sujeto 6 (control)	4	3	**25%**
sujeto 7 (control)	1	3	**-300%**
Subtotal (control)	**5**	**6**	**-17%**
Total	47	10	

Se desprenden de los datos recolectados las siguientes conclusiones:

- Los sujetos que han utilizado el producto de gestión de alimentos han reducido el desperdicio alimenticio en promedio un 89% a partir del momento en que se les ha instalado el producto de gestión desarrollado en este trabajo.

- Los sujetos de control quienes no han sido provistos de la herramienta de gestión han aumentado su desperdicio en promedio un 17% lo que se puede considerar como que se ha mantenido el mismo en el tiempo, lejos de reducirse cosa que notoriamente ha sucedido con el grupo de sujetos que han utilizado la herramienta de gestión.

Se concluye entonces que se ha cumplido el objetivo de este trabajo de demostrar que se puede reducir el desperdicio en hogares, hoteles. Comercios, etc. por el no consumo de productos alimenticios anteriormente a su fecha de caducidad mediante el desarrollo de un producto de gestión de alimentos que permite alertar al consumidor sobre los alimentos que se encuentran cercanos a su fecha de vencimiento a través de la utilización de tecnologías Android y códigos QR.

5 Futuras Líneas de Investigación

Como futuras líneas de investigación se proponen mejoras que pueden potenciar los resultados a la herramienta de gestión de alimentos desarrollada y a su vez darle mayor valor agregado:

- Ofrecer a los usuarios la posibilidad de cargar recetas y a su vez recibir mediante la utilización de mecanismos de inteligencia artificial la receta que más se adapte a los productos que el usuario tiene próximos a su vencimiento.

- Incluir la posibilidad de que el usuario ingrese el precio de cada producto para de esta manera lograr posteriormente reportes gráficos del gasto acumulado por heladera y gasto total.

- Incluir la posibilidad de que la herramienta genere listado de basándose en los productos vencidos o eliminados por el usuario por haberlos consumido.

- Añadir alertas hacia los usuarios localmente en el celular cuando un producto este próximo a su vencimiento, ya sea por medio de que el icono de la aplicación modifique su forma y/o color como así también hacer sonar o vibrar el dispositivo.

- Desarrollar la aplicación para ser multiplataforma es decir, incorporarla a iOS y a Windows Phone.

6 Fuentes de Información

6.1 Libros

- FAO (Organización de las Naciones Unidas para la Alimentación y la Agricultura; "Perdidas y desperdicios de alimentos en América latina y el Caribe", Julio 2014.

- Instituto sueco de alimentos y biotecnología encargado por la FAO ; "Global food losses and food waste" 2012.

- David Evans; "Food Waste: Home Consumption, Material Culture and Everyday Life" ED. Bloomsbury Academic. Diciembre 2014.

- Jonathan Bloom; "American Wasteland: How America Throws Away Nearly Half of Its Food", ED. Da Capo Lifelong Books, Agosto 2011.

- Tristram Stuart; "Waste: Uncovering the Global Food Scandal", ED. W. W. Norton & Company, Octubre 2009.

- Ángel Arias; "Aprende a Programar para Android", ED. Createspace, Febrero 2014.

- Sebastian Perochon; "Android. guía de desarrollo de aplicaciones para smartphones y tabletas", ED. ENI, Junio 2012.

- Frank Ableson, Robi Sen, Chris King; "Android. Guía para desarrolladores", ED. Anaya Multimedia, Mayo 2011.

- Jesús Tomás Gironés; "El Gran Libro de Android", ED. Marcombo, Febrero 2013.

- Joan Ribas Lequerica; "Desarrollo de aplicaciones para Android", ED. Anaya Multimedia, Enero 2013.

- Reto Meier; "Professional Android 4 Application Development", ED. Wrox, Mayo 2012.

- Aric Boyles;"The Complete Guide to QR Codes", ED. Mcgraw-Hill Osborne Media, Agosto 2012.

- Mick Winter; "Scan Me: Everybody's Guide to the Magical World of QR Codes", ED. Westsong Publishing, Abril 2011.

- Dr Kella B Price SPHR ; "QR Codes Made EZ: A Complete Guide to Creating and Implementing QR Codes", ED. CreateSpace Independent Publishing Platform, Enero 2014.

- Chris Branden ; "QR Codes: The Ultimate Guidebook", ED. Westsong Publishing, Abril 2011.

- Herbert Schildt; "Java: The Complete Reference"; ED. Mcgraw-Hill Osborne Media, Marzo 2014.

- Paul DuBois; "MySQL Cookbook", ED. O'Reilly Media, Enero 2007.

6.2 Páginas de Internet

http://www.fao.org/docrep/016/i2697s/i2697s.pdf
(2014)

http://www.fao.org/news/story/es/item/74327/
(2014)

http://www.oracle.com/technetwork/es/java/javase/downloads/jdk7-downloads-1880260.html
(2014)

http://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/indigosr2
(2014)

http://developer.android.com/sdk/index.html#download
(2014)

http://developer.android.com/sdk/installing/installing-adt.html
(2014)

http://www.androidpit.es/aplicacion/com.main.foodwastediary
(2012)

http://www.mtycic.org:8080/node/1019
(2012)

http://www.swissinfo.ch/spa/el-gran-desperdicio-alimentario/34255478
(2012)

http://www.qrcode.com/en/
(2014)

http://www.climate.com
(2014)

7 Anexos

7.1 Requerimientos Tecnológicos Mínimos Desarrollador.

7.1.1 Hardware.

- **RAM:** 1GB mínimo, 4GB recomendado.
- **Procesador:** 2.2ghz single core mínimo, core 2 Duo recomendado.
- **Espacio en disco**: 1GB mínimo, 3GB recomendado.

7.1.2 Software.

- **S.O.:** Windos, Linux, Mac OS, Solaris, todos en cualquiera de sus versiones 32 o 64 bits.
- **Java SDK 7:** http://www.oracle.com/technetwork/es/java/javase/downloads/jdk7-downloads-1880260.html
- **Entorno de desarrollo (IDE):** Eclipse IDE for Java Developers Indigo, http://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/indigosr2
- **Android SDK:** http://developer.android.com/sdk/index.html#download
- **Android Development Kit (ADT)**: http://developer.android.com/sdk/installing/installing-adt.html
- **Andoid Vitrual Device (ADV): Se** requiere crear una instancia para así simular un dispositivo celular.

7.2 Requerimientos Tecnológicos Mínimos Usuario.

7.2.1Hardware (Smartphone o Tablet).

- **RAM:** 512mb.
- **Procesador:** 1 Ghz.
- **Memoria Interna**: 512 Mb.

- **Pantalla**: 3,5''
- **Cámara: Resolución mínima de 1,3 Mpx.**
- **Acceso a Internet.**

7.2.2Software.

- Android 2.2 o superior.

Se requerirá también el código QR impreso con un tamaño mayor a 2 cm cuadrados.

7.3 Dispositivos soportados por la herramienta

A

- Alcatel One Touch 990
- Amazon Kindle Fire

B

- Nook
- Blu vivo 4.65
- Bq Aquaris 5
- Bq Aquaris E5 FHD

D

- Dell Mini 5

E

- Eken

F

- Fairphone

G

- Galaxy Nexus

H

- HTC Desire
- HTC Desire S
- HTC Dream
- HTC EVO 4G LTE
- HTC Hero
- HTC One
- HTC One (M8)
- HTC Sensation
- HTC Sensation XE
- HTC Wildfire
- Huawei Ascend D2
- Huawei Ascend G300
- Huawei Ascend G510
- Huawei Ascend G600
- Huawei Ascend Mate
- Huawei Ascend P1
- Huawei Ascend P2
- Huawei Ascend Y300
- Huawei G6600
- Huawei Honor
- Huawei U8110
- Huawei U8650

L

- LG G2
- LG G3
- Lg Optimus 4X HD
- LG Optimus G

M

- Moto 360
- Moto E
- Moto G
- Moto X
- Motorola Defy
- Motorola Defy Mini
- Motorola i1
- Motorola Milestone
- Motorola Xoom

N

- Nexus 10
- Nexus 4
- Nexus 5
- Nexus 7 (2012)
- Nexus 7 (2013)
- Nexus One
- Nexus Q
- Nexus S

O

- Ouya

S

- Samsung Galaxy
- Samsung Galaxy 3
- Samsung Galaxy Ace
- Samsung Galaxy Ace 2
- Samsung Galaxy Ace 3
- Samsung Galaxy Ace Plus
- Samsung Galaxy Camera
- Samsung Galaxy Gear
- Samsung Galaxy Mega 5.8
- Samsung Galaxy Note
- Samsung Galaxy Note 10.1
- Samsung Galaxy Note 3
- Samsung Galaxy Note II
- Samsung Galaxy Pocket
- Samsung Galaxy Pocket Plus
- Samsung Galaxy R
- Samsung Galaxy S Advance
- Samsung Galaxy S II
- Samsung Galaxy S II Plus
- Samsung Galaxy S III
- Samsung Galaxy S III Mini
- Samsung Galaxy S Plus
- Samsung Galaxy S4
- Samsung Galaxy S4 Mini
- Samsung Galaxy S4 Zoom
- Samsung Galaxy S5
- Samsung Galaxy S5 Mini
- Samsung Galaxy Spica
- Samsung Galaxy Tab
- Samsung Galaxy Tab 3
- Samsung Galaxy Tab 3 10.1

- Samsung Galaxy Trend
- Samsung i5500
- Samsung Galaxy S
- Sony Ericsson Live with Walkman
- Sony Ericsson Xperia Play
- Sony Ericsson Xperia X10 Mini
- Sony SmartWatch
- Sony Xperia E
- Sony Xperia E1
- Sony Xperia J
- Sony Xperia L
- Sony Xperia M
- Sony Xperia P
- Sony Xperia S
- Sony Xperia sola
- Sony Xperia SP
- Sony Xperia T
- Sony Xperia tipo
- Sony Xperia U
- Sony Xperia V
- Sony Xperia Z
- Sony Xperia ZL

7.4 Anexo CD

Se anexa CD que contiene la herramienta resultante.