UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

Grado en Ingeniería Informática

Aplicación móvil para el inventario de una despensa

Mobile application for pantry inventory

Realizado por
Juan Pérez de la Vega

Tutorizado por
David Santo Orcero

Departamento
Lenguajes y Ciencias de la Computación

MÁLAGA, Septiembre 2020

Resumen

Ir a hacer la compra es una tarea semanal, o incluso casi diaria, que todos hacemos o haremos. La funcionalidad de esta aplicación permite hacer las compras necesarias mucho más fácil, ya sea en una familia o en un piso compartido entre varias personas, logrando saber en todo momento que tenemos en casa, pudiendo así ahorrar gastos en compras innecesarias.

El proyecto se basa en una aplicación desarrollada en Android Studio, centrada en crear una despensa inteligente para almacenar en tiempo real los productos comprados por el usuario. Con funcionalidades tales como: lectura del código de barras para obtener el nombre del producto, organización por grupos, y la gestión y generación automática de la lista de la compra.

Palabras claves: inventario, código de barras, lista de la compra, aplicación, simple y despensa.

Abstract

Do weekly or even daily shopping is a task that all of us have to do or will do. The functionality of this application allows making the necessary purchases much easier, either in a family or in a shared flat between several people, making it possible to know at all times that we have at home, thus being able to save expenses on unnecessary purchases.

The project is based on an application developed in Android Studio, focused on creating an intelligent pantry to store the products purchased by the user in real time. With functionalities such as: barcode reading to obtain the product name, group organization, and automatic management and generation of the shopping list.

Keywords: inventory, barcode, shopping list, application, simple and pantry.

Índice

Índice de figuras

Índice de Tablas

Capítulo 1

Introducción

Contenido

1.1. Motivación

Ir a hacer la compra es una tarea semanal, o incluso casi diaria, que todos hacemos o haremos. La funcionalidad de esta aplicación permite hacer las compras necesarias mucho más fácil, ya sea en una familia o en un piso compartido entre varias personas, logrando saber en todo momento que tenemos en casa, pudiendo así ahorrar gastos en compras innecesarias.

1.2. Objetivo

La idea principal de la aplicación es mantener a disposición del usuario, en todo momento y cualquier lugar, la lista de todos los productos que él posea en su casa. Con ello se pretende lograr:

1. **Un mayor control de todos los productos que el usuario quiera inventariar**. El usuario no solo puede inventariar productos alimenticios o de limpieza, si no lo que el desee.

2. **Ahorro de tiempo a la hora de hacer la lista de la compra**. La propia aplicación posee una funcionalidad que agrega automáticamente los productos que son necesarios comprar, que previamente el usuario habrá indicado cuál es el número mínimo de unidades que puede haber de ese objeto en la despensa.

3. **Ahorro de dinero**. Uno de los consejos más comunes al ir a hacer la compra es llevar una lista de los productos necesarios, así evitamos la compra de artículos innecesarios.

Además, se ha tenido en cuenta la accesibilidad de la aplicación, en la que se prioriza un diseño simple para que cualquier persona pueda usarla de forma rápida y no tenga que invertir apenas tiempo en descubrir el funcionamiento de esta.

Otra del las piezas fundamentales de la aplicación es el uso de un lector de código de barra que, en principio, utiliza la propia cámara del móvil para poder añadir, nuevos artículos o ya existentes en la despensa, o eliminarlos de ella porque ya han sido utilizados y tirados a la basura.

1.3. Estructura del documento

1. **Capítulo 1: Introducción.** El presente capítulo es una introducción a la memoria, como su propio nombre indica. De tal forma que se explica en que se basará nuestro proyecto y cuál es el objetivo final al que queremos llegar.

2. **Capítulo 2: Estado del arte.** Se explicará en que consiste el estado del arte, y cual es la base de nuestra aplicación. Posteriormente se hará un estudio del mercado con las aplicaciones competidoras en nuestro campo.

3. **Capítulo 3: Requisitos de la aplicación.** Antes de empezar el desarrollo será necesario establecer que funcionalidades debe hacer nuestra aplicación. En este capítulo se enumerarán estos requisitos y se dará una breve descripción de estos.

4. **Capítulo 4: Herramientas del desarrollo.** Durante el desarrollo se usaran numerosas herramientas, ya sean para el diseño o para la implementación de la aplicación, así como para la redacción de la propia memoria. Aquí se explicará cuáles herramientas se han usado y por que.

5. **Capítulo 5: Diseño.** Establecidos los requisitos para el funcionamiento de la aplicación, es necesario profundizar un poco más cómo va a funcionar

la aplicación, por ello este capítulo dará una visión más gráfica, mostrando métodos y objetos, para comprender mejor la posterior implementación.

6. **Capítulo 6: Conclusión y líneas futuras.** Para terminar la memoria haremos una valoración del trabajo realizado y valoraremos posibles mejoras e ideas para sacar más partido de nuestra aplicación.

7. **Apéndice: Manual del usuario.** Finalmente, se mostrará como es el resultado de la implementación de la aplicación y se enseñará al usuario como ha de usar sus funcionalidades.

Capítulo 2

Estado del arte

Contenido

2.1. ¿Qué es el estado del arte?

En este capítulo de la memoria se aborda un punto bastante común en las tesis, el estado del arte. Antes de comenzar a desarrollar este capítulo de la memoria expliquemos brevemente de que se trata.

"El estado del arte es una modalidad de la investigación documental que permite el estudio del conocimiento acumulado (escrito en textos) dentro de un área específica. "[1] El estado del arte intenta ofrecer al investigador un punto de partida para que este comience el análisis de otros trabajos en los que se ha tratado, ya sea en menor o mayor medida, el punto que el investigador tratará en su trabajo.

En el ámbito tecnológico e industrial, más en concreto en el desarrollo de aplicaciones, la utilidad de este punto para el TFG es realizar una especie de estudio de mercado, con el que se analizan aplicaciones con características similares a la que será desarrollada, permitiendo así conocer algunos de los requisitos indispensables que necesita la aplicación. Aunque este punto en gran parte se centra

en el análisis que acabamos de explicar, también consta de otra serie de estudios y de información en los que se cimienta la aplicación que veremos a continuación.

2.2. Ahorro de tiempo y dinero

El tiempo y el dinero son dos recursos, inmaterial y material respectivamente, que rigen la vida de cualquier persona en el mundo. Estos se complementan entre si, necesitamos tiempo para poder invertir el dinero y necesitamos dinero para poder invertir el tiempo en cierta medida, sin dinero limitamos nuestras posibilidades de invertir ese tiempo.

Es por esto que en algo tan banal como ir a hacer la compra de la semana o del mes, muchas familias tienen que mirar meticulosamente el dinero que gastan, por ejemplo solo en España hay alrededor de 12 millones de personas que se encuentran en riesgo de pobreza o de exclusión social y alrededor del 48 % de la población llega con dificultades a fin de mes, por lo que han de pensar muy bien antes de llenar el carro de la compra.[2]

Una búsqueda en *Google* con cualquiera de las las siguientes frases: *"¿Cómo ahorrar dinero en la cesta de la compra?"*, *"Trucos para llegar a fin de mes"* o *"customer saving money weekly groceries list"* desemboca en infinidad de resultados, estudios y consejos para poder ahorrar dinero a la hora de ir a comprar. Entre las decenas de ellos encontramos algunos en los que se basan la aplicación.

« Do a pantry sweep» o traducido al español, hacer un barrido a la despensa. Básicamente se basa en hacer un inventario de lo que tenemos en la despensa, lo consideraríamos como el paso previo antes de ir a comprar al supermercado, haciendo esto se reduce el riesgo de comprar productos que ya tengamos o que no podamos comer antes de que caduquen, minimizando así el desperdiciar comida y asegurando que todo lo que se compra será utilizado, es decir ganamos mucho control y conocimiento, y evitamos las compras de *"por si acaso"*.[3]

Hacer una lista de la compra es otra de las acciones más recurridas, gracias a esto podemos lograr reducir el impulso a comprar productos innecesarios, planear las comidas de la semana. Además, anotar que se necesita en específico puede abaratar muchos los gastos.[4]

Estos dos consejos son la pieza clave a la hora de tener en cuenta el desarrollo de la aplicación ya que la posibilidad de ahorrar dinero es el motivo principal, pero hay un segundo factor como es el tiempo en el que también podemos salir beneficiados.

En el caso del tiempo sirven los mismos consejos previamente dados, hacer

una lista facilita la tarea de la compra consiguiendo saber que se necesita comprar, yendo directamente a por esos productos y evitando deambular por el supermercado, logrando así ceñirse a lo necesario y estando en el cajero lo antes posible.

En general la idea es ofrecer orden y organización a quien lo necesite consiguiendo beneficios en dos aspectos tan importantes hoy en día como son el dinero y el tiempo.

2.3. Aplicaciones Android

La búsqueda de cualquier tipo de *App* para facilitar las tareas del día a día se ha convertido en una acción muy recurrente, y desde luego la compra semanal o mensual no iba a ser menos.

La idea de modernizar la típica lista de la compra a papel y lápiz no es algo novedoso, de hecho podemos encontrar en las plataformas de descargas de *Apps* más famosas, como *Google Play* y *Apple Store*, decenas de aplicaciones que pueden realizar funciones similares a las que se están exponiendo en este TFG. Ya no solo centrado en el ámbito doméstico, si no también en el apartado empresarial para la gestión de inventarios, por ejemplo para las pequeñas o medianas *PYMES*.

Es por ello que en este apartado trataremos de analizar una serie de aplicaciones para dispositivos móviles, que realizan las funciones que se han comentado en los objetivos del primer capítulo, destacando los puntos fuertes y donde flaquean estas aplicaciones. Para la elección de estas no se ha tenido en cuenta la valoración que tienen en sus respectivas tiendas *online*, en una escala del 0 al 5 para ambas.

2.3.1. Aplicaciones para inventariar

Se analizaran aplicaciones que podemos encontrar de forma gratuita en *Google Play*, teniendo como característica indispensable tener la capacidad de realizar un inventario, por lo que se analizarán aplicaciones que posean otras características aunque tengan una visión más empresarial y no solo de carácter doméstico.

Stock and inventory simple 4.6/5

Analicemos la primera de todas, nos encontramos con una de las aplicaciones de mayor puntuación de la tienda con la funcionalidad que buscamos, con un carácter muy empresarial y destinada sobre todo a los autónomos que gestionen

el inventario de una pequeña o mediana empresa.

Posee un diseño amigable a la vista, la idea de tener un menú principal bien organizado con las opciones que nos permite hacer la aplicación en recuadros acompañados de dibujos, como se puede apreciar en la Figura 5.13, lo hace mucho más entendible para el usuario. Aun así, la complejidad de la aplicación es demasiado elevada, no lo es quizás para el uso de una gestión del stock de una tienda ya que tiene funcionalidades como la de importar proveedores así como clientes, sin duda es demasiado compleja para la idea principal del TFG, conseguir que sea sencilla y que cualquier persona sin apenas conocimiento de este tipo de aplicación pueda utilizarla.

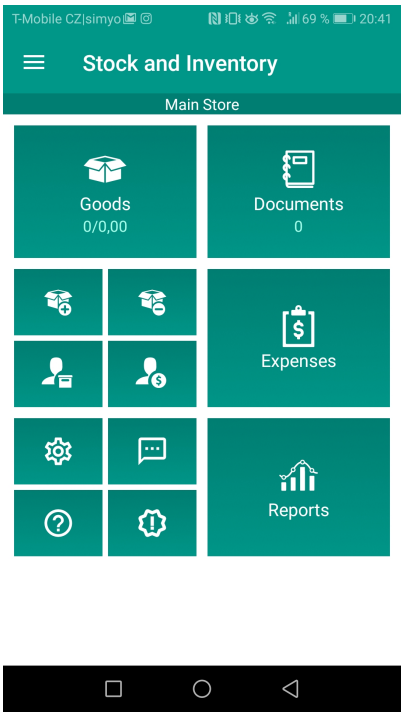


Figura 2.1: Menú principal de la aplicación 1

Empecemos a hablar de la forma de añadir un producto, otro de los puntos fundamentales que nos interesa.

Como vemos en la Figura 2.2 excepto la lectura del código de barra, la introducción de todos los demás parámetros, necesarios para guardar el producto en el inventario de la aplicación, han de ser escritos de forma manual. Esto requiere demasiado tiempo por parte del usuario para agregar un solo producto, es decir, se ha de buscar una forma más rápida y sencilla para agregar el artículo para utilizarlo en nuestro proyecto, sin duda es algo mejorable en esta aplicación.

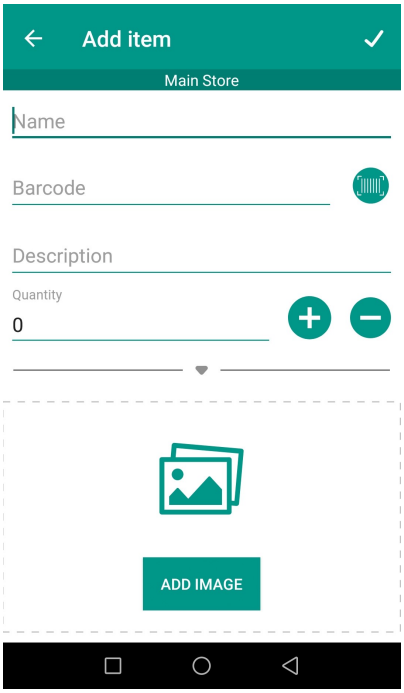


Figura 2.2: Añadir producto de la aplicación 1

Una vez introducido el artículo en el inventario, con los respectivos parámetros que hemos visto previamente, nos encontramos con dos que son los más destacables: uno indispensable, la cantidad de unidades que tenemos de ese producto, sin esto apenas tendría sentido la aplicación, y la cantidad mínima de productos que puede haber en el inventario. Hablemos de este último, esta aplicación informa, como se muestra en la Figura 2.3, que se ha superado el límite mínimo resaltando con un color rojo el número de unidades, sin embargo no te notifica con un mensaje ni una notificación al teléfono móvil.

Por otra parte, la aplicación tiene una forma un poco rudimentaria de aumentar y disminuir las unidades del producto que ya tenemos en el inventario, nos obliga a entrar al propio producto y desde los detalles del producto, eliminar las unidades deseadas. Claramente se podría recortar este paso si en el propio menú del inventario nos da la opción de aumentar y disminuir las unidades.
Aun así hay que resaltar algunas funcionalidades de la aplicación con casi sin nada que objetar como son: La idea de tener un tutorial dinámico que te enseña como funciona la aplicación conforme se accede a los diferentes apartados del menú, también el tener un menú lateral desplegable hace mucho más rápida la navegación y por último, una de las funcionalidades más útiles, la aplicación per-

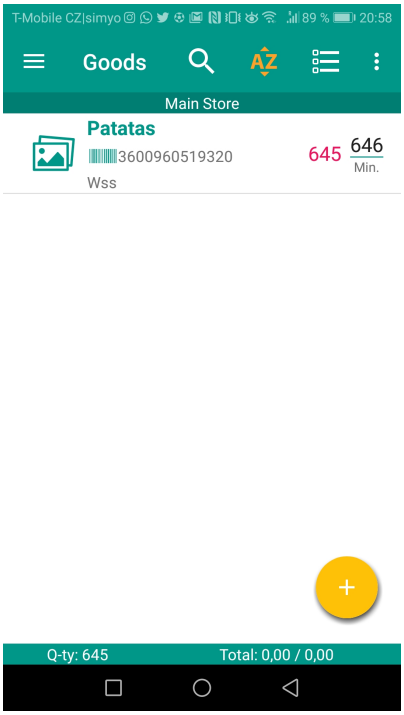


Figura 2.3: Existencias por debajo del limite de la aplicación 1

mite usar otro tipo de dispositivo, y no solo la cámara del teléfono móvil, para la lectura del código de barras. Sin duda esto es algo muy útil de cara a mejorar la aplicación y ahorrar tiempo para agregar los artículos al inventario si se usa un lector de código de barras.

Mobile inventory 4.5/5

Esta segunda aplicación es quizás de las que flaquea más en todos los aspectos. En principio no ofrece ninguna novedad, ni ninguna otra funcionalidad que cualquier aplicación de inventariar no posea, cumpliendo todas las necesidades básicas: agregar y eliminar artículos, lectura con código de barras, una lista con todos los productos que tenemos y la posibilidad de tener varios inventarios. Los inconvenientes vienen en la ejecución de estas funcionalidades.

Como se aprecia en la Figura 2.4, la aplicación presenta un diseño demasiado simple y tosco, los dos botones inferiores permiten agregar productos de una forma sencilla, de hecho la funcionalidad de la lectura del código de barras que podemos ver en la esquina inferior izquierda es bastante útil y hace muy cómodo añadir productos, primero se hace la lectura y después se escriben los parámetros del producto.

Otra vez el mismo error, al igual que en la primera aplicación analizada, a la hora de eliminar productos es necesario entrar al propio artículo y disminuirlo desde ahí, sin embargo como ya hemos comentado con anterioridad sería mas cómodo y rápido que en la propia lista del inventario se pudiese disminuir y aumentar.

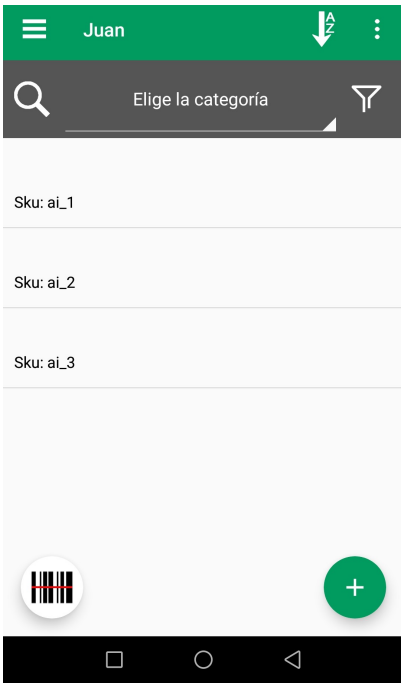


Figura 2.4: Menú principal de la aplicación 2

El historial de compra es la única novedad que presenta esta aplicación frente a la analizada anteriormente. Saber los movimientos de entrada y salida de productos puede ser algo interesante de cara a nuestra aplicación para conocer cuando se han comprado algunos productos, teniendo así en cuenta si hace mucho que se adquirieron y por lo tanto puede que estén caducados si estamos hablando de un producto fresco o sin fecha de caducidad. Sin embargo esta funcionalidad tiene mucho más utilidad en la aplicación que estamos analizando, porque al igual que la anterior posee un carácter más empresarial y está destinado para inventariar una pequeña o mediana tienda.

Sistema de inventario inteligente 4.5/5

Después de haber analizado aplicaciones, las cuales han tenido un carácter más para la gestión de una tienda, centrémonos en un ámbito más doméstico y por lo tanto en un competidor del mercado mucho más cercano. Aunque la característica principal la seguimos manteniendo, el poder inventariar, presenta una

funcionalidad muy útil: la posibilidad de crear grupos o etiquetas, expliquemos en que consiste esta opción. Crear un grupo o una etiqueta permite clasificar los productos, como se puede ver en la Figura 2.7, de una forma sencilla y además, a largo plazo cuando tengamos una gran cantidad de productos en el inventario facilita la búsqueda de estos, incluso se da la posibilidad de importar y exportar estos grupos. Por otra parte para agregar productos posee la forma más sencilla de hacerlo, lee un código de barras de un producto, si este no está ya agregado previamente a nuestro inventario la aplicación nos pregunta como queremos agregarlo, si como artículo, grupo o etiqueta como vemos en esta imagen.

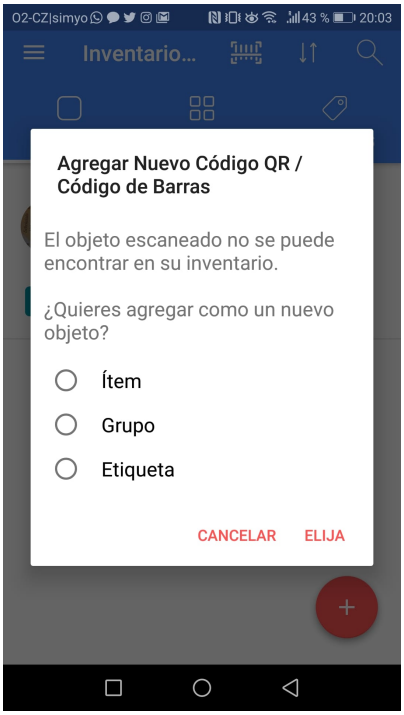


Figura 2.5: Ventana emergente al agregar producto de la aplicación 3

Sin embargo si el producto ya está en el inventario se nos abre automáticamente la información con todos los atributos y por lo tanto, podemos modificar los valores de estos campos como queramos, como se aprecia en la Figura 2.6

En lo que respecta a los atributos no hay ninguna novedad en los que aparecen de forma predeterminada, similares a los vistos en las anteriores aplicaciones, sin embargo esta aplicación ofrece la posibilidad, en el menú desplegable, de agregar los campos que deseemos a los artículos, grupos y etiquetas solamente a los existentes, a los nuevos o a ambos.Incluso, a un artículo en concreto.

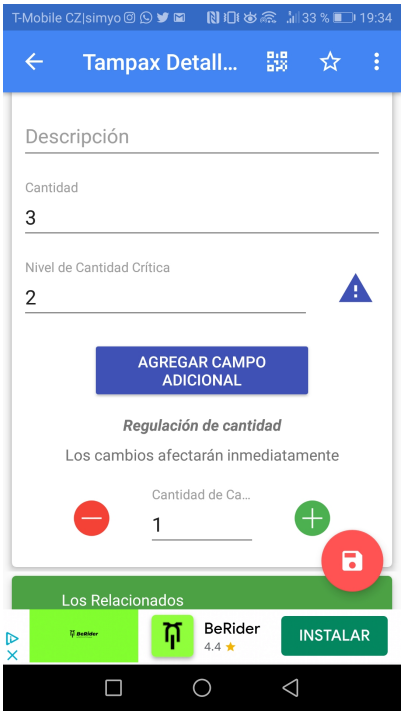


Figura 2.6: Agregar producto de la aplicación 3

Por si fuera poco esta aplicación ofrece una última innovación, un modo exploración con el que, gracias a la lectura de código de barras, podemos agregar o eliminar unidades del producto sin necesidad de tener que buscarlo en el inventario, haciendo mucho más rápidas y amenas esas tareas. En general estas 3 novedades, con respecto a las analizadas hasta ahora, la hacen una aplicación muy útil, muy sencilla y sin duda, la aplicación más referente hasta ahora.

No obstante, no todo es perfecto y esta aplicación sí que presenta alguna desventaja. Aunque podemos tener una cuenta asignada para tener disponible otros nuevos beneficios que nos ofrece, es necesario ser usuario *premium*, es decir la aplicación no es completamente gratuita e incluso nos vemos sorprendidos a veces con ventanas emergentes de publicidad. Siendo un poco más exigente, el modo exploración es un poco confuso y su diseño podría ser mejorado, así como la lectura del código es un poco lenta y no se puede activar la linterna del teléfono móvil para verlo con más nitidez, por lo que la lectura es más lenta, dificultando además la tarea en lugares con escasa luz. Y para terminar carece de lista de la compra, es decir para ver que es lo que falta en la despensa se obliga al usuario buscar en la lista del inventario cuáles son los productos que tienen pocas unidades, haciendo así más lenta la tarea de ir a hacer la compra, que supuestamente la aplicación trata de mejorarla y hacerlo más rápido.

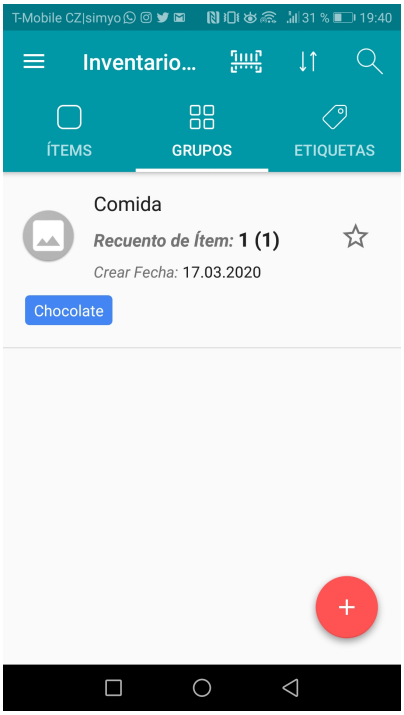


Figura 2.7: Grupos de la aplicación 3

Pantrify 3.4/5

Se nos presenta el caso más curioso de todas las aplicaciones que serán analizadas en este capítulo, la aplicación que tiene la valoración más baja de la tienda y sin embargo la más completa. Cuando hemos analizados las otras aplicaciones casi todas carecían de algo, si tenían un buen inventario necesitaban una lista de la compra o viceversa. Otras no estaban destinadas a un ambiente doméstico o simplemente su uso era demasiado complejo. Sin embargo esta aplicación logra unificar todas las funcionalidades que necesitamos.

En primer lugar al abrir la aplicación se nos presenta un tutorial bastante completo que nos explica el funcionamiento de todas las opciones que tiene aplicación. Todas estas opciones se muestran, en el menú principal, en la parte inferior de la pantalla como se puede ver en la Figura 2.8, y además posee un menú lateral desplegable bastante simple e intuitivo. Como hemos dicho con anterioridad, posee todas las funciones que se necesitan, la posibilidad de inventariar y de agregar los productos con la lectura del código de barras. Cabe destacar en la lectura del código que se realiza de forma muy rápida y además, presenta algo que ninguna de las otra aplicaciones tiene, no solo lee el código de barras y lo rellena

en el campo del producto si no que además en algunos casos rellena el campo del nombre aunque ese producto sea nuevo. Esto quiere decir que busca en una base de datos externa ese código de barras y si lo encuentra rellena el parámetro del nombre automáticamente, logrando, en caso de que el producto esté en esa base de datos externa, un ahorro de tiempo para agregarlo al inventario. Esta es una de las mejoras más importantes de todas las aplicaciones analizadas hasta ahora, consiguiendo que el usuario pierda el menor tiempo posible en rellenar parámetros del producto, pero no la única.

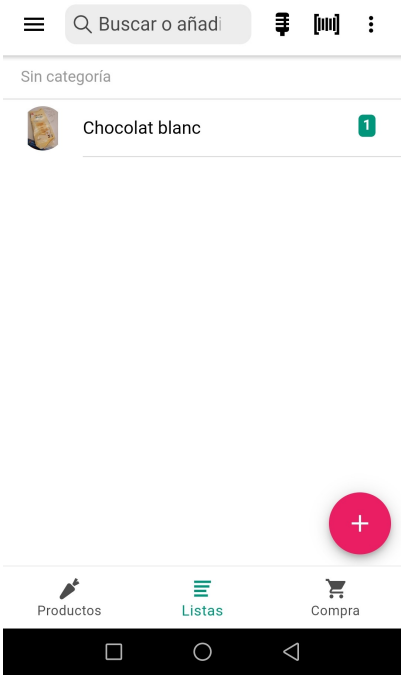


Figura 2.8: Menú de la aplicación 4

La lista de la compra no es ninguna novedad, ya lo hemos visto anteriormente aunque, esta nos ofrece la posibilidad de rellenarla de forma automática gracias al atributo del producto que nos dice el limite mínimo de unidades que puede haber en el inventario, de esta forma ahorramos mucho tiempo ya que no es necesario hacer la lista de la compra manualmente, ni siquiera tenemos que perder tiempo en buscar los productos que menos unidades quedan en el inventario, un botón y ya tenemos la lista de la compra. La segunda gran mejora con respecto a las anteriores,que además tiene la posibilidad de tener varias listas de la compra.

En lo que respecta a los atributos, no se pueden añadir personalizados, simplemente están los predeterminados. Pero si que se puede cambiar el tipo de cantidad del producto ya sea por ejemplo en unidades, botellas, cajas, kilogramos,

litros, latas, etc...

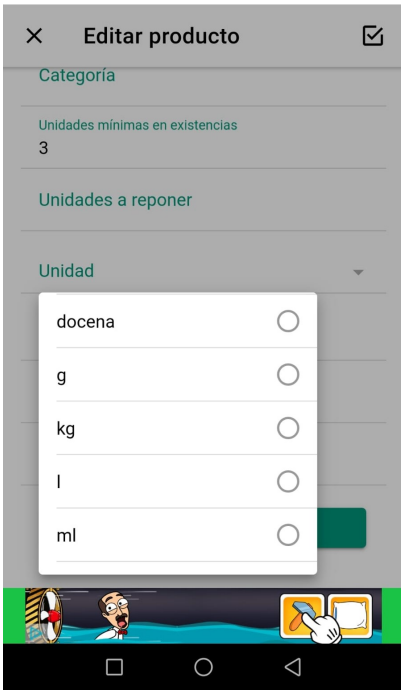


Figura 2.9: Tipo de unidades de la aplicación 4

Obviamente la aplicación no es perfecta, tiene funcionalidades muy interesantes que desde luego ahorran mucho tiempo al usuario, ideas en la que nos centraremos a la hora del desarrollo de nuestra aplicación ya que la optimización del tiempo es la idea principal en la que nos centraremos. Aun así hay varias cosas mejorables. De la forma de agregar un producto nuevo ya hemos hablado de ella, pero en caso de que este producto ya existiese en nuestro inventario, es necesario aumentar las unidades de este de forma manual, es decir, no podemos usar la lectura de código de barra para aumentar o decrementar ese producto por lo que se tiene que hacer de forma manual. No obstante esto no es lo que le otorga la peor nota de todas las aplicaciones analizadas, es necesario entrar en los comentarios de la tienda para conocer uno de los errores más graves que se ha encontrado entre todas. Actualizar la aplicación elimina los datos del inventario, por lo que borra nuestra despensa, es un error que no se puede permitir, aunque sin duda no es algo que una actualización futura no pueda arreglar.

Y para terminar, pero no menos importante, es la idea de colaboración. La aplicación permite compartir el inventario por medio de varias redes sociales, con la intención de que una familia o personas que compartan un piso y tengan la aplicación descargada, tengan el mismo inventario logrando así una mejor or-

ganización a la hora de hacer la compra y conocer que hay en casa.

Fooder 4/5

Esta será la última aplicación que analicemos, bastante similar a las anteriores debido a que es complicado encontrar nuevas funcionalidades que sean útiles para inventariar artículos o hacer listas de la compra. Aun así esta aplicación presenta alguna diferencia con respecto a las anteriores. Esta posee una serie de productos predeterminados típicos, por ejemplo, pasta, agua, aceite, arroz, leche, etc... Por lo que esto cambia en parte la metodología a la hora de agregar productos, ya no se agrega el producto a nuestro inventario, lo agregamos a la propia base de datos de productos de la aplicación y posteriormente a nuestro inventario. Aunque pueda parecer más intuitivo en primera instancia, al fin y al cabo en las anteriores aplicaciones logramos la misma finalidad con un coste inferior de tiempo, haciendo este método un poco redundante.

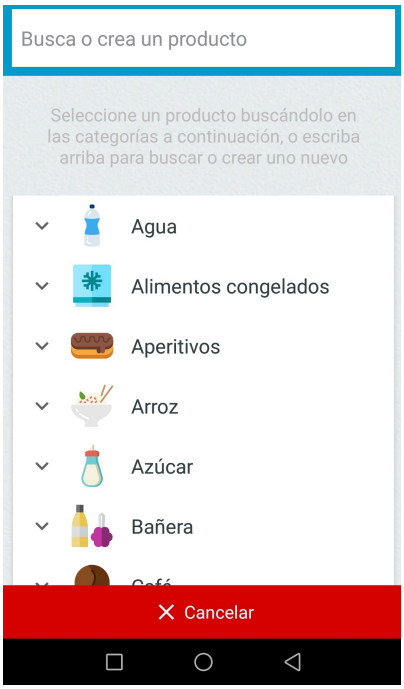


Figura 2.10: Metodología para agregar productos de la aplicación 5

Como ya hemos visto en las aplicaciones anteriores, posee las funcionalidades típicas, agregar un artículo con una serie de atributos predeterminados que en este caso no pueden ser ampliados por los que el usuario desee, una lista de la compra, un tutorial, un menú lateral desplegable, compartir la lista de la compra... Funcionalidades básicas que obligatoriamente tiene que tener nuestra apli-

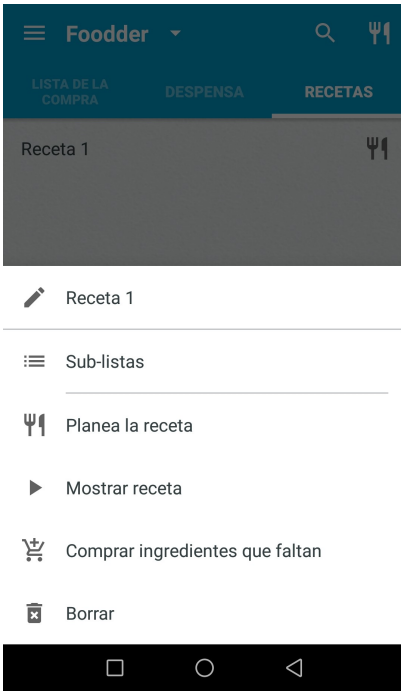


Figura 2.11: Menú desplegable en la receta de la aplicación 5

cación y cualquier aplicación destinada a a tarea que estamos analizando.

Aun así, la aplicación tiene otra funcionalidad novedosa que no hemos visto con anterioridad, la posibilidad de hacer recetas. Claramente esto difiere de las funcionalidades básicas que ha de tener pero es algo destacable que analizar. Con esta opción podemos crear nuestras propias recetas introduciendo los ingredientes que son necesarios para ser cocinadas, a parte en un menú desplegable, como vemos en la Figura 2.11, podemos ver cuáles son los ingredientes que tiene esta receta y además, con otra opción tenemos la posibilidad de introducir en la lista de la compra los ingredientes que faltan en nuestro inventario para hacer la receta.

En general es una aplicación que cumple con todos los requisitos básicos que se le puede pedir, aunque carece de lectura de código de barras por lo que la búsqueda de artículos ha de ser manual y se hace tediosa, esto es una desventaja que no se puede permitir en este tipo de aplicaciones ya que lo más importante es el ahorro del tiempo y una tarea tan recurrente en una aplicación para inventariar como puede ser buscar un producto para saber si este está no se puede permitir que se tenga que hacer manualmente.

2.3.2. Aplicaciones para lista de la compra

Ya hemos visto aplicaciones en el punto anterior que contienen la funcionalidad de hacer una lista de la compra a partir de un inventario, sin embargo, y aunque la idea principal de nuestra aplicación siga siendo conseguir una despensa en nuestro teléfono móvil, la posibilidad de tener una lista de la compra que se cree automáticamente es muy útil de cara a facilitar la tarea de hacer la compra semanal o mensual al usuario.
La lista de la compra no es ninguna novedad llegados a este punto, ya se ha analizado anteriormente, pero las otras aplicaciones no estaban centradas únicamente a esta función, por lo que puede que haya mejoras que no se han tenido en cuenta y que se podrá ver en las siguientes aplicaciones que procedemos a analizar.

En este caso para la selección de las aplicaciones hemos tomado las mejores, es decir las que cuenta con el mayor número de descargas y la mayor valoración en la tienda, ya que en este punto solo queremos analizar que tienen las mejores para ver si existen algunas mejoras de las que tomar nota.

Listonic **4.7/5**

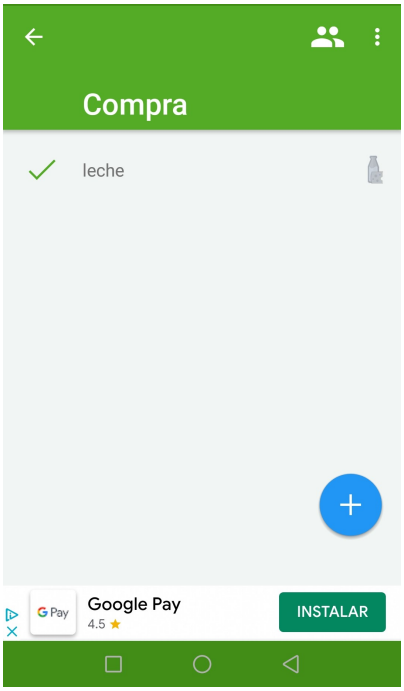


Figura 2.12: Ejemplo de objeto comprado en la lista de la aplicación 6

Nos encontramos ante una aplicación muy simple, como llevamos defendien-

do hasta ahora es una cualidad muy importante de cara al desarrollo de nuestra aplicación, aunque esta aplicación peca de ser demasiado simple. Apenas sin funcionalidades, la aplicación posee un menú principal en el que se puede crear una lista de la compra, llamándola como deseemos, permitiendo así poder crear más de una. La idea de poder crear varias listas de la compra puede ser interesante si lo que buscamos es organización pero pensando en nuestra aplicación, sería mucho más interesante si en la propia lista de la compra pudiésemos clasificar los productos por los grupos o etiquetas de los que ya hemos hablado anteriormente.

Dentro de la propia lista de la compra encontramos los productos, y un botón inferior para agregarlos. La forma de agregar los productos nos es familiar, un catálogo de productos predeterminado como en *Fooder* y la posibilidad de agregar un producto que no encontremos de forma manual. Obviamente no tenemos la posibilidad de agregar productos por el código de barras. Finalmente, cuando compramos el producto podemos tacharlo de la lista pulsando en él. Y por último la posibilidad de compartir la lista de la compra si estás registrado en la aplicación.

No se encuentra ninguna innovación con respecto a las funcionalidades que ya hemos analizados en las anteriores aplicaciones en el campo de hacer la lista de la compra.

Anylist 4.7/5

La mayoría del resto de aplicaciones que buscamos en la tienda con la funcionalidad de hacer lista de la compra son similares a la que acabamos de analizar. No presentan ninguna novedad destacable, por ejemplo esta ultima aplicación que vamos a analizar a continuación tiene la posibilidad de hacer la lista de la compra con productos predeterminados, igualmente que vimos en la anterior, con la única diferencia que en este caso los productos están en inglés y no se permite cambiar el idioma.

En el panel inferior vemos las otras funciones que posee. La siguiente que nos fijamos es la de poder hacer recetas. Esta idea no es novedosa, ya lo hemos visto en *Fooder* y prácticamente funciona del mismo modo, e incluso de forma no tan eficiente, ya que cuando agregamos los ingredientes a la lista de la compra lo agregamos todos y no los que nos faltan. Obviamente esto ultimo no puede funcionar de forma eficiente porque no existe un inventario en el que comprobar si ya tenemos esos ingredientes. Así mismo, se puede organizar por grupos de productos al igual que en aplicaciones anteriores. La única funcionalidad destacable es la posibilidad de crear un menú mensual con todas las recetas día por día como vemos en la Figura 2.13, logrando así una mayor organización.

En resumen, no hay apenas nada destacable que podamos sacar de las dos

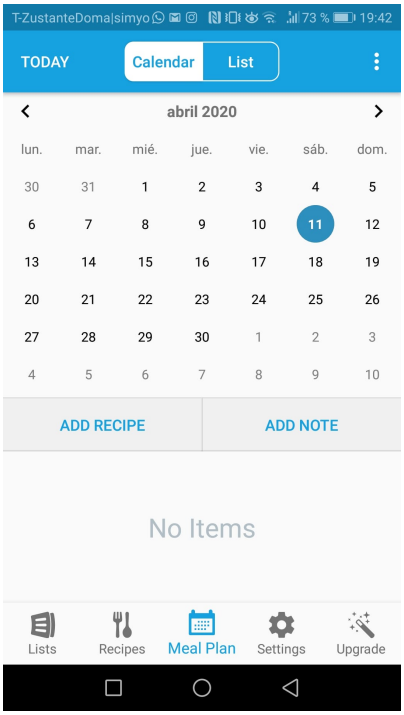


Figura 2.13: Calendario del menú de la aplicación 7

aplicaciones analizadas. Casi todas las funcionalidades ya se han visto en el resto de aplicaciones de inventariar.

2.4. Reflexión

Una vez realizado el análisis de las aplicaciones anteriores, las cuales son competencia directa para nuestra aplicación, podemos sacar infinidad de ideas, en general todas son útiles y tienen funcionalidades básicas y no tan básicas que destacar. Ninguna cumple con todos los requisitos que busca nuestra aplicación, si es cierto que hay algunos que son inamovibles como puede ser el poder inventariar, la posibilidad de tener una lista de la compra la cual se cree automáticamente a partir de los productos que falten en el inventario, y el poder agregar y eliminar productos usando el código de barra, siendo obvio entonces la necesidad de la lectura del código con nuestro dispositivo móvil como otra funcionalidad básica.

En lo que respecta a estas tres funcionalidades siempre hay una que todas las aplicaciones no logran hacerlo de la mejor forma posible, agregar y eliminar artículos de nuestro inventario, esta es la tarea que requiere más tiempo y la que

hay que mejorar en todas.

En casi todas las aplicaciones agregar un producto nuevo se puede hacer mediante la lectura del código de barras, pero solamente una permite agregar y eliminar unidades de artículos ya existentes en el inventario por medio del código de barras. Esto provoca una clara pérdida de tiempo que se podría ahorrar si la funcionalidad del código de barras no fuera únicamente para la búsqueda del producto, si no que además nos permita agregar y eliminar productos. Además ya no es solo la forma automática, la forma manual de agregar o eliminar unidades nos obliga a entrar al propio producto para cambiarlo, cuando sería mucho más cómodo si desde la propia lista del inventario nos permitiese alterar el número de unidades como explicamos previamente en la primera aplicación. La aplicación ha de centrarse en conseguir que la funcionalidad de agregar y eliminar productos del inventario no sea algo tedioso, logrando esto afianzaremos a que los usuarios quieran invertir un poco de tiempo en organizar su despensa.

Tras hablar de estas tres opciones indispensables en nuestra aplicación valoremos cuáles otras no tan básicas pueden ser de utilidad y qué podríamos cambiar de ellas para mejorarlas.

1. **Grupos y etiquetas.**"Sistema de inventario inteligente" nos permitía organizar los productos creando grupos y asociando estos a los artículos deseados. Una idea casi obligatoria si lo que se busca es organización, aun así es un poco redundante la idea de poder crear etiquetas ya que al fin y al cabo cumplen la misma funcionalidad, por lo que se podría obviar esta última.

2. **Menú desplegable.** Como hemos visto, casi todas poseen un menú que permite navegar por la aplicación sin obligarte volver al menú principal, consiguiendo no sobrecargarlo de opciones y hacerlo sencillo para que cualquier persona pueda usar las funcionalidades básicas sin tener que invertir mucho tiempo.

3. **Modo de exploración.** La idea de agregar y eliminar a partir del código de barras está representada con este modo, sin embargo tiene margen de mejora. Sería quizás más inteligente dividirlo en dos opciones, una para agregar y otra para eliminar, y no en el mismo modo las dos opciones ya que lo hace un poco confuso de primera mano.

4. **Colaboración.** Permitir que varios usuarios puedan conectarse a la misma lista de la compra es una función muy útil ya que habrá un gran número de familias o de personas que compartan piso a las que les puede ser de bastante ayuda que exista esta posibilidad de colaboración en la aplicación.

Incluso sería aun más interesante si se mostrase los gastos de cada integrante, controlando así que gasta cada uno y quien contribuye más al inventario del hogar, y poder llevar las cuentas de este.

5. **Automatización.** Como ya se ha dicho en numerables ocasiones la idea principal es el ahorro de tiempo del usuario, por lo que hay que lograr que todas las tareas sean lo más automáticas posibles. Cuando se agregue un producto hay que intentar que la mayor parte de los parámetros se rellenen con datos que se obtendrán de otra base de datos. Rellenar la lista de la compra con los productos que son necesarios comprar es otra de las mejoras de automatización.

Estas son las todas las mejores funcionalidades que se han podido encontrar a partir de las aplicaciones analizadas y aun así todavía quedan algunas otras que también pueden aportar bastante como puede ser un tutorial, la posibilidad de crear atributos personalizados en cada producto o tener un historial de todos los movimientos.

En resumen, nuestra aplicación se centrará en juntar todas estas buenas ideas, mejorarlas en lo que sea necesario y hacerla lo más accesible para todas las personas que la usen, logrando así que puedan invertir no demasiado tiempo en estas tareas domésticas. Es decir, nuestra aplicación no puede ganar frente a las otras en lo que respecta a novedades, como hemos visto todas las funcionalidades útiles ya son usadas por el resto, nuestro sello de diferencia ha de ser la simpleza de ella, la sencillez de usarla, comodidad, y sobre todo el ahorro de tiempo en todas las tareas que se puedan automatizar.

Capítulo 3

Análisis

Contenido

El desarrollo de un software, o en este caso una aplicación, es algunas veces un proceso tedioso y con muchas etapas en las que se va estructurando la idea de la aplicación. Por lo que es imprescindible, por no decir obligatorio, definir antes de empezar el desarrollo de lo que se va a encargar nuestra aplicación. Aquí es donde entran en juego los requisitos funcionales y no funcionales.

Con los requisitos funcionales definimos como reaccionará la aplicación frente al uso del usuario, es decir, prácticamente determinar cuáles serán las funcionalidades del software. Por otra parte los requisitos no funcionales nos ayudan a conocer que propiedades queremos que tenga nuestra aplicación, es decir no nos dice lo que hace el sistema como es el caso de los en los requisitos funcionales, si no que nos definen como queremos que se hagan esas funcionalidades.

En los siguientes apartados analizaremos los requisitos necesarios para nuestra aplicación.

3.1. Lista de requisitos funcionales

1. **RF1: El usuario tendrá que registrarse.** El usuario deberá registrarse para poder utilizar la aplicación. Será necesario que rellene un formulario con datos fundamentales como puede ser el nombre, un correo electrónico, una

contraseña ... Una vez terminado el usuario tendrá una cuenta personal en la aplicación

2. **RF2: El usuario dispondrá de una despensa personal.** La idea de la principal es mantener una despensa personal para cada uno de nuestros usuarios. Cuando este entre a la aplicación tendrá la opción de acceder a su inventario y saber los productos que tiene en cualquier momento.

3. **RF3: El usuario podrá agregar productos al inventario.** Una de las funcionalidades básicas para que la aplicación tenga sentido es la posibilidad de agregar nuevos productos a su inventario, ya sea de forma manual o algo más automática.

4. **RF4: El usuario podrá agregar unidades de un producto a su inventario.** En el caso de que el producto ya esté agregado por el usuario, se le permitirá modificar el número de unidades o cantidad de ese producto. Podemos contemplar dos maneras de aumentar las unidades, desde el menú de la despensa gracias a la forma manual, o mediante el código de barras, algo más automatizado.

5. **RF5: El usuario podrá eliminar unidades de un producto a su inventario.** Al igual que el requisito anterior, el usuario podrá eliminar de la despensa de una forma manual o algo más automática.

6. **RF6: Los productos agregados se revisarán.** Cuando el usuario agregue un producto a nuestra base de datos, este producto obtendrá el nombre del resultado en una búsqueda de Google. Este nombre será revisado, y el administrador de la base de datos dará como valido a los productos con nombre y caracteres que concuerden con el código de barras. Esto puede servir de cara a usar nuestra base de datos para que otras aplicaciones puedan acceder a un registro de productos con sus códigos de barras correspondiente

7. **RF7: El usuario dispondrá de una lista de la compra.** Otra de las funcionalidades principales de la aplicación es poder facilitar la tarea de ir a comprar mejorando la tradicional lista de la compra. Cada usuario tendrá una lista de la compra en la que podrá ver, en todo momento y cualquier lugar, los productos que necesita comprar.

8. **RF8: La lista de la compra se rellenará automáticamente.** La optimización de tiempo es algo que se prima mucho en la aplicación, por lo que una opción permitirá al usuario agregar de forma automática todos los productos que hayan bajado del límite de unidades que el propio producto posee en sus atributos, logrando así agilizar la tarea de crear la lista de la compra.

9. **RF9: Se podrá agregar productos a la lista de la compra de forma manual.** Si el producto que se desea agregar a la lista de la compra no se encuentra en nuestra despensa se podrá agregar de forma manual. Una opción nos permitirá agregar el nombre de ese producto, las unidades necesarias, así como al grupo que pertenece si se desea.

10. **RF10: Cada producto tendrá varios atributos.** Cuando un producto sea agregado a la despensa personal de un usuario o a la base de datos, será necesario rellenar unos campos obligatorios como podrá ser el código de barras, el nombre del producto o las unidades, y algunos optativos como podría ser el grupo al que pertenece dicho producto.

11. **RF11: El producto podrá agregarse manualmente.** Como ya se ha dicho anteriormente, el usuario podrá agregar productos a su despensa. Una opción del menú nos permitirá agregar el producto de forma manual, esto quiere decir que el usuario deberá añadir todos los atributos del producto él mismo.

12. **RF12: El producto podrá agregarse con el código de barra.** Otra opción para agregar el producto será de una forma más automática, mediante el código de barras. Una opción en el menú principal permitirá agregarlo de esta forma, la aplicación leerá el código de barras y buscará en Google el nombre del producto, posteriormente si no lo encuentra buscará en la base de datos propia de la aplicación. Si se da la situación que no la encuentra en ninguno de los casos, se le pedirá al usuario que lo introduzca de forma manual.

13. **RF13: Habrá un menú principal.** Cuando el usuario acceda a la aplicación la primera pantalla que se le mostrará será el menú principal. Allí se encontrarán todas las posibles funciones que ofrece la aplicación. Además este menú estará acompañado de un menú lateral desplegable para mejorar la navegabilidad.

14. **RF14: Se podrán eliminar productos del inventario y de la lista de la compra.** En caso de que se haya agregado un producto de forma errónea el usuario podrá eliminarlo.

15. **RF15: Habrá diferentes grupos.** Una forma para organizar los productos son los grupos. Gracias a ellos se permitirá clasificar los productos en unos grupos predeterminados, haciendo más fácil la búsqueda sesgando solo por grupos. Los grupos serán fijos en la aplicación, el usuario no puede crear sus propios grupos.

16. **RF16: El inventario se mostrará por grupos.** En el caso de que un usuario tenga una gran cantidad de productos, se mostrará la despensa organizada por grupos, logrando así que la búsqueda de un producto en concreto sea más sencilla.

17. **RF17: Agregar o eliminar unidades mediante el código de barras.** Una de las funcionalidades pensada para el ahorro de tiempo es la facilidad de agregar o eliminar más de una unidad al mismo tiempo. Es decir, una opción de la aplicación nos permitirá pasar solamente una vez el código de barras y el usuario introducirá cuantas unidades se introducen o eliminan de la despensa, permitiendo así que si se quieren añadir un gran número de unidades del mismo producto, solo sea necesario leer el código de barras una única vez.

18. **RF18: Búsqueda de productos en una base de datos propia.** La aplicación guardará todos los productos validados en una base de datos propia, y servirá para todos los usuarios los productos agregados por un usuario.

19. **RF19: Búsqueda del producto en Google** Previamente a la búsqueda en la base de datos de la propia aplicación, se hará una búsqueda en Google con el codigo de barras escaneado por el usuario. Una vez realizada la búsqueda, obtendremos el texto de una de las cabeceras de los enlaces en la primera página de resultados de Google. Este es el nombre que posteriormente será validado por el administrador ya que podrá existir un margen de error en la busqueda del nombre, pudiendo ser incorrecto o no devolviendo algún resultado.

20. **RF20: Un usuario solo poseerá una despensa.**

21. **RF21: Un usuario solo poseerá una lista de la compra.** Es innecesario que existan varias lista de la compra, para una mayor organización ya existen los grupos.

22. **RF22: Un usuario tendrá atributos propios.** Aunque a la hora de guardar los productos en la base de datos, el nombre del producto global será el encontrado en Google, sin embargo, todos los usuarios podrán personalizar el nombre de ese producto al que ellos deseen, así como también podrá establecer a que grupo pertenece ese producto. Es decir un producto que esté en el inventario de dos usuarios diferentes, puede encontrarse además en grupos diferentes.

23. **RF23: Restricción en las unidades de los productos.** Un producto tendrá varias restricciones en lo que respecta a su medida. La única medida serán

las unidades, es decir se omite la posibilidad de medir los productos en kilogramos, litros, etc..., para así lograr una mayor sencillez en la aplicación. Como es obvio las unidades no podrán ser negativas, y no podrán superar la cantidad de 99 unidades. Esta decisión viene dada para una mejor implementación del interfaz, así como de lógica, es poco probable que en el uso cotidiano de un usuario almacene más de 99 unidades de un solo producto.

24. **RF24: Un usuario podrá editar los atributos de sus productos.** Tras establecer por primera vez los atributos, un usuario podrá cambiar sus valores si lo desea.

25. **RF25: La lista de la compra se podrá borrar rápidamente.** Un botón nos permitirá eliminar todos los productos de la lista de una forma fácil y rápida, para así evitar perder tiempo en eliminar uno por uno los productos.

3.2. Lista de requisitos no funcionales

1. **RNF1: Cifrado de la contraseña.** Una vez registrado el usuario, la contraseña será guardada en nuestra base de datos mediante un algoritmo de cifrado para evitar que conozcan la contraseña de nuestros usuarios en caso de ataque.

2. **RNF2: Fluidez en la aplicación.** Se buscará un diseño sin mucha carga de datos ni de elementos para lograr una aplicación liviana, consiguiendo así que no sea una tara de cara a usar aplicación diariamente.

3. **RNF3: Un servidor disponible todo el tiempo.** El back-end de la aplicación estará albergado en una instancia virtual que nos ofrece Amazon, dando así disponibilidad 24 horas los 7 días de la semana a todos los usuarios. Por esto mismo el groso de usuarios que podrán conectarse al mismo tiempo estará condicionado con las características de esta instancia que Amazon nos ofrece durante 12 meses.

3.3. Casos de uso

Tras definir los requisitos funcionales y no funcionales del sistema, y para establecer con mayor precisión como se va a comportar el sistema frente a las interacciones del usuario se van a crear una serie de casos de usos. De esta forma

especificamos de forma más precisa las funcionalidades y evitamos errores a la hora del diseño y de la implementación.

Caso de uso	Agregar producto con código de barras
Resumen	El usuario podrá agregar productos al inventario mediante el código de barras
Actores	Usuario y administrador
Precondición	El producto puede estar agregado o no, esto quiere decir que si ya está agregado nos informará de los datos propios de ese producto de la despensa del usuario ya sea unidades, nombre y el propio código de barras. Si no está agregado, el usuario podrá ver los datos generales que corresponden a ese producto.
Postcondición	El producto es agregado por el administrador en el inventario.
Curso Normal	1. El usuario usa el lector del código de barras para agregar el producto. 2. El administrador busca el nombre del producto en la despensa del usuario. 3. El administrador encuentra el producto en la base de datos, y presenta en la interfaz del usuario los datos guardados en la base de datos por parte del usuario. 4. El usuario introduce los campos necesarios, como el nombre que ya es proporcionado automáticamente, las unidades y el tipo del producto. 5. El administrador verifica que los campos sean correctos, y lo agrega al sistema.

Curso Alternativo	1. El administrador no encuentra el producto en la despensa del usuario. Busca si se encuentra en la tabla de productos general. Cuando lo encuentra muestra el nombre a nivel global, para agregar posteriormente a la despensa personal del usuario. 2. El administrador no encuentra el producto ni en la despensa del usuario, y es un producto nuevo que ningún otro usuario ha introducido previamente. El administrador se encarga de hacer un scrapping a los resultados de la búsqueda del código de barras en Google. Posteriormente, presenta este nombre al usuario para que este complete los campos necesario para su agregación al sistema.

Tabla 3.5: Caso de uso: Añadir producto con el código de barras

Caso de uso	Registrar usuario.
Resumen	El usuario se registrará en la aplicación para poder acceder a las funcionalidades.
Actores	Usuario y administrador.
Precondición	El usuario no puede estar registrado en la aplicación. Así como el correo debe ser correcto y seguir el formato adecuado.
Postcondición	El usuario es creado en el sistema y se tendrá acceso a él.
Curso Normal	1. El usuario introducirá los datos necesarios. 2. El administrador comprobará que los datos introducidos sean correctos. 3. El administrador comprobará que el usuario no existe. 4. El administrador registrará al usuario en la base de datos. 5. El sistema informará que se ha registrado correctamente el usuario, y este podrá iniciar sesión en la aplicación.
Curso Alternativo	1. El sistema informa que alguno de los campos no son correctos. 2. El sistema informa que alguno el usuario ya existe e indica al usuario que registre otro correo electrónico. 3. El sistema informa que las contraseñas introducidas no son coincidentes. 4.
Observaciones	

Tabla 3.1: Caso de uso: Registrar usuario

Caso de uso	Iniciar sesión
Resumen	El usuario iniciará sesión para tener acceso a todas las funcionalidades de la aplicación.
Actores	Usuario y administrador.
Precondición	El usuario tiene que estar registrado en el sistema.
Postcondición	El usuario accede al menú principal de la aplicación donde tiene todas las funcionalidades disponible.
Curso Normal	1. El usuario rellena los campos para el inicio de sesión. 2. El administrador verifica que el correo tenga un formato correcto. 3. El administrador verifica que el usuario y contraseña estén relacionadas en el sistema. 4. El usuario accede al menú.
Curso Alternativo	1. El administrador informa al usuario de un error en el correo electrónico. 2. El administrador informa que el usuario no existe en el sistema y ha de introducirse otro. 3. El administrador informa que la contraseña es incorrecta con el correo electrónico dado

Tabla 3.2: Caso de uso: Inicio de sesión

Caso de uso	Agregar producto
Resumen	El usuario podrá agregar productos tanto de la lista de la compra como del inventario.
Actores	Usuario y administrador
Precondición	El producto no puede estar agregado, es decir no puede existir el mismo producto varias veces tanto en la lista de la compra como en el inventario
Postcondición	El producto es agregado por el administrador tanto en el inventario como en la lista de la compra.
Curso Normal	1. El usuario introduce los campos necesarios para agregar el producto. 2. El administrador verifica que todos los campos son correctos. 3. El administrador añade el producto, al inventario o la lista de la compra, del usuario.
Curso Alternativo	1. El usuario introduce de forma errónea los campos 2. El administrador notifica al usuario que alguno de los campos son erróneos.

Tabla 3.3: Caso de uso: Añadir producto

Caso de uso	Eliminar producto
Resumen	El usuario podrá eliminar productos tanto de la lista de la compra como del inventario.
Actores	Usuario y administrador
Precondición	El producto ha de estar agregado, tanto en el inventario o en la lista de la compra, para que este pueda ser eliminado
Postcondición	El producto es eliminado del inventario o de la lista de la compra.
Curso Normal	1. El usuario hace click en el botón de eliminar que cada producto tiene en su etiqueta. 2. El administrador elimina el producto de la base de datos. 3. El administrador refresca la interfaz.

Tabla 3.4: Caso de uso: Añadir producto

Caso de uso	Eliminar unidades con el código de barras
Resumen	El usuario podrá eliminar unidades, de un producto que se encuentre en su inventario, mediante el código de barras.
Actores	Usuario y administrador
Precondición	El producto tiene que existir en el inventario personal del usuario.
Postcondición	Las unidades notificadas por el usuario son eliminadas del sistema.
Curso Normal	1. El usuario usa el lector del código de barras para eliminar unidades del producto. 2. El administrador muestra en la interfaz los atributos del producto al que corresponde el código de barras. 3. El usuario indica las unidades a eliminar y el administrador verifica que sea inferior o igual al número de unidades totales. 4. Las unidades son eliminadas del sistema.
Curso Alternativo	1. El usuario lee un código de barras que no corresponde a ninguno perteneciente al de su inventario. El sistema muestra un mensaje de error en la interfaz.

Tabla 3.6: Caso de uso: Eliminar unidades con el código de barras

Caso de uso	Añadir o eliminar unidades el el inventario o en la lista de la compra
Resumen	El usuario podrá aumentar o disminuir la cantidad de unidades
Actores	Usuario y administrador.
Precondición	El usuario podrá aumentar las unidades siempre que se encuentre en el rango definido de unidades máximas y mínimas, en este caso 0 y 99.
Postcondición	Las unidades son agregadas en el sistema.
Curso Normal	1. El usuario hará click en el botón de agregar o eliminar unidades correspondientes a cada etiqueta del producto en el inventario. 2. El administrador verificará que las unidades estén en el rango correcto. 3. El administrador modificará las unidades en el sistema y refrescará la interfaz.

Tabla 3.7: Caso de uso: Agregar o eliminar unidades del inventario o de la lista de la compra.

Caso de uso	Editar producto
Resumen	El usuario podrá editar los atributos de un producto de su inventario ya añadido en su inventario, tales como: las unidades limites, el tipo del producto y el nombre.
Actores	Usuario y administrador.
Precondición	El producto tiene que existir en el inventario.
Postcondición	El administrador actualiza los valores de los atributos del producto en el sistema.
Curso Normal	1. El usuario hará click en el botón de editar correspondiente a la etiqueta de dicho producto. 2. El administrador mostrará en la interfaz un cuadro de dialogo con los atributos disponibles a cambiar. 3. El administrador verificará que todos los campos sean correctos. 4. El administrador cambiará los valores de los atributos en el sistema.
Curso Alternativo	1. El sistema informa que alguno de los atributos introducidos son erróneos.

Tabla 3.8: Caso de uso: Editar producto

Caso de uso	Borrar la lista de la compra
Resumen	El usuario podrá borrar la lista de la compra completamente pulsando un único botón
Actores	Usuario y administrador.
Precondición	La lista de la compra debe de tener al menos un producto.
Postcondición	El administrador borrará los productos de la lista de la compra del sistema.
Curso Normal	1. El usuario hará click en el botón de borrar la lista de la compra. 2. El administrador borrará los productos que se encuentren en la lista de la compra del usuario 3. El administrador refrescará la interfaz y el usuario podrá ver su lista de la compra actualizada.

Tabla 3.9: Caso de uso: Borrar la lista de la compra

Capítulo 4

Herramientas del desarrollo

Contenido

4.1. Android Studio

Android Studio [5] será el IDE (*Integrated Development Environment*) para desarrollar el front-end de la aplicación, con Java como lenguaje. Sin duda es el mejor entorno de programación si lo que se está buscando es desarrollar exclusivamente la aplicación en Android, como es el caso.

Su software basado en *IntelliJ IDEA* desarrollado por JetBrains, al igual que PhpStorm el otro IDE usado para el desarrollo de la aplicación, facilita en gran medida la fluidez en la programación. Andorid Studio no solo tiene de única

opción Java como lenguaje de programación, si no que también nos ofrece la posibilidad de complementar el desarrollo con *Kotlin*. Como *Kotlin*, hay decenas de lenguajes de programación y frameworks para poder desarrollar el front-end: JavaScript, ReactNative, NativeScript, Ionic... Así como también podemos encontrar otros entornos de desarrollo tales como *Visual Studio* [6] desarrollado por *Microsoft*. Elegir Android Studio como entorno simplemente es por el hecho de la facilidades que ofrece para desarrollar exclusivamente en Android, además, la decisión de usar Java como lenguaje también es definitiva, por lo que es el IDE ideal.

4.2. Java

Uno de los lenguajes de programación más usados para aplicaciones cliente-servidor es Java [7]. Una vez definido que el IDE iba a ser Android Studio, hay que definir cuál de los lenguajes que nos permite usar va a ser el elegido: Java o Kotlin. Aquí no se ha tomado una decisión en base a cuál de los lenguajes aporta mejores ventajas a la hora de programar, la elección ha sido tomada íntegramente por la experiencia y los conocimientos que ya se poseen en Java. Esta experiencia agilizará en gran medida el desarrollo, evitando la inversión de horas innecesaria en aprender otro lenguaje.

4.3. AWS

Al desarrollar una aplicación tenemos que tener en cuenta la gran cantidad de datos que serán necesarios de gestionar. De hecho por pequeña o escueta que sea nuestra aplicación estamos obligados a usar una Base de datos para almacenar la información de nuestros usuarios. Ahora bien, el almacenaje de todos estos datos no ocupan un espacio etéreo, como cualquier elemento es necesario guardarlos en físico. La propia definición de servidor proporcionada por la RAE nos dice que es una " Unidad informática que proporciona diversos servicios a computadoras conectadas con ella a través de una red ." [8]

Amazon Web Service es una plataforma en la nube que proporciona a sus usuarios más de 175 servicios a nivel global. Por norma general estos servicios suelen ser de pago, para que las empresas utilicen esta plataforma de Amazon en los campos de Big Data, Cloud Computing, desarrollo de aplicaciones web, almacenaje de datos... Sin embargo Amazon nos proporciona algunos servicios durante 12 meses de forma gratuita. AWS nos permite crear una instancia de un

sistema, en nuestro caso con Ubuntu 18.02 como sistema operativo, que usaremos como servidor para nuestra aplicación, es decir, en esta instancia la cuál nos ofrece Amazon se creará el servidor que albergará la base de datos.

AWS es sin duda de las únicas opciones que podemos escoger cuando se habla de montar un servidor sin coste alguno en un tiempo limitado, incluso nos podríamos atrever a decir que Amazon ofrece la mejor relación calidad precio, por la gran cantidad de servicios que posee.

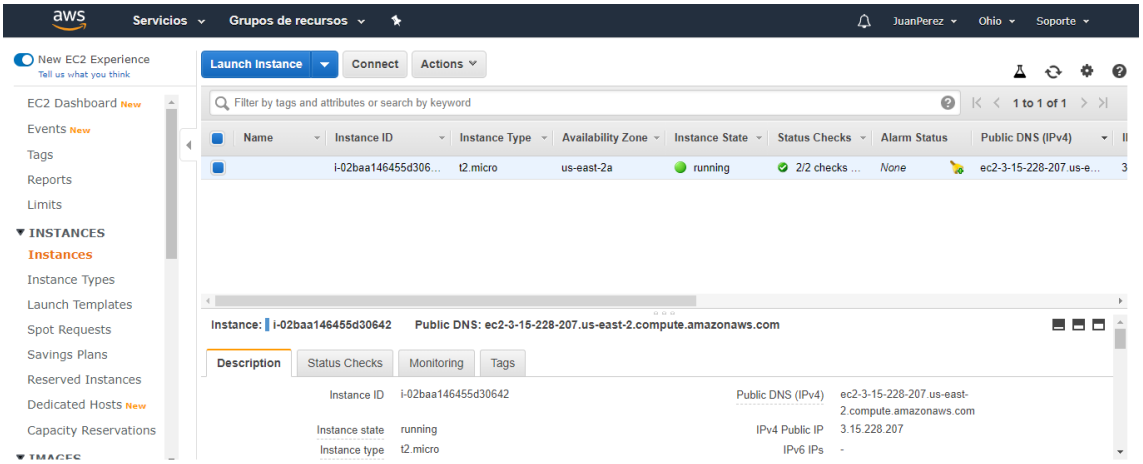


Figura 4.1: Menú principal de AWS

4.4. Apache

Al realizar una aplicación cliente-servidor como su propio nombre indica es necesario un servidor. Apache es un servidor HTTP el que se encargará de recibir las peticiones que se hagan en nuestro cliente, en este caso la aplicación en cuestión.

4.5. MySQL

MySQL [9] es el elegido para ser el SGDB (Sistema Gestor de Base de Datos). Cuando manejamos una base de datos es necesario tener un software como este para poder realizar peticiones SQL y obtener los datos de las tablas necesarias, instalamos este software en la instancia Ubuntu proporcionada por AWS. MySQL

funciona con una base de datos relacional, elegida para el desarrollo por la integridad en los datos que proporciona, además este software es de licencia pública, por lo que pertenece al sowtware libre y es gratuito.

4.6. PHP y PHPStorm

LAMP son las siglas para definir: Linux, el sistema operativo en el cuál funcionará la instancia del servidor, en este caso Ubuntu. Apache, el servidor como previamente hemos explicado y MySQL el sistema gestor de base de datos. La última sigla viene dada por PHP[10], el lenguaje de programación que nos permite escribir código que se ejecuta en el servidor generando código HTML, enviándolo al cliente y evitando que los usuarios puedan saber el código que hay en el servidor.

Por su parte, PHPStorm es el IDE en el que se va escribir el código necesario para el back-end.

4.7. phpMyAdmin

Para hacer más sencillo la comunicación con la base de datos y el servidor usaremos phpMyAdmin [11], un software que nos permite conectarnos a través del navegador web y nos proporciona una interfaz para gestionar la base de datos con la creación, alteración de tablas, así como la inserción de datos en ellas y otras tantas funcionalidades. El sotfware se comunica por debajo con el servidor mediante PHP, sin embargo el usuario solo necesitará realizar sentencias SQL y manejar la interfaz que se proporciona para la alteración de la base de datos sin necesidad de escribir ningún código en PHP.

Es un software imprescindible si se va a trabajar con PHP en el servidor y con una base de datos relacional.

4.8. Bitvise SSH Client Download

Ya que el servidor de nuestra aplicación se encuentra en la instancia alojada en los servidores de Amazon, es necesario poder comunicarse con el de una forma fácil y fluida. Para ello usaremos Bitvise [12], un cliente SSH que nos permitirá conectarnos a la instancia desde nuestro propio dispositivo local, además con la

posibilidad de usar una interfaz para el intercambio de ficheros, permitiendo escribir el código PHP en local, para posteriormente enviarlo al servidor de Apache y que nuestro cliente interactúe con dichos archivos.

4.9. Lucidchart

Para establecer el diseño de la aplicación y como va a estar estructurada son necesarios una serie de diagramas para hacer más fácil la posterior programación. Con Lucidchart[13] crearemos todos los diagramas UML (Unified Modeling Language) necesarios para nuestra aplicación. Definimos UML como: "Un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema"[14]. Lucidchart es un recurso online que nos permitirá crear estos diagramas, sin necesidad de descargar ningún software.

4.10. LaTeX y OverLeaf

Aunque Word es uno de software más usados para la redacción de documentos y sin duda es el más usado por personas amateurs, LaTeX es un sistema de composición de textos que presenten una alta calidad tipográfica. En general con LaTeX podemos conseguir una mayor simpleza y homogeneidad en nuestros documentos. Sin embargo, LaTeX ha de ser compilado, ya que basa su escritura en texto plano y comandos.

Para ello aparece OverLeaf, una plataforma en linea que nos permite crear nuestros documentos sin tener que descargarnos ningún software, y además poder usar plantillas para hacer más rápida y más profesional la escritura de nuestro documento.[15]

4.11. SQL Data Modeler

Al igual que usaremos Lucidchart para crear nuestros diagramas UML, SQL Data Modeler nos permitirá crear un modelo conceptual de nuestra base de datos y gracias a una opción de su software, se creará automáticamente el modelo entidad relación de la base de datos, evitándonos así fallos humanos con las relaciones o restricciones.

Capítulo 5

Diseño

Contenido

Durante el presente capítulo se hará una descripción del diseño de La Despensa, como ya se ha citado previamente el sistema sigue una arquitectura cliente-servidor por lo que se hablará, tanto en el apartado del cliente como del servidor. En el front-end especificaremos mediante diagramas UML las funcionalidades que el cliente puede hacer desde la aplicación para así fundamentar en el siguiente capítulo de la memoria la posterior implementación. De la misma forma se hará para el back-end de la aplicación, mientras se comentan los respectivos diagramas.

5.1. Arquitectura del sistema

Ya hemos hablado con anterioridad, en este mismo capítulo y en capítulos anteriores, cómo iba a ser la arquitectura de la aplicación: cliente-servidor.

La arquitectura cliente-servidor se basa en la conexión que existe entre dos dispositivos, el cliente que se encarga de hacer las peticiones al servidor para poder obtener las funcionalidades del sistema. Es decir, el cliente es el que pide la información y el servidor el que la suministra.

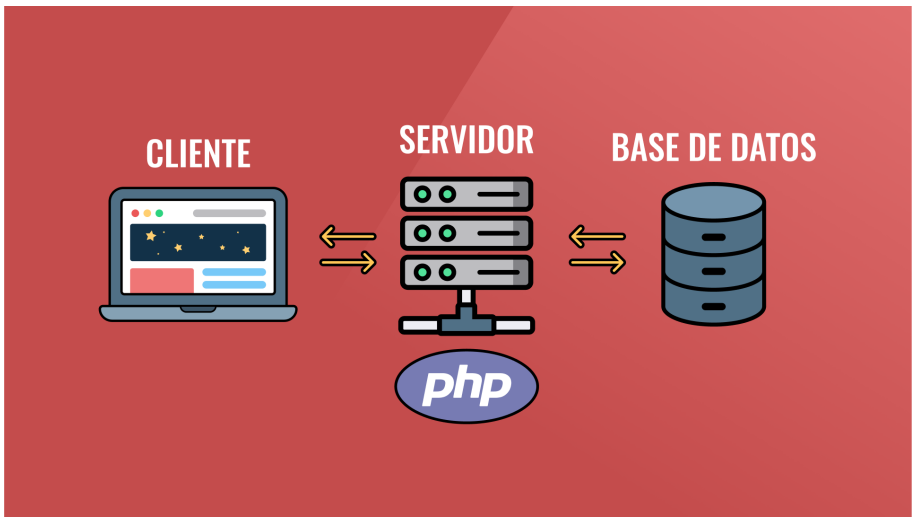


Figura 5.1: Ejemplo de la arquitectura cliente-servidor

En el desarrollo web y de aplicaciones nace el front-end que corresponde al apartado del cliente, el encargado de la lógica cuando se hace alguna petición, mientras que el back-end se encarga de la lógica de negocio y corresponde al servidor en la arquitectura descrita. [16]

5.2. Diseño del front-end

Para ayudar a entender el diseño del front-end de la aplicación se han creado diagramas de secuencia, permitiendo así mostrar el ciclo de vida de las funcionalidades que el usuario puede realizar cuando usa la aplicación.

5.2.1. Registro

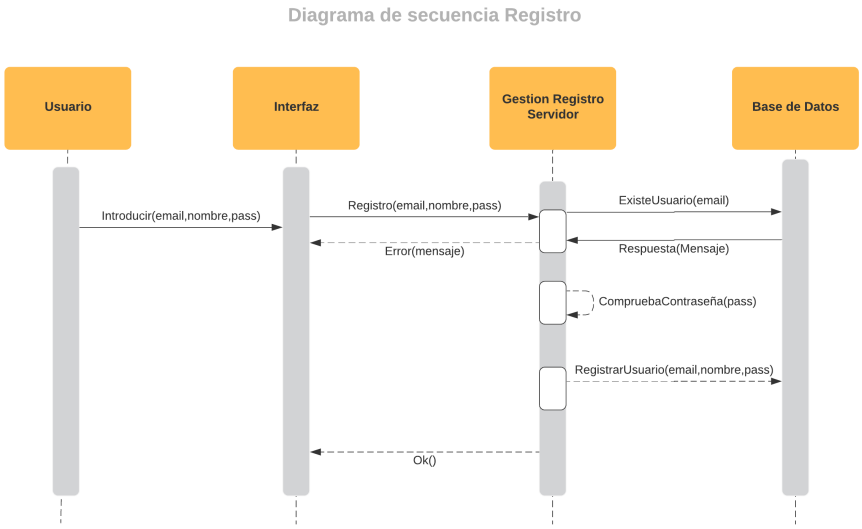


Figura 5.2: Diagrama de secuencia: Registro

Como nos dice el requisito funcional **RF1**, será necesario que el usuario es registren en nuestra aplicación, para ello será necesario que introduzca un correo electrónico, nombre de usuario y una contraseña. Una vez introducido en el cliente, estos atributos son enviados al servidor para su posterior consulta en la base de datos. Como vemos en el diagrama, se comprobará si ese usuario ya exista, en cuyo caso será enviado un mensaje de error, informando al usuario que es necesario introducir otro correo para el registro. En el caso que no exista el usuario, el servidor se encargará de enviar los datos para que el usuario sea registrado en la base de datos.

5.2.2. Inicio de sesión

Como es obvio, posterior al registro del usuario, necesitamos que el usuario inicie sesión para obtener todas las funcionalidades que ofrece la aplicación. Al igual que en el registro, una vez introducido los campos necesarios para el inicio de sesión no solo se comprueba en la base de datos si el usuario existe, si no que además se comprueba si la contraseña coincide con el correo electrónico. Por lo que en el caso que el correo no exista el cliente recibirá un mensaje de error diferente al caso en el que la contraseña y el correo no concuerden. Si no se pro-

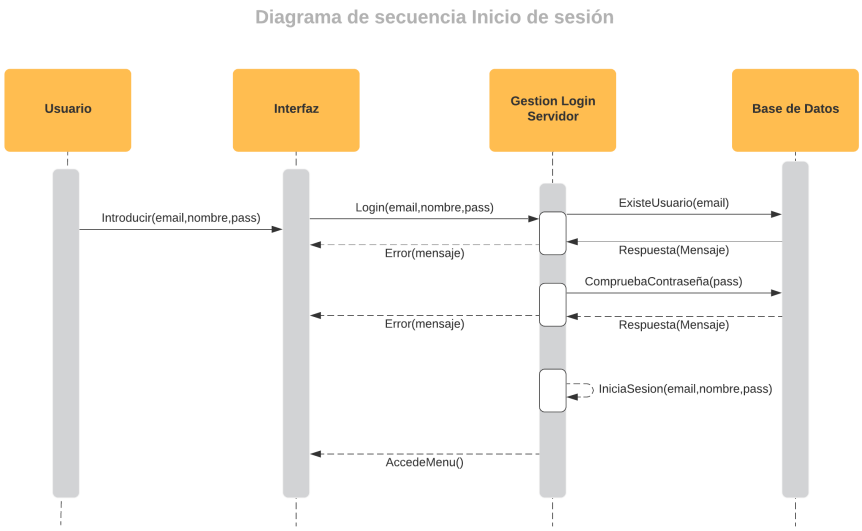


Figura 5.3: Diagrama de secuencia: Inicio de sesión

duce ningún fallo en la autenticación, la función **AccedeMenu()** que vemos en el diagrama, será la encargada de cargar el menú del usuario.

5.2.3. Acceder inventario

Una vez nos encontramos en el menú tras haberse realizado el inicio de sesión, el usuario tiene varias funcionalidades que puede realizar. Una de ellas es poder acceder a su Despensa tal y como nos define el requisito funcional **RF2**. El diagrama muestra de una forma sencilla como interactúa el cliente con el servidor para mostrar el inventario del usuario.

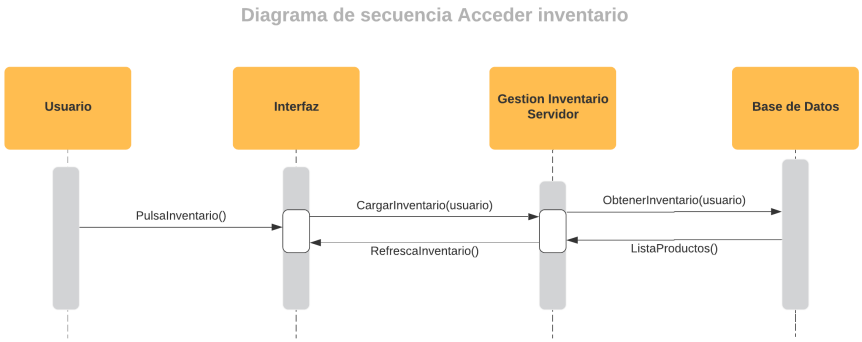


Figura 5.4: Diagrama de secuencia: Acceder al inventario

5.2.4. Acceder lista de la compra

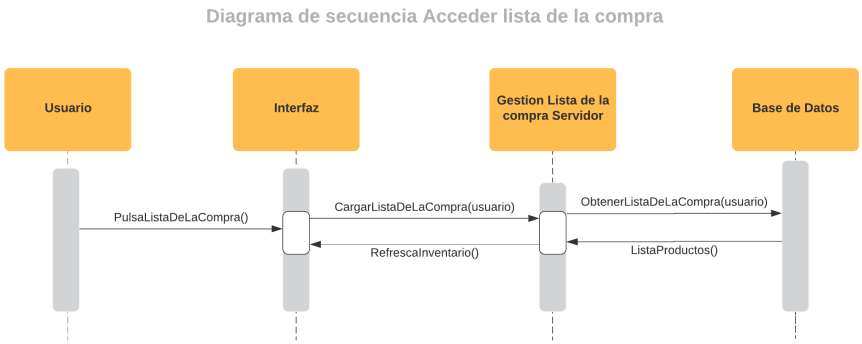


Figura 5.5: Diagrama de secuencia: Acceder a la lista de la compra

Al igual que el diagrama anterior y como se especifica en el requisito funcional **RF7**, el usuario posee y puede acceder a una lista de la compra. El diagrama, prácticamente idéntico al anterior, nos muestra como se carga el inventario en el cliente.

5.2.5. Agregar producto a la Despensa y lista de la compra

El requisito funcional **RF11** especifica que un usuario podrá agregar productos manualmente, esta funcionalidad será similar tanto para agregarlo al inventario al igual que para agregarlo a la lista de la compra. El usuario introducirá los atributos que la aplicación le requiera, comprobando que todos los campos intro-

ducidos estén completos. En el caso de que sea incorrecto, la aplicación mostrará un mensaje de error en la interfaz, justo como se ve en el diagrama.

Una vez comprobado que todos los campos sean correctos, el cliente envía el producto al servidor para agregarlo a la base de datos, aunque previamente el servidor, como de ve en la función **ExisteEnInventario(producto)**, se comprueba si el usuario ya posee el producto introducido. En el caso de que exista se envía un mensaje de error, mientras que si es correcto el producto se agregar al inventario.

Esta funcionalidad se puede extrapolar de forma idéntica a la lista de la compra.

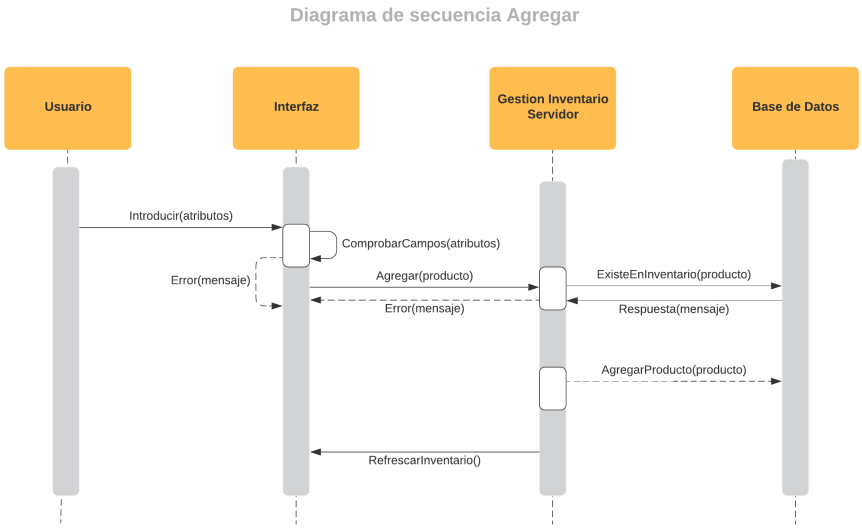


Figura 5.6: Diagrama de secuencia: Agregar productos

5.2.6. Eliminar producto de la Despensa y lista de la compra

El siguiente diagrama corresponde a la funcionalidad especificada en el requisito funcional **RF14**. El usuario podrá eliminar manualmente el producto de su despensa, al igual que ha podido agregarlo. Sin embargo, este diagrama difere del de agregar, porque no es necesario comprobar que el producto existe ya que la función **PulsarBotonBorrar(producto)** será específica de cada producto que se encuentre en el inventario, por lo que se da por su puesto que el producto existe para el usuario. Cuando el producto es borrado, el cliente actualiza el inventario.

Esta funcionalidad se puede extrapolar de forma idéntica a la lista de la compra.

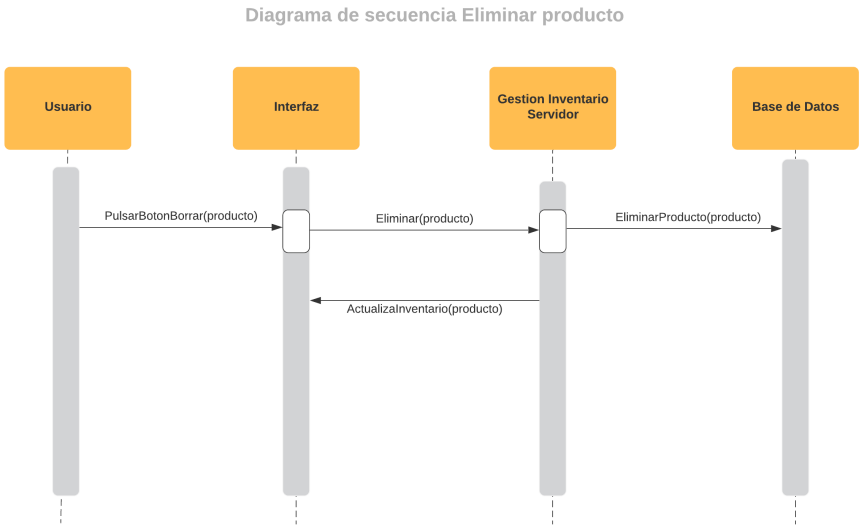


Figura 5.7: Diagrama de secuencia: Eliminar productos

5.2.7. Agregar unidades en la Despensa y lista de la compra

Cuando un usuario pulse el botón para agregar unidades de un producto se realizará un conjunto de acciones como vemos en el diagrama. Primeramente se comprobará en el cliente de la aplicación si se superan más de las unidades máximas, que un producto puede tener como se especifica en el requisito funcional **RF14**. Tras comprobar este requerimiento, el cliente informará de la actualización de las unidades totales al servidor y posteriormente refrescará la interfaz, con las unidades actualizadas. Se puede apreciar el diagrama en la figura 5.8

Esta funcionalidad se puede extrapolar de forma idéntica a la lista de la compra.

5.2.8. Eliminar unidades en la Despensa y lista de la compra

Este diagrama es prácticamente igual al de la figura 5.8, simplemente con la variación en la comprobación de las unidades mínimas en este caso. La función **ComprobarUnidadesMax()** cambiaría por **ComprobarUnidadesMin()**.

Esta funcionalidad se puede extrapolar de forma idéntica a la lista de la compra.

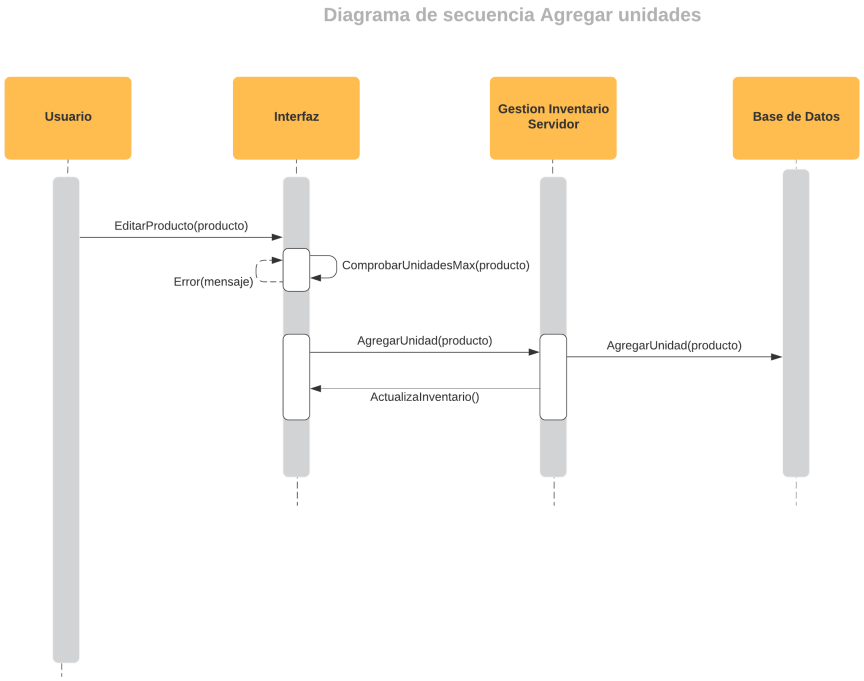


Figura 5.8: Diagrama de secuencia: Agregar unidades a un producto

5.2.9. Agregar producto a la Despensa con código de barras

Esta es una de las funcionalidades más importantes de la aplicación, como se muestra en el requisito funcional **RF12**. Mediante la lectura del código de barras del producto,la cuál se comprueba en el cliente de la aplicación que se correcta, el usuario podrá obtener el nombre automáticamente de dos formas posibles: El servidor se encargará de verificar de si el producto existe en la base de datos, es decir, si ha sido previamente añadido. Si el producto se encuentra el la base de datos, el usuario podrá ver en la interfaz de la aplicación el nombre extraído. En el caso de no encontrase, el servidor se encargará de hacer una petición a Google, como vemos en la función **BuscarProducto(producto)**, y obtener el nombre a raíz del código de barras. De esta forma el usuario tendrá una forma más mecanizada a la hora de agregar productos a su inventario, verá en la interfaz un nombre, predeterminado y todos los atributos correspondientes al producto, si el producto ya existía previamente en La Despensa del usuario.

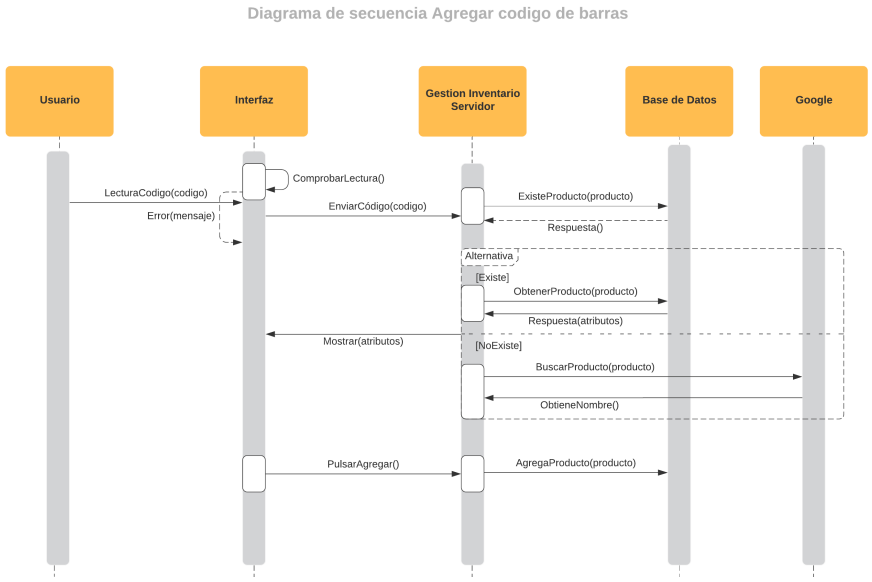


Figura 5.9: Diagrama de secuencia: Agregar producto con código de barras

Además un campo numérico permitirá al usuario introducir el número de unidades que desee, y mediante la función **AgregarProducto(producto)** se establecerá en el inventario del usuario, el nombre del producto con las unidades introducidas.

5.2.10. Eliminar unidades de la Despensa con código de barras

Al igual que se nos permite agregar productos mediante el código de barras, también es posible eliminarlos, tal y como se muestra en el requisito funcional **RF17**. Una vez hecha la lectura del código, y que se compruebe que es correcta, el cliente enviará el código al servidor que verificará si ese producto existe en el inventario del usuario, si no existe se mostrará un mensaje de error en la interfaz. En el caso de que el producto si exista en el inventario, se le mostrará al usuario todos los detalles de ese producto.

En ese momento el usuario podrá introducir una cantidad de unidades que desee, si las unidades a eliminar son mayores que la cantidad total de unidades que podemos encontrar en la despensa del usuario, se mostrará un mensaje de error, impidiendo eliminar tal cantidad. En el caso de que sea inferior o igual, el servidor actualizará la base de datos eliminando las unidades deseadas.

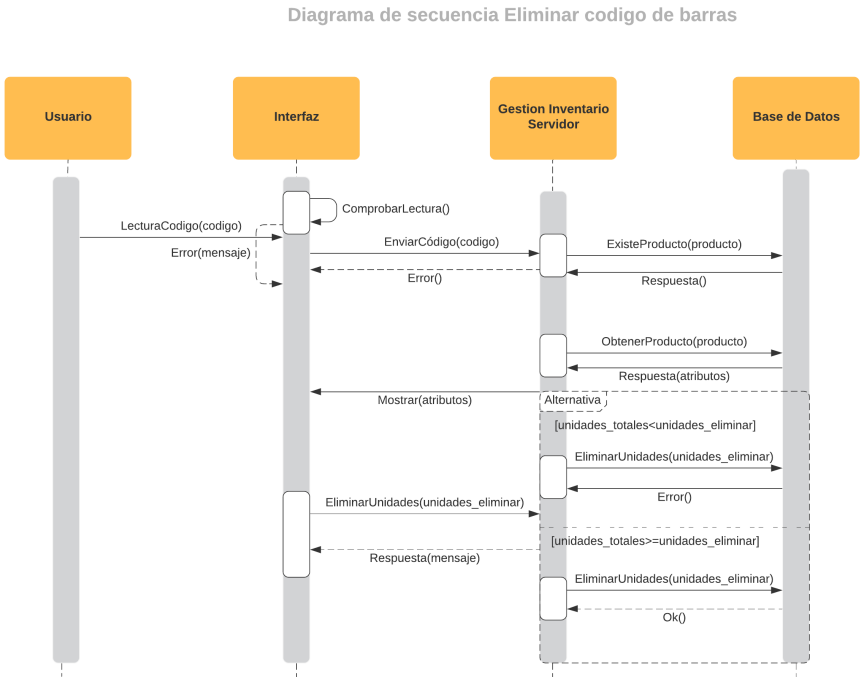


Figura 5.10: Diagrama de secuencia: Eliminar unidades con el código de barras

5.2.11. Editar producto de la Despensa

Cada producto del inventario tendrá la posibilidad de modificar algunos de sus atributos, como se especifica en el requisito funcional **RF24**. Al igual que el diagrama que vemos en la figura 5.7, no es necesario hacer una comprobación de la existencia del producto, ya que este se encuentra en la lista del inventario.

Cuando un usuario pulsa el botón de editar, que pertenece a un producto de su inventario, el servidor enviará los atributos de ese producto y se mostrarán a través de la interfaz del cliente.

Posteriormente cuando se acepten los nuevos campos editados del productos, serán enviados al servidor por parte del cliente, y finalmente modificados en la base de datos por la función **CambiarValores(producto)**. Finalmente la respuesta por parte del servidor refrescará el inventario en la interfaz del usuario.

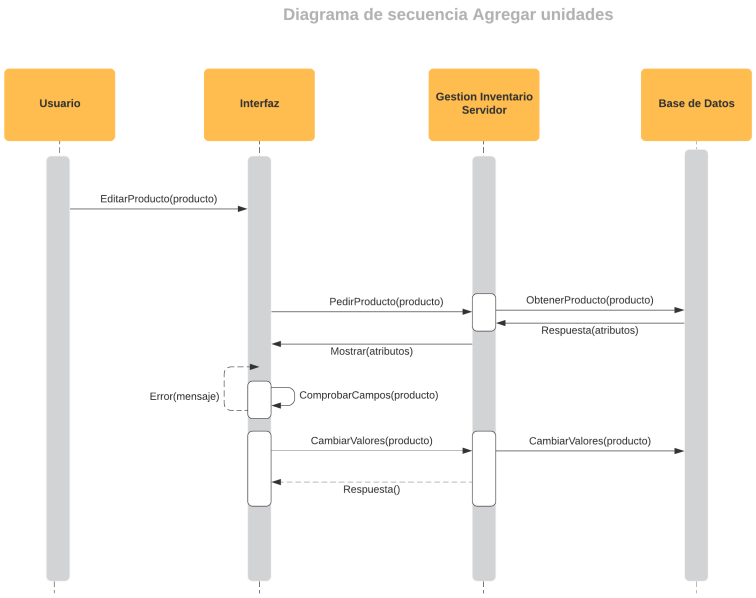


Figura 5.11: Diagrama de secuencia: Editar un producto

5.2.12. Auto-completar lista de la compra

Como vemos en el requisito funcional **RF8**, un usuario puede completar su lista de la compra automáticamente, este simple diagrama muestra el flujo desde que el el usuario pulsa el botón de auto-completar, hasta que el servidor devuelve la lista de productos.

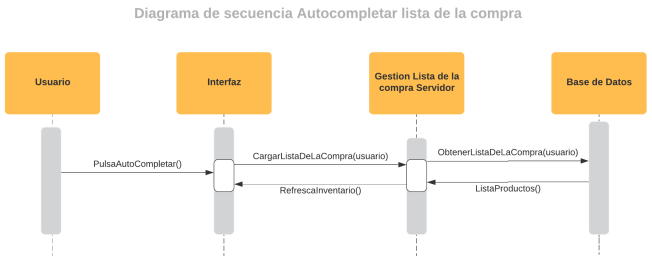


Figura 5.12: Diagrama de secuencia: Auto-completar lista de la compra

5.2.13. Borrar lista de la compra

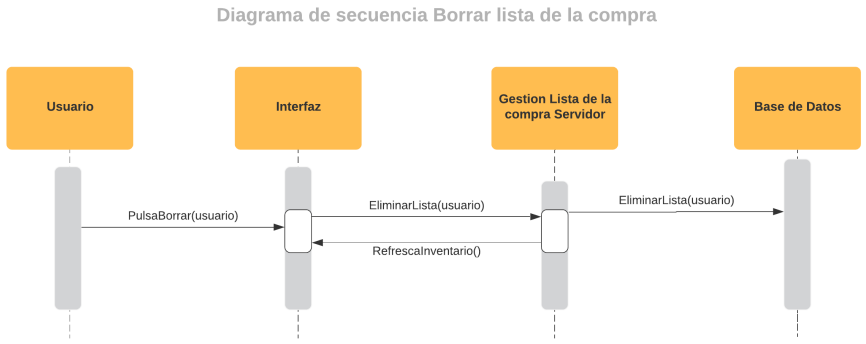


Figura 5.13: Diagrama de secuencia: Borrar lista de la compra

Este diagrama tiene un diseño similar al visto en la figura 5.12, debido a que posee una funcionalidad similar, solo que en este caso elimina todos los productos de la lista de la compra de un usuario como podemos ver en el requisito funcional **RF25**.

5.3. Diseño del back-end

Una vez explicado el diseño que se va a implementar en el cliente es necesario establecer como va a funcionar nuestro servidor.

Para ello estableceremos y comentaremos el diseño de la base de datos de nuestra aplicación, así como los scripts PHP que necesitaremos para comunicar nuestra base de datos con el cliente, mostrando así la información a nuestro usuario

5.3.1. Base de datos

Cuando creamos una base de datos es necesario hacer dos modelos previos: un modelo conceptual y un modelo relacional.

El modelo conceptual muestra la estructura de las tablas, con los campos los nombre de los atributos y el tipo de datos de estos campos. Al igual que también las restricciones, claves primarias de las tablas y las propias relaciones que existen entre tablas.

Este no es un modelo específico de una base de datos, por lo que para profundizar cómo va a ser físicamente la base de datos es necesario hacer uso de un modelo relacional. Con este modelo relacional o también llamado Entidad/Relación, podemos apreciar todas las tablas necesarias para la creación de la base de datos, así como la relación entre las claves primarias y foráneas entre tablas.[17]

Tras explicar en que consisten los dos modelados realizados para su posterior implementación, se procede a explicar la lógica que se ha seguido.

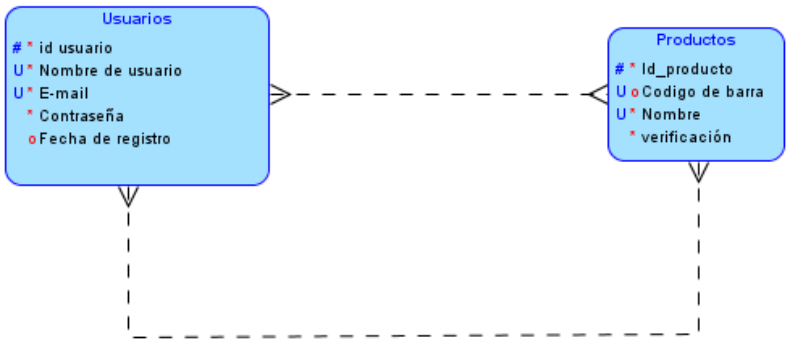


Figura 5.14: Modelo conceptual base de datos

Como vemos en la figura 5.14 correspondiente al modelo conceptual, se crean dos entidades que serán la base de nuestra base de datos.

Por una parte la entidad llamada usuarios, con una clave primaria *id* autogenerable cada vez que se añade un nuevo usuario. Un *nombre de usuario* y un *correo electrónico*, ya que estos serán los atributos correspondientes a la identificación del usuario, es decir, el usuario usará su correo junto a la *contraseña* para realizar su inicio de sesión, obviamente registrándose con anterioridad. Por último la fecha de registro es un campo meramente informativo para el administrador de la base de datos, permitiendo saber así la antigüedad del usuario.

Por otra parte la entidad llamada productos se encargará de almacenar todos los productos que los usuarios agreguen, al igual que la entidad usuarios, se hará uso de un *id* auto generable como clave primaria. Respecto al código de barras, tendrá una restricción de carácter única, de esa forma nos aseguramos que solo un producto tiene asociado un código de barras. Este diseño viene dado por la normativa **GS1** basada en el principio de no ambigüedad que nos dice: «*cada variante de cada producto debe tener un código único. Al código se le conoce como código GTIN, y se trata de una numeración única, universal y sin ambigüedad.*»[18]

La decisión de establecer el nombre como único viene dada por evitar la repe-

tición de nombres en la base de datos. Si un usuario decide agregar un producto llamado *Manzanas*, este producto será agregado teniendo una clave primaria id relacionada con ese nombre. De esta forma, si otro usuario añade a su Despensa otro producto con el mismo nombre no se agregará a la base de datos, si no que se obtendrá el id de ese producto y se relacionará con ese usuario como se verá más adelante, consiguiendo así una menor carga de datos en las tablas de productos.

Finalmente el atributo validez es el que cumple con el requisito funcional **RF6** y su funcionalidad se explicará en los futuros capítulos.

Entre las dos entidades encontramos también dos relaciones, ambas relaciones de muchos a muchos **(N:N)**. Parece una relación redundante pero que entenderemos mucho mejor con el modelo relacional. Resumiendo, una relación nos indica que un usuario puede inventariar muchos productos y un producto puede ser inventariado por muchos usuarios. Esta relación es idéntica para listar los productos en el caso de la funcionalidad de la lista de la compra.

Como hemos dicho, estas relaciones se entenderán mucho mejor con la visión de la figura 5.15 que veremos a continuación.

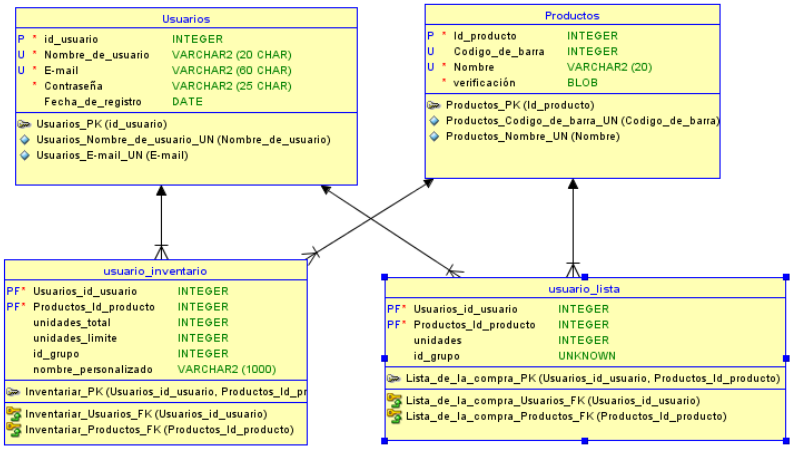


Figura 5.15: Modelo relacional base de datos

Con este modelo apreciamos las mismas dos tablas, al igual que en el modelo conceptual. Además, podemos ver el tipo de datos de los atributos con una longitud estimada que puede variar en la implementación. Sin embargo, hay una gran diferencia, en este modelo, las tablas de usuario y productos no están relacionadas directamente, asimismo se han creado dos nuevas entidades: *usuario inventario* y *usuario lista*.

Estas tablas son creadas automáticamente tras proporcionar el modelo conceptual que hemos comentado en la figura 5.14, sin embargo tienen una impor-

tancia de cara a crear las tablas en la base de datos. Las tablas citadas corresponden al resultado de ambas relaciones **N:N** entre los productos y los usuarios, ambas tablas poseen dos claves foráneas como son Usuarios_id_usuario y Productos_id_producto, las cuáles se relacionan con las claves primarias de las tablas Usuarios y Productos. Básicamente, en estas tablas guardaremos los datos de los inventarios y las listas de la compra de los usuarios con la relación del id_usuario junto al id_producto del producto agregado. No obstante las tablas no almacenan únicamente esa relación, si no también algunos atributos como se cita en el requisito funcional **RF10**.

En el caso de la tabla que almacena los datos del inventario encontramos campos tales como: las unidades totales de un producto, las unidades límite que permiten saber cuando ha de agregarse un producto a la lista automáticamente, el id del grupo en el que el usuario ha decidido guardar ese producto, y el nombre que ha decidido escribir para este. En cambio en la otra tabla simplemente se encuentran las unidades que son necesarias comprar y el id del grupo al que pertenece.

De esta forma obtenemos el diseño de la base de datos a implementar en el servidor, en general es una base de datos muy sencilla, con pocas columnas en las tablas y únicamente con 4 tablas, consiguiendo así que las sentencias SQL, que se realicen en los scripts PHP del servidor, no sean muy complejas y al mismo tiempo rápidas, para una mayor eficacia en el cliente de la aplicación.

5.3.2. Scripts PHP

La implementación del servidor no es algo muy definido, el básico funcionamiento será algo similar a lo siguiente: El cliente hará una petición a uno de los archivos PHP del servidor dependiendo de la funcionalidad que se esté requiriendo. Se puede apreciar un modelo en lo que nos basaremos para la implementación en la figura 5.16.

De esta forma crearemos archivos PHP, organizados por directorio según su funcionalidad, como puede ser **Registrar.php** e **Inciar_sesion.php** que serán los encargados de manejar la conexión de un usuario a la aplicación. O como puede ser **CargarInventario.php** y **AgregarProducto.php** que tendrán la funcionalidad de establecer la comunicación entre las acciones que realice el usuario en su inventario y la base de datos, al igual que su homónimo Lista_de_la_compra.

Quizás no es la forma más convencional de programar el back-end, pero como la aplicación no ha de ser muy compleja, no se esperan muchas dificultades.

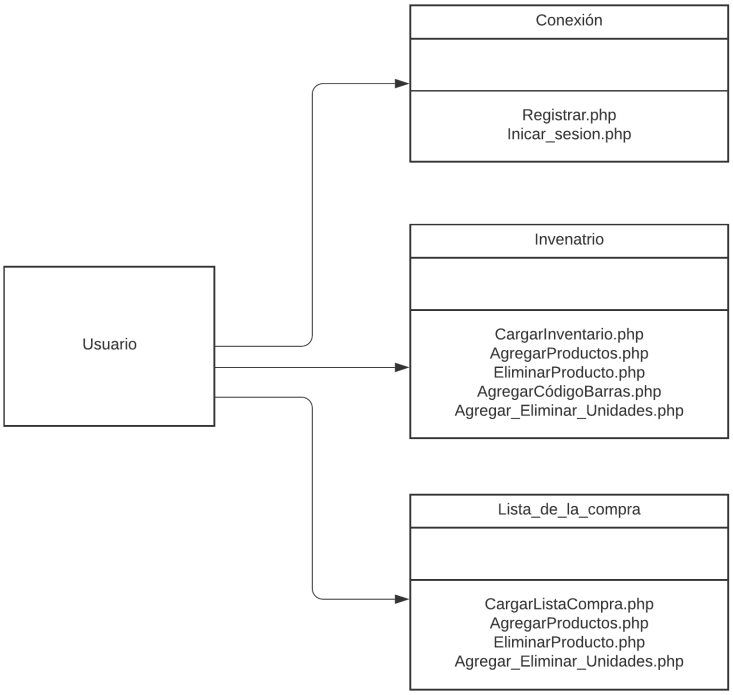


Figura 5.16: Directorio de los archivos PHP

5.4. Implementación

Hablemos ahora de cuál es el resultado de la implementación en lo que respecta a la navegación del front-end, así como la implementación de las diferentes funcionalidades que podemos encontrar tanto en el front-end como en el back-end.

5.4.1. Navegación

Como vemos en la figura 5.17, estas son las clases que se encargaran de la navegación en nuestra aplicación, en el apartado del cliente. Entre estas clases podemos diferenciar dos claros tipos: actividades y fragments. Ambas tienen una función clara, son las encargadas de albergar toda la parte gráfica de la aplicación con los archivos *.xml* correspondientes y las funcionalidades que se encargarán de hacer las respectivas peticiones al back-end. La única diferencia que existen entre los dos tipos de clases es, que las actividades tienen un apartado gráfico propio y por lo general no tienen nada en común entre ellas. Sin embargo, los

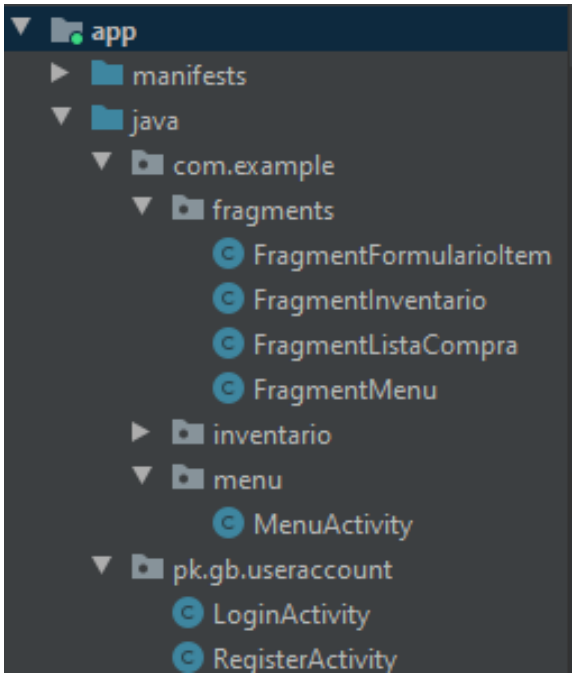


Figura 5.17: Archivos de navegación Java front-end

fragments poseen algunos atributos gráficos en común, en este caso se han usado fragments para mantener una barra superior en cualquier página de la aplicación, en la que se encuentran un botón para volver a la pantalla inicial, que corresponde al **MenuActivity**, y un borón de cerrar sesión, que nos cierra la aplicación y nos permite registrarnos con otro usuario si lo deseamos.

5.4.2. Funcionalidades

La aplicación posee una serie de funcionalidades que se han descrito en los capítulos anteriores, para llevarlas a cabo será necesario de explicar clases y código tanto del front-end como del back-end, así como también las librerías usadas para facilitar el trabajo a la hora de implementar dichas funcionalidades.

En primer lugar, para el front-end será necesario hacer uso de una clase **Producto**, que será la encargada de almacenar los datos de un producto y poder acceder a ellos de forma rápida y cómoda, tras la respuesta del servidor. De esta forma, cuando el usuario acceda a su inventario, se haga una petición una petición al back-end de todos los productos de ese usuario, será mucho más fácil su tratamiento con la clase citada.

Las clases **InentarioAdapter** y **ListadelaCompraAdapter** son las encargadas

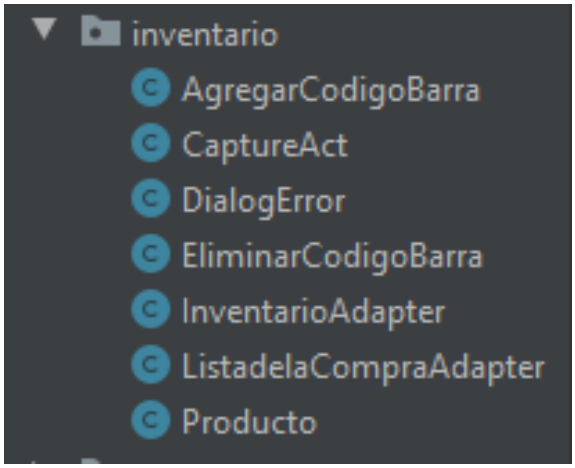


Figura 5.18: Archivos de funcionalidades Java front-end

de poder dividir los productos, del inventario y de la despensa, en los respectivos grupos ya preestablecidos en una lista estática. De esta forma creamos cómo se verán los productos y las diferentes funcionalidades que nos ofrece la etiqueta de un producto en la figura 5.19: agregar y eliminar unidades, editar los atributos y eliminarlo de la lista.

Figura 5.19: Ejemplo de cómo se muestra un producto

DialogError es simplemente una ventana emergente, que proporciona un mensaje según el error en el caso de que se intente hacer algo que la aplicación no permita, como por ejemplo eliminar un producto con el código de barras que no exista en el inventario.

Y finalmente las clases más importantes, del front-end, encargadas de la lectura del código de barras: **AgregarCodigoBarra**, **EliminarCodigoBarra** y **CaptureAct**. Para esta funcionalidad se ha usado una librería llamada **ZXing** con repositorio en *GitHub* para su libre uso.[19]

La clase **AgregarCodigoBarra**, será la encargada de lanzar **CaptureAct** como una nueva actividad, y realizará la lectura del código de barras para obtener el resultado numérico. Una vez realizado el escáner, el resultado es tratado en la primera clase citada, mostrando al usuario en un *dialog*, el nombre, las unidades en despensa, el código de barras y el grupo. La clase **EliminarCodigoBarra** actúa

de la misma forma que la anterior descrita, solo que en este caso es la encargada de eliminar las unidades del inventario, del producto leído.

Una vez analizado como son las funcionalidades que actúan en el front-end, procedamos a analizar las del back-end.

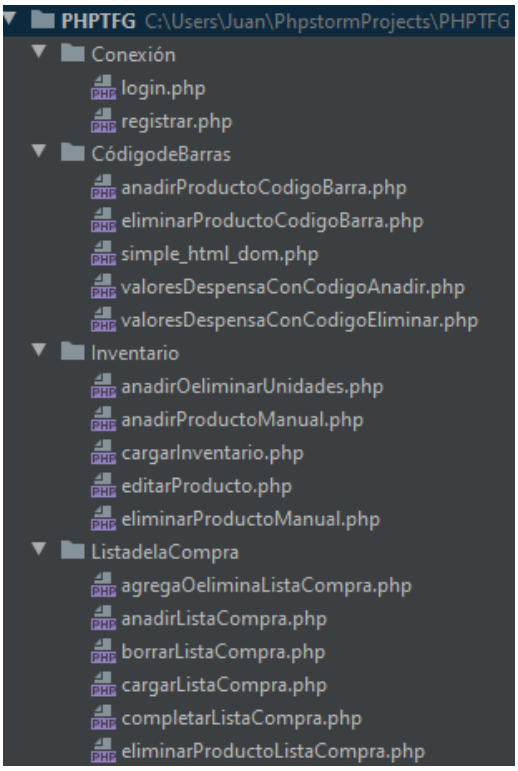


Figura 5.20: Archivos PHP en el back-end

Como se muestra en la figura 5.20, estos son los archivos que se encuentran en el servidor e interactúan con la base de datos. Organizados por directorios, es decir, todos los archivos en lo que respecta a la conexión de un usuario se encuentran en la carpeta **Conexión**, y así con las respectivas carpetas.

Por lo general todos los directorios se encargan de comunicarse con la base de datos, obtener productos para mostrarlo en el inventario o en la lista de la compra, agregar, eliminar o editar productos, así como añadir y eliminar unidades.

Sin embargo hay que profundizar un poco en el directorio **CódigodeBarras**. Los archivos que se encuentran en este directorio se encargan de todas las funcionalidades necesarias para gestionar los códigos de barras de los productos. Pero hay un archivo que se encarga de una de las funcionalidades más importante: el scrapping en Google.

simple_html_dom.php es una librería escrita en PHP que nos permite hacer peticiones a una URL para obtener varios campos deseados del HTML devuelto por la petición.[20] La idea principal de usar esta librería, viene dada para establecer la funcionalidad de autocompletar el nombre de un producto una vez es escaneado por el usuario. De esta forma, como ya hemos dicho en capítulos anteriores, agilizamos el agregar un producto mediante el código de barras, evitando que el usuario tenga que introducir el nombre manualmente, aunque puede hacerlo si lo desea o si el producto no es encontrado. Es decir, el método se basa en hacer una búsqueda en Google con el código de barras leído y obtener la cadena de caracteres que podemos encontrar en la primera entrada del resultado.

Por lo general, como se ha podido comprobar con ensayo prueba y error, la mayoría de los productos son encontrados por este método. Sin embargo hay un margen de error en el que el producto no es encontrado o el nombre es erróneo. Aun así, se ha llegado a la conclusión que es el mejor método, gratuito, para lograr esta funcionalidad. Se trató de usar algunas bases de datos de productos tales como **Barcode Lookup**, pero no se obtenían tan buenos resultados como el método descrito, ya que en la mayoría de los casos el producto no era encontrado.

Una última cosa a tener en cuenta en lo que respecta al código de barras es que cuando el usuario realiza una lectura de un producto, este código de barras es añadido automáticamente la base de datos de productos de nuestra aplicación, sin necesidad de que el usuario lo agregue a su propia despensa, logrando así almacenar los productos, y no volver a hacer el scrapping con ese código de barras, agilizando así un poco el trabajo y aumentando el número de productos en nuestra base de datos.

Como se citó en los requisitos no funcionales, la seguridad de nuestros usuario será algo que tener en cuenta. Por ello, en los archivos **registrar.php** y **login.php** se tratan las contraseña con las funciones que proporciona *Phyton* llamadas *password_hash()* y *password_verfy()* respectivamente. Con ellas podremos codificar mediante el hash las contraseñas de los usuarios, de forma que no tenemos el texto original guardado en la base de datos de la aplicación.

Capítulo 6

Conclusiones y líneas futuras

6.1. Conclusiones

Hemos cumplido todos los puntos que habíamos propuesto, y hemos logrado lo que pretendíamos: conseguir una aplicación rápida y sencilla.

Nos hemos centrado en el diseño bajo la premisa de que fuera lo más simple posible, para que personas adultas que no estén familiarizadas con el mundo tecnológico puedan utilizarla. Además, el código de barras permite tener una rapidez indiscutible en la operativa de manejo de la aplicación; permitiendo que un usuario escanee un producto y únicamente se tenga que preocupar de las unidades que ha comprado y del tipo de producto, ya que el nombre es proporcionado de forma automática, ahorrando así mucho tiempo y trabajo.

La aplicación resultante soluciona un problema concreto –la gestión de compras en el supermercado–, con un nivel de calidad y funcionalidad que ya permite ser utilizada en entornos reales tal y como la hemos creado.

6.2. Lineas futuras

Las líneas por las que se podría seguir el desarrollo serían:

6.2.1. Cruce con bases de datos dietéticas

Sería interesante cruzar la información de la cesta de la compra con bases de datos dietéticas, para tener en el terminal información relativa a macronutrientes

de lo que estamos comprando. También para poder descargar nuestras propias dietas, y que la propia aplicación nos indique qué debemos comprar para seguir una dieta en concreto.

6.2.2. Hacer la aplicación social

Permitir que los usuarios puedan valorar los productos que introducen usando la aplicación, de tal forma que cuando un producto sea escaneado, se pueda compartir la valoración de otros usuarios sobre ese producto, así como establecer un apartado de reseña para el mismo.

6.2.3. Venta de la base de datos

Permitir la venta de información de nuestra base de datos a otras empresas que necesiten, información sobre que productos son los más añadidos a nuestra aplicación. Además de poder crear una API que conecte nuestra base de datos ofreciendo una relación entre el código de barras y el nombre del producto, para los usuarios que lo necesiten. Incluso una mejora en los atributos de un producto, permitiría al usuario especificar características alimenticias tales como: valor energético, hidratos de carbono, sal, grasas, etc... Una información suplementaria para el producto, mejorando así la utilidad de la API.

Apéndice A

Manual de uso para el usuario

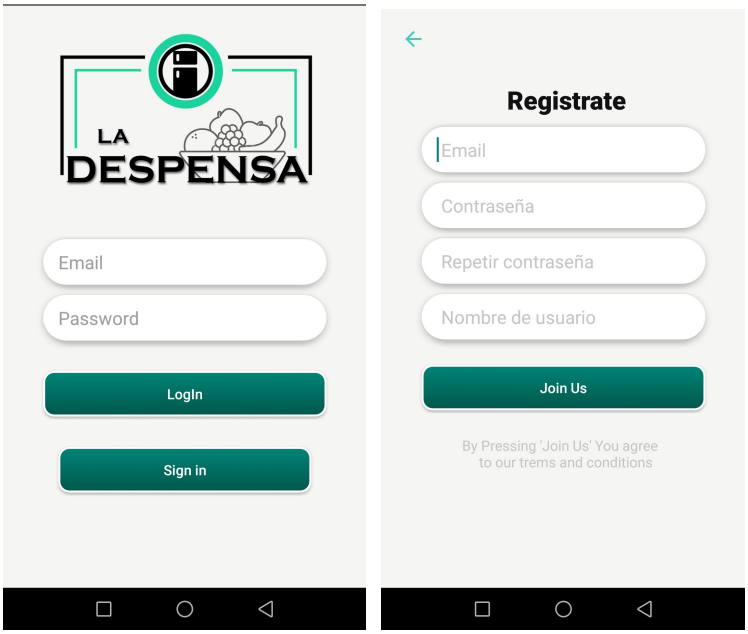
Tras ya haber terminado la aplicación, y haber comentado las lineas futuras y la conclusión de estas, hagamos un recorrido por toda la aplicación, mostrando todas las funcionalidades implementadas y enseñando al usuario cómo utilizarla.

Contenido

A.1. Registro de usuario

Como ya dijimos, lo primero que ha de hacer el usuario es registrarse en nuestra aplicación, para tener acceso a ella.

Como vemos el formulario de la figura A.1b es necesario completar todos los campos que se solicitan con valores correctos, ya que la aplicación los verificará previamente antes de validar al usuario. Si algunos de los campos es erróneo, se le notificará al usuario mediante un mensaje por pantalla. De la misma forma ocurrirá si el registro es correcto.



(a) Incio de sesión (b) Registro

Figura A.1: Registro e inicio de sesión en la aplicación

Será necesario, para que se valide el registro, que el correo electrónico siga el formato establecido, las contraseñas coincidan y el usuario no exista.

A.2. Inicio de sesión

Realizado el registro, el usuario deberá completar el formulario, en este caso el que vemos en la figura A.1a, con su correo electrónico y la contraseña introducidos en el registro. Cuando se verifique que son correctos ambos campos, que no tiene error de formato el correo electrónico y este está asociado con la contraseña, la aplicación dará acceso al usuario al menú principal.

A.3. Menú principal

Cuando el usuario acceda al menú principal verá las principales funcionalidades de la aplicación. Podrá acceder a su despensa para ver los productos que posee en ese momento. Además, podrá también entrar en su lista de la compra, así como agregar productos a La Despensa o eliminar unidades de un producto

que ya exista.

El diseño del menú se ha intentado hacer lo más simple posible, para que así cualquier persona, aun por poco que sepa usar el dispositivo, sea capaz de cuáles son las funcionalidades básicas.

Además, en la parte superior podemos encontrar un botón que nos regresa al menú principal, para agilizar la navegación, y en la parte superior derecha el botón de cerrar sesión.

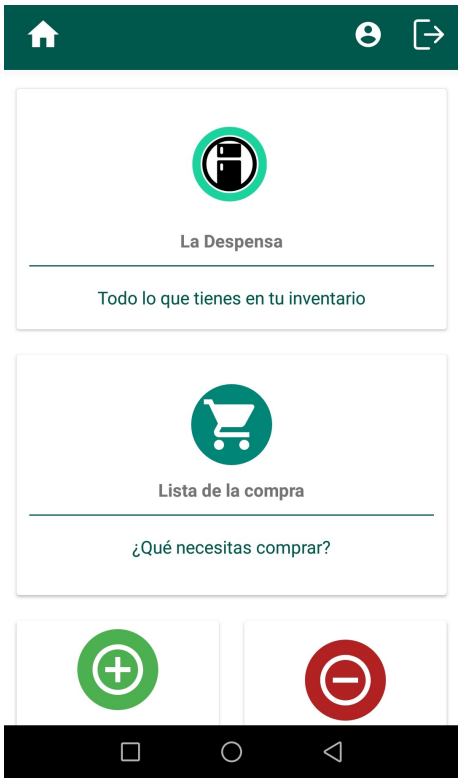
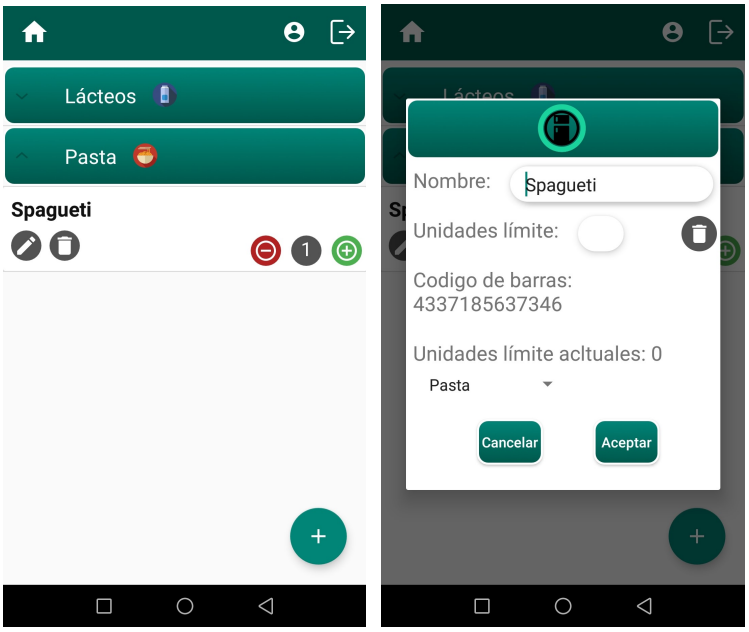


Figura A.2: Menu de la aplicación

A.3.1. La Despensa

Si el usuario pulsa el botón de La Despensa, accederá a su inventario, como podemos ver en el ejemplo de la figura A.3a, está dividido por grupos preestablecidos para su mejor organización. De esta forma el usuario tardará menos en buscar un producto y no tendrá mezclados productos que no tengan correlación entre ellos.



(a) Inventario (b) Editar producto

Figura A.3: Inventario y formulario para añadir productos

Como vemos en la etiqueta de los productos, el usuario podrá agregar y eliminar unidades con los botones verde y rojo respectivamente. El aumento y la disminución de unidades se verá reflejado en el número rodeado de la circunferencia gris.

En esta etiqueta también encontramos dos iconos justo debajo del nombre del producto, el primero de ellos nos abrirá una ventana emergente que permitirá cambiar diferentes atributos como se ve en la figura A.3b. Justo a la derecha de ese icono encontramos el botón de borrar ese producto, de esta forma eliminamos permanentemente el producto de nuestro inventario.

Por último en la parte inferior derecha encontramos un botón flotante que nos redirige a un formulario que se puede ver en la figura A.4. Este formulario nos permitirá añadir productos manualmente, de esta forma no será necesario siempre usar el código de barras y será muy útil cuando añadamos productos que no posean código de barras, como pueden ser los alimentos que se compran por pesaje.

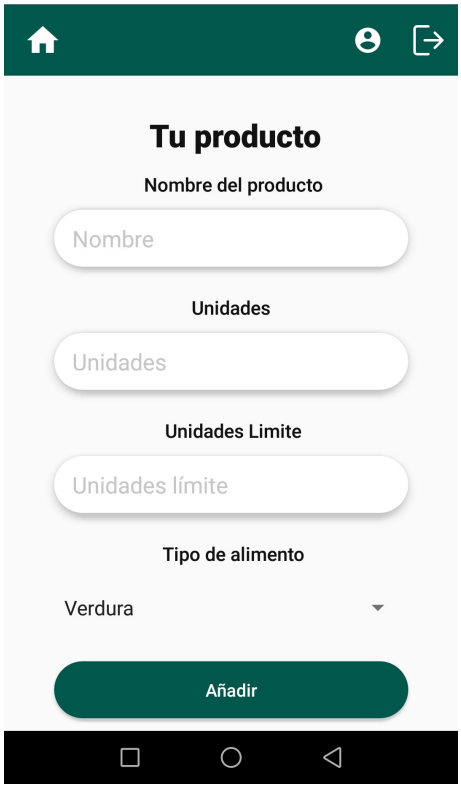


Figura A.4: Formulario para agregar producto

A.3.2. La lista de la compra

Otra de las funcionalidades que nos ofrece el menú principal es poder acceder a la lista de la compra. En principio nuestra lista de la compra estará vacía. Por ello, en la parte inferior derecha, como se aprecia en la figura A.5, podremos encontrar un botón flotante que nos permitirá hacer las siguientes funciones:

Agregar producto

Se podrá agregar un producto a la lista de la compra, mediante un cuadro de dialogo como vemos en la figura A.6 . El usuario deberá completar todos los campos de forma correcta para poder agregar el producto a su lista de la compra.

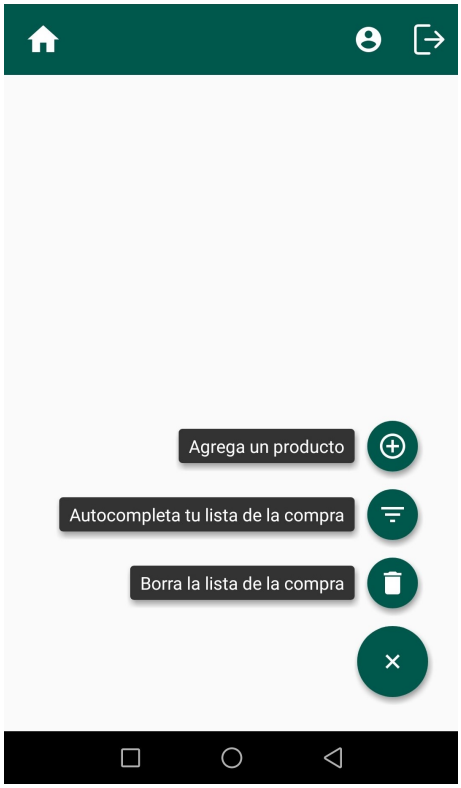


Figura A.5: Lista de la compra

Auto-completar lista de la compra

Otra de las funcionalidades que nos presenta ese botón flotante es la posibilidad de completar la lista de la compra automáticamente. Cuando el usuario pulsa el botón de editar de un producto se abre un cuadro de dialogo como hemos visto en la figura A.3b. Podemos apreciar que uno de los campos a completar es las unidades límites. Es decir, un usuario puede establecer un límite y cuando ese producto se encuentre por debajo en unidades significará que es necesario comprar ese producto.

Por ello, cuando el usuario utilice la funcionalidad de auto-completar la lista de la compra, añadirá todos los productos que se encuentren por debajo de su límite establecido, siendo las unidades que son necesarias para alcanzar ese límite, las que aparecerán en la lista de la compra, para que el usuario las compre.

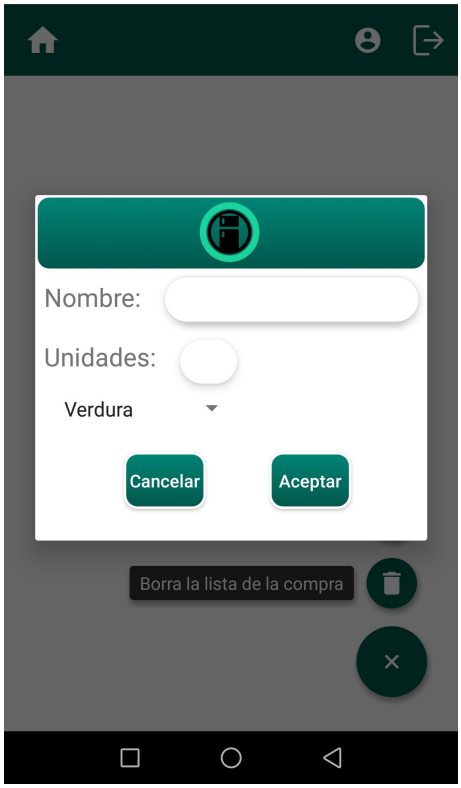


Figura A.6: Ventana emergente para agregar un producto

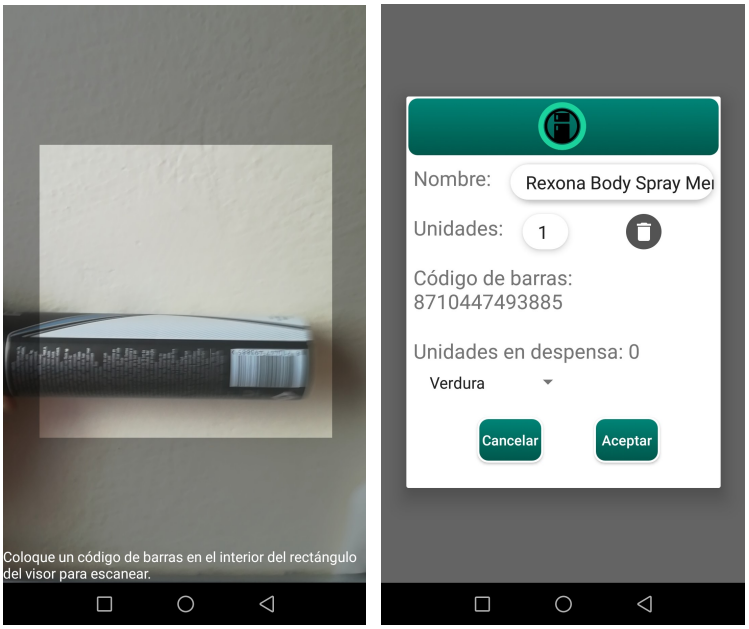
Borrar lista de la compra

Por último, otra funcionalidad básica que podemos hacer en nuestra lista de la compra es borrar completamente. De esta forma, cuando un usuario vaya al supermercado y compre todos los productos que le aparecen en la lista de la compra, no tendrá que eliminarlos uno por uno, un solo botón y la lista de la compra estará vacía

A.3.3. Agregar con código de barras

Por último en esta guía para el usuario analizaremos la funcionalidad que nos ofrece el escaneo del código de barras. El usuario, como se ve en la figura A.7a, escaneará el producto deseado y posteriormente como se ve en A.7b, se presentará en una ventana emergente los datos del producto escaneado.

En este caso, como vemos en el resultado, el producto no está añadido a nuestra Despensa, ya que hay 0 unidades en despensa. Predeterminadamente, apare-



(a) Escáner (b) Resultado de la lectura

Figura A.7: Escáner y resultado del código de barras

cerá un nombre en el caso de que sea encontrado en la base de datos de nuestros productos o sea encontrado con el scrapping en Google. Siempre podrá haber un margen de error en el que el campo del nombre se encuentre vacío.

Logramos así que al pasar un producto por el escáner, obtengamos el nombre completo y las unidades a añadir estarán predeterminadas en 1, ya que será lo más normal que un usuario agregue una sola unidad.

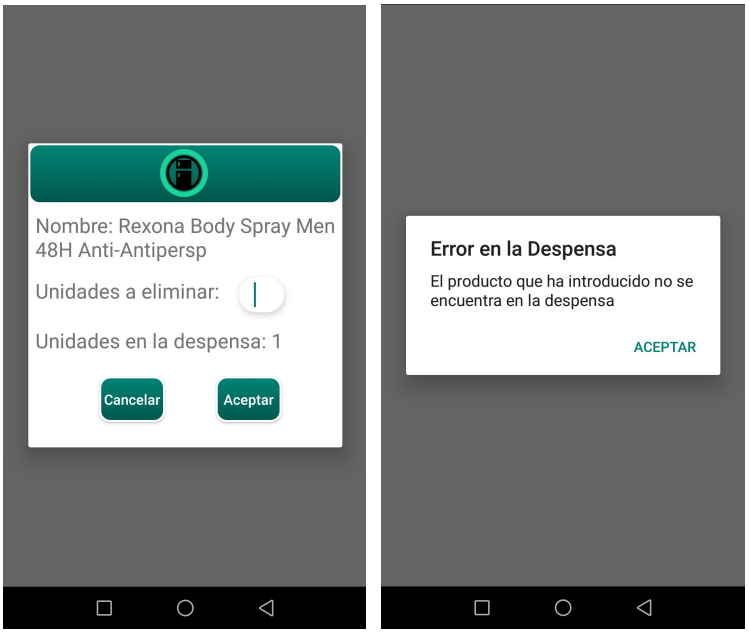
Por último antes de aceptar, el usuario podrá elegir en que grupo desea agregar el producto, para ello hará uso del desplegable que vemos justo encima del botón cancelar.

A.3.4. Eliminar con código de barras

Y para terminar, el código de barras también podrá ser usado para eliminar unidades de un producto.

Cuando se realice el escáner del producto que deseemos eliminar podrán pasar dos cosas:

El primer caso que puede ocurrir es que todo sea correcto, como se ve en la figura A.8a. Se nos muestra un cuadro de dialogo con el nombre del producto



(a) Resultado de la lectura para eliminar

(b) Error

Figura A.8: Resultado y error al eliminar con el código de barras

a eliminar y justo abajo, el número de unidades que queremos eliminar y las unidades totales que tenemos en la despensa. Aquí la aplicación controlará que el usuario no pueda eliminar más unidades de las que tiene en total, mostrando un mensaje en caso de error. Si todo es correcto y el usuario pulsa aceptar, se eliminarán esas unidades del inventario.

Y el segundo caso se daría si el usuario intenta eliminar un objeto que no se encuentra en su despensa, por lo que se mostraría un mensaje de error como se puede aprecia en la figura A.8b.

Bibliografía

[1] Nancy Molina. *¿Qué es el estado del arte?*, 2005. researchgate.net/publication/317162163_Que_es_el_estado_del_arte, Consultado el 10 de Abril de 2020.

[2] JOANA SÁNCHEZ. *En España, 12,3 millones personas se encuentran en riesgo de pobreza o exclusión social*, 2019. https://www.pymesyautonomos.com/actualidad/espana-12-3-millones-personas-se-encuentran-riesgo-pobreza-exclusion-social, Consultado el 10 de Abril de 2020.

[3] Susannah Snider. *How to Save Money When Grocery Shopping on a Budget*, 2019. https://money.usnews.com/money/personal-finance/spending/slideshows/how-to-save-money-when-grocery-shopping-on-a-budget?slide=17, Consultado el 12 de Abril de 2020.

[4] Susannah Snider. *How to Save Money When Grocery Shopping on a Budget*, 2019. https://money.usnews.com/money/personal-finance/spending/slideshows/how-to-save-money-when-grocery-shopping-on-a-budget?slide=23, Consultado el 12 de Abril de 2020.

[5] Android Studio. *Android Studio developer*. https://developer.android.com/studio, Consultado el 20 de Abril de 2020.

[6] Visual Studio. *Visual Studio IDE*. https://visualstudio.microsoft.com/es/, Consultado el 20 de Abril de 2020.

[7] *Java*. https://www.java.com/es/, Consultado el 20 de Abril de 2020.

[8] Real Academia de la Lengua Española. *Servidor*. https://dle.rae.es/servidor, Consultado el 12 de Abril de 2020.

[9] MySQL. *MYSQL SGBD*. https://www.mysql.com/, Consultado el 20 de Abril de 2020.

[10] *PHP*. https://www.php.net/manual/es/intro-whatis.php, Consultado el 20 de Abril de 2020.

[11] *phpMyAdmin*. https://www.phpmyadmin.net/, Consultado el 21 de Abril de 2020.

[12] *Download Bitvise SSH Clien*. https://www.bitvise.com/ssh-client-download, Consultado el 21 de Abril de 2020.

[13] *Gliffy*. https://www.gliffy.com/, Consultado el 21 de Abril de 2020.

[14] *Unified Modeling Language (UML)*. https://en.wikipedia.org/wiki/Unified_Modeling_Language, Consultado el 21 de Abril de 2020.

[15] *LaTeX*. https://www.latex-project.org/, Consultado el 21 de Abril de 2020.

[16] John Díaz. *¿Qué es BACKEND y FRONTEND?*, 2020. https://ed.team/blog/que-es-backend-y-frontend-guia-completa, Consultado el 3 de Mayo de 2020.

[17] campusMVP. *Diseñando una base de datos en el modelo relacional*, 2014. https://www.campusmvp.es/recursos/post/Disenando-una-base-de-datos-en-el-modelo-relacional.aspx, Consultado el 21 de Abril de 2020.

[18] latiendadelasbarras. *Códigos de barras: ¿qué información contiene el código de barras?*, 2019. https://www.latiendadelasbarras.com/barras/informacion-codigo-de-barras/, Revisado 4 de Septiembre 2020.

[19] rkistner. https://github.com/journeyapps/zxing-android-embedded, Consultado el 25 de Mayo de 2020.

[20] S.C. Chen. *PHP Simple HTML DOM Parser*. https://simplehtmldom.sourceforge.io/, Consultado el 1 de Junio de 2020.

UNIVERSIDAD DE MÁLAGA | uma.es

E.T.S de Ingeniería Informática
Bulevar Louis Pasteur, 35
Campus de Teatinos
29071 Málaga

E.T.S. DE INGENIERÍA INFORMÁTICA