



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA



UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Identificación de patrones de viabilidad para un negocio delivery en el turismo de ocio marítimo, Desarrollo de una aplicación de soporte para el negocio.

Trabajo Fin de Grado

Grado en Ingeniería Informática

AUTOR/A: Senent Alfonso, David

Tutor/a: Conesa García, María Pilar

CURSO ACADÉMICO: 2021/2022

Resumen

El presente Trabajo de Fin de Grado de Ingeniería Informática se basa en el desarrollo de una aplicación dirigida a dispositivos Android para la empresa EnBarco. Dicha aplicación consiste en un software que permite el reparto de productos a determinadas localizaciones, siendo estas puntos marítimos cercanos a la costa, es decir, una aplicación de *delivery* marítimo.

El proceso de desarrollo de este proyecto se ha desarrollado en dos fases bien distinguidas. Primeramente, el diseño basado en el usuario y prestando especial atención a la simplicidad y facilidad de uso. En segundo lugar, se ha implementado y desarrollado la aplicación usando tecnologías como Kotlin, Java, XML y *Firestore Storage*, además de los servicios de *Google Maps* y la API de *Firestore*, necesarios tanto para la geolocalización como para la autenticación de credenciales.

A lo largo del proyecto se han realizado numerosas pruebas y una evaluación de los resultados de satisfacción de los usuarios con el producto diseñado. Los resultados de estas pruebas y evaluación indican un alto grado de satisfacción por parte de los usuarios y un correcto funcionamiento de la aplicación.

Palabras clave: Aplicación Móvil, Geolocalización, *Delivery*, Desarrollo Ágil, Diseño Centrado en el Usuario, Usabilidad, Ingeniería del Software

Abstract

This Computer Engineering Final Degree Project is based on the development of an application for Android devices for the company EnBarco. This application consists of a software that allows the delivery of products to certain locations, being these maritime points near the coast, that is to say, a maritime delivery application.

The development process of this project has been developed in two distinct phases. Firstly, the design based on the user, and paying special attention to simplicity and ease of use. Secondly, the application has been implemented and developed using technologies such as Kotlin, Java, XML and Firebase Storage, in addition to Google Maps services and Firebase API, necessary for both geolocation and credential authentication.

Throughout the project, numerous tests and an evaluation of the results of user satisfaction with the designed product have been carried out. The results of these tests and evaluation indicate a high degree of user satisfaction and a correct functioning of the application.

Keywords: Mobile Application, Geolocation, Delivery, Agile Development, User-Centered Design, Usability, Software Engineering

Dedicatoria:

Todas las personas grandes han sido niños antes, pero pocas lo recuerdan (El principito).

Más que una dedicatoria, aspiro a que esto sirva como recordatorio a todas las personas que están iniciándose en cualquier campo, al igual que a quienes son expertos. Hace ocho años descubrí la informática, un campo que despertó mi interés al instante. Me resultó muy llamativo todo lo que se podía crear desde un ordenador, pero en unos instantes me di cuenta de que utilizarlos no era sencillo y me iba a costar trabajo hacerme con ellos. Hace cuatro años aprendí a programar, y tuve una sensación similar al ver la inmensidad de posibilidades que se abrían ante mí. De la misma forma, uno de mis primeros pensamientos fue el que nunca sería capaz de programar algo tan complejo como una aplicación o programa como los que yo usaba.

Mientras escribo estas palabras, y tras un largo camino de aprendizaje, echo la vista atrás y me doy cuenta de la inocencia de ese niño, frente al complejo y a la vez delicado ordenador, opacado por el amplio abanico de caminos que se abrían delante de él. Yo y muchas otras personas hemos estado en la misma situación, encontrar algo que te apasiona y sentirte abrumado por el reducido conocimiento que tienes acerca de ello.

Espero que estas palabras sirvan como motivación no solo a todos esos niños, que ansían convertirse en personas grandes, sino también para recordar a todas las personas grandes lo que lograron, ya que una vez, ellos también fueron niños.

Agradecimientos:

Quiero agradecer en primer lugar a mi abuela, quien me ha brindado apoyo durante toda mi vida. También a mis padres y a mi hermano, que me han acompañado y ayudado durante todo mi desarrollo académico y profesional. Mencionar de la misma forma a mi novia y mis amigos, quienes han transformado este camino en una colección de momentos inolvidables y siempre me han dado fuerza para seguir adelante, pero en especial, quiero agradecerme a mí mismo por enfrentarme y superar todos los obstáculos que han aparecido en mi camino. Siempre ha sido mi elección el camino a seguir, y tenga o no alguien respaldándome, siempre escogeré avanzar.

Se puede optar por retroceder hacia la seguridad o avanzar hacia el crecimiento. El crecimiento debe elegirse una y otra vez; el miedo debe ser superado una y otra vez. (Abraham Maslow)



Índice de contenidos

Introducción	11
1.1. Justificación del trabajo.....	11
1.2. Motivación.....	12
1.3. Objetivos.....	13
1.4. Impacto esperado	14
1.5. Metodología.....	14
1.6. Estructura.....	15
1.7. Colaboraciones	16
1.8. Convenciones	16
2. Estado del arte.....	17
2.1. Android	17
2.2. Versiones del Sistema Operativo	17
2.3. Análisis del sector	20
2.4. Por qué es mejor EnBarco	21
2.5. Crítica al estado del arte	22
3. Análisis del problema.....	23
3.1. Análisis de la seguridad	23
3.2. Análisis energético o de eficiencia de algorítmica	25
3.3. Análisis del marco legal y ético	26
3.4. Ética	26
3.5. Análisis económico	27
3.6. Solución propuesta	28
3.7. Plan de trabajo	29
3.8. Presupuesto	29
3.8.1. Costes humanos	29
3.8.2. Costes materiales	30
4. Diseño de la solución	31
4.1. Arquitectura del sistema.....	31
4.2. Entorno de desarrollo	31
4.3. Lenguaje de programación	32
4.4. API de Android	33

4.4.1.	Activities	33
4.4.2.	Layouts y ficheros XML.....	34
4.4.3.	Toolbar	36
4.4.4.	Intents.....	37
4.4.5.	El fichero manifest.xml	38
4.4.6.	Sensor GPS	40
4.4.7.	Mensajes Toast	40
4.5.	Diseño detallado	42
4.5.1.	Diseño de pantallas	42
4.5.2.	Diseño de la interfaz	44
4.5.3.	Diseño de la base de datos.....	48
4.5.4.	Diseño de scripts	49
5.	Desarrollo de la solución propuesta.....	51
6.	Implantación	53
7.	Pruebas.....	54
7.1.	Pruebas realizadas por los desarrolladores	54
7.1.1.	Pruebas unitarias.....	54
7.1.2.	Pruebas de integración	54
7.1.3.	Pruebas funcionales.....	55
7.2.	Test de usabilidad realizado	55
8.	Aplicación final.....	57
8.1.	Login	57
8.2.	Pantallas de registro de empresa	62
8.3.	Pantalla principal de la aplicación	64
8.4.	Menú del usuario	64
8.5.	Modificar información usuario	65
8.6.	Productos favoritos.....	67
8.7.	Historial de pedidos.....	69
8.8.	Resultado de búsqueda	70
8.9.	Carrito de la compra	71
8.10.	Menú de la empresa	72
8.11.	Modificar datos de la empresa	72
8.12.	Historial de pedido de la empresa	73
8.13.	Listado de productos de la empresa.....	73
8.14.	Añadir producto nuevo	74
8.15.	Modificar producto	75



9.	Conclusiones	76
9.1.	Problemas encontrados y como se han solucionado	76
9.2.	Errores cometidos y como evitarlos.....	77
9.3.	Aprendizaje del alumno tanto profesional como personalmente.....	78
10.	Trabajos futuros	79
10.1.	Posibles flecos interesantes de desarrollar, pero que no ha sido posible	79
10.2.	Líneas de desarrollo abiertas para aplicar estos resultados a otras áreas ...	80
10.3.	Mejoras y ampliaciones tanto de eficiencia como de funcionalidades del trabajo realizado	81
11.	Bibliografía	83
ANEXOS		86
POLÍTICA DE PRIVACIDAD		87
OBJETIVOS DE DESARROLLO SOSTENIBLE.....		92
ANÁLISIS DE REQUISITOS		95
TEST DE USABILIDAD.....		98

Índice de ilustraciones

<i>Ilustración 1 - Android</i>	17
<i>Ilustración 2 - Gráfico de porcentaje de versiones</i>	19
<i>Ilustración 3 - Cifrado de contraseñas con base 64</i>	24
<i>Ilustración 4 - Algoritmo para calcular distancias</i>	26
<i>Ilustración 5 - Logotipo de Android Studio</i>	32
<i>Ilustración 6 - Logotipos de Kotlin y Java</i>	33
<i>Ilustración 7 - Ejemplo de relación entre Activity y layout</i>	34
<i>Ilustración 8 - Ejemplo de Adapter de productos</i>	35
<i>Ilustración 9 - Muestra de Toolbar de usuario 1</i>	36
<i>Ilustración 10 - Muestra de Toolbar de usuario 2</i>	37
<i>Ilustración 11 - Ejemplo de Intents</i>	38
<i>Ilustración 12 - Fichero Manifest.xml</i>	39
<i>Ilustración 13 - Mensaje Toast en el código</i>	41
<i>Ilustración 14 - Mensaje Toast ejecutado</i>	41
<i>Ilustración 15 - Diagrama de flujo</i>	43
<i>Ilustración 16 - Interfaz 1</i>	44
<i>Ilustración 17 - Interfaz 2</i>	46
<i>Ilustración 18 - Interfaz 3</i>	48
<i>Ilustración 19 - Login</i>	58
<i>Ilustración 20 - Registro</i>	61
<i>Ilustración 21 - Registro de empresa</i>	63
<i>Ilustración 22 - Pantalla principal</i>	64
<i>Ilustración 23 - Perfil de usuario</i>	64
<i>Ilustración 24 - Modificar datos usuario</i>	67
<i>Ilustración 25 - Productos favoritos</i>	68
<i>Ilustración 26 - Historial de pedidos</i>	69
<i>Ilustración 27 - Resultados de búsqueda</i>	70
<i>Ilustración 28 - Carrito de productos</i>	71
<i>Ilustración 29 - Menú de empresa</i>	72
<i>Ilustración 30 - Modificar datos empresa</i>	72
<i>Ilustración 31 - Historial de pedidos de empresa</i>	73
<i>Ilustración 32 - Lista de productos de empresa</i>	73
<i>Ilustración 33 - Añadir producto empresa</i>	74
<i>Ilustración 34 - Modificar producto empresa</i>	75



Índice de tablas

<i>Tabla 1 – Tabla de versiones</i>	<i>19</i>
<i>Tabla 2 - Costes de envío</i>	<i>28</i>

Introducción

Durante los últimos años, los dispositivos móviles se han convertido en herramientas indispensables en el día a día de cualquier persona debido a su comodidad, eficiencia y facilidad de uso. Esto ha propiciado que los avances en las tecnologías móviles nos hagan ver imposibles de afrontar tareas tan mundanas como una conversación entre dos personas o la adquisición de algún producto de forma online sin su ayuda. Cada día aparecen nuevas aplicaciones que nos ofrecen formas de realizar tareas ordinarias y diarias desde la palma de nuestra mano.

Es innegable el hecho que los beneficios son mucho mayores que los inconvenientes. Esta tecnología ha contribuido a optimizar nuestro día a día en prácticamente cualquier aspecto, desde acceder a la información deseada hasta la compra de cualquier producto de manera inmediata, simple y desde cualquier lugar.

1.1. Justificación del trabajo

Debido a la pandemia provocada por el Covid-19, se incrementó la necesidad de diseñar nuevos modelos de negocio y aplicaciones para dispositivos móviles con el fin de encontrar formas de entregar productos a domicilio. Esto ha provocado la aparición de nuevas empresas que buscan suplir esta necesidad. Se ha requerido que estas empresas se adapten a nuevas maneras de consumo y evolucionen para poder proveer sus servicios y productos sin los inconvenientes que ha provocado la pandemia. A lo largo de esta pandemia, las compras por Internet, al igual que su uso, se han incrementado notablemente, de forma que han surgido muchos nuevos usuarios y ha propiciado un mayor grado de familiarización con estas compras online y, por tanto, su demanda. Esto supone un reto para el sector, ya que el *delivery* tendrá un papel fundamental para poder saciar estas nuevas necesidades, brindando mayor personalización y rapidez a los negocios de este estilo.

Teniendo en cuenta que gran parte del consumo de productos y servicios se realiza ahora mismo de forma online, y que esto ha aumentado notablemente con la pandemia, podemos deducir que durante el futuro próximo esto tendrá una importancia inmensa y será un requisito indispensable en cualquier empresa. La gran mayoría de los consumidores han aumentado su grado de familiarización acerca de las herramientas digitales, muchas de las cuales ya han sido incluidas en su vida cotidiana, modificando así sus hábitos de consumo. Los nuevos consumidores cuentan con Internet como su mayor aliado para obtener toda la información necesaria. Esta



metamorfosis digital ha originado la aparición de nuevos modelos de negocio que reinterpretan el concepto de beneficio, situándose del lado de la innovación y apoyando a las personas como las verdaderas creadoras de valor. Todos, como personas, tenemos necesidades y gustos distintos, por lo que es necesaria la existencia de muchos proveedores que nos puedan satisfacer. Actualmente, existen innumerables negocios aun por descubrir, y esto impulsa a emprendedores a modificar o transformar conceptos de negocio, como el *delivery*, a otros terrenos.

Todas estas empresas incorporan plataformas que convierten a la realidad la necesidad de entregar y modificar las entregas de sus productos. De la misma forma, las plataformas actuales se han visto abrumadas y han sufrido la misma necesidad de evolucionar. Muchas han modificado la forma en que manejan las entregas o han necesitado reorganizar todo su esquema de funcionamiento.

El presente Trabajo Final de Grado busca dotar de un software competente a la innovadora empresa EnBarco, brindando a esta la opción de incluirse en el disputado mercado. Esta empresa busca crear un modelo de negocio basado en la entrega de productos a embarcaciones fondeadas, es decir, un *delivery* marítimo. Para que esta empresa pueda rendir en su máxima eficiencia, necesita un sector software que le permita cumplir con todas las entregas que requieran realizar. La aplicación móvil de EnBarco espera funcionar de forma sencilla e intuitiva a la vez que eficiente, de forma que pueda satisfacer las necesidades de todos los clientes de EnBarco al mismo tiempo que permitirle aumentar su alcance y competir en el sector marítimo.

1.2. Motivación

La elaboración del presente Trabajo Final de Grado me consiente culminar mis estudios en el Grado de Ingeniería Informática y gracias a ello iniciar, al fin, una carrera profesional en el mundo del desarrollo de software, que tanto interés despierta en mí.

Previo al estudio de esta carrera, ya me apasionaba la posibilidad de poder desarrollar aplicaciones y/o programas de calidad que todo el mundo pudiese utilizar, además de los ordenadores y todo lo que los envuelve.

Durante toda la experiencia universitaria y laboral que he ido desarrollando a través del estudio de esta carrera, sin perder la pasión por la informática, ha despertado en mí un interés relacionado con el mundo de la administración y gestión de empresas. Esto me ha llevado a ponerme en contacto con María Pilar Conesa García (tutora del TFG),

quien me instruyó en primero de carrera en la asignatura “Fundamentos de organización de empresas”, y a quien le trasladé la idea de realizar un TFG que uniese ambos mundos. De esta manera, María me puso en contacto con Alejandro Sánchez Gómez (estudiante de Ingeniería de la Organización Industrial y fundador de la empresa EnBarco) y me propuso la creación y desarrollo del software de su aplicación como tema para el TFG. Además, podría formar parte del trabajo de Alejandro, adquiriendo así nociones acerca de la creación y gestión de una empresa como lo es EnBarco.

Al cursar la rama de Ingeniería del Software, no me llevo mucho tiempo darme cuenta de que estaba perfectamente dotado para cumplir esta propuesta, además de que saciaba mis intereses. Uniendo las conclusiones previamente mencionadas, nos encontramos en el punto de inicio idóneo para el desarrollo de este Trabajo Final de Grado.

1.3. Objetivos

Cómo principal objetivo en el desarrollo de la aplicación de *delivery* marítimo EnBarco, tenemos el proveer de un software accesible para todos los públicos, sencillo de usar y con gran nivel de eficiencia a la empresa EnBarco. Tanto el nivel de importancia de esta empresa como su cercana relación con este Trabajo Final de Grado vendrá detallado en el punto 1.6 Colaboraciones.

Otro de los objetivos principales a los que aspira este proyecto es la percepción y forma de llegar a los usuarios, por lo que necesitamos contar no solo con una interfaz amigable, sino también con una gran facilidad de uso, que nos permita brindarles una experiencia sencilla y liviana.

La seguridad es otro de los objetivos a los que aspiramos con esta aplicación, otorgando una confianza y sensación de protección hacia los usuarios.

También tenemos como objetivo ajustar el desarrollo del software a las necesidades de la empresa y a las de sus usuarios finales. Puesto que el principal objetivo es dotar de software a la empresa EnBarco, este software debe ajustarse a las necesidades de la empresa y adaptarse a sus cambios y evolución. De la misma forma, debe satisfacer a los usuarios finales independientemente de la situación en la que nos encontremos, teniendo la adaptabilidad como uno de los principales valores.



Cómo último de estos objetivos destacaremos el desafío que conlleva realizar un proyecto software, cómo lo es el desarrollo de una aplicación móvil, desde cero, planificando todas y cada una de sus etapas, desde el estudio y análisis, hasta las pruebas requeridas, pasando por la codificación y evaluación de los resultados. Para esto es necesaria una planificación minuciosa del tiempo y recursos, además llevar a la práctica todas las habilidades y conocimientos alcanzados durante toda la carrera.

1.4. Impacto esperado

Debido a que nuestro objetivo principal es el de dotar a la empresa EnBarco de un software de calidad, el impacto inicial esperado es el de permitir a dicha empresa un ingreso en el mercado con un software al nivel de la competencia.

Una vez EnBarco se consolide en el mercado, se espera que las diferencias que marca nuestro software con respecto al de la competencia, nos permitan escalar cuota de mercado y aumentar la audiencia.

Ya que actualmente no existe una competencia directa en el sector del *delivery* marítimo, esperamos poder saciar una necesidad sin el inconveniente de compartir cuota de mercado. Aspiramos a no solo ser la primera empresa en este sector, sino a ser líder del mercado en el sector durante los próximos años. Aunque nuestra aplicación cumple a la perfección su cometido y rivaliza con las de la competencia, somos conscientes de que también tiene muchos aspectos a mejorar, en especial cuando esta se integre uniformemente en el mercado y consiga un número considerable de usuarios. Algunas de estas posibles mejoras se verán en el capítulo 10. Trabajos futuros.

1.5. Metodología

Después de un estudio acerca de las distintas metodologías existentes y gracias a la puesta a prueba de estas a lo largo de la carrera, se ha escogido para este proyecto el uso de una metodología ágil, la cual nos brinda una temprana visión del resultado final y nos ofrece la posibilidad de realizar cambios a lo largo del desarrollo si estos son necesarios [1].

Este desarrollo se realizará en 6 *Sprints* de dos semanas cada uno, centrándonos en cada uno de estos en un apartado distinto de la aplicación, pero sin dejar el resto de lado. Profundizaremos más en las fases de realización del proyecto en el punto 3.7 Plan de trabajo.

1.6. Estructura

La memoria está estructurada mediante los siguientes capítulos:

En el Capítulo 1, el capítulo en el que se encuentra esta sección, se introducen los motivos que han llevado al desarrollo de este TFG, los objetivos que se pretenden seguir, el impacto esperado y las colaboraciones realizadas.

En el Capítulo 2 se describe el estado del arte de las técnicas que se han empleado a lo largo de este proyecto, se analiza el mercado y las aplicaciones actualmente existentes que cumplen propósitos idénticos o similares, y se exponen las ventajas en el uso de EnBarco frente al resto de alternativas disponibles.

En el Capítulo 3 se procede al análisis del problema desde distintas perspectivas, a la identificación de posibles soluciones, se precisa el plan de trabajo y se indica el presupuesto.

En el Capítulo 4 se pretende diseñar la solución al problema propuesto. Busca definir la arquitectura del sistema, detallar la solución y mostrar la tecnología utilizada.

En el capítulo 5 se desarrolla la solución propuesta. Aquí es donde mostraremos a grandes rasgos el código y cómo se ha desarrollado, los problemas encontrados durante este desarrollo y como se han solventado estos problemas.

En el capítulo 6 se explica la implantación de la aplicación en el mundo real. Desde cómo realizar la instalación, hasta la como se pretende implantar en la empresa y, desde esta, en el mundo real.

En el capítulo 7 se muestran las pruebas y test realizados para constatar la correcta funcionalidad de la aplicación.

En el capítulo 8 se elaboran las conclusiones por parte del estudiante acerca del desarrollo de este proyecto.

En el capítulo 9 se proponen posibles mejoras o líneas de trabajo que quedan abiertas tras la finalización de este TFG. También se indican caminos que no seguir para evitar cometer errores y el porqué.

Tras estos capítulos se muestran las referencias bibliográficas utilizadas para la realización de este trabajo y la redacción de esta memoria, además de los anexos que completan la información descrita previamente.

1.7. Colaboraciones

Como se ha mencionado anteriormente, el presente Trabajo Final de Grado es, en parte, una colaboración con el TFG realizado por Alejandro Sánchez Gómez. Aunque no es un TFG conjunto, ya que no se implica directamente en el desarrollo del proyecto, este trabajo tiene como pilar fundamental el recientemente mencionado, pues consiste en el diseño, creación y gestión de la hipotética empresa EnBarco.

Gran parte de los datos y valores usados en el proyecto vienen dados por parte de la empresa EnBarco, además del nombre, logotipo de la aplicación, precios y funcionamiento.

1.8. Convenciones

Este apartado simplemente tiene la función de especificar el formato de texto utilizado a lo largo de esta memoria para clarificar su lectura. Estas convenciones no se aplicarán a los anexos, tratándose estos de documentos distintos.

Se utilizará Arial como fuente para todo el cuerpo del texto y Trebuchet MS como fuente para títulos y subtítulos.

Todo el cuerpo del texto está escrito en tamaño 11.

Los títulos y subtítulos de los capítulos se escribirán en negrita y serán de tamaño 14 y 13 respectivamente.

Siempre que se utilice alguna palabra en inglés u otro idioma se escribirá en cursiva.

También se utilizará cursiva siempre que se haga referencia a una imagen o tabla de esta misma memoria.

2. Estado del arte

2.1. Android



Ilustración 1 - Android

Android es un sistema operativo móvil basado principalmente en el kernel de Linux. Este fue diseñado específicamente pensando en dispositivos móviles dotados de pantalla táctil, como *smartphones*, tabletas o *smartwatches*. Fue desarrollado por Android Inc. junto al respaldo económico de la conocida empresa Google, quien posteriormente adquirió la empresa en 2005.

Entre los aspectos determinantes del sistema operativo, podemos encontrar como uno de los principales su orientación multiplataforma, una cualidad realmente novedosa que propulsó su uso a nivel mundial, dado que anteriormente los sistemas operativos estaban diseñados específicamente para un único dispositivo. Gracias a esto, además de su facilidad de uso y alta eficiencia, lo han convertido en el sistema operativo más utilizado actualmente.

2.2. Versiones del Sistema Operativo

Android se encuentra en una continua evolución desde el lanzamiento de su versión inicial. La primera actualización del sistema operativo fue lanzada en febrero de 2009 por Google. Esta se diferenciaba de la versión anterior por brindar a los usuarios la posibilidad de adjuntar archivos en los mensajes. Echando la vista atrás, si lo comparásemos con una versión reciente, entenderíamos la inconmensurable evolución del sistema operativo. Cada una de las siguientes actualizaciones ha añadido funcionalidades nuevas u optimizado las ya existentes.

Todas y cada una de las versiones de Android, hasta la versión 10, ha sido nombrada haciendo referencia a distintos postres, así mismo, en orden alfabético.

- a) A: *Apple Pie* (v1.0): tarta de manzana.
- b) B: *Banana Bread* (v1.1): pan de plátano.
- c) C: *Cupcake* (v1.5): Magdalena.
- d) D: *Donut* (v1.6): Rosquilla.
- e) E: *Éclair* (v2.0/v2.1): Pastel francés.
- f) F: *Froyo* (v2.2) (abreviatura de «frozen yogurt»): Yogur helado.
- g) G: *Gingerbread* (v2.3): Pan de jengibre.
- h) H: *Honeycomb* (v3.0/v3.1/v3.2): Panal de miel.
- i) I: *Ice Cream Sandwich* (v4.0): Sándwich de helado.
- j) J: *Jelly Bean* (v4.1/v4.2/v4.3): Gominola.
- k) K: *KitKat* (v4.4): Tableta de chocolate con leche.
- l) L: *Lollipop* (5.0): Piruleta
- m) M: *Marshmallow* (6.0): Malvavisco.
- n) N: *Nougat* (7.0): Turrón.
- o) O: *Oreo* (8.0): Oreo.
- p) P: *Pie* (9.0): Tarta.
- q) Android 10 (10.0).
- r) Android 11 (11.0).
- s) Android 12 (12.0).

En la actualidad del mercado, coexisten muchas de las versiones mencionadas anteriormente. Esto es a causa de que un gran número de dispositivos no se han actualizado a las últimas versiones. En la *Tabla 1* podemos observar los porcentajes de dispositivos móviles que usan las distintas versiones.

Versión	Nombre	Porcentaje
4.1-4.3	Jelly Bean	0.4%
4.4	KitKat	1.4%
5.0-5.1	Lollipop	3.9%
6.0	Marshmallow	5.1%
7.0-7.1	Nougat	6.3%
8.0-8.1	Oreo	13.7%
9.0	Pie	18.2%
10.0	Android 10	26.5%
11.0	Android 11	24.3%

Tabla 1 – Tabla de versiones

O también podemos observar el mismo porcentaje en forma de gráfico en la ilustración 1:

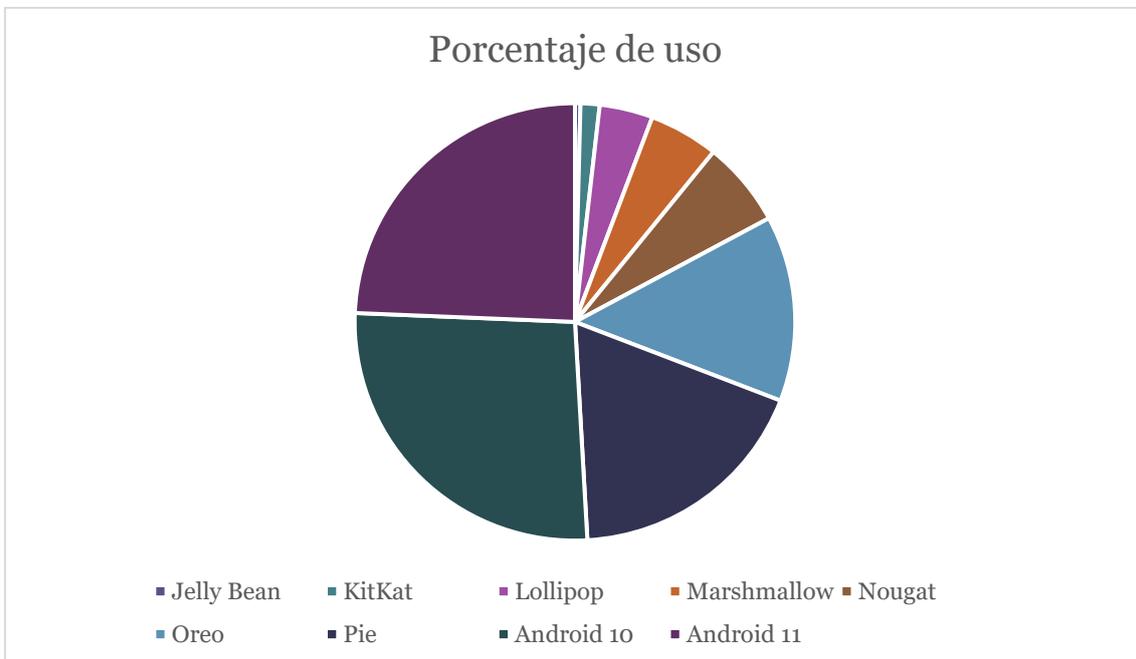


Ilustración 2 - Gráfico de porcentaje de versiones

Actualmente Android, como hemos mencionado anteriormente, se sitúa como el sistema operativo más usado a lo largo del mundo, y teniendo en cuenta el estudio realizado sobre el uso de sus versiones, la mayoría de los usuarios utilizan la versión *Android Oreo* o superiores, siendo solo un 17% del mercado quien utiliza versiones anteriores a la *Oreo* [2]. Sumando esto al público objetivo de la empresa EnBarco, hemos decidido utilizar *Android Oreo* en como versión de nuestro sistema operativo en este proyecto.

2.3. Análisis del sector

En los últimos años se ha incrementado la influencia de las plataformas de entrega de productos o *delivery* en España, lo que se traduce en una duplicación del número de clientes finales. La aparición de las plataformas *delivery* ha aumentado el rango de opciones para los clientes finales, brindándoles la posibilidad de acceder simultáneamente a un número mucho mayor de servicios de alimentación. Recientemente el modelo de plataforma digital se ha vuelto omnipresente en muchas actividades económicas. La capacidad de la tecnología para convertir el intercambio y distribución de bienes y servicios y el almacenamiento y procesamiento de información a bajo costo, sumado al uso generalizado del Internet móvil, han creado una situación favorable para la creación e implementación de estos nuevos e innovadores productos. Durante la pandemia de COVID-19 se han creado nuevas costumbres y hábitos, los cuales se están arraigando al día a día de los propietarios de restaurantes y consumidores. La innovación y digitalización están en el centro de toda esta transformación y seguirán revolucionando el sector. Esta digitalización se ha convertido en un componente esencial del modelo de negocio de los restaurantes. Desde que los establecimientos reabrieron tras cerrar en 2020, el reparto de los productos a domicilio ha mantenido un ritmo constante. El modelo digital del restaurante es una respuesta a esta creciente necesidad. La innovación de la entrega ha despertado un gran desafío para las marcas en el panorama digital disruptivo. Es imperdonable, hoy más que nunca, el llegar tarde. Todas las empresas se exigen el máximo para poder brindar a los clientes un servicio de entrega unido a una sobresaliente experiencia de compra. El mercado de la comida a domicilio en España tiene menos competencia desde julio del año 2021, cuando uno de los gigantes del sector, Deliveroo, dejó de estar activo en el país. Después de seis años como líder en

el sector, esta compañía de origen británico ha dejado un vacío, llenado rápidamente por la competencia, desde que anunció su intención de abandonar el mercado español.

En España, los proveedores de servicios de comida a domicilio en línea son sumamente populares. Según los datos de la Encuesta Global de Consumidores de Statista, en 2021, el 50% de los usuarios de internet españoles dijeron utilizar estos servicios. El líder en este mercado es Telepizza. Indican que el 49% de las personas que han pedido comida a domicilio online en el último año han utilizado este servicio. Just Eat (48%), Glovo (42%), Domino's (34%), Uber Eats (25%) y el anteriormente nombrado y actualmente discontinuado Deliveroo(24%) son plataformas de entrega a domicilio online.

2.4. Por qué es mejor EnBarco

La aplicación de la empresa EnBarco no intenta realmente ofrecer una experiencia distinta a las previamente indicadas, ya que, como hemos podido observar, estas han tenido éxito en el mercado. Sin embargo, EnBarco no busca ser similar a la actual oferta de aplicaciones *delivery*, sino ser una versión mejorada de estas, ofreciendo un servicio con una facilidad de uso óptima, sencillez a la hora de navegar a través de las distintas pantallas de la aplicación. EnBarco pretende brindar a cualquier usuario, con o sin experiencia en el uso de dispositivos móviles, la capacidad de poder operar de forma intuitiva y eficiente en el momento de realizar cualquier pedido.

Además de esto, al tratarse de una empresa en sus inicios, opta por ofrecer la opción a las empresas de crearse una cuenta como tales, con la posibilidad de ofrecer sus servicios sin necesidad de contactar directamente con nosotros y, una vez más, de la forma más simple posible evitando cualquier complicación innecesaria.

En resumen, EnBarco aspira a ser una versión sumamente simplificada e intuitiva, a la vez que eficiente, de las aplicaciones software que dan soporte a las empresas actualmente exitosas.



2.5. Crítica al estado del arte

De la misma forma, después de un estudio acerca de trabajos finales previamente realizados, no solo de informática, sino relacionados de cualquier forma posible con el tópico, este Trabajo Final de Grado anhela ser una versión mejorada de los mismos.

Si bien la mayoría de estos exponen correctamente el tema debatido respectivamente y argumentan por qué su proyecto funciona de una forma correcta, esta memoria además de esto pretende instruir a cualquier persona sin nociones de informática en el diseño y desarrollo de una aplicación de reparto, pretende de la misma forma mostrar el uso real de una aplicación, en este caso dotando de software a una empresa, e intenta, además de todo lo indicado anteriormente, exponer no solamente las mejoras respecto al resto de aplicaciones en el mercado, sino las debilidades de la misma y proponer soluciones a estas.

3. Análisis del problema

Dando un salto atrás en el tiempo, con la aparición de *Nevera Roja* y *Sin Delantal*, se empezó a digitalizar el proceso de venta online de comida a domicilio. Así apareció el primer directorio de alimentación a domicilio en el que el usuario final tenía la posibilidad de escoger la comida deseada entre un gran rango de restaurantes y agilizar la compra con unos simples clics. Todos ellos desde una misma web.

Desde el origen de *Globo* y la entrada en el sector de *Deliveroo* o *Take Eat Easy* a España, el modelo *delivery* evolucionó. Así inició una revolución tecnológica motivada principalmente gracias a los *marketplaces*, se desarrollaron las primeras plataformas basadas en su totalidad en aplicaciones móviles.

El cliente tenía acceso a una infinidad de establecimientos desde la palma de su mano. Estas plataformas se apropiaron de datos que desde páginas web sería mucho más complicado conseguir, entre ellos los modos de uso, preferencias de gusto, edad y una infinidad de datos que determinan el perfil de los clientes.

Debido a esto, los usuarios han dejado de ser de los restaurantes, pasando a ser clientes de las plataformas de reparto detrás de estos. Sin embargo, atraer clientes a estas plataformas requiere un alto coste que no ha dejado de aumentar, creando así la necesidad de aumentar las comisiones de servicios a los restaurantes para poder encontrar rentabilidad.

Aquí es donde encontramos uno de los mayores problemas de este modelo *delivery*. En el presente capítulo recopilaremos distintos problemas encontrados después de un exhaustivo análisis del sector, exponiendo sus orígenes y ofreciendo soluciones. El objetivo es poder nutrir al software de *EnBarco* de una mejora significativa respecto al resto de la competencia, paliando, o incluso eliminando, los mayores problemas que afectan a una gran parte del sector.

3.1. Análisis de la seguridad

Echando la vista atrás, podemos encontrar problemas respecto a la seguridad en los sistemas software, no solo en el sector de la comida a domicilio, sino en todos los casos y ámbitos posibles.

Desde el gigante *Facebook* y sus conocidos juicios causados por filtraciones en la seguridad y venta de datos de sus usuarios, hasta el hackeo o robo de datos a empresas menores, propiciadas por una seguridad insuficiente, normalmente por no prestar la suficiente atención a este apartado.

Para evadir este problema, hemos decidido como paliativo principal usar la API *Firebase* de Google. Esta API provee muchos servicios útiles y utilizados en este proyecto, pero dos de los principales son la persistencia de los datos y la autenticación. Respecto al apartado de la persistencia, ante la necesidad de guardar cualquier dato relacionado con los usuarios, sean estos clientes o empresas, se ha usado *Firebase Realtime Database*. Esta es una base de datos asíncrona ofertada por Google, quien también se encarga de su seguridad. La segunda, autenticación, también ha sido solventada mayormente por esta API, quien nos brinda la librería *Firebase Auth*, librería utilizada para confirmar la autenticación de credenciales en el *Login*, la cual protege al mismo tiempo al usuario mientras tenga la sesión iniciada.

En adición y con el objetivo de evitar cualquier problema, aun si ocurriese una filtración por parte de Google, las contraseñas se filtran automáticamente al introducirlas en la aplicación, por lo que aun si alguien tuviese acceso a la base de datos protegida por Google, la contraseña encontrada no correspondería con la original y si intentase entrar se le mostraría obviamente el mensaje *Toast* indicando el error.

En la *Ilustración 3* se puede observar la simplicidad del código escrito para cifrar y descifrar cadenas de texto utilizando el codificador en base 64.

```
private fun cifrar(password:String): String{
    // encode a string using Base64 encoder
    val encoder: Base64.Encoder = Base64.getEncoder()
    val encoded: String = encoder.encodeToString(password.toByteArray())
    println("Encoded Data: $encoded")
    return encoded
}

private fun descifrar(password:String): String {
    // decode the encoded data
    val decoder: Base64.Decoder = Base64.getDecoder()
    val decoded = String(decoder.decode(password))
    println("Decoded Data: $decoded")
    return decoded
}
```

Ilustración 3 - Cifrado de contraseñas con base 64

3.2. Análisis energético o de eficiencia de algorítmica

De la misma forma, aunque no implique una importante diferencia en la aplicación, se ha intentado optimizar todos los algoritmos empleados de la mejor forma posible. En especial, hemos hecho hincapié en todas las conexiones con la base de datos, ya que al tratarse de dispositivos móviles, muchos de ellos disponen de un hilo principal que no debe ser obstruido. A diferencia de los ordenadores, donde pueden trabajar varios hilos simultáneamente sin problema, los dispositivos móviles disponen de un hilo de suma importancia, el cual debe evitar ser bloqueado a toda costa.

Con objetivo de solucionar esto, hemos utilizado una base de datos asíncrona, es decir, al hacer una petición a la base de datos, el código sigue siendo ejecutado sin esperar a la respuesta. Esto ha significado una dificultad añadida debido a que nunca había utilizado una base de datos de este tipo, pero al mismo tiempo ha resultado en una aplicación con mayor grado de productividad y menor riesgo de errores.

Siguiendo el mismo objetivo, todas las consultas recuperan todos los objetos de un determinado tipo de la base de datos y comparan, asíncronamente y desde nuestro dispositivo, cuál es el objeto determinado que pretendemos encontrar. Esto ayuda a una rápida recuperación de los datos, evitando problemas relacionados con la velocidad o calidad de la conexión a Internet, además de, una vez más, evitar bloquear el hilo principal.

De igual modo, también cabe destacar el cálculo de distancia entre el usuario y la empresa a la cual se desea realizar el pedido. Para este cálculo se utiliza la posición indicada por la empresa en el registro y la ubicación actual del usuario, la cual se consigue en el momento de realizar la búsqueda de productos, para que solo se le ofrezcan los que se encuentran a una distancia adecuada.

Este algoritmo se ha diseñado para calcular la distancia entre dos puntos en Kilómetros de la forma más eficiente posible a través de cálculos matemáticos, como podemos observar en la *Ilustración 4*.



```
private static double calculateDistance(double lat1, double lon1, double lat2, double lon2) {
    if ((lat1 == lat2) && (lon1 == lon2)) {
        return 0;
    }
    else {
        double theta = lon1 - lon2;
        double dist = Math.sin(Math.toRadians(lat1)) * Math.sin(Math.toRadians(lat2)) +
            Math.cos(Math.toRadians(lat1)) * Math.cos(Math.toRadians(lat2)) * Math.cos(Math.toRadians(theta));
        dist = Math.acos(dist);
        dist = Math.toDegrees(dist);
        dist = dist * 60 * 1.1515;

        dist = dist * 1.609344;

        return (dist);
    }
}
```

Ilustración 4 - Algoritmo para calcular distancias

Posteriormente, simplemente utilizamos la distancia calculada para mostrar los productos que se encuentren al alcance y calcular el precio de envío de estos.

3.3. Análisis del marco legal y ético

Para cumplir el marco legal en el uso de esta aplicación software es necesario cumplir varios requisitos.

Primeramente, es necesario aceptar la Política de Privacidad para poder registrar una cuenta. Es esta aceptamos que se recojan y manejen nuestros datos con fines de cumplir los pedidos realizados en la aplicación. Podemos consultar dicha Política de datos y privacidad en su respectivo anexo.

Sumado a esto, se pide permiso al usuario tanto para acceder a la galería del teléfono, en caso de que sea necesario, como a la ubicación al iniciar sesión en la aplicación, con el fin de mostrar únicamente los productos que se encuentren al alcance del usuario.

3.4. Ética

Respecto a la ética, somos conscientes que, aunque cumplamos los requisitos establecidos en el marco legal, como desarrolladores debemos cumplir también unos estándares éticos y morales. Con la intención de respetar estos estándares, recogemos la mínima información necesaria para poder garantizar el correcto

funcionamiento. Más allá de la ley, todos los programadores que manejamos información personal de usuarios tenemos la posibilidad de hacer uso indebido de esta, corriendo siempre el riesgo de que nos desenmascaren. Considerando esto, ningún desarrollador capacitado debería hacer ni proclamar un uso indebido de esta información personal.

En lo que respecta al equipo de desarrollo de EnBarco, nos comprometemos a no utilizar más información de la necesaria. Además de esto, nos hacemos totalmente responsables en el caso de que ocurriese cualquier tipo de filtración de datos en nuestro sistema y, de la misma forma, haremos todo lo que esté en nuestra mano para evitar este escenario.

Concluiremos este apartado citando una célebre frase del compositor estadounidense Lester Raymond Brown: *“Acepta la responsabilidad de tu vida. Debes saber que eres tú el que te llevará a donde quieres ir, no hay nadie más”*.

3.5. Análisis económico

Respecto al sector económico, y como hemos mencionado anteriormente, uno de los mayores problemas del sector del *delivery* radica en el sobrecoste que le perciben las plataformas del sector a los restaurantes con los que trabajan.

En EnBarco, la metodología es totalmente distinta respecto al resto de plataformas. Si una empresa quiere aparecer en nuestras listas, esta debe descargar la aplicación y darse de alta como empresa. Una vez iniciada la sesión como cuenta empresarial, simplemente debe añadir los productos que ofrece y organizarlos según su tipo. Gracias a este sistema de captación de empresas, no se les cobra nada a los establecimientos por prestar sus servicios, de forma que estos se sientan atraídos a incluirse en nuestro sistema.

Nuestros beneficios se alcanzan mediante el cobro a los usuarios de los envíos de los respetivos envíos. El importe para deducir se calcula según la distancia que se deba recorrer para entregar el pedido, como podemos observar en la *Tabla 2*.



DISTANCIA(KM)	0,5	1	1,5	2	2,5	>2,5
TARIFA VARIABLE	0,75€	1,50€	3,25€	4,00€	4,75€	5,00€
TARIFA FIJA	15,00€	15,00€	15,00€	15,00€	15,00€	15,00€
TARIFA DEL SERVICIO	15,75€	16,50€	18,25€	19,00€	19,75€	20,00€

Tabla 2 - Costes de envío

La tarifa fija supondrá la mayor parte del servicio. Hemos establecido este precio, después de analizar servicios similares y la opinión de los consumidores.

3.6. Solución propuesta

La solución propuesta a los inconvenientes encontrados ha sido desarrollada a lo largo de este capítulo, pero vamos a realizar un breve resumen en este apartado.

Como primer problema encontramos la seguridad. Hemos cifrado la información delicada, la cual podría ser problemática tener sin su correspondiente seguridad. También nos hemos respaldado en la API de Google para optar a un nivel de seguridad en la persistencia de datos, el cual nos sería imposible conseguir por nuestra cuenta.

También nos encontramos el problema de la eficiencia energética o algorítmica. Este ha sido resuelto mediante una programación correcta y medida, haciendo hincapié en reducir el trabajo por parte de la capa de persistencia y no obstruir el principal hilo de los dispositivos Android.

El problema del marco legal ha sido paliado, tras un estudio de este campo, mediante una correcta política de privacidad, notificaciones *pop up* advirtiendo a los usuarios de la información requerida y un correcto cifrado de datos. Respecto al apartado ético de este problema, no existe una solución efectiva en el 100% de los casos. Teniendo eso en mente, nos comprometemos a realizar un correcto uso de la información y no ejercer ninguna mala praxis.

Finalmente, respecto al sector económico, se ha optado por no cobrar comisiones a los restaurantes, sino recaudar las ganancias exclusivamente a través de los envíos de los pedidos que se realicen en nuestra aplicación.

3.7. Plan de trabajo

El plan de trabajo utilizado ha sido registrado en la página www.trello.com.

Esta se trata de una herramienta la cual nos proporciona un tablero Kanban. En este tablero se han creado las columnas “Unidades sin confirmar”, “Unidades confirmadas”, “En proceso”, “Pruebas”, “Unidades finalizadas”, además de una columna por cada *Sprint* realizado. En estas columnas se han organizado las unidades temáticas planteadas y desarrolladas a lo largo de proyecto, permitiéndonos conocer así su orden, nivel de prioridad, estado actual y previsión del trabajo por realizar.

También nos ha sido útil para poder medir las horas trabajadas durante todo el proyecto, además de las que hubiésemos invertido si cumpliésemos todas las funcionalidades pensadas a lo largo del desarrollo.

3.8. Presupuesto

Ahora desarrollaremos el presupuesto que nos ha costado desarrollar un proyecto de esta magnitud, teniendo en cuenta los costes humanos, es decir, el salario de los programadores, y los costes materiales, entre los cuales se encuentran las licencias, programas y bases de datos utilizados.

3.8.1. Costes humanos

Dentro los costes humanos encontraremos únicamente nuestro salario, ya que nos hemos encargado tanto de la gestión de la base de datos, la programación de la aplicación y la realización de pruebas, como de su futuro mantenimiento.

Como hemos mencionado en el apartado anterior, gracias a nuestra herramienta Trello, hemos podido conocer la cantidad de horas que nos ha costado desarrollar este proyecto. A lo largo de todos *Sprints* hemos registrado alrededor de unas 180 horas de proyecto, teniendo en cuenta el diseño, programación y realización de pruebas. Si incluyésemos el tiempo empleado en el estudio de tecnologías para este desarrollo, también registrado en Trello, tendríamos alrededor de 200 horas de trabajo.



Teniendo en cuenta el hecho de que el sueldo base promedio de un desarrollador Android en España es de 32.906€, esto nos da un sueldo promedio por hora de 17,43€. Multiplicando este sueldo promedio por horas por la cantidad de horas trabajadas, siendo estas 180 y omitiendo el tiempo empleado en el estudio de las tecnologías, tendríamos unos costes humanos de 3.137,40€.

3.8.2. Costes materiales

Además de los costes humanos, debemos incluir los costes materiales asumidos a lo largo del desarrollo de este proyecto.

En este caso, como hemos mencionado anteriormente, la mayoría de las licencias utilizadas han sido sin coste alguno, ya que esta era una de nuestras prioridades.

La licencia del programa de desarrollo, *Android Studio*, es gratuita. De la misma forma, la licencia de la base de datos empleada y de nuestro servicio de *Google Maps* también son gratuitas. Si en el futuro evolucionamos a un mejor escenario de la empresa, con un mayor número de usuarios, como mencionaremos en el capítulo 10, sería necesario adquirir una licencia de pago de esta base de datos para aumentar su capacidad.

El coste de publicar la aplicación en *Google Play Store* es de 25 dólares [3], es decir, alrededor de 20€. Además de esto, deberemos de proporcionarles el 30% de la facturación total de la venta de aplicaciones que se tengan disponibles en la tienda.

En conclusión, el coste material en el caso de esta aplicación ha sido un poco mayor a 20€ ya que únicamente debemos pagar por la licencia de *Google Play Store*.

4. Diseño de la solución

En el presente capítulo se explica el proceso de diseño de la aplicación EnBarco. El diseño proporciona una idea completa del software desarrollado en el proyecto. Asimismo, se justifican las decisiones escogidas para el futuro desarrollo. Con esto en mente, el capítulo se divide en 4.1 Arquitectura del sistema, 4.2 Entorno de desarrollo, 4.3 Lenguaje de programación, 4.4 API de Android, 4.5 Diseño detallado. En el punto 4.4 podemos encontrar 4.4.1 *Activities*, 4.4.2 *Layouts* y ficheros XML, 4.4.3 *Toolbar*, 4.4.4 *Intents*, 4.4.5 Fichero manifest.xml, 4.4.6 Sensor GPS y 4.4.7 Mensajes *Toast*. Dentro del punto 4.5, este se subdivide en 4.5.1 Diseño de pantallas, 4.5.2 Diseño de la interfaz, 4.5.3 Diseño de la base de datos, 4.5.4 Diseño de scripts.

El diseño del proyecto ha sufrido cambios leves a lo largo del desarrollo, por lo que, aunque se pretende hacer hincapié en las versiones finales, se mostrarán y se hará referencia a versiones anteriores, además de la evolución del diagrama de flujo de la aplicación. Dichas modificaciones han venido motivadas por: Pruebas no satisfactorias, no satisfacción de los requisitos, adición de algún requisito, adquisición de nuevos conocimientos y habilidades [4].

4.1. Arquitectura del sistema

La aplicación se sirve de una arquitectura que sigue el modelo **cliente – servidor**. Como cliente encontramos a la propia aplicación Android mientras que, por el otro, el servidor se corresponde a una base de datos *Realtime Database* proporcionada por los servicios de la API de Google que ofrece soporte para la persistencia de datos.

Este punto pretende describir el software utilizado para el desarrollo del proyecto. Es necesario puntualizar que, desde el inicio del proyecto, se ha procurado el uso de software libre (*opensource*) debido al hecho de no tener que pagar licencias, la versatilidad que este nos brinda y en especial, por buscar soporte en tecnologías que actualmente se encuentran extendidas y demandadas en el mercado laboral de hoy en día, lo cual es un estímulo a la hora de realizar este TFG.

4.2. Entorno de desarrollo

Respecto a la plataforma de desarrollo, hemos escogido *Android Studio*. Este fue el primer candidato propuesto y, sin duda, una de las mejores opciones posibles. *Android*



Studio nos brinda un amplio abanico de opciones de programación, sencillez de organización, interfaces claras y personalizables, fácil acceso a cualquier parte del código, resulta sumamente intuitivo incluir librerías y el uso de emuladores desde este, además de la gran cantidad de recursos sobre su funcionamiento que existen en internet.



El desarrollo de interfaces también ha sido realizado desde esta misma plataforma. Todos los *layout* de la aplicación final han sido programados en XML desde utilizando las opciones nativas de *Android Studio*.

Ilustración 5 - Logotipo de Android Studio

De la misma forma, el emulador usado a lo largo de todo el desarrollo de la aplicación ha sido el AVD (*Android Virtual Devices*), emulador nativo de la plataforma *Android Studio*. Este nos proporciona no solamente con todas las funcionalidades de un dispositivo real, sino además con la opción de emular el GPS del dispositivo en cualquier parte del mundo, funcionalidad apreciada para realizar pruebas en una aplicación como esta, que requiere de pruebas en distintas ubicaciones. Además, también nos brinda la opción de modificar desde el dispositivo hasta la versión de la imagen del sistema. De la misma forma, esto nos permite realizar pruebas en distintos dispositivos con cualquier versión de Android y distintos tamaños de pantalla.

4.3. Lenguaje de programación

Cómo lenguaje de programación, aprovechándonos de la versatilidad que nos ofrece *Android Studio*, hemos utilizado tanto Kotlin como Java. Mientras que en la mayoría de los trabajos finales únicamente se utiliza un lenguaje de programación, en las empresas en el mundo real coexisten múltiples lenguajes para poder explotar los beneficios de cada uno. De la misma forma, en este proyecto he decidido utilizar tanto Kotlin, para nutrirnos de su simpleza, velocidad de programación y aprendizaje, y sencillez para escribir y leer código, cómo Java, un lenguaje mucho más maduro que nos ofrece algoritmos más complejos y librerías más desarrolladas, sin dejar de lado

que este es el lenguaje que hemos utilizado en mayor medida a lo largo de la carrera y, a su vez, con el que siento más cómodo.



Ilustración 6 - Logotipos de Kotlin y Java

4.4. API de Android

Entendemos Android como un *framework* tanto para los lenguajes de programación Kotlin como Java. Con esto nos referimos a que representa en esquema de desarrollo preciso de la app acerca de los lenguajes mencionados. Generalmente, el beneficio que nos brinda el uso de un *framework* consiste en que el programador no necesita establecer una estructura general del proyecto, sino que simplemente debe ir incorporando estructuras y servicios ofertados por el mencionado *framework* acerca de los lenguajes en los que se desarrolla. Debido a esto, seguidamente se van a describir un conjunto de estructuras básicas y elementos de la programación en Android, teniéndolo en cuenta como un *framework* de desarrollo.

4.4.1. Activities

Un *Activity* expone la lógica empresarial de una pantalla de la aplicación plasmada a través a una interfaz gráfica (en Android las interfaces gráficas se desarrollan en formato XML).

Para crear una nueva *Activity* simplemente necesitamos heredar la clase *Activity*, la cual está predefinida en Android. Además de esto, cabe destacar la existencia de algunas subclases de la misma, las cuales también permiten aplicar la herencia para agregar alguna funcionalidad añadida. Como ejemplos podemos observar *FragmentActivity*, para insertar *fragments* en el proyecto, *ListActivity* para añadir listas...

4.4.2. Layouts y ficheros XML

Entendemos cómo *layout* una estructura que determina la posición en la que se sitúan los elementos en una interfaz gráfica. En la programación en Android, estos *layouts* pueden ser declarados dinámicamente, es decir, en tiempo de ejecución, o de un modo mucho más sencillo, el cual hemos aplicado durante el desarrollo de este proyecto, utilizando la edición de ficheros XML.

Esto nos permite dividir la capa de presentación de la capa lógica. *Android Studio* nos brinda la opción de previsualizar los ficheros XML mientras estos son modificados. De esta forma, podemos apreciar un resultado muy cercano al final mientras escribimos el código XML.

Estos ficheros se relacionan con la capa de negocio mediante una referencia (identificador), la cual se incluye en el método *onCreate* dentro de cada *Activity*.

```
package com.example.enbarco.activities

import ...

class

Login : AppCompatActivity() {
    @RequiresApi(Build.VERSION_CODES.O)

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_login)
    }
}
```

Ilustración 7 - Ejemplo de relación entre *Activity* y *layout*

Coexisten distintos tipos de *layouts*, cada uno con funcionalidades e implementaciones distintas. A continuación, se exponen algunos de los que han empleado en el desarrollo del proyecto. Sus implementaciones aparecen en los anexos situados al final de la memoria.

- **LinearLayout:** Delimita los elementos que contiene de forma vertical u horizontal, situándolos cada uno justo después del anterior en la orientación indicada. Se ha utilizado no solo en los formularios de registro o *login*, sino también para compactar de forma correcta los productos o pedidos realizados.
- **RelativeLayout:** Sitúa los elementos en posiciones relativas, definiendo la ubicación relativa entre elementos “hermanos” (*siblings*). Ha sido especialmente útil en algunos formularios y para situar botones.
- **ScrollView:** Aunque necesita otro contenedor dentro de este para poder recoger los elementos, el *ScrollView* nos permite poder deslizar la pantalla para poder visualizar todo el conjunto de elementos si estos ocupan una extensión vertical mayor a la del dispositivo utilizado.
- **ListView:** Brinda la posibilidad de plasmar listas con una *barra de scroll* incluida, dotando así de la opción de deslizar sin la necesidad de utilizar el *ScrollView* visto anteriormente. Esto ha sido empleado obviamente en aquellas interfaces en las cuales es necesario listar elementos.



Ilustración 8 - Ejemplo de Adapter de productos

Adapters

Los *Adapters* son estructuras utilizadas para relacionar las interfaces de usuario escritas en XML, como ya hemos mencionado, con el código, tanto en Kotlin como en Java, de la aplicación. Estas se han empleado en gran medida con el objetivo de completar listas de elementos. Aunque *Android Studio* nos ofrece utilizar *Adapters* por defecto, nosotros hemos optado por la creación y programación de unos nuevos y personalizados, que se adapten de forma correcta a las necesidades que requiera la aplicación.

En la *Ilustración 8* podemos observar el uso de un *Adapter* para recoger una lista ficticia de productos dentro de la aplicación y mostrarla al usuario en un formato mucho más amigable.

4.4.3. Toolbar

Esta estructura dota a la aplicación de una sencilla e intuitiva navegación a través de las distintas interfaces, unificándolas al mismo tiempo. En nuestra aplicación se han empleado dos *Toolbars* o barras de tareas distintas. La primera es la utilizada a lo largo de todas las pantallas relacionadas con los clientes, mientras que la segunda aparece siempre que nos encontremos en una sesión con una “cuenta de empresa”.

Estas barras unifican las pantallas y, al mismo tiempo, sirve para distinguir claramente desde que tipo de cuenta hemos iniciado la sesión. En las *Ilustraciones 9 y 10* podemos observar los ejemplos mencionados anteriormente.



Ilustración 9 - Muestra de Toolbar de usuario 1



Ilustración 10 - Muestra de Toolbar de usuario 2

4.4.4. Intents

Estos elementos son utilizados en Android, generalmente con el objetivo de realizar una transición entre *Activities*, además de ofrecer la posibilidad de transmitir datos entre ellas. Aunque los *Intents* pueden ser tanto implícitos como explícitos, por sencillez hemos limitado su uso únicamente a los de tipo explícito. Los *Intents* de tipo implícito suelen ser usados para relacionar y comunicar las aplicaciones predeterminadas de los dispositivos Android con las nuevas apps que instala el usuario, cómo por ejemplo el correo electrónico, los contactos telefónicos, la galería de imágenes... Por este motivo, los *Intents* implícitos no tienen cabida en este proyecto.

En la *Ilustración 11* podemos observar el uso de tres *Intents* que redirigen a tres pantallas distintas dependiendo del botón de la *Toolbar* que pulse el usuario.

```

toolbarMenu.setOnClickListener(view -> {
    Intent intent = new Intent(getApplicationContext(), Menu.class);
    startActivity(intent);
    finish();
});

toolbarProfile.setOnClickListener(view -> {
    Intent intent = new Intent(getApplicationContext(), MyAccountUser.class);
    startActivity(intent);
    finish();
});

toolbarShoppingCart.setOnClickListener(view -> {
    Intent intent = new Intent(getApplicationContext(), ShoppingCart.class);
    startActivity(intent);
    finish();
});

```

Ilustración 11 - Ejemplo de Intents

4.4.5. El fichero manifest.xml

Cualquier proyecto desarrollado en Android debe incluir este fichero. En este es donde se declaran las *Activities* existentes en la aplicación. También deben incluirse los permisos de acceso al hardware del dispositivo, como lo son la cámara, GPS, etc. De la misma forma, se incluyen los permisos de acceso a servicios software a los que se necesita acceder (galería de imágenes, contraseñas almacenadas, contactos, etc.).

Adicionalmente, es importante destacar que aquí es donde se define el nivel mínimo de API de Android que es requerido por la aplicación para que esta funcione, es decir, la versión más antigua que debe tener instalada el dispositivo para que se permita la instalación de la aplicación. En nuestro caso, hemos optado utilizar la API 26. Esto se debe tanto a que no se trata de una API excesivamente moderna, por lo que la mayoría de los dispositivos tienen acceso a ella, además de que tampoco es excesivamente antigua, por lo que ofrece un gran nivel de librerías y funcionalidades predefinidas que son de gran ayuda en el momento de la programación.

```

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.enbarco">

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />

    <application
        android:allowBackup="true"
        android:hardwareAccelerated="false"
        android:icon="@mipmap/ic_launcher"
        android:label="EnBarco"
        android:largeHeap="true"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.EnBarco">
        <activity
            android:name=".activities.ShowPedidoEmpresa"
            android:exported="false" />
        <activity
            android:name=".activities.PreviousOrdersEmpresa"
            android:exported="false" />
        <activity
            android:name=".activities.ModifyEmpresaData"
            android:exported="false" />
        <activity
            android:name=".activities.MyAccountEmpresa"
            android:exported="false" />
        <activity
            android:name=".activities.ShowPedido"
            android:exported="false" />
        <activity
            android:name=".activities.PreviousOrders"
            android:exported="false" />
        <activity
            android:name=".activities.ShowFavorites"
            android:exported="false" />
    </application>
</manifest>

```

Ilustración 12 - Fichero Manifest.xml

4.4.6. Sensor GPS

Indudablemente, una de las mayores dificultades relacionada con la programación en Android es el uso de los servicios de localización. Estos nos brindan una API de gran potencia que nos permite administrar el uso de los sensores incorporados en los dispositivos móviles, como el GPS o el Wi-Fi. Para facilitarnos su uso, Android reduce la dificultad permitiéndonos el acceso al hardware del dispositivo mediante la implementación de métodos como: *onStatusChange*, *onProviderEnabled*, *onLocationChanged*, *onProviderDisable...*

Para obtener la ubicación de un dispositivo móvil, dependemos de que el sensor esté recibiendo la señal correctamente en un determinado momento y, de la misma forma, de la calidad de la misma. Debido a esto, no es de extrañar que la precisión de las coordenadas pueda variar. En el desarrollo de este proyecto hemos prestado especial importancia a la precisión de estas coordenadas, puesto que es necesario saber la posición exacta del usuario no solo para comprobar si puede o no realizar un pedido, sino desde detalles tan simples como que productos se le van a mostrar, el precio del envío que se le va a cobrar o si están habilitados los productos que haya guardado como favoritos.

4.4.7. Mensajes Toast

Entendemos como mensaje *Toast* a notificaciones *pop up* que aparecen en pantalla durante un periodo corto de tiempo. Estos mensajes son extremadamente útiles para ofrecer información adicional en eventos determinados. Su tiempo de duración es determinado y generalmente corto, y cuando este expira desaparecen automáticamente. Durante el desarrollo de la aplicación se han utilizado para comprobar el correcto funcionamiento de distintas partes del código y evitar errores, comprobando si se podían llegar a ejecutar determinadas partes del mismo a las cuales no se debería acceder. Podemos tomar como ejemplos los siguientes eventos:

- Errores de conexión o formateo relacionados con la base de datos
- Errores de relación entre formularios
- Comprobar que determinadas acciones se han ejecutado de forma correcta
- Verificar el correcto funcionamiento de algoritmos complejos

Además, en la versión final se han dejado distintos mensajes *Toast* que ayudan a informar al usuario de las acciones que está realizando y de que estas se están ejecutando correctamente. Esto es muy útil para que los usuarios no se sientan

perdidos y le proporcionan seguridad en el momento de introducir datos o navegar por la aplicación. Este tipo de mensajes ayuda a retroalimentar la comunicación entre la aplicación y sus usuarios, sin mencionar que son sutiles, ocupan poco espacio y duran tan solo unos pocos segundos, con lo que no llega a incomodar a usuario más avanzados.

```
if (empresa.password == loginPasswordTextView.text.toString()) {
    Empresa.empresaLogged=empresa
    Toast.makeText(applicationContext, text: "Successfully logged", Toast.LENGTH_LONG).show()
    goToRVActivity()
} else {
    Toast.makeText(applicationContext, text: "Password is wrong", Toast.LENGTH_LONG).show()
}
} else {
    Toast.makeText(applicationContext, text: "User not found", Toast.LENGTH_LONG).show()
}
}

override fun onCancelled(error: DatabaseError) {
    Toast.makeText(applicationContext, text: "Error", Toast.LENGTH_LONG).show()
    throw error.toException() // never ignore errors
}
```

Ilustración 13 - Mensaje Toast en el código



Ilustración 14 - Mensaje Toast ejecutado

4.5. Diseño detallado

En este apartado se muestra al lector el diseño propuesto para el desarrollo de la aplicación. También se pretende exponer la razón de este diseño y hacer entender al lector la funcionalidad de cualquier evento en la aplicación.

Teniendo en cuenta los problemas encontrados en el capítulo 3, se ha diseñado el software de forma que dé una respuesta a la incógnita de cómo solucionar estos inconvenientes.

4.5.1. Diseño de pantallas

Respecto a las pantallas y la navegación a través de la aplicación, hemos seguido el ejemplo que nos brindó el célebre físico alemán Albert Einstein: *“Hazlo todo tan simple como sea posible, pero no más simple”*.

Hemos buscado la forma de simplificar al máximo la navegación a través de la aplicación sin dejar de lado su funcionalidad. De la misma forma que sólo se requiere la mínima información necesaria por parte de los usuarios, también se les brinda la mínima información que requieran para realizar los pedidos [5]. Hemos tenido este aspecto especialmente en cuenta a lo largo del desarrollo del proyecto, ya que la mayoría de los usuarios no están familiarizados con el uso de los dispositivos móviles. De esta forma les podemos facilitar un uso intuitivo durante todo su transcurso por la aplicación.

Con objetivo de ilustrar la navegación entre pantallas planteada para este proyecto, podemos observar la *Ilustración 15*, que corresponde al diagrama de flujo de la aplicación.

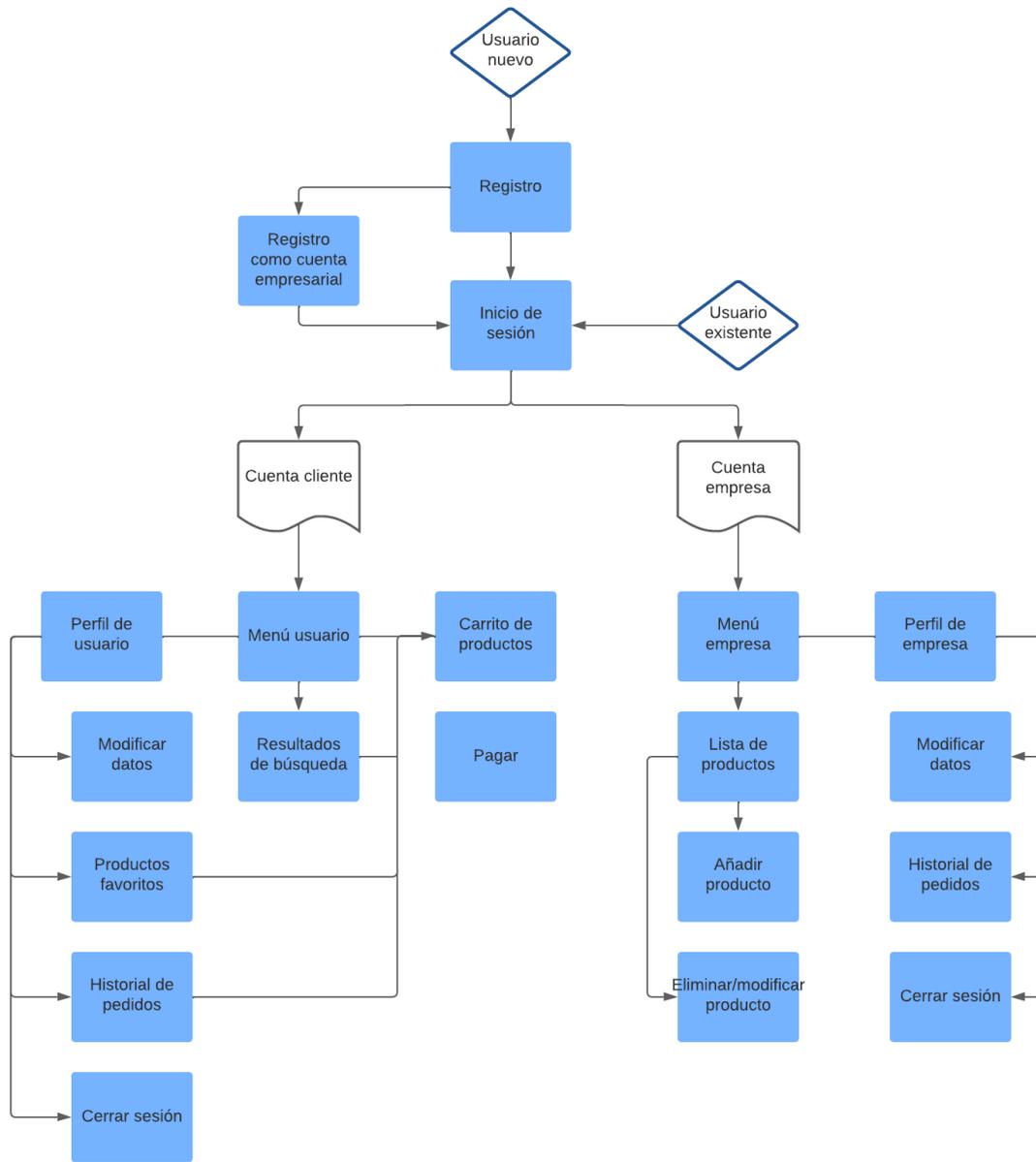


Ilustración 15 - Diagrama de flujo

Más tarde veremos detalladamente la navegación entre las distintas pantallas y el resultado final en el capítulo 8, Aplicación final.

4.5.2. Diseño de la interfaz

Respecto al diseño de las interfaces, estas han sufrido diversos cambios a lo largo del desarrollo del proyecto, tanto por conveniencia al programar, como para añadir funcionalidades y, obviamente, algunas por estética.

La herramienta utilizada para el diseño de estas interfaces ha sido Figma. Figma es una plataforma online que nos permite diseñar interfaces con un nivel bastante avanzado de detalles, permitiéndonos así ver una versión cercana a la final desde el inicio del proyecto. Previo al diseño en Figma, se han dibujado bocetos muy simples en papel, los cuales han servido para dar forma y estructura a las interfaces detalladas.

Como podemos ver en la Ilustración 16, podemos apreciar desde las versiones iniciales, menos desarrolladas y con más imperfecciones, hasta las interfaces finales, las cuales cuentan con un nivel de detalle mucho mayor.

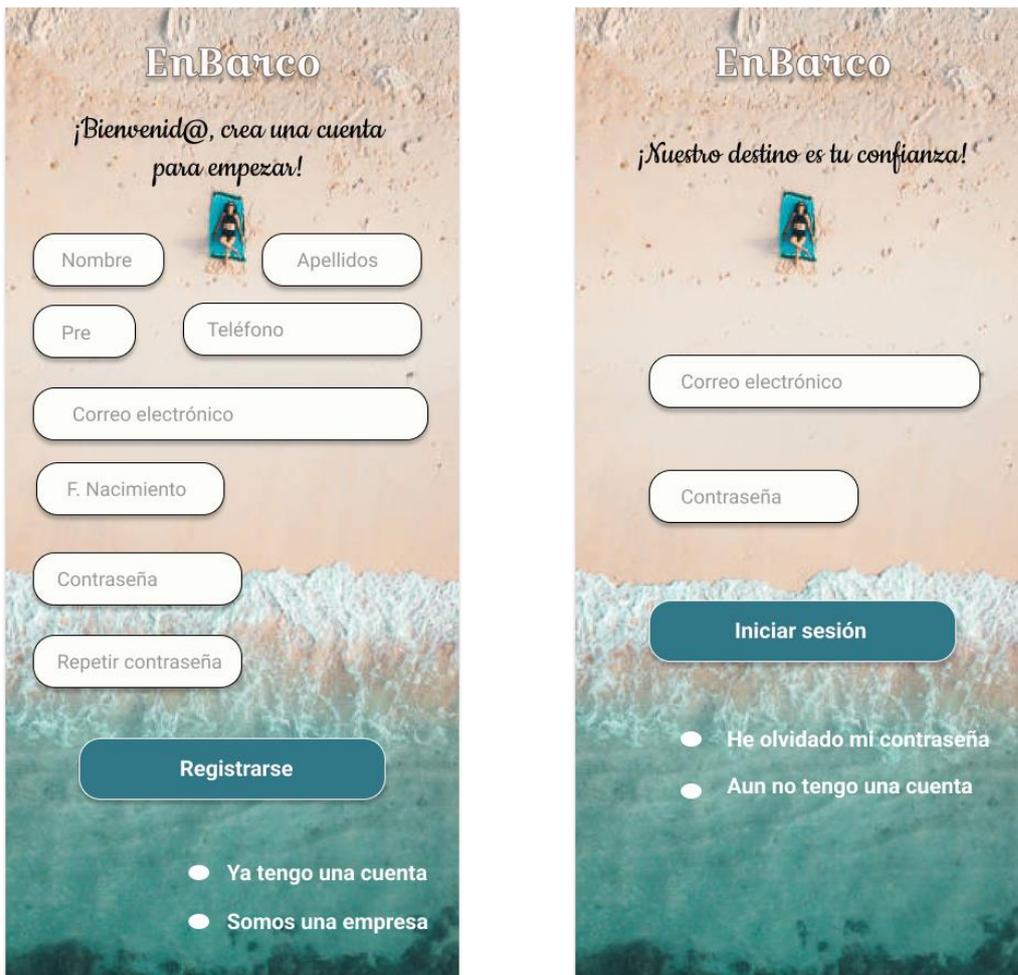


Ilustración 16 - Interfaz 1

Estas fueron las primeras interfaces diseñadas, dirigidas a las pantallas de registro e inicio de sesión respectivamente.

Estos diseños contaban con diversos errores. Respecto a la pantalla de registro, el logotipo de la empresa no llega a destacar de la forma que nos gustaría, pero el problema principal que nos plantea es el exceso de información que se le requiere al usuario antes de poder empezar a realizar un pedido [6].

En cuanto al inicio de sesión, este diseño es bastante correcto, pero sigue presentando problemas estéticos, como la asimetría en algunos puntos, además de la incorrecta lectura de algunas palabras.



Ilustración 17 - Interfaz 2

En estos nuevos diseños, podemos apreciar una mejor lectura del texto y un fondo más amigable, que nos permite una mayor facilidad a la hora de crear una continuidad de colores a lo largo de la aplicación [7]. Además, hemos separado el proceso de registro en tres pantallas distintas, evitando así que el usuario se sienta abrumado al tener que introducir mucha información simultáneamente. En la primera pantalla simplemente introducirá información mundana como el nombre, correo y la comprobación de la mayoría de edad, y en la segunda incluirá el teléfono y la contraseña, además de la política de privacidad.

Si el usuario es una empresa que quiere formar parte de nuestros servicios, de la misma forma que a los clientes se le pedirá en una primera pantalla el nombre de la empresa, correo, ubicación de esta y una imagen de perfil que se mostrará al resto de usuarios. La segunda pantalla es exactamente la misma que para los usuarios. En la versión final se ha incluido una tercera pantalla de registro en la que se pide la introducción del código enviado para la verificación del número de teléfono.

También, como podemos apreciar en la *Ilustración 18*, se ha incluido unos *mockups* tempranos, tanto del menú principal de los usuarios y de las empresas, como de la lista de resultados. Para ver la versión final, la cual sigue la misma temática estética que las pantallas anteriores, podemos dirigirnos al capítulo 8 de esta memoria, Aplicación Final.

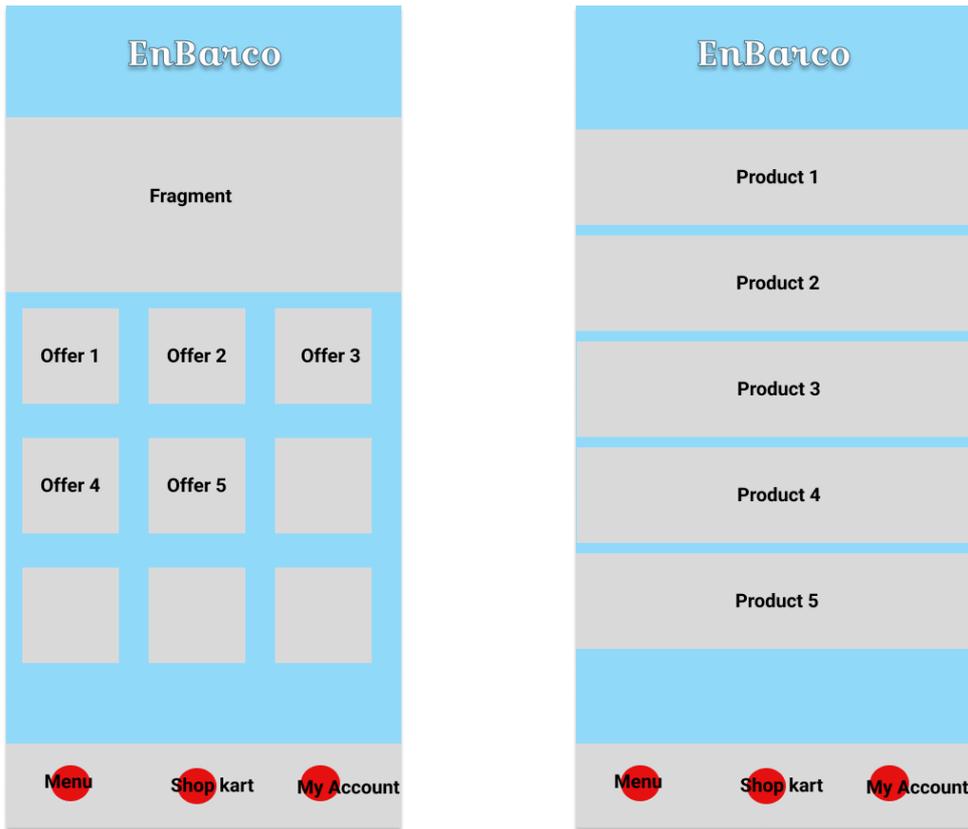


Ilustración 18 - Interfaz 3

4.5.3. Diseño de la base de datos

En cuanto a la base de datos, como ya hemos mencionado, utilizaremos *Firestore Realtime Database*. A diferencia de SQL, esta no es una base de datos relacional, por lo que no es posible diseñar una estructura previa a su creación. La estructura de esta base de datos se crea a medida que van apareciendo campos en ella.

Teniendo esto en cuenta, hemos esbozado su estructura previa a la programación de la aplicación, de forma que solo se creen los campos deseados. Esta base de datos está compuesta por objetos, los cuales son creados por la capa lógica del programa y trasladados a la capa de persistencia posteriormente, ya con la forma deseada.

Para simplificar la estructura y reducir la carga de datos que deba soportar esta capa de persistencia, solo existen cinco tipos de objetos: Empresa, Favorito, Pedido, Producto y Usuario. Tanto Usuario como Empresa se crean en la fase de registro de la aplicación. Posteriormente, se diseñaron los productos, los cuales pertenecen a una

empresa e incluyen, por tanto, el objeto Empresa al que hacen referencia. Los objetos Favorito incluyen un Usuario y una lista de objetos Producto, con la intención de relacionar estos productos con este usuario. Finalmente, los objetos Pedido están compuestos por un Usuario y otra lista de objetos Producto, pero además incluyen datos adicionales como la fecha en la que se realizó el pedido. Si un Pedido pertenece a una empresa, tratamos este como un Usuario junto a la lista de objetos Producto que pertenecen a dicha empresa, ya que hemos considerado que una empresa no requiere saber los productos que este usuario ha realizado a otras empresas, aunque esto ocurra en un mismo pedido.

4.5.4. Diseño de scripts

Finalizando el apartado de diseño, describiremos el diseño y desarrollo de los scripts, es decir, la programación realizada.

Aplicando la metodología ágil al proyecto, este se ha dividido en diversos *Sprints* con objetivos determinados [8]. En este apartado veremos el orden de realización de estos y los puntos programados en cada uno.

En el primer *Sprint*, aunque no cumpla exactamente el objetivo de la metodología ágil, decidimos dedicarnos a programar tanto el inicio de sesión como la pantalla de registro. Optamos por esto para disponer lo antes posible de los objetos Usuario y Empresa mencionados anteriormente, los cuales sirven como cimientos para la estructura de la base de datos.

Posteriormente, el segundo *Sprint* se dedicó a brindar la posibilidad a las empresas de añadir productos y a los usuarios de poder visualizarlos. Esto conllevó mucho esfuerzo porque no solo había que pensar en cómo mostrar los productos, ya que estos cuentan con un *layout* distinto para ellos, sino que también debimos diseñar el algoritmo que mostrase solo los productos relevantes para el usuario en su determinado momento.

El tercer *Sprint* cumplió con el objetivo de optimizar los eventos programados durante el *Sprint* anterior, ofreciendo la opción de modificar y eliminar productos a las empresas. De la misma forma, ofreció a los usuarios la opción de escoger productos deseados y guardar estos en el carrito de productos. También se pueden visualizar los productos escogidos desde la pestaña respectiva del producto.



El cuarto *Sprint* fue destinado a programar la creación de pedidos y el pago de estos, además de almacenarlos en la base de datos.

El quinto *Sprint* tuvo el cometido de otorgar la capacidad de modificar los datos personales tanto a los clientes como a las empresas. Además, les permite visualizar los pedidos realizados y los productos guardados como favoritos.

El sexto y último *Sprint* se dedicó simplemente a añadir flecos al programa, solucionar problemas encontrados anteriormente y corregir cualquier inconveniente que pudiese aparecer durante la realización de las pruebas. Es importante incluir este último *Sprint* en la programación para evitar inconvenientes que puedan ocurrir en los últimos momentos del proyecto [9].

5. Desarrollo de la solución propuesta

En este capítulo describiremos brevemente el guion que ha seguido el desarrollo de este proyecto. Distintos puntos se explican más detalladamente en sus respectivos capítulos, como los problemas encontrados o las posibles mejoras.

Inicialmente, y de acuerdo con el orden de los *Sprints*, el primer paso fue desarrollar las interfaces y programar su funcionamiento. Para esto nos ayudamos de la API de Google mencionada anteriormente. Esta fue necesaria para poder marcar en un mapa la ubicación de las empresas o la autenticación en el *Login*. Durante este apartado nos preparamos para facilitarle el uso lo máximo posible al usuario, preocupándonos por incluir el @ en todos los campos de texto relativos al correo electrónico, iniciar la introducción de los nombres o apellidos con mayúscula automáticamente, o no pedir demasiados datos simultáneamente al cliente.

Una vez finalizado el registro e inicio de sesión, nos preocupamos de programar los menús de usuario y establecimiento. Aquí surgieron varios fallos relacionados con el rendimiento de la aplicación, ya que, al no tener experiencia previa en la programación en Android, introdujimos varias imágenes de alta calidad, las cuales disminuían la fluidez de la aplicación. Podemos encontrar más información acerca de este problema y su solución explicada en el punto 9.1. Sin tener en cuenta este problema, el desarrollo de este apartado siguió el diseño planeado, iniciamos creando el menú, preparamos una interfaz para poder visualizar los productos y los mostramos según los filtros que haya seleccionado el usuario. También creamos una barra de tareas que nos ayudará posteriormente a simplificar la navegación a través de la aplicación [16].

Próximamente, incluimos la posibilidad de añadir, modificar y eliminar productos desde una cuenta establecimiento. Además de esto, codificamos un carrito de productos que permite agruparlos y mostrarlos al cliente. También programamos las pantallas de perfil tanto de cliente como de empresa. Buscamos simplificar estas lo más posible ya que, en el diseño de nuestra aplicación, no tenemos pensado que dos clientes puedan comunicarse entre sí, por lo que no vemos necesario profundizar en la personalización.

Con el carrito de compra finalizado y la posibilidad de añadir y modificar productos correctamente implementada, decidimos que programar la forma de pago sería el siguiente paso, proporcionándonos así una aplicación básica, pero al fin, funcional. Para la programación de esta forma de pago, utilizamos la lista de productos



guardados en el carrito, recogiendo el precio de cada uno de estos y, mediante un algoritmo que nos ayuda a calcular la distancia entre estos y el cliente, calculamos el coste de envío respectivo.

Con todo esto funcionando de forma adecuada, tenemos una aplicación funcional, aunque pobre. Por eso, el próximo paso que decidimos tomar fue añadir la posibilidad de guardar los productos que más le gusten a cada usuario determinado como favoritos. Esto resultó bastante sencillo, teniendo en cuenta que realmente es una lista de productos, sin siquiera la necesidad de guardar la cantidad de estos como en el carrito de productos. La pestaña de favoritos añade mucha personalización y comodidad al usuario, ya que le permite no tener que buscar siempre un producto que pide asiduamente.

Además de esto, vimos necesario un historial de pedidos que guarde los pedidos realizados. Esto tampoco provocó grandes dificultades, si entendemos un pedido como guardar en la capa de persistencia un carrito de productos con la hora en la que se ha pagado. La única complicación fue como mostrar estos pedidos en la interfaz, ya que no nos decidíamos por qué información era necesario mostrar y de qué forma. Este historial, por parte de las empresas, solo almacena los productos que se han pedido a esta respectiva empresa. Esto se ha programado creando pedidos derivados de un mismo pedido según a que empresa pertenece cada producto. Por cada pedido realizado por el cliente, se pueden crear hasta un número n de pedidos para cuentas establecimiento, siendo n el número de productos distintos que incluye el pedido realizado.

Finalmente, se han realizado las pruebas necesarias, acompañadas de un test de usabilidad que explicaremos en mayor detalle en el capítulo 7. Después de realizar estas pruebas y este test, se han corregido los fallos encontrados en estas, siendo la mayoría pequeños despistes en el código [10].

6. Implantación

Este capítulo cumple con el objetivo de mostrar cómo se implantará la aplicación en el mundo real. Destacaremos, antes de profundizar, la imposibilidad de subir la aplicación a Google Play de forma gratuita, por lo que supondremos esto de forma que su implantación sea realista.

La forma de instalación es realmente simple, pues únicamente hay que acceder a la aplicación nativa de nuestro dispositivo Android, Google Play y escribir en la barra de búsqueda "EnBarco". Una vez escoges la opción "instalar", esta se instala directamente en tu dispositivo.

Por parte de los clientes, lo único que se les requerirá será la creación de una cuenta con su información básica y aceptar el acceso a la ubicación de su dispositivo con el fin de poder ofrecer los productos relevantes a su situación.

En cuanto a las empresas, su forma de empezar a formar parte de nuestros servicios no es muy distinta de la de los usuarios. Estas deben crearse igualmente una cuenta, pero esta debe ser una cuenta empresarial. Durante su registro, indicarán su ubicación con el fin de que sus productos se muestren en un rango de hasta 5 kilómetros de donde se encuentre situada. Una vez creada la cuenta, deberá añadir los productos que ofrezcan para repartir. Con este simple gesto, los usuarios podrán realizar los pedidos que deseen y les aparecerá a las empresas en su lista de pedidos. Asimismo, uno de nuestros repartidores irá a realizar y recoger el pedido en persona, indicando que este fue a través de la empresa y mostrando el justificante.



7. Pruebas

Las pruebas realizadas para confirmar el correcto funcionamiento de la aplicación se han separado en dos sectores, que engloban la totalidad del proyecto. Primeramente, la fase de pruebas por parte de los desarrolladores que han codificado el proyecto. Estas se dividen en pruebas unitarias, de integración y funcionales [11]. En segundo lugar, tenemos las pruebas realizadas a usuarios reales externos al desarrollo del proyecto, la mayoría de ellos sin conocimientos avanzados en informática ni en el desarrollo de aplicaciones. Estas últimas pruebas se han realizado mediante un test de usabilidad creado por nosotros, que mide la facilidad de uso de la aplicación, la información proporcionada y el correcto funcionamiento de esta [12].

7.1. Pruebas realizadas por los desarrolladores

En primer lugar, tenemos las pruebas realizadas por nosotros mismos, los desarrolladores. Estas consisten en verificar el correcto funcionamiento del código escrito realizando tareas específicas que cumplan un objetivo determinado.

7.1.1. Pruebas unitarias

Empezaremos mencionando las pruebas unitarias. Estas pruebas solo comprueban la funcionalidad específica de una única clase. Se han realizado frecuentemente desde el inicio del proyecto, incluso repetidas veces a la misma clase. Son pruebas simples, como por ejemplo ejecutar un determinado evento o comprobar que un botón cambia de color al pulsarse. Estas son las pruebas más comunes durante el desarrollo de todos los proyectos y son necesarias para verificar minuciosamente el comportamiento del código escrito.

7.1.2. Pruebas de integración

Las pruebas de integración, como su nombre indica, son pruebas que verifican que las relaciones existentes entre las diferentes clases y servicios funcionan correctamente. Este tipo de pruebas intenta encontrar todos los problemas que pueden aparecer al combinar distintas capas de nuestra aplicación. Un ejemplo muy visual acerca de estas pruebas es comprobar que la capa de persistencia almacena determinada

información proporcionada desde la capa lógica. También podemos destacar el envío e introducción del mensaje de teléfono para confirmar el registro en el sistema.

Suelen ser pruebas más costosas de desarrollar y ejecutar que las pruebas unitarias, ya que en cada una de ellas es necesario integrar diferentes puntos del sistema para confirmar que funcionan de forma correcta en conjunto.

7.1.3. Pruebas funcionales

Por último, mencionaremos las pruebas funcionales. Estas pruebas verifican el comportamiento del sistema para afirmar que se cumple con la funcionalidad completa.

Este tipo de pruebas se realizan con los requisitos de funcionamiento en mano, y no se centran en detalles menos del sistema, que ya deberían estar comprobados gracias a los dos grupos anteriores.

En el test de usabilidad realizado se usan continuamente este tipo de pruebas, ya que buscamos que los usuarios nos brinden una opinión acerca de una versión final, consolidada en su gran mayoría.

Como ejemplo, podemos encontrar la elección de los productos deseados, el pago de estos y revisarlo comprobando el historial de pedidos. Esto comprobaría una gran cantidad de clases al mismo tiempo, además de la conexión entre la capa lógica y la de persistencia. Sumado a esto, podremos ver el grado de dificultad necesario para realizar la tarea y la efectividad en su desempeño.

7.2. Test de usabilidad realizado

Para comprobar el correcto funcionamiento de la aplicación y conocer el nivel de satisfacción de los usuarios, preparamos un test de usabilidad y se lo realizamos a un total de 20 personas. De estas 20 personas, solo se encontraban 4 estudiantes de Ingeniería Informática, siendo las otras 16 personas ajenas a este campo y, algunas de ellas, sin nociones en el uso de estas tecnologías.

Las respuestas en su gran mayoría fueron satisfactorias y nos indicaron una correcta funcionalidad de la aplicación, además de una navegación instintiva a través de esta y



un uso sencillo. También estuvieron de acuerdo en que la información requerida no es excesiva y no les resultó incómodo realizar el proceso de registro.

Todas pudieron identificar a la empresa EnBarco como propietaria de la aplicación y no tuvieron ningún problema en manejar cuentas de tipo empresa, haciéndonos conocer la facilidad de uso de estas y, mostrando a la vez, un genial desempeño del proyecto.

A pesar de los resultados mayormente positivos, también sirvió para encontrar algunos pequeños fallos de programación que se nos habían escapado después de realizar nosotros mismos las pruebas. También hubo una persona mayor de 70 años, a quien se le dificultó más que al resto navegar a través de la aplicación. Aunque no se trata del público objetivo de la empresa, buscamos que nuestra aplicación sea accesible y de fácil uso para todos los públicos, por lo que tuvimos esto en cuenta como una pequeña brecha en nuestro objetivo de un uso intuitivo y sencillo de la aplicación.

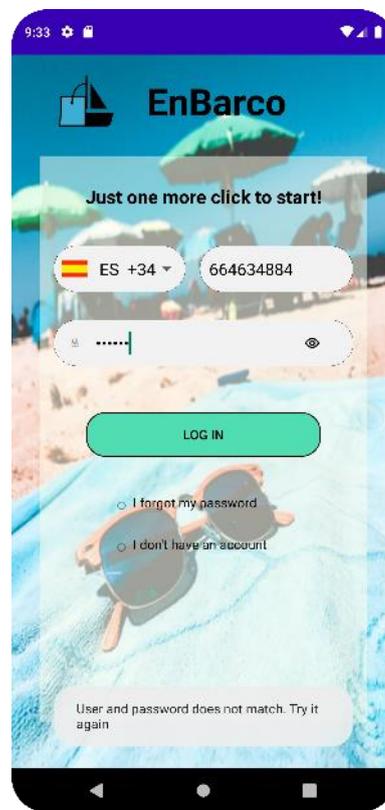
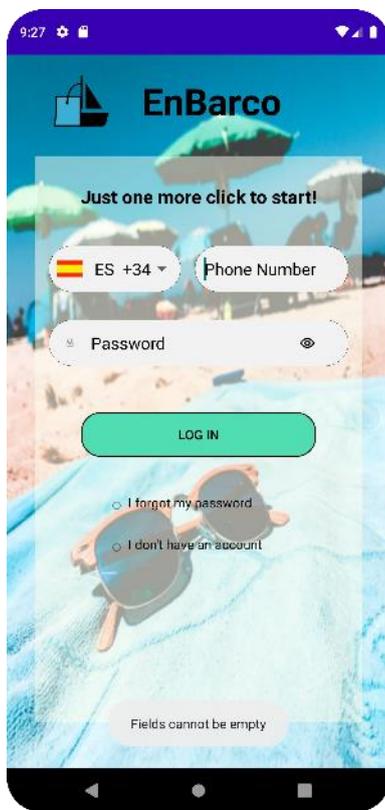
El test realizado cuenta con cuatro partes diferenciadas a lo largo de su realización. La primera es una breve introducción del usuario, que nos permite conocer su grado de familiaridad con las tecnologías usuales, edad y conocimiento acerca del sector. Posteriormente, se muestran distintas pantallas y se pide a los usuarios que identifiquen determinada información en ellas. En tercer lugar, estos realizan tareas específicas a través de la aplicación, dándonos a conocer la facilidad con la que se encuentra la información que buscan y cuan intuitivo es la ejecución de determinadas tareas. Finalmente, se le permite navegar libremente durante cinco minutos a lo largo de toda la aplicación y se le realizan preguntas acerca de esta navegación, conociendo así el grado de satisfacción del usuario y la comodidad de este durante su uso.

Podemos encontrar el test realizado en su respectivo anexo, Test de Usabilidad.

8. Aplicación final

Seguidamente, se mostrarán las pantallas de la aplicación final junto a su funcionamiento. En adición, se mostrarán algunos mensajes *Toast*, de los que hablamos en el punto 4.4.7, con el objetivo de apreciar la información y retroalimentación que estos brindan al usuario.

8.1. Login



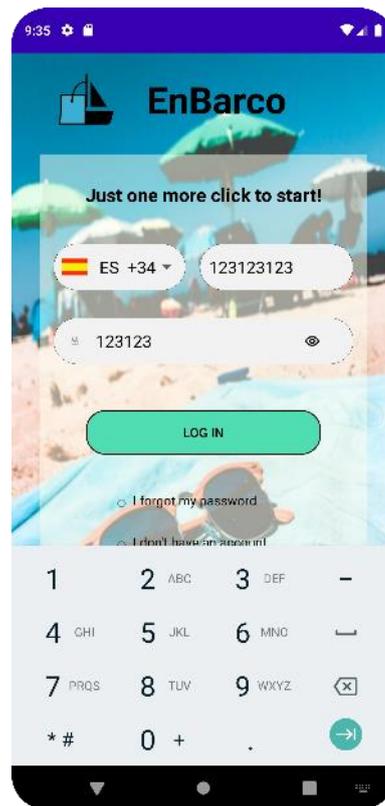
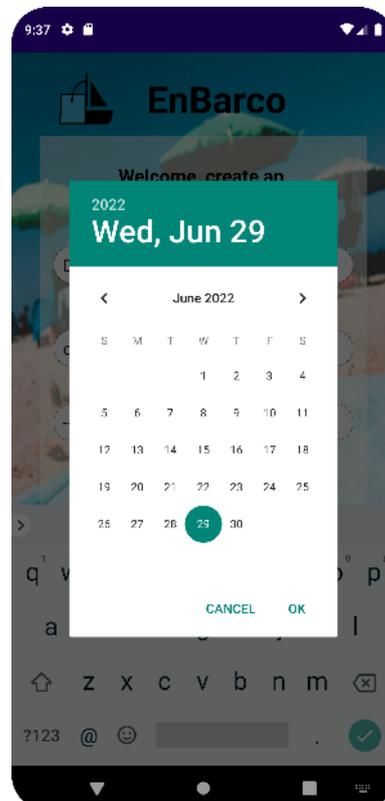
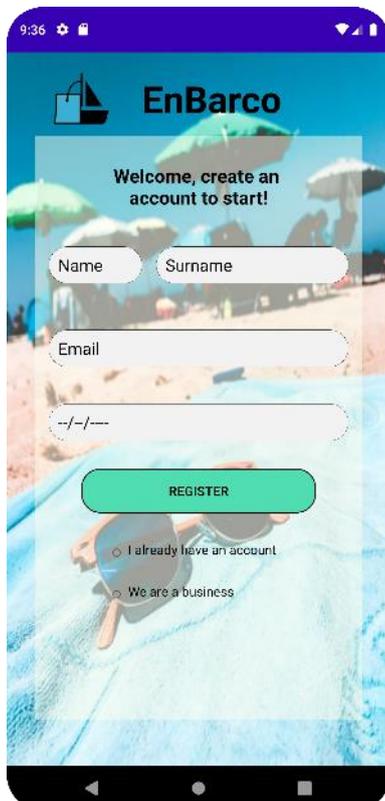
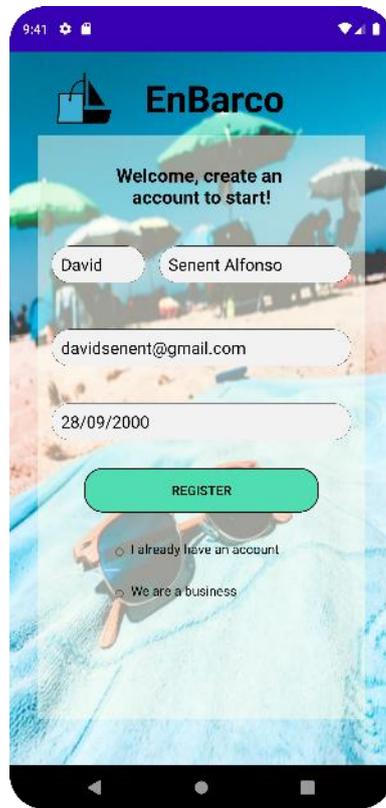
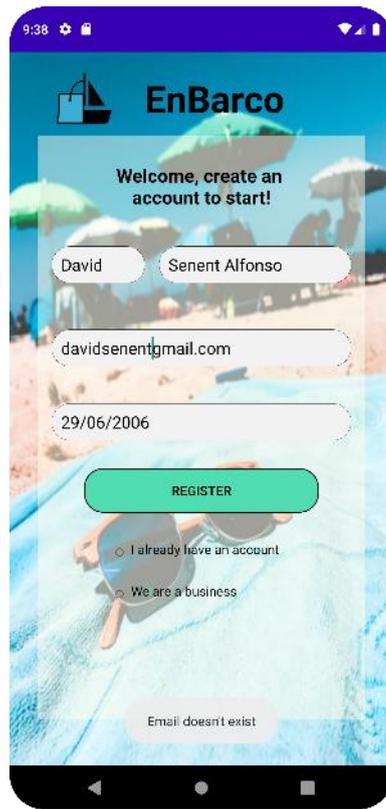
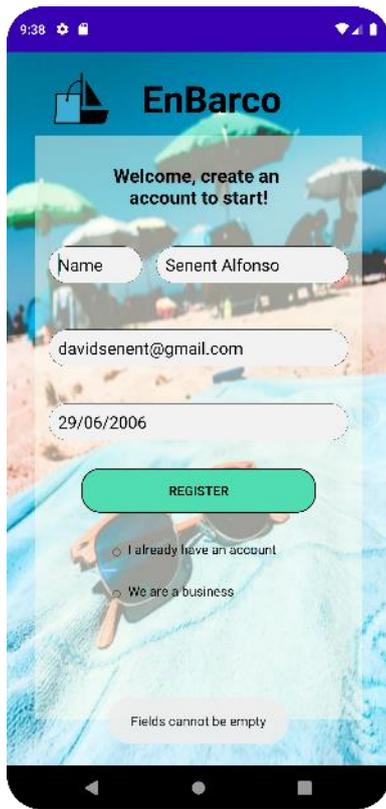


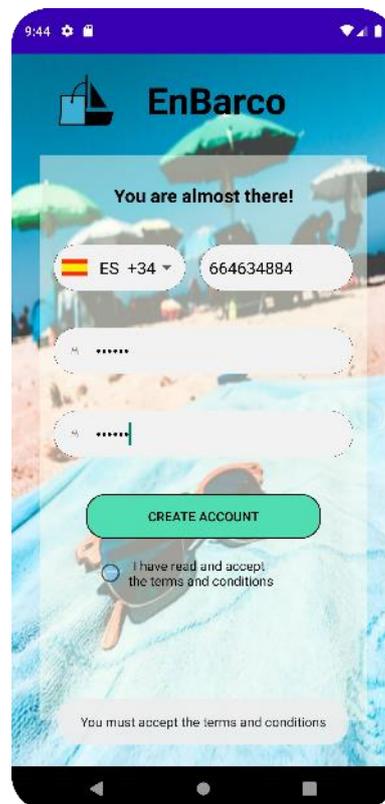
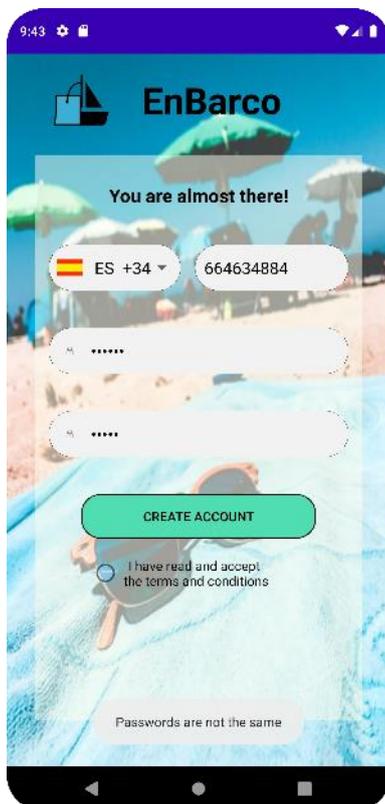
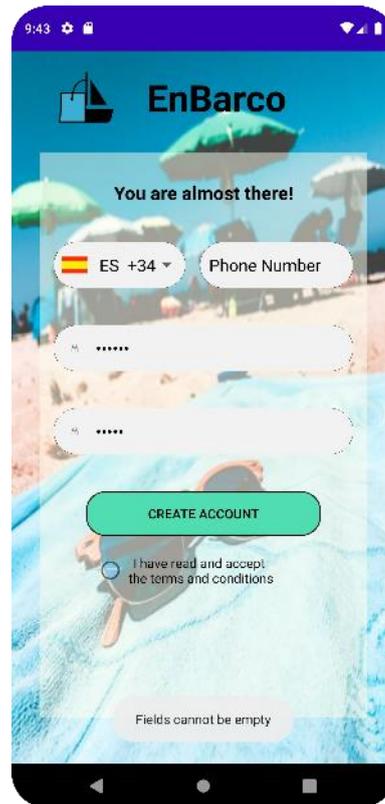
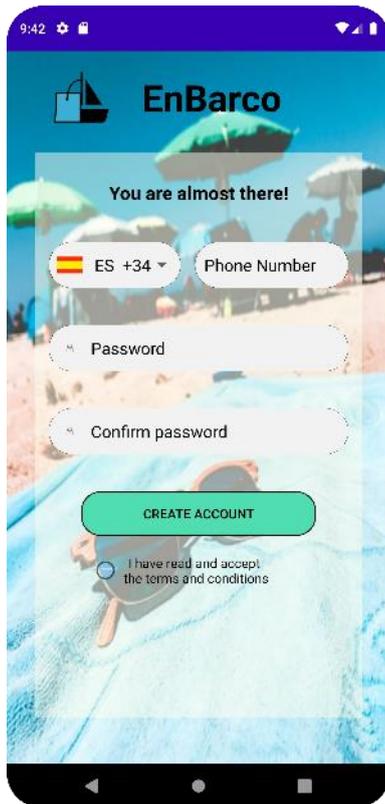
Ilustración 19 - Login

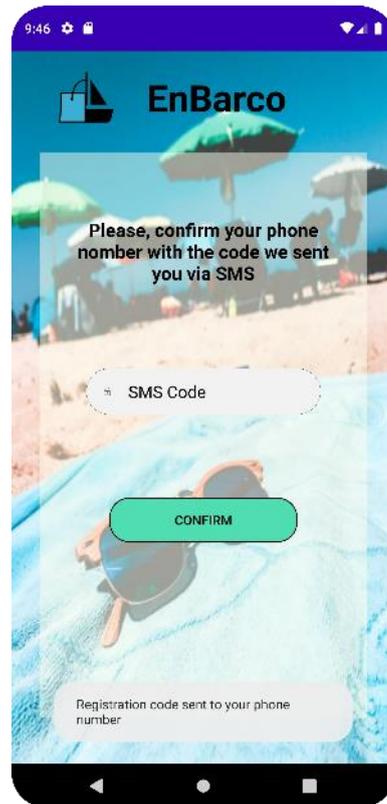
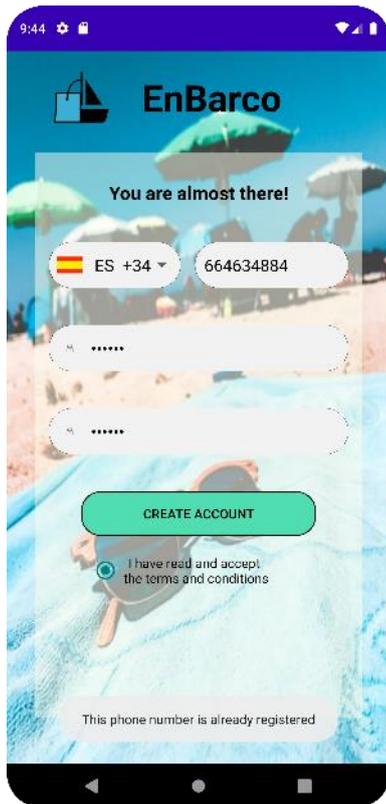
Registro de usuario





Desarrollo de una aplicación Android para la empresa EnBarco





Ejemplo de mensaje Toast usado para advertir al usuario que ha ocurrido un evento externo

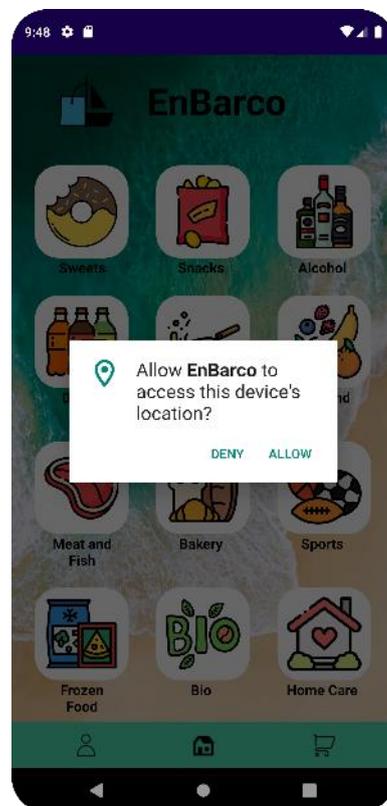
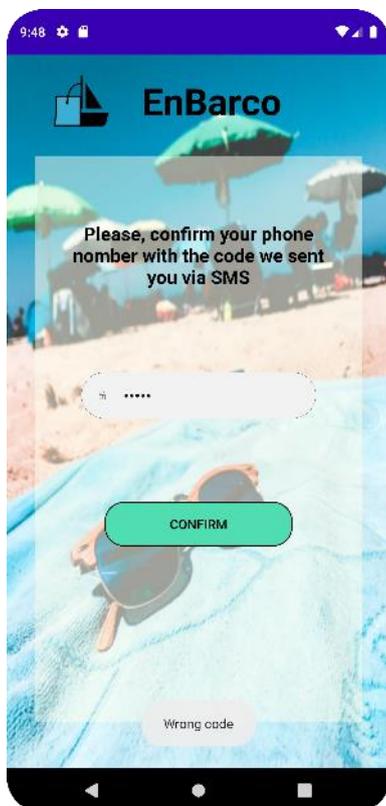
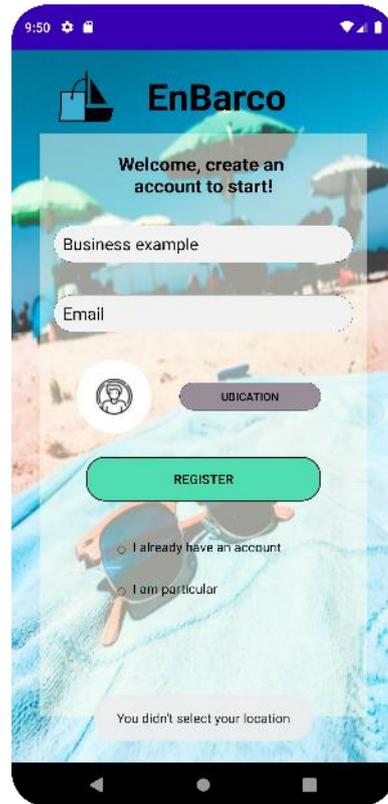
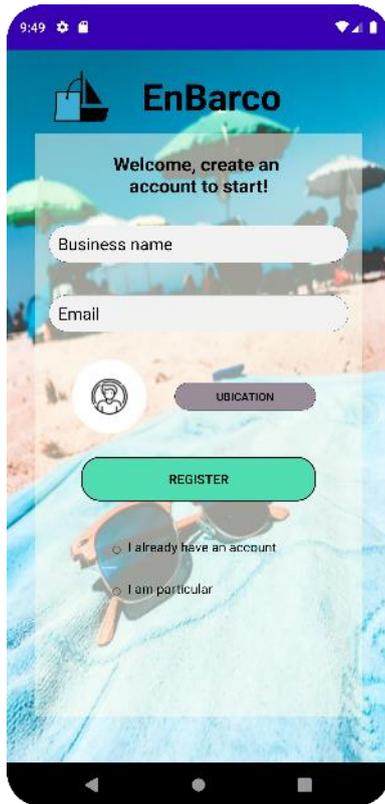
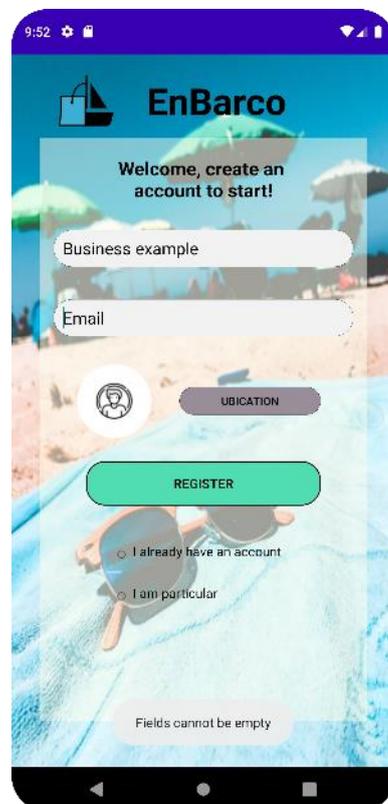


Ilustración 20 - Registro

8.2. Pantallas de registro de empresa



Podemos observar distintos mensajes Toast que ayudan al usuario a entender si está incluyendo información incorrectamente o si no la está incluyendo y esta es necesaria



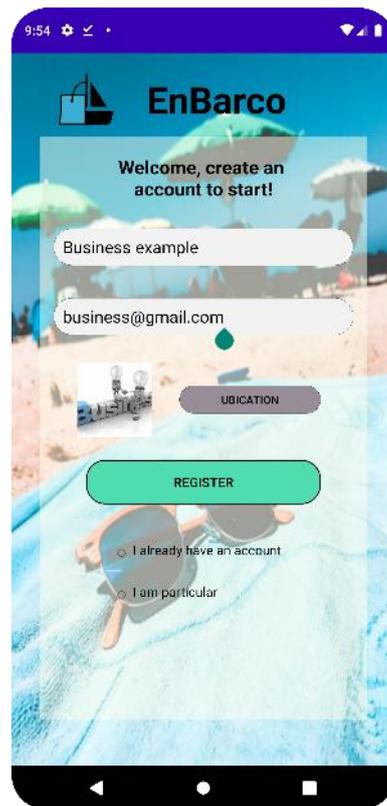
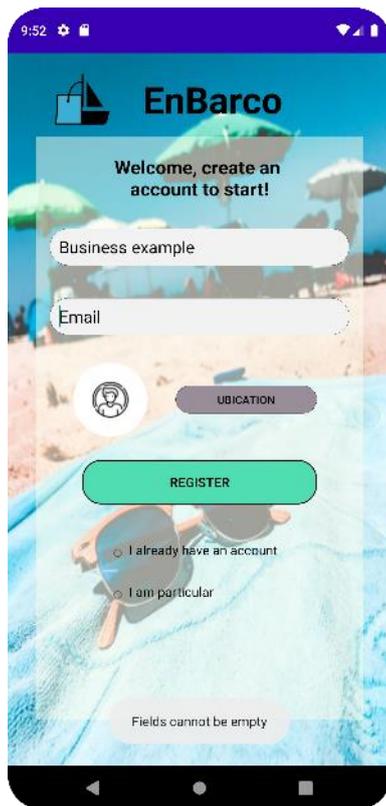


Ilustración 21 - Registro de empresa

Aquí se redirige al usuario a la misma pantalla mostrada anteriormente para que introduzca su número de teléfono y contraseña y, posteriormente, enviar el SMS de confirmación. Esto se ha desarrollado de esta forma para evitar crear interfaces innecesarias, optimizar el tiempo de programación y reducir o, a poder ser, eliminar el código duplicado [13].

8.3. Pantalla principal de la aplicación

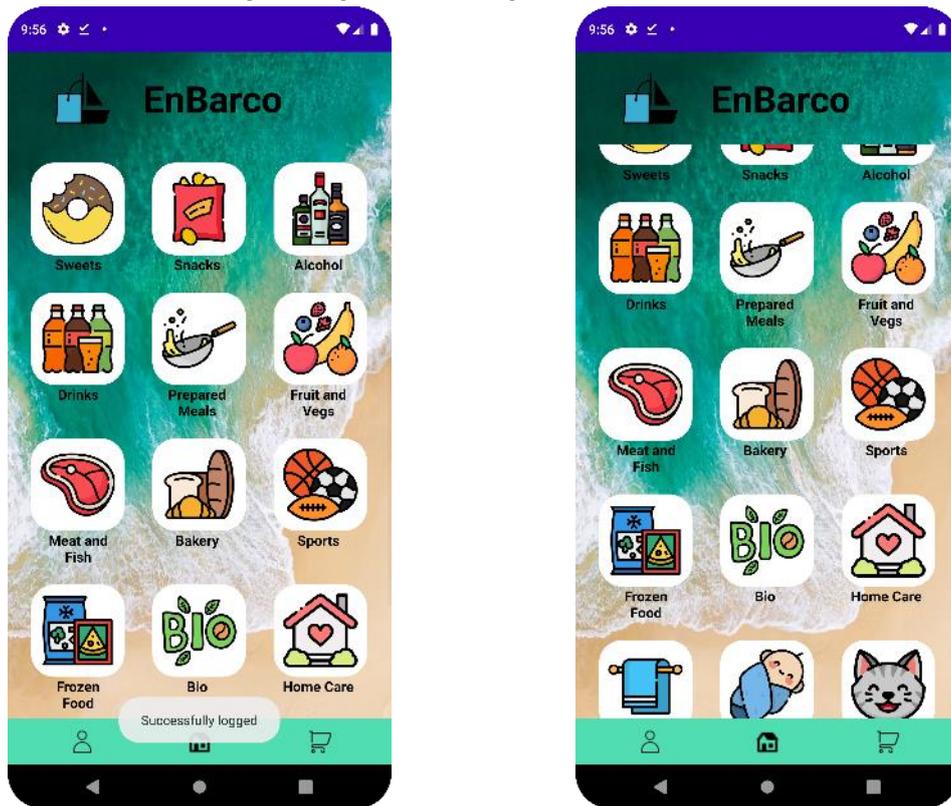


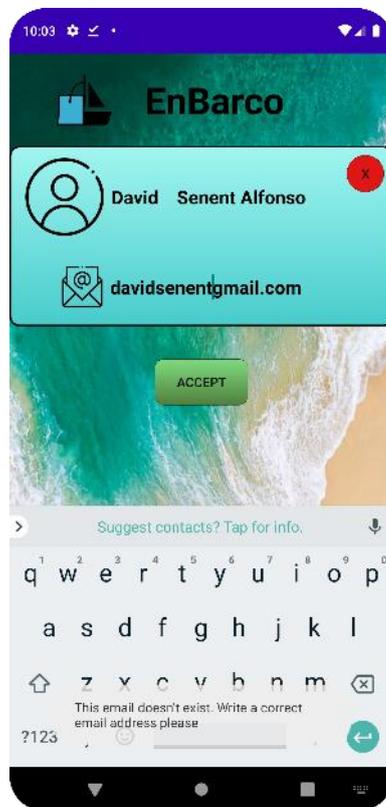
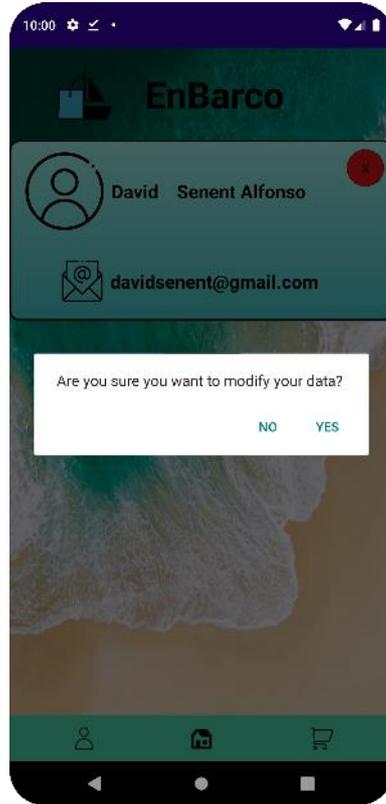
Ilustración 22 - Pantalla principal

8.4. Menú del usuario



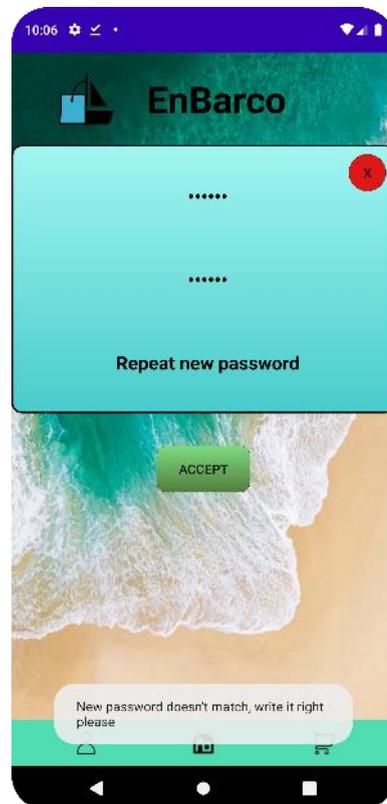
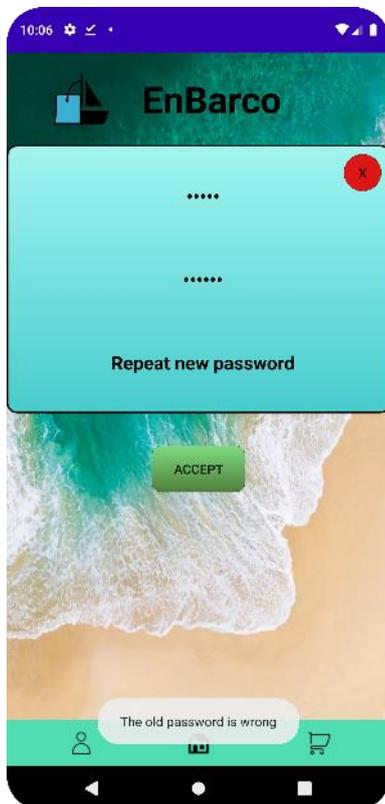
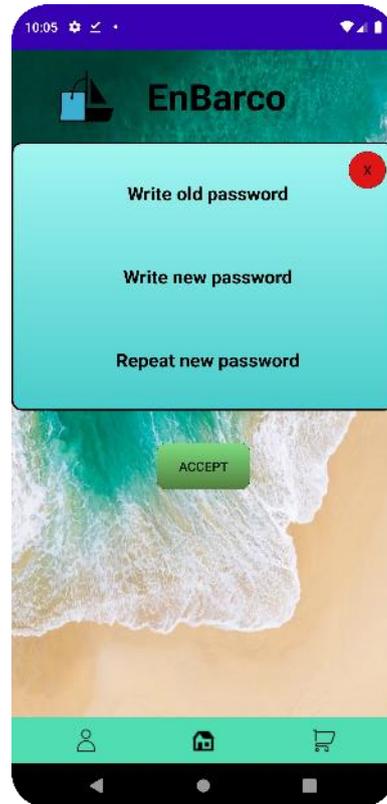
Ilustración 23 - Perfil de usuario

8.5. Modificar información usuario



Desarrollo de una aplicación Android para la empresa EnBarco

Podemos observar más mensajes Toast que proveen consistencia cuando el usuario quiere modificar sus datos y escribe el correo de forma incorrecta o intenta eliminar su nombre o apellidos completos.



Lo mismo ocurre con la contraseña. Se brindan mensajes de apoyo al usuario informando que la información introducida es incorrecta o es necesaria más información que la proporcionada.

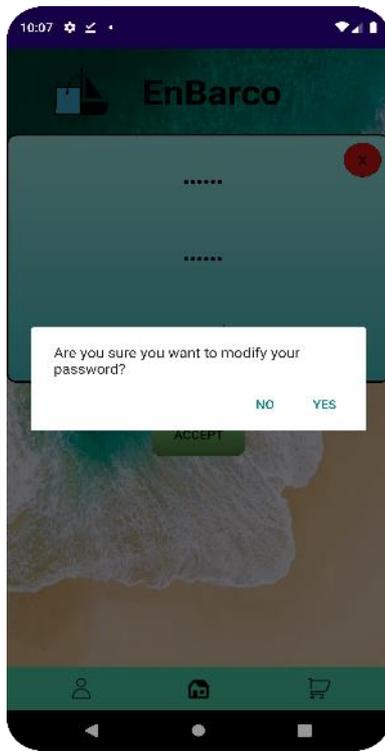


Ilustración 24 - Modificar datos usuario

8.6. Productos favoritos



Desarrollo de una aplicación Android para la empresa EnBarco

Podemos observar layouts de productos deshabilitados. Esto se debe a que las empresas que proveen estos productos se encuentran a una distancia mayor que la máxima permitida, por lo que no es posible la adquisición de estos productos en este momento. La tarjeta se habilitará en cuanto el usuario se encuentre a una distancia adecuada.

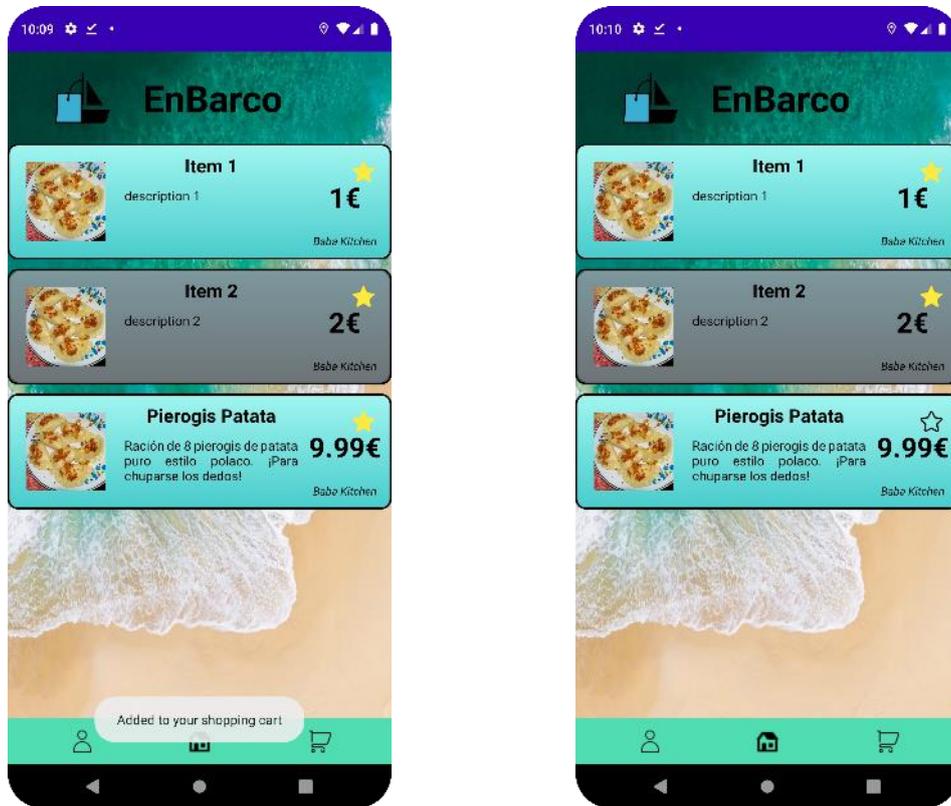
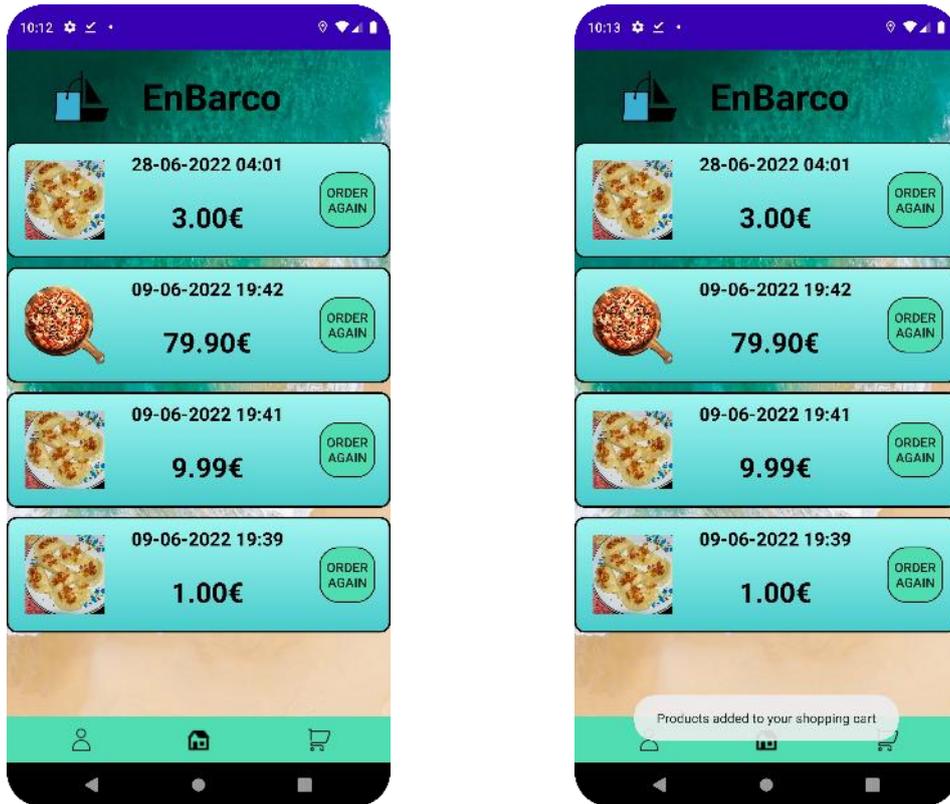


Ilustración 25 - Productos favoritos

8.7. Historial de pedidos



Siguiendo la misma mecánica que con los productos favoritos, no será posible volver a realizar un pedido si alguno de sus productos pertenece a una empresa, la cual se encuentra fuera del alcance del usuario.

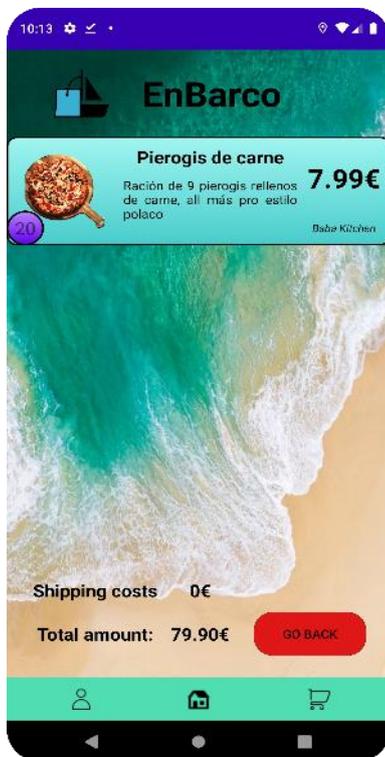


Ilustración 26 - Historial de pedidos

8.8. Resultado de búsqueda

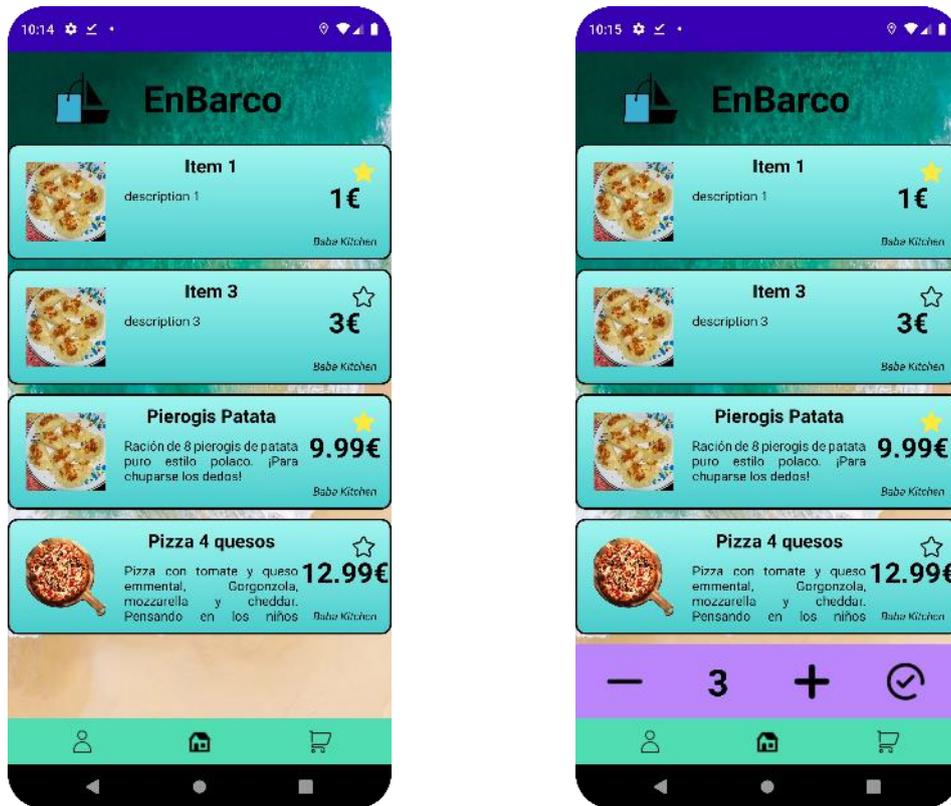


Ilustración 27 - Resultados de búsqueda

8.9. Carrito de la compra

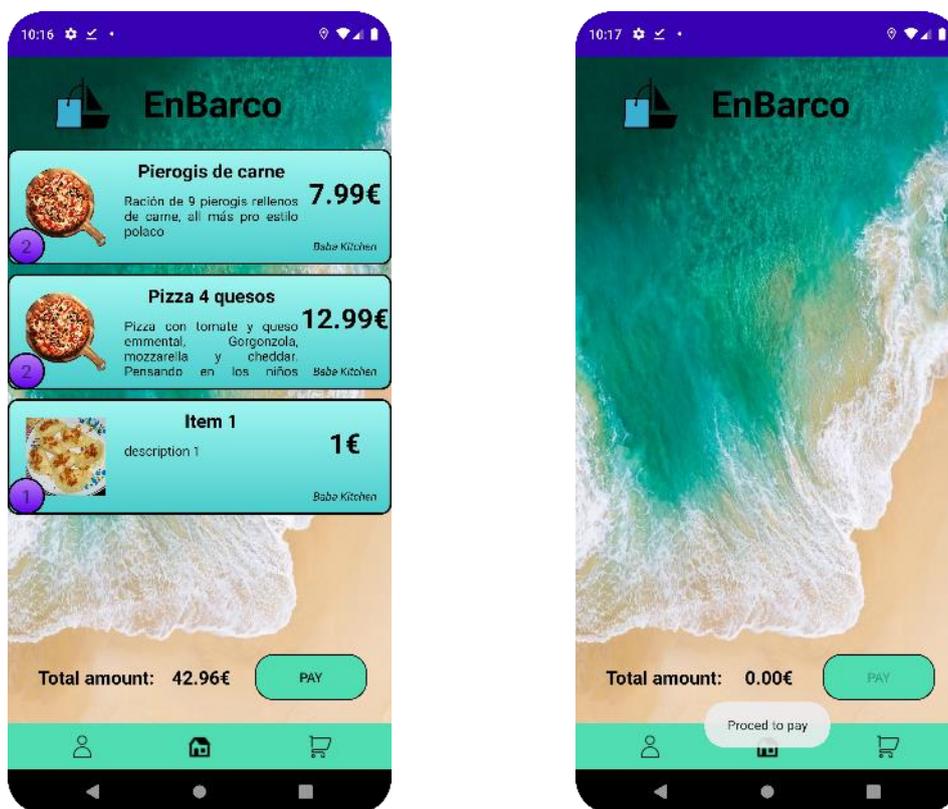


Ilustración 28 - Carrito de productos

Una vez realizado el pago, el botón de pagar se desactiva. Esto ocurre siempre que el carrito de compra se encuentra vacío para evitar que el usuario realice un pedido sin productos.

8.10. Menú de la empresa



Ilustración 29 - Menú de empresa

8.11. Modificar datos de la empresa

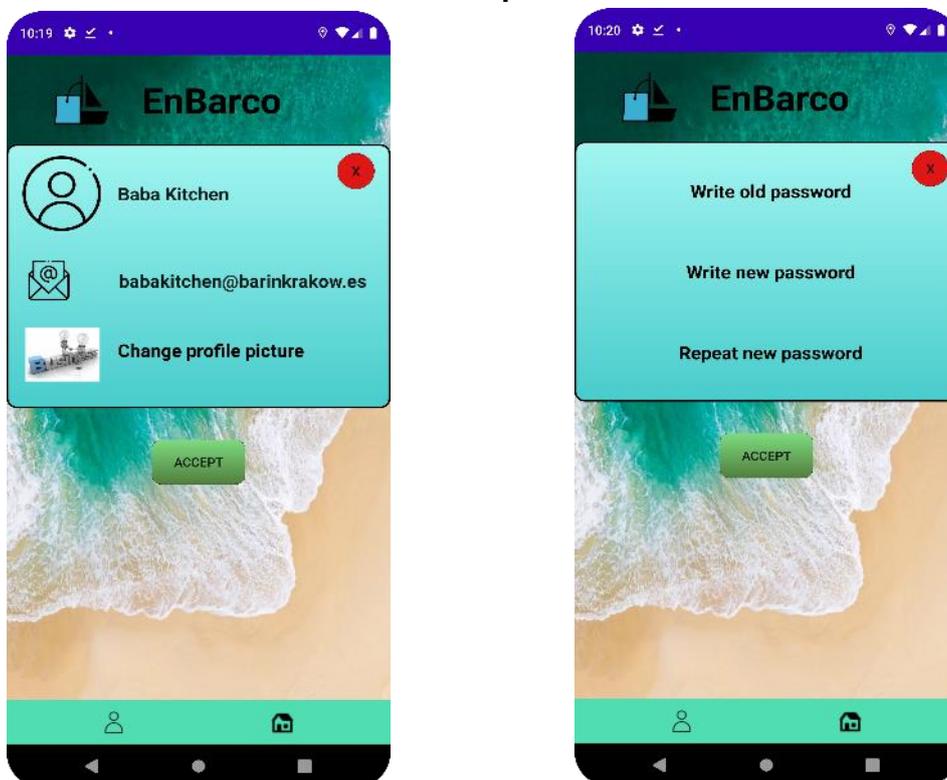


Ilustración 30 - Modificar datos empresa

8.12. Historial de pedido de la empresa

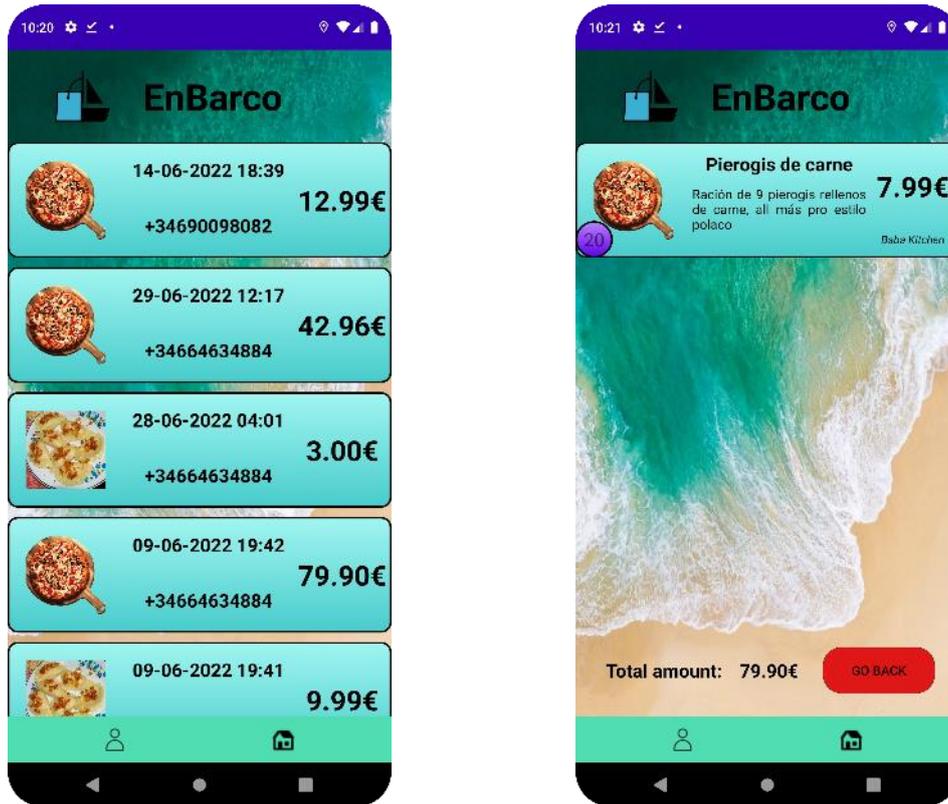


Ilustración 31 - Historial de pedidos de empresa

8.13. Listado de productos de la empresa

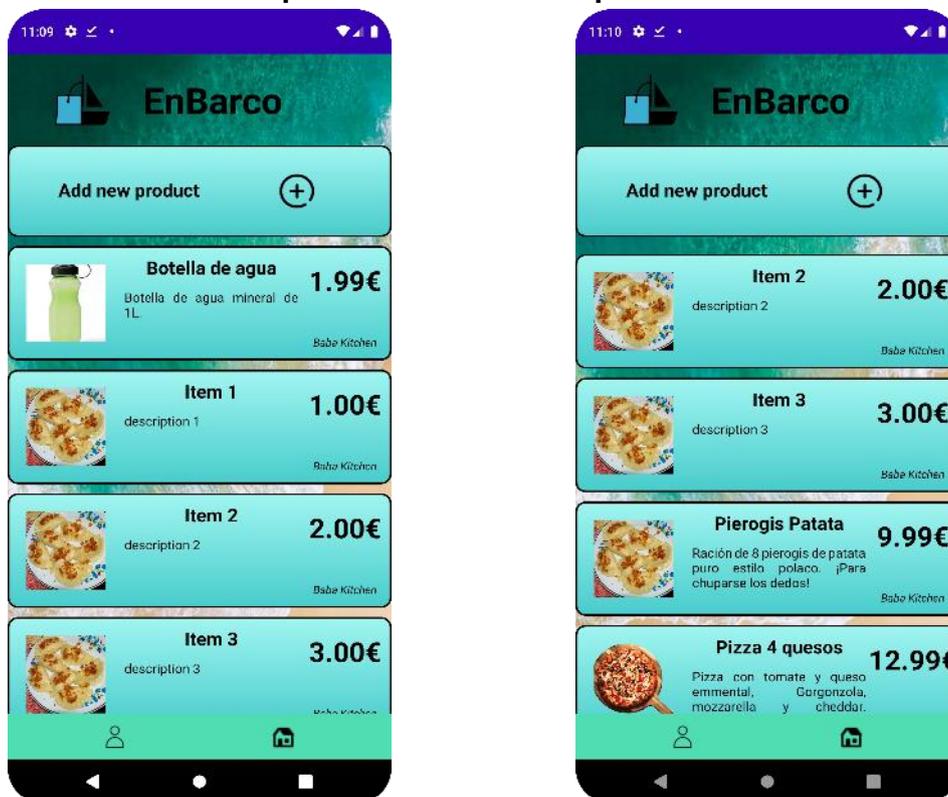


Ilustración 32 - Lista de productos de empresa

8.14. Añadir producto nuevo

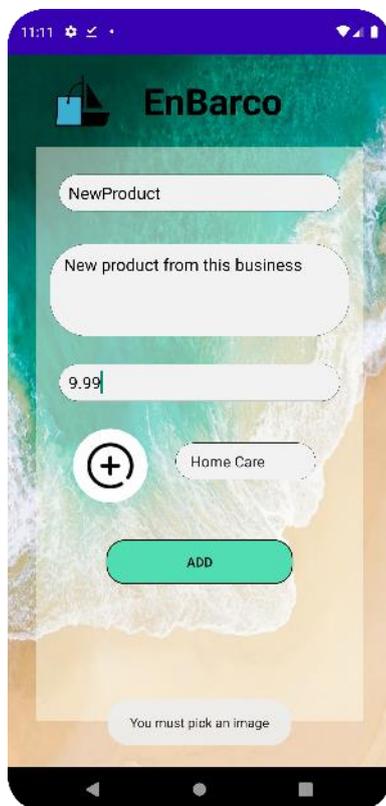
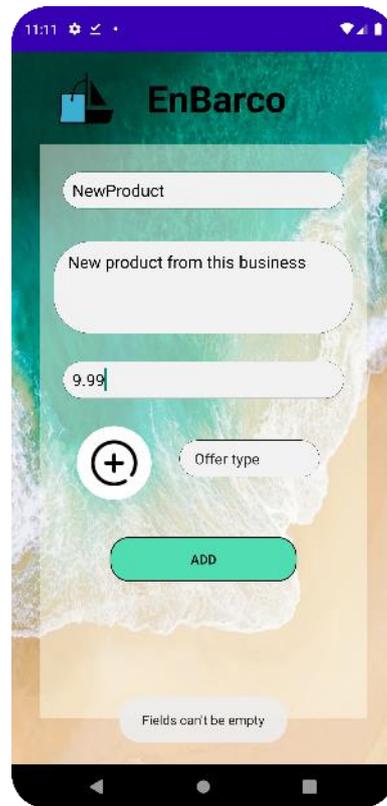
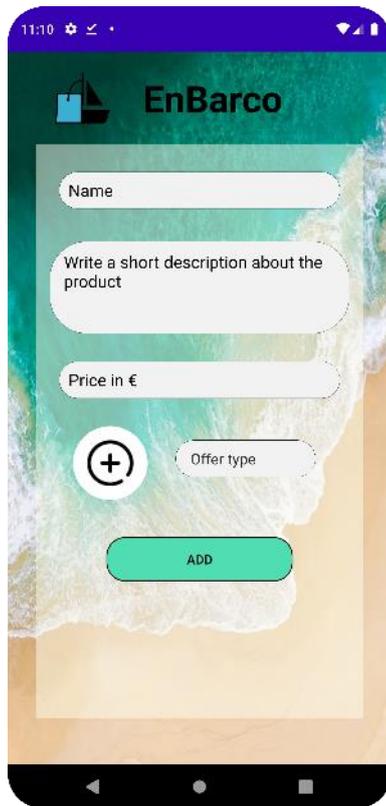


Ilustración 33 - Añadir producto empresa

8.15. Modificar producto

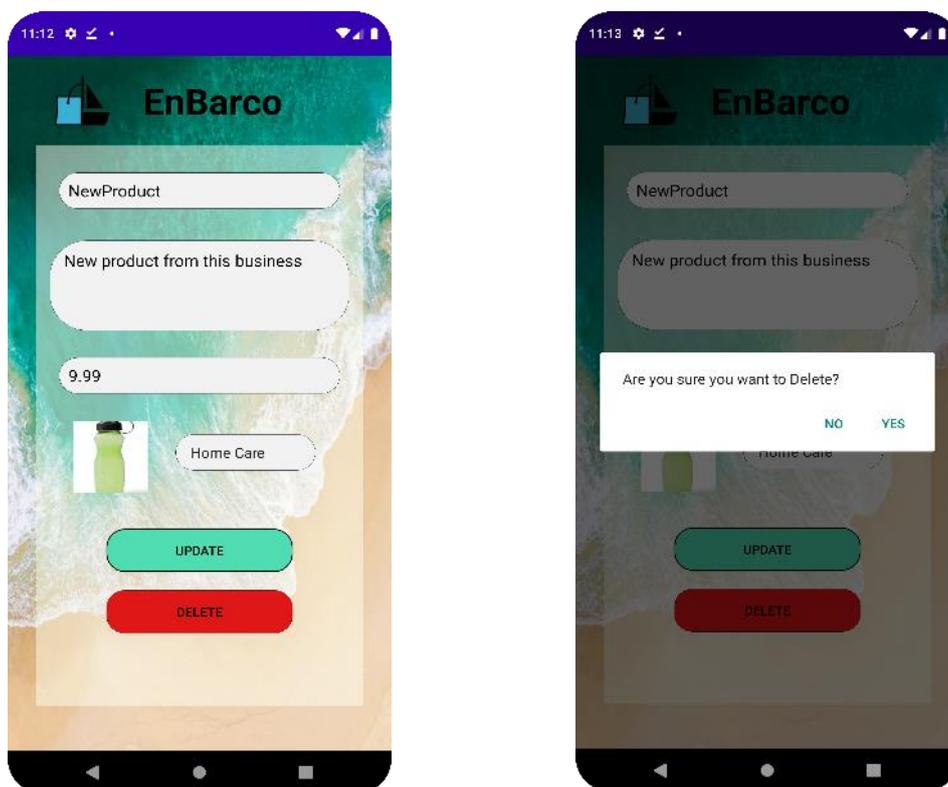


Ilustración 34 - Modificar producto empresa

Al modificar un producto, se cargan automáticamente su nombre, descripción, imagen y tipo en la pantalla, de forma que al modificar el valor de alguno de estos y escoger el botón de "Update", estos valores se modificarán. Si se marca el botón "Delete" y se confirma la decisión, el producto será eliminado de la empresa, pero los pedidos que lo contenían anteriormente lo mantendrán en el historial.

9. Conclusiones

En este capítulo destacaremos problemas que hayan aparecido durante el desarrollo del proyecto, errores cometidos, aprendizaje personal y la relación del trabajo con los estudios cursados. Finalmente, desarrollaremos brevemente las conclusiones personales acerca de la realización de este TFG.

9.1. Problemas encontrados y como se han solucionado

A lo largo del desarrollo de este proyecto se han encontrado varios problemas, los cuales han requerido diseñar algoritmos optimizados, disminuir la calidad de algunos apartados o rediseñar alguna parte de la aplicación para poder solucionarlos.

Entre estos problemas, unos de los principales, el cual ya ha sido mencionado anteriormente, ha sido el diseño de las interfaces. Para solucionar este problema, se ha hecho un estudio acerca de las interfaces utilizadas en aplicaciones similares con el objetivo de diseñar una que pudiese mejorar su estructura. Nuestra tutora nos redirigió a Jorge Serrano Cobos, profesor de la UPV que imparte asignaturas como “Branding personal y corporativo”, “Marketing digital” o “Estrategias de comunicación digital de una empresa”, con la esperanza que nos ayudase a implementar un mejor diseño de interfaces. Este profesor nos encaminó en la dirección correcta, advirtiéndolo de errores en las interfaces actuales, tanto estéticos como funcionales.

Otro problema que encontramos durante el desarrollo fue una sobrecarga de la memoria del dispositivo, causado por nuestro desconocimiento en el desarrollo en Android. Esto ocurrió diseñando el perfil principal, al incluir un número relevante de imágenes de alta calidad en esta ventana. Se produjo por no optimizar el tamaño de las imágenes, provocando así un excesivo nivel de cargar en la memoria RAM en el momento de ejecutar la aplicación. Este fue solucionado tras un estudio acerca del tópico. Escogimos otras imágenes que se adecuasen más al diseño de nuestra aplicación y las transformamos en iconos mediante la herramienta online *Android Asset Studio* [14]. Este servicio nos permite no solo convertir imágenes a iconos, incluyéndolos así de forma adecuada en el proyecto, sino también crear el icono principal de la aplicación, iconos para las notificaciones de nuestra app o diseñar nuestra *Toolbar* o barra de tareas.

9.2. Errores cometidos y como evitarlos

Partiendo de los problemas mencionados en el apartado anterior, uno de los errores que cometimos durante el desarrollo fue no establecer un diseño definitivo de las interfaces, provocando volver a programar el inicio de sesión y registro después de tener que modificar estas interfaces. La mejor forma de evitar esta errata sería establecer la estructura de las interfaces, permitiéndonos modificar pequeños detalles que no influyan en su funcionamiento.

Otro error común en este tipo de proyectos suele ser el de no definir un diagrama de flujo de la aplicación. En nuestro caso no cometimos este error, pero esto suele ocurrir en desarrolladores principiantes y consideramos que es importante mencionarlo. Un correcto diagrama de flujo previo al inicio de la programación nos permite escribir el código sabiendo su funcionalidad exacta y la dirección en la que se desea navegar desde cualquier pantalla de la aplicación.

Un error más, usualmente cometido en el desarrollo de cualquier programa con el uso de metodología ágil, es no dejar un tiempo posterior al último *Sprint* para corregir errores [15]. Al finalizar la primera fase de desarrollo de un programa, siempre ocurren errores y sobreviven algunas funcionalidades incorrectas o erróneas que los programadores no han podido prever. Por esta razón, es necesario dejar un tramo de tiempo entre la finalización del último *Sprint* y la entrega del proyecto, y dedicar este tiempo a realizar pruebas y corregir los errores que aparezcan en estas.

Finalmente, destacaremos otro error bastante usual y el cual sí hemos visto varias veces a lo largo del desarrollo de esta aplicación, y este es el crear funciones excesivamente largas. En la programación las funciones realizan tareas o eventos específicos, y normalmente, si estas son muy largas, se dividen en otras funciones de menor extensión. En muchos casos, al programar una función determinada se estima que esta tenga una extensión menor a que termina teniendo. Esto provoca que, si hay una necesidad de refactorizar esa parte del código un tiempo después de que se haya escrito, se puedan encontrar dificultades para hallar el problema o la parte específica que se desea modificar. Una buena praxis en el ámbito de la programación es la de separar las funciones según el propósito que estas tienen en el código y que su extensión no sea superior de unas diez líneas. Si aplicamos esta recomendación, nuestro código quedará mucho más legible y ordenado y, por tanto, será mucho más sencillo de manipular en el futuro. Después de tratar este tema, concluiremos con una divertida cita famosa del autor y profesor de filosofía estadounidense Martin P.



Golding: *“Programa siempre como si la persona que finalmente mantendrá tu código fuera un psicópata violento que sabe dónde vives”*.

9.3. Aprendizaje del alumno tanto profesional como personalmente

Después de la realización de este proyecto hemos puesto en práctica muchos conocimientos adquiridos durante el estudio de la carrera, pero también hemos adquirido muchos nuevos.

Empezaremos denotando el uso de Java, el cual ha sido aprendido y mejorado durante toda la carrera universitaria. Además de Java, también se ha utilizado Kotlin durante el desarrollo de este proyecto y, aunque no se trata de un lenguaje de programación, ha sido necesario aprender a tratar con los ficheros XML. Este trabajo ha propiciado un incremento enorme en nuestro nivel de conocimiento de estos dos lenguajes de programación, en especial en Kotlin, ya que nunca habíamos utilizado el lenguaje previamente.

También hemos puesto en práctica nuestras nociones acerca del diseño y desarrollo de un proyecto software. Una vez más, estos conocimientos se han incrementado gracias al trabajo en esta aplicación, evitándonos producir errores en el desarrollo de proyectos futuros. Hemos aprendido a manejar y diseñar interfaces de forma que estas cumplan con la funcionalidad deseada y guíen al usuario a través de su paso por la aplicación. También hemos reforzado nuestro conocimiento acerca de la importancia de cumplir las fechas planificadas al inicio del proyecto y de dejar un tramo de tiempo posterior para la solución de errores.

Otro punto a tener en cuenta ha sido familiarizarse con los presupuestos y planes de trabajo de desarrolladores software. Ha sido necesario realizar un pequeño estudio acerca de este tema para poder constatar el costo que habría tenido el desarrollo del proyecto no solo según el tamaño de este, sino también para las horas trabajadas por programador.

10. Trabajos futuros

En el presente capítulo desarrollaremos posibles caminos o líneas de desarrollo posibles en un futuro para el proyecto. También aprovecharemos para mencionar flecos que nos ha sido imposible programar por falta de tiempo o experiencia.

Esperamos que, si en un futuro se consolida la empresa EnBarco y este software tiene continuidad, se puedan aplicar las mejores destacadas en este trabajo. Esto cumpliría definitivamente el objetivo de crear una aplicación que compita e incluso mejore a las de la competencia.

10.1. Posibles flecos interesantes de desarrollar, pero que no ha sido posible

Este apartado cumple con el objetivo de indicar funcionalidades pensadas para la aplicación, pero que no ha sido posible implantar por determinados motivos.

Una de las principales funcionalidades fue la búsqueda de una localización exacta en *Google Maps*. Aunque la aplicación cumple a la perfección con el objetivo deseado, entendemos que al crearse una cuenta como establecimiento o empresa proporcionaría una mayor comodidad poder escribir en una barra de búsqueda la dirección exacta. Nuestro modelo actual obliga a los establecimientos a introducir manualmente el punto exacto en el que se encuentra la empresa. Esto se decidió así por una falta de conocimiento acerca de la API de *Google Maps* y carencia de tiempo para estudiarla en mayor grado.

Otro error detectado a lo largo del desarrollo de la aplicación ha sido no tener en cuenta los horarios de los establecimientos. Nuevamente, por falta de tiempo, no se ha podido solucionar, ya que requeriría un gran nivel de refactorización el en código. Este sería un fleco importante por implantar antes de introducir la aplicación en el mercado.

Adicionalmente, no está entre nuestras prioridades, pero incluir un chat entre el repartidor y el usuario sería un posible añadido más de la aplicación. Esto ayudaría a mejorar la comunicación entre los dos lados y brindaría un mayor nivel de seguridad y confianza al usuario.

También pensamos en proporcionar un mayor grado de personalización al usuario, introduciendo desde la posibilidad de añadir imágenes de perfil o enviar ofertas a su



correo electrónico si así lo desea, hasta incluir algoritmos que recomienden productos teniendo en cuenta los pedidos realizados previamente. La mejora planteada tampoco está entre nuestras mayores prioridades, ya que no influye directamente en el funcionamiento de la aplicación y puede provocar menor sensación de simplicidad, la cual es una de nuestras principales fortalezas.

Finalmente, pensamos en aumentar levemente el grado de confianza de los usuarios, permitiéndoles realizar tareas como buscar si existen determinados productos o cuanto les costaría el envío sin la necesidad de estar registrados. Esto no requiere un grado elevado de refactorización, pero, como hemos comentado anteriormente, no nos ha dado tiempo a poder implementar todos los posibles flecos en los que pensamos a lo largo del desarrollo de la aplicación.

10.2. Líneas de desarrollo abiertas para aplicar estos resultados a otras áreas

Después del desarrollo de esta aplicación, obviamente tenemos los conocimientos necesarios para desarrollar otra aplicación del sector *delivery* para cualquier otra empresa. Además de esto, estamos convencidos de que también hemos reunido los conocimientos requeridos para el desarrollo de aplicaciones de conducción, al estilo de Uber o Cabify, que nos devuelvan la ruta más corta entre dos puntos.

Otra área en la que podríamos realizar un proyecto a la perfección es en la de la administración, puesto que una aplicación sobre, por ejemplo, la gestión de un club de pádel o las plazas ocupadas de un garaje, son de menor complejidad que una del sector de reparto de productos. Adicionalmente, la mayoría de las funcionalidades necesarias para programar una aplicación destinada a la administración ya ha sido desarrollada en este proyecto, por lo que no creemos que nos supondría un gran grado de dificultad.

Finalmente, teniendo en cuenta todos los conocimientos y capacidades adquiridas durante la realización de este trabajo, podríamos trabajar en proyectos de desarrollo de aplicaciones para dispositivos Android. Obviamente, todas las áreas de desarrollo son distintas y necesitaríamos un estudio acerca del área de trabajo que usaríamos, pero después de realizar este estudio y familiarizarnos con el nuevo sector, creemos que nos sería posible trabajar en la mayoría de los trabajos desarrollando software para estos dispositivos.

10.3. Mejoras y ampliaciones tanto de eficiencia como de funcionalidades del trabajo realizado

Además de los posibles flecos mencionados en el apartado anterior, también destacaremos algunas mejoras posibles respecto a la aplicación, las cuales no formaban parte de nuestras futuras mejoras, pero serían interesantes de implantar.

Empezaremos mencionando el registro en la aplicación. Pensamos que sería interesante incluir medios con los que los usuarios ya estén familiarizados para registrarse, como la cuenta de Google o Facebook. Esto les evitaría realizar todo el proceso de registro, pero necesitaríamos tanto que Google nos proporcionase sus librerías, como tener acceso a estas cuentas, cosa que en la actualidad es impensable.

Siguiendo la misma línea, nos gustaría brindar la posibilidad real de realizar los pagos mediante Visa o PayPal, pero estas funcionalidades no se pueden programar debido a que es necesario que dichas empresas nos brinden el código necesario para acceder a sus servicios. Al tratarse de una aplicación que proporciona soporte a una empresa hipotética, no nos es posible conseguir estas librerías. Cabe destacar que incluir estas funcionalidades sin contactar previamente con estas empresas sería una ilegalidad y podrían tomar acciones legales contra nosotros.

Otra funcionalidad posible, pero que no nos incumbe actualmente, sería calcular la ruta más corta entre todos los establecimientos que formen parte de un pedido y el usuario que lo ha realizado. Nuestros repartidores suplen actualmente esta funcionalidad con el uso de *Google Maps*, por lo que, aunque los proporcionará un levemente mayor grado de comodidad, no es estrictamente necesaria la programación de esta.

Para concluir, destacaremos la necesidad de aumentar tanto la capacidad de la base de datos como de gestión de pedidos. La empresa EnBarco está activa en la actualidad únicamente en una cala de Mallorca, por lo que no esperamos un excesivo número de pedidos simultáneos. Según el estudio realizado por mi colaborador Alejandro Sánchez Gómez, esperamos un máximo de entre 20 y 30 pedidos al día, siendo 15 una media aproximada. Sin embargo, la aplicación está preparada para una expansión de la empresa, cumpliendo su correcto funcionamiento en cualquier parte del mundo. Aunque esta esté diseñada de esta forma, el plan actual de la base de datos no soportaría tal nivel de carga, por lo que en el supuesto caso de que la empresa consiguiera un incremento notable en el número de usuarios, sería necesario



aumentar la capacidad de nuestra *base de datos*. De igual forma, actualmente se notifica al único punto de repartidores mediante un correo electrónico indicando los datos de pedido. Si ocurriese este incremento en el número de usuarios, el método actual dejaría de ser útil y aparecería nuevamente una necesidad de cumplir esta funcionalidad. Esto se solucionaría, por ejemplo, asignando un repartidor específico a cada pedido con un correo individual y enviando la información del pedido a este correo. También sería necesario, de la misma forma que un semáforo, bloquear el recibimiento de pedidos de este repartidor hasta que marque como finalizado el que está actualmente realizando.

11. Bibliografía

[1] Agile Methods and the Agile Manifesto. En: *Agile Software Construction* [en línea]. London: Springer London, [sin fecha], pp. 9–30. ISBN 9781852339449 [consultado el 3 de julio de 2022]. Disponible en: doi:10.1007/1-84628-262-4_2

[2] COLABORADORES DE LOS PROYECTOS WIKIMEDIA. Android - Wikipedia, la enciclopedia libre. *Wikipedia, la enciclopedia libre* [en línea]. 12 de noviembre de 2007 [consultado el 25 de junio de 2022]. Disponible en: <https://es.wikipedia.org/wiki/Android>

[3] ADEVA, Roberto. Cómo publicar una app en Google Play y cuánto cuesta. *Cinco Días* [en línea]. 1 de febrero de 2015 [consultado el 20 de mayo de 2022]. Disponible en : https://cincodias.elpais.com/cincodias/2015/02/01/lifestyle/1422792260_243066.html#:~:text=Por%20lo%20tanto,%20el%20precio,tengan%20disponibles%20en%20la%20tienda

[4] HERNÁNDEZ-RAMÍREZ, Luisa Fernanda, Fabián Andrés GIRALDO-GIRALDO y Jonathan Ray CÁRDENAS-CASTRILLÓN. GPLAD: Programación Estructurada Sobre Dispositivos Android. *Lámpsakos* [en línea]. 2012, 1(8), 23. ISSN 2145-4086 [consultado el 3 de julio de 2022]. Disponible en: doi:10.21501/21454086.675

[5] DÜCHTING, Markus, Dirk ZIMMERMANN y Karsten NEBE. Incorporating User Centered Requirement Engineering into Agile Software Development. En: *Human-Computer Interaction. Interaction Design and Usability* [en línea]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 58–67. ISBN 9783540731047 [consultado el 3 de julio de 2022]. Disponible en: doi:10.1007/978-3-540-73105-4_7

[6] PRUITT, John. *The persona lifecycle: Keeping people in mind throughout product design*. Amsterdam: Elsevier, 2006. ISBN 0125662513.

[7] SHYAMALA. Mobile User Interface and Usability Guide. *Shyamala's Blog* [en línea]. 17 de junio de 2010 [consultado el 19 de abril de 2022]. Disponible



en: <https://pshyama.wordpress.com/2010/06&17/mobile-user-interface-and-usability-guide/>

[8] BRHEL, Manuel et al. Exploring principles of user-centered agile software development: A literature review. *Information and Software Technology* [en línea]. 2015, **61**, 163–181. ISSN 0950-5849 [consultado el 3 de julio de 2022]. Disponible en: doi:10.1016/j.infsof.2015.01.004

[9] CHO, Lily. Adopting an Agile Culture A User Experience Team's Journey. En: *2009 Agile Conference (AGILE)* [en línea]. IEEE, 2009. ISBN 9780769537689 [consultado el 3 de julio de 2022]. Disponible en: doi:10.1109/agile.2009.76

[10] LIAO, Sean. Structure Your App. En: *Migrating to Android for iOS Developers* [en línea]. Berkeley, CA: Apress, 2014, pp. 61–171. ISBN 9781484200117 [consultado el 3 de julio de 2022]. Disponible en: doi:10.1007/978-1-4842-0010-0_3

[11] TURRADO, Jorge. ¿Qué tipos de pruebas de software son habituales para un desarrollador? - campusMVP.es. *campusMVP.es* [en línea]. 6 de abril de 2020 [consultado el 3 de julio de 2022]. Disponible en: <https://www.campusmvp.es/recursos/post/que-tipos-de-pruebas-de-software-son-habituales-para-un-desarrollador.aspx>

[12] ENRIQUEZ, Juan Gabriel y Sandra Isabel CASAS. Usabilidad en aplicaciones móviles. *Informes Científicos Técnicos - UNPA* [en línea]. 2014, **5(2)**, 25–47. ISSN

[13] *Characteristics of software quality*. Amsterdam: North-Holland Pub. Co., 1978. ISBN 0444851054.

[14] GOMEZ OLIVER, Salvador. Curso de Programación Android | *sgoliver.net*. *sgoliver.net* [en línea]. 7 de septiembre de 2012 [consultado el 16 de marzo de 2022]. Disponible en: <http://www.sgoliver.net/blog/curso-de-programacion-android/1852-4516> [consultado el 3 de julio de 2022]. Disponible en: doi:10.22305/ict-unpa.v5i2.71

[15] *Agile User Experience Design* [en línea]. Elsevier, 2013. ISBN 9780124159532 [consultado el 3 de julio de 2022]. Disponible en: doi:10.1016/c2011-0-06229-4

[16] MURPHY, Mark L. *The busy coder's guide to Android development*. 2^a ed. [United States]: CommonsWare, 2009. ISBN 9780981678009.



ANEXOS

ANEXO 1

POLÍTICA DE PRIVACIDAD

El siguiente documento corresponde a la Política de Privacidad utilizada en nuestra aplicación, la cual es necesario aceptar para poder crear una cuenta de cualquier tipo:

Privacy Policy

Last updated: March 16, 2022

This Privacy Policy describes Our policies and procedures on the collection, use and disclosure of Your information when You use the Service and tells You about Your privacy rights and how the law protects You.

We use Your Personal data to provide and improve the Service. By using the Service, You agree to the collection and use of information in accordance with this Privacy Policy.

Interpretation and Definitions

Interpretation

The words of which the initial letter is capitalized have meanings defined under the following conditions. The following definitions shall have the same meaning regardless of whether they appear in singular or in plural.

Definitions

For the purposes of this Privacy Policy:

Account means a unique account created for You to access our Service or parts of our Service.

Affiliate means an entity that controls, is controlled by or is under common control with a party, where "control" means ownership of 50% or more of the shares, equity interest or other securities entitled to vote for election of directors or other managing authority.

Application means the software program provided by the Company downloaded by You on any electronic device, named EnBarco

Company (referred to as either "the Company", "We", "Us" or "Our" in this Agreement) refers to EnBarco, Avenida Primavera 26, Colonia de Sant Jordi.

Country refers to: Spain

Device means any device that can access the Service such as a computer, a cellphone or a digital tablet.

Personal Data is any information that relates to an identified or identifiable individual.

Service refers to the Application.

Service Provider means any natural or legal person who processes the data on behalf of the Company. It refers to third-party companies or individuals employed by the Company to facilitate the Service, to provide the Service on behalf of the Company, to perform services related to the Service or to assist the Company in analyzing how the Service is used.

Usage Data refers to data collected automatically, either generated by the use of the Service or from the Service infrastructure itself (for example, the duration of a page visit).

You means the individual accessing or using the Service, or the company, or other legal entity on behalf of which such individual is accessing or using the Service, as applicable.

Collecting and Using Your Personal Data

Types of Data Collected

Personal Data

While using Our Service, We may ask You to provide Us with certain personally identifiable information that can be used to contact or identify You. Personally identifiable information may include, but is not limited to:

- Email address
- First name and last name
- Phone number
- Usage Data

Usage Data

Usage Data is collected automatically when using the Service.

Usage Data may include information such as Your Device's Internet Protocol address (e.g. IP address), browser type, browser version, the pages of our Service that You visit, the time and date of Your visit, the time spent on those pages, unique device identifiers and other diagnostic data.

When You access the Service by or through a mobile device, We may collect certain information automatically, including, but not limited to, the type of mobile device You use, Your mobile device unique ID, the IP address of Your mobile device, Your mobile operating system, the type of mobile Internet browser You use, unique device identifiers and other diagnostic data.

We may also collect information that Your browser sends whenever You visit our Service or when You access the Service by or through a mobile device.

Information Collected while Using the Application

While using Our Application, in order to provide features of Our Application, We may collect, with Your prior permission:

- Information regarding your location
- Pictures and other information from your Device's camera and photo library

We use this information to provide features of Our Service, to improve and customize Our Service. The information may be uploaded to the Company's servers and/or a Service Provider's server or it may be simply stored on Your device.

You can enable or disable access to this information at any time, through Your Device settings.

Use of Your Personal Data

The Company may use Personal Data for the following purposes:

To provide and maintain our Service, including to monitor the usage of our Service.

To manage Your Account: to manage Your registration as a user of the Service. The Personal Data You provide can give You access to different functionalities of the Service that are available to You as a registered user.

For the performance of a contract: the development, compliance and undertaking of the purchase contract for the products, items or services You have purchased or of any other contract with Us through the Service.

To contact You: To contact You by email, telephone calls, SMS, or other equivalent forms of electronic communication, such as a mobile application's push notifications regarding updates or informative communications related to the functionalities, products or contracted services, including the security updates, when necessary or reasonable for their implementation.

To provide You with news, special offers and general information about other goods, services and events which we offer that are similar to those that you have already purchased or enquired about unless You have opted not to receive such information.

To manage Your requests: To attend and manage Your requests to Us.

For business transfers: We may use Your information to evaluate or conduct a merger, divestiture, restructuring, reorganization, dissolution, or other sale or transfer of some or all of Our assets, whether as a going concern or as part of bankruptcy, liquidation, or similar proceeding, in which Personal Data held by Us about our Service users is among the assets transferred.

For other purposes: We may use Your information for other purposes, such as data analysis, identifying usage trends, determining the effectiveness of our promotional campaigns and to evaluate and improve our Service, products, services, marketing and your experience.

We may share Your personal information in the following situations:

- **With Service Providers:** We may share Your personal information with Service Providers to monitor and analyze the use of our Service, to contact You.
- **For business transfers:** We may share or transfer Your personal information in connection with, or during negotiations of, any merger, sale of Company assets, financing, or acquisition of all or a portion of Our business to another company.
- **With Affiliates:** We may share Your information with Our affiliates, in which case we will require those affiliates to honor this Privacy Policy. Affiliates include Our parent company and any other subsidiaries, joint venture partners or other companies that We control or that are under common control with Us.
- **With business partners:** We may share Your information with Our business partners to offer You certain products, services or promotions.
- **With other users:** when You share personal information or otherwise interact in the public areas with other users, such information may be viewed by all users and may be publicly distributed outside.



- **With Your consent:** We may disclose Your personal information for any other purpose with Your consent.

Retention of Your Personal Data

The Company will retain Your Personal Data only for as long as is necessary for the purposes set out in this Privacy Policy. We will retain and use Your Personal Data to the extent necessary to comply with our legal obligations (for example, if we are required to retain your data to comply with applicable laws), resolve disputes, and enforce our legal agreements and policies.

The Company will also retain Usage Data for internal analysis purposes. Usage Data is generally retained for a shorter period of time, except when this data is used to strengthen the security or to improve the functionality of Our Service, or We are legally obligated to retain this data for longer time periods.

Transfer of Your Personal Data

Your information, including Personal Data, is processed at the Company's operating offices and in any other places where the parties involved in the processing are located. It means that this information may be transferred to — and maintained on — computers located outside of Your state, province, country or other governmental jurisdiction where the data protection laws may differ than those from Your jurisdiction.

Your consent to this Privacy Policy followed by Your submission of such information represents Your agreement to that transfer.

The Company will take all steps reasonably necessary to ensure that Your data is treated securely and in accordance with this Privacy Policy and no transfer of Your Personal Data will take place to an organization or a country unless there are adequate controls in place including the security of Your data and other personal information.

Disclosure of Your Personal Data

Business Transactions

If the Company is involved in a merger, acquisition or asset sale, Your Personal Data may be transferred. We will provide notice before Your Personal Data is transferred and becomes subject to a different Privacy Policy.

Law enforcement

Under certain circumstances, the Company may be required to disclose Your Personal Data if required to do so by law or in response to valid requests by public authorities (e.g. a court or a government agency).

Other legal requirements

The Company may disclose Your Personal Data in the good faith belief that such action is necessary to:

- Comply with a legal obligation
- Protect and defend the rights or property of the Company



- Prevent or investigate possible wrongdoing in connection with the Service
- Protect the personal safety of Users of the Service or the public
- Protect against legal liability

Security of Your Personal Data

The security of Your Personal Data is important to Us, but remember that no method of transmission over the Internet, or method of electronic storage is 100% secure. While We strive to use commercially acceptable means to protect Your Personal Data, We cannot guarantee its absolute security.

Children's Privacy

Our Service does not address anyone under the age of 13. We do not knowingly collect personally identifiable information from anyone under the age of 13. If You are a parent or guardian and You are aware that Your child has provided Us with Personal Data, please contact Us. If We become aware that We have collected Personal Data from anyone under the age of 13 without verification of parental consent, We take steps to remove that information from Our servers.

If We need to rely on consent as a legal basis for processing Your information and Your country requires consent from a parent, We may require Your parent's consent before We collect and use that information.

Links to Other Websites

Our Service may contain links to other websites that are not operated by Us. If You click on a third party link, You will be directed to that third party's site. We strongly advise You to review the Privacy Policy of every site You visit.

We have no control over and assume no responsibility for the content, privacy policies or practices of any third party sites or services.

Changes to this Privacy Policy

We may update Our Privacy Policy from time to time. We will notify You of any changes by posting the new Privacy Policy on this page.

We will let You know via email and/or a prominent notice on Our Service, prior to the change becoming effective and update the "Last updated" date at the top of this Privacy Policy.

You are advised to review this Privacy Policy periodically for any changes. Changes to this Privacy Policy are effective when they are posted on this page.

Contact Us

If you have any questions about this Privacy Policy, You can contact us:

- By email: administracion@enbarco.com



ANEXO 2

OBJETIVOS DE DESARROLLO SOSTENIBLE

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	Alto	Medio	Bajo	No Procede
ODS 1. Fin de la pobreza.				X
ODS 2. Hambre cero.			X	
ODS 3. Salud y bienestar.				X
ODS 4. Educación de calidad.				X
ODS 5. Igualdad de género.				X
ODS 6. Agua limpia y saneamiento.				X
ODS 7. Energía asequible y no contaminante.		X		
ODS 8. Trabajo decente y crecimiento económico.	X			
ODS 9. Industria, innovación e infraestructuras.	X			
ODS 10. Reducción de las desigualdades.				X
ODS 11. Ciudades y comunidades sostenibles.		X		
ODS 12. Producción y consumo responsables.			X	
ODS 13. Acción por el clima.			X	
ODS 14. Vida submarina.		X		
ODS 15. Vida de ecosistemas terrestres.			X	
ODS 16. Paz, justicia e instituciones sólidas.				X
ODS 17. Alianzas para lograr objetivos.				X

Reflexión sobre la relación del TFG/TFM con los ODS y con el/los ODS más relacionados.

Los Objetivos de Desarrollo sostenible (ODS) conforman una petición universal con los objetivos de poner fin a la pobreza, proteger el planeta y mejorar las vidas y las perspectivas de las personas en todo el mundo. En 2015, todos los Estados Miembros de las Naciones Unidas aprobaron 17 objetivos como parte de la Agenda 2030 para el Desarrollo Sostenible, en la cual se establece un plan para alcanzar los Objetivos en 15 años.

Dicha agenda está compuesta por los 17 ODS y a su vez 169 metas, para conseguir lograr los objetivos el plan previsto para 2030, se fundamenta en 5 dimensiones, conocidas como las 5P:

- **Personas:** El primero de estos objetivos es terminar con la pobreza, el hambre y que todas las personas tengan la posibilidad de desarrollarse en una sociedad igual y justa.
- **Planeta:** Buscar políticas sostenibles que ayuden a frenar el cambio climático y contribuyan a conservar y proteger el planeta.
- **Prosperidad:** Lograr que todas las personas puedan desarrollarse y tener una vida plena y que este proceso vaya alineado con la naturaleza.
- **Paz:** Conseguir una sociedad igualitaria, libre de temor, violencia y guerra.
- **Participación colectiva:** Contribuir colectivamente para alcanzar los objetivos 2030 a través de todos los medios posibles.

Aquí podemos observar una tabla con los 17 ODS:



Después de estudiar los ODS observamos que EnBarco cumple con algunos de estos objetivos 2030, ya que tiene una visión compartida en cuanto a la importancia del medio ambiente y el desarrollo económico y social para conseguir prosperidad.

EnBarco se identifica principalmente con los siguientes ODS:

- **7 energía asequible y no contaminante:** Como hemos mencionado anteriormente una de las cinco dimensiones es el **planeta** y EnBarco tiene integrada esta preocupación entre sus principales valores. Debido a esto, contamos con el objetivo de evitar generar gases de efecto invernadero a través de la utilización de energías limpias. Por este motivo, la entrega de productos se lleva a cabo mediante bicicletas en el medio terrestre y con motores eléctricos en el marítimo. Con estas medidas contribuiremos a lograr los objetivos previstos y conservar el planeta.
- **8 trabajo decente y crecimiento económico:** Uno de los puntos clave de nuestra organización es crear un mayor desarrollo económico de la zona de implantación del modelo de negocio, ampliando el alcance de los negocios locales con este nuevo servicio. Así mismo, esto genera mayor prosperidad tanto al negocio local como a los empleados de la organización, fomentando así el desarrollo económico de la zona.

Por último, destacaremos la importancia de estos objetivos y que su cumplimiento es obligación de todos y de manera conjunta se pueden conseguir con pequeñas acciones como las implantadas por EnBarco.

ANEXO 3

ANÁLISIS DE REQUISITOS

Aquí se muestra el análisis de requisitos realizado previo al desarrollo de la aplicación. Este fue uno de los primeros pasos tomados y, por tanto, uno de los primeros documentos generados del proyecto. Creemos que dada su importancia es necesario incluirlo en un anexo para poder facilitar su consulta.

Análisis de requisitos de EnBarco

Alcance

Esta aplicación permite dotar a la hipotética empresa EnBarco de un software funcional para poder cumplir el propósito de la empresa, el cual se explica en puntos posteriores.

Está planeada desarrollarse en 3 meses aproximadamente siguiendo una metodología ágil, la cual está explicada en el punto ‘Plan de trabajo del proyecto’.

Propósito

La aplicación debe permitir a los usuarios poder pedir y recibir comida o materiales desde las zonas marítimas cercanas a la costa.

Se espera que después de un registro y el pago, estos reciban su pedido en la ubicación indicada.

Información fundamental

Aplicaciones existentes como Uber Eats, Getir o Deliveroo ya cumplen con la función que intentamos conseguir en territorios urbanos, así que uno de los propósitos para la aplicación sería replicar la funcionalidad de estas aplicaciones actualmente muy usadas cambiando el público y zonas destino, siendo estos, como ya hemos indicado anteriormente, gente que se encuentre en embarcaciones cercanas a la costa.

Definición del sistema



La entrada inicial para un usuario nuevo será la página de registro de la aplicación. Desde esta página podrá introducir sus datos personales, los cuales son: número de teléfono, nombre completo y correo electrónico.

Una vez haya introducido sus datos, deberá aceptar tanto la ‘Política de Privacidad’, como los ‘Términos y Condiciones’ propuestos.

Finalmente podrá clicar en registrarse, lo que le creará una cuenta en el sistema y guardará sus datos para próximos usos.

Desde la misma página inicial, si el usuario ya dispone de una cuenta registrada previamente, puede iniciar sesión en el sistema.

Posteriormente se le dirige a la página principal de la aplicación, desde la cual se puede añadir dirección de entrega, apareciendo por defecto la ubicación actual del dispositivo en caso de que esta esté activa. Si esta dirección es accesible, se indicará el precio y tiempo estimados de entrega, mientras que, si nos es imposible entregar en la dirección seleccionada, simplemente aparecerá un mensaje avisando de esta situación.

También es posible escoger el establecimiento desde el cual se quiere realizar el pedido, apareciendo los que se encuentran actualmente disponibles en una lista.

Clicando en cualquiera de estos establecimientos, se pueden marcar los productos que se desea recibir para añadirlos a un carrito de compra. Escoger un producto, además de actualizar el carrito, actualizará el precio actual de la compra.

Es posible seleccionar el carrito desde la parte superior derecha de la pantalla, donde aparece tanto su icono como el precio actual de la compra. Desde esta pestaña se puede revisar los productos seleccionados junto a su precio y la cantidad, para poder cerciorarnos de que los hemos escogido correctamente. También es posible proceder al pago clicando en el botón correspondiente.

Si escogemos esta opción, podremos ver en pantalla otra escena desde la cual es posible escoger el método de pago, añadir notas al envío, indicar el código promocional que se desea usar, si es que se dispone de uno, revisar por última vez tanto los productos como la dirección de entrega y finalmente pagar para proceder a la entrega. Justo antes de que se realice el pago, se vuelven a indicar los ‘Términos y Condiciones de Servicio’, los cuales deben volver a aceptarse para que la entrega se pueda realizar.

Volviendo a la página principal, hay una última opción disponible desde la cual puedes acceder a tu cuenta clicando en el icono de perfil. Esto desplegará una escena desde la cual puedes tanto modificar tus datos personales, a excepción del teléfono móvil, cómo escoger opciones personalizadas. Entre estas se encuentran ‘Productos favoritos’, ‘Pedidos anteriores’, ‘Ayuda’ y ‘Desconectarse’.

Cómo indican los propios nombres, las opciones son atajos para poder pedir determinados productos con mayor facilidad y rapidez, enviar un correo de ayuda a los técnicos de la aplicación y cerrar sesión en este dispositivo.

Requerimientos técnicos

Se prevé usar el lenguaje de programación Kotlin para el desarrollo esta aplicación, dado que este nos dota de una mayor facilidad y velocidad a la hora de escribir el código que otros lenguajes más complejos, además de un gran grado de eficiencia.

Sumado a esto, este lenguaje es uno de los más usado en el desarrollo de aplicaciones Android, lo que ayuda enormemente tanto al desarrollo cómo al mantenimiento gracias a la gran cantidad de programadores que dominan este lenguaje.

También vamos a ayudarnos la API de Google FireBase Realtime Database para crear la base de datos, siendo esta fácil de usar y de coste gratuito.

Plan de trabajo del proyecto

Escogiendo como fecha final el domingo 26 de junio, el desarrollo de esta aplicación seguirá una metodología ágil y constará de 6 *Sprints* o etapas, iniciando con el Sprint 0, el cual se encargará de realizar el trabajo previo a la programación de la aplicación, permitiéndonos así tener una idea clara de que es lo que vamos a programar y dotándonos de un guion que seguir para poder llevar un progreso constante.

Vamos a utilizar la página web trello.com para crear un tablero Kanban y así llevar constancia del trabajo tanto realizado, cómo por realizar. Aquí iniciaremos con un '*Brainstorm*' que siga el organigrama de la aplicación para escoger las posibles unidades a programar en el futuro, seleccionando los días previos al inicio de cada Sprint las unidades correspondientes a programar en este término de tiempo.

También aplicaremos pruebas de aceptación al finalizar cada Sprint para comprobar el correcto funcionamiento del código escrito. En caso positivo, estas quedarán aceptadas y la unidad se marcará como finalizada, mientras que, en caso de fallo, la unidad volverá a fase de desarrollo para corregirla o, a falta de tiempo, se pasará al siguiente Sprint con un grado de importancia mayor.

ANEXO 4

TEST DE USABILIDAD

Modelo de Test de Usuario

Prueba de Usabilidad

Aplicación móvil: EnBarco

Entrevistador: David Senent Alfonso

PRESENTACIÓN

Buenos días, le agradezco su disposición de participar en esta “Prueba de Usabilidad” que nos ayudará a evaluar la aplicación móvil que desarrollamos y detectar posibles fallos que ésta tenga.

Comenzaremos con una breve presentación que no permita saber quién es usted y cómo utiliza internet.

Estas preguntas se deben hacer cuando el usuario está mirando la pantalla inicial y antes de comenzar a navegar o hacer “click” sobre algún contenido.

1. ¿Es suficiente la información que se ofrece en pantalla para saber a qué institución corresponde el sitio?
2. ¿De los elementos que muestra esta pantalla, hay algo que usted crea que está fuera de lugar?
3. ¿Qué colores diría que son los predominantes en la pantalla?
4. ¿Hacia qué tipo de audiencia cree usted que está dirigido este sitio? ¿Por qué?

Tareas específicas

A continuación, le pediremos que nos ayude a revisar algunos aspectos particulares del sitio, realizando tres tareas específicas. Le recordamos manifestar en voz alta sus impresiones durante este proceso.

Tarea 1: Si tuvieras que crearte una cuenta como usuario y cerrar la sesión posteriormente, ¿cómo lo harías?

Pregunta: ¿Le fue sencillo realizar la tarea? ¿Qué elementos le ayudaron a encontrar la forma de realizarla?

Tarea 2: Guarde en favoritos los productos que desea pedir y realice desde la pestaña favoritos un pedido.

Pregunta: ¿Se sintió natural guardar los productos como favoritos y acceder a estos posteriormente? ¿Le resultó simple realizar el pedido?

Tarea 3: Ahora ingrese sesión con la cuenta de empresa con credenciales Usuario:123123123, Contraseña: 123123. Añada un producto y posteriormente modifique su precio al doble del valor inicial.

Pregunta: ¿Fue intuitivo realizar la tarea? ¿Encontró algún elemento fuera de lugar o tedioso?

Tarea 4: Ahora consiga encontrar el último pedido realizado a esta empresa.

Pregunta: ¿Encuentra suficiente el contenido proporcionado acerca del pedido realizado?

Fin de tareas específicas



Navegación

Estas preguntas se deben hacer luego de permitir al usuario navegar el sitio, con el fin de que se forme una opinión acerca de lo que está viendo y la forma de navegar por sus contenidos

Contenidos:

1. ¿Le parece adecuada la selección de contenidos para el registro en la aplicación? ¿Eché en falta alguna área de información? ¿Le parece que sobra algo?
2. Al hacer click en los contenidos del menú, ¿encontró la información que esperaba o los contenidos a los que deseaba llegar?
3. En general, ¿considera que los nombres de los enlaces son suficientemente descriptivos de lo que se ofrece en las páginas hacia las cuales llevan? ¿Hubo alguno que lo confundió?
4. En caso de haber información relacionada con lo que estaba viendo, ¿se le ofreció de manera simple o tuvo que volver a navegar para encontrarla?
5. En general, ¿cree que los textos introductorios a información, como títulos, descripciones, botones y enlaces son claros y concisos? ¿Le costó leerlos?

Cierre

1. ¿Qué aspecto considera que podría incluir o mejorar respecto al funcionamiento de la aplicación?
2. Si tuviera que calificar la aplicación móvil, ¿qué nota le pondría del 1 al 10?

Gracias por realizar la encuesta.