Mercapp: Estudio y desarrollo de un prototipo que centraliza y aporta visibilidad a los mercados de la ciudad

Daniel Luis Garcia

Resumen– El proyecto Mercapp nace a partir de una descentralización y falta de información relacionada con los mercados y sus comercios, además de la pérdida de visibilidad que están sufriendo durante los últimos años. El objetivo principal se basa en desarrollar un prototipo de aplicación móvil multiplataforma que sea capaz de centralizar aquella información relevante y de interés para los usuarios y, de esta forma, ayudar a incrementar la visibilidad del comercio local y de proximidad. El sistema ha sido desarrollado con el kit de desarrollo de *software* Flutter [1], el lenguaje de programación Dart [2] y enfocado en el diseño centrado en el usuario [3], siguiendo una metodología de desarrollo iterativa e incremental [4]. Como resultado del proyecto, se ha obtenido un prototipo de aplicación que satisface las necesidades de los usuarios involucrados. Al mismo tiempo, un alto cargo del comercio de proximidad local está interesado en la continuidad del desarrollo del proyecto y espera su presentación en el Institut de Mercats Municipals de Barcelona [5].

Palabras clave– centralizar, comercios, Dart, Flutter, mercados, Mercapp, prototipo, visibilidad.

Abstract– The Mercapp project was born out of a decentralisation and lack of information related to markets and their businesses, as well as the loss of visibility they have suffered in recent years. The main objective is based on developing a prototype of a multi-platform mobile application capable of centralising relevant information of interest to users and, in this way, helping to increase the visibility of local and proximity commerce. The system has been developed with the Flutter software development kit [1], the Dart programming language [2] and focused on user-centred design [3], following an iterative and incremental development methodology [4]. As a result of the project, a prototype application has been obtained that meets the needs of the users involved. At the same time, a manager of the proximity trade sector is interested in the continuity of the project after the publishing of this thesis and is waiting for its presentation at the Institut de Mercats Municipals de Barcelona [5].

Keywords– centralising, commerces, Dart, Flutter, markets, Mercapp, prototype, visibility.

1 Introducción

La pandemia que azota el planeta desde el año pasado ha traído consigo un confinamiento y unas restricciones de movilidad severas que, sin lugar a duda, han perjudicado a los mercados, a los comerciantes y al consumo de proximidad. Dos semanas después de que se decretase el estado de alarma, los mercados de Barcelona comenzaron a potenciar y reforzar los servicios de reparto a domicilio de la compra en línea, con motivo de reducir los desplazamientos de los ciudadanos a los mercados de la ciudad condal, y por consiguiente, evitar aglomeraciones innecesarias que ponen en riesgo a la población [6]. Otro de los motivos clave por lo que los mercados comenzaron a potenciar y reforzar esos servicios es para no perder visibilidad y continuar con la generación de ingresos a través de sus ventas, las cuales decrementaron notablemente.

El proyecto Mercapp es concebido a partir de una necesidad real, los mercados de Barcelona y sus comercios necesitan ganar visibilidad de calidad sin dejar de ofrecer sus servicios a la ciudadanía. Este proyecto pretende centralizar toda la información relevante y de interés tanto para clientes

• E-mail de contacto: Daniel.Luis@e-campus.uab.cat
• Mención realizada: Ingeniería del *Software*
• Trabajo tutorizado por: Helena Bolta Torrell (Ciencias de la Computación)
• Curso 2020/2021

habituales como para los potenciales.

Paralelamente, toda información estará sometida a revisiones periódicas con motivo de ofrecer a los usuarios información veraz y precisa.

2 OBJETIVOS

El objetivo principal de este proyecto se basa en elaborar el prototipo de un sistema para dispositivos móviles tanto Android como iOS para que aporte visibilidad a los mercados de la ciudad de Barcelona a la vez que centraliza toda la información que pueda ser relevante o de interés para la población.

A continuación, se muestran los objetivos a alcanzar a largo plazo con la elaboración de este proyecto.

- Centralizar toda la información relevante de los mercados alimenticios de la ciudad junto con sus comercios.
- Aumentar la visibilidad de los mercados alimenticios de la ciudad y de sus comercios.
- Mejorar la percepción de calidad de los clientes de los mercados a partir de la centralización de la información de estos.
- Incrementar el número de consumidores de los mercados de la ciudad.
- Contribuir a reducir la incertidumbre económica de los mercados.

Una vez finalizado el proyecto, se esperan conseguir los siguientes beneficios:

- Centralización de la información relevante y de interés para los comerciantes y consumidores.
- Incremento de la visibilidad de los mercados y de los comercios durante y después esta época azotada por el Sars-Cov-2.
- Inclusión de todos y cada uno de los mercados y comercios sin necesidad de que inviertan capital.
- Descentralizar el consumo de alimentos en los grandes supermercados, induciendo a los usuarios a consumir de los mercados.
- Incremento del consumo de proximidad en la ciudad, procedente de los mercados alimenticios.

3 ESTADO DEL ARTE

Distintos portales web como el blog Som de mercat! [7], la guía de Barcelona [8] y las páginas oficiales de algunos de los mercados, entre otros sitios web de interés, se han ido actualizando, de forma periódica, para ofrecer información sobre los horarios, el modo de reparto a domicilio y vías de contacto con los mercados y/o comercios.

Como se ha comentado anteriormente, es posible encontrar información sobre los mercados en diversos sitios web y también redes sociales, pero se puede comprobar que en numerosas ocasiones la información no está actualizada o no concuerda con las otras fuentes de información, y por tanto, no aporta fiabilidad al usuario. Lo mismo ocurre con Google Maps, la ficha del mercado muestra información que quizás no concuerda con el resto de las fuentes.

También existe una situación similar con el servicio de envío a domicilio, muchos comercios ofrecen este tipo de servicio de forma personal y otros lo ofrecen mediante servicios de terceros como son Manzaning [9] y Ulabox [10], que son servicios de pago. Los diferentes portales web también actualizan la información sobre los envíos a domicilio periódicamente, se puede observar que existen comercios que no realizaban envíos a domicilio, o no utilizaban ningún servicio de terceros y más adelante sí y viceversa por distintas razones.

De igual manera ocurre con los teléfonos de contacto y las redes sociales como WhatsApp e Instagram. Los comerciantes se suman a utilizar su teléfono móvil para recibir pedidos o reservas, pero siempre puede existir un cambio de número de teléfono o el abandono de las redes sociales por incomodidad o falta de experiencia. El hecho de que las fuentes donde se puede hallar este tipo de información no se actualizan de forma tan frecuente provoca que no se pueda garantizar la veracidad de la información que ofrecen.

En conclusión, existe una descentralización y falta de información que afecta a la visibilidad de los mercados y de sus comercios, y con este proyecto se pretende aportar una solución a esta situación.

4 METODOLOGÍA

La siguiente sección presenta las metodologías de desarrollo que han permitido la correcta evolución del proyecto, y permite conocer la filosofía que ha garantizado el éxito del trabajo. Al mismo tiempo, expone la estructura de descomposición del trabajo, una técnica fundamental para el desarrollo de proyectos que facilita la definición del abasto y su línea base, mejora las estimaciones y costes y, por último, aporta una visibilidad global del trabajo.

4.1. Desarrollo iterativo e incremental

El desarrollo iterativo e incremental surge como resultado de la combinación de las metodologías de desarrollo iterativo y desarrollo incremental [4].
Por un lado, las iteraciones se basan en procesos cíclicos donde se desarrolla, se hacen pruebas y, posteriormente, se refina el producto o proceso que se está llevando a cabo [11]. Por otro lado, los incrementos se basan en la construcción del sistema por porciones, las cuales deben ser diseñadas y construidas de forma independiente [12].

Al aplicar esta metodología, se comenzó con una planificación inicial, seguidamente se inició dichas iteraciones con una planificación, y más adelante se llevaba a cabo una elicitación de requisitos, análisis y diseño del *software* e implementación. Llegados a este punto, se efectuó un despliegue de la porción desarrollada del producto para la obtención de realimentación por parte de los usuarios involucrados en el desarrollo del proyecto, y una vez realizado el despliegue, la iteración siguió curso con la parte de las pruebas y la evaluación. Una vez finalizada la evaluación, se dio por terminado el incremento e iteración y el proceso volvía a comenzar.

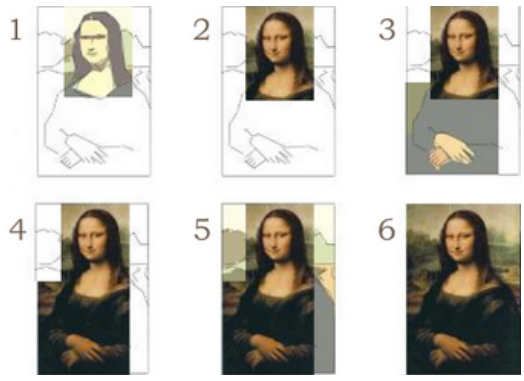

Fig. 1: Proceso de desarrollo iterativo e incremental [13].

4.2. Kanban

En el desarrollo de *software*, el método Kanban [14] se suele usar junto con otros marcos de trabajo y/o metodologías de desarrollo. En esta ocasión, se ha utilizado junto con el desarrollo iterativo e incremental, metodología mencionada en la subsección 4.1, y el diseño centrado en el usuario, que se presentará en la siguiente subsección.

Para aplicar esta metodología se han anotado todas las tareas del proyecto en la herramienta Trello [15], la cual permite gestionar proyectos y tareas a partir un tablero dividido en distintas fases, lo que ha permitido disponer una visión sobre todos los procesos desde su inicio a su finalización [16]. A su vez, el uso de la metodología con Trello facilitó la toma decisiones sobre qué elaborar, en qué momento y qué cantidad.

A continuación, en la Fig. 2, se muestra el tablero Kanban del proyecto Mercapp.

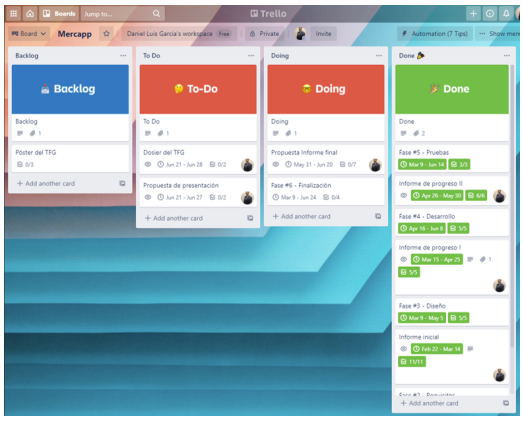

Fig. 2: Tablero Kanban del proyecto.

4.3. Diseño centrado en el usuario

El diseño centrado en el usuario [3] es una filosofía y metodología de desarrollo que tiene como objetivo garantizar el éxito de un producto o sistema a partir de la involucración del usuario en todas y cada una de las fases de desarrollo del proyecto mediante un proceso iterativo.

El proceso de iteración se basó en la realimentación continua por parte de las personas involucradas en el proyecto, con el objetivo de aumentar la satisfacción del usuario o cliente a partir de incrementar la productividad de las personas involucradas. Gracias a ello, se reducen los costes de formación y el tiempo de desarrollo, dado que únicamente se desarrollan las funcionalidades que necesita el usuario [3].

Las metodologías ágiles se encuentran muy centradas en el desarrollo de *software*. La falta de investigación de usuarios y definición de requisitos de las personas interesadas en este tipo de proyectos, hace que la sinergia entre los métodos ágiles y el diseño centrado en el usuario genere la estrategia y conceptualización necesarias para poder definir la implementación de funcionalidades del sistema [3].

4.4. Planificación

La planificación de Mercapp consta de seis fases: Planificación, Requisitos, Diseño, Desarrollo, Pruebas y Finalización. A continuación, en la Tabla 1, se presentan las fases mencionadas y el tiempo dedicado para cada una de ellas.

Fase	Inicio	Fin
Planificación	15/02/2021	05/03/2021
Requisitos	08/03/2021	12/03/2021
Diseño	09/03/2021	05/05/2021
Desarrollo	16/04/2021	08/06/2021
Pruebas	09/03/2021	14/06/2021
Finalización	09/03/2021	25/06/2021

Tabla 1: Resumen de la planificación del proyecto.

Durante la primera fase se ha elaborado la definición del producto a partir de determinar el abasto y los objetivos y mediante la aprobación de las personas interesadas en el proyecto. Al mismo tiempo, se definieron las actividades a realizar durante el transcurso del trabajo, los recursos necesarios, una estimación de costes y el plan de contingencia frente a los posibles riesgos que podía sufrir el desarrollo del proyecto.

Seguidamente, en la segunda fase, se llevó a cabo la elicitación y documentación de requisitos a todas aquellas personas interesadas en el proyecto.

A continuación, la tercera fase se enfocó en el diseño de la interfaz gráfica y de las funcionalidades del sistema. Para lograr el objetivo de esta fase, se elaboraron *sketchboards*, *mockups* y prototipos, además del diseño de los casos de uso, del diagrama de clases y de las bases de datos. Todo ello en base a las necesidades de los usuarios involucrados.

Acto seguido, en la cuarta fase, se procedió al desarrollo del sistema, lo que implicó establecer las conexiones con las bases de datos, gestionar la localización del dispositivo móvil y mostrar las diferentes vistas de la aplicación.

Llegados al punto donde el desarrollo permitía mostrar los resultados y avances comienzan las pruebas del *software*. Dichas pruebas pretendían poner a prueba el dispositivo frente a situaciones reales y cotidianas de las personas involucradas en el proyecto, de esta manera se consiguió detectar cualquier tipo de error, visual o funcional, que pudiera existir a partir de un mal desarrollo de *software* y, por consiguiente, también permitió su temprana solución antes de la finalización del proyecto.

Para poder observar el diagrama de la estructura de descomposición de trabajo del proyecto, ver anexo A.1.

5 ELICITACIÓN DE REQUISITOS

La siguiente sección expone la descripción del sistema que se ha desarrollado a partir de la documentación de requisitos, una técnica esencial que engloba cualquier tipo de descripción, más o menos formal, que facilita la comunicación entre las partes interesadas del proyecto e incrementa la calidad de los requisitos que han sido documentados.

5.1. Requisitos funcionales

A continuación, en la Tabla 2, se presentan los requisitos funcionales del sistema en lenguaje natural, aquellos que definen la funcionalidad que tiene que ofrecer el sistema a desarrollar.

ID	Descripción
RF-001	El sistema Mercapp tiene que mostrar al usuario una lista con todos los mercados de la ciudad condal.
RF-002	El sistema Mercapp tiene que mostrar al usuario una lista con todos los comercios de la ciudad condal.
RF-003	El sistema Mercapp tiene que mostrar al usuario un mapa de la ciudad condal.
RF-004	El sistema Mercapp tiene que mostrar al usuario la ubicación de todos los mercados de la ciudad condal mediante el uso de marcadores en el mapa.
RF-005	El sistema Mercapp podrá proveer al usuario la capacidad de cambiar la vista del mapa.
RF-006	El sistema Mercapp tiene que proveer al usuario la capacidad de seleccionar un mercado de la lista de mercados.
RF-007	El sistema Mercapp tiene que proveer al usuario la capacidad de seleccionar un comercio de la lista de comercios.
RF-008	El sistema Mercapp tiene que mostrar al usuario un conjunto de imágenes de un mercado.
RF-009	El sistema Mercapp tiene que mostrar al usuario un conjunto de imágenes de un comercio.
RF-010	El sistema Mercapp tiene que mostrar al usuario la descripción de un mercado.
RF-011	El sistema Mercapp tiene que mostrar al usuario la descripción de un comercio.
RF-012	El sistema Mercapp tiene que mostrar al usuario los horarios de apertura de un mercado.
RF-013	El sistema Mercapp tiene que mostrar al usuario los horarios de apertura de un comercio.
RF-014	El sistema Mercapp tiene que mostrar al usuario las fechas de apertura de un mercado.
RF-015	El sistema Mercapp tiene que mostrar al usuario las fechas de apertura de un comercio.
RF-016	El sistema Mercapp tiene que mostrar al usuario si un mercado dispone de aparcamiento.
RF-017	El sistema Mercapp tiene que mostrar al usuario si un mercado es accesible para personas con movilidad reducida.
RF-018	El sistema Mercapp tiene que mostrar al usuario si es posible ingerir alimentos en un mercado.
RF-019	El sistema Mercapp tiene que mostrar al usuario si es posible el reparto a domicilio por parte de un comercio.
RF-020	El sistema Mercapp podrá proveer al usuario la capacidad de contactar mediante llamada telefónica con un mercado.
RF-021	El sistema Mercapp podrá proveer al usuario la capacidad de contactar mediante llamada telefónica con un comercio.
RF-022	El sistema Mercapp podrá proveer al usuario la capacidad de contactar mediante WhatsApp con un mercado.
RF-023	El sistema Mercapp podrá proveer al usuario la capacidad de contactar mediante WhatsApp con un comercio.
RF-024	El sistema Mercapp podrá proveer al usuario la capacidad de visitar el perfil de Instagram de un mercado.
RF-025	El sistema Mercapp podrá proveer al usuario la capacidad de visitar el perfil de Instagram de un comercio.
RF-026	El sistema Mercapp podrá proveer al usuario la capacidad de conocer la ubicación un mercado mediante Google Maps.
RF-027	El sistema Mercapp podrá proveer al usuario la capacidad de conocer la ubicación un comercio mediante Google Maps.
RF-028	El sistema Mercapp podrá proveer al usuario la capacidad de añadir un mercado a favoritos.
RF-029	El sistema Mercapp podrá proveer al usuario la capacidad de añadir un comercio a favoritos.
RF-030	El sistema Mercapp podrá mostrar al usuario una lista con todos los mercados favoritos.
RF-031	El sistema Mercapp podrá mostrar al usuario una lista con todos los comercios favoritos.

Tabla 2: Requisitos funcionales del sistema.

5.2. Requisitos de calidad

A continuación, en la Tabla 3, se exponen en lenguaje natural los requisitos de calidad o requisitos no funcionales, aquellos que definen las cualidades deseadas del sistema a desarrollar, y a menudo influyen en la arquitectura del sistema más que los requisitos funcionales.

ID	Descripción
RC-001	El sistema debe estar disponible en el idioma Castellano.
RC-002	El sistema debe estar disponible en el idioma Catalán.
RC-003	El sistema debe estar disponible en el idioma Inglés.
RC-004	El sistema debe de estar disponible para el sistema operativo Android.
RC-005	El sistema debe de estar disponible para elsistema operativo iOS.
RC-006	Toda interacción del usuario con la interfaz gráfica del sistema debe dar respuesta en menos de 5 segundos.
RC-007	Las actualizaciones de datos en la base de datos deben estar disponibles para todos los usuarios en la próxima inicialización del sistema.
RC-008	El sistema debe estar desarrollado de forma amigable e intuitiva para el usuario, a fin de garantizar la adecuada y cómoda visualización por parte del usuario.
RC-009	El sistema debe tener una disponibilidad superior al 95 %.
RC-010	El procedimiento de desarrollo del sistema debe cumplir con la legislación vigente de la Unión Europea.

Tabla 3: Requisitos de calidad del sistema.

6 DISEÑO DEL SISTEMA

La siguiente sección presenta el diagrama de casos de uso del sistema, el cual permite observar y comprender el comportamiento del sistema en función de las posibles acciones del usuario. Al mismo tiempo, se muestran algunos de los diseños de baja y alta fidelidad del sistema, concretamente *sketchboards* y *mockups*.

6.1. Diagrama de casos de uso

A continuación, en la Fig. 3, se muestra el diagrama de casos de uso, en el cual se presentan todas las funciones que el sistema Mercapp permite realizar al usuario, además de las relaciones entre funcionalidades.

Fig. 3: Diagrama de casos de uso del sistema.

6.2. *Sketchboards* y *mockups* del sistema

En esta subsección se recogen algunos de los diseños de baja y alta fidelidad que han sido creados como prototipo de aplicación para dispositivos móviles del sistema Mercapp. Estos diseños han servido para mostrar conceptos y probar opciones de diseño, al mismo tiempo se han utilizado para que las personas involucradas en el proyecto pudieran realizar pruebas de usabilidad como *first click testing* y *user testing*.

A continuación, en las Fig. 4 y 5, se muestra el diseño de la lista de mercados en un prototipo de baja fidelidad y en uno de alta, respectivamente.

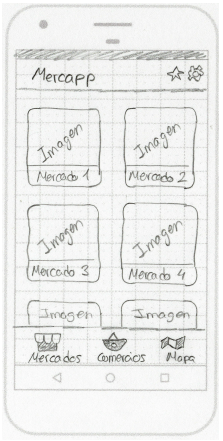

Fig. 4: *Sketchboard* de la lista de mercados en forma de cuadrícula.

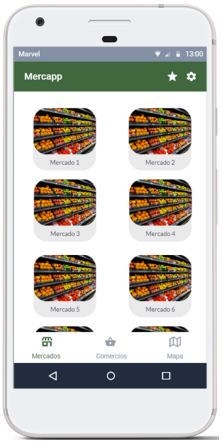

Fig. 5: *Mockup* de la lista de mercados en forma de cuadrícula.

Seguidamente, en las Fig. 6 y 7, se presenta el diseño de la vista detallada de un mercado en un prototipo de baja fidelidad y en uno de alta, también de manera respectiva. Y acto seguido, en las Fig. 8 y 9, se expone el diseño del mapa con marcadores en un *sketchboard* y en un *mockup*, respectivamente.

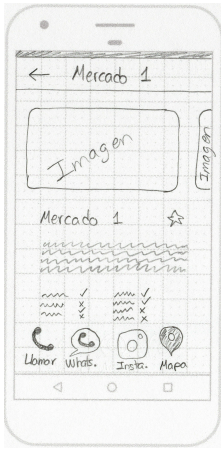

Fig. 6: *Sketchboard* de la vista detallada del mercado.

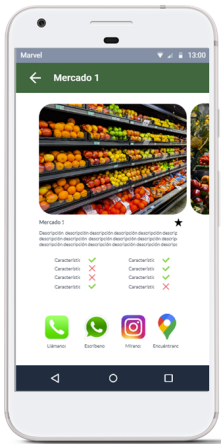

Fig. 7: *Mockup* de la vista detallada del mercado.

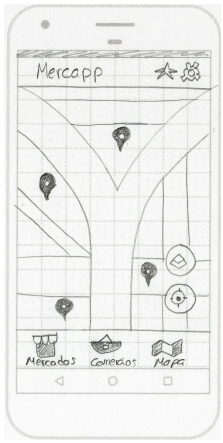

Fig. 8: *Sketchboard* de la vista del mapa con marcadores.

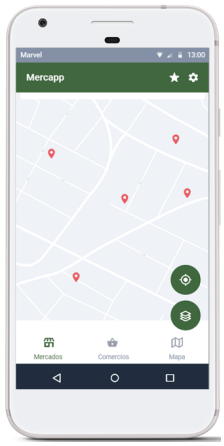

Fig. 9: *Mockup* de la vista del mapa con marcadores.

Para poder observar más diseños *sketchboard* y *mockup* del sistema, ver anexo A.2.

7 Desarrollo del sistema

Esta sección expone las funcionalidades desarrolladas del sistema con mayor relevancia para las personas involucradas en el proyecto en función de sus deseos y necesidades, que a su vez, todas y cada una de ellas han sido desarrolladas con el lenguaje de programación Dart y el kit de desarrollo de *software* Flutter, ambos creados por Google.

7.1. Modelos de datos

Antes de seleccionar bases de datos y comenzar con su implementación, ha sido conveniente tener claro el tipo de información que se deseaba mostrar a los usuarios del sistema en cada una de las vistas de la aplicación. Una vez se consideraron todos los requisitos documentados en la sección 5, se decidió generar dos modelos de datos independientes, dado que los mercados y comercios no tendrán ni utilizarán los mismos atributos para mostrar la información en la interfaz gráfica.

Dentro de cada modelo se hallan todos los atributos, en formato `JSON`, que se han utilizado para mostrar la infor-

mación deseada sobre cada cada uno de los mercados y comercios de la ciudad condal.

Para poder observar los distintos modelos de datos y poder ver sus diferencias, ver anexo A.3.

7.2. Bases de datos

Después de analizar algunas de las opciones disponibles en el mercado, las bases de datos que han sido utilizadas para el proyecto son Realtime Database de Firebase [17], Cloudinary [18] y SQLite [19].

Por un lado, tanto Realtime Database como Cloudinary son bases de datos que se alojan en la nube, no necesitan ninguna operación ni ningún mantenimiento de servidores. Ambas han proporcionado al sistema Mercapp todo lo necesario para ser completamente funcional utilizando las versiones gratuitas de ambos servicios de almacenamiento de datos.

Por otro lado, SQLite es el motor de base de datos SQL más popular, y se encuentra disponible en casi cualquier dispositivo. Al igual que las dos primeras, SQLite también ha ofrecido al proyecto todo lo necesario para ser funcional sin ninguna inversión de capital.

A continuación, se detalla el motivo por el cual se han escogido las bases de datos mencionadas anteriormente:

- **Firebase - Realtime Database:** Se trata de un *Backend-as-a-Service* que permite almacenar y sincronizar datos con una base de datos no relacional [17] alojada en la nube. Toda la información, almacenada en formato `JSON`, se sincroniza con todos los usuarios del sistema en tiempo real en lugar de lanzar las típicas solicitudes `HTTP`. Cabe decir, que la API de Realtime Database está diseñada para permitir, únicamente, operaciones que puedan ejecutarse rápidamente. Al mismo tiempo, proporciona un lenguaje flexible de reglas basadas en expresiones, llamado reglas de seguridad de Firebase Realtime Database, para definir cómo se deberían estructurar los datos y en qué momento se pueden leer o escribir [17].

- **Cloudinary:** Se trata de un *Software-as-a-Service* que permite cargar, gestionar, manipular y proveer imágenes y/o vídeos para sitios web o aplicaciones de manera eficiente y efectiva. Conviene resaltar, que esta plataforma ha sido utilizada únicamente para almacenar todos aquellos recursos que no han sido obtenidos mediante internet, es decir, los recursos facilitados por los mercados, comerciantes, o bien, elaborados u obtenidos por el propio desarrollador del sistema.

- **SQLite:** Para usar el motor de SQLite, los desarrolladores de Flutter pueden escoger de entre un abanico de diferentes opciones para usarlo en sus proyectos. Uno de los complementos para Flutter más populares que realiza las operaciones de crear, leer, actualizar y eliminar, también llamadas operaciones CURD, es el complemento `sqflite` [20]. Este es el elegido para poder utilizar SQLite junto con el kit de desarrollo de *software* Flutter.

7.3. Mostrar mercados y comercios

La funcionalidad clave de este sistema es la capacidad de mostrar todos los mercados de la ciudad condal y sus comercios. El desarrollo de esta funcionalidad ha sido posible a partir de la conexión con la base de datos Realtime Database mediante el complemento para Flutter `http` [21], el cual ha permitido llevar a cabo las peticiones `HTTP` a las direcciones `URL` que contienen la información correspondiente a los mercados y comercios.

Una vez se estableció la conexión y se obtuvo la información procedente de la base de datos, se generó una lista en forma de cuadrícula que mostraba la imagen principal y el nombre de cada uno de los mercados, y otra lista para los comercios, siguiendo el mismo formato.

Además, si el usuario desea obtener información sobre un mercado o comercio en concreto, la aplicación permite mostrar en pantalla toda la información disponible en el modelo de datos referente al mercado o comercio que haya sido seleccionado de su lista. De esta forma, los usuarios pueden observar información detallada y de interés sobre cada uno.

A continuación, en las Fig. 10 y 11, se muestra una lista con los mercados de la ciudad y la información detallada del mercado de Sant Antoni, respectivamente.

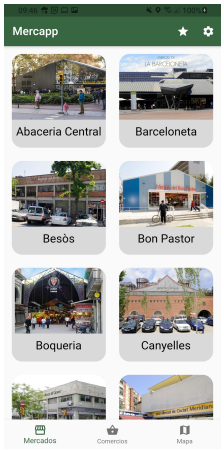

Fig. 10: Lista de mercados.

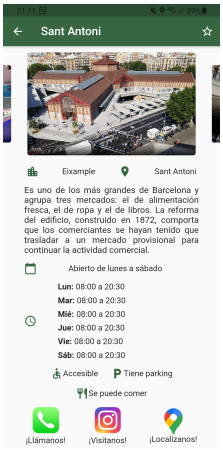

Fig. 11: Información detallada del mercado de Sant Antoni.

Para observar el fragmento de código que permite establecer la conexión con la base de datos y obtener la información de todos los mercados, ver anexo A.4.

7.4. Establecer y mostrar favoritos

Una de las funcionalidades más interesantes que ofrece el sistema es que el usuario pueda marcar como favorito un mercado o comercio y que este conste en la sección de favoritos de su aplicación. En el instante en que el usuario marca como favorito un mercado o comercio, este aparecerá, en tiempo real, en la sección de mercados o comercios favoritos de la aplicación.

Ha sido posible desarrollar esta funcionalidad gracias al complemento para Flutter llamado `provider` [22], el cual ofrece la posibilidad de colocar cualquier estado de un objeto en el árbol de `widgets` y, al mismo tiempo, hacerlo visible desde otros `widgets`. De esta forma, cuando un

mercado o comercio se establece como favorito su estado cambia y se actualiza el árbol de `widgets`, permitiendo así observar este cambio de forma instantánea en la sección de favoritos correspondiente.

A continuación, en las Fig. 12 y 13, se muestra la vista detallada de un comercio marcado como favorito y la lista de comercios favoritos que incluye el comercio mencionado, respectivamente.

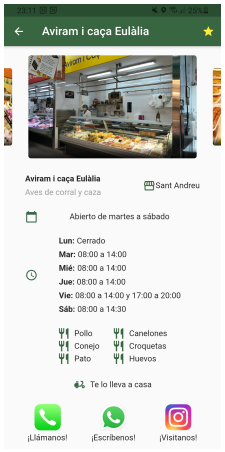

Fig. 12: Comercio marcado como favorito.

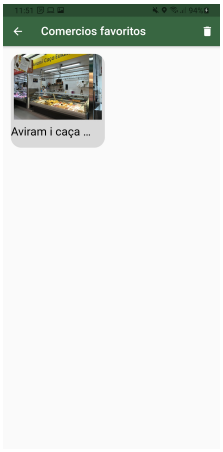

Fig. 13: Lista de comercios favoritos.

Para observar más imágenes relacionadas con la vista de elementos favoritos en la aplicación, o ver el fragmento de código correspondiente al método que permite mostrar una lista con los comercios que hayan sido marcados como favoritos, ver anexo A.5.

7.5. Interacción con los mercados y comercios mediante otras aplicaciones

Con motivo de ofrecer aún más visibilidad a los mercados y comercios con Mercapp, se ha desarrollado una funcionalidad que permite a los usuarios del sistema interactuar con los mercados de la ciudad y sus comercios mediante el uso de distintas aplicaciones desde el mismo sistema. Para ello se ha recurrido al complemento de Flutter `url_launcher` [23], el cual permite ejecutar cualquier aplicación desde Mercapp a partir de una dirección `URL` facilitada previamente por el modelo de datos.

A partir del desarrollo de la funcionalidad con este complemento en el proyecto se ha podido ofrecer el uso de las siguientes aplicaciones para obtener más información o interactuar con los mercados y comercios:

- **Mercados:** Teléfono, Instagram y Google Maps.
- **Comercios:** Teléfono, WhatsApp e Instagram.

Para observar el fragmento de código correspondiente al método que permite usar la aplicación Teléfono para realizar llamadas telefónicas a un mercado o comercio determinado, cuyo número de contacto se encuentra almacenado en la base de datos del sistema y se usará para generar la `URL` mencionada anteriormente, ver anexo A.6.

7.6. Mostrar mapa con marcadores y ubicación del dispositivo

Otra de las funcionalidades que ofrece Mercapp es mostrar todas las ubicaciones exactas de cada uno de los mercados de la ciudad de Barcelona. Para desarrollar esta funcionalidad ha sido necesario utilizar el complemento `google_maps_flutter` [24], el cual provee el `widget` de Google Maps para Flutter, y también dos API Key, las cuales permitieron el uso de los servicios del mapa de Google tanto para sistemas operativos Android como para iOS.

Además, si un marcador del mapa es seleccionado por un usuario, este expondrá en pantalla el nombre y una imagen del mercado en cuestión y permitirá redirigir al usuario a la vista detallada del mercado.

Paralelamente, el sistema también permite mostrar la ubicación en tiempo real del dispositivo móvil, siempre y cuando el usuario conceda los permisos requeridos por la aplicación. Esta funcionalidad ha sido posible desarrollarla mediante el complemento `location` [25], el cual se encarga de obtener la ubicación del dispositivo en tiempo real cada vez que esta se ve alterada.

A continuación, en las Fig. 14 y 15, se presenta la vista del mapa con los marcadores de ubicación de los mercados de la ciudad y se muestra el marcador de posición del dispositivo móvil, respectivamente.

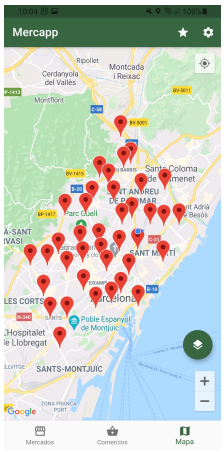

Fig. 14: Vista del mapa con marcadores.

Fig. 15: Marcador de ubicación del dispositivo móvil.

Para observar más imágenes relacionadas con la vista del mapa de la aplicación, o ver el fragmento de código correspondiente al método que permite mostrar el mapa de Google Maps junto con la localización del dispositivo y los marcadores de cada mercado de la ciudad, ver anexo A.7.

8 Pruebas del sistema

8.1. *First click testing*

First click testing es una estrategia para validar y medir la usabilidad de una aplicación, un sitio web o un diseño averiguando el nivel de complejidad de completar una tarea previamente dada. El objetivo de este estudio residía en verificar que el primer clic que realiza un usuario sobre una interfaz es intuitivo, y para ello se ha utilizado la plataforma Optimal Workshop [26]. Además, el estudio *first*

click permite ordenar tareas a los participantes e investigar su comportamiento frente a diferentes situaciones de forma totalmente independientes [26].

Los clics en una interfaz pueden ocurrir en lugares inesperados, y pueden resaltar zonas confusas de una interfaz. Es por eso por lo que el hecho de visualizar dónde hacen clic los usuarios brinda una gran comprensión del diseño de un sistema. Al mismo tiempo, *first click* permite mostrar qué expectativas tiene el usuario frente a elementos comunes de una interfaz. Concretamente, la herramienta ofrece la medición del tiempo que se toman los usuarios para hacer clic, esto puede ser muy relevante para determinar si una acción concreta es fácil de realizar o tiene algún tipo de complejidad, y así comparar la usabilidad de las alternativas de diseño del sistema [26].

A continuación, en las Fig. 16 y 17, se presenta uno de los resultados surgidos de llevar a cabo esta prueba. Concretamente, se trata de una tarea que tenía como finalidad marcar el mercado que se muestra en pantalla como favorito, de este modo sería posible conocer si este proceso es suficientemente intuitivo para el usuario o no.

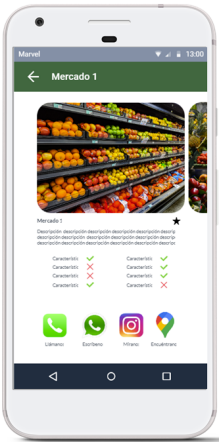

Fig. 16: *Mockup* presentado a los participantes para llevar a cabo la tarea.

Fig. 17: *Mockup* anterior con un mapa de calor en función de los clics de los participantes al realizar la tarea.

Como se observar en el mapa de calor de la Fig. 17, el 90 % de los participantes consiguieron completar la tarea correctamente con un tiempo medio de 8.3s, un tiempo un tanto elevado para hallar la solución de esta tarea.

Una vez concluido el estudio, muchos de los participantes alegaron su inconformidad con la ubicación y el color del icono de favoritos. A causa de los resultados del estudio y de la realimentación obtenida por parte de los participantes se elaboró un nuevo diseño de la vista, con el fin de ofrecer a los usuarios una interfaz mucho más intuitiva. A continuación, en la Fig. 18, se puede observar la nueva versión de la vista.

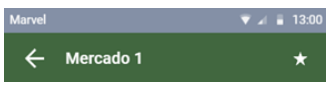

Fig. 18: Resultado de la modificación del *mockup* en cuanto a la ubicación y el color del icono de favoritos.

8.2. *User testing*

El *user testing* o pruebas de usuario es un proceso en el cual se ha involucrado al usuario para llevar a cabo tareas específicas sobre la aplicación Mercapp. Este tipo de prueba es esencial para evaluar la usabilidad del producto tanto a nivel funcional como de la interfaz de usuario.

Para llevar a cabo las pruebas de usuario ha sido necesario que las personas interesadas en el sistema elaborasen, de forma totalmente natural, una serie de tareas sencillas con el producto. Así pues, durante el desarrollo de las diferentes pruebas se ha podido observar y obtener realimentación en directo sobre si la aplicación era lo suficientemente intuitiva y *user-friendly* para los usuarios.

De forma paralela, se ha conseguido conocer y comprender qué requisitos no se estaban cumpliendo y qué errores tenía el *software* [27].

A continuación, se exponen dos resultados extraídos de este tipo de prueba:

- **Información del marcador:** Inicialmente, si el usuario se encontraba en la vista del mapa y trataba de seleccionar algún marcador, el sistema Mercapp no daba respuesta a esa acción. Los participantes solicitaron que la aplicación mostrase información sobre qué mercado se encuentra en cada marcador cuando este se selecciona. Además, algunos de los usuarios involucrados sugirieron que se concediese la posibilidad de redirigir al usuario a la vista del mercado en cuestión desde el mismo marcador.

  Para observar el resultado del desarrollo de esta funcionalidad, ver anexo A.7.

- **Icono de favoritos:** Inicialmente, el icono de favoritos era estático y no proporcionaba realimentación al usuario en el momento que el mercado o comercio se establecía como favorito o dejaba de serlo. Los participantes solicitaron esa realimentación por parte del sistema, para dar a conocer cuando un mercado o comercio está marcado como favorito o no. Para poder satisfacer esta necesidad, se desarrollaron unas líneas de código que permiten el cambio de color, desde un simple bordeado blanco a un relleno amarillo y viceversa, en función de si el elemento en cuestión se ha establecido como favorito o no.

  Para observar el resultado del desarrollo de esta funcionalidad, ver el anexo A.5.

8.3. Pruebas exploratorias

Las pruebas exploratorias o *exploratory testing* es un enfoque, una actitud o una mentalidad que permite, simultáneamente, aprender sobre el sistema, diseñar casos de prueba y ejecutarlos. Con la ayuda de este tipo de pruebas, ha sido posible obtener una mayor profundidad y realimentación sobre las distintas funcionalidades que ofrece el sistema, al mismo tiempo que ha facilitado el aprendizaje de los usuarios involucrados en el proyecto.

Dado que no se dispone ningún teléfono móvil con sistema operativo iOS ni ningún ordenador macOS, no se ha podido llevar a cabo ningún tipo de pruebas del sistema con

terminales Apple ni con emuladores de iOS. Es por esta razón que únicamente se han utilizado un dispositivo móvil y un emulador con sistema operativo Android.

A continuación, en la Tabla 4, se muestran las especificaciones sobre los dispositivos que han sido utilizado para las pruebas exploratorias.

	Dispositivo #1	Dispositivo #2
Modelo	Galaxy S9+	Pixel 4XL
Tipo	Físico	Emulador
SO	Android	Android
Versión SO	10	11

Tabla 4: Dispositivos utilizados para las pruebas.

9 Resultados

A día de hoy, el proyecto se ha convertido en un prototipo para dispositivos móviles, el cual comienza a cumplir con algunos de los objetivos y requisitos especificados en las secciones 2 y 5.
El sistema desarrollado con Dart y Flutter permite la visualización de todos los mercados alimenticios de la ciudad condal de forma totalmente centralizada y con toda aquella información relevante para los usuarios involucrados en el sistema, como es la disponibilidad de imágenes, horarios de apertura, localización y servicios disponibles, entre otros elementos también de interés, además de permitir a los usuarios interactuar con los distintos mercados a través de terceras aplicaciones como son Teléfono, WhatsApp, Instagram o Google Maps.

Al mismo tiempo, la aplicación permite la visualización de todos los comercios presentes en cada uno de los mercados de Barcelona. Sin embargo, por la falta de información de cada uno de los establecimientos y la necesidad de obtención de permisos por parte del Institut de Mercats Municipals de Barcelona, únicamente se han podido incluir unos pocos comercios del mercado de Sant Andreu con toda la información relevante y de interés para los usuarios involucrados, a saber, imágenes del establecimiento, horarios de apertura, localización, recomendación de productos, entre otros elementos, además de permitir a los usuarios interactuar con los distintos comercios a través de terceras aplicaciones como son Teléfono, WhatsApp o Instagram.

Paralelamente, el sistema ofrece la posibilidad de mostrar en un mapa integrado dentro de la aplicación la ubicación exacta de todos los mercados de Barcelona señalizados mediante marcadores de posición. Además de presentar la ubicación, estos marcadores permiten mostrar información sobre cada mercado y ofrecen la posibilidad de redirigir al usuario a la vista principal del mercado deseado.

La aplicación también incluye una sección de favoritos, la cual permite al usuario visualizar todos aquellos mercados y/o comercios que se hayan establecido como tal en sus respectivas vistas mediante el botón de favorito, representado con una estrella de color blanco o amarillo, dependiendo si el mercado o establecimiento ha sido marcado como favorito previamente o no.

Conviene resaltar que todas las herramientas utilizadas en el desarrollo de este proyecto han sido totalmente gratuitas.

10 Conclusiones

El objetivo principal del proyecto era tratar de ofrecer un prototipo de aplicación que aporte visibilidad a los mercados de la ciudad de Barcelona a partir de la centralización de información en un mismo sistema.

Hasta el día de hoy se ha conseguido desarrollar una primera versión de Mercapp, la cual ofrece información sobre todos los mercados de la ciudad, información sobre algunos comercios del mercado de Sant Andreu, la funcionalidad de almacenar mercados y/o comercios como favoritos y, por último, la vista de un mapa que informa de la ubicación mediante marcadores de todos aquellos mercados que se encuentren en la ciudad.

En este punto, puedo afirmar que ha sido todo un reto enfrentarme a un proyecto como Mercapp. Estos últimos meses han sido intensos en cuanto a investigación, aprendizaje y desarrollo, sin embargo, como todo esfuerzo tiene su recompensa, también he disfrutado mucho viendo crecer el proyecto a medida que pasaban las semanas. Pero sobre todas las cosas, con lo que más he disfrutado es con la realimentación positiva y constante de las personas que me rodean.

Durante el transcurso del proyecto he podido aplicar muchos conocimientos y habilidades vistos durante el grado, los cuales he tenido la oportunidad de ampliar día tras día. Las habilidades y conceptos aplicados tienen su origen en la gestión de proyectos, requisitos, diseño e ingeniería del *software* y bases de datos, por ejemplo. Sin embargo, me gustaría destacar la actitud con la que he encarado el proyecto, la cual poco a poco me ha permitido crecer a nivel personal y profesional.

10.1. Futuro del proyecto

Semana tras semana, Mercapp ha ido alzando el vuelo hasta transformarse en un proyecto con mucho potencial, después de ver los resultados que han generado su desarrollo. Aún siendo necesario confirmar la veracidad de la información que ya está presente en el sistema y recopilar información sobre todos los comercios de la ciudad, no se trata de un proyecto que haya llegado a su fin.

Durante la recolecta de información por los comercios del mercado de Sant Andreu junto al principal usuario involucrado en el proyecto y tras realizar diversas demostraciones de la aplicación, recibí muy buenas opiniones y muchos ánimos por parte de algunos comerciantes para continuar con el desarrollo una vez presentado el Trabajo de Fin de Grado, para que algún día el proyecto pueda salir a la luz. Al mismo tiempo, tuve la oportunidad de entablar una conversación con la directora del mercado, Montse, cuya reacción al ver una demostración de la aplicación fue tal que se puso en contacto, pocos minutos después, con personal del Institut Municipal de Mercats de Barcelona para que supieran de mí y, notificarles que tal vez algún día, me pondría en contacto con ellos para poder presentarles el proyecto de forma oficial.

Si el Institut Municipal de Mercats de Barcelona resulta estar interesado en que este proyecto se convierta en una aplicación real y que se encuentre disponible para la población, deberíamos reunirnos para elaborar una planificación, obtener requisitos, elaborar estudios de viabilidad y

un modelo de negocio. En caso de ser un proyecto viable y financiado por el organismo, se firmaría un contrato y se procedería a llevar a cabo el desarrollo de la aplicación.

AGRADECIMIENTOS

En primer lugar, me gustaría agradecer a Helena todo el apoyo y todos los consejos que me ha brindado durante estos meses, toda la paciencia que ha tenido con mis interminables informes y las buenas vibras que me ha transmitido en cada reunión y en cada *e-mail*.

En segundo lugar, quiero dar las gracias a Pep, por estar al pie del cañón siempre que lo he necesitado y por ser una pieza clave en el desarrollo de este proyecto.

En tercer lugar, me gustaría agradecer a mi familia todo el soporte que me han dado en cada uno de mis pasos desde el primer día y por estar a mi lado siempre, en los buenos momentos y en los no tan buenos.

En último lugar, quiero dar las gracias a esa persona que se volvió a cruzar en mi camino hace un tiempo y que desde entonces camina a mi vera, dándome apoyo y amor incondicional. Sin ella esto no habría sido posible.

REFERENCIAS

[1] "Flutter - Beautiful native apps in record time." https://flutter.dev/ (accedido mar. 07, 2021).

[2] "Dart programming language | Dart." https://dart.dev/ (accedido mar. 07, 2021).

[3] M. Garreta and E. Mor, "Diseño centrado en el usuario."

[4] "Iterative and incremental development - Wikipedia." accedido: mar. 07, 2021. [En línea]. Disponible: https://en.wikipedia.org/wiki/Iterative_and_incremental_development.

[5] "Institut Municipal de Mercats de Barcelona | Mercats de Barcelona | Ajuntament de Barcelona." https://ajuntament.barcelona.cat/mercats/ca/canal/institut-municipal-de-mercats-de-barcelona (accedido may 24, 2021).

[6] "Els mercats reforcen els seus serveis a domicili | Som de Mercat," mar. 25, 2020. https://ajuntament.barcelona.cat/somdemercat/ca/continguts/els-mercats-reforcen-els-seus-serveis-domicili (accedido mar. 07, 2021).

[7] "Som de Mercat." https://ajuntament.barcelona.cat/somdemercat/ca (accedido mar. 07, 2021).

[8] "Guía BCN: agenda de actividades, directorios y cursos de Barcelona." https://guia.barcelona.cat/es (accedido mar. 07, 2021).

[9] "MANZANING: Las tiendas de siempre, como nunca." https://manzaning.com/ (accedido mar. 07, 2021).

[10] "ulabox.com | The Online Supermarket." https://www.ulabox.com/en (accedido mar. 07, 2021).

[11] "Iterative design - Wikipedia." accedido: mar. 07, 2021. [En línea]. Disponible: https://en.wikipedia.org/wiki/Iterative_design.

[12] "Incremental build model - Wikipedia." accedido: mar. 07, 2021. [En línea]. Disponible: https://en.wikipedia.org/wiki/Incremental_build_model.

[13] J. Patton, "Don't Know What I Want, But I Know How to Get It - Jeff Patton & Associates." https://www.jpattonassociates.com/dont_know_what_i_want/ (accedido mar. 07, 2021).

[14] "Kanban (development) - Wikipedia." Accedido: mar. 07, 2021. [En línea]. Disponible: https://en.wikipedia.org/wiki/Kanban_(development).

[15] "Trello." https://trello.com/es (accedido mar. 09, 2021).

[16] L. Lanz, "Kanban vs Scrum | OpenWebinars," dec. 23, 2019. https://openwebinars.net/blog/kanban-vs-scrum/ (accedido mar. 07, 2021).

[17] "Firebase Realtime Database | Firebase Realtime Database." https://firebase.google.com/docs/database?hl=es-419 (accedido may 16, 2021).

[18] "Image and Video Upload, Storage, Optimization and CDN." https://cloudinary.com/ (accedido may 10, 2021).

[19] "SQLite Home Page." https://www.sqlite.org/index.html (accedido may 16, 2021).

[20] "sqflite | Flutter Package." https://pub.dev/packages/sqflite (accedido may 16, 2021).

[21] "http | Dart Package." https://pub.dev/packages/http (accedido may. 6, 2021).

[22] "provider | Flutter Package." https://pub.dev/packages/provider (accedido may 21, 2021).

[23] "url_launcher | Flutter Package." https://pub.dev/packages/url_launcher (accedido may. 10, 2021).

[24] "google_maps_flutter | Flutter Package." https://pub.dev/packages/google_maps_flutter (accedido Mar. 09, 2021).

[25] "location | Flutter Package." https://pub.dev/packages/location (accedido may 23, 2021).

[26] "Optimal Workshop | User Experience (UX) Research Platform." https://www.optimalworkshop.com/ (accedido mar. 09, 2021).

[27] "What is User testing? Definition - Omniconvert," 2019. https://www.omniconvert.com/what-is/user-testing/ (accedido may. 26, 2021).

ANEXO

A.1. Estructura de descomposición de trabajo del proyecto

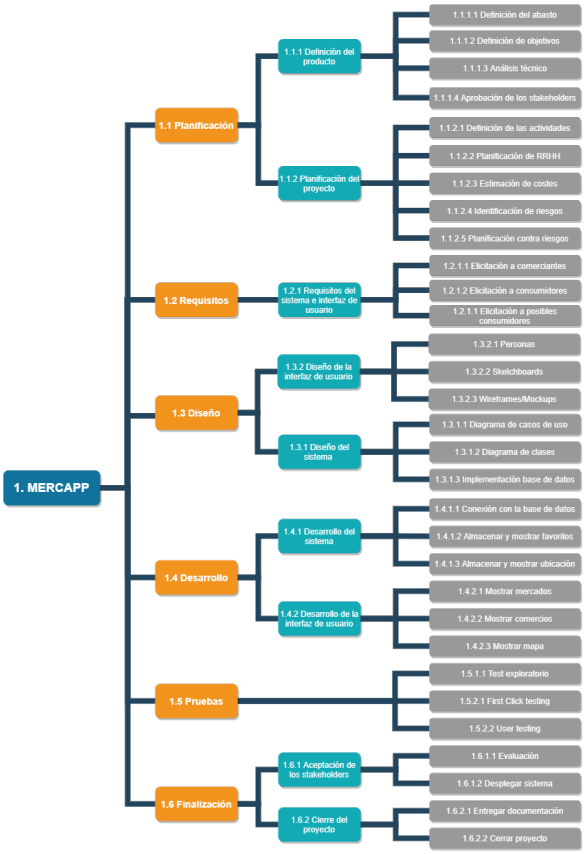

Fig. 19: Diagrama de la estructura de descomposición del trabajo del proyecto.

A.2. *Sketchboards* y *mockups* del sistema

Fig. 20: *Sketchboard* del menú lateral de la aplicación.

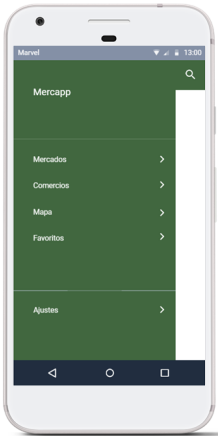

Fig. 21: *Mockup* del menú lateral de la aplicación.

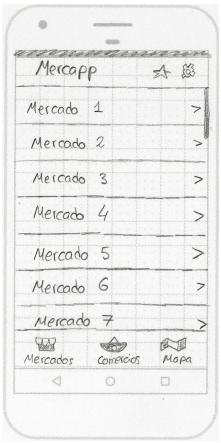

Fig. 22: *Sketchboard* de la vista de mercados.

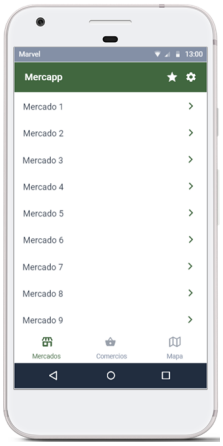

Fig. 23: *Mockup* de la vista de mercados.

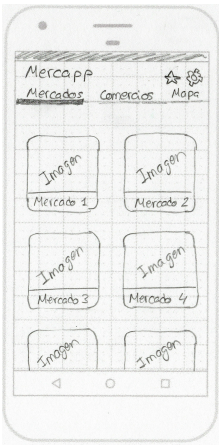

Fig. 24: *Sketchboard* de la vista de mercados.

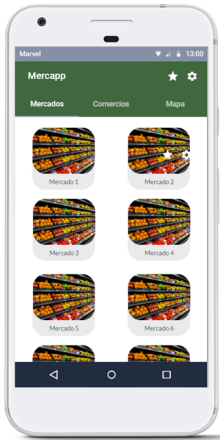

Fig. 25: *Mockup* de la vista de mercados.

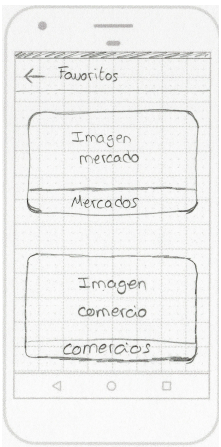

Fig. 26: *Sketchboard* de la sección de favoritos.

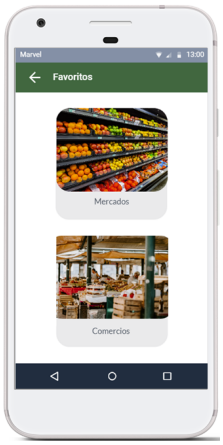

Fig. 27: *Mockup* de la sección de favoritos.

A.3. Modelos de datos

A.3.1. Modelo de mercado

```
class MarketModel {
  MarketModel({
    this.address,
    this.canEatThere,
    this.description,
    this.district,
```

```
      this.fridayOpening,
      this.id,
      this.image1,
      this.image2,
      this.image3,
      this.image4,
      this.image5,
      this.instagram,
      this.isFunctionalDiversityAdapted,
      this.latLng,
      this.mondayOpening,
      this.name,
      this.neighborhood,
      this.openingDates,
      this.hasParking,
      this.phoneNumber,
      this.saturdayOpening,
      this.thursdayOpening,
      this.tuesdayOpening,
      this.wednsdayOpening,
    });
```

Lista 1: Clase `MarketModel` del fichero `market_model.dart`.

A.3.2. Modelo de comercio

```
  class CommerceModel {
    CommerceModel({
      this.canHomeDelivery,
      this.category,
      this.description,
      this.fridayOpening,
      this.id,
      this.image1,
      this.image2,
      this.image3,
      this.image4,
      this.image5,
      this.instagram,
      this.market,
      this.mondayOpening,
      this.name,
      this.openingDates,
      this.phoneNumber,
      this.product1,
      this.product2,
      this.product3,
      this.product4,
      this.product5,
      this.product6,
      this.saturdayOpening,
      this.thursdayOpening,
      this.tuesdayOpening,
      this.wednsdayOpening,
    });
```

Lista 2: Clase `CommerceModel` del fichero `commerce_model.dart`.

A.4. Mostrar mercados y comercios

```
  Future<List<MarketModel>> getFirebaseMarkets()
      async {
  final _marketCollectionURL = '$_firebaseURL/
      mercados.json';
  final databaseResponse = await http.get(
      _marketCollectionURL);
  final Map<String, dynamic>
      _decodedDatabaseData =
      json.decode(databaseResponse.body);

  if (_decodedDatabaseData == null) {
    return [];}

  _decodedDatabaseData.forEach((id, market) {
    final temporalMarket = MarketModel.fromJson(
        market);
    temporalMarket.id = id;
    _marketsList.add(temporalMarket);});
  return _marketsList;
}
```

Lista 3: Método `getFirebaseMarkets()` del fichero `firebase_provider.dart`.

A.5. Establecer y mostrar favoritos

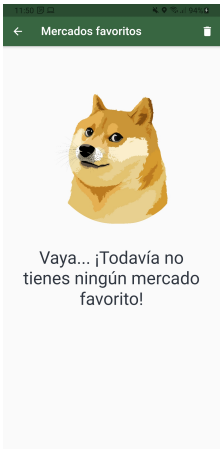

Fig. 28: Lista de mercados favoritos vacía.

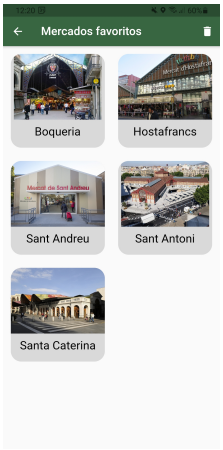

Fig. 29: Lista de mercados favoritos con cinco mercados.

```
  Widget _commerceListGenerator(
      SQLiteRealtimeProvider
      sqliteRealtimeProvider) {
if (sqliteRealtimeProvider.favoriteCommerces.
    isEmpty) {
  String noFavoritesMessage =
    'Vaya... ¡Todavía no tienes ningún comercio
        favorito!';
  return NoFavoritesOrPermissions(message:
      noFavoritesMessage);}
return GridView.builder(
  gridDelegate: const
      SliverGridDelegateWithFixedCrossAxis
  Count(
    crossAxisCount: 2,
    crossAxisSpacing: 25,
    mainAxisSpacing: 25),
  padding: EdgeInsets.all(15.0),
  itemCount: sqliteRealtimeProvider.
      favoriteCommerces.length,
  itemBuilder: (context, i) => Gesture
  Detector(
    child: RoundedSquaredButton().commerceType(
      context, sqliteRealtimeProvider.
          favoriteCommerces[i]),
    onTap: () {
      GoToMarketOrCommercePage().
          goToCommercePage(
        context, sqliteRealtimeProvider.
            favoriteCommerces[i]);},));
}
```

Lista 4: Método `_commerceListGenerator()` del fichero `favoriteCommerces_page.dart`.

A.6. Interacción con los mercados y comercios mediante otras aplicaciones

```
Column makeAPhoneCall(String phoneNumber) {
return Column(children: <Widget>[
  GestureDetector(
      onTap: () {
        _launched = usePhoneApp('tel:
            $phoneNumber');},
      child: Image.asset(
        "assets/appLinkIcons/phone.png",
        height: 50,
        width: 50,)),
    sizedBoxes.onlyHeight(height5),
    Text('$phoneAppDescription')]);
}
```

Lista 5: Método `makeAPhoneCall()` del fichero `appLinks_helper.dart`.

A.7. Mostrar mapa con marcadores y ubicación del dispositivo

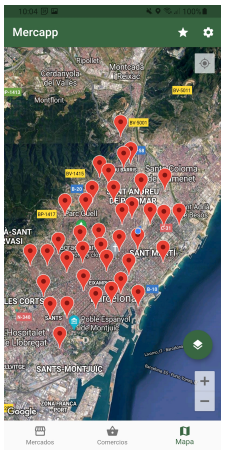

Fig. 30: Vista página del mapa en modo satélite.

Fig. 31: Vista página del mapa después de haber seleccionado uno de los marcadores disponibles.

```
getUserLocation() async {
bool _serviceEnabled;
PermissionStatus _permissionGranted;
_serviceEnabled = await location.
    serviceEnabled();
if (!_serviceEnabled) {
  _serviceEnabled = await location.
      requestService();
  if (!_serviceEnabled) {
    return;}}
_permissionGranted = await location.
    hasPermission();
if (_permissionGranted == PermissionStatus.
    denied) {
  _permissionGranted = await location.
      requestPermission();
  if (_permissionGranted != PermissionStatus.
      granted) {
    return;}}
location.onLocationChanged.listen(
  (LocationData currentLocation) {
    _locationPosition = LatLng(
      currentLocation.latitude,
      currentLocation.longitude,);
    ...
      notifyListeners();
  },);}
```

Lista 6: Método `getUserLocation()` del fichero `locations_provider.dart`.

```
Widget _markersAndMapGenerator() {
return FutureBuilder(
  future: _getfirebaseMarkets,
  builder:
      (BuildContext context, AsyncSnapshot<List
          <MarketModel>> snapshot) {
    if (snapshot.hasData) {
      final allFirebaseMarkets = snapshot.data;
      generateMarkers = allFirebaseMarkets.map
          ((firebaseMarket) {
        return Marker(
          markerId: MarkerId(firebaseMarket.id),
          position: firebaseMarket.getLatLng(
              firebaseMarket.latLng),
          draggable: false,
          onTap: () {
            _showMarkerInformation(context,
                firebaseMarket);},);
      }).toList();
      return googleMaps();
    } else {
      return Center(child:
          CircularProgressIndicator());
    }},);}
```

Lista 7: Método `_markersAndMapGenerator()` del fichero `map_page.dart`.

A.8. Otras vistas del sistema

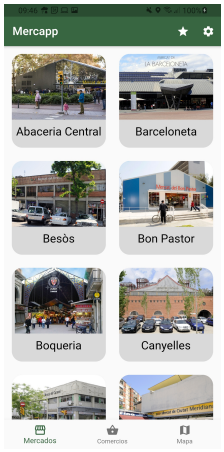

Fig. 32: Vista página de mercados.

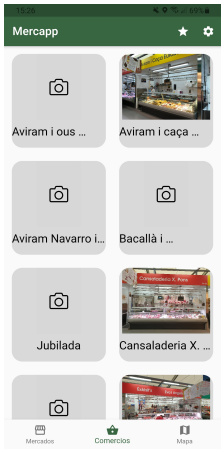

Fig. 33: Vista página de comercios.

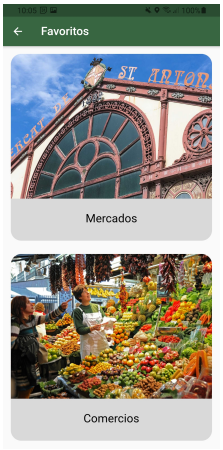

Fig. 34: Vista página de comercios.

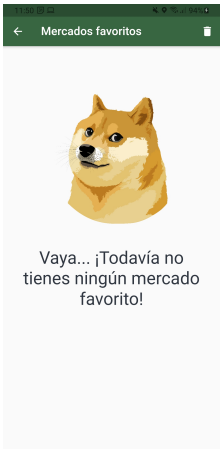

Fig. 35: Vista página de comercios favoritos vacía.

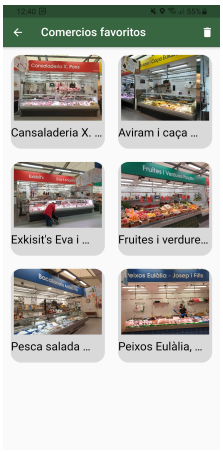

Fig. 36: Vista página de comercios.

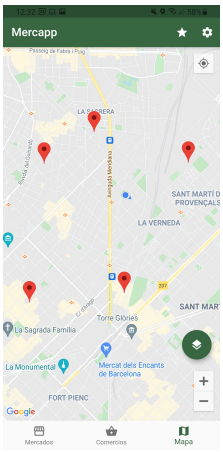

Fig. 37: Vista terreno del mapa ampliado con la ubicación del dispositivo.

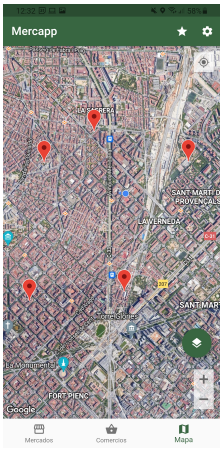

Fig. 38: Vista satélite del mapa ampliado con la ubicación del dispositivo.