Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Agrari. Un sistema de e-commerce directo del agricultor al consumidor

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Arturo Sánchez Díaz-Güemes

Tutor: Carlos Herrero Cucó

Curso 2020-2021

Resumen

Agrari es un proyecto de emprendimiento que surge como respuesta ante la falta de un producto software que promueva el consumo de productos agrícolas, ofreciendo una herramienta de comunicación y compra directa entre productores y clientes sin intermediarios; asegurando una retribución justa en todos los eslabones necesarios de esta cadena de suministro.

El proyecto ha ganado el Challenge IDEAS UPV 2020 y ha sido incorporado al ecosistema de startups de la UPV StartUPV. Además, se ha podido presentar en varias conferencias y seminarios como el Digital Enterprise Show en Madrid.

La aplicación es una *Progressive Web App* (PWA) desarrollada utilizando Angular como framework para el desarrollo web, Ionic como librería grafica multiplataforma y Capacitor para exportarlo a aplicativos móviles.

Palabras clave: Emprendimiento, aplicación móvil, comercio electrónico, sector agroalimentario

Abstract

Agrari is an entrepreneurial project that appears as a response to the lack of a software product that promotes the consumption of agricultural products, offering a tool for communication and direct purchase between producers and customers without intermediaries, ensuring a fair compensation in all the parts of the supply chain.

The project has won the Challenge IDEAS UPV 2020 and has been incorporated into the startup ecosystem of the UPV StartUPV. In addition, it has been presented at multiple conferences and seminars such as the Digital Enterprise Show in Madrid.
The application is a Progressive Web App (PWA) developed using Angular as a framework for web development, Ionic as a multiplatform graphic library and Capacitor to export it to mobile applications.

Keywords: Entrepreneurship, mobile application, e-commerce, agri-food industry

Resum

Agrari és un projecte d'emprenedoria que sorgeix com a resposta davant la falta d'un producte software que promogui el consum de productes agrícoles, oferint una ferramenta de comunicació i compra directa entre productors i clients sense intermediaris i assegurant una retribució justa entre tots els esglaons d'aquesta cadena de suministres.

El projecte ha guanyat el Challenge IDEAS UPV 2020 i ha sigut incorporat a l'ecosistema de startups de la UPV StartUPV. A més, s'ha pogut presentar en diverses conferències i seminaris com el Digital Enterprise Show a Madrid.

L'aplicació és una *Progressive Web App* (PWA) duta a terme utilitzant Angular com a framework per al desenvolupament web, Ionic com a llibreria gràfica multiplataforma i Capacitor per a exportar-lo a aplicatius mòbils.

Paraules clau: Emprenedoria, aplicatius mòbils, comerç electrònic, indústria agroalimentària

Agradecimientos

A mi familia por darme los medios y las herramientas para llegar donde estoy y ayudarme a descubrir mis pasiones.

A mis amigos, por su gran ayuda a lo largo de todos estos años de la carrera, en especial a Israel Sanjurjo Cuenca por ayudarme a superarme día a día y descubrir mi pasión y a Leo Almohano Vidick por su apoyo incondicional a lo largo de todos los años.

A Clara Espí Fayos por ayudarme en estos últimos años de la carrera y por su gran ayuda a mi desarrollo personal y profesional.

Al equipo de Agrari por permitirme acompañarlos en este proyecto y descubrir nuevos puntos de vista desde los que resolver cualquier desafío.

Por último, muchas gracias a mi tutor Carlos Herrero Cucó por apoyarme y guiarme en cada paso de la elaboración de este TFG.

Tabla de contenidos

1. Introducción

1.1. Motivación

En la actualidad el mercado del e-commerce supone un canal de ventas imprescindible para muchos negocios, suponiendo un total del 19,5% del total de las ventas mundiales [7]. Este aumento de ventas ha supuesto un cambio radical que ha forzado a muchas empresas a crear nuevos canales de ventas online donde publicar sus productos y ha permitido a nuevas empresas proliferar. Algunos ejemplos de estas empresas que ofrecen servicios de venta en línea son Amazon con la venta de todo tipo de productos, Vinted con la venta de ropa de segunda mano o Uber Eats con su servicio de entrega de comida a domicilio.

Aunque son muchas las empresas que han sabido adaptarse a estos cambios tecnológicos, con casi el 23,3% de las empresas pequeñas vendiendo online, aun existen diversos sectores reacios a estos cambios [21]. Uno de estos grandes sectores es el agrónomo, donde tan solo un 0,6% de las ventas de productos frescos se realizan de manera online [15]. Además de esta brecha tecnológica que sufre el sector por diversos motivos, también se observan unos márgenes en la cadena de valor que aumentan el precio de los productos en hasta un 600% que no es percibido por el productor [14].

Ante esta problemática nace la idea de Agrari, una aplicación móvil que busca conectar compradores interesados en un producto fresco, de calidad y local, con agricultores de la zona que puedan proveer a estos clientes de forma barata y asegurando una reducción de coste de intermediarios.

El desarrollo de esta idea surgió a partir de la técnica *Design Thinking* y en el desarrollo se ha seguido la metodología de *Lean Startup*. Esta metodología permitió un desarrollo ágil iterativo definido en una serie de *sprints*, tras los cuales se evaluaron los resultados y se tomaron acciones directas en función de los resultados observados.

Este TFG es llevado a cabo con el objetivo de aplicar los conocimientos obtenidos a lo largo de la carrera en un proyecto real dentro de un contexto de emprendimiento apoyado por StartUPV y el *Mercat Agroecològic* de la UPV.

1.2. Objetivos

El objetivo de este trabajo es documentar el desarrollo y puesta en marcha de un proyecto de ingeniería informática en el ámbito del emprendimiento, creando una aplicación multiplataforma (web y móvil) que permita a los agricultores publicar sus productos para su venta, permitiendo así que se preparen los pedidos realizados con antelación, además de proporcionar un canal de comunicación entre clientes y productores.

Para cumplir estos objetivos definidos al inicio del proyecto se establecieron las siguientes pruebas para su validación:

- Realización de un estudio del mercado actual, para así descubrir herramientas similares ya existentes.
- Análisis de la viabilidad económica del proyecto a largo plazo
- Generación de una aplicación modular, altamente escalable y que asegure la privacidad de los datos de usuarios.
- Validación del producto en un entorno real y realización de encuestas para verificar las suposiciones hechas a cerca del mercado.

Los objetivos que conciernen a este trabajo de fin de grado son los siguientes:

- Poner a prueba en un entorno real todo el conocimiento adquirido a cerca del proceso de desarrollo del software, ligando así varias asignaturas en el proceso.
- Llevar a cabo un proceso de desarrollo software ágil en un marco de contexto de emprendimiento.
- Aprendizaje sobre el despliegue de proyectos en ámbitos de preproducción y producción.

1.3. Equipo de trabajo

El proyecto de Agrari es un trabajo conjunto entre varios miembros de distintos ámbitos. Los miembros que han participado en el proyecto son los siguientes:

- **Arturo Sánchez Diaz-Güemes:** Actúa en el papel de CTO (Director de Tecnología) de la empresa. Desarrollo de la aplicación móvil, configuración del *backend*, lanzamiento del proyecto, despliegue del servidor web y la web, desarrollo de las pruebas de validación con los clientes y despliegue del proyecto. Respecto a la idea de negocio ha contribuido creando una tabla de características comparando los distintos aplicativos existentes.
- **Jaume Merino Murgui:** Lleva a cabo el rol de CEO (Director Ejecutivo) en la empresa. Se ha hecho cargo del desarrollo de la idea de negocio, del estudio del mercado y de elaborar el análisis *DAFO* del proyecto.
- **Jesús Deusa Alonso:** COO (Director de operaciones) de la empresa, a cargo de la parte de relaciones con el cliente, conferencias y logística de envíos.
- **Aitana Castellanos Román:** Es la CMO (Directora de Marketing) de la empresa, encargada de la creación de la identidad de Agrari, elaborando un logo, una paleta de colores, una selección de fuentes y un manual corporativo de la marca. También ha elaborado los mockups de alta fidelidad y la experiencia de usuario que se aplicaron a lo largo del proyecto, estos *mockups* entran en juego en el segundo incremento del MVP.

1.4. Estructura de la memoria

La estructura de la memoria es la siguiente:

En el primer capítulo se introduce el proyecto, y la motivación por la que se ha decidido llevar a cabo, los objetivos que buscan cumplirse durante el desarrollo y por ultimo los miembros que conforman el equipo y sus roles en este.

En el segundo capítulo se muestra cómo surge la idea de negocio, comentando todo el proceso desde los orígenes de la idea por parte de Jaume Merino y Jesús Deusa, hasta la idea actual de Agrari. Además, se demuestra el análisis realizado tras la generación de la idea, mediante la evaluación de la idea de negocio. En esta se incluye el razonamiento detrás de las decisiones que marcan la diferenciación de nuestro producto comparándolo con aplicaciones similares, mostrando un estudio de mercado que elaborado a partir de las técnicas de análisis *DAFO* y *Lean Canvas*. Para dar fin a este punto se recapitula lo ya expuesto mediante una breve conclusión inferida de los datos previamente presentados.

En el cuarto capítulo se detalla el proceso de desarrollo del proyecto, desde la fase de análisis, preparación, etc. hasta la fase de despliegue del proyecto, incluyendo diversos retos resueltos en el proceso de implementación.

En el quinto capitulo se elabora la distribución y gestión del tiempo destinado a este trabajo de fin de grado, detallando así las distintas fases en las que se ha dividido el proyecto y el *roadmap* con las fechas establecidas que se ha completado.

En el sexto capitulo se presenta una conclusión sobre todo el trabajo realizado, haciendo una reflexión sobre el cumplimiento de los objetivos establecidos al inicio del proyecto y futuras tareas o áreas de mejora que se han detectado a lo largo del desarrollo del proyecto.

Finalmente se añade como documento anexo una guía de usuario de la aplicación redactada con el objetivo de ayudar al usuario final con el funcionamiento del producto.

2. Evaluación de la idea de negocio

Este capítulo está dedicado a la generación y análisis de la idea de negocio, el estudio de mercado, elaboración del mapa de características para identificar nuestra diferenciación, el modelo de negocio que se ha seguido en el proyecto y la proyección económica de este. Por último, se realiza una breve conclusión sobre la viabilidad de la idea y la propuesta de valor que se busca ofrecer para obtener esa distinción sobre el resto de los productos que existen actualmente en el mercado.

Aunque algunos apartados de este capítulo no fueron realizados por el autor del TFG, es de vital importancia la información resultante de estos trabajos de investigación y análisis pues los datos que proporcionan se ven reflejados directamente en las decisiones de desarrollo que fueron tomadas en este proyecto.

En la realización de la sección 2.1, participaron Jaume Merino y Jesús Deusa. Las secciones 2.2, 2.7 las elaboró Jaume Merino de manera exclusiva y la sección 2.5 es el resultado de una tarea conjunta de todo el equipo.

2.1. Resumen de Agrari y su evolución

Agrari nació a través de la técnica de *Design Thinking* por parte de Jaume Merino y de Jesús Deusa en verano de 2020, se planteó originalmente como una app que permitiera a cualquier usuario, tanto profesional como amateur, publicar sus productos a la venta para su posterior reparto a domicilio.

Tras evaluar la idea y analizando el mercado actual se optó por dar otro giro a esta idea incorporando aspectos de red social que permitan generar una comunidad activa en la app que esté bien informada del producto que quieren comprar y que puedan compartir sus experiencias en un canal específico para ellos.

Este aspecto social se ha resuelto a través de la creación de un sistema de chat directo entre usuario y productor que permite resolver todas las dudas que puedan surgir. Además, también se planteó la creación de un sistema de blog y noticias de actualidad que permitiese a los usuarios estar concienciados de las causas locales más importantes del momento.

Dado el gran tamaño del proyecto para el primer MVP se redujo el alcance drásticamente y se limitaron las funcionalidades para acabar obteniendo un *marketplace* que permitiese crear pedidos que serían recogidos en un mercado de forma posterior. Con el objetivo de validar la necesidad de un canal de comunicación entre productor y cliente se introdujeron una serie de funcionalidades que diesen un alma de red social como las reseñas de productos o el chat directo entre usuarios.

2.2. Lean Canvas

Lean Canvas es una herramienta ampliamente utilizada en el mundo empresarial que permite visualizar de manera rápida y resumida todos los elementos clave de un proyecto. Crear esta tipo de tabla facilita exponer un proyecto de negocio a otras personas y transmitir de manera mas eficaz los pilares en los que se basa el proyecto.

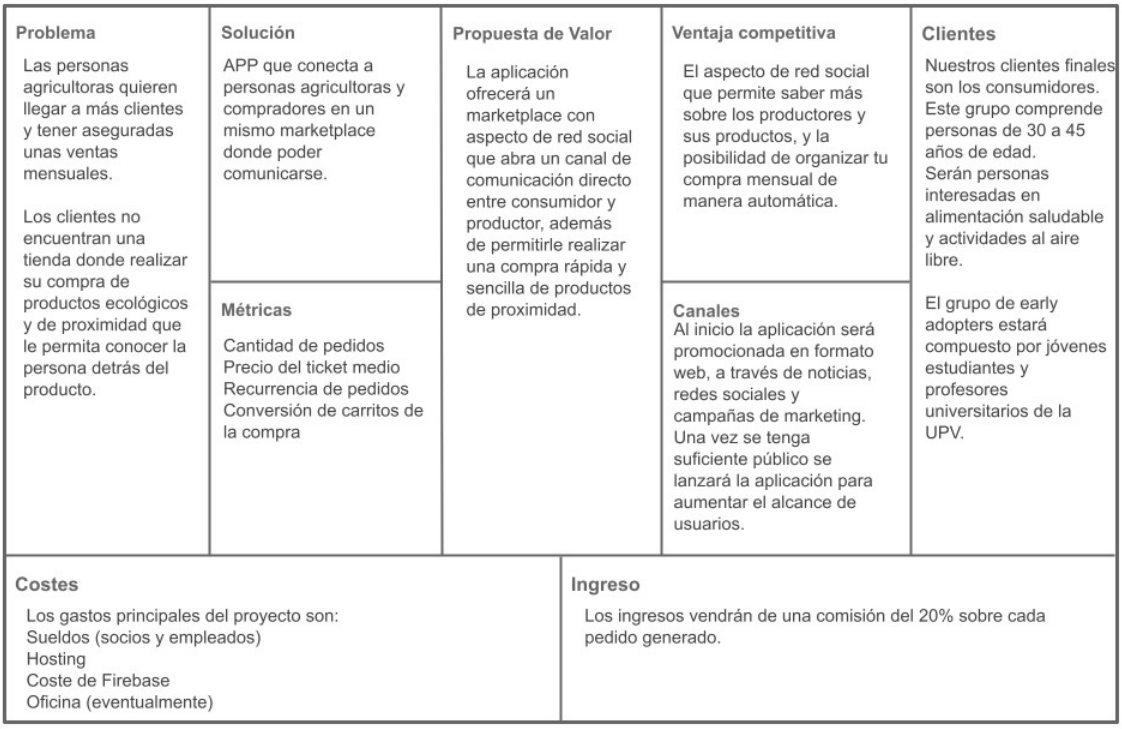


Figura 1. Lean Canvas

2.3. Estudio de mercado

Antes de llevar a cabo la idea de negocio es necesario realizar un estudio de mercado previo de otras herramientas similares ya existentes para poder entender que funcionalidades son básicas, cuales son clave y cuáles pueden ser añadidas a nuestra propuesta de valor que no forman parte de ninguna de las soluciones existentes.

Con este objetivo se inicia un proceso de recopilación de los principales competidores que existen en el mundo del comercio electrónico de productos de proximidad sostenibles que van directos del productor al cliente. Este apartado fue realizado íntegramente como parte del TFG por su autor.

Farmidable:

Farmidable es un Marketplace web que busca ofrecer una compra online con una experiencia lo más parecida posible a un supermercado online. Esto lo consigue ofreciendo categorías de compra similares a los pasillos de un supermercado y con una variedad de productos muy dispersa, incluyendo incluso jabones.

Esta web ofrece además del envío a domicilio de los productos un sistema de puntos de recogida que permiten a los usuarios acercarse a diversos ubicaciones repartidas por la ciudad y recoger sus pedidos en unas franjas de tiempo específicas.

A diferencia de Agrari solo puedes hacer pedidos en la comunidad de Madrid y no busca promover y empoderar a sus productores mostrando información sobre las personas que producen cada alimento ni dar herramientas de red social para promover su interacción.

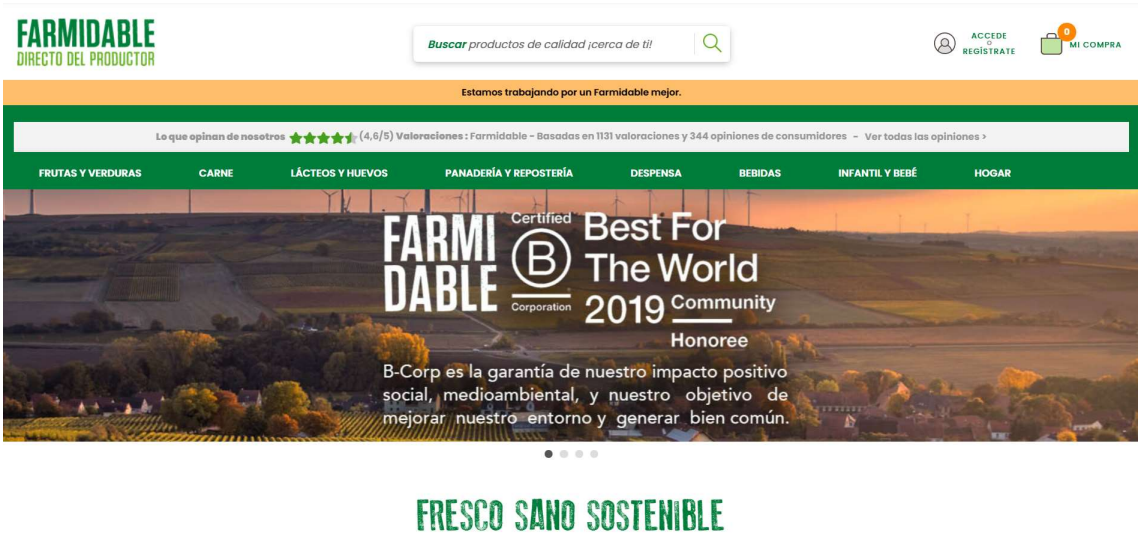


Figura 2. Web Farmidable

Terreta Sana:

Es una plataforma web de compra de proximidad en Valencia con envío a domicilio que está enfocada a los agricultores, ofrecen mucha sensibilidad hacia los productores y abogan por la sostenibilidad en todos los ámbitos. Asimismo, permite poder colaborar con las distintas causas que buscan promover.

Permite hacer pedidos de una amplia variedad de productos y elegir el momento que mejor nos venga para recibirlos en los días que realizan repartos. Además, también nos permite seleccionar una periodicidad para nuestro pedido y de esta forma recibir cada semana, quincena o mes las cestas que deseemos.

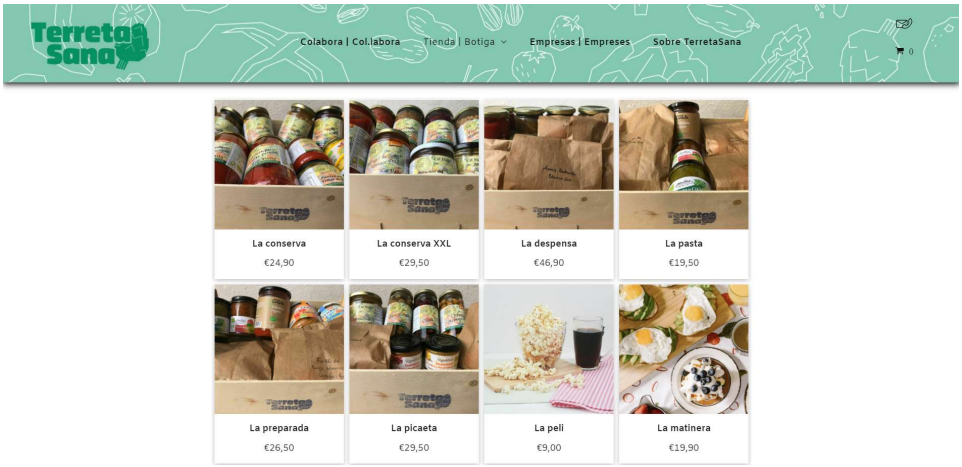


Figura 3. Web Terreta Sana

Aunque esta plataforma sea mucho más sensible con los temas sociales no promueve los diversos productores, sino que están muy centrados en los productos y no se informa de quien es la persona detrás de estos.

Correos Market:

Correos Market es la plataforma de compra online enfocada a personas productoras rurales con el objetivo de luchar contra la despoblación de estas áreas. Al ser una empresa de Correos realizan envíos a todo el territorio nacional y cuentan con una parte logística muy potente. Además de ofrecer productos alimenticios también venden artesanía y tecnología.

Una buena funcionalidad que incluye es la de promover a los productores, permitiéndonos ver perfiles personales de estos, con sus productos individuales y algo de información acerca de estos.

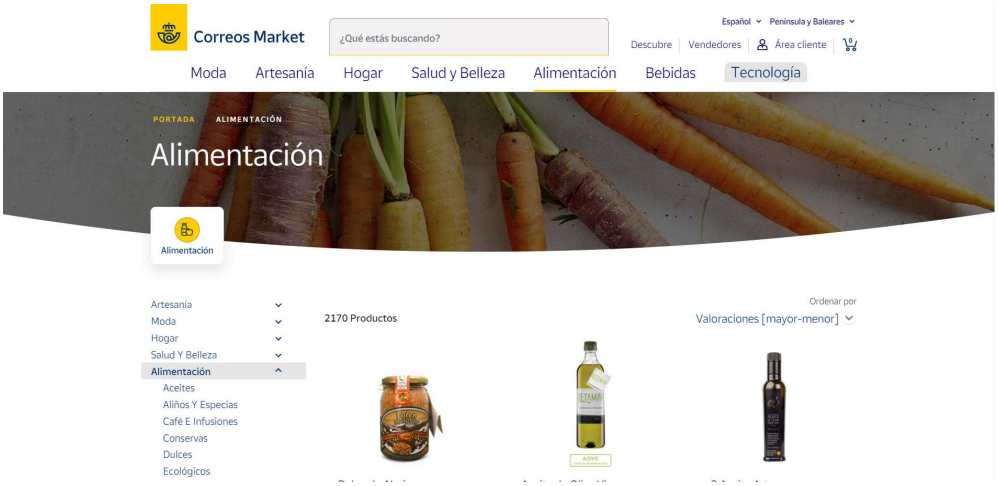


Figura 4. Web Correos Market

A diferencia de Agrari no incluye ninguna parte social ni de interacción que permita al usuario sentir que forma parte de un grupo y que apoya a estas personas.

La compra así da gusto:

La compra así da gusto es una propuesta similar a la de Farmidable pero incluye algunas funcionalidades clave que la diferencian de esta, como podría ser la compra de productos en base a recetas (similar a las cajas, pero orientada a preparar un alimento).

Además de esto también nos permiten comprar platos preparados que tan solo deben ser calentados para consumir.

Figura 5. Web La compra así da gusto

Aunque introduce propuestas creativas y funcionalidades nuevas que el resto no, esta sigue sin tener como foco principal el promover la sostenibilidad y el aspecto más social y de apoyo hacia los productores.

FreshCo's:

Freshco's es un Marketplace web que busca hacer lo más sencillo posible la compra ecológica de productos. Para ayudar a ampliar su mercado ofrecen además el formato app para poder hacer los pedidos de forma más sencilla.

Aunque a diferencia del resto de competidores ofrece un proceso muy sencillo de compra similar al de cualquier tienda en línea actual y no se limita tan solo al mercado web, no tiene un enfoque orientado al productor como es el de Agrari y todos los productos son muy simples, tan solo informan de su precio, nombre y cantidad a comprar, no da una vista de quién es el productor, tampoco permite conocer más acerca del producto que vamos a comprar y su calidad.

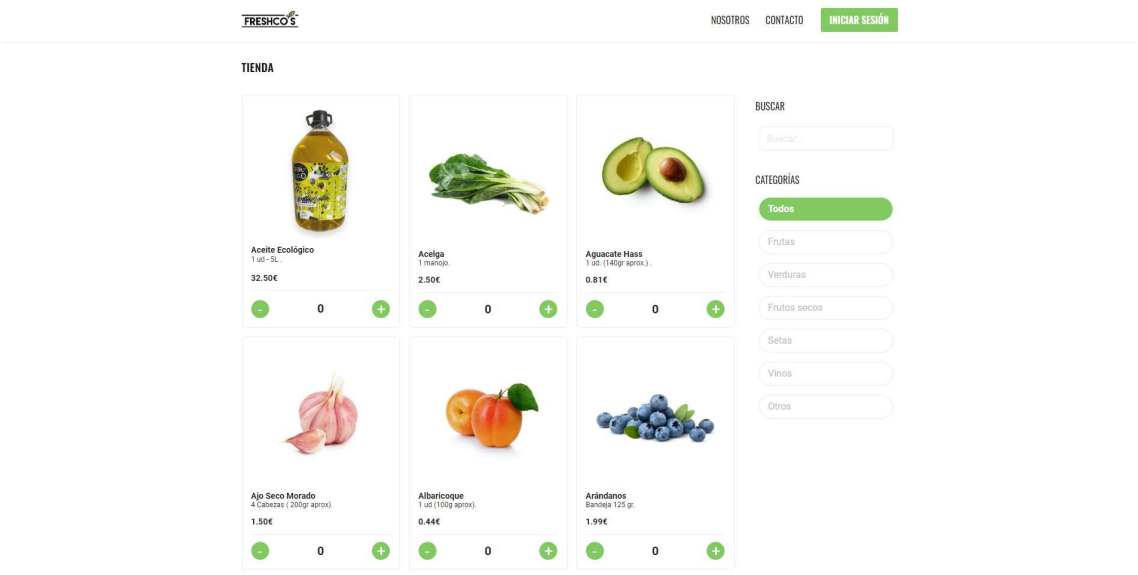


Figura 6. Web FreshCo's

APPRECIEM:

Appreciem es un Marketplace que busca acercar lo mejor del mercado local a la compra online. Esto lo hace creando una experiencia de compra sencilla donde además se permite al comprador elegir los productores que le proveerán su compra. Las fichas que ofrece de cada producto son muy informativas y muestran una total transparencia respecto al comercio que los vende.

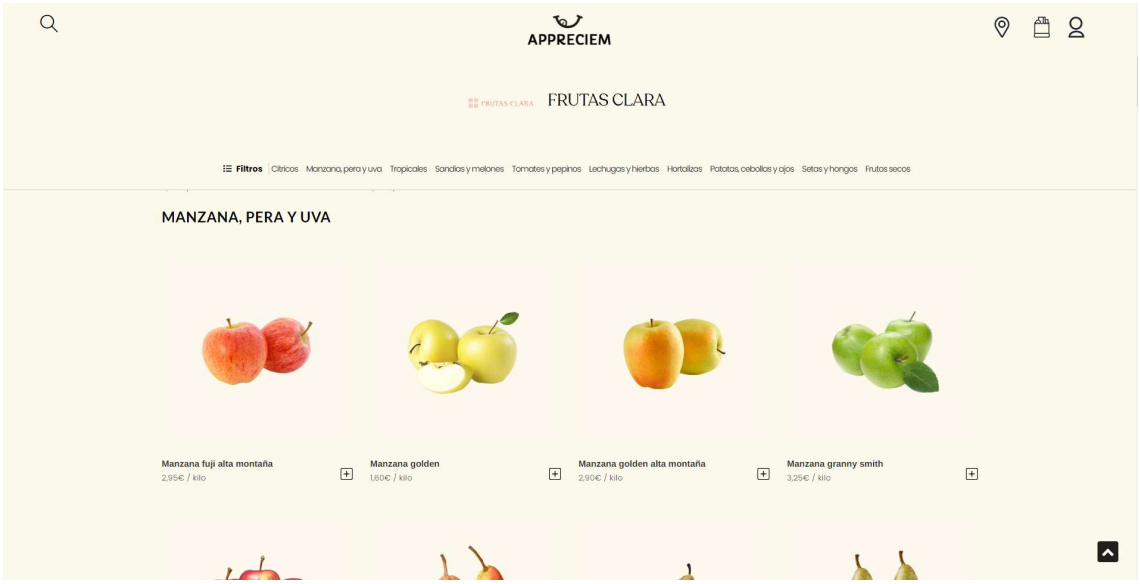


Figura 7. Web Appreciem

Aunque Appreciem haga envíos a domicilio, estos envíos son directos del local al consumidor por lo que se pueden generar varios repartos dentro de un mismo pedido, Agrari busca evitar esto realizando un envío único al usuario con todos los productos que ha pedido.

2.4. Tabla de características

Característica	Farmidable	Terreta Sana	Correos market	La compra así da gusto	FreshCo's	Appreciem	Agrari
Descuentos y promociones		X*		X	X		X
Blog/Noticias	X						X
Reseña de productos	X		X				X
Información del productor		X	X	X		X	X
Chat con productores							X
Recogida en Mercados	X	X	X			X	X
Envíos a domicilio	X	X	X	X	X	X	X
Estadísticas para los productores							X
Formato app					X		X
Ubicaciones de los productores			X			X	X
Filtrado o búsqueda de productos	X	X**	X	X	X	X	X
Cestas o cajas	X	X			X		
Pago *online*	X	X***	X	X	X	X	X
Repetición de pedidos anteriores							X
Recurrencia de pedidos		X					X
Recomendación de productos	X				X		X
Etiquetas o certificaciones	X	X					X
Sistema de cuentas	X		X		X	X	X
Pedido unificado	X	X	X	X	X		X
Compra basada en recetas				X			
Sistema de suscripciones	X						X

* No es automático, te lo dan cuando haces varios pedidos seguidos de manera manual
** Tan solo deja filtrar por categoría, no permite por producto
*** Tan solo a través de Bizum

2.5. Modelo de negocio

Uno de los aspectos más importantes de una idea de negocio es el modelo detrás de esta idea que servirá para validar su rentabilidad a largo plazo. Para el proyecto de Agrari se tuvieron en cuenta varios modelos de negocio entre ellos la introducción de anuncios dentro de la aplicación, esta opción fue desestimada dado que los ingresos dependen directamente del público que utilice la aplicación y generalmente este tipo de monetización no suele tener una buena acogida por parte del usuario final.

Una opción que también se planteó utilizar, una vez el volumen de ventas aumente, es el cobro de una suscripción por el uso de los servicios de la plataforma, como herramientas de gestión avanzadas y generación de albaranes, en función del volumen de ventas de cada usuario. También se propuso implantar anuncios destacados que den un mejor posicionamiento a los productores dentro del buscador de productos.

Finalmente, en Agrari se decidió que los ingresos surgirán principalmente a través de una comisión por cada pedido realizado en la plataforma como es habitual en los *marketplaces online* como Amazon o eBay. Basado en el margen de beneficio que existe actualmente en los productos agrícolas desde que el producto sale del campo hasta que llega a las estanterías de las tiendas (que puede llegar incluso al 600% en algunos casos), se busca reducir ampliamente estos márgenes abusivos ofreciendo inicialmente una comisión del 20% sobre el precio final del pedido, pero se buscará reducir ese porcentaje según el negocio aumente sus ingresos para poder ofrecer mejores condiciones a los productores.

2.6. Proyección económica

Para validar la viabilidad económica del proyecto se desarrolló una proyección económica a 5 años. Esta proyección es un cálculo en el que se tuvieron en cuenta todos los gastos de la empresa (fijos y variables) y los ingresos que se esperaban obtener cada año. La proyección económica se realizó a partir del inicio de la actividad económica en 2022 y fue realizada por el autor de este TFG.

Para comenzar se analizaron los costes del personal del equipo, se consideró que el equipo de trabajo estará compuesto por 4 personas, con la ayuda temporal de 2 personas a tiempo parcial, y en 2026 un total de 4 personas a tiempo completo. Para el sueldo se consideró comenzar con el salario mínimo interprofesional e ir aumentando paulatinamente según aumenten los ingresos y volumen de trabajo.

Por otro lado, se utilizaron como gastos variables el coste de la logística de pedidos (embalaje de productos, transporte, y otros costes extras). Estos costes son directamente proporcionales con el aumento de ventas que se pudiesen percibir.

Respecto a los costes fijos se incluyeron aquellos gastos que son constantes sin importar el aumento de ventas los gastos de publicidad y promoción de la aplicación, o los gastos del mantenimiento de los servidores web. También se tuvo en cuenta el coste del alquiler de una oficina para el trabajo del equipo y se consideró el coste de adquirir equipos informáticos para los miembros del equipo.

Concepto	2022	2023	2024	2025	2026
Coste del dominio	10€	10€	10€	10€	10€
Hosting	360€	520€	650€	1020€	2150€
Coste Firebase	400€	1060€	2000€	4080€	8640€
Anuncios	1.000€	2.000€	4.000€	8.000€	16.000€
Licencia iOS	100€	100€	100€	100€	100€
Licencia Android	50€	0€	0€	0€	0€
Sueldo socios	82.410€	98.952€	148.428€	213.736€	277.857€
Sueldos empleados	15.000€	19.000€	50.000€	60.000€	90.000€
Alquiler oficinas	0€	0€	12.000€	20.000€	20.000€
Total	99.330€	121.642€	217.188€	306.946€	414.757€

El cálculo total se obtuvo utilizando como dato base el consumo anual que cada persona en España gasta en frutas y verduras, el cual supone un total de 454 euros [8] (aunque la oferta de Agrari no se limitaba a estos productos, estos se cuentan como el principal volumen de ventas y por lo tanto se decidió usar esta cifra más modesta para un cálculo más realista). Esta cantidad es el importe facturado, para calcular el beneficio total que se obtendrá sobre cada pedido medio se calculó solo el 20% (el margen de intermediario) por lo que el beneficio final resultó en 91 euros por carrito de compra medio.

Concepto	2022	2023	2024	2025	2026
Clientes anuales	500	1000	3600	7200	18000
Cantidad facturada (451€)	227.000€	454.000€	1.634.400€	3.268.800€	8.172.000€
Margen de intermediario (91€)	45.500€	91.000€	327.600€	655.200€	1.638.000€

Utilizando los datos calculados anteriormente se estimó el coste de mantenimiento del proyecto a 5 años para validar la viabilidad económica de este. Como se puede observar, el proyecto se calculó rentable a partir del tercer año, aunque el balance de caja no fuese positivo hasta el tercer año con un total de 110412€ de beneficio, con un total de 37940€ en caja (descontando la deuda acumulada en los dos años previos)

Concepto	2022	2023	2024	2025	2026
Beneficios Anuales	-53.830€	-30.642€	110.412€	348.254€	1.223.243€
Balance de caja	-53.830€	-84.472€	37.940€	406.194€	1.649.437€

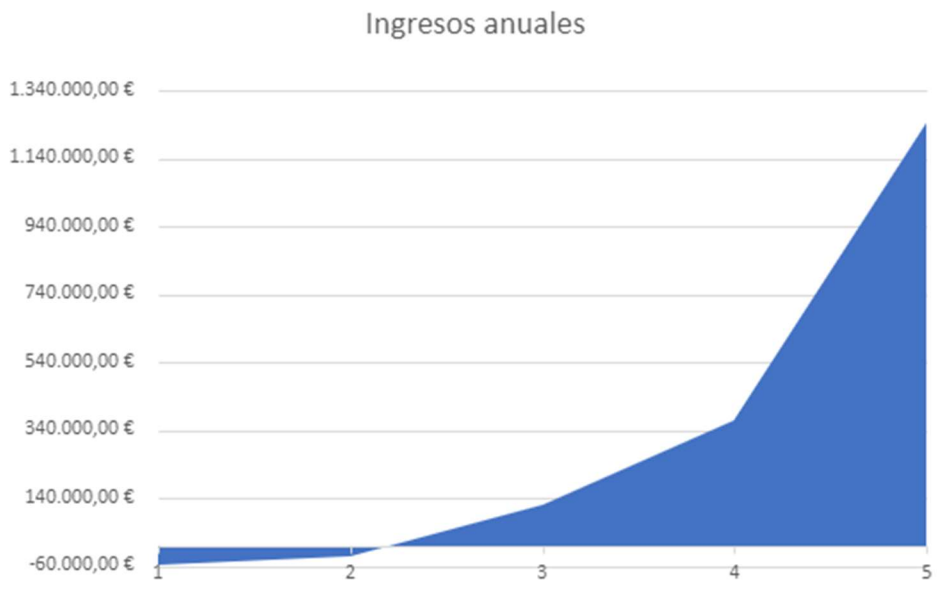


Figura 8. Ingresos Anuales

2.7. Análisis *DAFO*

El análisis *DAFO* (Debilidades, Amenazas, Fortalezas y Oportunidades) es una herramienta que permite saber la situación en la que se encuentra Agrari actualmente y cuáles son los distintos factores de éxito o de riesgo que puede encontrarse un proyecto empresarial. Este análisis es utilizado frecuentemente en las *startups* dado que permite detectar posibles ventajas competitivas respecto a la competición que se ha encontrado y cuáles son los puntos en los que se debe prestar especial atención y mejorar para conseguir sacar adelante dicho proyecto.

Debilidades	Amenazas
• Ya existen otras soluciones similares con un modelo de negocio similar	• Brecha tecnológica de algunos agricultores
Fortalezas	**Oportunidades**
• Sentimiento de pertenencia y cercanía en un ecosistema social y respetable.	• Convergencia de los valores de sostenibilidad en fase de crecimiento. • Plataforma móvil accesible

Como se puede ver en la tabla, una de las mayores amenazas será la brecha tecnológica que pueden sufrir algunos agricultores y el posible elevado coste de aprendizaje de uso de la plataforma. Aunque ya existen soluciones similares, no todas incluyen las funcionalidades que busca ofrecer Agrari, y ninguna está fuertemente establecida, por lo que si el producto ofrece mejores funcionalidades es probable que el público que haga uso del *marketplace* quiera quedarse con esta solución.

Por otro lado, Agrari ofrece un gran valor añadido con su idea de ecosistema social que busca promover el contacto entre agricultores y clientes que podría ayudar a promover mucho la propuesta social que ofrecemos, además el mundo de la agricultura sostenible y la compra de proximidad está en auge y ofrecer esto en un formato accesible para el consumidor como pueden ser las *apps* con las que ya están altamente familiarizados.

2.8. Conclusión de la evaluación de la idea de negocio

Como conclusión se destaca que Agrari era un proyecto de emprendimiento muy ambicioso con muchas posibilidades, aunque esto también llevaba muchos riesgos como pueden ser las distintas soluciones similares ya existentes, de las cuales se debía diferenciar fuertemente. Con este conocimiento se elaboró una serie de propuestas de valor únicas que fueron validadas a lo largo del MVP para asegurar su adopción en el mercado y así poder escalar la idea en caso de conseguir cumplir todos los hitos planteados tras la finalización de este TFG.

El proyecto resultaría viable a partir del tercer año de desarrollo, teniendo tan solo 3600 familias realizando la compra mensual de fruta y verdura, y un total de 20 productores abasteciendo alimentos a estos clientes. Considerando el amplio margen que existe en los productos y el valor total del sector agroalimentario sobre el PIB de España es del 9,1% lo cual se traduce en casi 100.000 millones de euros [9] la proyección se consideró modesta y razonable.

Ante una proyección económica favorable y una propuesta de diferenciación clave al resto de los productos del mercado se concluyó que Agrari era un proyecto que tenía su propio hueco de mercado y amplias oportunidades para crecer.

3. Desarrollo de la idea de negocio

En esta sección del TFG se detalla como fue el proceso de desarrollo de Agrari a lo largo de los dos experimentos por la parte del autor de este trabajo, desde la fase de definición de las unidades de trabajo, el flujo de trabajo seguido, su implementación y finalmente la validación con los *early adopters* en el *Mercat Agroecològic* de la UPV.

Para la organización del proyecto se decidió utilizar la herramienta Trello, que permite crear tableros Kanban en el cual podemos agregar distintas cartas que van avanzando por el flujo definido.

En todo momento las tareas estaban organizadas, cuanto más alto en la lista de tarjetas mayor prioridad respecto a las otras. Esta organización se mantuvo en todo momento a lo largo del desarrollo del proyecto.

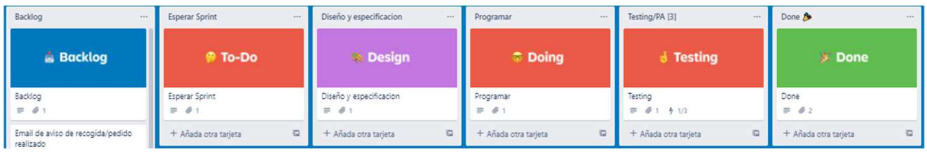


Figura 9. Fases del desarrollo Kanban

El flujo definido fue el siguiente:

- *Backlog*:
  - Aquí es donde se incluyeron todas las tareas al registrarse en las distintas sesiones de *brainstorming*, reuniones con los diversos *stakeholders* y recomendaciones del tutor.
- Esperar *sprint*:
  - En esta fase se filtraron todas aquellas unidades de trabajo candidatas a ser abordadas en el próximo incremento del MVP, las tareas que entraron aquí podían ser desestimadas en caso de que hubiese más carga de trabajo de la que se pudiese abordar. En esta sección se especificaron los requisitos de cada unidad de trabajo, así como una estimación de las horas de trabajo que podían suponer. Un ejemplo de tarea especificada puede verse en la figura 10.

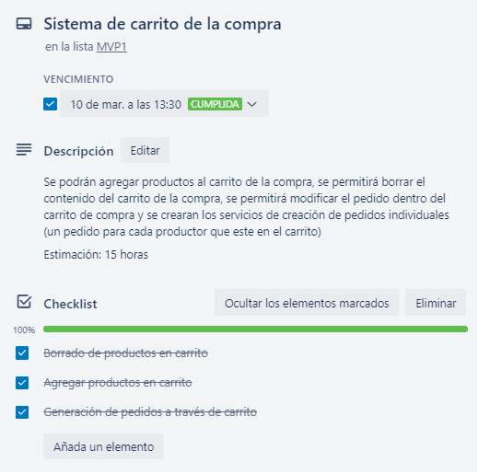


Figura 10. Ejemplo de especificación de una unidad de trabajo

- Diseño:
  - Esta fase es donde se preparó cada tarea y se definió el diseño de la implementación utilizando un mockup. El diseño de los mockups del primer MVP forman parte del TFG y fueron realizados por el autor, mientras que los mockups que se utilizaron para la segunda prueba fueron realizados por Aitana Castellanos.
- Programar:
  - Esta es la fase en la cual la unidad de trabajo se llevó acabo, cada ut fue desarrollada como un paquete de trabajo y se intentó mantener en todo momento el trabajo en progreso por debajo de 2 tareas para evitar sobrecarga y priorizar el cerrar tareas.
- Validación:
  - En esta fase se aplicaron las distintas pruebas que validaban su correcto funcionamiento, esta validación fue realizada por el programador y por algún miembro del equipo ajeno al desarrollo para verificar el correcto funcionamiento.
- Terminado
  - Aquí se quedaron las tareas una vez acabadas. Para facilitar la identificación de las tareas abordadas en cada *sprint* de desarrollo, estas tareas fueron incluidas en una lista con un identificador único que representa este *sprint*.

3.1. Desarrollo del primer MVP

Para desarrollar el primer MVP se marcó como objetivo desarrollar una aplicación móvil que permitiese a los clientes hacer un pedido para recoger en el *Mercat Agroecològic* de la UPV y que los productores pudiesen prepararlo antes de la fecha de recogida.

Para elegir las diversas Unidades de Trabajo que se incluyeron en este primer MVP se realizó una sesión de lluvia de ideas para encontrar posibles soluciones a los problemas que busca solucionar Agrari. Para elegir las tareas que realmente aportan valor y por lo tanto deben ser más prioritarias se elaboró un mapa MoSCoW (del inglés, *Must Should Could Won't*).

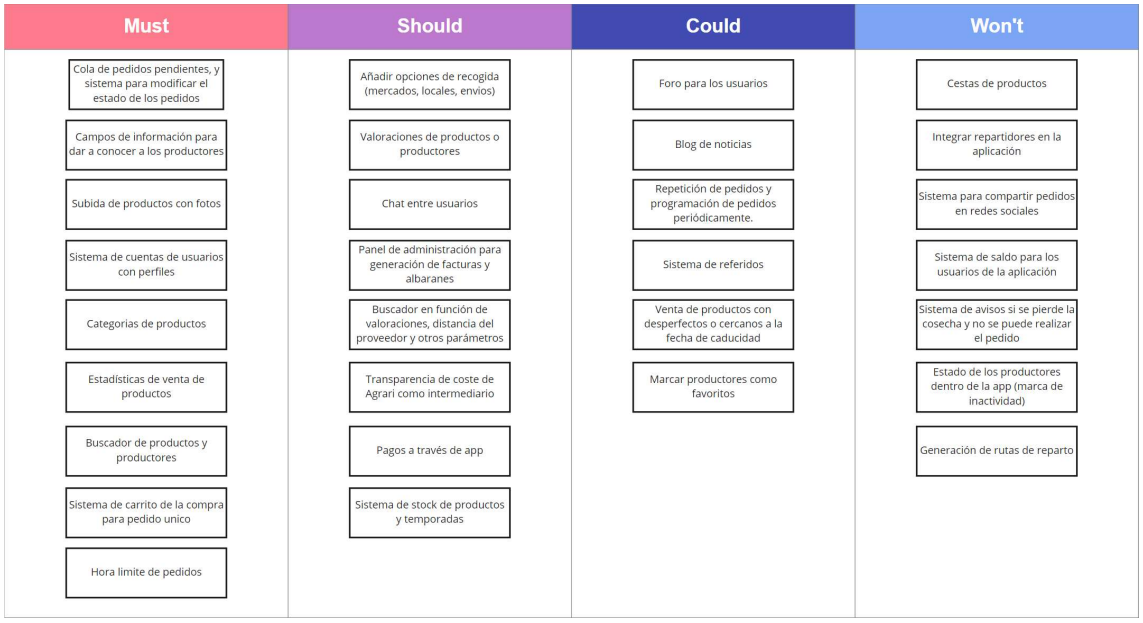


Figura 11. Tablero MoSCoW

El sistema MoSCoW permite organizar tareas según su importancia en función de las siguientes claves:

- *Must*: tareas indispensables para el éxito del proyecto y por lo tanto deberán realizarse.
- *Should*: tareas que deberían abordarse a lo largo de la vida del proyecto pero que no son clave para el éxito del proyecto.
- *Could*: tareas que pueden aportar valor al producto, pero no da un aspecto diferencial único que permita ganar una ventaja competitiva.
- *Won't*: Tareas desestimadas completamente y que no serán contempladas para el producto final.

Tras realizar esta organización se detectaron factores clave que deberán ser realizados en futuros incrementos del *MVP* y por lo tanto se indicaron en rojo. Aquellas tareas necesarias para lanzar el primer *MVP* fueron indicadas en verde. Y, por último, las tareas indicadas en naranja son las que se contemplaba no abordar en caso de no existir la capacidad suficiente en el equipo de trabajo.

De esta forma surgen las siguientes *UT*:

- **Sistema de Registro/Inicio de sesión:** se desarrolla un registro para los dos roles de usuarios que acepta la aplicación, productor y cliente, y así permitir el interactuar con la aplicación. Este registro es mantenido por Firebase y la aplicación tan solo se encargará de enviar los datos necesarios para crear y consultar las cuentas. En el registro se muestra un enlace a los términos y condiciones de uso de la aplicación y la ley de protección de datos que deberá aceptar el usuario para poder crear su cuenta.
- **Perfiles de usuario:** se prepara una vista simple de los datos de los dos tipos de usuarios distintos y se mostrará ahí la información relevante. En el caso de que el usuario sea un productor se mostrara un botón para acceder a su perfil.

- **Listado de productores:** se elabora un servicio que recuperará la información de todos aquellos usuarios con el rol de productor y se mostrará al usuario un listado de estos que le dirigirá a su perfil para que pueda ver sus productos publicados.
- **Sistema de productos:** se crea un formulario con los campos correspondientes al modelo de datos para almacenar en base de datos esta información. Se crea además un servicio que permita recuperar la información y obtener un listado de productos. Al seleccionar uno de estos productos se redirige al usuario a una vista donde se muestran los datos relevantes del producto.
- **Búsqueda de productos:** se elabora un buscador en el cual el usuario puede introducir texto y este será utilizado para filtrar la lista de productos obtenida en esa vista. También se elaboran filtros para mostrar tan solo productos de algún productor en concreto y así mostrar en su perfil tan solo los productos que él tiene publicados, y otro filtro que muestre solo los productos de una categoría en concreto.
- **Sistema de subida de imágenes:** se elabora un servicio que dada una ruta y un archivo en base64 los sube al almacenamiento de Firebase y relaciona el documento pertinente con ese archivo. Esta subida podrá realizarse de dos formas distintas, en ordenador se pedirá al usuario seleccionar en un archivo, pero desde la aplicación móvil se lanzará la cámara del dispositivo desde la cual se puede tomar una foto y posteriormente subirla o seleccionar una foto de la galería.
- **Sistema de carrito de la compra:** se añade en las vistas del producto un selector de cantidad y un calculador del precio para poder añadir ese producto al carrito. Estos productos se podrán ver reflejados en una nueva vista donde se mostrará al usuario el listado completo de productos y la opción de modificar la cantidad de productos introducida.

El diagrama de casos de uso resultante de estas unidades de trabajo es el siguiente:

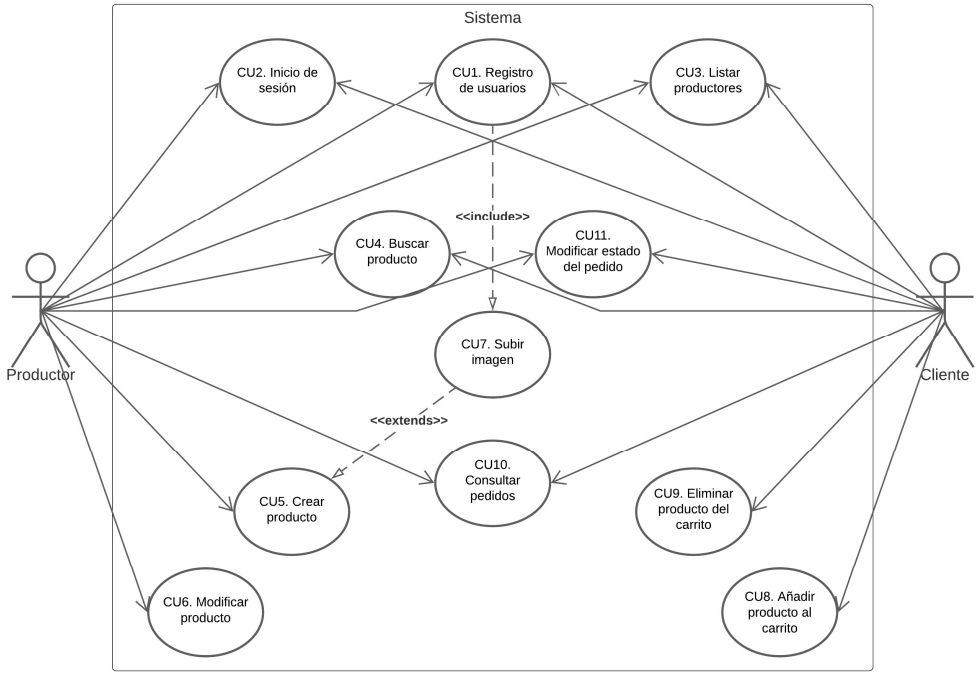


Figura 12. Diagrama de Casos de Uso del primer MVP

Las horas de dedicación a las tareas han sido las siguientes:

Nombre de la tarea	Horas estimadas	Horas reales
Sistema de registro/inicio de sesión	20h	15h
Perfiles de usuarios	18h	12h
Listado de productores	15h	10h
Sistema de productos	32h	28h
Búsqueda de productos	15h	10h
Subida de imágenes	5h	9,5h
Sistema de carrito de la compra	15h	13h
Total	**120h**	**97,5h**

Tras finalizar el desarrollo y despliegue los resultados obtenidos fueron los siguientes:

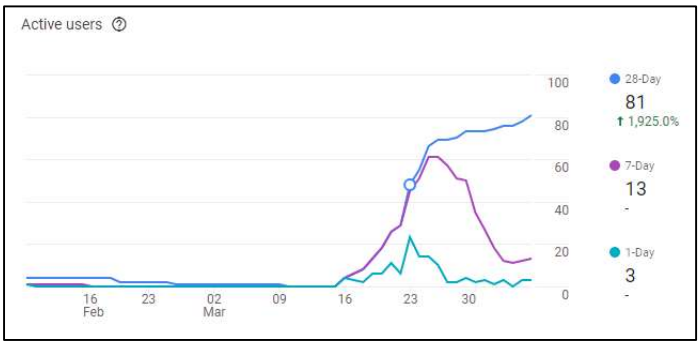


Figura 13. Resultados primer experimento

Se observó que hasta 81 usuarios distintos accedieron a visitar la página web de Agrari, de los cuales se registraron 51 cuentas de usuario que produjeron un total de 20 pedidos a los productores de la aplicación.

3.2. Desarrollo del incremento del MVP

Tras lanzar el experimento para el primer MVP se reajustó el backlog y se añadieron una serie de Unidades de Trabajo de arreglo a partir de las incidencias surgidas, así como algunas mejores sobre las funcionalidades ya existentes. Por parte del equipo también se decidió crear nuevas funcionalidades que aumentasen la propuesta de valor y por lo tanto diesen a Agrari una mayor diferenciación frente al resto de competidores del mercado. En esta parte Aitana Castellanos se encargó de la parte de mejoras en la interfaz de usuario ya existente, mientras que la parte del desarrollo ha sido realizada íntegramente como parte de este TFG, por Arturo Sánchez, autor del mismo.

Tras evaluar la importancia y teniendo en cuenta las restricciones de tiempo del equipo, el incremento quedo de la siguiente manera:

- **Mejora de diseños:** se consigue una especificación más clara de las diversas vistas de la aplicación y posibles cambios a los flujos existentes. Estos se aplicarán para ofrecer una mejor experiencia de usuario y ayudar a incluir el máximo número de personas dentro de la aplicación.
- **Hora de pedidos límite:** por parte de los productores llegó una queja reiterada a cerca de la posibilidad pedir productos muy cerca de la fecha de entrega de estos en el mercado y que suponía que no tenían el tiempo necesario para prepararlos y tenían que cancelar algunos pedidos. Esta hora de pedidos limite será mostrada en el propio producto y en el perfil del productor.
- **Componente de vista de productor:** se creará un componente modular que, a partir del número de identificación de un productor, cargue una vista resumen con los datos esenciales (foto y nombre) y un acceso directo a su perfil, para que así las personas interesadas en un tipo de producto puedan ver más productos similares de este.
- **Arreglo del sistema de inicio de sesión:** a la hora de cerrar sesión la app no actualiza correctamente y se muestra la barra inferior que permite la navegación dentro de la app, ante esto se decide arreglar este comportamiento y evitar la lectura de archivos tan solo para los usuarios registrados y así dar más seguridad.
- **Sistema de recogidas en mercados:** durante el experimento las fechas de recogida del mercado debían ser cambiadas semanalmente de manera manual, por lo que se crea un sistema que, a partir de un día de la semana, una hora de inicio, otra de fin y los días en los que el mercado no está disponible, calcula la fecha correcta y muestra los datos pertinentes. Esto facilita la tarea de actualización de fechas o cancelación de ellas como sucedió varias semanas durante el experimento a causa de las lluvias. Este sistema de recogida además mostrara al usuario la ubicación exacta del mercado para que aquella persona que no conozca sobre la iniciativa del *Mercat Agroecològic* de la UPV pueda acceder a este con facilidad
- **Opción de recuperar contraseña:** varios usuarios perdieron su contraseña y tuvieron que contactar con nosotros para iniciar el proceso de restablecimiento de la contraseña, este proceso ha sido automatizado y tan solo debe introducir su email el usuario que le enviara un email desde el cual restablecer la contraseña.

- **Modificaciones de perfil:** se permitirá al usuario modificar sus datos de perfil, como foto y nombre, y además se permitirá añadir una descripción para indicar los valores e iniciativa o historia de cada uno de los productores que se encuentran dentro de la *app*.
- **Mejoras de accesibilidad:** a raíz de una charla con algunos de los participantes del mercado se llega a la conclusión que parte del *target* de la aplicación puede verse excluido a causa de tamaños de fuente muy pequeños o imágenes con tamaño muy reducido. Es por esto por lo que se decide crear una funcionalidad que al pulsar sobre cualquier imagen esta aparezca como un modal a tamaño completo, y además se aumenta el tamaño de las fuentes en varias partes críticas. También se añadirán funcionalidades como un sistema de avisos dentro de la *app* que informen al usuario del resultado de las operaciones de la base de datos o la localización de las fechas que se muestran en las vistas, que se muestran en formato estadounidense y no en formato europeo y ha generado varias confusiones.
- **Valoraciones de productores:** esta funcionalidad se aborda para dar mayor confianza a los usuarios de que los productos que compran son de confianza, estas valoraciones serán mostradas a manera resumen en el perfil del productor y podrán ser introducidas sobre cada pedido (tan solo una vez por pedido realizado).
- **Asignación de fecha de recogida a pedidos:** los pedidos generados no tienen una fecha de mercado en el que se deben recoger, y esto puede causar problemas si se realizan pedidos el mismo día del mercado o pasada la fecha límite de pedidos, se programa un sistema que a partir de la fecha de creación del pedido, y del sistema de recogida en mercados, genera una fecha de recogida individual para cada pedido de manera dinámica, así si existen varios pedidos sobre el mismo mercado podremos saber cuáles deben ser recogidos en que fechas.
- **Chat entre usuarios:** se añade a la propuesta de valor un chat directo entre productor y cliente para así facilitar la compra y posible resolución de dudas que puedan surgir a los clientes. Este chat tendrá como objetivo la seguridad de la información por lo que se cifraran estos posibles mensajes sensibles para evitar problemas.
- **Seguridad de la base de datos:** la mayoría de los ficheros no están protegidos de ninguna manera y las únicas reglas de seguridad protegiendo los archivos son las puestas en el portal, pero estas pueden ser saltadas por un usuario experto con malas intenciones. Es por esto por lo que se crean un conjunto de reglas de Firebase que reflejen las restricciones programadas en la lógica de negocio del portal.
- **Arreglar la pila de vistas perfil y productos:** al realizar una carga de una vista, los datos se quedan almacenados en cache y al volver a acceder a esta misma vista con otros datos de entrada estos no se refrescan y dan lugar a confusión. Se fuerza la carga de cada vista al acceder a ella.

El diagrama de casos de uso resultante de estas unidades de trabajo es el siguiente:

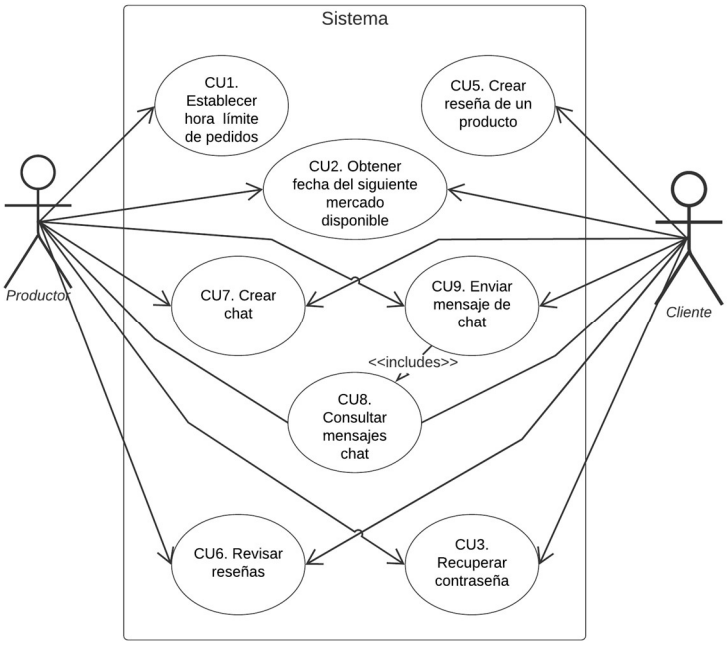


Figura 14. Diagrama de casos de uso del incremento del segundo MVP

Las horas de dedicación a este incremento resultaron ser mucho mayores debido a las nuevas unidades de trabajo de arreglos. En vez de acortar el sprint y reducir tareas, se decidió alargar el plazo de entrega y así no excluir ninguna funcionalidad clave.

La dedicación final fue la siguiente:

Nombre de la tarea	Horas estimadas	Horas reales
Mejora de diseños	15h	13h
Hora de pedidos limite	5h	12h
Componente de vista del productor	10h	10h
Arreglo del sistema de inicio de sesión	6h	6h
Sistema de recogidas en mercados	12h	12,5h
Opción de recuperar contraseña	8h	2h
Modificaciones del perfil	7h	7h
Mejoras de accesibilidad	10h	6h
Valoraciones de productores	30h	32,5h
Asignación de fecha de recogida a pedidos	15h	14,5h
Chat entre usuarios	25h	13h
Seguridad de la base de datos	2h	2h
Arreglar pila de vistas perfil y productos	5h	2h
Total	**145h**	**132,5h**

Tras finalizar el desarrollo y despliegue los resultados obtenidos son los siguientes:

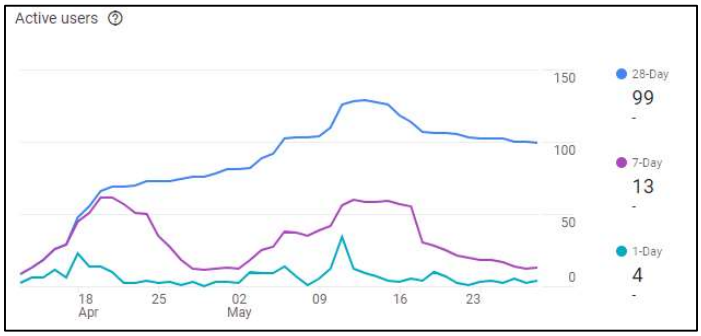


Figura 15. Resultados segundo experimento

Al finalizar esta última prueba el volumen de visitas aumentó considerablemente con un total de 129 usuarios únicos activos en la aplicación, un total de 68 cuentas de usuarios generadas y un total de 43 pedidos realizados a lo largo de 2 semanas del *Mercat Agroecològic*.

4. Aspectos técnicos

En este apartado se muestra la selección de las diversas herramientas de desarrollo utilizadas a lo largo del proyecto, una explicación de la herramienta y el papel que cada una tomó en el proyecto.

El proyecto nació con la idea de ser una aplicación móvil disponible tanto para Android como para iOS, este requerimiento implicaba hacer un desarrollo nativo (como seria utilizando Android Studio o XCode para sus respectivas plataformas) era inviable debido a las limitaciones de tiempo del proyecto. Es por esto por lo que se decidió hacer un desarrollo que pueda cubrir ambas plataformas en un mismo desarrollo con adaptaciones mínimas para cada una.

Habiendo decidido el tipo de producto a desarrollar se comenzó una fase de investigación sobre las diversas tecnologías existentes. Aparecieron varias opciones como pueden ser Flutter o React, pero finalmente se eligió optar por utilizar Angular por la familiaridad del equipo de trabajo con esta herramienta. Utilizar Angular como *framework* base permitió realizar un desarrollo web ágil y sencillo, que junto al resto de herramientas utilizadas (Ionic y Capacitor) ha permitido crear una única aplicación multiplataforma que ahorrase un importante tiempo de desarrollo.

4.1. Herramientas utilizadas

Figura 16. Logo Angular

Angular

Angular es un *framework* para desarrollo de páginas web progresivas utilizando Typescript, HTML y CSS. Esta herramienta proporciona a sus desarrolladores una base amplia de funcionalidades reutilizables para evitar tener que instalar librerías externas o hacer desarrollos propios que consuman grandes cantidades de tiempo.

El motivo por el cual se optó por utilizar Angular en el desarrollo fue principalmente para reducir significativamente las horas de aprendizaje necesarias para este trabajo para poder dedicarlas al desarrollo y así aumentar el alcance del mismo.

Figura 17. Logo Ionic

Ionic

Ionic es un framework de desarrollo orientado en el desarrollo de aplicaciones móviles mediante una amplia colección de componentes para crear una interfaz móvil sin la necesidad de escribir mucho código.

Para el proyecto se decidió utilizar Ionic para proporcionar una apariencia y una experiencia de usuario lo mas similar posible a una aplicación nativa. Aunque muchos de los componentes estaban ya implementados, estos fueron modificados para seguir las guías de estilo del proyecto.

Figura 18. Logo Capacitor

Capacitor

Capacitor es una herramienta que facilita la creación de aplicaciones híbridas cuya base de código está basada en tecnologías web y que se encuentran encapsuladas en una aplicación nativa y se comunican entre si a través de una serie de APIs que dan acceso completo a las funciones nativas del teléfono como podría ser la geolocalización o giroscopio, entre otros.

Utilizar Capacitor permitió tener una base única de código tanto para web como para ambas aplicaciones (Android e iOS) , con la posibilidad de acceder a todas las funcionalidades nativas necesarias para el correcto funcionamiento de esta como lo fueron el acceso a la memoria interna del teléfono y la cámara.

Figura 19. Logo Firebase

Firebase

Para ahorrar tiempo de desarrollo se buscó una solución completa de *backend* para la aplicación que permitiese dedicar todo el tiempo de desarrollo a la aplicación y así obtener mejores resultados. Firebase permite tener en una misma plataforma gratuita la gestión de Google Analytics de la web, una base de datos no relacional utilizando su tecnología de Firebase Storage y además facilita el almacenamiento de las distintas imágenes de los productos o usuarios que se suban a través de la aplicación.

Figura 20. Logo Android Studio

Android Studio

Android Studio es la herramienta de desarrollo por excelencia para Android, esta permite desarrollar proyectos nativos en Java o Kotlin y compilarlos para ser utilizados en dispositivos Android. El desarrollo que se llevó a cabo en Android Studio es mínimo gracias al uso de Capacitor puesto que solo fue necesario gestionar los permisos de los dispositivos necesarios para el funcionamiento y que se le piden al usuario. Una vez configurados estos permisos (como pueden ser la cámara y localización entre otros) tan solo fue necesario compilar el proyecto para generar el archivo instalable.

Figura 21. Logo Github

Github y Gitflow

Para el versionado y mantenimiento de la web se usó la plataforma de gestión de versiones Github que permitió guardar en la nube todos los cambios que se realizaron al código durante el proyecto y mantener un historial de los cambios que se fueron produciendo.

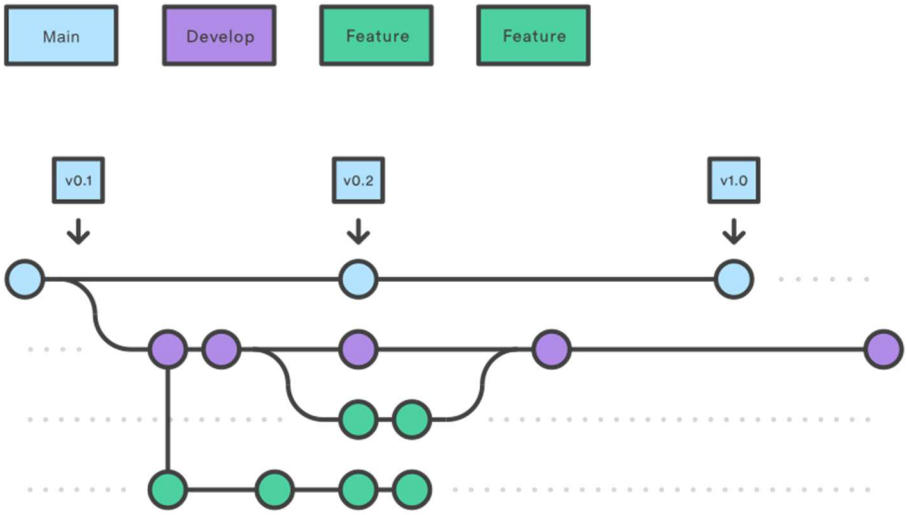


Figura 22. Diagrama Gitflow

Para el gestionado de versiones también se siguió una técnica llamada Gitflow que se basa en desarrollar cada una de las distintas funcionalidades de la app por separado en su propia rama sin que el estado de desarrollo de cualquiera de estas pudiera afectar al resto.

Para esto se crearon 2 ramas fijas, la rama *Main* donde se guardaban los cambios subidos a producción, con todas las características integradas ya validadas en la fase de pruebas, y la rama *Develop* donde se encontraban aquellas características que aún no habían sido comprobadas para poder validar el producto en conjunto. Por último, existe otro tipo de ramas, llamadas *Feature* donde se desarrollaron las características individuales independientemente del resto de desarrollos paralelos para evitar fallos ajenos al desarrollo actual.

Una vez se terminó de desarrollar una *Feature*, se integraron los cambios realizados en la rama *Develop* donde, si recibían el visto bueno, esperaban hasta la fecha de lanzamiento de la nueva versión, donde finalmente se integraban en la rama *Main* para desplegarlo en producción.

Figura 23. Logo Figma

Figma

Para desarrollar la primera propuesta de interfaz de usuario se usó la herramienta de diseño y mockups Figma, esta herramienta además permitía simular la navegación entre las distintas vistas de la aplicación y así poder validar previamente entre todos los miembros del equipo la arquitectura de la información escogida al principio del desarrollo.

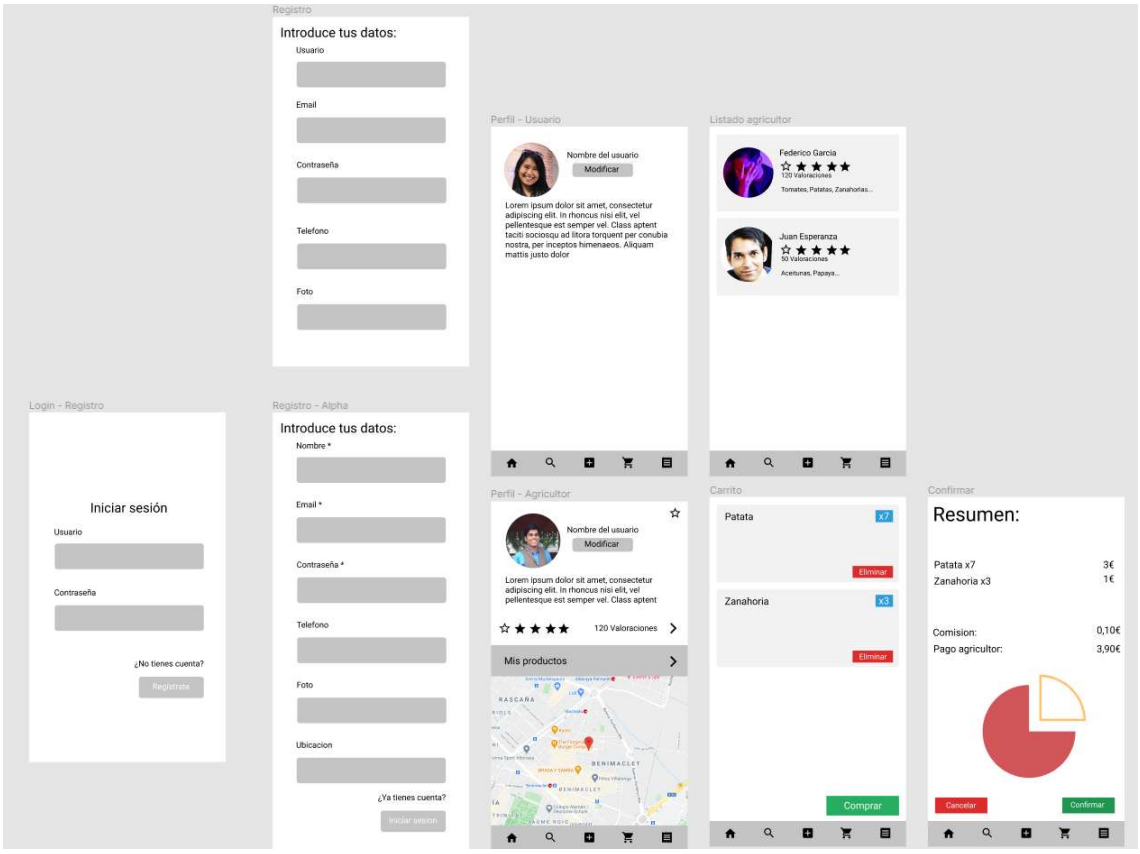


Figura 24. Wireframe de baja fidelidad

Figura 25. Logo Trello

Trello

Trello es una herramienta de Atalassian para la creación de tableros Kanban. Dado que el proyecto se realizó siguiendo un proceso por fases, esta herramienta se usó para hacer el seguimiento del proyecto a lo largo de todas sus etapas y así tener reflejado en un mismo sitio el estado global del proyecto para facilitar su seguimiento.

4.2. Entornos de desarrollo

El proyecto nació con un solo entorno de producción, pero esto cambió rápidamente debido a la necesidad de crear nuevas funcionalidades sin afectar a la experiencia de los usuarios que la utilizaban durante el mercado. Es por esto por lo que se crearon dos entornos de desarrollo principales, preproducción y producción.

Ambos entornos de desarrollo se encuentran alojados en una maquina con Plesk alojada en un servidor de DigitalOcean, cada entorno se encuentra en un dominio distinto, en el caso de la aplicación se sitúa en app.agrari.es. También se crearon dos proyectos separados de Firebase para así poder hacer cambios a los esquemas de datos y realizar pruebas de pedidos sin afectar a los productores y clientes que estaban participando en los experimentos.

Para compilar el proyecto en cada uno de los entornos se generaron dos comandos de NodeJS, que modificasen el archivo de configuración que contiene las variables de entorno para preproducción y producción. Esto hacía muy sencillo compilar el proyecto y realizar pruebas de forma ágil en ambos entornos.

Para el despliegue se buscó seguir un proceso de CI/CD, por lo que se preparó un repositorio de GitHub con dos ramas, una para cada entorno de desarrollo. Estas dos ramas están conectadas de forma automática mediante *webhooks* al servidor, el cual se encarga de descargar de manera automática la nueva versión y desplegarla.

4.3. Modelo de datos

Para el desarrollo del proyecto se estableció un proceso de especificación previo del funcionamiento y relación de las diversas clases que existen. Este modelo de datos estaba elaborado siguiendo la especificación UML y permitía ver de manera sencilla como estaban relacionadas las distintas clases para así facilitar las tareas de implementación y el esquema de almacenamiento de datos.

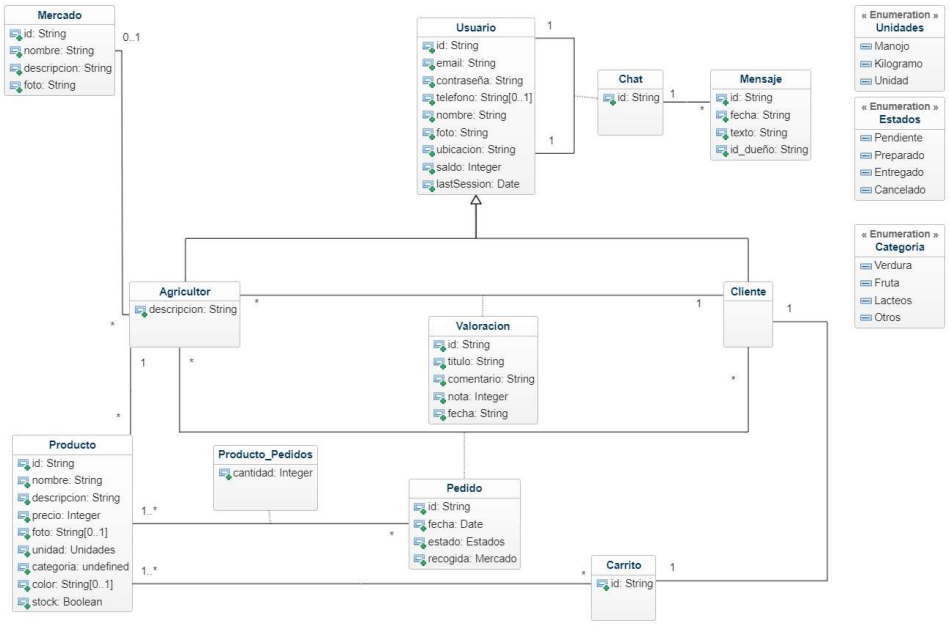


Figura 26. Modelo de Datos

Este conjunto de clases se tradujo en las siguientes colecciones de almacenamiento de datos:

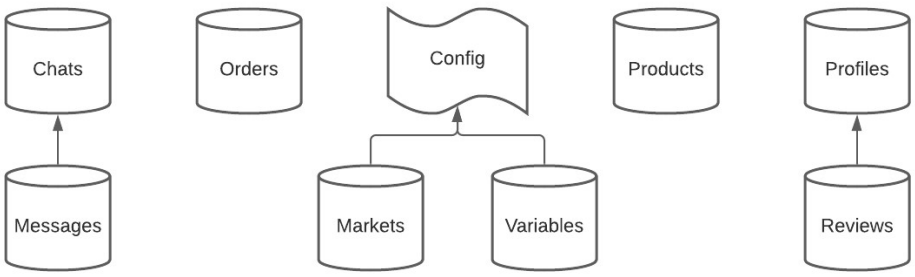


Figura 27. Resumen tablas base de datos

4.4. Arquitectura del proyecto

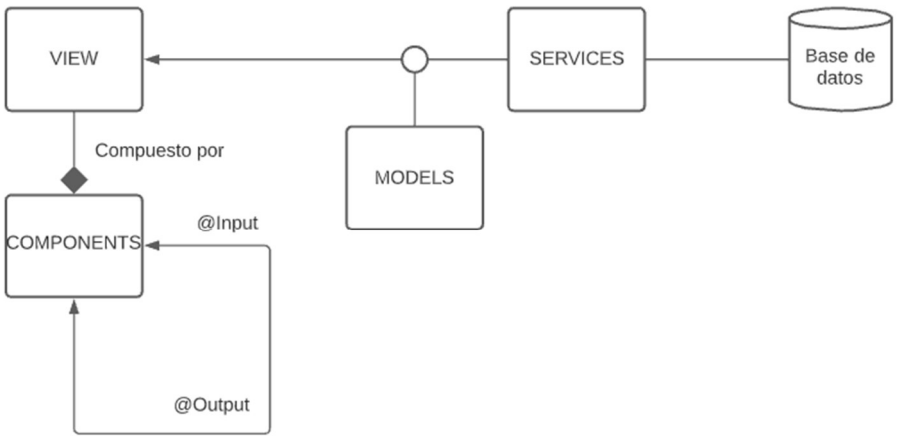


Figura 28. Diagrama de la arquitectura del proyecto

Para la arquitectura del proyecto se optó por crear una separación de responsabilidades, donde las vistas tan solo se encargaban de recuperar los datos de base de datos (en función de una serie de modelos definidos), los cuales se alimentaban a los componentes. En estos componentes se establecieron unos canales de comunicación directa para procesar los datos y notificar de los cambios producidos.

El proyecto desarrollado en Angular buscaba una alta modularidad que posibilitase una alta reutilización del código y de las vistas implementadas, dado que a lo largo del proyecto se observaron varias funcionalidades similares que podían ser reutilizadas en múltiples partes de la aplicación.

Para obtener este modularidad se optó por una arquitectura basada en componentes. Esta arquitectura buscaba crear una serie de vistas, encargadas de obtener los datos a presentar, estas vistas estarían compuestas por componentes, que son los elementos reutilizables intercomunicados con la vista y otros componentes.

Esta comunicación se realizó a través del patrón *@Input* y *@Output* que proporciona Angular, que permite compartir valores obtenidos en un componente padre con los distintos hijos (mediante la directiva *@Input*) y avisar de cambios realizados sobre estos datos en el componente hijo (a través de la directiva *@Output*).

Figura 29. Directivas Angular

Esta funcionalidad posibilitó obtener en una sola llamada toda la información necesaria como los datos del perfil de usuario, y repartirlos entre los distintos componentes que trataban los datos.

Un ejemplo de estos componentes reutilizables es el siguiente:

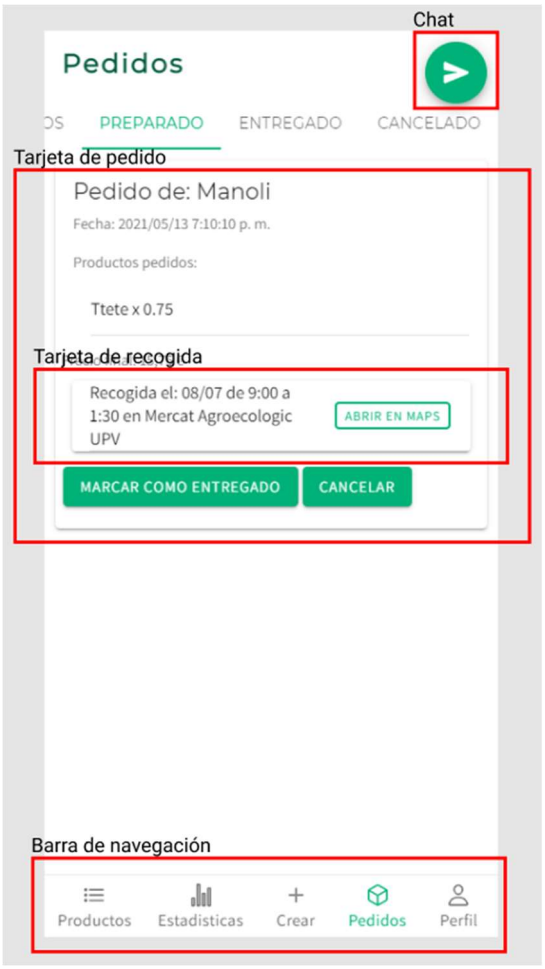


Figura 30. Ejemplo de componentes

El intercambio de datos con el servidor se realizó a través de los servicios de Angular. Estos servicios son una serie de clases que siguen el patrón de diseño *Singleton* a través del cual se obtenían, borraban, creaban y modificaban los datos en función a la lógica de negocio programada en los servicios. El patrón *Singleton* (instancia única) permitió tener en todo momento un punto en común donde consultar y almacenar datos, esto es especialmente útil a la hora de generar pedidos, puesto que permite almacenar en todo momento los productos que se han añadido al carrito sin necesidad de realizar ningún almacenamiento de datos local o en base de datos.

4.5. Desafíos de programación

A la hora de desarrollar el proyecto han surgido diversos problemas y complicaciones, tanto previstos como no previstos, que han debido ser resueltos con un esfuerzo mayor al esperado.

4.5.1. Cálculo de fechas dinámicas

Uno de los primeros retos fue el cálculo de fechas de entrega de los pedidos y de las fechas límite para realizar estos. Para realizar este cálculo, y puesto que las fechas y lugares de los mercados podían variar, se crearon en la base de datos una serie de parámetros que permitían modificar cualquier parámetro según fuese necesario.

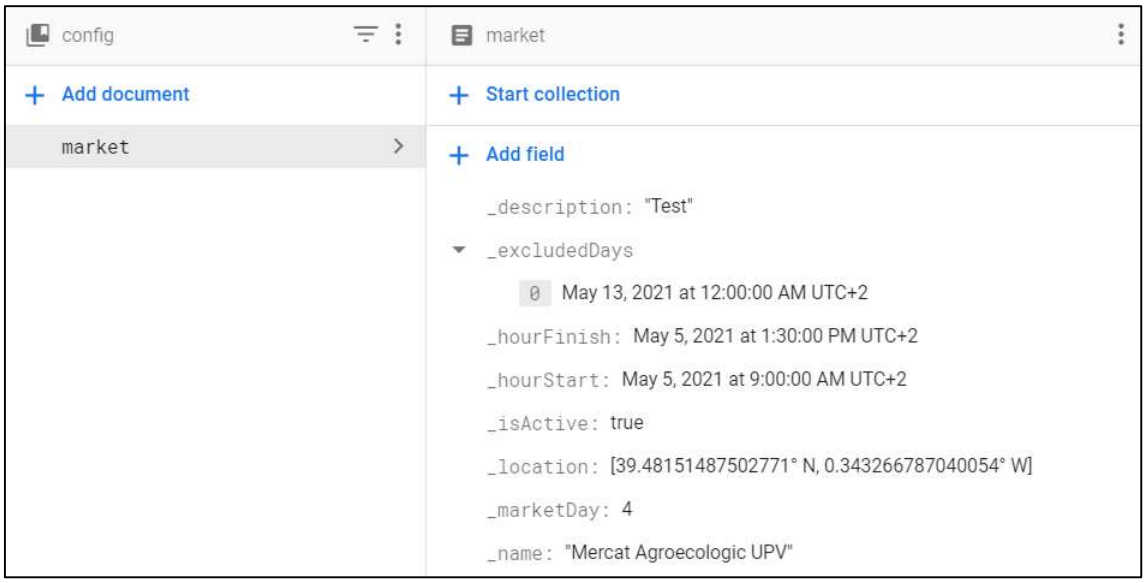


Figura 31. Datos del mercado

Como se ve en la figura 28, los parámetros son los siguientes:

- location: Coordenadas de donde se realiza el mercado que se utilizan para mostrar la dirección en un mapa.
- hourStar/ hourFinish: Se usan para indicar de que hora a qué hora se realiza el mercado.
- marketDay: Qué día de la semana se realiza el mercado, en este caso es el cuarto día de la semana.
- name: Nombre del mercado para facilitar su identificación.
- description: Una breve descripción para obtener información extra del mercado.
- isActive: Si el mercado está operativo actualmente o se encuentra cerrado.
- excludedDays: Una lista de todos los días en los cuales no se va a realizar el mercado.

Finalmente, para obtener la fecha del pedido o del próximo mercado se usó un algoritmo simple que obtuviese los datos necesarios correspondientes al mercado deseado. Con esos datos, se calculaba la próxima fecha disponible a partir de la fecha actual, y en caso de que esta fecha estuviese incluida en el listado de fechas excluidas se volvía a realizar este mismo calculo a partir de esa fecha no disponible.

```
this.marketService.getCurrentMarket().then(value => {
  this.marketInfo = value;

  let today = new Date();
  let nextMarketDay = nextDay(today, this.marketInfo.marketDay);
  let invalidDays = this.marketInfo.excludedDays.filter( predicate: (day : Date ) => {
    return isSameDay(day, nextMarketDay);
  });
  while (invalidDays.length > 0) {
    today = add(today,  duration: {weeks: 1});
    nextMarketDay = nextDay(today, this.marketInfo.marketDay);
    invalidDays = this.marketInfo.excludedDays.filter( predicate: (day : Date ) => {
      return isSameDay(day, nextMarketDay);
    });
  }
  this.nextAvailableDay = nextMarketDay;
});
```

Figura 32. Código algoritmo de fechas

4.5.2. Conversión Entidad-Modelo

Otro problema que surgió durante el desarrollo del proyecto fue la obtención de datos por parte del portal web. Cuando se consultaba la base de datos y se recibían las diferentes entradas (o entidades) se debían convertir a un objeto utilizable en la aplicación (modelo), para esto se introducían una serie de operaciones que realizaban esta conversión según las necesidades. Esta conversión solía ser básica (adaptar los datos recibidos a la clase deseada), pero en algunas ocasiones se necesitaba añadir algo de lógica extra como podría ser la conversión de tipos, o la modificación de un objeto anidado por una referencia a ese objeto.

En estas transformaciones detectamos dos tipos principales, la conversión de imágenes a rutas que las referencien, y la conversión del identificador de una entrada a su objeto correspondiente.

```
_category: 7
_description: "110 gr"
_name: "Paté Alberginia / Tahin negre / Comí"
_ownerId: "K3C1pddzHUdrUNChyRryhl6Ejfa2"
_picture: "/products/S6tabTJkj9Ojfh0gXBCGr3wxrgA3/0b
_price: 3.5
_stock: true
_unit: 2
```

Figura 33. Datos de un producto

Para la conversión de imágenes se usó la dirección interna del archivo dentro del almacenamiento de Firebase. Esta ruta, por temas de seguridad, no podía ser convertida directamente a una dirección al archivo, por lo que se debía tratar y convertir de manera opaca al usuario.

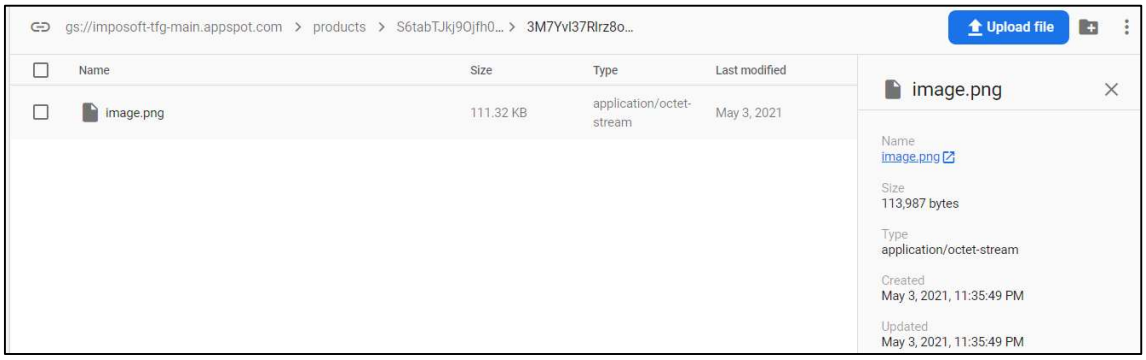


Figura 34. Imagen almacenada en Firebase

Es por esto que en los procesos de carga se ha añadido una función de conversión que transforme esa ruta a una dirección url que dirija al archivo correspondiente.

```
loadUserById(userId: string){
  return new Promise<any>( executor: (resolve, reject) => {
    this.userCollection.doc(userId).valueChanges( options: {idField: '_id'}).subscribe( next: userEntity => {
      const userModel = new User().convertFromDTO(userEntity);
      this.loadProfileImage(userModel.picture).then(imageUrl => userModel.picture = imageUrl);
      resolve(userModel);
    });
  });
}
```

Figura 35. Ejemplo de servicio cargando de Firestore

```
async loadProfileImage(imagePath: string){
  const ref = this.storage.ref(imagePath);
  return new Promise<any>( executor: (resolve, reject) => {
    ref.getDownloadURL().subscribe( next: url => {
      resolve(url);
    });
  });
}
```

Figura 36. Funcion de carga de imagen a partir de ruta

En el caso contrario, se realizó una operación en dos pasos que posibilitara subir el archivo al almacenamiento de Firebase y guardar la referencia a este en la entrada de la base de datos. Para ello primero se convirtió la imagen seleccionada por el usuario a formato base64, y una vez se había almacenado el archivo, se devolvía la ruta en la cual se había almacenado el archivo para recuperarlo posteriormente.

```
uploadPictureToProfile(picture: any, userID: string): string {
  const file = picture;
  const filePath = '/userProfilePics/' + userID + '/' + 'myPicture.png';
  const ref = this.storage.ref(filePath);
  ref.putString(file,  format: 'base64');
  return filePath;
}
```

Figura 37. Código de subida de imágenes

Otro tipo de conversión realizada fue la obtención de objetos anidados gracias a su identificador y viceversa. En la figura 30 se muestra un identificador de otro documento bajo la variable *ownerId*, esta hace referencia al usuario que se muestra a continuación.

Figura 38. Datos perfil de usuario

A la hora de trabajar desde la aplicación era conveniente tener todos los datos necesarios precargados y evitar de esta forma que el usuario tuviera que esperar a varias consultas o tener que implementar el mismo código en diversos puntos del código.

Es por esto por lo que se crearon una serie de funciones en los servicios que manejasen esta conversión modelo-entidad y así evitar datos duplicados en base de datos. Un ejemplo claro de esta problemática sucedía en los pedidos, donde se tiene una lista de productos que se debían mostrar con su nombre y foto. Cuando se obtenían los datos, se debía cargar también los datos de los productos para facilitar las tareas de desarrollo, y a la hora de almacenar estos datos se debía evitar subir estos datos y causar duplicidad de datos.

Las funciones de obtención de datos son simples, en la misma carga del objeto se incluyó una lógica extra que recupera de la base de datos las entidades a través de su identificador, y las funciones que persistían los objetos en base de datos se limitaban a eliminar estos objetos para asegurar que nunca se almacenaran.

```
createOrderEntity(order: Order): any {
  const orderEntity = classToPlain(order);
  orderEntity._orderedItems.forEach(product => {
    delete product._product;
  });
  return orderEntity;
}
```

Figura 39. Conversión modelo-entidad

```
loadAllOrders(){
  let orderList: Order[] = [];
  return new Promise<any>( executor: (resolve, reject) => {
    this.orderCollection.valueChanges({idField: '_id'}).subscribe( next: orderEntities => {
      orderList = [];
      orderEntities.forEach((orderEntity) => {
        const orderModel = new Order().convertFromDTO(orderEntity);
        orderModel.orderedItems.forEach((product, index) => {
          this.productService.loadProductById(product.productId).then(value => {
            orderModel.orderedItems[index].product = value;
          });
        });
        orderList.push(orderModel);
      });
      resolve(orderList);
    });
  });
}
```

Figura 40. Conversión entidad-modelo

4.5.3. Sistema de mensajería instantánea

El último gran reto que se consiguió superar es el del servicio de mensajería instantánea, este servicio genera una clase asociación entre los dos usuarios que han creado la conversación, y dentro de esta clase se almacenan los mensajes que han creado con los metadatos necesarios para su tratamiento.

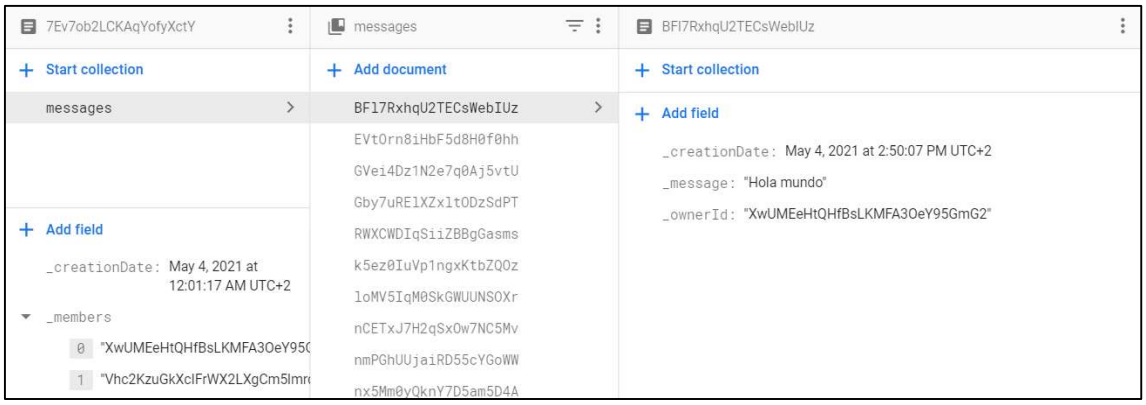


Figura 41. Datos de una conversación

En el establecimiento de una conversación entre dos usuarios se pueden observar dos casuística distintas según si estos usuarios han establecido con anterioridad una conversación, en caso de que esto fuese cierto, se procede a la creación de una nueva entrada en la colección de

chats con el identificador de ambos usuarios y una lista donde se almacenan los mensajes intercambiados, y en el caso de que esta conversación ya exista, se redirige al usuario a la ventana de *chat* correspondiente.

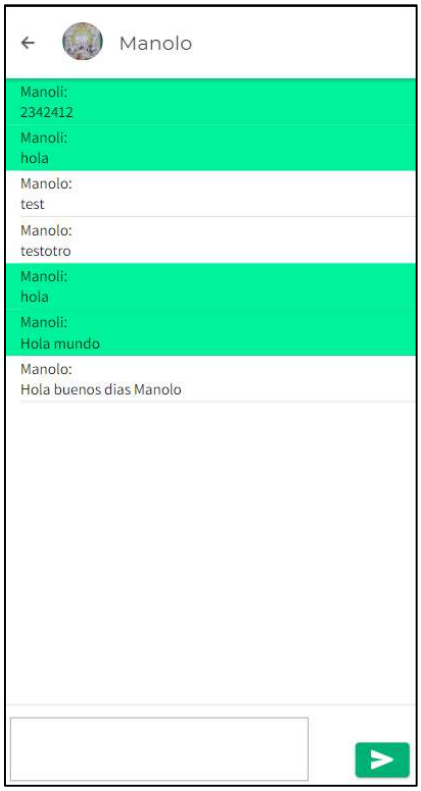


Figura 42. Ejemplo de un chat entre usuarios

Para realizar el sistema de mensajería instantánea fue de vital importancia la comunicación en tiempo real con el servidor para recuperar los mensajes compartidos por los dos usuarios lo antes posible. Esto se consiguió utilizando el patrón de diseño Observador que proporciona la *API* de Firebase, este observador se ha encargado de actualizar los datos del chat cada vez que se generaba un mensaje nuevo por parte de cualquiera de los dos usuarios.

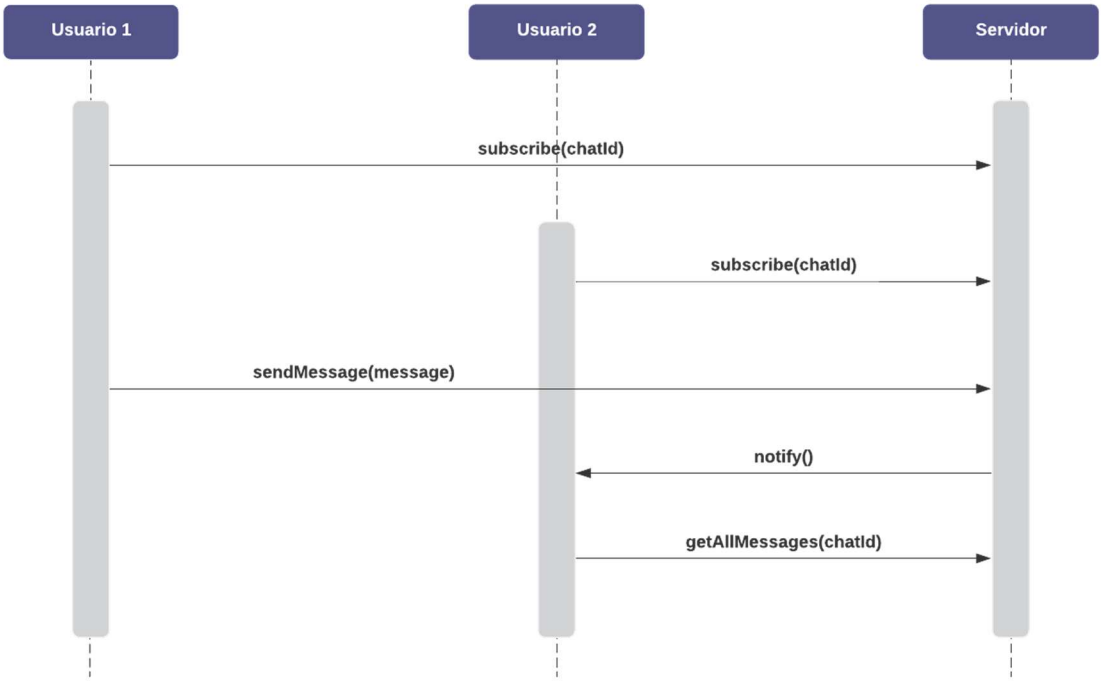


Figura 43. Diagrama de secuencia del patrón observador

Este intercambio de información resulta en el siguiente diagrama de secuencia, el cual permite entender más detalladamente esta comunicación entre el sistema y los dos usuarios. Al establecer una conversación cada usuario se suscribía a la colección en la que se encuentra este chat. Una vez creada esta suscripción, cada vez que se realizaba una operación que modificase o añadiese datos activaría una llamada que notificase al usuario que debía volver a obtener los datos.

Finalmente, y para mejorar la experiencia de usuario, se realizó una lógica adicional que permitiese ordenar los mensajes en función del metadato referente a la fecha de creación del mensaje, y otra que permitía distinguir el creador de cada mensaje filtrando aquellos que tuviesen el mismo id que el usuario actual y añadiendo un booleano que representase la autoría de ese mensaje para evitar realizar una comprobación por cada mensaje del chat y facilitar la distinción a nivel visual de quien compartía cada mensajes.

4.5.4. Sistema de inicio de sesión y ciclo de vida de las actividades

A lo largo de la primera fase de desarrollo del MVP fue reportada una incidencia con el sistema de inicio de sesión, los usuarios indicaban que tras cerrar sesión la barra de navegación aún estaba presente en la barra inferior y podían realizar las acciones habituales sin una cuenta de usuario. Tras realizar una sesión de depuración, se encontró la incidencia en el estado inconsistente de la barra de navegación frente al resto de la aplicación que causaba que la barra de navegación no fuese notificada del evento de cierre de sesión.

Para solucionar este problema se optó por mover el evento de cierre de sesión de la vista del perfil de usuario a un nuevo servicio de autenticación que fue el encargado de mantener el estado del inicio de sesión (si existe una sesión activa) y todos los datos del usuario actual que después serian consultados por el resto de componentes.

```
constructor(public afAuth: AngularFireAuth, public router: Router, public firestore: AngularFirestore) {
  this.afAuth.authState.subscribe( next: user => {
    if (user) {
      this.userId = user.uid;
      this.isLogged = true;
      this.userCollection = firestore.doc( path: 'profiles/' + user.uid);
      this.userCollection.valueChanges().subscribe( next: userData => {
        const realUser = new User().convertFromDTO(userData);
        this.isFarmer = realUser.role === UserRoles.PRODUCER;
      });
    } else {
      this.isLogged = false;
    }
  });
}

logout(){
  return new Promise<any>( executor: (resolve, reject) => {
    this.afAuth.signOut().then(() => {
      this.clearAllUserData();
      window.location.reload();
      resolve( value: true);
    });
  });
}
```

Figura 44. Codigo servicio AuthService

Como se puede observar en la figura 43, el código del nuevo servicio se encargaba de la obtención de los datos del usuario tras el inicio de sesión. Una vez se lanzase el evento de cierre de sesión desde cualquier parte de la aplicación, estos campos eran vaciados y se procedía a reiniciar todas las actividades activas para asegurar un estado consistente a través de toda la aplicación.

4.5.5. Ciclo de vida de las actividades

Otro problema a la hora del desarrollo surgió a raíz del ciclo de vida de las páginas que implementa Ionic que causaba inconsistencias en la interfaz de usuario mostrando los mismos datos cuando abandonabas una página y volvías a entrar a esta. Este problema causaba que los clientes al consultar dos productos, uno tras el otro, tan solo pudiesen ver la información del primero, para los productores este problema se causaba en la creación de productos, que resultaba en datos duplicados o vacíos.

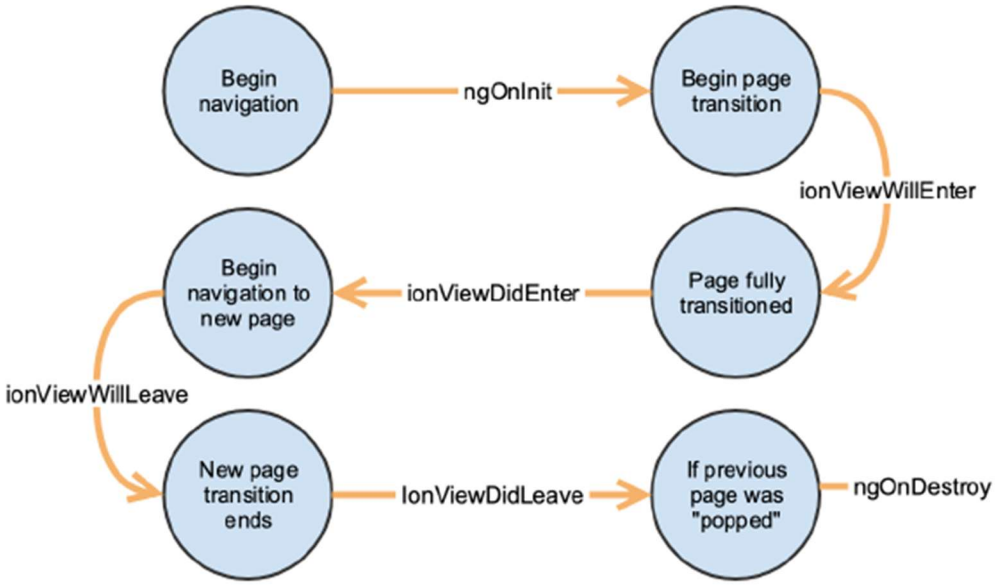


Figura 45. Ciclo de vida de Ionic

El ciclo de vida que ofrece Ionic se puede ver representado en la figura 44, y nos muestra como este se integra con el ciclo de vida de Angular. Los estados son representados por círculos mientras que los eventos que se producen entre estos estados son representados por flechas.

El problema surgía en la carga de los datos, una vez se cargaban los datos de una de las paginas, estos quedaban en memoria hasta que se forzase una actualización puesto que los datos eran cargados en el evento ngOnInit, que se produce cuando se inicializa cada página, pero nunca se producía el evento ngOnDestroy que se asegurase de que se eliminasen de la memoria las paginas una vez el usuario las abandonaba.

```
constructor(private router: Router, private route: ActivatedRoute, public enumToArrayPipe: EnumToArrayPipe,
            public firestore: AngularFirestore, public afAuth: AngularFireAuth,
            public storage: AngularFireStorage, public cartService: CartServiceService,
            public productService: ProductService, public profileService: ProfilesService,
            public alertController: AlertController) {
  this.product = new Product();
  this.productOrder = new ProductOrder();
  this.productOwner = new User();
  this.unitsList = this.enumToArrayPipe.transform(UnitTypeEnum);
}

ionViewWillEnter(){
  this.route.params.subscribe( next: params => {
    if (params.id) {
      if (this.pathId !== params.id) {
        this.pathId = params.id;
        this.productService.loadProductById(this.pathId).then(value => {...});
      }
    }
  });
}
```

Figura 46. Primera solución del ciclo de vida

Para solucionar este problema se decidió por modificar aquellas páginas que ya habían sido desarrolladas y se optó por mantener la creación de los objetos en el constructor y por mover la lógica de carga de los datos del evento ngOnInit al evento ionViewWillEnter que se produce cuando se renderiza la página y por lo tanto se aseguraría que las paginas tuviesen los datos correctos y aun se permitiese volver a la página anterior si se intentaba navegar atrás.

```
navigateToUrl(url: string){
  this.router.navigate( commands: [url]);
}

navigateToUrlIonic(url: string){
  this.navigationController.navigateRoot(url);
}
```

Figura 47. Código de navegación antes y después

Finalmente, en aquellas paginas que no se desease la navegación hacia atrás (como lo eran las rutas raíz que se encuentran en la barra de navegación) se optó por añadir una lógica extra que asegurase que las paginas no se almacenasen en la cache del dispositivo y por lo tanto ocupase menos espacio en memoria y asegurasen un mejor uso de los recursos por parte de la aplicación. Esto se consiguió utilizando el controlador de navegación de Ionic para navegar de una página a otra en vez de utilizar el navegador de Angular, esto suponía que las paginas ya navegadas fuesen eliminadas de memoria y no se pudiese navegar de vuelta a ellas al pulsar el botón para navegar atrás.

4.6. Pruebas realizadas

Para validar las distintas funcionalidades y el resultado de ambos experimentos, aparte de realizar las pruebas de validación definidas en la fase de especificación, también se decidió realizar una serie de cuestionarios de usabilidad al usuario que posibilitaran identificar deficiencias inesperadas. Para ambas pruebas se han utilizado usuarios reales de la aplicación, en la primera tanto productores como compradores, y en la segunda tan solo clientes.

Figura 48. Foto del equipo de Agrari

4.6.1. Validación del primer incremento del MVP

Para validar los resultados obtenidos tras realizar el primer experimento en el *Mercat Agroecològic* de la UPV, se optó por crear un formulario de usabilidad estándar PSSUQ. Se decidió optar por este tipo de formulario para validar posibles carencias en la experiencia de usuario y organización de la información.

Este formulario de usabilidad se basaba en 16 preguntas con una nota del 1-7 (siendo 1 la mejor nota, y 7 la peor) con la posibilidad de optar por no responder a ciertas preguntas. La estructura de la encuesta fue la siguiente:

1. En general, estoy satisfecho con lo fácil que es utilizar esta app.
2. Fue sencillo utilizar esta app.
3. Pude completar las tareas (subir un pedido, hacer un pedido, navegar por la app) y escenarios rápidamente usando esta app.
4. Me sentí cómodo usando esta app.
5. Fue fácil aprender a utilizar esta app.
6. Creo que utilizar esta app podría ayudarme rápidamente a resolver mis necesidades.
7. La app me mostro mensajes de error que me indicaban claramente cómo solucionar problemas.
8. Siempre que se ha producido un error he podía recuperarme fácil y rápidamente.
9. La información (como ayuda al usuario, mensajes en pantalla y otra documentación) proporcionada con este sistema era clara.

10. Fue fácil encontrar la información que necesitaba.
11. La app me dio suficiente información para ayudarme a completar las tareas y los escenarios.
12. La organización de la información en las pantallas del sistema fue clara.
13. La interfaz de este sistema fue agradable.
14. Me gustó usar la interfaz de este sistema.
15. Esta app tiene todas las funciones y capacidades que esperaba que tuviese.
16. En general, estoy satisfecha/o con este sistema.

La evaluación se podía dividir en 3 subpartes que agrupaban distintas preguntas y nos permitán detectar de manera más especifica qué aspectos del sistema podían estar dando problemas. Estas partes eran:

- Utilidad del sistema (SYSUSE): se calcula con la media de las preguntas 1 a la 6, este valor permite validar como de útil es el sistema para el usuario final.
- Calidad de la información (INFOQUAL): se calcula realizando la media de las preguntas 7 a la 12 lo cual permite validar como de buena es la información que se le muestra al usuario.
- Calidad de la interfaz (INTERQUAL): se calcula mediante la media de las preguntas 13 a la 15 y permite conocer si la interfaz de usuario cumple las necesidades de los usuarios o existen posibles deficiencias a subsanar.

La encuesta fue enviada a todos los participantes a través de un documento de Google Forms, una vez completada la encuesta por todos los usuarios se volcó esa información a Excel para tratarla y así permitir entender mejor los datos obtenidos.

Los resultados de la encuesta fueron los siguientes:

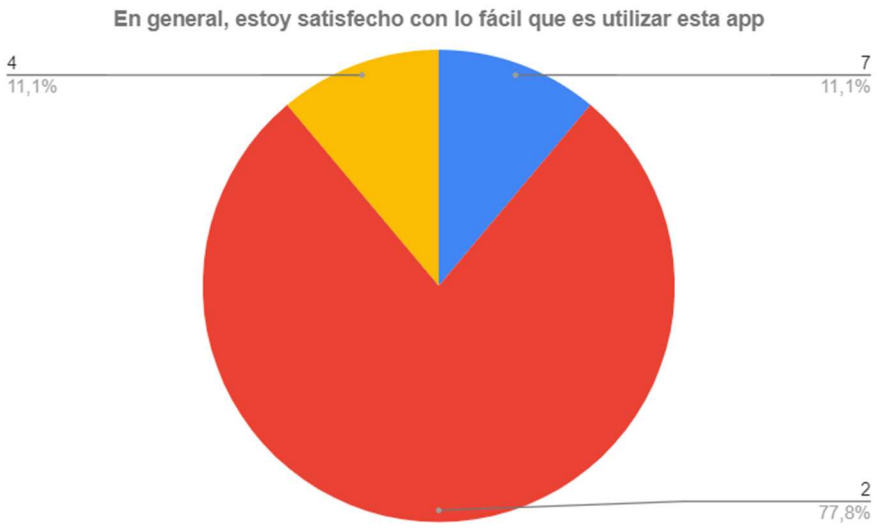


Figura 49. Pregunta 1 del formulario

Gran parte de los usuarios dio una muy buena nota en general a la sencillez de uso de la aplicación, tan solo encontramos un usuario que haya evaluado negativamente la experiencia en la aplicación, y esta persona tiene el rol de persona productora.

Figura 50. Pregunta 2 del formulario

Cuando preguntamos más específicamente por la usabilidad de la aplicación se puede ver que los resultados eran similares, pero algún usuario reportaba haber tenido algunas dificultades con la app.

Figura 51. Pregunta 3 del formulario

Respecto a la dificultad de utilizar la app y hacer pedidos u otras funcionalidades clave se observa que podían existir ciertas deficiencias en la usabilidad puesto que las notas bajan de manera notable.

Figura 52. Pregunta 4 del formulario

Respecto a la comodidad de usar la aplicación se pudieron identificar varias quejas de los usuarios. Al preguntarles de manera individual para que explicasen en profundidad estas calificaciones los usuarios reportaron una serie de errores en la navegación de la aplicación que causaban confusión.

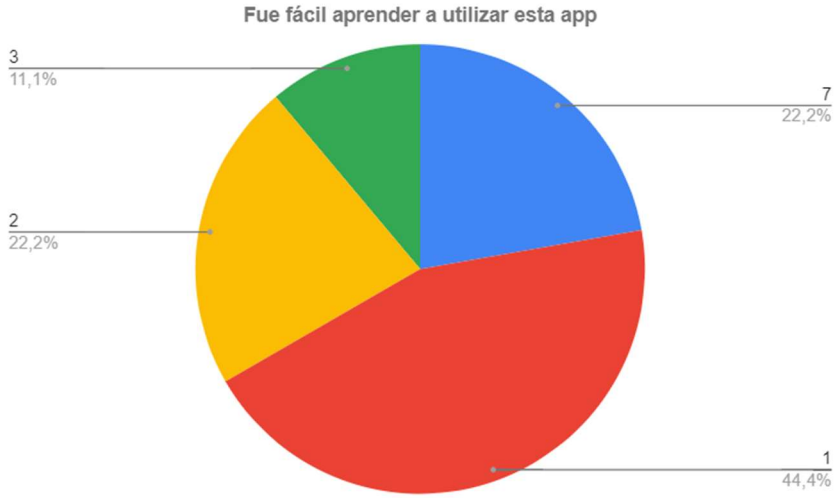


Figura 53. Pregunta 5 del formulario

Sobre la facilidad de aprendizaje de la aplicación se ve que existía una opinión muy positiva, debido a la similitud de usabilidad de la aplicación con otras aplicaciones ya existentes. Aunque la nota fuese positiva, se detectaba que algunos usuarios menos familiarizados con el mundo de las aplicaciones móviles habían tenido problemas utilizando la App, por esto se decidió elaborar una guía de usuario que les permitiese entender con rapidez cómo realizar las tareas claves que necesiten.

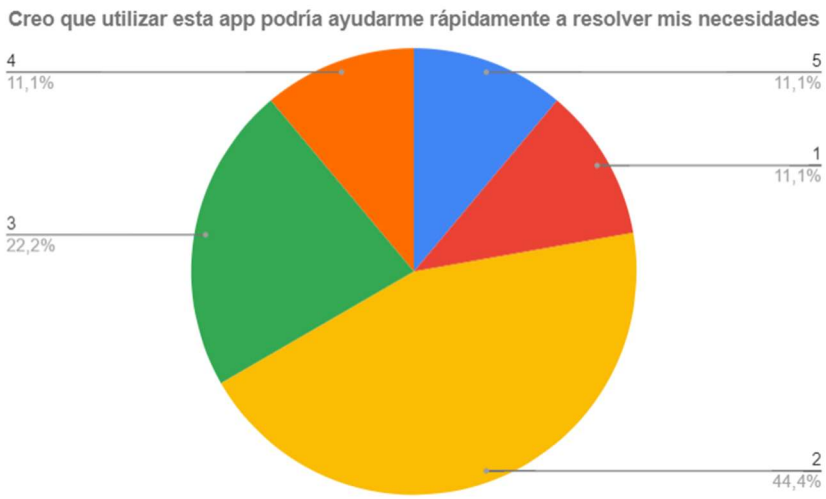


Figura 54. Pregunta 6 del formulario

Como podemos observar en este grafico la recepción de la idea de Agrari fue positiva, con un total de 7 personas dando una nota positiva y tan solo dos personas indicando que el producto necesitaba cierta mejoría y funcionalidades para cubrir todas sus necesidades.

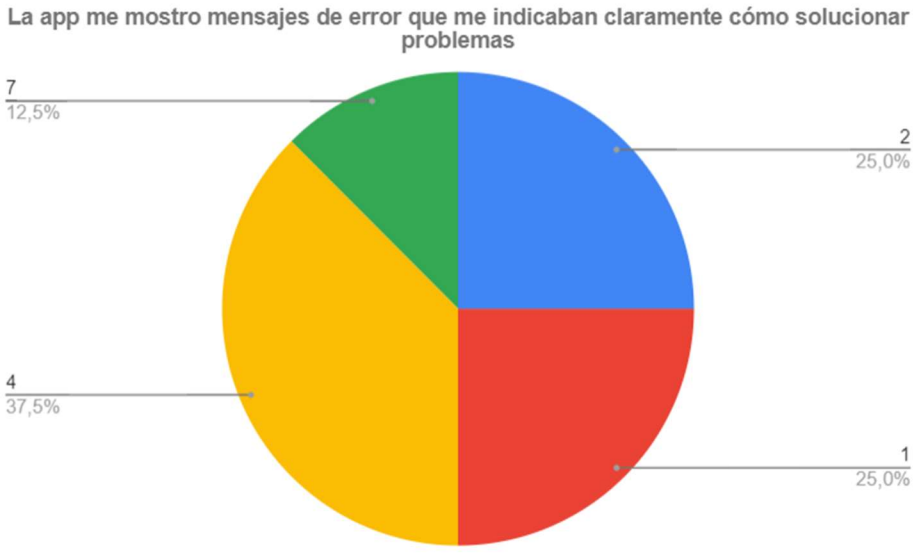


Figura 55. Pregunta 7 del formulario

Un área en el que se detectaron posibles deficiencias es el campo de comunicación de las actividades y errores con el usuario. Aquí se puede ver cómo varios usuarios han sufrido errores con la aplicación sin saber por qué se habían producido, antes este problema se decidió añadir mensajes de error y éxito en todas las operaciones que podían dar algún tipo de error.

Figura 56. Pregunta 8 del formulario

Respecto a la recuperación ante errores del sistema se observó que muchos usuarios se quejaban de no haber podido subsanar situaciones de error en el momento. Tras preguntar de manera individual a los usuarios que pusieron las quejas se detectó un error a la hora de subir productos que eliminaba los datos acerca de este.

Figura 57. Pregunta 9 del formulario

Respecto a la información mostrada al usuario se identificaron varias quejas sobre los textos que indicaban al usuario cómo hacer uso de algunas funcionalidades de la aplicación, con mensajes confusos o que no guiaban al usuario hasta su objetivo final. Ante este problema se decidió abordar todos los futuros textos a redactar con una escritura basada en *UX Writing* (una técnica basada en adecuar todos los textos al usuario final y guiarlo hasta su objetivo).

Figura 58. Pregunta 10 del formulario

Respecto a la organización de la información se vieron notas muy dispares, esto es debido a que el perfil de cliente y el perfil de productor tenían dos esquemas de información muy distintos, las peores notas venían por parte de usuarios con el perfil de productor y por lo tanto nos indicaba la necesidad de reorganizar los distintos elementos de una manera más natural al usuario.

Figura 59. Pregunta 11 del formulario

Al igual que la pregunta sobre la calidad de la información mostrada, se identificaban algunas quejas por parte de los usuarios a la hora de cumplir las tareas establecidas. Con el objetivo de aumentar el número de pedidos y productos dentro de la aplicación se buscó mejorar los textos que daban instrucciones al usuario.

Figura 60. Pregunta 12 del formulario

Con respecto a la organización de la información en cada una de las pantallas se buscó arreglar los problemas encontrados mediante la introducción del rediseño propuesto por Aitana Castellanos. Este rediseño intentó mejorar la usabilidad del sistema y la información que se muestra en él.

Figura 61. Pregunta 13 del formulario

Respecto al diseño de la interfaz no encontramos ninguna deficiencia notable.

Figura 62. Pregunta 14 del formulario

La mayoría de la gente no tuvo una mala experiencia de usuario al interactuar con la aplicación durante el experimento.

Figura 63. Pregunta 15 del formulario

La mayor parte de notas sobre las funcionalidades fueron positivas, aunque no fueron las más altas por lo que se reconocía que existían ciertas necesidades del usuario que no se estaban cubriendo completamente y que necesitaban ser implementadas en un futuro, como podían ser mejores herramientas de gestión de pedidos para los productores.

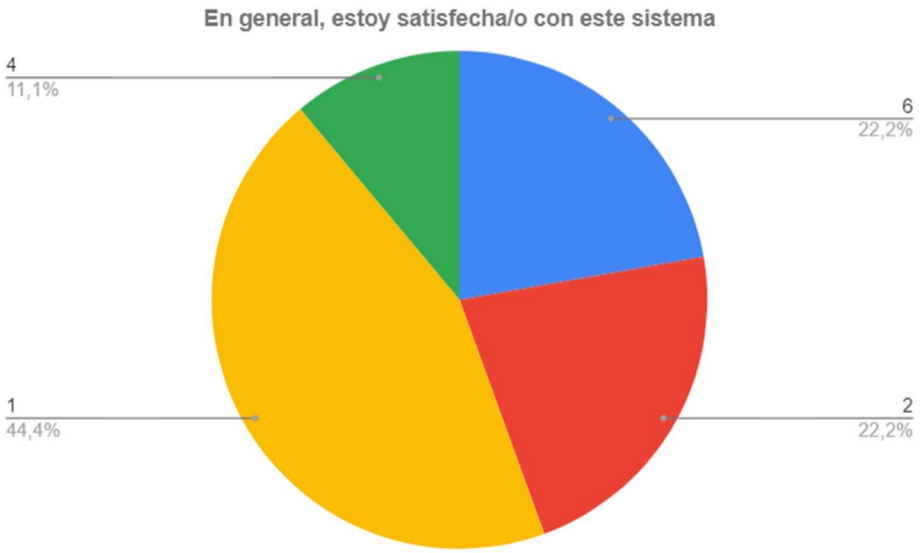


Figura 64. Pregunta 16 del formulario

Por último, podemos ver como en general la mayoría de los usuarios estaban contentos con la aplicación. Aunque en general la visión era positiva se han podido detectar múltiples puntos de mejora a través de este formulario que serán tratados a lo largo del incremento del MVP y posteriormente validados otra vez por los usuarios.

4.6.2 Validación del segundo incremento del MVP

Para esta segunda fase se decidió realizar una prueba más a fondo con entrevistas directas al usuario. Para esta entrevista se establecieron una serie de pruebas para validar con usuarios nuevos en la aplicación la facilidad de navegar por la aplicación y cumplir distintas tareas. Las pruebas definidas llevaban asociadas unas métricas que permitían validar el éxito o fallo de estas, como por ejemplo la cantidad de clics que le llevaba completar un pedido o el tiempo gastado completando una tarea.

Este tipo de pruebas fueron más abiertas que las anteriores y se realizaron de manera presencial. Realizar la segunda validación de esta manera permitió identificar problemas que mediante otro tipo de pruebas no habrían sido detectados, como el posicionamiento incomodo de algunos botones, textos confusos que no transmiten el mensaje deseado al usuario o información mal agrupada.

A lo largo de estas entrevistas se iban tomando notas sobre el comportamiento del usuario para ser revisadas posteriormente junto con las respuestas y para así proporcionar de un contexto completo para entender los pasos y pensamientos que tiene el usuario al completar las diversas tareas.

Dado a la situación médica actual causada por el COVID-19 y ya que este tipo de pruebas requerían de la presencialidad del usuario en un espacio controlado, este cuestionario fue realizado de manera limitada a conocidos y familiares del equipo que ya habían utilizado previamente la aplicación.

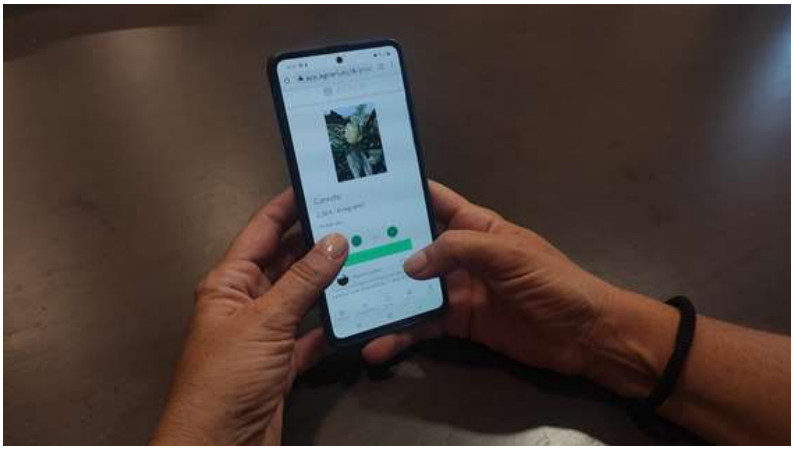


Figura 65. Usuario realizando el test

El proceso de entrevista se realizó de la siguiente forma:

Se efectuó una breve introducción del estudio que se iba a realizar y las distintas tareas que se deberían completar, además se le pedía a la persona que estaba realizando la entrevista que narrase las acciones que realizaba y sus pensamientos. Estas instrucciones buscaban preparar al usuario y hacerle sentir de una forma natural y con una vista de que es lo que necesitábamos de él.

Las tareas a realizar eran las siguientes:

1. Acabas de crear tu cuenta y quieres comprar los siguientes productos para hacer una receta que has visto por internet: patata (250gr), cebolla (2 unidades), alcachofa (1kg) y una hogaza de pan (1 unidad). Añade estos productos a tu carrito y completa el pedido.

Esta tarea buscaba asegurar que el proceso de añadir productos al carrito era claro, sobre todo a la hora de trabajar con distintos tipos de unidades y cantidades. Se buscaba que la tarea fuese completada en menos de 8 minutos para considerarse que se había resuelto correctamente.

2. Revisa tu pedido y ves que has introducido mal la cantidad de algún producto, intenta contactar con el productor para indicárselo y cancela el pedido.

En esta tarea el objetivo era observar si la funcionalidad de chat era intuitiva y sencilla para el usuario o si era difícil de encontrar a la hora de querer comunicarse con el productor. Se consideró que la tarea había sido completada correctamente si se tardaba tan solo 4 minutos en completarla.

3. Te ha llegado tu pedido y estás contento con la compra, deja una reseña positiva al productor para que sepa qué opinas de su pedido. Una vez la completes revísala en su perfil para ver que se ha publicado correctamente.

Con esta tarea se buscó validar que la función de reseñas seguía un ciclo lógico y los pasos a seguir eran fáciles de comprender. Se consideró que la tarea había sido realizada con éxito si se completaba en 8 minutos.

Tras acabar las entrevistas se gradeció su participación y se pidió una opinión general sobre el producto para asegurarnos que todo lo que estaba pensando el participante había sido reflejado en las notas que hemos tomado.

Tras realizar las entrevistas todos los participantes consiguieron completar las tareas en el tiempo estipulado, aunque con algunas dificultades. Las debilidades detectadas a través de las respuestas de los entrevistados fueron las siguientes:

- El botón del chat estaba en una posición incómoda para los usuarios, requería cambiar la posición de la mano. Para dar solución a este problema se buscó reorganizar las distintas secciones de la barra de navegación para incluir en esa parte el chat y volver a validar con los usuarios su funcionamiento.
- Tras enviar un mensaje en el chat no se desplazaba hacia abajo ni tampoco se borraba el mensaje que había escrito, y por lo tanto un usuario se sintió frustrado al no entender qué estaba sucediendo.
- Las reseñas eran sencillas de crear, pero se podrían priorizar los pedidos que no tenían reseñas para que se mostrasen primero. Podrían quedar pedidos sin reseñas solo porque el usuario no quisiese bajar hasta abajo para revisar todos sus pedidos. Ante esto se decidió no reordenar los pedidos, sino plantear una nueva organización del menú de pedidos. Los pedidos completados y cancelados pasaron a estar dentro del perfil del usuario para que los pudiese revisar cuando lo desease. Por ultimo los pedidos nuevos serían los que se mostraran en la sección de pedidos, y aquellos que estuviesen pendientes de ser reseñados se mostrarían a través de una ventana emergente dentro de esta sección y la próxima vez que se abriese la app después de completar un pedido.

5. Cronología del TFG

Figura 66. Cronología del TFG

En el proceso de desarrollo del TFG se identificaron hasta un total de 7 etapas que se pueden organizar en 3 grupos distintos:

- Inicio del proyecto: Se inició el TFG y se acordó entre todos los miembros el backlog de tareas que se abordarían a lo largo de los dos experimentos.
- Primer *sprint* de desarrollo: En este grupo se incluyeron las fases de desarrollo de las unidades de trabajo, validación de las pruebas en un entorno real a través de experimentos, y por último la reunión de retrospectiva con el equipo.
- Segundo *sprint* de desarrollo: Las etapas fueron las mismas que en el anterior *sprint* dado que es un proceso iterativo que se repetirá a lo largo de todos los ciclos de desarrollo que se planifiquen.

La dedicación al proyecto no ha sido completa, por lo que se añade una tabla debajo para mostrar de una manera resumida la dedicación de horas a cada una de las partes del proyecto.

Tarea	Tiempo
Formación de las tecnologías y buenas practicas	40h
Definición de la idea de negocio y especificación de tareas	25h
Desarrollo y validación de la aplicación	230h
Redacción de la memoria	65h
Total	360h

6. Conclusiones y trabajos futuros

A lo largo de este proyecto se ha buscado validar una serie de hipótesis que fueron planteadas al inicio y así descubrir si la solución planteada efectivamente resolvía satisfactoriamente esos problemas.

Tras haber realizado ambos experimentos el equipo ha podido validar parte de la idea de negocios, y darse cuenta de esta forma que existe una necesidad de un mercado de productos ecológicos de proximidad que agrupe al máximo de productores posibles.

El desarrollo del proyecto en equipo ha resultado ser una muy buena herramienta para poner a prueba las técnicas y conocimientos obtenidos a lo largo de la carrera en un entorno real bajo el contexto de emprendimiento.

Tras finalizar los dos experimentos hemos concluido que el desarrollo de la idea de negocio de Agrari ha resultado todo un éxito obteniendo muy buenas reseñas por parte de los usuarios. Estas calificaciones se han visto incrementadas entre los dos experimentos que se han realizado y han validado finalmente todos los objetivos planteados al inicio del proyecto.

El proyecto además ha conseguido generar mucho interés por parte de los medios como PlazaRadio [5], Valencia Plaza [6], o ActualFruVeg [19], y ha contado con mucho apoyo por parte de StartUPV que está ayudando activamente a llevar adelantes este proyecto y diversos grupos de consumo nos han mostrado interés por participar en futuros experimentos.

Este proyecto ha sido posible gracias a los conocimientos adquiridos durante la carrera, estos conocimientos me han permitido desarrollar un proyecto software completo de pequeño tamaño. Gracias a asignaturas como PIN (Proyecto de Ingeniería del Software) que ayudó a establecer esas bases de conocimiento sobre emprendimiento y técnicas habituales en el proceso de desarrollo de software ágil, como establecer una serie de hitos, un proceso de desarrollo por fases, reuniones de retrospectiva con el equipo entre otros, DDS (Diseño del Software) que permitió aprender sobre buenas técnicas de desarrollo a través de *Clean Code* y patrones de diseño que posibilitasen hacer proyectos escalables, DSM (Diseño del Software por Modelado) que enseñó cómo plantear un sistema complejo clase a clase para diseñar soluciones robustas y completas sin fallos y CSO (Calidad del Software) que consiguió establecer una serie de herramientas y procesos para validar productos software desarrollados con el cliente.

Trabajos futuros:

Actualmente el proyecto de Agrari sigue en desarrollo y se han reportado varias ideas para futuros *sprints* que buscan abordar ciertas necesidades en el mercado que originalmente no se habían detectado y que podrían ser objetivo de futuros desarrollos. Algunas de estas ideas son la posibilidad de ofrecer paquetes personalizados

Aunque el futuro del proyecto es muy ambicioso se cuenta con mucho apoyo por parte de StartUPV que ayuda activamente a enfocar el proyecto y facilitarnos al máximo la experiencia emprendedora. Además, se está preparando una expansión del equipo para abordar futuros problemas que puedan aparecer en el *roadmap*, incluyendo una persona de marketing que ayudara con el lanzamiento en redes sociales y campañas publicitarias del proyecto cuando se lance al público, y un perfil de ADE que ayude a llevar las finanzas del proyecto y posibles financiaciones que surjan.

El futuro de Agrari busca establecer un *marketplace* de productos agrícolas que cubra toda el área de España y que sea utilizado por familias para realizar sus compras mensuales.

7. Referencias

1. *Angular* (Sin fecha) *Angular.io*. Disponible en: https://angular.io/ (Accedido: 23 de agosto de 2021).
2. Appreciem (Sin fecha) APPRECIEM, Appreciem.com. Disponible en: https://appreciem.com/ (Accedido: 6 de septiembre de 2021).
3. Cheng, F. (2018) *Build mobile apps with ionic 4 and firebase: Hybrid mobile app development*. 2.ª ed. Nueva York, NY, Estados Unidos de América: APRESS.
4. Correos Market (sin fecha) Correos.es. Disponible en: https://www.market.correos.es/ (Accedido: 6 de septiembre de 2021).
5. Cremades, P. (2021) *Jaume Merino, de Agrari: "Eliminamos intermediarios para darle valor al agricultor"*, *Valencia Plaza*. Disponible en: https://plazaradio.valenciaplaza.com/a-pie-de-calle-t1x180-desde-universidad-politecnica-de-valencia-con-jaume-merino (Accedido: 23 de agosto de 2021).
6. Ediciones Plaza, S. L. (sin fecha) *La UPV incorpora siete nuevas startups a su ecosistema emprendedor*, *Valenciaplaza.com*. Disponible en: https://valenciaplaza.com/la-upv-incorpora-siete-nuevas-startups-a-su-ecosistema-emprendedor (Accedido: 23 de agosto de 2021).
7. Editors, E. (2021) *Worldwide ecommerce will approach $5 trillion this year*, *Emarketer.com*. Insider Intelligence. Disponible en: https://www.emarketer.com/content/worldwide-ecommerce-will-approach-5-trillion-this-year (Accedido: 23 de agosto de 2021).
8. Europa Press (2016) *¿Cuánto gastamos los españoles al año en fruta y verdura? Unos 454 euros de media por persona*, *elEconomista*. Disponible en: https://www.eleconomista.es/economia/noticias/7615389/06/16/Los-espanoles-segundos-consumidores-de-fruta-y-verdura-en-el-mundo-con-un-gasto-de-454-euros-en-2015.html (Accedido: 23 de agosto de 2021).
9. Europa Press (2021) «El sector agroalimentario aportó 100.000 millones de euros a la economía española en 2020, el 9,7% del PIB». Disponible en: https://www.europapress.es/economia/noticia-sector-agroalimentario-aporto-100000-millones-euros-economia-espanola-2020-97-pib-20210708141047.html (Accedido: 23 de agosto de 2021).
10. Farmidable (Sin fecha) Farmidable.es. Disponible en: http://farmidable.es/ (Accedido: 23 de agosto de 2021).
11. Freshco's (Sin fecha) Freshcosapp.com. Disponible en: https://freshcosapp.com/ (Accedido: 6 de septiembre de 2021).
12. Ionic (sin fecha) *Cross-platform mobile app development: Ionic framework*, *Ionicframework.com*. Disponible en: https://ionicframework.com/ (Accedido: 23 de agosto de 2021).
13. La compra así da gusto (Sin fecha) Lacompraasidagusto.com. Disponible en: https://lacompraasidagusto.com/ (Accedido: 6 de septiembre de 2021).
14. Millán, M. (2020) *Los caros céntimos en el eslabón de los agricultores*, *Heraldo de Aragón*. Disponible en: https://www.heraldo.es/noticias/aragon/2020/02/09/agricultores-precio-gastos-produccion-en-el-campo-1357691.html?autoref=true (Accedido: 23 de agosto de 2021).
15. Montes, L. (2019) *Este gráfico muestra cómo el ecommerce está ganando terreno al supermercado físico para llenar la cesta de la compra de los españoles*, *Business Insider*

España. Disponible en: https://www.businessinsider.es/grafico-muestra-como-ecommerce-gana-cuota-sector-alimentacion-450455 (Accedido: 23 de agosto de 2021).
16. Nayrolles, M. (2018) *Angular Design Patterns: Implement the Gang of Four patterns in your apps with Angular*. Birmingham, Inglaterra: Packt Publishing.
17. O'Hanlon, P. (2019) *Advanced TypeScript Programming Projects: Build 9 different apps with TypeScript 3 and JavaScript frameworks such as Angular, React, and Vue*. Birmingham, Inglaterra: Packt Publishing.
18. *PSSUQ (Post-Study System Usability Questionnaire)* (2019) *Uiuxtrend.com*. Disponible en: https://uiuxtrend.com/pssuq-post-study-system-usability-questionnaire/ (Accedido: 23 de agosto de 2021).
19. Redaccion, A. N. V. (2021) *AGRARI, la app para los productos agrícolas de proximidad*, *Actualfruveg.com*. Disponible en: https://actualfruveg.com/2021/04/27/agrari-app-proximidad/ (Accedido: 23 de agosto de 2021).
20. TerretaSana (sin fecha) Terretasana.es. Disponible en: https://www.terretasana.es/ (Accedido: 6 de septiembre de 2021).
21. Salvatierra, J. (2020) *El uso del comercio electrónico se dispara entre las empresas españolas*, *Elpais.com*. Disponible en: https://elpais.com/economia/2020-10-20/el-uso-del-comercio-electronico-se-disparo-en-2019-entre-las-empresas-espanolas.html (Accedido: 23 de agosto de 2021).
22. VanderKam, D. (2019) *Effective typescript: 62 Specific ways to improve your typescript*. O'Reilly Media.
23. *What is the use of DTO instead of Entity?* (sin fecha) *Stackexchange.com*. Disponible en: https://softwareengineering.stackexchange.com/questions/373284/what-is-the-use-of-dto-instead-of-entity (Accedido: 23 de agosto de 2021).

8. Glosario

***E-Commerce**:* Comercio electrónico, todo aquel tipo de ventas de cualquier carácter que se realizan en línea.

***Mockup**:* Diseños tempranos sin funcionalidad para mostrar el aspecto del producto o la estructura de este.

***Roadmap**:* Trayectoria planificada de un proyecto con una serie de hitos a cumplir.

***Design Thinking**:* Técnica de generación de ideas centrada en el usuario que busca dar soluciones a sus problemas.

***Marketplace**:* Plataforma de venta en línea en la que participan múltiples vendedores.

***Startup**:* Empresa nueva y de reducido tamaño con carácter usualmente tecnológico y una capacidad de crecimiento muy elevada.

***App**:* Aplicación móvil.

***Early Adopter**:* Adoptante temprano del producto que conoce del sector y puede ayudar en el desarrollo a través de sus opiniones.

***Brainstorming**:* Lluvia de ideas en la cual se busca generar la mayor cantidad posible de ideas sin juzgarlas para después cribarlas.

***Stakeholder**:* Aquella persona interesada e involucrada en la empresa o proyecto, como podrían ser otras empresas, inversores o clientes.

***Target**:* Publico objetivo de una aplicación que cumple una serie de condiciones como puede ser edad o sexo entre otros.

***Framework**:* Base de código previa que permite a los desarrolladores ahorrar tiempo proporcionándoles de herramientas y buenas prácticas a seguir en un desarrollo.

***Backend**:* Desarrollo de la parte no visible de un software, como puede ser la lógica de operaciones sobre base de datos, servicios etc.

***MVP**: Minimum Viable Product.* Mínimo Producto Viable es la versión más sencilla y con funcionalidades limitadas del producto diseñado que se busca lanzar al mercado.

***CI**: Continuous Integration*. Integración continua del código que asegura su correcto funcionamiento.

***CD**: Continuous Deployment.* Distribución continua del producto en los diversos entornos de desarrollo.

***Webhooks**:* Un sistema que realiza una llamada a un servicio que se dedica a escucharlo para informarle de un evento y realizar acciones en consecuencia a este.

Sprint: Fase de desarrollo de un proyecto de tiempo reducido que tiene como objetivo crear un producto o un incremento de este.

8. Índice de figuras

9. Anexos

Guía de usuario de Agrari:

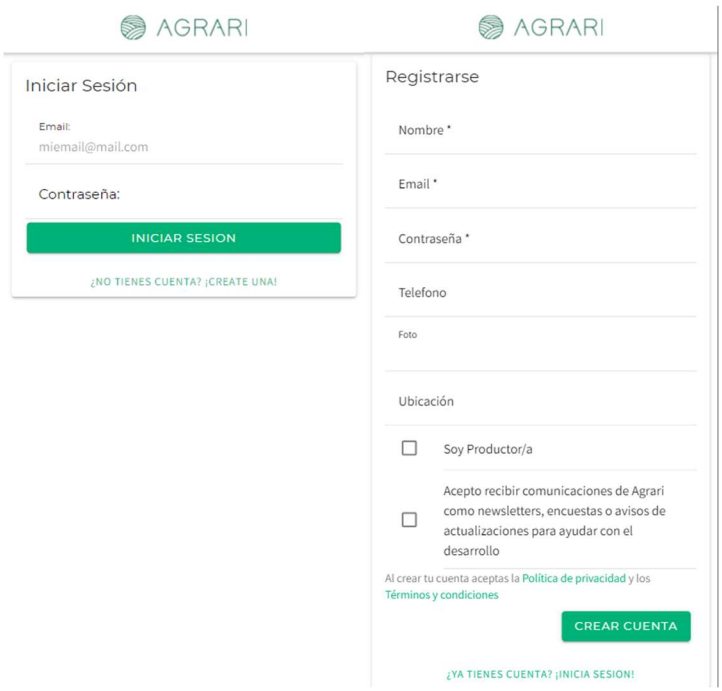
Agrari es la aplicación que busca conectar personas productoras con compradores de alimentos. Para poder acceder a la app hay que abrir la web app.agrari.es desde tu móvil (aunque también funciona en ordenador).

CÓMO USAR LA APP

Pantalla de inicio

Nada más entrar al portal se nos muestra una pantalla para identificarnos. En caso de ya tener una cuenta creada basta con poner el correo electrónico y la contraseña (en caso de olvidarla os podemos mandar un enlace para recuperarla).

Sino tienes una cuenta hay que pulsar el botón para crear una cuenta, esto nos llevará a un formulario para crear tu cuenta, los campos obligatorios son el nombre, el email y la contraseña, aunque es muy recomendable poner una foto y os animamos a ello para daros a conocer mejor a los compradores. Para crear una cuenta de productor será necesario marcar la casilla "Soy Productor/a". En caso de querer recibir comunicaciones de nuestra parte podéis indicarlo en la casilla de justo debajo.



Barra de navegación

Una vez inicies sesión se mostrará una barra de navegación en la parte de abajo. Esta barra sirve para navegar por la aplicación. Cada icono se relaciona con una de las siguientes secciones:

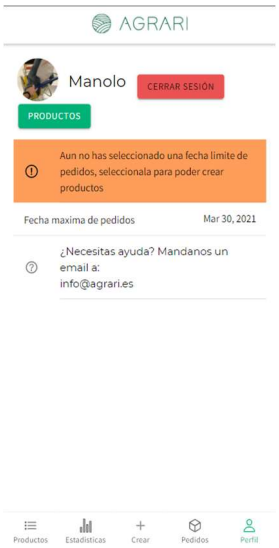
- **Productos:** es un buscador de todos los productos subidos en la plataforma, sirve para hacerse una idea de que productos ofrece la aplicación.
- **Estadísticas:** en esta vista se nos muestra un breve resumen de todos los pedidos realizados hasta la fecha para poder tener de forma resumida ingresos generados, productos más vendidos o el total de pedidos.
- **Crear:** se nos ofrece un formulario para publicar los distintos productos que tengamos y todos los datos que ayuden a su búsqueda.
- **Pedidos:** en esta pestaña se nos mostrarán todos los pedidos que hemos recibido separados según su estado y ordenados por fecha. Se nos permitirá avanzar los pedidos por sus diferentes fases hasta la entrega del producto.
- **Perfil:** es el panel de control de nuestro perfil, aquí se permitirá cambiar la fecha de pedidos límite, visualizar nuestros productos y solicitar ayuda.

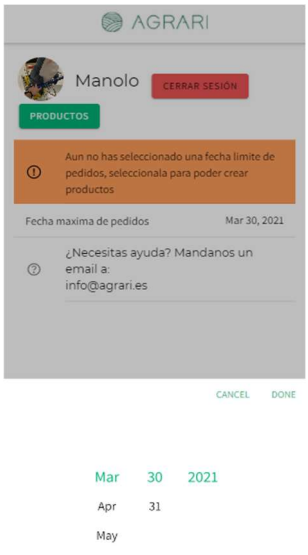
Imagen 3.

Vista de perfil

Nada más iniciar sesión se nos llevará por defecto a la vista de nuestro perfil, aquí deberemos indicar la fecha máxima de pedidos (hasta cuando se aceptarán pedidos para el siguiente mercado agroecológico de la UPV).

Si pulsamos sobre productos se mostrarán todos los productos que ya hayas creado en la aplicación, y en caso de necesitar ayuda con cualquier cosa puedes pulsar el botón de "¿Necesitas ayuda?" para enviarnos un email, o contactar directamente con nosotros llamando a través de nuestro número de teléfono.

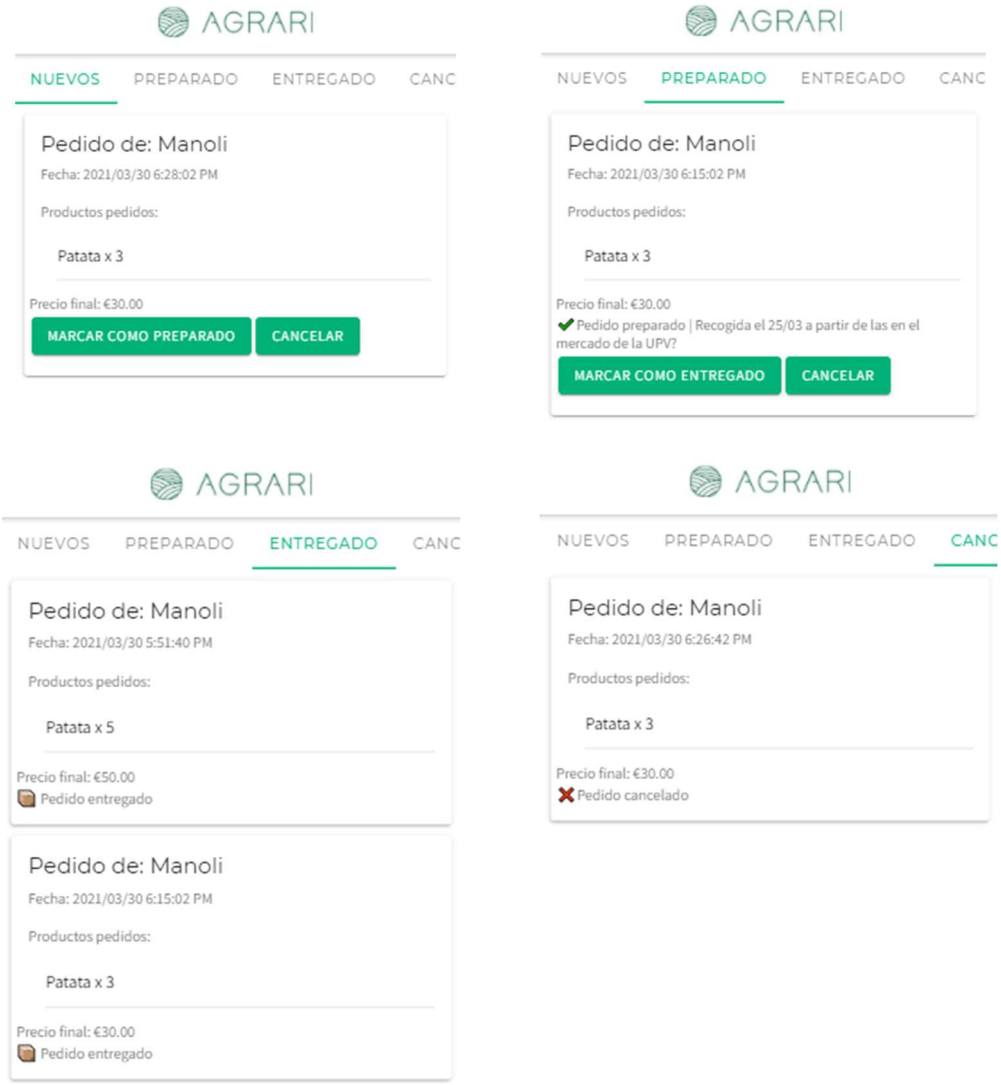




Pedidos

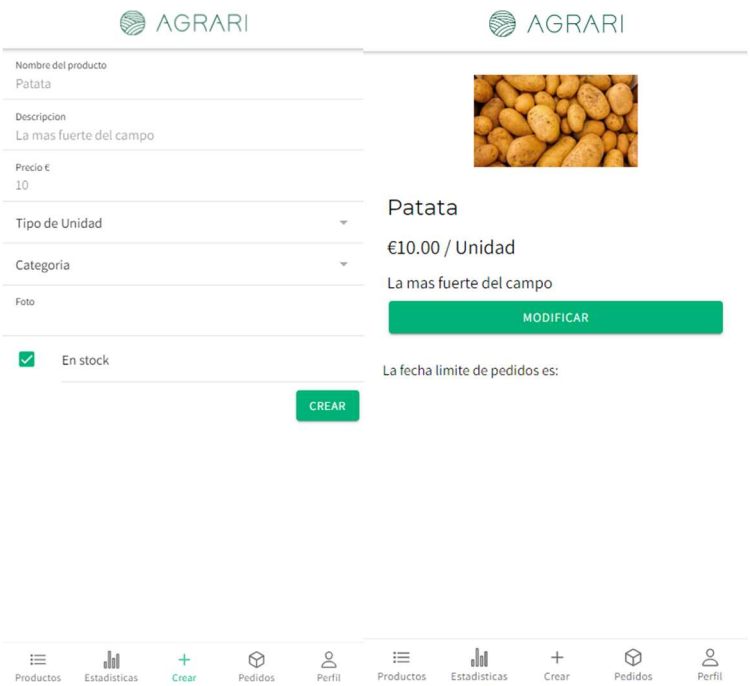
El apartado de pedidos es un resumen de los distintos pedidos a tus productos que se han realizado, estos pueden estar en 1 de 4 estados:

- **Nuevos:** Estado en el que comienzan todos los pedidos, aquí se le da la oportunidad al cliente de cancelar el pedido en caso de existir cualquier error.
- **Preparado:** Es el siguiente estado a nuevo e indica que todos los productos ya están preparados y reservados para su entrega en el próximo mercado.
- **Entregado:** En este estado aparecerán todos los pedidos que han sido entregados con éxito en el mercado, nos permite hacer un seguimiento de los pedidos más detallado.
- **Cancelado:** El último estado indica todos los pedidos que han sido cancelados por el usuario o por el productor.



Crear

Aquí podréis publicar vuestros productos en la plataforma, para ello es necesario rellenar todos los datos que se piden, e indicar si hay stock (producto disponible). En caso de agotarse un producto o que no vaya a venderse en el próximo mercado, podéis indicarlo desde “Perfil”, y luego entrando a “Productos”.



Estadísticas

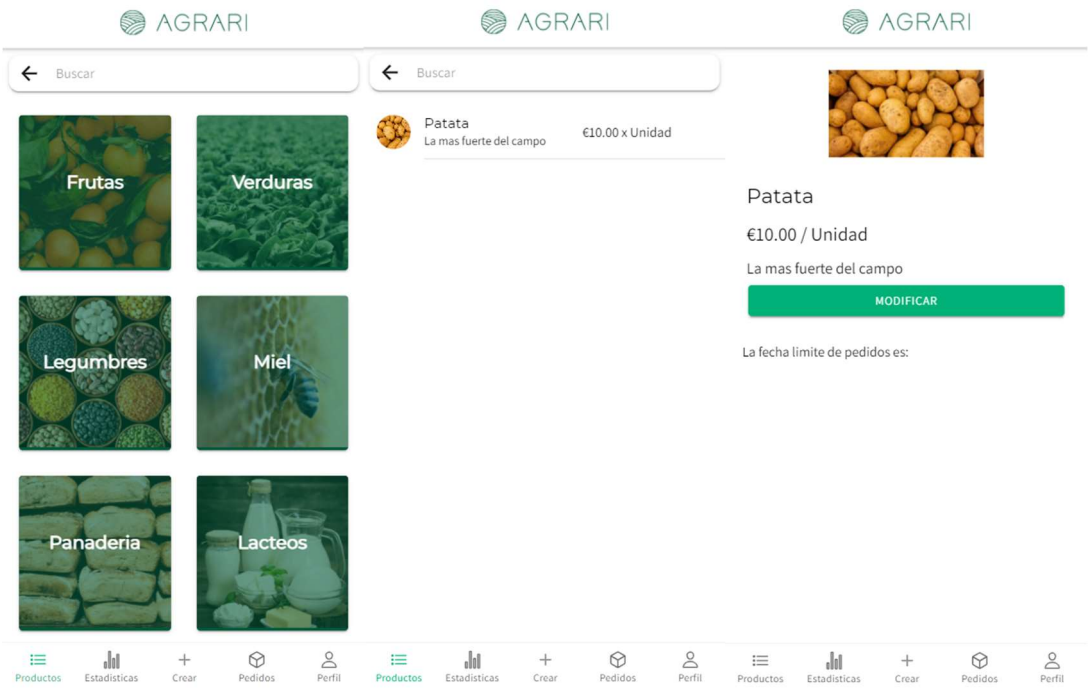
En el apartado de estadísticas se mostrará un resumen global de las ventas realizadas, dinero que se ha facturado, y el número total de pedidos.

Productos

Por último, está la sección de productos, que es el listado de todos los productos que existen en la aplicación. Aquí podrás ver lo que han publicado otros productores además de los tuyos.

Para filtrar los productos tenemos 8 categorías globales que son las mismas que se piden cuando se crea un producto, también hay una barra de búsqueda para filtrar los productos por su nombre.

Al pulsar sobre un producto se abrirá su ficha y desde aquí se podrán modificar si es un producto que has creado, o ver todos los datos de un producto en específico.


AGRARI
Buscar
Frutas
Verduras
Legumbres
Miel
Panaderia
Lacteos
Productos
Estadisticas
Crear
Pedidos
Perfil
AGRARI
Buscar
Patata
La mas fuerte del campo
€10.00 x Unidad
Productos
Estadisticas
Crear
Pedidos
Perfil
AGRARI
Patata
€10.00 / Unidad
La mas fuerte del campo
MODIFICAR
La fecha limite de pedidos es:
Productos
Estadisticas
Crear
Pedidos
Perfil