

2021-03

# DISEÑO E IMPLEMENTACIÓN DE UNA PLATAFORMA DE MARKETPLACE INTEGRADO AL SITIO INFOREDCHILE

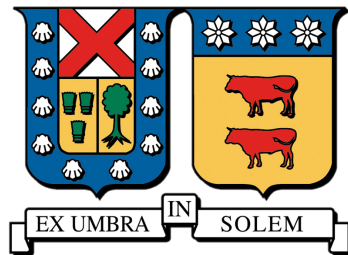
ZÚÑIGA CACERES, MATIAS

---

<https://hdl.handle.net/11673/50096>

*Repositorio Digital USM, UNIVERSIDAD TECNICA FEDERICO SANTA MARIA*

**UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA**  
**DEPARTAMENTO DE ELECTRÓNICA**  
**VALPARAÍSO - CHILE**



**DISEÑO E IMPLEMENTACIÓN DE UNA  
PLATAFORMA DE MARKETPLACE  
INTEGRADO AL SITIO INFOREDCHILE**

**MATÍAS ZÚÑIGA CÁCERES**

**MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE INGENIERO CIVIL  
TELEMÁTICO**

**PROFESOR GUÍA: MOHAMED ABDELHAMID**

**PROFESOR CORREFERENTE: FRANCISCO CABEZAS BERRÍOS**

**MARZO - 2021**

*A mis padres, por enseñarme a ser lo que soy;  
a mis hermanos, por soportarme este último año;  
y a mis amigos, por apoyarme cuando faltaba motivación.*

## **Resumen**

El presente trabajo tiene el objetivo de documentar el diseño e implementación de un sistema de MarketPlace multi-vendedor, el cual puede ser integrado con un directorio público de MiPymes (InforedChile), y surge como respuesta a la problemática que muchas de estas empresas tienen en el contexto de la pandemia de COVID-19.

Para esto, se analizan las alternativas existentes de sistemas que resuelven problemáticas similares, escogiendo uno para ser tomado como base de la solución propuesta.

A partir de la base elegida, se diseñan e implementan interfaces de usuario que armonizan con la plataforma InforedChile, y se crean extensiones de integración para permitir que cada uno de los usuarios de InforedChile pueda ofrecer sus productos.

Además, se desarrolla una integración con la pasarela de pagos Khipu, con el objetivo de poder realizar pagos en la plataforma con tarjetas de crédito y débito chilenas.



## **Abstract**

The present work presents the design and implementation of a multi-vendor MarketPlace system, which can be integrated with a public business directory (InforedChile), and arises as a response to the problem that many *MiPyme* companies have in the context of the COVID-19 pandemic.

For this, existing alternatives that solve similar problems are analyzed, choosing one to be taken as the basis of the proposed solution.

Starting from the chosen base, user interfaces are designed and implemented that harmonize with the InforedChile platform, and extensions are created to integrate the InforedChile platform and allow each of its users to offer their products.

In addition, an integration is developed with the Khipu payment gateway, in order to be able to make payments on the platform with Chilean credit and debit cards.

## Glosario

**API** Application Programming Interface. Interfaz que habilita la interacción de una aplicación con otra pieza de software .

**Django Framework** para desarrollo de aplicaciones Web, de código abierto, y escrito en **Python** .

**E-commerce** Comercio electrónico .

**Framework** Plataforma para la creación de aplicaciones, que provee funcionalidad genérica para ser modificada .

**Frontend** La interfaz del lado del cliente, que el usuario final ve y con la que interactúa .

**Hash** Resultado de una función criptográfica, de la cual es inviable obtener de vuelta el valor original .

**JavaScript** Lenguaje de programación multi-paradigma .

**Marketplace** Sitio web de comercio electrónico en el que la información sobre productos o servicios es proporcionada por múltiples ofertantes .

**MiPyme** Micro, Pequeña y Mediana Empresa .

**NodeJS** Entorno de ejecución de código **JavaScript**, para la capa de servidor .

**PHP** Lenguaje de scripting de propósito general .

**Plugin** Extensiones que añaden funcionalidad extra a un software existente .

**Python** Lenguaje de programación multi-paradigma .

**Rails** Framework para desarrollo de aplicaciones Web, de código abierto y escrito en **Ruby** .

**Ruby** Lenguaje de programación multi-paradigma .

**WordPress** Sistema de gestión de contenidos, escrito en PHP y altamente extensible .

# Índice general

<b>1</b>	<b>Introducción</b>	<b>1</b>
1.1	Análisis del desafío y alternativas de problemas específicos a resolver . . .	1
1.2	Definición del problema . . . . .	1
1.3	Soluciones existentes . . . . .	2
<b>2</b>	<b>Alternativas de Diseño</b>	<b>3</b>
2.1	Marco Conceptual . . . . .	3
2.2	Elección de la base . . . . .	16
<b>3</b>	<b>Propuesta</b>	<b>17</b>
3.1	Modelo de Negocios . . . . .	17
3.2	Requisitos del Sistema . . . . .	23
3.3	Arquitectura del Sistema . . . . .	27
3.4	Gestión de Riesgos . . . . .	29
3.5	Diseño de Módulos del Sistema. . . . .	31
3.6	Diseño de Interfaces del Sistema. . . . .	33
<b>4</b>	<b>Resultados</b>	<b>45</b>
4.1	Marco General. . . . .	45
4.2	Revisión del Prototipo. . . . .	48
4.3	Plan de Testing. . . . .	52
4.4	Verificación y Validación. . . . .	57
<b>5</b>	<b>Conclusiones</b>	<b>59</b>
5.1	Proyecciones . . . . .	60

<b>Anexo A GraphQL</b>	<b>61</b>
A.1 Descripción . . . . .	61
A.2 Esquemas . . . . .	62

# Capítulo 1

## Introducción

### 1.1 Análisis del desafío y alternativas de problemas específicos a resolver

InforedChile cuenta con un directorio de instituciones y servicios públicos, entregando datos básicos de ubicación, contacto y breve descripción de la organización; además cuenta con una importante cantidad en número de visitas, empresas registradas en la plataforma y páginas indexadas en Google, lo que le permite posicionarse dentro de los mejores directorios web de **MiPymes** y servicios públicos del país. Adicionalmente existe un gran incremento en la necesidad de MiPymes por adherirse al creciente mercado del **E-commerce**. Finalmente InforedChile no está generando ingresos con su plataforma web, por lo que se plantea sobre los antecedentes generales actuales la creación de un modelo de negocios en base a un desarrollo tecnológico adicional a la plataforma web que poseen.

### 1.2 Definición del problema

El E-commerce, o comercio electrónico, ha tomado cada vez una mayor relevancia en la compra de productos y servicios en el último tiempo, de hecho su crecimiento desde su creación ha sido en forma exponencial, sobre todo este último año en el que se han triplicado las ventas a través de este medio en Chile<sup>1</sup>. Frente a esto, el segmento de MiPyme se encuentra en una gran desventaja ante las grandes empresas, que cuentan con plataformas de E-commerce establecidas en su mayoría. Mientras que en el caso de MiPymes, menos de

---

<sup>1</sup>Mayo 2020: Existe una contracción de ventas físicas de -21%, y un crecimiento de 214% en las ventas respecto al mismo período del pasado año, por el lado del comercio electrónico. (Transbank, con cálculos complementarios CCS para determinar variaciones de tiendas físicas. Correspondientes a compras con tarjeta de crédito y débito)[34].

un tercio realiza comercio electrónico<sup>2</sup> o sus canales E-commerce no ofrecen la confianza necesaria a los consumidores para realizar compras, por lo que poseen un importante déficit y experiencia en este ámbito. Dado esto nace la necesidad de conexión con este canal de ventas, por parte de este segmento de empresas, con una plataforma de Marketplace que ofrezca confianza tanto para usuarios de la plataforma como para clientes de esta, oportunidad de crecimiento de negocios y que esté al alcance de micro, pequeñas y medianas empresas.

### 1.3 Soluciones existentes

Existen algunos sitios que pueden ser utilizados para que las PYMEs vendan sus productos en línea, entre los cuales destacan:

- **MercadoLibre:** Plataforma no pensada específicamente en PYMEs, donde cualquiera puede ofrecer sus productos. Si bien ofrecen un servicio gratuito, esta tiene límite de ventas anuales, y límite de tiempo durante el cual la publicación está disponible. Para eliminar estas restricciones, se debe utilizar algunas de las opciones pagadas, que cobran un gran porcentaje de comisión por cada venta[5].
- **ApañoTuPyme:** Como el nombre le dice, esta plataforma si está pensada en PYMEs, y se requiere tener un RUT de empresa para poder ofrecer productos. Si bien dicen no cobrar comisiones, hay comisiones por cada venta dependiendo del medio de pago utilizado[1].

Ninguno de estos sitios tiene a su disposición algo similar a la base de datos de InforedChile, ni integración a alguna plataforma similar a su directorio.

---

<sup>2</sup>Marzo 2019: 20,6% de los ingresos nacionales lo realizan las PYMES, asimismo un 27,2% de PYMES efectúa comercio electrónico. [6]

# Capítulo 2

## Alternativas de Diseño

### 2.1 Marco Conceptual

Existe un número importante de softwares de código abierto que ofrecen la creación sitios de comercio electrónico, las que podrían ser usadas como base para resolver este desafío. Los más importantes, categorizados por tecnología, serán listadas a continuación.

#### 2.1.1 PHP

##### 2.1.1.1 WooCommerce [40]

Es un sistema basado en WordPress que tiene una gran cantidad de extensiones gratuitas. WordPress es un sistema de gestión de contenidos utilizado ampliamente que, a pesar de ser altamente extensible, está diseñado principalmente para contenidos de estilo Blog, y no para comercio electrónico.

WooCommerce ofrece un sistema para tiendas de un vendedor, pero no un sistema estilo Marketplace. Entre las extensiones existentes, hay las que se acercan a lo requerido por el problema a resolver [35][36]. Estas extensiones, sin embargo, tienen los siguientes problemas:

- Gran cantidad de características innecesarias, lo que agrega complejidad en la estructura de estas.
- Falta de funcionalidades precisadas para la resolución de este desafío, sobre todo en cuanto a integración.

Debido a esto, en caso de utilizar WooCommerce como base de una solución, se requiere modificar en gran manera alguna de las extensiones mencionadas -agregando complejidad extra- o crear una extensión con las características necesarias para este sistema. Cualquiera de estas dos opciones requieren de un trabajo en PHP usando las APIs de WordPress.

## Shop

### Scubapro 10 litre 232 bar cylinder – single valve

10 Litre 232 Bar Cylinder with single outlet valve from Scubapro. Includes tank boot but the tank carry handle is not included and is shown for illustration purposes only.


[Add to cart](#) **£229.00**

[More details →](#)




Sort by newness ▾ Showing 1–10 of 126 results

1 2 3 4 ... 11 12 13 ▶



Build Your DSLR

[Read More](#)



Lowepro Slingshot Edge 250 AW

£109.95

[Add to cart](#)

### Cart

No products in the cart.

### Product Categories

- Clothing
  - Bags
  - Blouses
  - Dresses
  - Footwear
  - Hats
  - Hoodies
  - Shirts
  - Skirts
  - T-shirts
  - Trousers
- Electronics
  - Cameras
    - Accessories
    - Lenses
  - DVD Players
  - Headphones
  - MP3 Players
  - Radios
  - Televisions
- Kitchen
  - Blenders
  - Colanders
  - Kettles
  - Knives
  - Pots & Pans
  - Toasters

Figura 2.1: Interfaz de usuario de WooCommerce



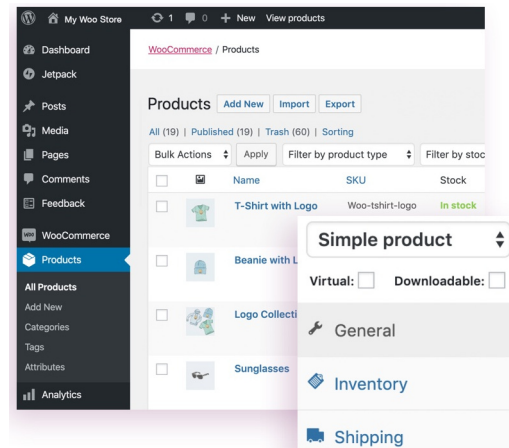


Figura 2.2: Interfaz de administración de WooCommerce

**2.1.1.1.1 Comunidad y Actividad** Existe una activa comunidad en slack de desarrolladores de extensiones[39], de donde es posible obtener sugerencias y apoyo. El proyecto es desarrollado activamente, con una gran cantidad de cambios semanales[41].

#### 2.1.1.2 Magento [2]

Sistema de comercio electrónico propiedad de Adobe, que permite crear una tienda en línea, aunque no con múltiples vendedores.

Este sistema cuenta con gran cantidad de extensiones, aunque la mayoría es de pago. En este caso, la extensión que se adecua a lo requerido tiene un costo de 349 dólares estadounidenses[38].

Dado que el precio de la extensión es prohibitivo, para basar la solución en este software, es necesario desarrollar una extensión en PHP que modifique el funcionamiento interno, utilizando su API.

**2.1.1.2.1 Comunidad y Actividad** La comunidad cuenta con un foro[4] donde se realizan y responden consultas, aunque no cuenta con mensajería instantánea al respecto. El proyecto se desarrolla de forma activa, teniendo un repositorio que recibe cambios constantemente[3].

#### 2.1.1.3 PrestaShop [18]

Proyecto independiente que ofrece un sistema de comercio en línea. Este utiliza un sistema de plantillas web que permite personalizar los temas de la tienda, y agregar nuevas funciones a través de módulos adicionales.

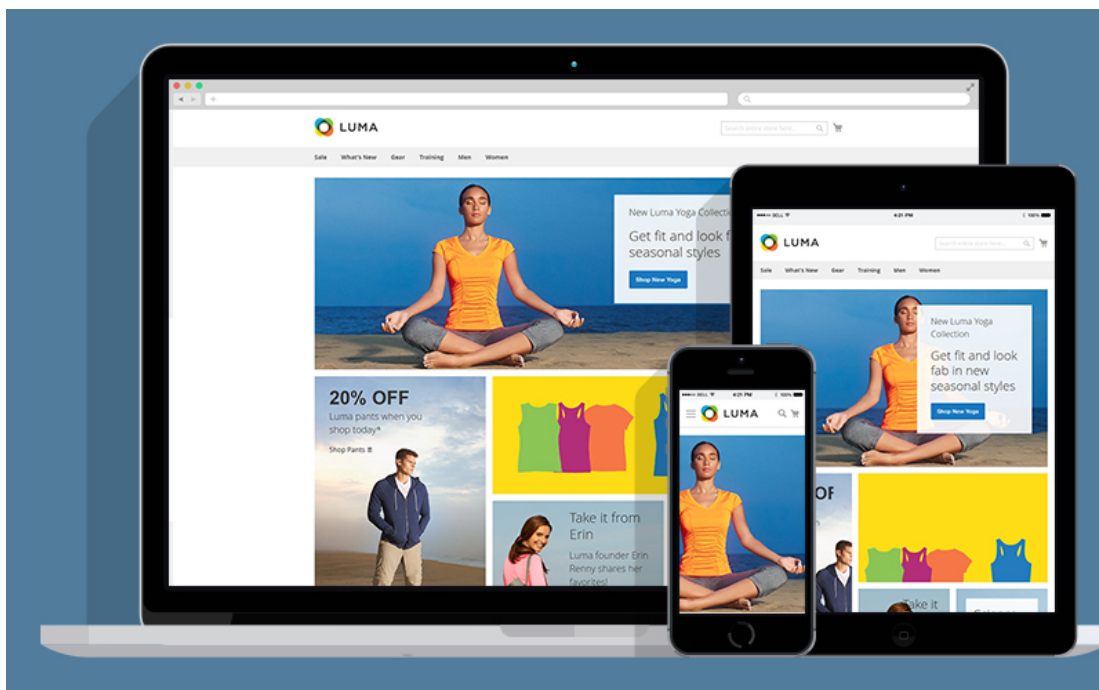


Figura 2.3: Interfaz de usuario de Magento

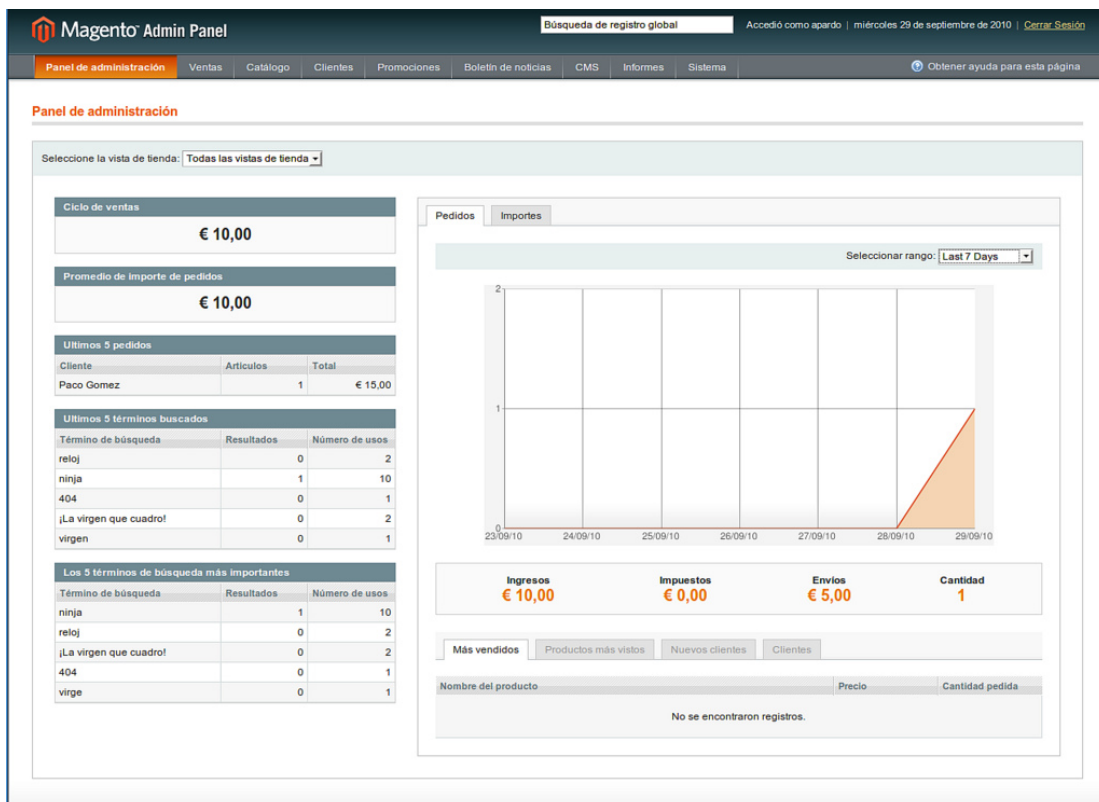
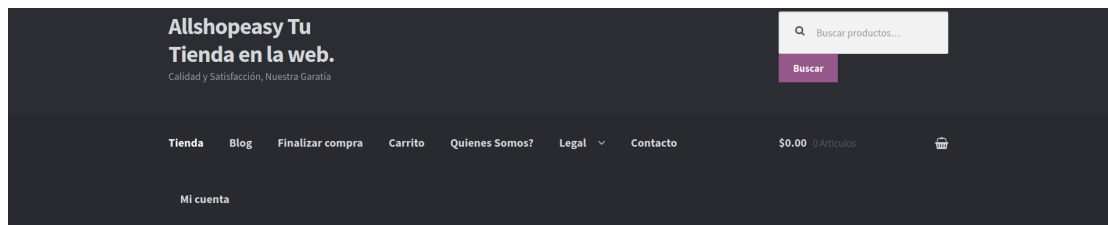


Figura 2.4: Interfaz de administración de Magento



## Tienda

Ordenar por precio: bajo a alto ▼ Mostrando todos los resultados (9)



Design It Simple Style Styrofoam Fruit-Red Apple

\$3.04

Añadir al carrito



Styrofoam Fruit-Yellow/Green Pears

\$3.04

Añadir al carrito



Design It Simple Decorative Fruit-Mini Cherries

\$4.64

Añadir al carrito

## Buscar Producto

Buscar productos...

Buscar

## Carrito

No hay productos en el carrito.

Figura 2.5: Interfaz de usuario de allshopeasy.com, basado en PrestaShop

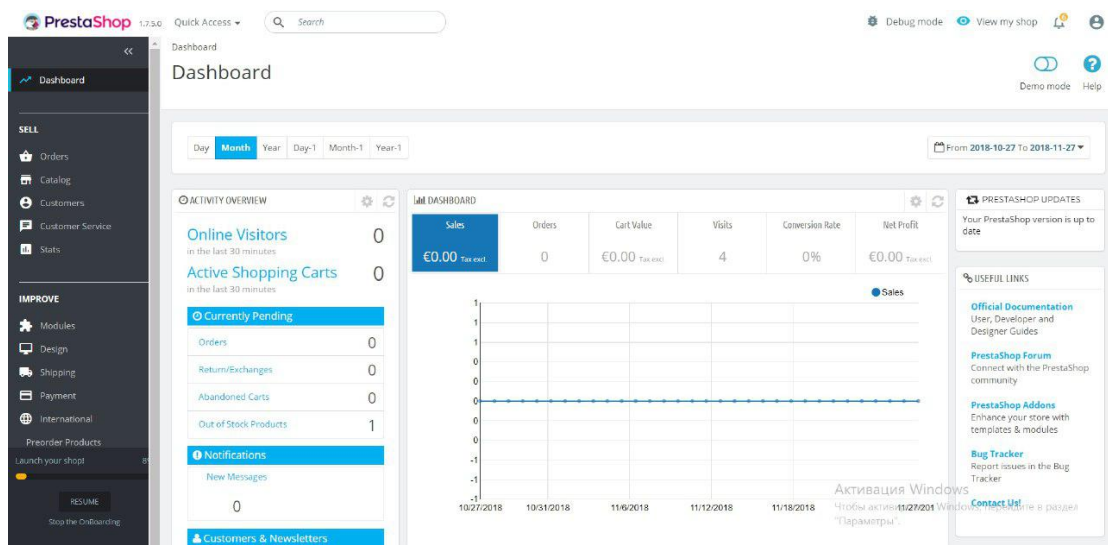


Figura 2.6: Interfaz de administración de PrestaShop

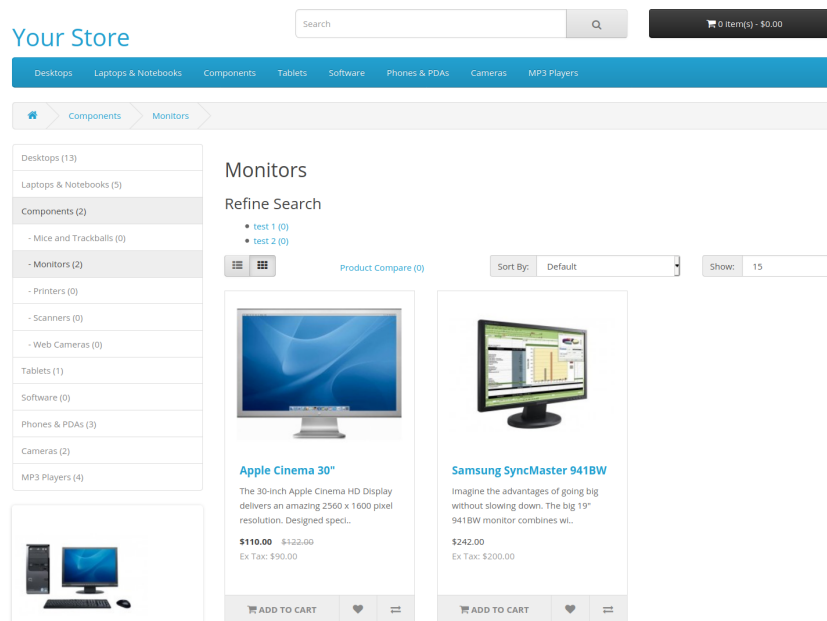


Figura 2.7: Interfaz de una tienda de ejemplo de OpenCart

Dispone de un módulo de pago para la creación de marketplaces multi-vendedor[37]. Puesto que para utilizar el módulo se debe pagar 100 euros, se requiere crear un complemento con características necesarias para este sistema usando el lenguaje PHP.

**2.1.1.3.1 Comunidad y Actividad** Existe un foro público donde la comunidad puede compartir y realizar preguntas[19]. A pesar de que no es un chat, los tiempos de respuesta son cortos y la comunidad es activa. Se puede comprobar en el repositorio de código que el software es mantenido constantemente, recibiendo mejoras y arreglos [20].

#### 2.1.1.4 OpenCart [10]

Plataforma de comercio electrónico utilizado por grandes marcas en Chile [11]. Este sistema puede manejar distintas tiendas en simultáneo [9] las que, sin embargo deben tener subdominios distintos, y no pueden navegarse bajo un mismo sitio.

Al igual que en las alternativas mencionadas previamente, existen distintas extensiones disponibles. Para lograr tener distintos vendedores en un mismo sitio, como se necesita para solucionar la problemática, se requiere una extensión de pago (70 dólares)[21], o desarrollar una alternativa desarrollada en PHP.

**2.1.1.4.1 Comunidad y Actividad** El lugar oficial para la comunicación de la comunidad es el foro proporcionado por Opencart[12]. El foro está destinado principalmente al soporte,

The screenshot displays the OpenCart administration interface. On the left is a dark sidebar with a 'NAVIGATION' menu containing links to Dashboard, Catalog, Extensions, Design, Sales, Customers, Marketing, System, and Reports. The 'Sales' menu is expanded, showing 'Orders' as the selected option. Below the sidebar, there are progress bars for 'Orders Completed', 'Orders Processing', and 'Other Statuses', all showing 0%.

The main content area is titled 'Orders' and shows a breadcrumb 'Home > Orders'. Below this is a sub-header 'Order List' and a table of orders. The table has columns for checkboxes, Order ID, Customer, Status, Total, Date Added, Date Modified, and Action. The table contains 10 rows of pending orders.

On the right side of the main content area is a 'Filter' sidebar with input fields for Order ID, Customer, Order Status, Total, Date Added, and Date Modified.

	Order ID	Customer	Status	Total	Date Added	Date Modified	Action
<input type="checkbox"/>	10833	Bui Dung	Pending	\$105.00	15/01/2021	15/01/2021	
<input type="checkbox"/>	10832	Sandro buraaaaaaaaaaaaa	Pending	\$130.00	15/01/2021	15/01/2021	
<input type="checkbox"/>	10831	Sandro buraaaaaaaaaaaaa	Pending	\$130.00	15/01/2021	15/01/2021	
<input type="checkbox"/>	10830	matija matija	Pending	\$105.00	15/01/2021	15/01/2021	
<input type="checkbox"/>	10829	matija matija	Pending	\$305.00	15/01/2021	15/01/2021	
<input type="checkbox"/>	10828	matija matija	Pending	\$205.00	15/01/2021	15/01/2021	
<input type="checkbox"/>	10827	matija matija	Pending	\$205.00	15/01/2021	15/01/2021	
<input type="checkbox"/>	10826	matija matija	Pending	\$105.00	15/01/2021	15/01/2021	
<input type="checkbox"/>	10825	matija matija	Pending	\$105.00	15/01/2021	15/01/2021	
<input type="checkbox"/>	10824	Matija Fumic	Pending	\$105.00	15/01/2021	15/01/2021	

Figura 2.8: Interfaz de administración de OpenCart

anuncios y reporte de errores, sin embargo hay foros generales donde es posible realizar consultas a otros usuarios.

Puede observarse al explorar el repositorio de código que este se encuentra bien mantenido, recibe cambios y mejoras continuamente [20].

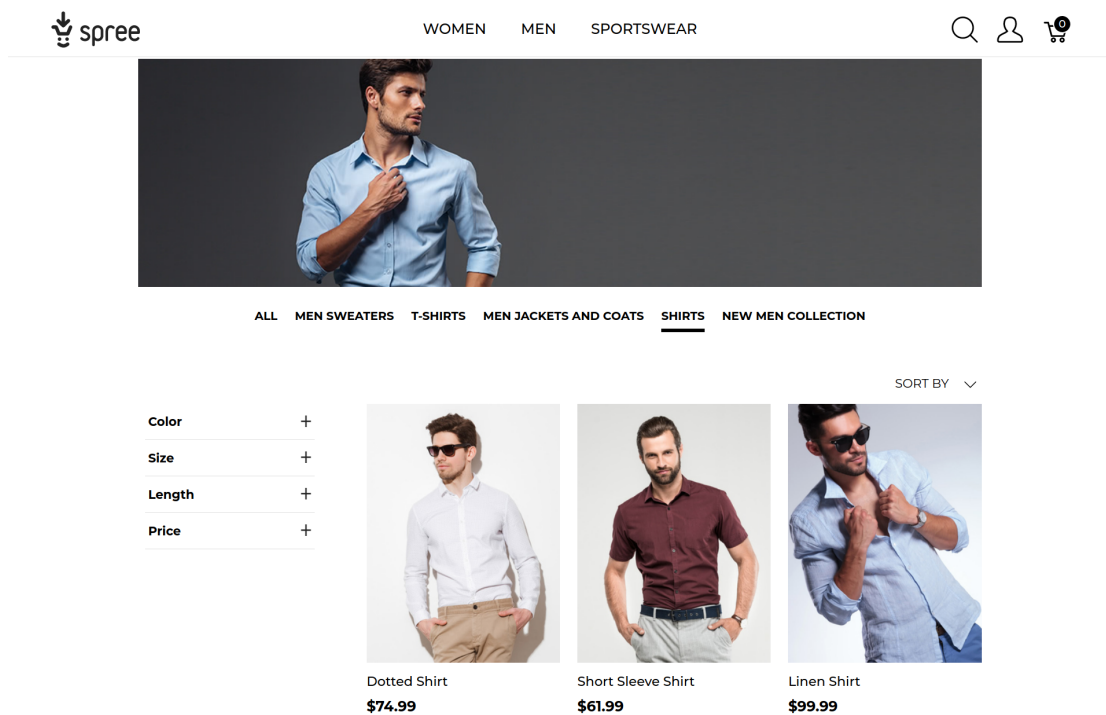


Figura 2.9: Interfaz de una tienda de ejemplo de Spree

## 2.1.2 Rails (Ruby)

### 2.1.2.1 Spree [29]

Spree es una solución joven que ofrece una interfaz moderna, pensada en los dispositivos móviles.

Existe una extensión multi-vendedor básica [30], que es semi-oficial y no tiene gran cantidad de documentación. A pesar de esto, sigue siendo necesario realizar la integración con InforedChile, y ajustar al mercado chileno, por lo que para usar este software como base habría que desarrollar extensiones escritas en Ruby.

*NOTA:* La versión de Spree 4.2, a ser lanzada oficialmente poco después del término del presente informe, se mejora e incluye oficialmente el soporte de múltiples vendedores, y se agregan distintas configuraciones para ajustar el sistema a países distintos a los norteamericanos [31].

**2.1.2.1.1 Comunidad y Actividad** Existen salas de chat donde se puede interactuar tanto con miembros de la comunidad, como con desarrolladores principales de la plataforma [32]. De esta forma es posible tener una conversación fluida con estos en caso de tener dudas para el desarrollo. El código recibe una gran cantidad de nuevas funcionalidades y arreglo de errores con rapidez [33].

COMPLETED AT	NUMBER	RISK	STATE	PAYMENT STATE	SHIPMENT STATE	EMAIL	TOTAL
2020-05-06 2:07 PM	R495097572	Safe	complete	paid	ready	customer@example.com	\$693.87
2020-05-06 2:06 PM	R069654044	Safe	complete	paid	ready	customer@example.com	\$211.91
2020-05-06 2:06 PM	R142583254	Safe	complete	paid	shipped	customer@example.com	\$58.99
2020-05-06 6:54 AM	R991064122	Safe	complete	balance due	pending	aga.kappauf@sparksolutions.co	\$28.99
2020-05-06 6:43 AM	R256242356	Safe	complete	paid	ready	spree@example.com	\$33.99
2020-05-02 5:02 AM	R054536085	Safe	complete	balance due	pending	micah@sparksolutions.co	\$175.97
2020-04-23 6:45 AM	R987654321	Safe	complete	failed	pending	spree@example.com	\$91.88

Figura 2.10: Interfaz de administración de Spree

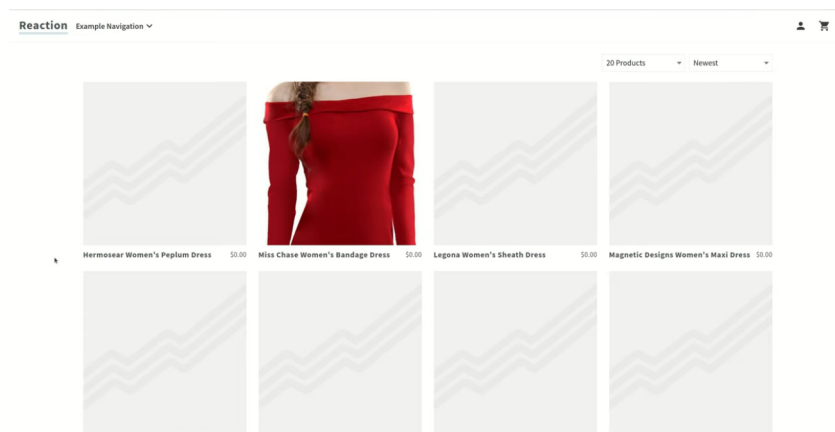


Figura 2.11: Interfaz de una tienda de ejemplo basada en Reaction

## 2.1.3 NodeJS (JavaScript)

### 2.1.3.1 Reaction [22]

Sistema moderno basado en NodeJS, React, y GraphQL, propiedad de Mailchimp desde abril del año 2020. Cuenta con un sistema de **plugins** escritos en JavaScript [26], siendo todas las funcionalidades provistas por alguno de los mas de 20 plugins base.

Reaction está pensado desde su núcleo con el concepto de múltiples tiendas [25], sin embargo las interfaces de ejemplo no le sacan provecho a dicha característica. Debido a esto, para solucionar la problemática tomando como base a Reaction, hay que modificar las interfaces para ofrecer productos de múltiples vendedores, además de desarrollar plugins de integración con InforedChile.

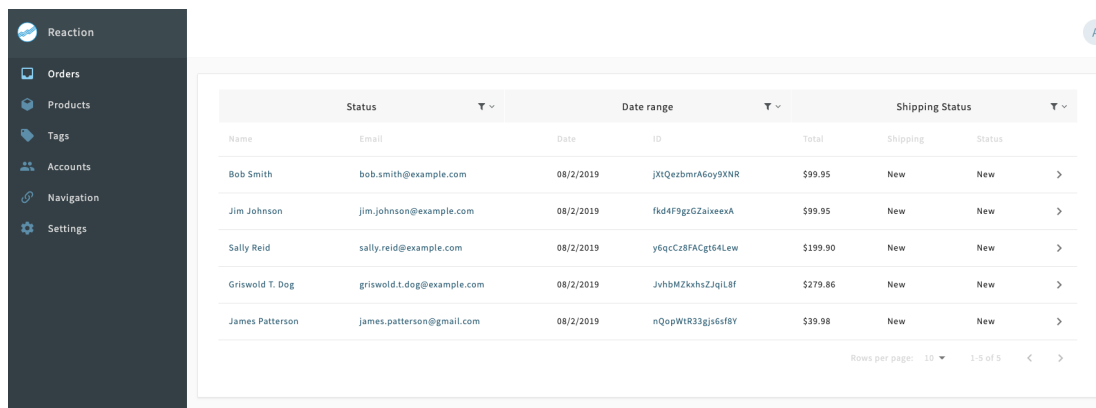


Figura 2.12: Interfaz de administración de Reaction

**2.1.3.1.1 Comunidad y Actividad** Reaction dispone un chat para interactuar con miembros de la comunidad de desarrollo, tanto independientes como del equipo [24], a los cuales se les puede hacer preguntas directamente en caso de necesitar orientación. Al revisar la página del proyecto en GitHub, donde se alojan los distintos plugins que conforman el sistema, se puede sacar la conclusión de que estos reciben mantención constante [23].

## 2.1.4 Django (Python)

### 2.1.4.1 Saleor [28]

Saleor se define como un sistema modular de E-commerce que utiliza Python, GraphQL, Django, y ReactJS.

Lamentablemente, no existe soporte para múltiples vendedores ni una buena documentación de su sistema de extensiones, por lo que habría que modificar de gran manera el funcionamiento interno para tener los resultados queridos.

**2.1.4.1.1 Comunidad y Actividad** Hay disponible un conjunto de salas de chat, separadas por tópico, para que la comunidad pueda comunicarse y apoyarse [27], además de un chat general donde pueden resolverse consultas rápidas [7]. El repositorio de código demuestra una buena actividad en cuanto a la corrección de errores y agregado de mejoras [8].

### 2.1.4.2 Oscar [16]

A diferencia de todas las alternativas ya mencionadas, Oscar no es una solución lista para comercio en línea, pero un Framework para la construcción de sitios de E-commerce basado en Django (Python). Este posee el concepto de “Partners” [17], quienes pueden administrar



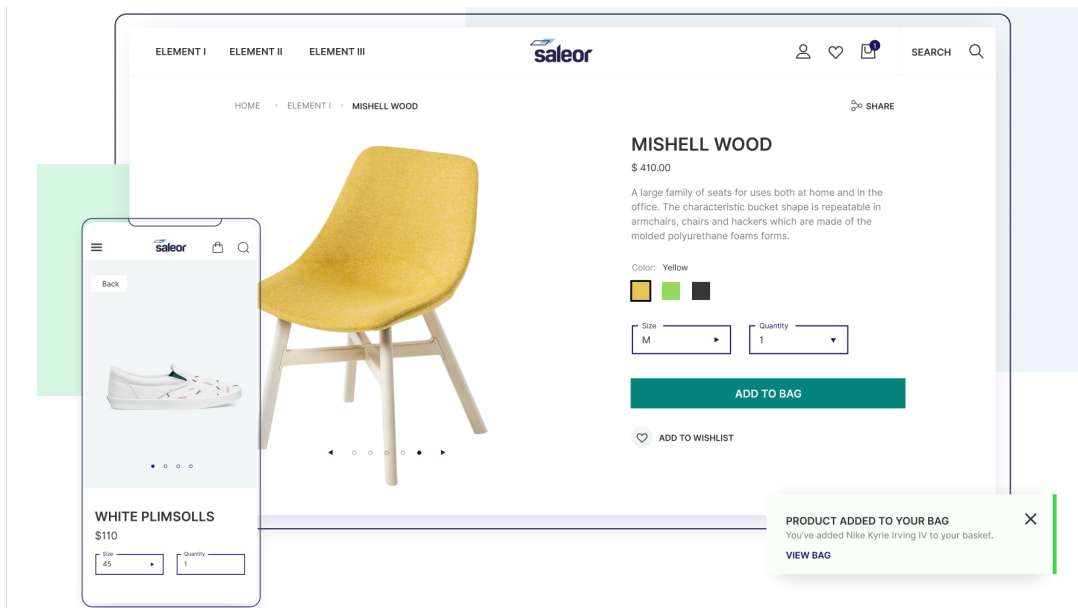


Figura 2.13: Interfaz de ejemplo de Saleor

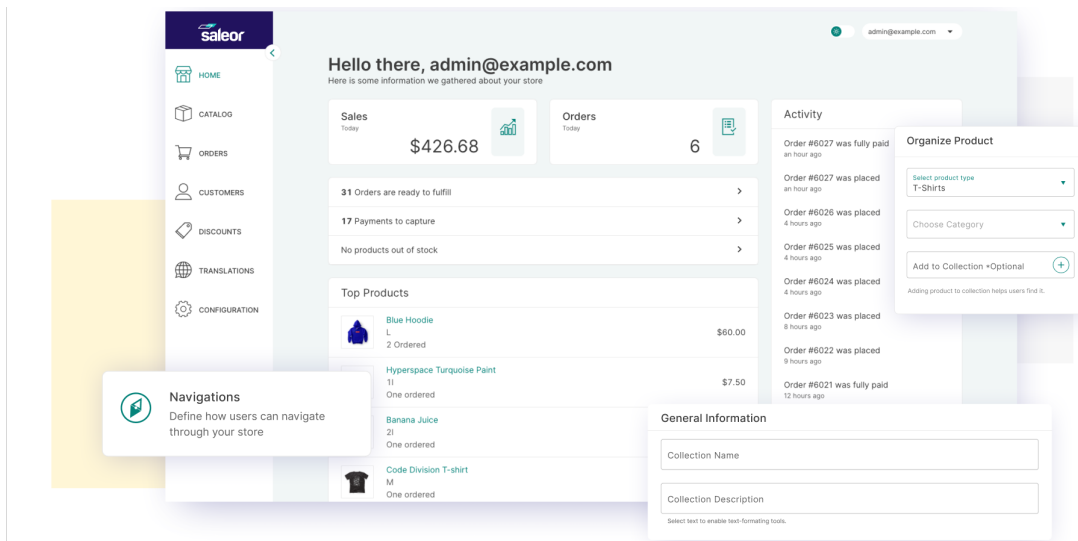


Figura 2.14: Interfaz de administración de Saleor

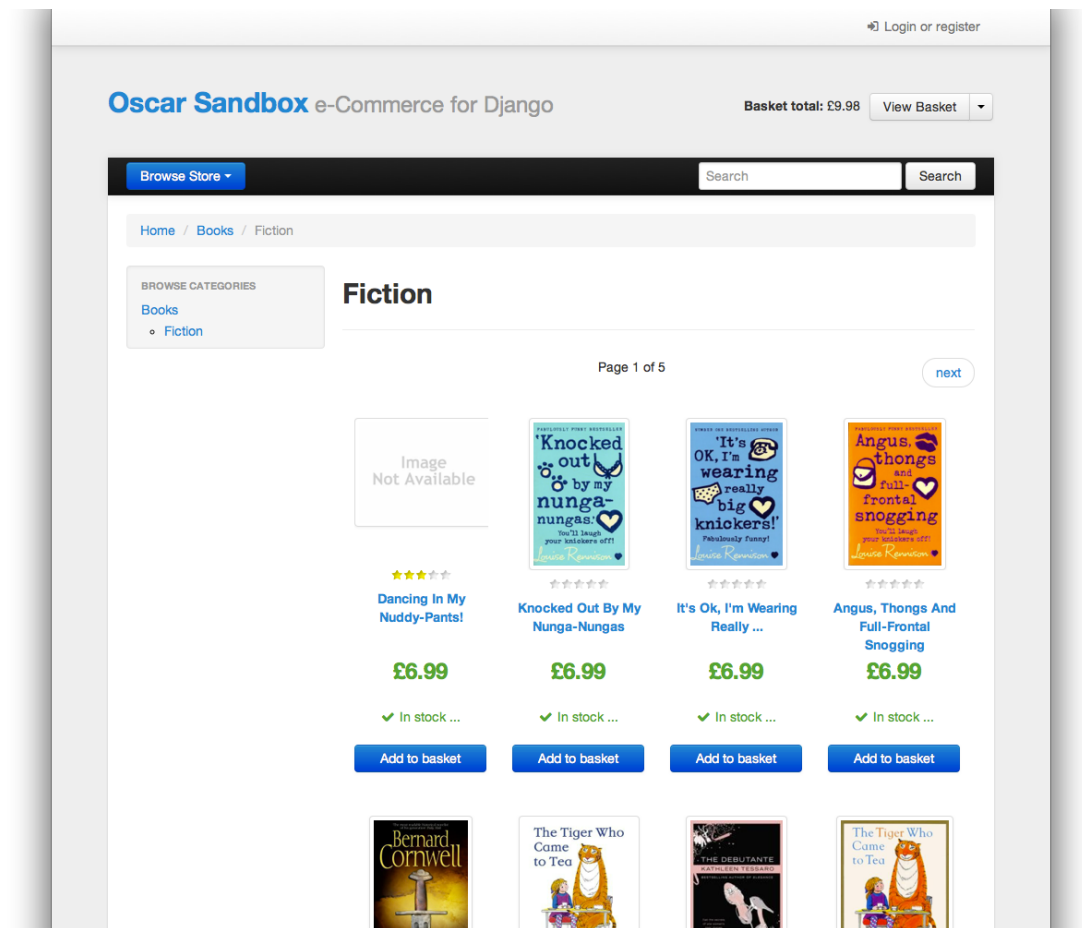


Figura 2.15: Interfaz de ejemplo de Oscar

productos, pero no tienen un catálogo separado. Dado que Oscar es un Framework para la creación de sistemas de E-commerce, existe una detallada documentación del funcionamiento interno, y es altamente extensible [15].

**2.1.4.2.1 Comunidad y Actividad** No existen medios de comunicación instantánea para los miembros de la comunidad, siendo la única forma de interacción una página de grupo de Google [13]. El código se encuentra mantenido, pero no se reciben mejoras frecuentemente, como se puede comprobar al revisar el repositorio [14].

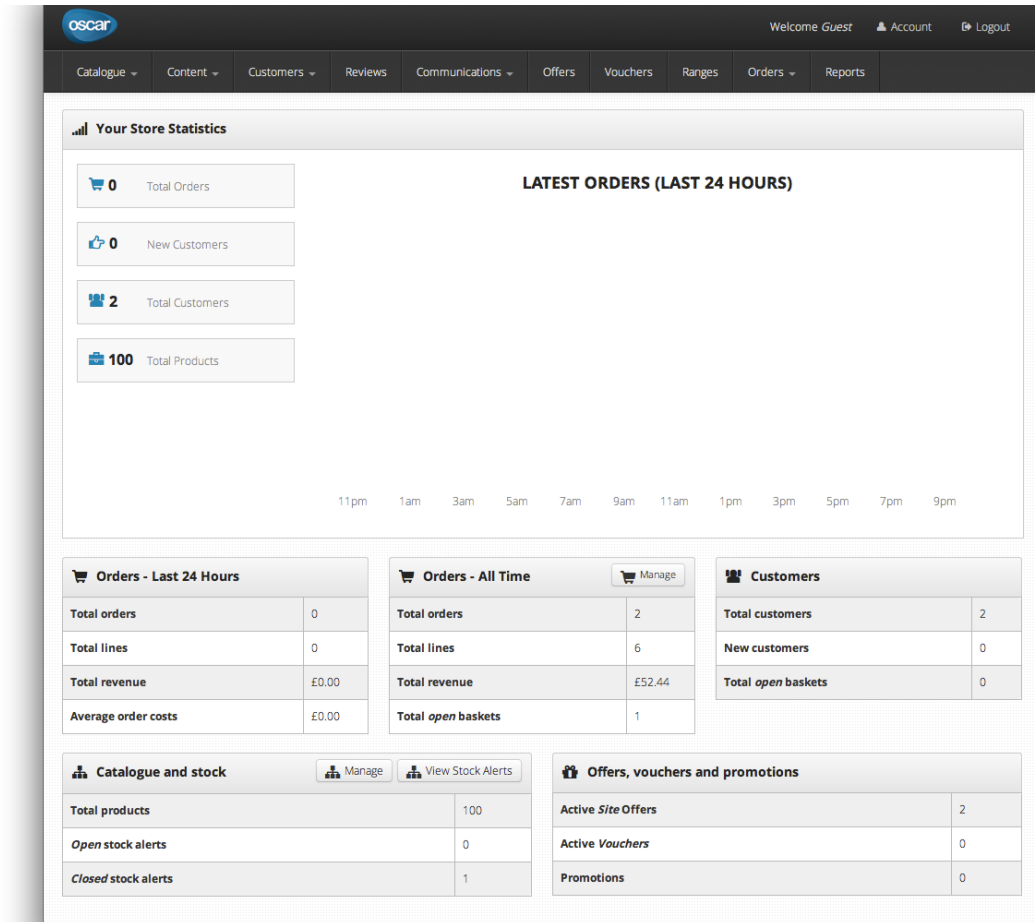


Figura 2.16: Interfaz de administración de Oscar

## 2.2 Elección de la base

Para la elección de la base, se eligen 4 puntos como comparación:

- Lenguaje y tecnologías Django-oscar, Saleor, Reaction y Spree están desarrollados de forma modular, utilizando tecnologías modernas. Spree no recibe la máxima calificación únicamente por preferencia en cuanto a lenguajes de programación.
- Adaptabilidad al contexto Django-oscar y Reaction pueden ser adaptados al contexto del problema, dado que fueron creados considerando el concepto de múltiples vendedores, o similar; en adición a estos, Spree y WooCommerce cuentan con extensiones que posibilitan añadir esta funcionalidad. Por otro lado, el resto ofrece extensiones de pago, que pueden ser reemplazadas por un desarrollo nuevo a realizar para este problema, con la excepción de Saleor que no cuenta con buena documentación para lograr un resultado exitoso.
- Actividad En este punto, el único que no obtiene puntuación máxima es Oscar, que a pesar de ser mantenido, no recibe corrección ni mejoras frecuentes.
- Comunidad Se califica como con buen apoyo a la comunidad a todos los proyectos que cuentan con un chat para su comunidad. Aquellos que ofrecen un foro o algo similar, son calificados como aceptables.

Alternativa	Lenguaje y tecnologías	Adaptabilidad al contexto	Actividad	Comunidad
[PHP] WooCommerce	✗	✓	✓	✓
[PHP] Magento	✗	✓	✓	✓
[PHP] PrestaShop	✗	✓	✓	✓
[PHP] OpenCart	✗	✓	✓	✓
[Rails] Spree	✓	✓	✓	✓
[NodeJS] Reaction	✓	✓	✓	✓
[Django] Saleor	✓	✗	✓	✓
[Django] Oscar	✓	✓	✓	✓

Cuadro 2.1: Comparativa de alternativas

La evaluación a las alternativas se encuentra resumida en la Tabla 2.1. Tomando esto en consideración, se puede concluir que Reaction es la mejor opción a tomar como base para diseñar una solución.

# Capítulo 3

## Propuesta

### 3.1 Modelo de Negocios

#### 3.1.1 Lean Canvas

LEAN CANVAS				
<b>Problema</b> 1. El segmento MiPymes tiene un importante déficit de presencia en internet y experiencia en comercio electrónico. 2. El desafío es potenciar la presencia en internet de las MiPymes, utilizando los recursos de InfordChile.	<b>Solución</b> Crear una plataforma multilateral de exposición de MiPymes en formato de directorio y marketplace online. Esta plataforma debe: 1) Permitir la comunicación entre las empresas del segmento MiPyme con sus potenciales compradores. 2) La generación de transacciones de compra desde el catálogo de oferta emitido por cada empresa.  <b>Métricas Claves</b> 1. Número de usuarios registrados. 2. Número de usuarios activos. 3. Número de visitas mensuales - anuales 4. Número de visitas simultáneas	<b>Propuesta de Valor</b>  <b>Propuesta de valor usuarios vendedores:</b> 1. Plataforma amigable. 2. Permite conectar con redes sociales y sitios webs propios. 3. Permite tener contacto directo con sus clientes.  <b>Propuesta de valor usuarios compradores:</b> 1. Acceder a una gran oferta de productos o servicios de MiPymes de forma fácil y directa.	<b>Segmento de Clientes</b> Empresas del segmento MiPymes que no realizan comercio electrónico o se estén integrando a este canal de ventas online  <b>Canales de Distribución y Comunicación</b> 1. Plataforma e-commerce. 2. Redes Sociales 3. Publicidad 4. Mailing	<b>Ventajas Competitivas</b> <b>¿Sobre quién?</b> 1. Compañías de plataformas E-commerce tradicionales. 2. Competidores más cercanos: Mercantil, Apaño tu Pyme, Mercado Libre. <b>¿Cómo?</b> 1. Ofrecer el servicio de plataforma e-commerce especializado para el segmento de MiPymes, a la que estas pueden acceder pagando una baja comisión en comparación al mercado. 2. Posicionar a MarketRed como una plataforma que permite a las MiPymes conectarse entre sí, y ser una vitrina atractiva para los usuarios compradores.
<b>Estructura de Costos</b> 1. Inversión inicial (Desarrollo de la Plataforma MarketRed: RR.HH- Herramientas de desarrollo). 2. Costos fijos y variables para la operación: RR.HH, actividades de Marketing y Ventas, actividades de soporte, entre otros.		<b>Flujos de Ingreso</b> 1. Ingresos por comisión - ventas de las MiPymes: comisión de un 1% por el total de transacciones realizadas por cada empresa de forma mensual. 2. Cobro por publicidad: \$8.990 CLP mensuales.		
<b>Atributos Complementarios</b> <b>Atributo 1: Crowdsourcing and data driven</b> Con el servicio de conexión B2B, la comunidad de usuarios vendedores podrá potenciar sus negocios y encontrar tanto a proveedores, como a empresas que cubran necesidades que deban externalizar, y se encuentren en posiciones económicas similares. De esta manera se estarán potenciando entre sí y compartiendo información.		<b>Atributo 2: Publicidad mediante valoración</b> Los anunciantes podrán tener mejores posiciones en la página inicial de la plataforma e-commerce, si se encuentran dentro de los más valorados por parte de los compradores de la plataforma. <b>Atributo 3: Software as a Service mediante plataforma web</b> Actúa con una plataforma web en la que abre nuevos servicios de venta e-commerce y conecta diferentes stakeholders del segmento Mipyme de la economía chilena. Al mismo tiempo entregará la opción de conexión en tiempo real entre usuarios vendedores y compradores desde la propia		

Figura 3.1: Lean Canvas

#### 3.1.1.1 Problema

El segmento de MiPymes se encuentra con una importante desventaja frente a las grandes empresas en cuanto a la presencia en línea: estas últimas en su mayoría cuentan con plataformas de E-commerce establecidas, mientras que menos de un tercio de las MiPymes realiza comercio electrónico -o sus canales E-commerce no ofrecen la confianza necesaria en los

consumidores para realizar compras-, por lo que poseen un importante déficit de experiencia en este ámbito.

Es por eso que nace la necesidad de conexión con este canal de ventas, por parte de este segmento de empresas, con una plataforma de Marketplace que ofrezca confianza tanto para usuarios de la plataforma como para clientes de esta, oportunidad de crecimiento de negocios y que esté al alcance de micro, pequeñas y medianas empresas.

#### **3.1.1.2 Solución**

Crear una plataforma multilateral de exposición de MiPymes en formato de directorio y Marketplace en línea. Esta plataforma debe:

1. Permitir la comunicación entre las empresas del segmento MiPyme con sus potenciales compradores.
2. La generación de transacciones de compra desde el catálogo de oferta emitido por cada empresa.

#### **3.1.1.3 Métricas claves**

1. Número de usuarios registrados.
2. Número de usuarios activos.
3. Número de visitas mensuales - anuales
4. Número de visitas simultáneas aproximadas durante día hábil.

#### **3.1.1.4 Segmento de clientes**

Empresas del segmento MiPymes que no realizan comercio electrónico o se estén iniciando en este método de ventas y que además comercialicen en su mayoría productos.

#### **3.1.1.5 Propuesta de valor**

##### **Propuesta de valor MiPymes (vendedores)**

1. Plataforma amigable
2. Permite conectar con redes sociales y sitios webs propios
3. Permite tener contacto directo con sus clientes

### **Propuesta de valor para usuarios compradores**

1. Acceder a una gran oferta de productos o servicios de MiPymes de forma fácil y directa.

#### **3.1.1.6 Canales de distribución y comunicación**

- Plataforma e-commerce.
- Redes Sociales
- Publicidad
- Mailing

#### **3.1.1.7 Ventajas competitivas**

##### **¿Sobre quién?**

- Compañías de plataformas E-commerce tradicionales.
- Competidores más cercanos: Mercantil, Apaño tu Pyme, Mercado Libre.

##### **¿Cómo?**

- Ofrecer el servicio de plataforma e-commerce especializado para el segmento de MiPymes, a la que estas pueden acceder pagando una baja comisión en comparación al mercado.
- Posicionar a MarketRed como una plataforma que permite a las MiPymes conectarse entre si, y ser una vitrina atractiva para los usuarios compradores.

#### **3.1.1.8 Estructura de costos**

1. Inversión inicial (Desarrollo de la Plataforma MarketRed: RR.HH- Herramientas de desarrollo).
2. Costos fijos y variables para la operación: RR.HH, actividades de Marketing y Ventas, actividades de soporte, entre otros.

### 3.1.1.9 Flujos de ingresos

1. Ingresos por comisión ventas de las MiPymes: comisión de un 1% por el total de transacciones realizadas por cada empresa de forma mensual.
2. Cobro por publicidad: \$8.990 CLP mensuales.

### Proyecciones de ingresos realista

Crecimiento esperado	Precio Promedio por Venta	% Crecimiento Precio-Venta	% de Ingreso por Comisión Cobrada	%Crecimiento de Comisión	Ingreso Por Venta Unitario
Año 1	\$10000	-	2,4%	-	240
Año 2	\$12300	23%	2,460%	2,5%	303
Año 3	\$15129	23%	2,583%	5,0%	391
Año 4	\$15582,87	3%	2,712%	5,0%	423
Año 5	\$16050,3561	3%	2,848%	5,0%	457

Cuadro 3.1: Proyección de ingresos por venta, escenario realista

### Flujo de ingresos realista

			Año 1	Año 2	Año 3	Año 4	Año 5
TOTAL INGRESOS	Suma	\$ miles	1.312	4.776	17.142	64.220	140.267
TOTAL EGRESOS	Suma	\$ miles	856	3.826	12.784	39.655	179.626
FLUJO DEL MES	Suma	\$ miles	52	130	522	2.678	-39.359
FLUJO ACUMULADO	Suma	\$ miles	456	1.406	5.764	30.329	40.942
FLUJO ANUAL	Suma	\$ miles	456	950	4.358	24.565	-39.359
FLUJO DESCONTADO	Suma	\$ miles	371	560	1.938	8.217	-10.546

Cuadro 3.2: Flujos de Efectivo, escenario realista



### Proyecciones de ingresos optimista

Crecimiento esperado	Precio Promedio por Venta	% Crecimiento Precio-Venta	% de Ingreso por Comisión Cobrada	%Crecimiento de Comisión	Ingreso Por Venta Unitario
Año 1	\$15000	-	2,4%	-	360
Año 2	\$18450	23%	2,460%	2,5%	454
Año 3	\$22693,5	23%	2,583%	5,0%	586
Año 4	\$23374,31	3%	2,712%	5,0%	634
Año 5	\$24075,53	3%	2,848%	5,0%	686

Cuadro 3.3: Proyección de ingresos por venta, escenario optimista

### Flujo de ingresos optimista

			Año 1	Año 2	Año 3	Año 4	Año 5
TOTAL INGRESOS	Suma	\$ miles	3.397	19.227	92.883	443.228	1.221.318
TOTAL EGRESOS	Suma	\$ miles	2.460	17.777	104.827	294.290	527.229
FLUJO DEL MES	Suma	\$ miles	111	310	-127	16.258	694.089
FLUJO ACUMULADO	Suma	\$ miles	937	2.388	-9.556	139.382	5.577.147
FLUJO ANUAL	Suma	\$ miles	937	1.450	-11.944	148.938	694.089
FLUJO DESCONTADO	Suma	\$ miles	755	742	-6.187	49.616	174.305

Cuadro 3.4: Flujos de Efectivo, escenario optimista

### **3.1.1.10 Atributos complementarios**

- **Atributo 1: Crowdsourcing and data driven**

Con el servicio de conexión B2B, la comunidad de usuarios vendedores podrá potenciar sus negocios y encontrar tanto a proveedores, como a empresas que cubran necesidades que deban externalizar, y se encuentren en posiciones económicas similares. De esta manera se estarán potenciando entre si y compartiendo información.

- **Atributo 2: Publicidad mediante valoración**

Los anunciantes podrán tener mejores posiciones en la página inicial de la plataforma e-commerce, si se encuentran dentro de los más valorados por parte de los compradores de la plataforma.

- **Atributo 3: Software as a Service mediante plataforma web**

Actúa con una plataforma web en la que abre nuevos servicios de venta e-commerce y conecta diferentes stakeholders del segmento MiPyme de la economía chilena. Al mismo tiempo entregará la opción de conexión en tiempo real entre usuarios vendedores y compradores desde la propia plataforma.

## **3.2 Requisitos del Sistema**

### **3.2.1 Requisitos Funcionales**

- RF1 El sistema debe ofrecer un Marketplace de productos. Esto es: se presentan distintos productos a un precio determinado.
- RF2 El sistema debe ser accesible para todas las empresas registradas en el sitio principal.
- RF3 Cada empresa debe poder administrar su y solo su catálogo de productos. Estos catálogos pueden verse en la página de forma independiente.
- RF4 El sistema debe permitir el registro de usuarios, los que pueden explorar y buscar productos en los catálogos.
- RF5 El sistema debe diferenciar entre usuarios y empresas, con privilegios distintos en el sistema.
- RF6 El sistema debe funcionar de manera independiente a la página principal de InfoRed-Chile.
- RF7 El sistema debe poder interactuar con el indexador del motor de búsqueda del sitio principal, pudiendo realizar actualizaciones en este de acuerdo a los productos que se agreguen.
- RF8 El sistema debe ser capaz de incrustar resultados de búsqueda de productos en el buscador del sitio principal.
- RF9 El sistema debe ser capaz de incrustar una lista de productos del catálogo de cierta empresa, al ver detalles de dicha empresa en el sitio principal.
- RF10 Los usuarios deben poder agregar productos de un catálogo a un carro de compras, y solicitar los productos o servicios.
- RF11 Estos carros de compra de los usuarios deben ser independientes para cada catálogo de empresas distintas.
- RF12 El sistema debe permitir al usuario pagar por los productos y/o servicios, una vez que la empresa haya confirmado la disponibilidad.

### **3.2.2 Requisitos no Funcionales**

- RNF1 El diseño estético del sistema debe concordar con el diseño de la página principal.
- RNF2 El sistema debe funcionar con el orden de miles de usuarios diarios.

RNF3 Fácil de usar por los clientes y de administrar.

RNF4 Los pagos en el sistema deben realizarse de forma segura.

### 3.2.3 Requisitos de Interfaces

La Tabla 3.5 muestra la lista de eventos externos a los que el sistema responde. La primera columna es el nombre del evento y la segunda su descripción. El “iniciador” es el componente externo al sistema que inicia el evento. Los parámetros son los datos asociados al evento. La respuesta es el nombre de una respuesta, cuya descripción está en la Tabla 3.6.

Respuesta	Descripción	Parámetros
Mostrar resultado de la búsqueda	Una vez realizada la búsqueda el sistema identificará las palabras claves de esta, y generará un resultado de empresas y productos asociados a estas y lo que haya buscado el usuario	Palabra clave, palabra de búsqueda
Cargar página del producto	Una vez que el producto es seleccionado por el posible usuario del sistema, se busca en la base de datos la información necesaria y se carga la página de dicho producto	Identificador del producto en particular
Cargar página de la empresa o del servicio	Al hacer click sobre el resultado de la búsqueda, correspondiente a una empresa o servicio	Identificador de la empresa o del servicio
Agregar el producto a un carro de compra	Una vez visto el producto, será posible agregarlo a un carro de compra, el cual permitirá realizar una cotización para una eventual compra del producto	Identificador del producto, la empresa y del usuario del sistema
Generar una cotización	El usuario puede generar una cotización luego de agregar la cantidad de productos que estime convenientes al carrito de compra de la respectiva tienda. Cada usuario tiene un carro distinto por casa tienda	Identificador del producto, la empresa y del usuario del sistema

Cuadro 3.6: Respuestas del sistema

<b>Evento</b>	<b>Descripción</b>	<b>Iniciador</b>	<b>Parámetros</b>	<b>Respuesta</b>
Hacer click sobre el botón de búsqueda	El posible usuario del sistema realiza una búsqueda de un producto	El posible usuario del sistema	Palabra clave, palabra de búsqueda	Mostrar resultado de la búsqueda
Hacer click sobre un producto	Un posible usuario selecciona uno de los productos mostrados luego de realizar una búsqueda	El posible usuario del sistema		Cargar página del producto
Hacer click sobre una empresa	Un posible usuario selecciona una de las empresas o servicio mostrados luego de realizar una búsqueda	El posible usuario del sistema		Cargar página de la empresa o el servicio
Hacer click en agregar producto a la canasta	El posible usuario puede hacer click en la opción de agregar un producto a la canasta, lo que le permitiría realizar una cotización	El posible usuario del sistema		Agregar el producto a un carro de compra
Hacer click en generar cotización	El posible usuario puede generar una cotización a partir del producto o 'el servicio'	El posible usuario del sistema		Generar una cotización

Cuadro 3.5: Eventos Externos

### 3.2.4 Requisitos de Ambiente

#### 3.2.4.1 Hardware de Desarrollo

HD1 Computador x86\_64 o Arm

HD2 Servidor

#### 3.2.4.2 Software de Desarrollo

SD1 Sistema Operativo Unix de 64-bits.

SD2 Sistema Operativo Windows de 64-bits.

SD3 Hipervisor (de software) para alojar máquinas virtuales

SD4 Interprete de JavaScript.

SD4 Interprete de Python.

SD5 Tiempo de ejecución NodeJS.

SD6 Tiempo de ejecución para contenedores.

SD7 Entorno de desarrollo integrado.

### 3.2.5 Perfiles de Usuario

<b>Perfil</b>	<b>Socioeconómico y cultural</b>	<b>Ocupacional</b>	<b>Etario</b>	<b>Características físicas, fisiológicas, psicológicas.</b>	<b>Otros</b>
<b>Administrador de empresa</b>	Clase Media-Alta	Administrador de empresa del segmento MyPymes	Mayor a 25 años	Cualquiera	
<b>Posible Usuario</b>	Clase Media	Cualquiera	Mayor a 15 años	Cualquiera	
<b>Administrador</b>	Clase Media	Administra el sitio web del sistema	Entre 25 y 65 años	Cualquiera	

Cuadro 3.7: Perfiles de Usuario

## 3.3 Arquitectura del Sistema

### 3.3.1 Diagrama de Contexto

En el diagrama de contexto (Figura 3.2), el sistema está graficado como una caja negra, junto a los principales entes externos al sistema que interactúan con él. Las flechas que unen los entes externos con el sistema indican flujos de datos desde o hacia el sistema.

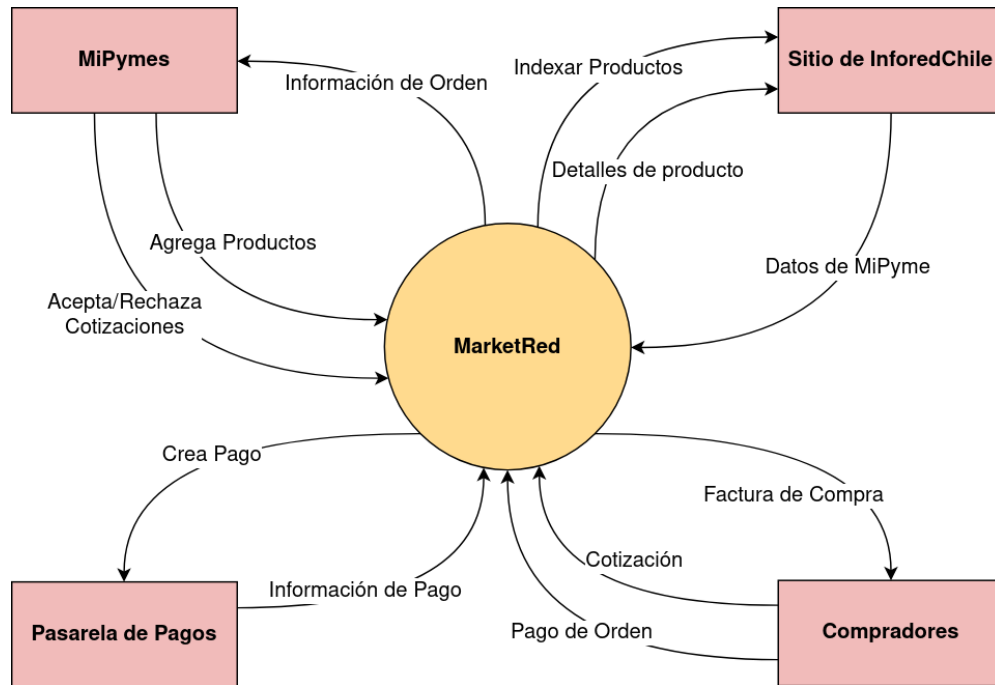


Figura 3.2: Diagrama de contexto del sistema

### 3.3.2 Diagrama de Arquitectura

La Figura 3.3 muestra los componentes internos del sistema y sus principales interacciones.

Dado que 'MarketRed API' está compuesto por una gran cantidad de complementos con diversas interacciones, se listan los que deberán ser creados específicamente para esta solución (con borde grueso), junto a los que interaccionan fuertemente con estos (con borde delgado).

Además, hay 2 interfaces o 'frontends'. Por un lado, la interfaz denominada 'Tiendas' es por la cual se muestran públicamente los productos, y donde se pueden realizar compras. Por otro lado, 'Admin' es la interfaz donde solo pueden acceder las MiPymes, y donde se publican los productos.

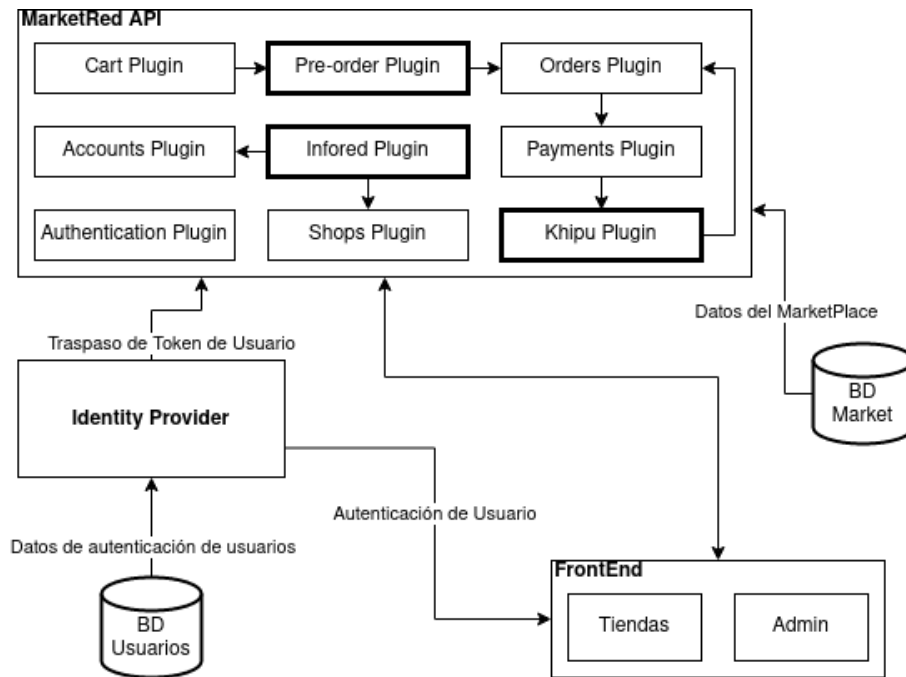


Figura 3.3: Diagrama de arquitectura del sistema

### 3.3.3 Enumeración de Componentes

La Tabla 3.8 muestra los componentes de la Figura 3.3. Por cada componente se entrega un breve párrafo descriptivo de su propósito además de la sección en donde se especifica el componente en detalle.

Componente	Propósito	Sección
MarketRed API (MA)	El componente principal que expone la API a ser consumida por FE. Maneja los datos y la lógica de negocios. A pesar de estar representado como un solo componente, la funcionalidad interna está separada en múltiples PlugIns	3.5.1
Frontend (FE)	Interfaces de usuario con las cuales usuarios y empresas interactúan, consumiendo los datos de la API	3.5.2
Identity Provider (IP)	Maneja los datos de autenticación, tanto de usuarios como empresas, y las sesiones iniciadas. Cuando se inicia una sesión en FE, le envía información de esto tanto a MA como FE para que luego puedan interactuar entre ellos de forma autenticada.	3.5.3

Cuadro 3.8: Componentes de la arquitectura del sistema



### 3.3.4 Matriz de Requisitos Funcionales y Componentes

La tabla 3.9 muestra qué componentes del sistema implementan qué requisitos funcionales. Las celdas con una X indican que la componente de la columna respectiva ayuda a implementar el requisito funcional de la fila respectiva.

Se incluye en la matriz el componente ‘Incrustador de Resultados’ (IR), dado que implementa dos de los requisitos iniciales. Este componente agrega a la página de InforedChile datos de distintos productos, consumiendo los datos de la API; sin embargo, InforedChile decidió crear dicho componente por su lado, para no ceder acceso a su sistema actual.

	MA	IR	FE	IP
RF1	X			
RF2				X
RF3	X		X	
RF4			X	X
RF5	X			X
RF6	X		X	
RF7	X			
RF8		X		
RF9		X		
RF10			X	
RF11			X	
RF12	X		X	

Cuadro 3.9: Matriz de requisitos funcionales y componentes

## 3.4 Gestión de Riesgos

### 3.4.1 Supuestos

- Los datos de autenticación de usuario son actualmente guardados de forma segura (con **hashes**).
- El servidor que aloje las máquinas virtuales o contenedores tiene un acceso seguro y actualizaciones de seguridad constantes.

### 3.4.2 Dependencias

- Acceso a datos de empresas/MiPymes registradas en la plataforma de InforedChile
- Acceso a servidores, bases de datos y entornos de trabajo propios de InforedChile

- Registro del sitio en alguna plataforma de pago (Flow, TransBank, PagoFácil, etc)

### 3.4.3 Restricciones

- Restricciones de tiempo, al tener que planificar lo posible dentro del PMM.
- Uso de datos de empresas en el formato actual de las bases de datos de InforedChile.

### 3.4.4 Riesgos

Riesgo	Medida de Mitigación
Los datos actuales de autenticación se guardan en texto plano.	Modificar los datos actuales de InforedChile, para agregar y utilizar datos derivados (e.g. hashes).
Que las empresas/MiPymes no estén dispuestas a ser parte de la nueva plataforma, que no quieran ofrecer sus productos en la plataforma	Hacer campañas o generar incentivos para las empresas que ofrezcan sus productos en la plataforma

## 3.5 Diseño de Módulos del Sistema.

### 3.5.1 Componente [MA]

#### 3.5.1.1 Definición del Componente

<b>Propósito</b>	Manejar los datos y la lógica de negocios, exponiendo una API a ser consumida por el resto de los módulos.
<b>Alcance</b>	El componente principal que presenta y muta los datos de acuerdo a las solicitudes recibidas.
<b>Dependencias</b>	Es necesario el funcionamiento de la base de datos.
<b>Supuestos</b>	Se ejecuta sobre un kernel Linux para arquitectura x86_64.
<b>Restricciones</b>	Ninguna.
<b>Estructura General</b>	A pesar de estar representado como un solo componente, la funcionalidad interna está separada en múltiples complementos. Los datos son expuestos a través de una API GraphQL (ver detalles en el Anexo A), mediante la cual se pueden recibir consultas o mutaciones. Los complementos pueden interactuar internamente entre ellos por medio de estas mismas mutaciones. En adición a los complementos por defecto del sistema Reaction, hay 3 principales que se deben adicionar para el funcionamiento de esta solución: <b>Pre-order Plugin</b> , <b>Infored Plugin</b> y <b>Khipu Plugin</b> .

#### 3.5.1.2 Pre-order Plugin

Este Plugin es el encargado de generar y enviar la cotización a la empresa a la que se encuentran asociados los productor del carrito de compra. Para ello utiliza los siguientes datos del usuario comprador: Nombre, dirección, correo electrónico, el método de envío seleccionado y el método de pago de la eventual compra. Además de esto, aquellos productos que se encuentren en el carro, que sean exclusivos de la tienda a la que se está realizando la cotización, se recogen los siguientes datos: Nombre del producto, marca, variación, cantidad, precio, costos de envíos, además del precio total de la orden y los comentarios realizados por el comprador. Con todos estos datos se envía la cotización a la empresa que es la encargada de aceptarla y generar la orden respectiva de venta, o de lo contrario, rechazarla. En ambos casos el comprador recibe una notificación con la orden y la posibilidad de pagar o una eventual cancelación de la cotización con el comentario de las razones por parte de la empresa.

#### 3.5.1.3 Infored Plugin

Dado que hay ciertos datos relativos a las MiPymes que son manejados en el sistema de InforedChile, como nombres y datos de contacto, y que son necesarios conocer en MarketRed,

se diseña este complemento que tiene el propósito de mantener dichos datos sincronizados. Para esto, se planea ofrecer una mutación GraphQL, la cual será utilizada por el sistema InforedChile cada vez que haya cambios en los datos de cierta MiPyme.

#### 3.5.1.4 Khipu Plugin

Para gestionar los pagos en MarketRed se elige la pasarela de pagos Khipu.

Khipu gestiona pagos por medio de tarjetas bancarias chilenas y transferencias, ofreciendo una API simple y limpia.

Dado que los pagos no son inmediatos, y tienen un plazo para completarse, este complemento además tiene la función de cambiar el estado de la orden una vez la operación de pago haya terminado.

El flujo general para realizar estas operaciones se encuentra ilustrado en la Figura 3.4.

### 3.5.2 Componente [FE]

#### 3.5.2.1 Definición del Componente

<b>Propósito</b>	Exponer a los usuarios las interfaces gráficas para interactuar con el sistema
<b>Alcance</b>	Procesa las distintas solicitudes de los usuarios, llamando las mutaciones correspondientes de la API. Recibe de parte de IP la información de autenticación en caso que corresponda.
<b>Dependencias</b>	Depende de que la MA exponga la API para consumirla, y de IP para realizar acciones que requieran autenticación.
<b>Supuestos</b>	Ninguno
<b>Restricciones</b>	Ninguna
<b>Estructura General</b>	Este componente está separado en 2 'frontends' distintos: uno para la visualización de las tiendas, y otro para la administración de estas (que requiere autenticación). Ambos usan apollo-client para la interacción con la API de MA y el manejo de estado, usando React para el renderizado de los elementos visuales.

### 3.5.3 Componente [IP]

#### 3.5.3.1 Definición del Componente

<b>Propósito</b>	Manejar los datos de autenticación de los usuarios y MiPypmes.
------------------	--

<b>Alcance</b>	Ante la petición de inicio de sesión de FE, autentica el usuario, retornando a FE un token. FE utiliza este token con MA, quien finalmente comprueba la validez con IP.
<b>Dependencias</b>	Es necesario el funcionamiento de la base de datos con los datos de ingreso de los usuarios.
<b>Supuestos</b>	El canal de comunicación IP-MA es seguro.
<b>Restricciones</b>	Ninguno
<b>Estructura General</b>	Este componente se divide en: 1) Ory Hydra, un servidor que gestiona el flujo de autenticación OpenID y 2) Un proveedor de identidad, que maneja los datos de inicio de sesión de los usuarios comunes, o interactúa con el sitio de InforedChile para autenticar a las MiPymes.

Este módulo utiliza el sitio actual de InforedChile como proveedor de usuarios para los clientes de InforedChile que podrán administrar sus empresas. Debe manejarse de forma distinta el acceso cuando el cliente es el “Frontend” de administración, donde solo MiPymes pueden acceder, a cuando es el cliente de la tienda.

Esto se logra detectando en el proveedor de usuarios MarketRed cuando la solicitud proviene del “cliente de administración”, y redirigiendo en ese caso al agente de usuario al sitio de InforedChile. Adicionalmente, se puede ofrecer una opción de “Inicio de sesión con cuenta InforedChile” en el “cliente de la tienda”, para que realice igualmente esta redirección.

El flujo de este procedimiento se ilustra en la Figura 3.5.

De esta forma se separa el manejo de las datos de inicio de sesión para empresas -cuentas administradas por InforedChile- de las cuentas de los usuarios que simplemente pueden comprar -administradas por el sistema de MarketRed-. Utilizando este flujo de autenticación, la interacción de los distintos componentes con el Proveedor de Identidad (Hydra+Proveedor de Usuarios) puede modelarse de la forma ilustrada en la Figura 3.6

## 3.6 Diseño de Interfaces del Sistema.

### 3.6.1 Modelo de Navegación.

Un modelo de navegación permite entender el contexto en el que se va a crear la estructura, de esta forma se puede observar el flujo de interacciones que debe realizar un usuario para acceder a las distintas funcionalidades de la plataforma.

La Figura 3.7 muestra los pasos que debe realizar un usuario que desea comprar un producto dentro del Marketplace. Desde realizar la cotización, la espera de la confirmación y la emisión de la orden, hasta la posibilidad de pagar dicha orden o cancelarla.

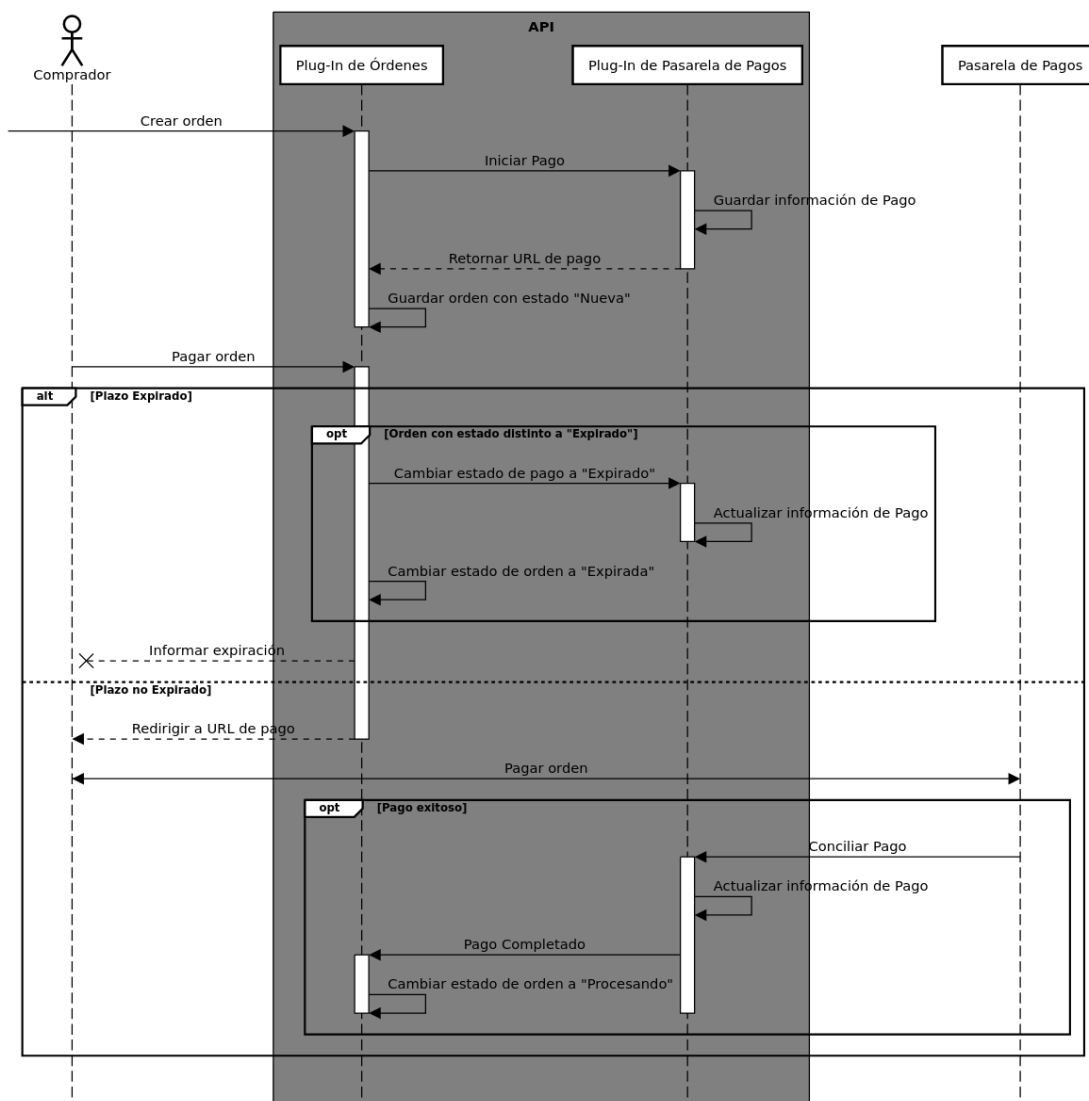


Figura 3.4: Flujo de pago.

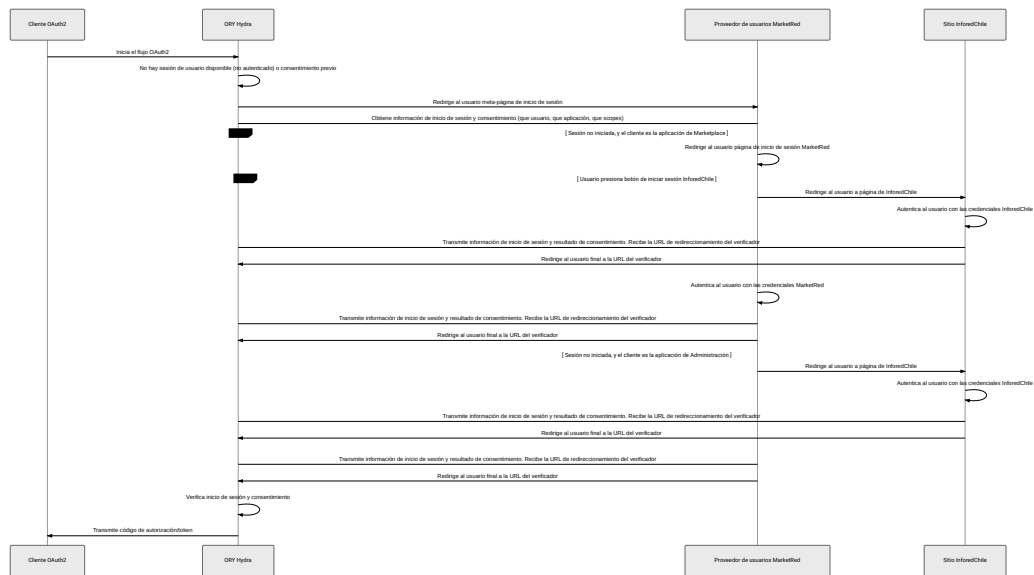


Figura 3.5: Flujo de inicio de sesión.

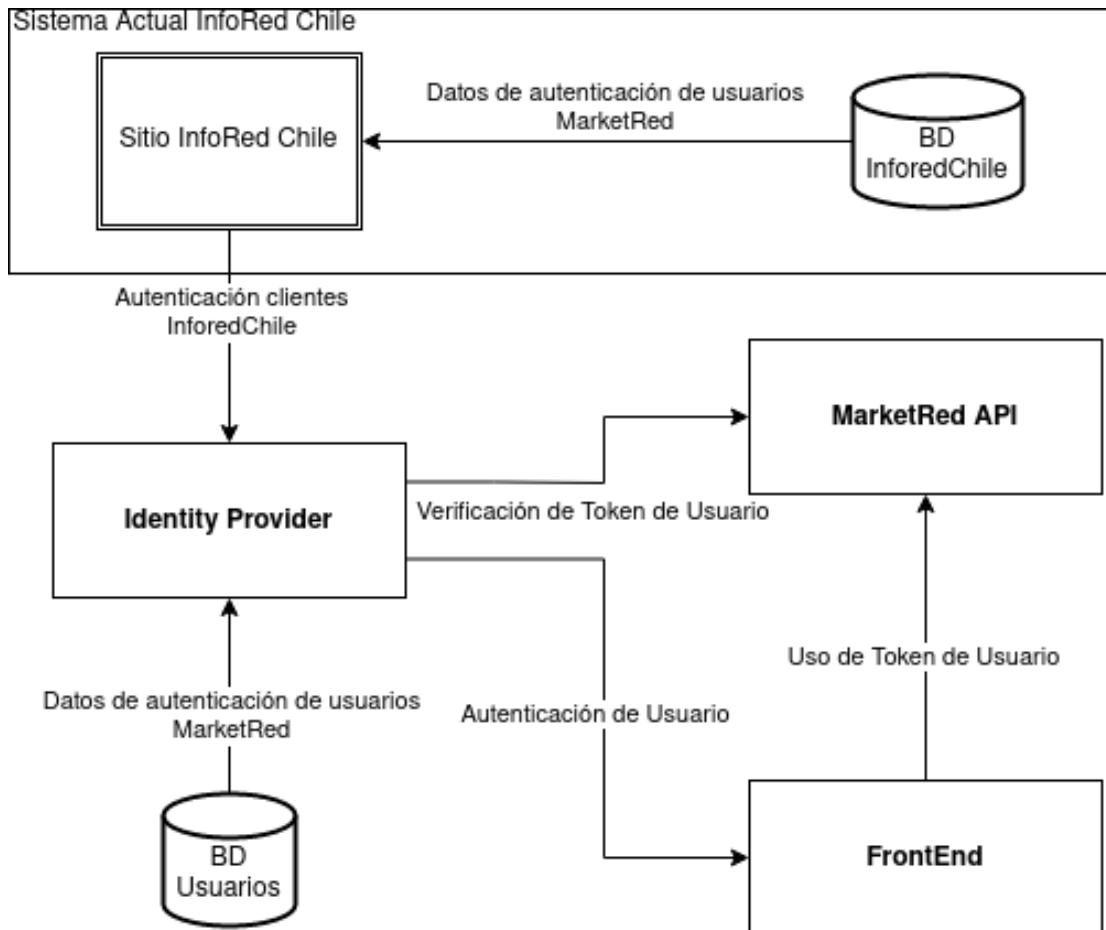


Figura 3.6: Diagrama del componente de autenticación.

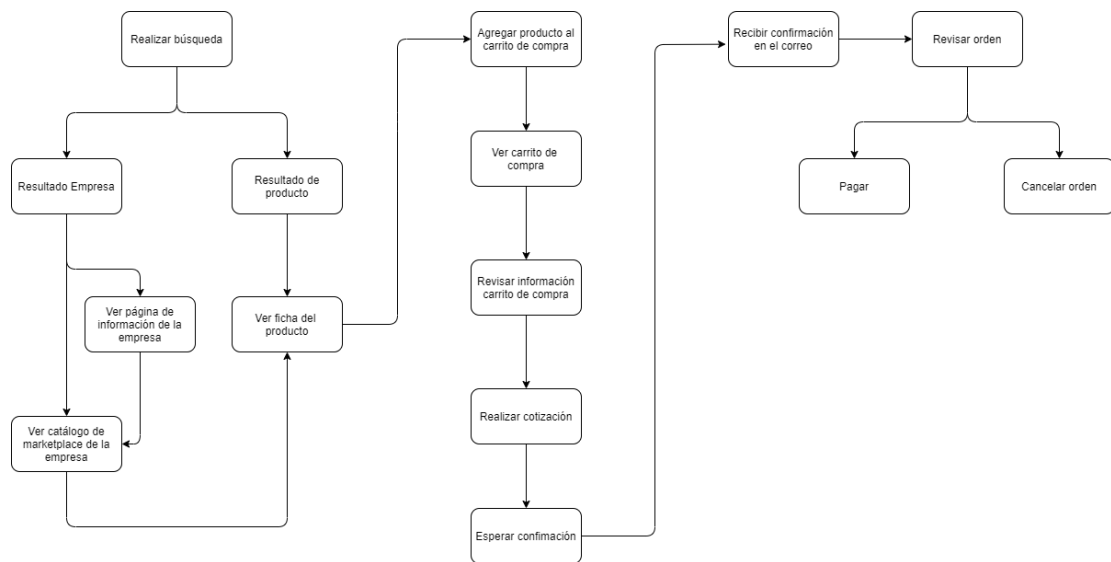


Figura 3.7: Modelo de navegación de compra.

En caso de que el usuario no se encuentre logueado en el sistema cuando reciba la confirmación de la orden o desee ver las cotizaciones que ha enviado y se encuentran pendientes de confirmación, puede iniciar sesión en la página y acceder a estas opciones desde su cuenta de usuario, dichas interacciones se encuentran diagramadas en la Figura 3.8

Finalmente, podemos ver del lado de la empresa, en la Figura 3.9, las opciones que tiene posterior a iniciar sesión dentro de la plataforma.



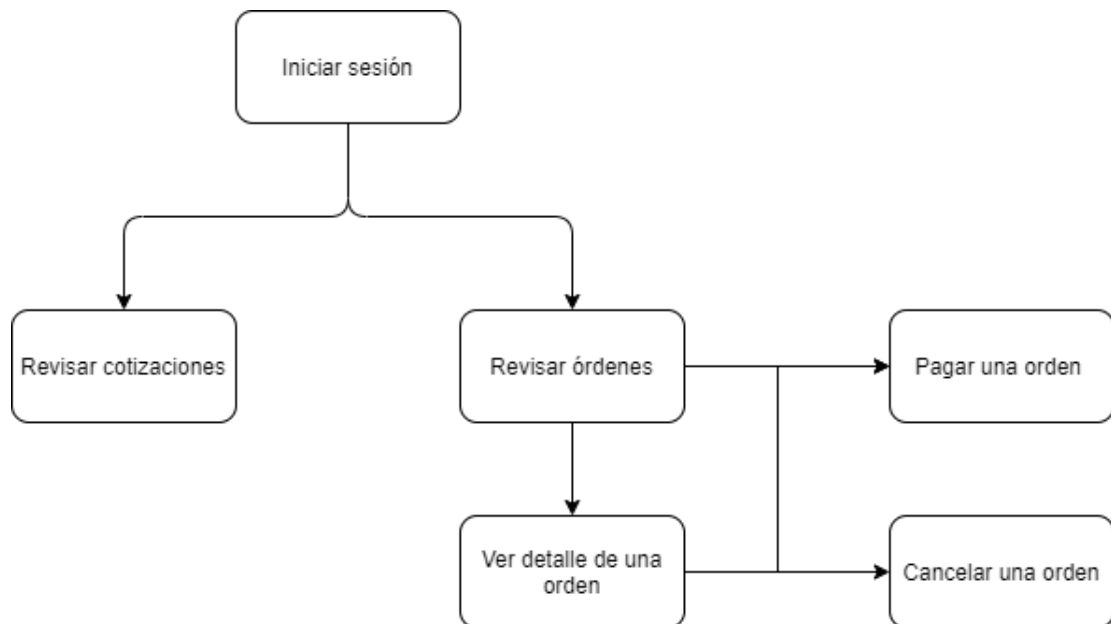


Figura 3.8: Modelo de navegación de usuario para revisar cotizaciones y órdenes de Infored.

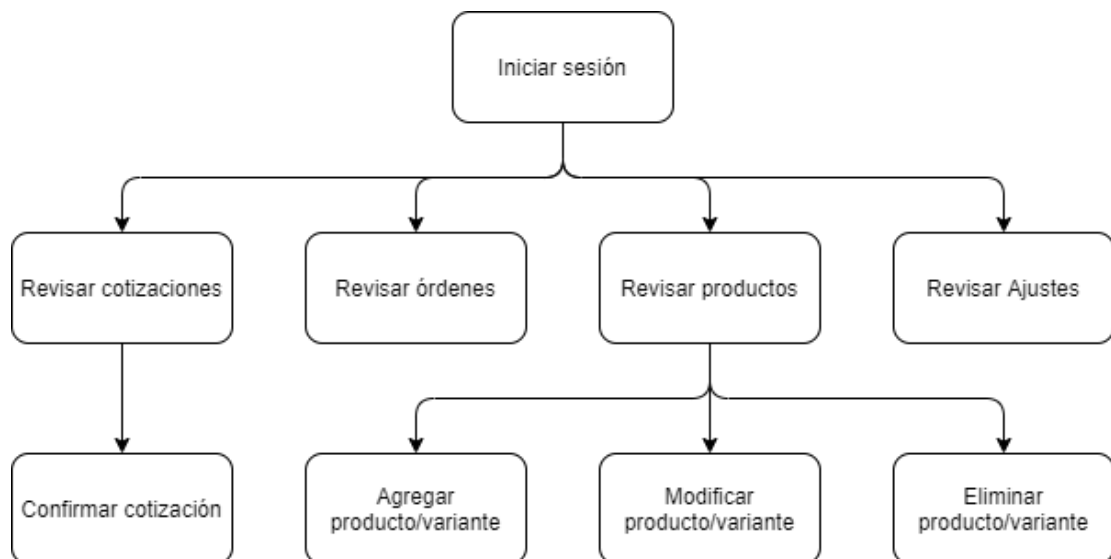


Figura 3.9: Modelo de navegación de una empresa registrada en Infored.

### 3.6.2 Diseño de Interfaces Usuarías

El diseño de interfaces usuarias permite realizar un enfoque de la plataforma basada en la experiencia de usuario y su interacción, de esta forma, ayuda a que las aplicaciones sean más atractivas y además, a hacer que la interacción con el usuario sea lo más intuitiva posible.

A continuación se presenta el diseño de las interfaces usuarias que se espera desarrollar para la plataforma de Marketplace del sitio InforedChile.



Figura 3.10: Página de inicio InforedChile

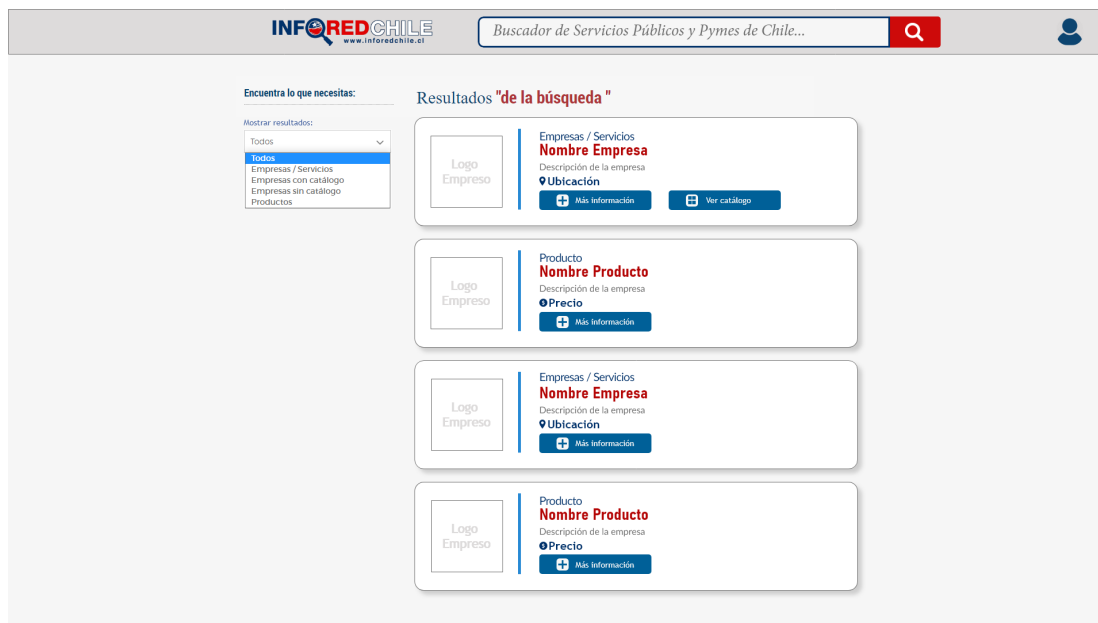


Figura 3.11: Página de resultados de búsqueda



Figura 3.12: Catálogo de empresa / servicio



Figura 3.13: Ficha de producto

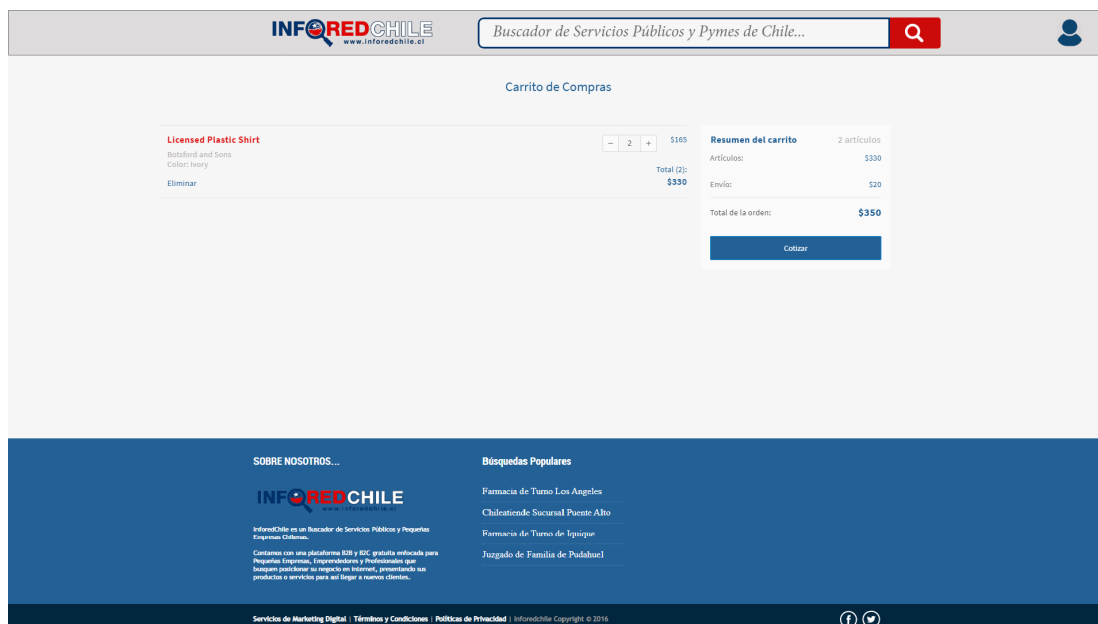


Figura 3.14: Carrito de compra





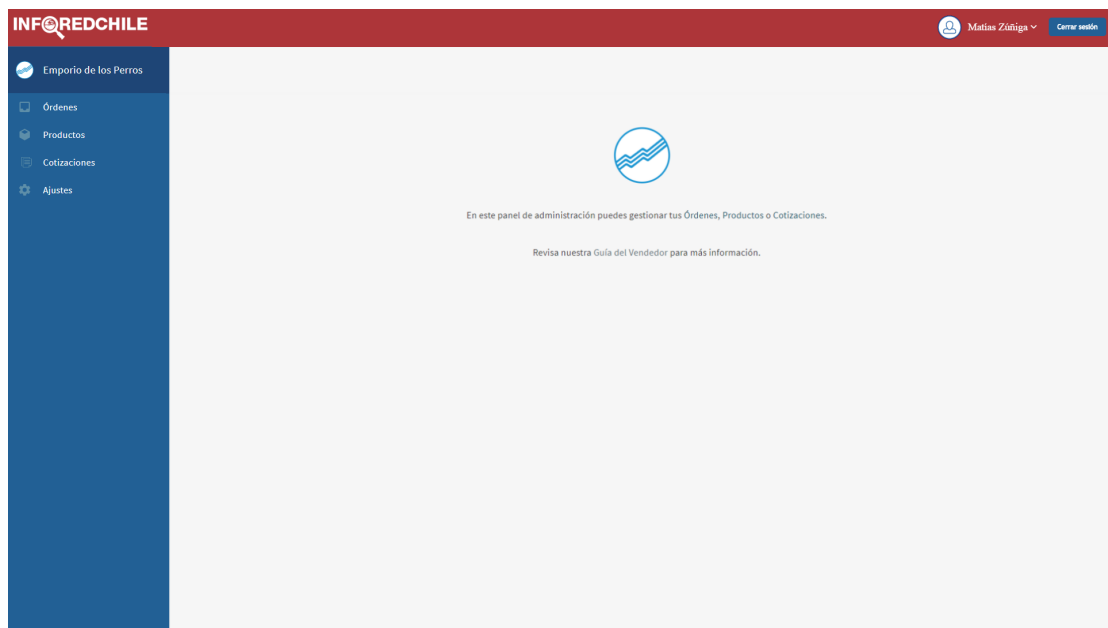


Figura 3.19: Perfil de empresa al iniciar sesión (propuesta)

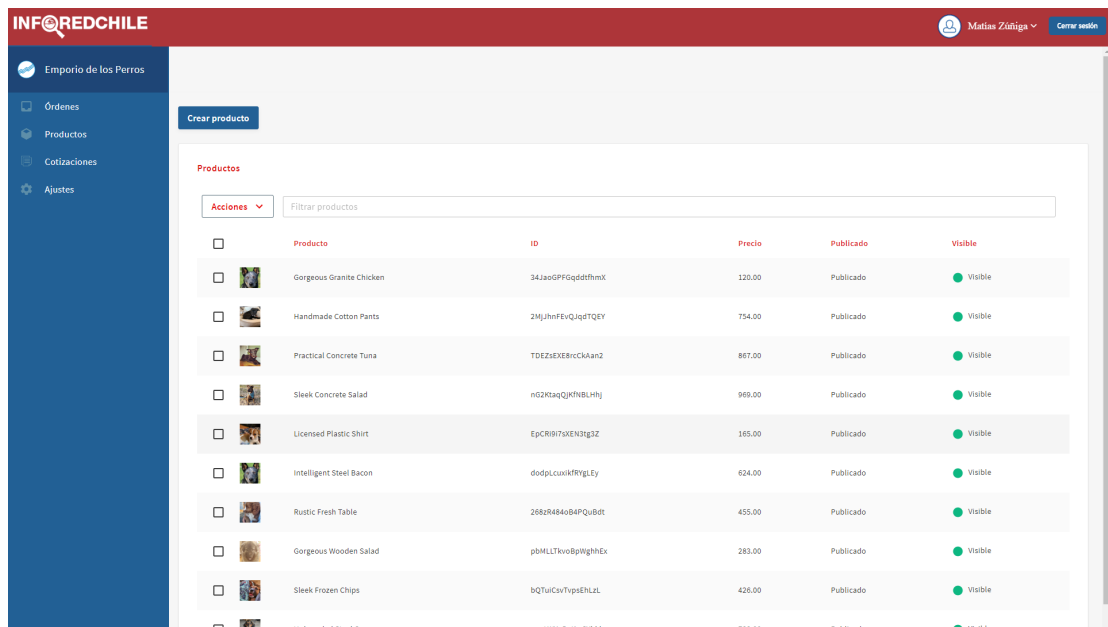


Figura 3.20: Página de productos de la empresa

**INFOREDCHILE** Matias Zúñiga Centro sesión

**Emporio de los Perros** ← Productos Publicado

**Órdenes**  
**Productos**  
**Cotizaciones**  
**Ajustes**

**Gorgeous Granite Chicken** Visible ...

**Pink** Visible

**Gorgeous Granite Chicken** ...

**Detalles**

**Título**  
Gorgeous Granite Chicken

**Subtítulo**  
Gorgeous Granite Chicken

**Proveedor**  
Frami, Ruecker and Monahan

**Descripción**  
Dolorem qui quaerat nihil quasi illo. Porro possimus aperiam. Eaue consetetur tenetur eos est voluptatem quas. Architecto quia sint su

**País de origen**  
Kyrgyzstan

Guardar cambios

**Galería de medios**

Orden Media

Figura 3.21: Página para agregar producto

**INFOREDCHILE** Matias Zúñiga Centro sesión

**Emporio de los Perros** ← Productos Publicado

**Órdenes**  
**Productos**  
**Cotizaciones**  
**Ajustes**

**Gorgeous Granite Chicken** Visible ...

**Pink** Visible

**Gorgeous Granite Chicken - Pink** ...

**Detalles**

**Etiqueta de atributo**  
Color  
La etiqueta de atributo describe la categoría de la variante, por ejemplo, "Color" o "Tamaño". En la mayoría de los casos, debe ingresar el mismo valor aquí para todas las variantes del mismo nivel.

**Título corto**  
Pink  
El título corto es el valor de la etiqueta del atributo. Por ejemplo, una variante con la etiqueta de atributo "Color" podría tener un título corto "Rojo". Esto se muestra en una lista de selección en la página de detalles del producto en tu tienda.

**Título**  
Gorgeous Granite Chicken - Pink  
El título completo generalmente se muestra en el carrito, al pagar, en los resúmenes de órdenes, y en las facturas. Debe describir completamente la variante configurada. Por ejemplo, si esta es una opción con título corto "Grande", su variante principal tiene título corto "Rojo", y el título del producto es "Camiseta elegante", entonces un buen título podría ser "Camiseta de lujo - Rojo - Grande".

**País de origen**  
Kyrgyzstan

**Anchura (Centímetros)** **Longitud (Centímetros)**  
52 67

**Altura (Centímetros)** **Peso (Grams)**  
40 13

Figura 3.22: Página para agregar variante de un producto



# Capítulo 4

## Resultados

### 4.1 Marco General.

#### 4.1.1 Hitos Principales del Proyecto.

##### Hito 1 Sistema e-commerce base

Este primer hito se alcanzó a mediados de Septiembre. Se trata de un sistema de e-commerce multi-vendedor, que ofrece distintas tiendas de las empresas, cada una con su propio catálogo visible de forma separada.

Los repositorios con los códigos correspondientes a este hito se listan en la Tabla 4.1. La forma mas fácil de probarlo localmente, es siguiendo las instrucciones de <https://github.com/reactioncommerce/reaction-development-platform>, y cambiando los repositorios por los correspondientes, junto a las imágenes de contenedores creadas desde estos. En caso que los repositorios sean privados, se debe solicitar acceso o una copia de este.

Repositorio	Commit
<a href="https://gitlab.labcomp.cl/mnzuniga/reaction">https://gitlab.labcomp.cl/mnzuniga/reaction</a>	a4152d3bdcf99bfb2aa09c11dd7f2f32f7202819
<a href="https://gitlab.labcomp.cl/mnzuniga/reaction-admin/-/commits/MarketRed">https://gitlab.labcomp.cl/mnzuniga/reaction-admin/-/commits/MarketRed</a>	32add95f4117a7bc167506ce27580c0d0ac8feb0
<a href="https://gitlab.labcomp.cl/mnzuniga/marketred-storefront/-/commits/MarketRed">https://gitlab.labcomp.cl/mnzuniga/marketred-storefront/-/commits/MarketRed</a>	b45ef3cd40707d95cf4292972d9fb4715ac82098
<a href="https://gitlab.labcomp.cl/mnzuniga/marketred-identity/-/commits/MarketRed">https://gitlab.labcomp.cl/mnzuniga/marketred-identity/-/commits/MarketRed</a>	d3b123bef5bf9b0ab18d402f2376adbb3ef604ac

Cuadro 4.1: Repositorios del desarrollo del Hito 1

**Hito 2** Sistema e-commerce integrado al indexador Dado el cambio respecto a los objetivos iniciales, para llegar a este hito solo es necesario, desde la parte del sistema, ofrecer una API que InforedChile pueda consumir para obtener los datos. Por lo tanto, no son necesarias modificaciones sobre lo ya existente en el hito 1, además de las modificaciones independientes a realizar por InforedChile para consumir los datos.

**Hito 3** Sistema e-commerce integrado al sitio de búsquedas Nuevamente debido a los cambios en los requisitos durante las últimas semanas, no son necesarios cambios extras sobre la API ya existente. A pesar de esto, se requirieron cambios -completados a principios de Octubre- para armonizar visualmente la página de resultados de búsqueda de InforedChile, con la de los catálogos de MarketRed. Para esto, además de los repositorios anteriores, se debe actualizar lo listado en la tabla 4.2

Repositorio	Commit
<a href="https://gitlab.labcomp.cl/mnzuniga/marketred-storefront/-/commits/MarketRed">https://gitlab.labcomp.cl/mnzuniga/marketred-storefront/-/commits/MarketRed</a>	b45ef3cd40707d95cf4292972d9fb4715ac82098

Cuadro 4.2: Repositorios del desarrollo del Hito 3

**Hito 4** Sistema e-commerce integrado al sistema B2C/B2B completo Este hito fue alcanzado a mediados de Octubre, según lo mostrado en la Tabla 4.3. Aún falta confirmación de parte de InforedChile sobre el método a utilizar para sincronizar los datos necesarios, por lo que no se listará el commit correspondiente hasta haber validado el funcionamiento.

Repositorio	Commit
<a href="https://gitlab.labcomp.cl/mnzuniga/reaction-admin/-/commits/MarketRed">https://gitlab.labcomp.cl/mnzuniga/reaction-admin/-/commits/MarketRed</a>	32add95f4117a7bc167506ce27580c0d0ac8feb0
<a href="https://gitlab.labcomp.cl/mnzuniga/api-plugin-marketred">https://gitlab.labcomp.cl/mnzuniga/api-plugin-marketred</a>	-

Cuadro 4.3: Repositorios del desarrollo del Hito 4

**Hito 5** Sistema e-commerce integrado con sistema de pagos Este hito corresponde a lograr que al realizar una orden, se registre la operación de pago con la pasarela Khipu. Luego de esto, al cliente se le redirige a la página correspondiente, y se procesa el mensaje obtenido desde la plataforma para validar cuando el pago se efectúa. Este hito fue alcanzado con el código mostrado en la Tabla 4.4.

Repositorio	Commit
<a href="https://gitlab.labcomp.cl/mnzunga/reaction-payment-hipu">https://gitlab.labcomp.cl/mnzunga/reaction-payment-hipu</a>	4ffe596668dc0df5be8a0c3e67a73678b1192d8e

Cuadro 4.4: Repositorios del desarrollo del Hito 5

**Hito 6** Sistema e-commerce con flujo de compra refinado. Al llegar a este hito, el flujo del final de una compra pasa a ser el siguiente: 1) El vendedor marca el envío como completado, momento en que automáticamente se inicia una petición de revisión al comprador; 2) el comprador realiza una revisión y valoración de la experiencia de compra, asignando un puntaje y; 3) la valoración promedio del vendedor es actualizada, y mostrada en su página de catálogo; además, la orden es marcada como "completada".

Repositorio	Commit
<a href="https://gitlab.labcomp.cl/mnzunga/reaction-admin/-/commits/MarketRed-v2">https://gitlab.labcomp.cl/mnzunga/reaction-admin/-/commits/MarketRed-v2</a>	cb2018b9bd965ee9fac7754772e039bc8dc9785a
<a href="https://gitlab.labcomp.cl/mnzunga/marketred-storefront/-/commits/MarketRed">https://gitlab.labcomp.cl/mnzunga/marketred-storefront/-/commits/MarketRed</a>	3d4a0403becb6bf891ed864c777d0fd5724bf512
<a href="https://gitlab.labcomp.cl/mnzunga/reaction">https://gitlab.labcomp.cl/mnzunga/reaction</a>	df6c5b1947929cc59c8dc7f7c336542c417687da

Cuadro 4.5: Repositorios del desarrollo del Hito 6

### 4.1.2 Revisión de Requerimientos.

A continuación se listan los requerimientos actualizados del sistema, en base a las modificaciones a lo largo de las iteraciones del desarrollo del proyecto. Además, en la sección 4.4.2 se listan sus niveles de realización.

- RF1 El sistema debe ofrecer un Marketplace de productos. Esto es: se presentan distintos productos a un precio determinado.
- RF2 El sistema debe ser accesible para todas las empresas registradas en el sitio principal.
- RF3 Cada empresa debe poder administrar su y solo su catálogo de productos. Estos catálogos pueden verse en la página de forma independiente.
- RF4 El sistema debe permitir el registro de usuarios, los que pueden explorar y buscar productos en los catálogos.

- RF5 El sistema debe diferenciar entre usuarios y empresas, con privilegios distintos en el sistema.
- RF6 El sistema debe funcionar de manera independiente a la página principal de InfoRed-Chile.
- RF7 El sistema debe poder interactuar con el indexador del motor de búsqueda del sitio principal, pudiendo realizar actualizaciones en este de acuerdo a los productos que se agreguen.
- RF8 El sistema debe ser capaz de incrustar resultados de búsqueda de productos en el buscador del sitio principal.
- RF9 El sistema debe ser capaz de incrustar una lista de productos del catálogo de cierta empresa, al ver detalles de dicha empresa en el sitio principal.
- RF10 Los usuarios deben poder agregar productos de un catálogo a un carro de compras, y solicitar los productos o servicios.
- RF11 El sistema debe permitir al usuario pagar por los productos y/o servicios, una vez que la empresa haya confirmado la disponibilidad.

## **4.2 Revisión del Prototipo.**

### **4.2.1 Implementación de Componentes.**

#### **4.2.1.1 MarketRed API**

Cada uno de los componentes nuevos se desarrolló en su propio repositorio, y los detalles mas importantes de su implementación se listan a continuación.

#### **4.2.1.2 Pre-order Plugin**

El plugin para realizar cotizaciones se encuentra en el repositorio <https://github.com/Elav95/api-plugin-quotations.git>.

#### **4.2.1.3 Infored Plugin**

Este componente se encuentra disponible en un repositorio privado en <https://gitlab.labcomp.cl/mnzuniga/api-plugin-marketred>. Las mutaciones a ofrecer para cumplir con los casos de uso son del siguiente tipo:

```

mutation marketRedAccount($accountId: ID!, $email: Email!,
    $fname: String!, $lname: String!) {
  createOrUpdateMarketRedAccount(marketRedAccountId: $accountId,
    firstName: $fname, lastName: $lname, address: $email) {
    marketRedAccountId
    queuedForLater
  }
}

mutation marketRedShop($shopId: ID!, $name: String!, email: Email!,
    $slug: String!, $description: String!, $owner: ID!) {
  createOrUpdateMarketRedShop(marketRedShopId: $shopId,
    name: $name, address: $email, slug: $slug,
    description: $description, owner: $owner) {
    marketRedShopId
    queuedForLater
  }
}

```

#### 4.2.1.4 Khipu Plugin

El plugin para interactuar con la pasarela de pagos Khipu se encuentra en el repositorio <https://gitlab.labcomp.cl/mnzuniga/reaction-payment-khipu>.

#### 4.2.1.5 Frontend

Este componente está compuesto por dos partes: por un lado, una aplicación web NextJs que encuentra alojado en <https://gitlab.labcomp.cl/mnzuniga/marketred-storefront>, y por otra una aplicación Meteor ubicada en <https://gitlab.labcomp.cl/mnzuniga/reaction-admin>.

La comparación entre las interfaces diseñadas y las implementadas puede verse en la sección [4.2.2](#).

#### 4.2.1.6 Identity Provider

Este componente es una simple aplicación web desarrollada en Meteor, que encuentra disponible en <https://gitlab.labcomp.cl/mnzuniga/marketred-identity>, y que utiliza el componente Ory Hydra.

Realiza la autenticación de usuarios locales y, cuando se realiza una solicitud de login desde el cliente de administración, o se elige la autenticación de InforedChile, se redirige al usuario a una dirección en el sitio de estos.

Para aceptar el login desde el sitio externo, se debe enviar una solicitud POST al endpoint `/oauth2/auth/requests/login/accept?login_challenge=${c_code}` del servicio Hydra, con los parámetros:

- subject: el ID de usuario
- remember: si Hydra debe recordar el usuario, y no volver a preguntar a InforedChile
- remember\_for: el plazo de tiempo por el que se recordaría

Ejemplo:

```
PUT(
  'HR/oauth2/auth/requests/login/accept?login_challenge=${c_code}',
  params={
    'subject': user.ID,
    'remember': true,
    'remember_for': 5400 // 1.5 horas
  },
  headers = {
    'Content-Type': 'application/json',
    'Accept': 'application/json'
  })
```

El id de usuario retornado debe existir en el sistema MarketRed, por lo que se debe estar sincronizado utilizando el plugin de MarketRed.

## 4.2.2 Interfaces Implementadas en el Prototipo.

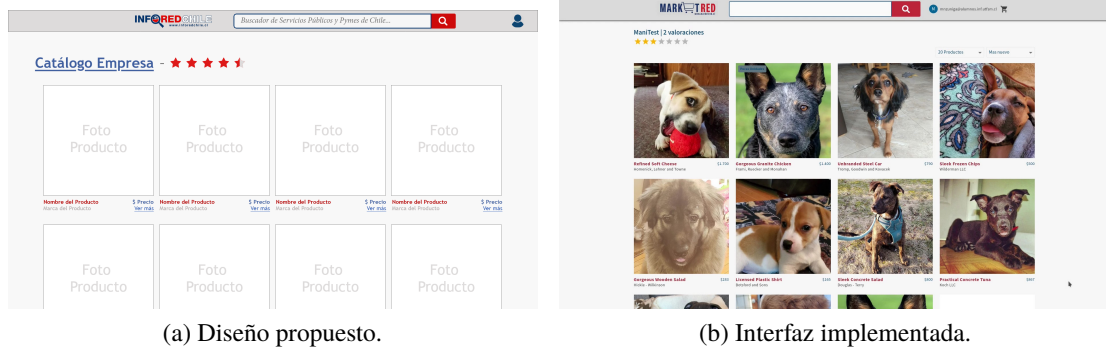
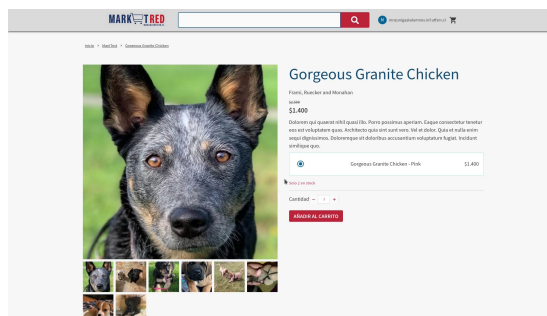


Figura 4.1: Catálogo de tienda

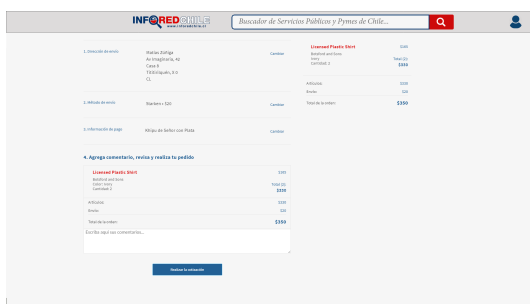


(a) Diseño propuesto.

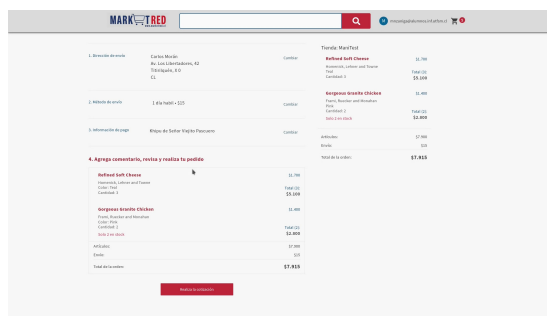


(b) Interfaz implementada.

Figura 4.2: Página del producto

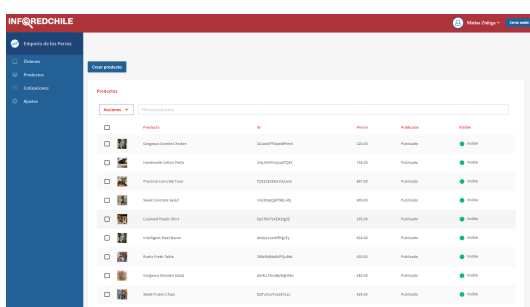


(a) Diseño propuesto.

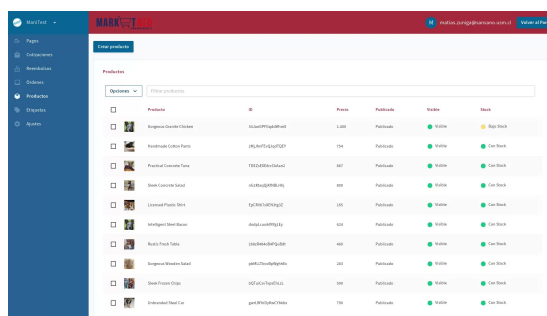


(b) Interfaz implementada.

Figura 4.3: Carrito de compra

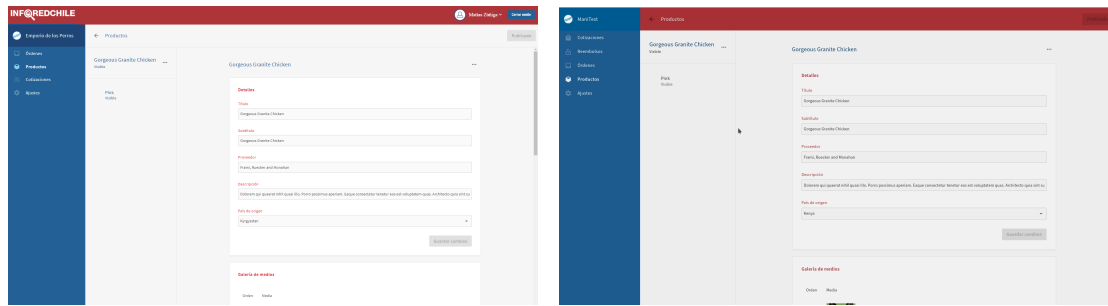


(a) Diseño propuesto.



(b) Interfaz implementada.

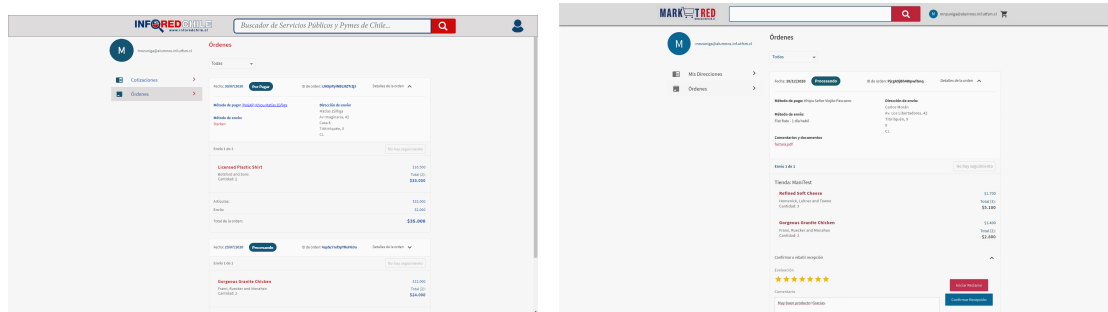
Figura 4.4: Página de productos



(a) Diseño propuesto.

(b) Interfaz implementada.

Figura 4.5: Agregar/editar producto



(a) Diseño propuesto.

(b) Interfaz implementada.

Figura 4.6: Página de órdenes

## 4.3 Plan de Testing.

Para el testeo del prototipo, se realizan distintas pruebas de integración (para probar la funcionalidad de conjunto integrado de los componentes) y pruebas unitarias (el funcionamiento de un componente en específico).

### 4.3.1 Pruebas de integración.

Se tiene un entorno de pruebas, utilizando los códigos implementados. En este entorno se tiene una base de datos interna en la cual se tienen almacenados los distintos usuarios de prueba, junto a datos de tiendas y productos.

Dado lo anterior, se han realizado pruebas con las distintas interfaces y la interacción entre ellas.

- Agregar productos a un catálogo, y modificarlos.



- Listar los productos de los catálogo en la tienda, como usuario comprador.
- Agregar productos al carro, y crear órdenes a partir de este.
- Pagar las órdenes utilizando la pasarela de pagos Khipu. Para ello se han realizado pruebas de pago con las opciones de prueba que tiene la plataforma, que incluye un banco de ejemplo para simular transferencias.
- Listar las órdenes de una tienda, como usuario vendedor, y poder cambiarles el estado, incluyendo el completarlas.

No se ha podido realizar pruebas de integración respecto a las funcionalidades que requieren interacción con el sitio de InforedChile, dada la pobre comunicación y nula respuesta de parte de ellos.

#### **4.3.2 Pruebas unitarias.**

Estas son pruebas definidas en código, cuyo objetivo es testear una parte específica de un sistema. En los Códigos [1](#), [2](#) y [3](#) se muestran 3 de las pruebas (resumidas) implementadas para el componente de pagos Khipu.

```

test("should call khipuClient.paymentsGet and find the order object," +
  " modifying its status", async () => {
  const paymentData = {
    amount: 1234, currencyCode: "CLP", shopId: "SHOP_ID",
    billingAddress: "Av. Imaginaria, 123, Quilpué",
    email: "matias.zuniga@sansano.usm.cl",
    paymentData: { fullName: "Santa Claus", referenceId: "ORDER_REFID" }
  };

  const khipuPaymentPostResult = {
    payment_id: "3456789ABCDE",
    payment_url: "https://fake.k.com/pay/3456789ABCDE",
    app_url: "https://fake.k.com/app/3456789ABCDE"
  };

  nock("https://khipu.com:443")
    .post("/api/2.0/payments", (b) => b.transaction_id == "ORDER_REFID")
    .reply(200, khipuPaymentPostResult);
  const res = await createAuthorizedPayment(mockCtx, paymentData);

  // should expire in a week
  const nowDate = new Date().getTime();
  const expireDate = new Date(res.data.expiresAfter).getTime();
  expect(expireDate).toBeLessThanOrEqual(nowDate + (7*24*60*60000));
  // a week (minus a minute) from now
  expect(expireDate).toBeGreaterThan(nowDate+(7*24*60*60000)-60000);

  expect(res.address).toBe("Av. Imaginaria, 123, Quilpué");
  expect(res.amount).toBe(1234);
  expect(res.currencyCode).toBe("CLP");
  expect(res.data.webURL).toBe("https://fake.k.com/pay/3456789ABCDE");
  expect(res.data.appURL).toBe("https://fake.k.com/app/3456789ABCDE");
  expect(res.processor).toBe("khipu");
  expect(res.shopId).toBe("SHOP_ID");
  expect(res.status).toBe("created");
  expect(res.transactionId).toBe("3456789ABCDE");
});

```

Código 1: Prueba de creación de pagos.

```

test("should call khipuClient.paymentsGet with the proper parameters" +
    " and return a properly formatted payment", async () => {
    const paymentData = { notificationToken: "ABCDEF123ABBBBBBBBBBBBBB" };
    const khipuPaymentsGetResult = {
        amount: 123, receiver_id: "test",
        transaction_id: "ORDER123456ABC",
        payment_id: "0A1B2C3D",
        status: "done", status_detail: "payed",
        expires_date: new Date(Date.now() + 60 * 1000)
    };

    nock("https://khipu.com:443").get("/api/2.0/payments")
        .query({ notification_token: "ABCDEF123ABBBBBBBBBBBBBB" })
        .reply(200, khipuPaymentsGetResult);

    const paymentData = {
        _id: "DB_PAYMENT_ID", shopId: "SHOP_ID",
        processor: "khipu", status: "created",
        transactionId: "0A1B2C3D" // ... RECORTADO ...
    };
    const items: [{ _id: "ITEM_1", quantity: 5 },
        { _id: "ITEM_2", quantity: 1 }];
    mockCtx.collections.Orders.findOne.mockReturnValueOnce(
        Promise.resolve({ _id: "ORDER_1", shipping: [{ items }],
            shopId: "SHOP_ID", payments: [{...paymentData}],
            workflow: { status: "new", workflow: ["new"] }
        }));

    mockCtx.collections.Orders.findOneAndUpdate.mockReturnValueOnce(
        Promise.resolve({ modifiedCount: 1, value: {} }));

    const result = await updateKhipuInfo(mockCtx, paymentData);
    expect(mockCtx.mutations.cancelOrderItem).toHaveBeenCalledTimes(0);

    expect(mockCtx.collections.Orders.findOneAndUpdate)
        .toHaveBeenCalledWith({ _id: "ORDER_1" },
            { $push: { "workflow.workflow": "coreOrderWorkflow/processing" }
            $set: {
                "payments": [{ ...paymentData, status: "completed" }],
                "workflow.status": "coreOrderWorkflow/processing",
                "updatedAt": jasmine.any(Date)
            }
        }, { returnOriginal: false }
    );
});

```

Código 2: Prueba de confirmación de pagos.

```

test("should cancel the payment on full refunds", async () => {
  const paymentData = {
    _id: "DB_PAYMENT_ID",
    processor: "khipu",
    status: "created",
    transactionId: "0A1B2C3D",
    shopId: "SHOP_ID"
    // ... RECORTADO ...
  };

  nock("https://khipu.com:443")
    .delete("/api/2.0/payments/0A1B2C3D")
    .reply(200, { message: "TEST MESSAGE" });

  mockCtx.collections.Orders.findOne.mockReturnValueOnce(
    Promise.resolve({ _id: "ORDER_1", shipping: [],
      shopId: "SHOP_ID",
      payments: [{ ...paymentData }],
      workflow: { status: "new", workflow: ["new"] }
    }));

  await reactionPaymentKhipuCreateRefund(mockCtx, paymentData, 123,
    "Order has been cancelled", { shopId: "SHOP_ID" });
  expect(mockCtx.collections.Orders.findOneAndUpdate).toHaveBeenCalledWith(
    { _id: "ORDER_1" },
    { $set: {
      payments: [{ ...paymentData, status: "canceled" }],
      updatedAt: jasmine.any(Date)
    } }, { returnOriginal: false }
  );
  expect(mockCtx.collections.ReactionPaymentKhipuRefunds.insertOne)
    .toHaveBeenCalledWith({ reason: "Order has been cancelled",
      transactionId: "0A1B2C3D",
      shopId: "SHOP_ID",
      status: "completed"
      // ... RECORTADO ...
    });
});

```

Código 3: Prueba de creación de reembolsos.

En los ejemplos mostrados puede verse que:

- Se utilizan directamente los funciones de interés, con datos de prueba.

- Se emula la respuesta a recibir del sistema web de Khipu.
- Se emulan las respuestas de las bases de datos.
- Se comprueba que los datos devueltos sean de cierto tipo, y que se guarda en la base de datos la información esperada.

## 4.4 Verificación y Validación.

### 4.4.1 Verificación.

En la sección 4.2.2 se muestra una comparativa entre las interfaces diseñadas y las implementadas en el sistema. Se puede ver que las implementaciones de las interfaces siguen en su mayoría lo propuesto en los primeros diseños: se adoptaron los colores y la disposición de los elementos de manera muy similar a lo planteado inicialmente, y se implementaron sin grandes variaciones las barras y botones. Sin embargo esto, hay pequeños detalles que difieren de lo diseñado, como por ejemplo: el tamaño de algunos textos -reducidos para que luzcan mejor en la página-, o el logo, que se decidió modificar dada la definición como equipo de un nombre e imagen distintos para el nuevo sitio -siguiendo la idea original del nombre de InforedChile, pero adaptándolo a la temática de Marketplace-.

En cuanto a la implementación de los componentes, la mayoría de ellos se encuentran ya implementados y funcionando. A la API de MarketRed le falta un plugin que incorporar, el cual se encuentra en proceso de desarrollo a falta de unos detalles de estar listo. Los demás se encuentran integrados y funcionando, así como también lo están los demás componentes. Se han realizado pruebas de su funcionamiento, pero aún está pendiente el tema de la integración con el sitio web principal de InforedChile.

### 4.4.2 Validación.

Para validar que el sistema cumple las expectativas de InforedChile, se realizan reuniones periódicas con tal de mostrar el comportamiento del sistema para distintos casos de uso.

En la tabla 4.6 se muestra el listado revisado de requerimientos funcionales del sistema, junto a su estado actual de implementación. Los requerimientos que se declaran como implementados, pero en color naranja, son aquellos en los que solo hacen falta cambios del lado de InforedChile para mostrar resultados.

De la lista, solo un requerimiento se encuentra incompleto, dado que el comprador debe pagar inmediatamente al crear la orden, sin necesitar confirmación del vendedor.

Requerimiento	Implementación
[RF1] El sistema debe ofrecer un Marketplace de productos. Esto es: se presentan distintos productos a un precio determinado.	✓
[RF2] El sistema debe ser accesible para todas las empresas registradas en el sitio principal.	✓
[RF3] Cada empresa debe poder administrar su y solo su catálogo de productos. Estos catálogos pueden verse en la página de forma independiente.	✓
[RF4] El sistema debe permitir el registro de usuarios, los que pueden explorar y buscar productos en los catálogos.	✓
[RF5] El sistema debe diferenciar entre usuarios y empresas, con privilegios distintos en el sistema.	✓
[RF6] El sistema debe funcionar de manera independiente a la página principal de InfoRedChile.	✓
[RF7] El sistema debe poder interactuar con el indexador del motor de búsqueda del sitio principal, con el objetivo de realizar actualizaciones de acuerdo a la disponibilidad de los productos.	✓
[RF8] El sistema debe ser capaz de incrustar resultados de búsqueda de productos en el buscador del sitio principal.	✓
[RF9] El sistema debe ser capaz de incrustar una lista de productos del catálogo de cierta empresa, al ver detalles de dicha empresa en el sitio principal.	✓
[RF10] Los usuarios deben poder agregar productos de un catálogo a un carro de compras, y solicitar los productos o servicios.	✓
[RF11] El sistema debe permitir al usuario pagar por los productos y/o servicios, una vez que la empresa haya confirmado la disponibilidad.	✗

Cuadro 4.6: Tabla de implementación de requerimientos

## Capítulo 5

### Conclusiones

A pesar de las dificultades producto de la falta de comunicación de la parte interesada InforedChile, los principales hitos del proyecto fueron alcanzados, y se encuentra registro de ello en el repositorio de MarketRed. Para lograr esto se definieron los distintos actores que estarán en interacción con el sitio web, y se implementaron las formas en que interactúan.

Las interfaces de usuario fueron implementadas siguiendo lo que se había propuesto inicialmente en el diseño de éstas, y se encuentran funcionales. Estas además han sido verificadas y validadas con los requerimientos definidos al inicio del proyecto, y su respectiva actualización de acuerdo a las iteraciones y el feedback realizado por parte de la contraparte. En lo relativo al desarrollo de la plataforma, queda faltante la integración con el sitio web principal de InforedChile, para lo cual será necesario realizar nuevas pruebas, comprobar en práctica la compatibilidad de los componentes, y realizar los ajustes que sean necesarios.

La mayoría de los riesgos han sido abordados, principalmente los que están relacionados con el tema de seguridad de los datos de autenticación: esto por el uso del Identity Provider, que permite desacoplar la lógica de inicio de sesión, y comprobar de forma segura con el sitio de InforedChile cuando un cliente se encuentra logueado. Falta comprobar, sin embargo, que el manejo y traspaso de información entre ambos sitios sea segura, según lo que se implemente por parte del sitio de InforedChile.

Los riesgos aún latentes están relacionados a factores externos, como la posibilidad de que las empresas/MiPymes no estén dispuestas a ser parte de la nueva plataforma, y no quieran ofrecer sus productos en ella. Esto se debe evaluar una vez que el sitio se encuentre en funcionamiento, y podría ser mitigado con campañas publicitarias o generando incentivos para las empresas que ofrezcan sus productos en la plataforma. Asociado a esto, también existe la posibilidad de que la plataforma genere ingresos menores que los necesarios para mantener el sitio funcionando: dadas las circunstancias actuales de pandemia y restricciones es que el comercio electrónico ha experimentado un boom, con lo que han surgido varias plataformas que asoman como posible competencia para la plataforma e-commerce de InforedChile; es por eso que se debe posicionar la marca dentro del rubro con un programa de publicidad.

## 5.1 Proyecciones

En base a los resultados obtenidos y lo que se espera, o se podría llegar a hacer con el proyecto es que se plantean los siguientes puntos como un posible trabajo a futuro

- Validar y completar integración con InforedChile. Dado que las funcionalidades de integración de usuarios MiPyme no ha podido ser probada.
- Automatizar la función de pago a las MiPymes. Nuestro sistema puede generar informes de pago, con todas las transacciones aún no rendidas, para que estas sean pagadas, considerando el descuento de comisión. Sin embargo, esto debe iniciarse manualmente.
- Integrar opciones de delivery, para quitar a las MiPymes la carga de manejar las entregas
- Realizar actividades de marketing y difusión para potenciar la marca y atraer a las MiPymes y a compradores.



# Anexo A

## GraphQL

GraphQL es un lenguaje para la consulta y manipulación de datos que ofrece una arquitectura para APIs alternativa a REST.

### A.1 Descripción

Con GraphQL, se especifica la estructura de datos esperada al momento de realizar solicitudes. Como consecuencia a esto, no se gastan recursos de red en enviar datos innecesarios, ni recursos de procesamiento por derivar información extra.

Por ejemplo, enviando a un servidor que entienda GraphQL el texto en Código 4 como la variable query en una solicitud, se obtiene de respuesta del Código 5. Como se puede ver, la respuesta obtenida tiene exactamente los datos solicitados.

```

query GetAllOrders {
  orders {
    _id
    customer {
      _id
    }
    products {
      product {
        _id
        name
        price
      }
      quantity
    }
    total
  }
}

```

Código 4: Consulta de ejemplo

Como puede verse, la solicitud anterior retorna un listado de todas las órdenes existentes, pero usualmente se deben hacer peticiones respecto a órdenes de un cliente en específico. Esto es posible llamando a una consulta con variables, como la del Código 6. En este caso, el valor de los parámetros puede enviarse como el campo `variables` en la solicitud HTTP, siguiendo el ejemplo del Código 7.

El signo de exclamación (!) indica que un parámetro no es opcional, y no puede ser nulo.

La respuesta a esta solicitud puede verse en el código 8, siendo esta similar a la de la primera consulta de este anexo, pero limitado a las órdenes del cliente especificado.

Un problema que puede observarse en los ejemplos anteriores, es que en en ambas peticiones debe especificarse la estructura de datos esperada, aunque esperen datos muy similares. Para ahorrar el tener que incluir a cada momento la misma estructura, es que existen los fragmentos.

Un fragmento describe una parte de un tipo de datos a ser solicitada, reusable fácilmente para distintas peticiones. Por ejemplo, la solicitud del Código 9 es equivalente a el primer ejemplo de este Anexo, conteniendo parte de la descripción de los datos de tipo Order retornados.

## A.2 Esquemas

En la sección anterior se mencionó los fragmentos, que describen una parte de un objeto de cierto tipo. En GraphQL, todos los datos tienen cierto tipo, y son definidos en el servidor

```

{ "data": {
  "orders": [
    {
      "_id": "abcdef123456",
      "customer": { "_id": 1 },
      "products": [
        {
          "product": {
            "name": "Mesa",
            "price": 1000
          },
          "quantity": 10
        }, {
          "product": {
            "name": "Silla",
            "price": 150
          },
          "quantity": 40
        }
      ],
      "total": 16000
    }, {
      "_id": "abcdef123456",
      "customer": { "_id": 2 },
      "products": [
        {
          "product": {
            "name": "Naipes",
            "price": 500
          },
          "quantity": 5
        }
      ],
      "total": 2500
    }
  ]
}}

```

Código 5: Respuesta de ejemplo

por esquemas como el mostrado en el Código 10.

Cada tipo de objeto está compuesto por distintos campos, que a su vez tienen cierto tipo. Similar a su significado en los parámetros de una solicitud, el signo de exclamación (!) denota que un campo en específico no puede ser nulo.

```

query GetClientOrders($client: ID!) {
  customerOrders(customer: $client) {
    _id
    products {
      product {
        _id
        name
        price
      }
      quantity
    }
    total
  }
}

```

Código 6: Consulta con variables

```

{
  "client": 2
}

```

Código 7: Variables de la consulta

Las operaciones son definidas de la misma forma, habiendo 2 tipos de estas: 1) las consultas (*queries*), que es para obtener datos, y 2) las mutaciones (*mutations*), que son usadas para modificar datos. Un ejemplo de la definición de distintas operaciones es mostrada en el Código 11. Además, el Código 12 es un ejemplo del uso de la mutación definida.

```

{
  "data": {
    "customerOrders": [
      {
        "_id": "abcdef123456",
        "products": [
          {
            "product": {
              "name": "Naipes",
              "price": 500
            },
            "quantity": 5
          }
        ],
        "total": 2500
      }
    ]
  }
}

```

Código 8: Respuesta a la consulta con variable

```

query GetAllOrders {
  orders {
    customer {
      _id
    }
    ...OrderCommon
  }
}

fragment OrderCommon on Order {
  _id
  products {
    product {
      _id
      name
      price
    }
    quantity
  }
  total
}

```

Código 9: Consulta usando fragmentos

```

type User {
  _id: ID!
  name: String
  email: Email!
}
type Product {
  _id: ID!
  name: String!
  description: String
  price: Float!
}
type Order {
  _id: ID!
  customer: User!
  products: [Product]
  total: Float
  status: String
}

```

Código 10: Esquema de datos

```

type Query {
  orders: [Order]
  customerOrders(customer: ID!): [Order]
}
type Mutation {
  cancelOrder(orderId: ID!): Order
}

```

Código 11: Definición de tipos de solicitud

```

mutation CancelOrder($client: ID!) {
  cancelOrder(customer: $client) {
    ...OrderCommon
  }
}

```

Código 12: Uso de ejemplo de una mutación

# Bibliografía

- [1] ApañoTuPyme. Pagos | ApañoTuPyme Manual de Inscripción. [en línea] <https://unete.apanotupyme.cl/docs/pages/payments.html>. [Accedido: 14-06-2020].
- [2] Magento. eCommerce Platforms | Best eCommerce Software for Selling Online | Magento. [en línea] <https://magento.com/es>. [Accedido: 29-04-2020].
- [3] Magento. Github - magento/magento2. [en línea] <https://github.com/magento/magento2>. [Accedido: 09-01-2021].
- [4] Magento. Magento Forums. [en línea] <https://community.magento.com/>. [Accedido: 09-01-2021].
- [5] MercadoLibre. Costos de vender un producto. [en línea] [https://www.mercadolibre.cl/ayuda/Costos-de-vender-un-producto\\_870](https://www.mercadolibre.cl/ayuda/Costos-de-vender-un-producto_870). [Accedido: 14-06-2020].
- [6] Ministerio de Economía, Fomento y Turismo. Quinta Encuesta Longitudinal de Empresas (ELE5). [en línea] <https://www.economia.gob.cl/2019/03/12/quinta-encuesta-longitudinal-de-empresas-ele5.html/>. [Accedido: 13-06-2020].
- [7] mirumee. mirumee/saleor. [en línea] <https://gitter.im/mirumee/saleor>. [Accedido: 16-01-2021].
- [8] mirumee. mirumee/saleor: A modular, high performance, headless e-commerce platform built with Python, GraphQL, Django, and ReactJS. [en línea] <https://github.com/mirumee/saleor>. [Accedido: 16-01-2021].
- [9] OpenCart. Multi-Store - OpenCart Documentation. [en línea] <http://docs.opencart.com/en-gb/administration/multi-store/>. [Accedido: 29-04-2020].
- [10] OpenCart. OpenCart - Open Source Shopping Cart Solution. [en línea] <https://www.opencart.com/>. [Accedido: 29-04-2020].
- [11] OpenCart. OpenCart Chile - Latinoamérica. [en línea] [https://www.opencart.cl/#us\\_grid\\_1](https://www.opencart.cl/#us_grid_1). [Accedido: 10-01-2021].
- [12] OpenCart. OpenCart Community. [en línea] <https://forum.opencart.com>. [Accedido: 14-01-2021].

- [13] Oscar. django-oscar - Grupos de Google. [en linea] <https://groups.google.com/g/django-oscar>. [Accedido: 16-01-2021].
- [14] Oscar. django-oscar/django-oscar: Domain-driven e-commerce for Django. [en linea] <https://github.com/django-oscar/django-oscar>. [Accedido: 16-01-2021].
- [15] Oscar. Oscar - django-oscar 2.0 documentation<. [en linea] <https://django-oscar.readthedocs.io/en/2.0.4/>. [Accedido: 29-04-2020].
- [16] Oscar. Oscar - Domain-driven e-commerce for Django. [en linea] <http://oscarcommerce.com/>. [Accedido: 29-04-2020].
- [17] Oscar. Partner - django-oscar 2.0 documentation<. [en linea] <https://django-oscar.readthedocs.io/en/2.0.4/ref/apps/partner.html>. [Accedido: 29-04-2020].
- [18] PrestaShop. Crear Tienda Online Gratis con Prestashop | Venta Online. [en linea] <https://www.prestashop.com/es>. [Accedido: 29-04-2020].
- [19] PrestaShop. Forums Ecommerce - PrestaShop. [en linea] <https://www.prestashop.com/forums/>. [Accedido: 10-01-2021].
- [20] PrestaShop. PrestaShop/PrestaShop: PrestaShop is a fully scalable open source e-commerce solution. [en linea] <https://github.com/PrestaShop/PrestaShop>. [Accedido: 10-01-2021].
- [21] PurpleTree Software LLP. No. 1 BestSeller Multi Vendor Marketplace Opencart Extension | 50% OFF. [en linea] <https://www.purpletreesoftware.com/multi-vendor-marketplace-opencart.html>. [Accedido: 29-04-2020].
- [22] Reaction Commerce. Reaction Commerce | Connecting the world through commerce. [en linea] <https://reactioncommerce.com/>. [Accedido: 29-04-2020].
- [23] Reaction Commerce. Reaction Commerce · GitHub. [en linea] <https://github.com/reactioncommerce>. [Accedido: 02-05-2020].
- [24] Reaction Commerce. reactioncommerce/reaction - Gitter. [en linea] <https://gitter.im/reactioncommerce/reaction>. [Accedido: 02-05-2020].
- [25] Reaction Commerce. Shops · Reaction Docs. [en linea] <https://docs.reactioncommerce.com/docs/concepts-shops>. [Accedido: 29-04-2020].
- [26] Reaction Commerce. Understanding API Plugins · Reaction Docs. [en linea] <https://docs.reactioncommerce.com/docs/core-plugins-intro>. [Accedido: 29-04-2020].
- [27] Saleor. Saleor Commerce community. [en linea] <https://spectrum.chat/saleor>. [Accedido: 16-01-2021].



- [28] Saleor. Saleor – A headless, GraphQL-first, open-source e-commerce platform. [en línea] <https://saleor.io/>. [Accedido: 29-04-2020].
- [29] Spark Solutions Sp. z o.o. Spree Commerce - an ecommerce platform with over 1 million downloads. [en línea] <https://spreecommerce.org/>. [Accedido: 29-04-2020].
- [30] Spark Solutions Sp. z o.o. Spree Multi Vendor marketplace extension. [en línea] [https://github.com/spree-contrib/spree\\_multi\\_vendor](https://github.com/spree-contrib/spree_multi_vendor). [Accedido: 29-04-2020].
- [31] Spree. Improved Multi Store support. [en línea] [https://github.com/spree/spree/blob/release/4-2-0-rc3/guides/src/content/release\\_notes/4\\_2\\_0.md#improved-multi-store-support](https://github.com/spree/spree/blob/release/4-2-0-rc3/guides/src/content/release_notes/4_2_0.md#improved-multi-store-support). [Accedido: 14-01-2021].
- [32] Spree. Join Spree Commerce on Slack! [en línea] <http://slack.spreecommerce.org>. [Accedido: 14-01-2021].
- [33] Spree. spree/spree: Spree is an open source E-commerce platform for Rails 6 with a modern UX, optional PWA frontend, REST API, GraphQL, several official extensions and 3rd party integrations. [en línea] <https://github.com/spree/spree>. [Accedido: 14-01-2021].
- [34] Transbank y CCS. Comercio online se triplica, pero tiendas físicas, turismo y entretenimiento extienden su profunda crisis.  
[en línea] <https://www.ccs.cl/2020/05/29/comercio-online-se-triplica-pero-tiendas-fisicas-turismo-y-entretenicion-extienden-su-profunda-crisis/>. [Accedido: 13-06-2020].
- [35] WC Lovers. WCFM Marketplace - Best Multivendor Marketplace for WooCommerce - WordPress plugin. [en línea] <https://wordpress.org/plugins/wc-multivendor-marketplace/>. [Accedido: 29-04-2020].
- [36] WC Marketplace, The Grey Parrots. WC Marketplace - WordPress plugin. [en línea] <https://wordpress.org/plugins/dc-woocommerce-multi-vendor/>. [Accedido: 29-04-2020].
- [37] WebKul SoftWare Private Limited. Advanced Multi Vendor Marketplace. [en línea] <https://addons.prestashop.com/es/creacion-marketplace/8057-advanced-multi-vendor-marketplace.html>. [Accedido: 29-04-2020].
- [38] WebKul SoftWare Private Limited. Multi Vendor Marketplace. [en línea] <https://marketplace.magento.com/webkul-module-marketplace.html>. [Accedido: 29-04-2020].
- [39] WooCommerce. Join the WooCommerce Community Slack. [en línea] <https://woocommerce.com/community-slack/>. [Accedido: 09-01-2021].

- [40] WooCommerce. WooCommerce - Sell Online With The eCommerce Platform for WordPress. [en línea] <https://woocommerce.com/>. [Accedido: 29-04-2020].
- [41] WooCommerce. woocommerce/woocommerce: An open source eCommerce plugin for WordPress. [en línea] <https://github.com/woocommerce/woocommerce/>. [Accedido: 09-01-2021].