

Trabajo Práctico Profesional

Desarrollo de un Sistema para la Gestión del Suministro de Cannabis Medicinal

Autores:

Nombre	Padrón	Email
Del Pino Cardenas, Ronnie	93.575	rdelpino@fi.uba.ar
Mundani Vegega, Ezequiel	102.312	emundani@fi.uba.ar
Otegui, Matías Iñaki	97.263	motegui@fi.uba.ar

Tutor: Prof. Dr. Diego P. Corsi

20 de febrero de 2025

Índice

1. Introducción	3
1.1. Acerca de Cultivando Conocimiento	3
2. Objetivos	4
3. Diseño de la solución	6
4. Reglas de negocio	8
5. Funcionalidades de la aplicación	10
5.1. <i>Landing page</i>	10
5.2. Backoffice	10
5.2.1. Usuarios	10
5.2.2. Productos	10
5.2.3. Pedidos	10
5.2.4. Sedes	11
5.2.5. Turnos	11
5.2.6. Pagos	11
5.2.7. Métricas	11
5.2.8. Sitio	11
5.3. Vista de socio	11
5.3.1. Información	12
5.3.2. Catálogo	12
5.3.3. Pedidos	12
5.3.4. Carrito	12
5.3.5. Pagos	12
5.3.6. Contacto	12
6. Implementación	13
6.1. Aplicación Web	13
6.2. Servicio Usuarios	15
6.3. Servicio Productos	15
6.4. Servicio Pedidos	16
6.5. Implementaciones destacadas	17
6.5.1. Autenticación y autorización	17

6.5.2. Subida de imágenes y documentos	18
6.5.3. Descarga de imágenes y documentos	19
7. Despliegue	21
7.1. Compra del dominio	22
7.2. Bases de datos	23
7.3. Almacenamiento de documentos e imágenes	23
7.4. Servicios	23
8. Dificultades encontradas	24
9. Seguridad	28
9.1. JWT (JSON Web Tokens)	28
9.2. OAuth de Firebase	28
10. Accesibilidad	29
11. Futuro del proyecto	30
11.1. Entrevista con los directores de Cultivando Conocimiento	30
11.2. Soporte técnico	31
11.3. Oferta de la aplicación a otras organizaciones	31
12. Conclusiones	32
13. Referencias	34

Anexos

A. Wireframes	35
----------------------	-----------

1. Introducción

En la actualidad, gran parte de las asociaciones civiles de América Latina y el Caribe padecen de un grado bajo de acceso a la tecnología[1]. En este marco y como propuesta de Trabajo Profesional, se propuso trabajar en conjunto con la ONG Cultivando Conocimiento para el desarrollo e implementación de un sistema de gestión de socios e inventario, capaz de generar órdenes de pago, que cuente con métricas acerca de las operaciones y que facilite a sus socios la creación de pedidos.

Este proceso de transformación digital permitirá a la ONG modernizar y optimizar sus procesos de gestión, que en consecuencia mejorará el bienestar y calidad de vida de sus socios.

1.1. Acerca de Cultivando Conocimiento

La Asociación Civil Cultivando Conocimiento es una ONG vinculada a la salud encargada de desarrollar e implementar acciones de promoción y prevención dirigidas a los equipos de salud y organizaciones de la sociedad civil que aborden la temática del cannabis medicinal. Busca atender la salud integral de las personas a las cuales se les indique como modalidad terapéutica, medicinal o paliativa del dolor, el uso de la planta de cannabis o sus derivados. En virtud de ello, provee a sus socios con genéticas de cannabis o sus productos. Sus socios deben estar inscriptos en el Registro del Programa Nacional de Cannabis Medicinal[2] (REPROCANN) y contar con una suscripción con la ONG para poder acceder a estos beneficios.

Producto del incremento constante de socios existe una gran problemática de trabajo en distintas áreas de la asociación, principalmente por falta de medios tecnológicos eficientes y autónomos, como un sitio web que permita comunicar, administrar, obtener métricas y monitorear aspectos relacionados con el trabajo diario.

2. Objetivos

Con el fin de mejorar la eficiencia del trabajo realizado dentro de la organización, disminuir costos y mejorar las herramientas con las que contaban, luego de reuniones con los directores y de observar su actual manejo de las distintas actividades realizadas en la asociación, se establecieron los siguientes objetivos generales del proyecto:

- Optimizar la eficiencia del trabajo realizado dentro de la organización automatizando tareas repetitivas y manuales.
- Mejorar la comunicación entre los socios y el personal de la ONG Cultivando Conocimiento.
- Centralizar la gestión de inventario, socios y pedidos de tal manera que sean accesibles para la toma de decisiones.
- Ofrecer una interfaz de usuario intuitiva para que los socios puedan realizar sus pedidos de manera directa.
- Seguimiento de las transacciones realizadas con los socios y administradores.
- Implementación de medidas de seguridad para la protección del acceso a la información.

Respecto de los objetivos específicos se establecieron los siguientes:

- Desarrollar un sistema virtual (*backoffice*) para que los administradores de la ONG puedan gestionar el inventario de productos, la información de los socios y los pedidos que estos realizan.
- Monitorear las diferentes actividades con el fin de extraer conclusiones para mejorar el servicio.
- Contar con métricas para mejorar la gestión de inventario, ingresos, pedidos y socios.
- Que los socios puedan realizar sus pedidos sin necesidad de comunicarse previamente con la organización.
- Actualizar y mejorar su página web anterior, haciéndola *responsive*, permitiendo modificarla desde el backoffice y permitiendo informar novedades.
- Registro y seguimiento de transacciones y documentación de los socios y administradores.
- Gestión de roles de usuarios para que los socios puedan (o no) acceder a las diferentes secciones del sistema de manera segura y delimitada.
- Establecer distintas categorías que permitan acceder a cantidades diferentes de productos.

Desarrollo iterativo

Cabe aclarar que a pesar de haber establecido objetivos antes de comenzar a implementar el proyecto, hubo varias reuniones con los directores en las que se fueron validando las implementaciones y ellos fueron ajustando el alcance del proyecto e incluso solicitando que se quiten o agreguen ciertas funcionalidades. Entre las agregadas se encuentran:

- Poder generar comprobantes de pago automáticamente para los socios.
- Que los socios puedan expresar opiniones sobre el servicio por cada pedido realizado.
- Métricas para conocer el estado del inventario de productos.
- Que los socios puedan cambiar su categoría siempre y cuando esta sea inferior a la actual.

3. Diseño de la solución

Durante la primera reunión con los directores de Cultivando Conocimiento, se pidió una aplicación mediante la cual sus socios pudieran realizar directamente los pedidos, en vez de tener que coordinarlos con los directores. También surgió la necesidad de un sistema para gestionar el inventario, a los socios y a los productos; en su momento contaban con una planilla de Excel.

Inicialmente los directores comunicaron que iban a necesitar un sistema capaz de soportar a 200 usuarios simultáneamente con posibilidad de que la asociación se expanda más, por lo que se optó por una arquitectura orientada a microservicios, permitiendo escalabilidad. Además, pidieron que la aplicación sea *mobile*.

Se optó por estructurar el sistema en tres partes principales:

- Backoffice: diseñado para los administradores, este panel de gestión permite manejar eficientemente todas las operaciones del sistema, incluyendo la administración de inventario, socios, pedidos, métricas y demás configuraciones del sitio web.
- Vista de socios: un espacio para los socios, donde pueden realizar pedidos, cargar comprobantes de pago, gestionar documentos como su contrato y REPROCANN. También pueden autogestionar parte de sus datos personales y el acceso a la plataforma mediante la posibilidad de cambiar el correo electrónico y su contraseña.
- *Landing page*: una página principal orientada a la presentación del servicio, proporcionando información clave y permitiendo el acceso a las demás secciones del sistema.

En una segunda reunión se validó la solución pensada, se mostraron *Wireframes* (ver anexo A) de lo que podría ser la aplicación y se les planteó a los directores que de acuerdo a lo que pedían, una aplicación Web parecía ser más pertinente, principalmente debido a los costos con que se debe contar para mantener una aplicación *mobile* en las plataformas de distribución como Apple Store o Play Store. Los directores decidieron seguir con la solución Web.

Estando en condiciones para comenzar una planificación más en profundidad, se realizaron historias de usuario. Como herramienta de gestión de la documentación se utilizó GitHub Projects[3], que permitió la gestión de tareas, de prioridades y la distribución de trabajo entre los integrantes de equipo.

Luego se comenzó por la implementación de una *landing page* básica, un servicio de Usuarios y un servicio de Productos.

En general, se tuvo una reunión cada una o dos semanas entre los integrantes de equipo, en las que se mostraban las funcionalidades implementadas, se debatían posibles enfoques a los problemas que fueron

surgiendo y se planificaba en qué iba a trabajar cada uno.

Se tuvieron reuniones mensuales con los directores para validar lo implementado, despejar dudas sobre cómo implementar funcionalidades futuras y entender en mayor profundidad las reglas de negocio. A su vez y con el objetivo de mantener la comunicación con ellos de manera fluida, se armó un grupo de WhatsApp para despejar posibles dudas que hubiesen surgido o que no se hubiesen contemplado en las reuniones.

4. Reglas de negocio

A partir de las varias reuniones con los directores, se obtuvieron las siguientes reglas de negocio, las cuales fueron consideradas y validadas al construir el sistema:

- 1) La ONG Cultivando Conocimiento, como toda ONG en Argentina[4], es sin fines de lucro, por lo que no realiza ventas ni tiene clientes.
- 2) La asociación cuenta con dos directores.
- 3) Existen socios, quienes pagan una cuota para asociarse a la organización.
- 4) Estar asociados les permite abastecerse de las genéticas de marihuana que provee la asociación.
- 5) Los socios pueden pertenecer a diferentes categorías. Al momento de realizar el sistema existen cuatro: A, B, C y D, pero en el futuro se podrían agregar más.
- 6) Cada categoría tiene una cantidad de gramos asociada, dicha cantidad establece cuántos gramos pueden pedir por mes. Siendo A de 10g, B de 20g, C de 30g y así sucesivamente.
- 7) La asociación también ofrece productos derivados del cannabis, los cuales tienen un equivalente en gramos para que los socios puedan abastecerse con ellos.
- 8) Los socios necesitan estar inscriptos en el REPROCANN y deben firmar un contrato que les presenta la ONG para poder asociarse.
- 9) La autorización del REPROCANN cuenta con un vencimiento.
- 10) El mecanismo para asociarse es tener una reunión presencial, validar que el futuro socio esté en el REPROCANN y la firma de un contrato con la asociación.
- 11) Los socios pueden acceder a su información desde el sistema pero únicamente editar su teléfono, email y contraseña.
- 12) Inicialmente cada socio puede elegir a qué categoría pertenecer, para aumentar de categoría es necesario solicitar una recategorización. Para disminuirla es necesario no haber realizado pedidos ese mes.
- 13) Los socios pueden ver el catálogo de productos y genéticas.
- 14) Los diferentes ítems del catálogo contienen hasta cinco imágenes y una descripción de los mismos.
- 15) Los socios pueden agregar ítems a un carrito.

- 16) Los socios pueden realizar pedidos a partir de sus carritos.
- 17) Los socios pueden realizar dos pedidos mensuales como máximo.
- 18) Tanto los socios como los administradores pueden cancelar pedidos.
- 19) Los pedidos deben ser de diez gramos o más por pedido.
- 20) Los pedidos de los socios no deben exceder el cupo de gramos establecido para su categoría.
- 21) Para el abastecimiento de genéticas de marihuana, solo es posible hacerlo en múltiplos de cinco gramos.
- 22) Los pedidos deben ser de hasta cuarenta gramos, debido a que ese es el máximo legal que una persona puede transportar en la vía pública[2].
- 23) Debido a que existe un mínimo y máximo de pedidos y solo se pueden realizar dos por mes, el pedido restante debe poder cumplir con los máximos y mínimos de gramos ya establecidos.
- 24) La organización cuenta con dos sedes en las cuales se entregan los pedidos.
- 25) Se debe contar con un sistema de turnos para que el socio elija en qué momento y sede retirar el pedido.
- 26) Los administradores deben confirmar cada pedido, teniendo en cuenta el tiempo para prepararlo.
- 27) Se debe contar con un inventario de las diferentes genéticas y productos para poder realizar los pedidos.
- 28) Dicho inventario debe ser reservado automáticamente cada vez que se realiza un pedido y disminuido automáticamente cada vez que se entrega un pedido.
- 29) Los posibles estados de un pedido son pendiente, confirmado, entregado o cancelado.
- 30) Los administradores deben poder ocultar ítems del catálogo de genéticas y productos.
- 31) Los administradores deben poder limitar ciertos productos, de manera que los socios solo puedan utilizar hasta la mitad de su cuota en ellos.
- 32) Se debe mostrar que un socio está retrasado con su cuota, pero igualmente debe poder realizar pedidos.

5. Funcionalidades de la aplicación

5.1. *Landing page*

Es la página web mostrada al acceder a <https://cultivandoconocimiento.org.ar>. En esta se muestra información de la organización, cómo asociarse, una breve descripción de las instalaciones, algunas de las genéticas que trabajan, una sección de contacto para que los interesados puedan contactarse con la asociación y finalmente un botón para acceder a la cuenta del socio o administrador (login). Parte de los datos e imágenes que se muestran en la *landing page* se obtienen de la base de datos, por lo que los administradores pueden modificarlas en cualquier momento.

5.2. Backoffice

Es únicamente accesible por los administradores luego de iniciar sesión con su usuario (email) y contraseña. Cuenta con las siguientes secciones:

5.2.1. Usuarios

Desde esta sección, los administradores pueden crear o editar socios, dar de alta nuevos administradores, almacenar su información, crear sus categorías (las cuales indican cuántos gramos por mes pueden pedir a la asociación y por lo tanto el valor de la cuota social), subir los contratos firmados con ellos y subir su documento de inscripción en el REPROCANN. También se listan los socios paginados con filtros para facilitar la búsqueda.

5.2.2. Productos

Desde esta sección se pueden crear, editar, ocultar y eliminar productos, además de crear y editar sus categorías. Los productos cuentan con las imágenes que son mostradas en el catálogo para que los socios puedan solicitarlos. También se muestra el stock total, el reservado en pedidos y el disponible. Debido a que los directores dijeron que la asociación cuenta con pocos productos (menos de 40) no fue necesario implementar filtros o paginadores.

5.2.3. Pedidos

Desde esta sección se pueden ver los diferentes pedidos realizados, se puede obtener la información de un pedido en particular, como su estado, los productos asociados y el costo equivalente en gramos al momento de realizar el pedido. Se cuenta con filtros y paginadores para facilitar la búsqueda. Además los administradores pueden confirmar los pedidos, cancelarlos o marcarlos como entregados.

5.2.4. Sedes

Desde esta sección se pueden crear o eliminar sedes, las cuales son elegidas por los socios al momento de retirar el pedido.

5.2.5. Turnos

Desde esta sección se pueden crear o eliminar los turnos para las distintas sedes. Los turnos se crean para una sede, en determinado día por franjas horarias. Los turnos creados se muestran en una tabla con filtros para facilitar la búsqueda. Se pueden filtrar por sede y por día. Además, se puede visualizar si un turno está ocupado por un pedido y ver el pedido asociado.

5.2.6. Pagos

Desde esta sección se puede ver si cada socio ya pagó su cuota y se pueden generar automáticamente las órdenes de pago para los diferentes socios. Originalmente, se propuso a los directores tener una API para que los socios puedan pagar directamente desde la aplicación, pero prefirieron simplemente un sistema con el cual se puedan subir comprobantes de pago luego de realizar las transferencias. También pueden subir la factura correspondiente por el pago de la cuota social para que los socios la puedan descargar.

5.2.7. Métricas

Desde esta sección se pueden ver diferentes métricas sobre el uso de la aplicación. Entre ellas se encuentran socios nuevos en los últimos 12 meses, cantidad de socios según estado, cantidad de socios por categoría, facturación en función del tiempo, cantidad de pagos realizados, entre otras.

5.2.8. Sitio

Desde esta sección se puede modificar información de la *landing page*, como información de contacto, imágenes de las diferentes genéticas a mostrar, poner al sitio en mantenimiento y configurar la información de las cuentas bancarias a las cuales se hacen las órdenes de pago.

5.3. Vista de socio

Para acceder a la vista de socio es necesario haber sido dado de alta por un administrador y luego autenticarse. Una vez en la vista se puede acceder a las siguientes secciones:

5.3.1. Información

Desde esta sección se puede ver la información subida por los administradores, el estado del REPRO-CANN, modificar la contraseña y solicitar recategorizaciones. Hay una cantidad limitada de información que un socio puede editar sin necesidad de un administrador, dado que esta es utilizada para generar órdenes de pago y contratos.

5.3.2. Catálogo

Desde esta sección los socios pueden ver los distintos productos con los cuales abastecerse, información sobre cuántos gramos restantes pueden utilizar para su siguiente pedido y agregar productos al carrito.

5.3.3. Pedidos

Desde esta sección pueden visualizar los pedidos realizados, ver la información de un pedido en particular, como su estado, los productos asociados y lugar de entrega. También pueden cancelar el pedido dependiendo de su estado.

5.3.4. Carrito

Desde esta sección, se puede acceder al carrito, ver información sobre los requisitos que debe cumplir el pedido, elegir la sede y turno en los que se retira el pedido y finalmente realizar el pedido.

Una vez realizado el pedido, se reserva el stock de los productos y los administradores pueden verlo y confirmarlo para eventualmente entregarlo o cancelarlo.

5.3.5. Pagos

Se pueden visualizar las órdenes de pago y subir comprobantes de pago así como descargar la factura por la cuota social si estuviese disponible. También se visualizan los datos bancarios para que se puedan realizar los pagos, cuya información se actualiza desde el Backoffice, sección Configuración.

5.3.6. Contacto

Desde esta sección se pueden ver las diferentes formas de contacto con la asociación. Esta información también es configurable desde el Backoffice.

6. Implementación

Habiendo tenido el alcance del proyecto y las funcionalidades que se iban a necesitar, se decidió implementar una arquitectura orientada a microservicios, en la cual existe un cliente que es una aplicación web y un backend distribuido. Cada microservicio del servidor tiene su propia base de datos SQL utilizando PostgreSQL como gestor. Los distintos microservicios exponen una API que sigue las convenciones REST[5] y siguen las convenciones de códigos de retorno de HTTP establecidas por el RFC 9110[6].

Para la autenticación de usuarios, el servicio Usuarios se comunica con el componente Authentication de Firebase, una empresa de Google.

Para el almacenamiento de imágenes y documentos (como el REPROCANN, contratos u órdenes de pago) se utilizó el componente Storage, también de Firebase.

Para el desarrollo del cliente y los servicios del servidor se utilizó GitHub como herramienta de control de versiones, para poder gestionar las distintas versiones del proyecto y poder trabajar en conjunto entre los integrantes del grupo.

Se utilizaron contenedores Docker tanto para las aplicaciones como para las bases de datos, ya sean de pruebas o para utilizar durante el despliegue. Esto nos permitió reducir los tiempos de configuración y despliegue de los distintos entornos de desarrollo, pruebas y puesta en producción, además de otros beneficios que provee la herramienta relacionados con la escalabilidad, seguridad, etc. [7]

6.1. Aplicación Web

Angular[8] es un framework de desarrollo de aplicaciones Frontend de código abierto mantenido por Google Inc., sigue una arquitectura basada en componentes, lo que permite desarrollar componentes reutilizables. Utiliza el lenguaje tipado TypeScript que se compila a JavaScript. Angular cuenta con componentes que hacen un desarrollo muy eficiente y fácilmente escalable, como Guards para el control de acceso a rutas e Interceptors para la configuración y manipulación de las solicitudes HTTP.

Además de los componentes core de Angular, se utilizó la librería PrimeNG para la implementación de componentes de interfaz de usuario como calendarios, modales, etc. Para el diseño de formas, colores y layouts se utilizó TailwindCSS.

Se siguieron los estándares de formato y accesibilidad recomendados por la herramienta analizadora de código de Angular.

Esta aplicación interactúa con los microservicios del servidor para realizar las consultas HTTP a

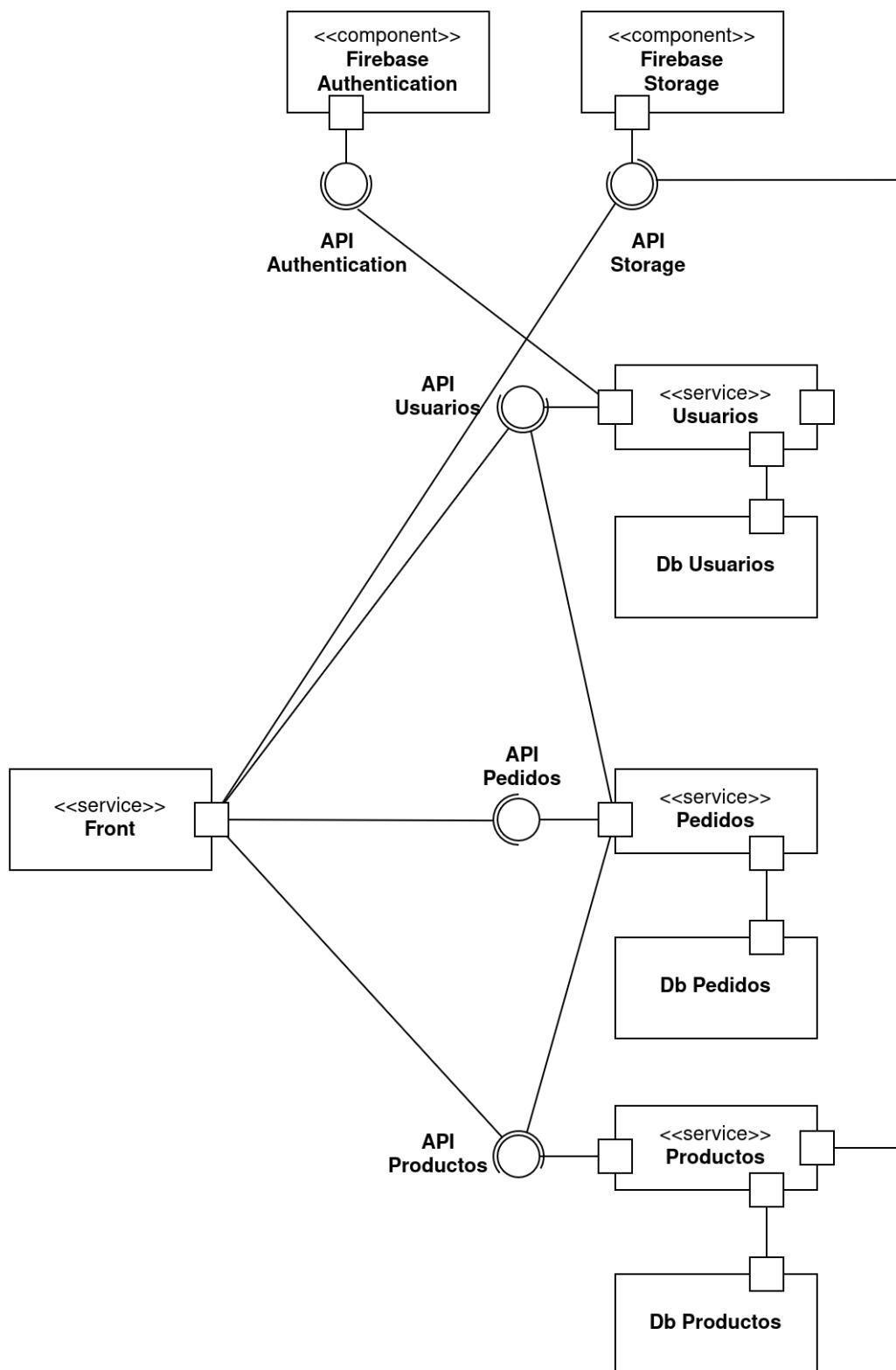


Figura 1: Arquitectura de la aplicación.

través de las APIs documentadas. Además, interactúa con el Storage de Firebase para subir las imágenes mediante URLs firmadas[9] generadas por el servicio Productos, de manera que no cualquiera pueda subir imágenes. Cabe aclarar que una URL firmada es una URL que proporciona permisos y tiempo limitados para realizar una solicitud.

6.2. Servicio Usuarios

Este servicio se encarga de todas las tareas relacionadas con la creación, lectura, edición y eliminación de socios y administradores, en delegar la autenticación a Firebase y en subir a Firebase los documentos .pdf de los socios.

Fue implementado en Ruby, utilizando el framework Padrino. Tiene una API RESTful para las distintas consultas HTTP. Está conectado a una base de datos SQL propia.

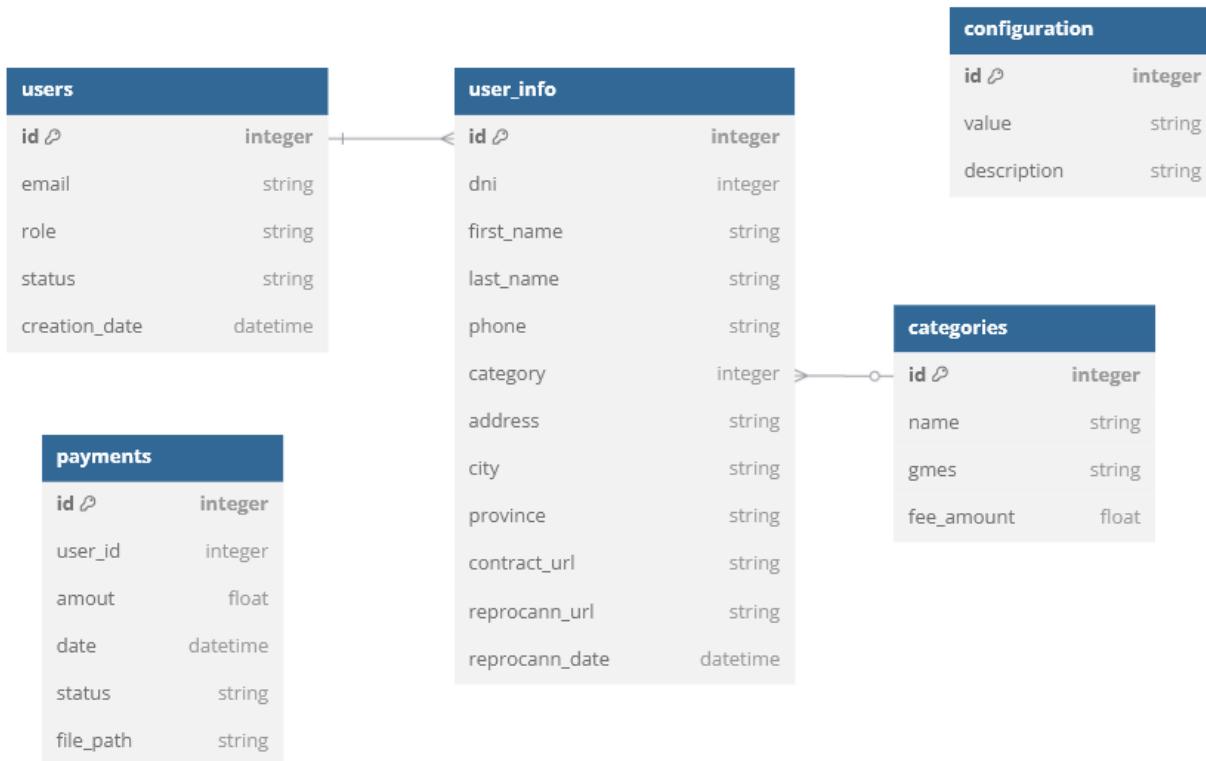


Figura 2: Base de datos del servicio usuarios.

6.3. Servicio Productos

Este servicio implementa todas las tareas relacionadas con la creación, lectura, edición y eliminación de productos junto con las posibles categorías. Mantiene un stock actualizado, considerando el total, el

reservado y el disponible. Además se encarga de crear las URLs firmadas para subir o descargar imágenes del servicio Storage de Firebase. Cuenta con una base de datos SQL propia, teniendo una para pruebas y otra para la aplicación desplegada.

Fue implementado en Ruby, utilizando Sinatra para su API RESTful. Fue testeado utilizando RSpec, con una cobertura mayor al 95 % de acuerdo a la herramienta SimpleCov, utilizando *mocking* para Storage de Firebase.

En el repositorio de GitHub se establecieron acciones de integración continua, verificando que la última versión siempre pase todas las pruebas.

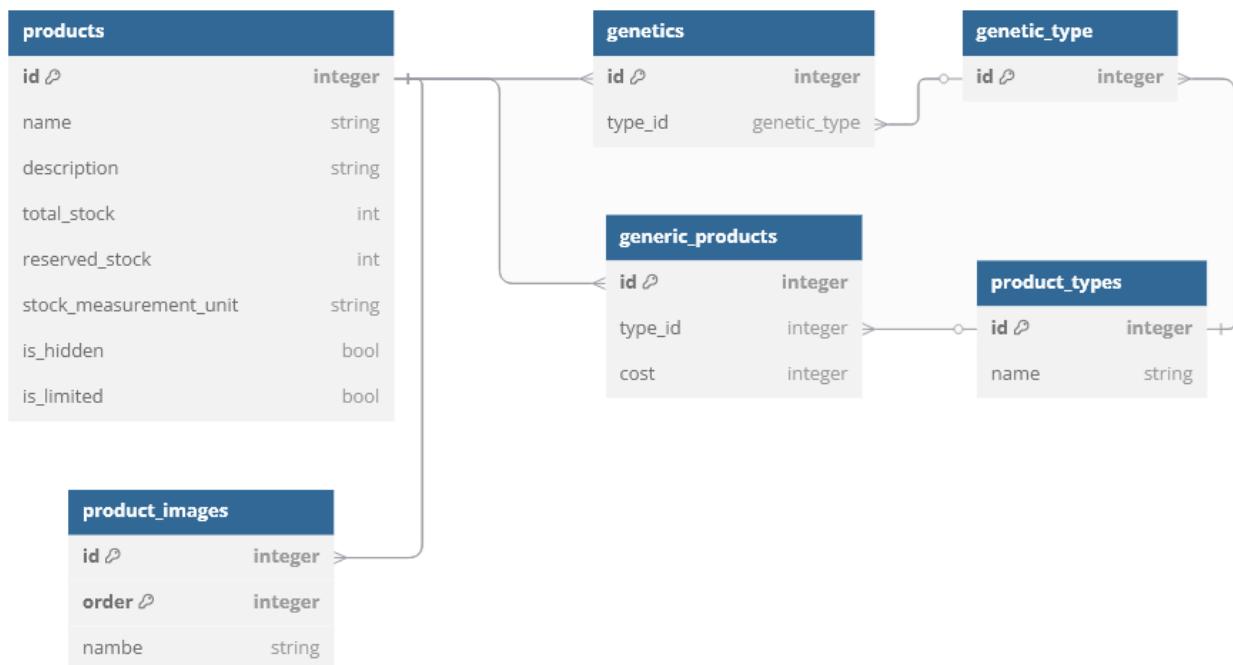


Figura 3: Base de datos del servicio productos.

6.4. Servicio Pedidos

Al igual que productos este servicio fue implementado en Ruby utilizando Sinatra. Posee una cobertura de pruebas mayor al 80 % y se utilizaron mocks para probar la interacción con Productos y Pedidos. También tiene su propia base de datos SQL.

Sus principales responsabilidades son las de gestionar y validar los pedidos, mantener el carrito de los socios, gestionar las sedes y los turnos; todo sin almacenar información repetida con Productos ni con Usuarios para evitar inconsistencias.

Al momento de querer obtener un pedido, este servicio se comunica con el de Productos y Usuarios

para obtener la información pertinente a estos y así devolver el pedido completo. También se encarga de informar a Productos cuándo es que hay que disminuir o reservar el stock luego de haberse hecho, cancelado o entregado un pedido.

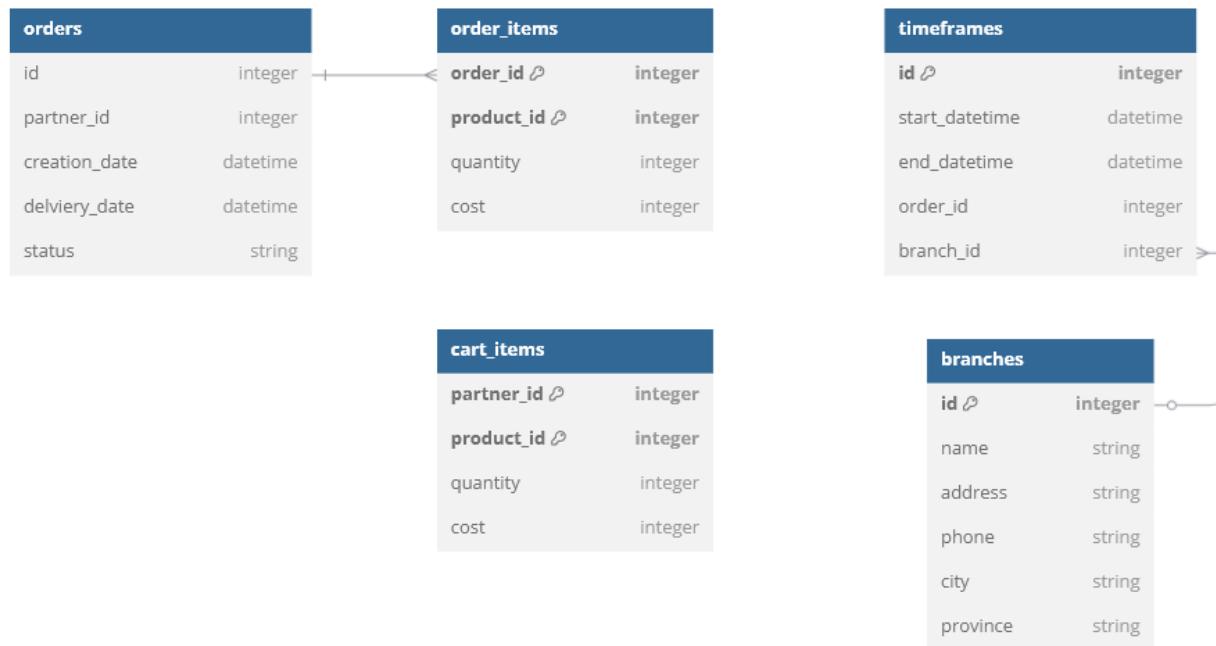


Figura 4: Base de datos del servicio pedidos.

6.5. Implementaciones destacadas

De las varias funcionalidades desarrolladas, algunas son de particular interés, ya sea por cómo interactúan los servicios entre sí o su complejidad.

6.5.1. Autenticación y autorización

Firebase Authentication es un servicio gratuito provisto por la empresa de Google Inc. que permite autenticar a los usuarios con distintos métodos, como correo y contraseña, Google, Facebook, X, etc. Utiliza el protocolo estándar de autenticación OAuth 2.0 y OpenId Connect. Utiliza tokens criptográficos JWT para autenticar a los usuarios y algoritmo SHA-256. Se optó por utilizar el método de correo y contraseña ya que es ampliamente utilizado y el que mejor se adapta a las necesidades del proyecto.

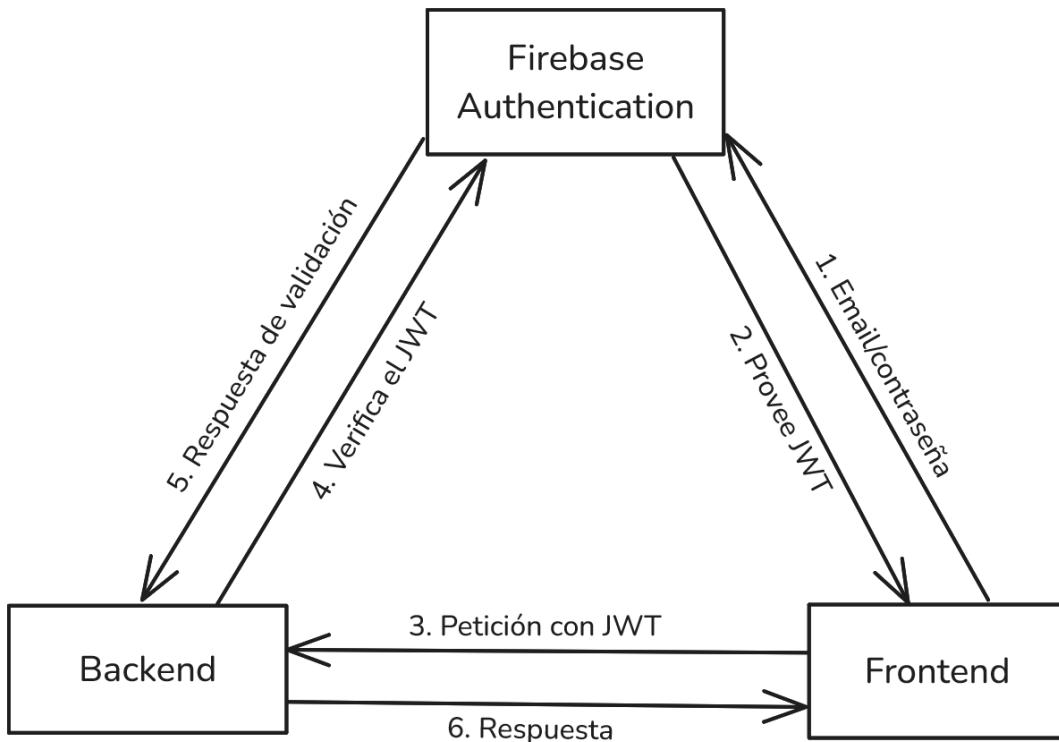


Figura 5: Flujo de autenticación.

Todos los servicios tienen validaciones de roles para ejercer un control sobre los recursos a los que pueden acceder, para lo cual se utiliza la información codificada en el token para identificarlos y autorizarlos.

6.5.2. Subida de imágenes y documentos

La subida de las imágenes al crear productos se realiza mediante el servicio Productos. Primero se envía una solicitud con la información del producto y los nombres de sus imágenes. Productos solicita a Firebase la creación de URLs firmadas para la subida del archivo de cada imagen y luego le devuelve a la aplicación web dichas URLs. Finalmente la aplicación web las utiliza para subir las imágenes a Firebase. Las URLs firmadas tienen una duración de 5 minutos y se generan con la información del nombre del archivo y el tipo de archivo que se va a subir, de esta manera se evita que cualquier usuario pueda subir imágenes al Storage de Firebase.

El mecanismo de subida de las imágenes de la *landing page* es similar.

Para subir documentos se utilizó el SDK de Firebase Cloud Storage para Ruby. Los documentos en formato PDF se suben directamente desde el servicio Usuarios. Estos pueden ser contratos, inscripción al REPROCANN, comprobantes de pago o facturas.

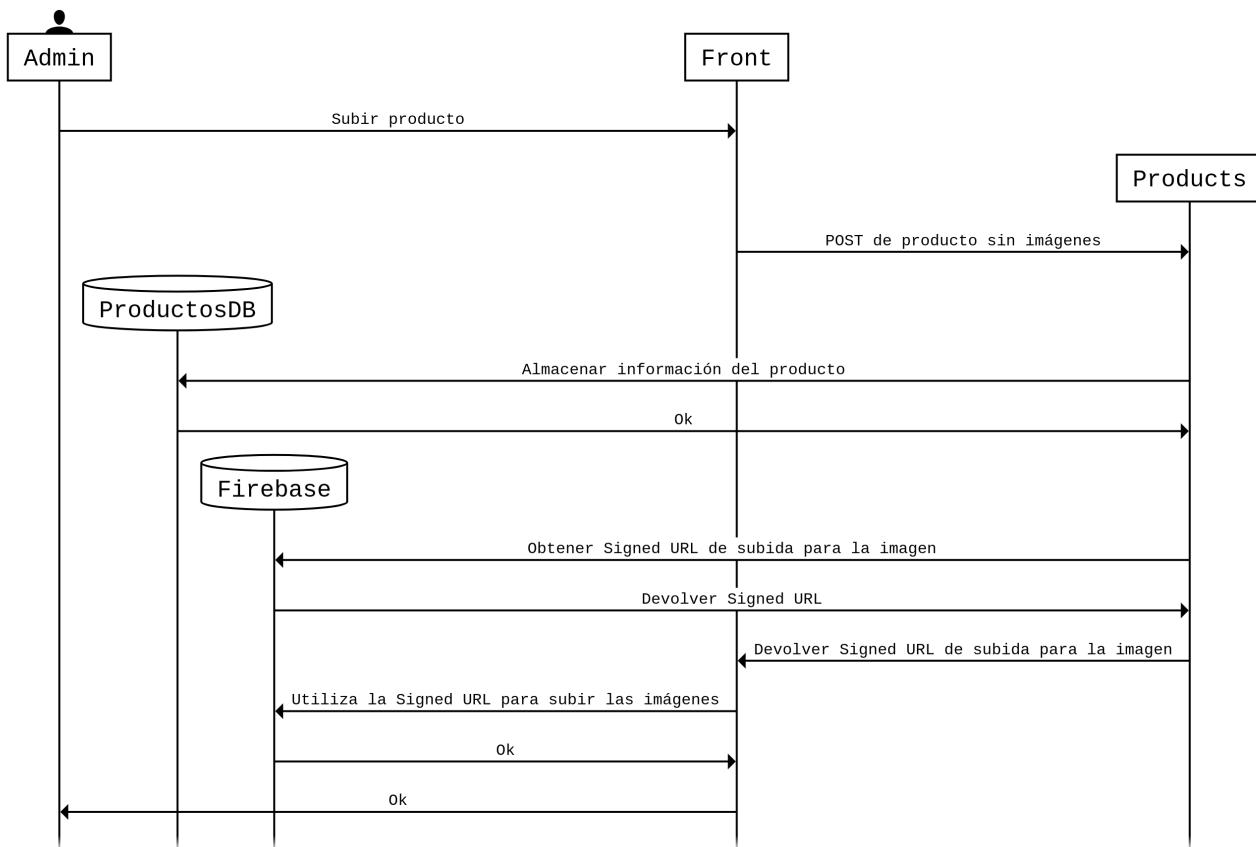


Figura 6: Diagrama de secuencia para subir un producto desde la Web App.

6.5.3. Descarga de imágenes y documentos

Siendo que las imágenes del catálogo del producto no son públicas, cuando una socio decide ver el catálogo, el servicio Productos envía una solicitud a Firebase para que genere URLs firmadas para la descarga de las imágenes. Luego se envían las URLs firmadas a la aplicación web y esta se encarga de mostrar la imagen en el catálogo. De esta manera se evita que cualquier usuario pueda descargar las imágenes del catálogo.

Para la descarga de las imágenes de la *landing page* no se utiliza ningún mecanismo de seguridad, siendo que estas son públicas.

Para la descarga de documentos, el servicio Usuarios solicita las URL firmadas a Firebase Cloud Storage que luego se envían al Frontend para que el usuario las pueda descargar durante los próximos 5 minutos.

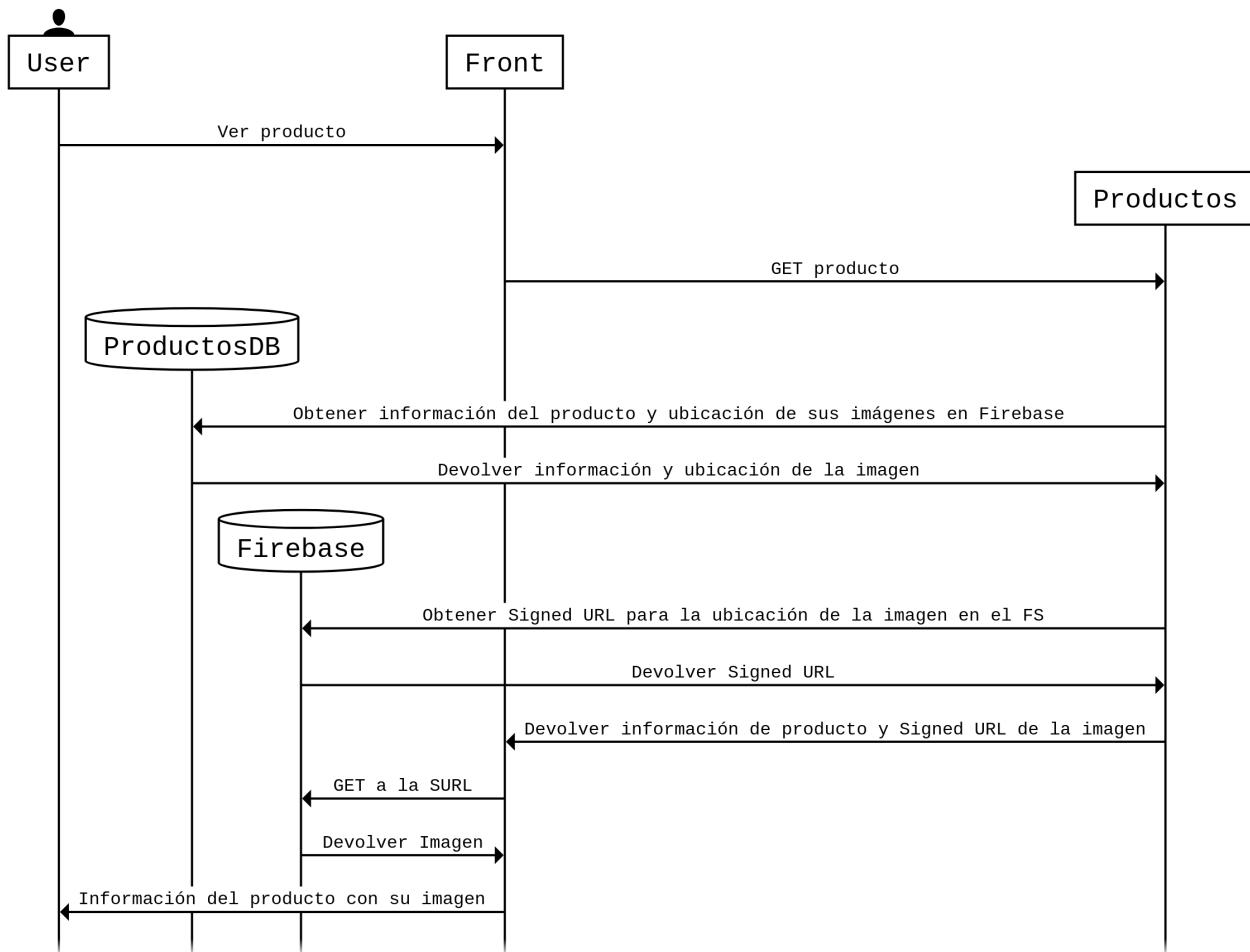


Figura 7: Diagrama de secuencia para ver un producto desde la Web App.

Esta implementación permite que únicamente los socios puedan ver y descargar las imágenes del catálogo, evitando el reenvío de las imágenes a través de varios servicios, disminuyendo la latencia.

7. Despliegue

Como práctica en el ciclo de vida del desarrollo y operaciones(DevOps) se utilizó CI/CD o más comúnmente conocido como *Continuous integration* y *Continuous Delivery* para lo cual se configuraron pipelines en GitHub Actions. Esto permitió desplegar los contenedores de los servicios de manera automática en los servidores de Google Cloud Run.

La ventaja de utilizar Google Cloud Run radica principalmente en la escalabilidad y la capacidad de manejo de recursos de manera automática, lo que permitió desplegar los servicios de manera eficiente y sin preocupaciones por la configuración y mantenimiento de los servidores. Además, al ser un servicio *serverless* permite a la asociación optimizar los costos y pagar solamente por los recursos utilizados. Se pudieron elegir la cantidad de recursos como RAM y CPU sumado a la cantidad de instancias consideradas adecuadas.

Otra cuestión importante es que permite realizar el monitoreo en tiempo real de los servicios, permite obtener métricas de uso y errores para identificar problemas y tomar medidas para mejorar la calidad y eficiencia del servicio. También se tiene a disposición a disposición los registros de errores o actividad(log) de los distintos servicios.

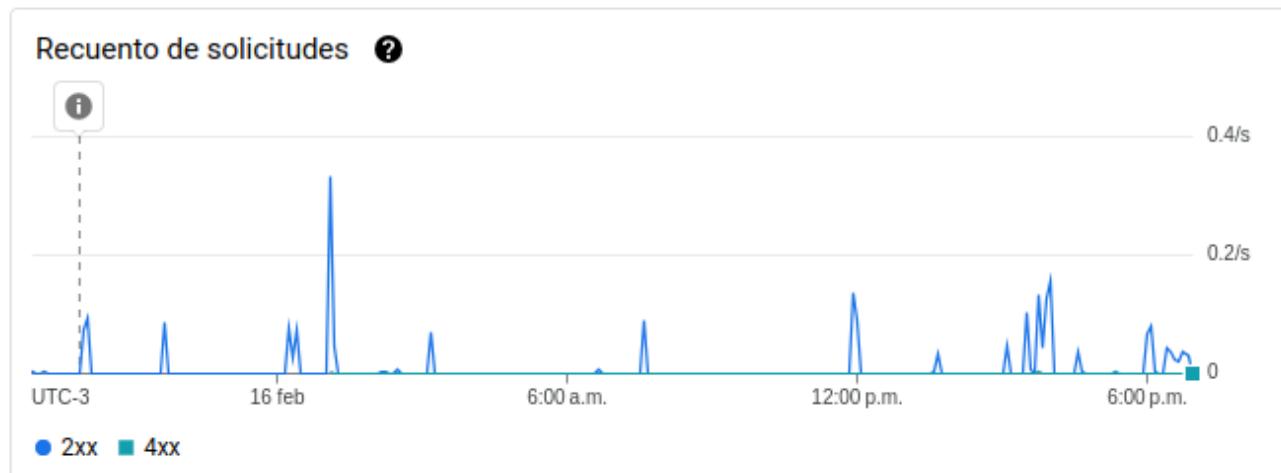


Figura 8: Captura de Google Cloud Run, solicitudes.

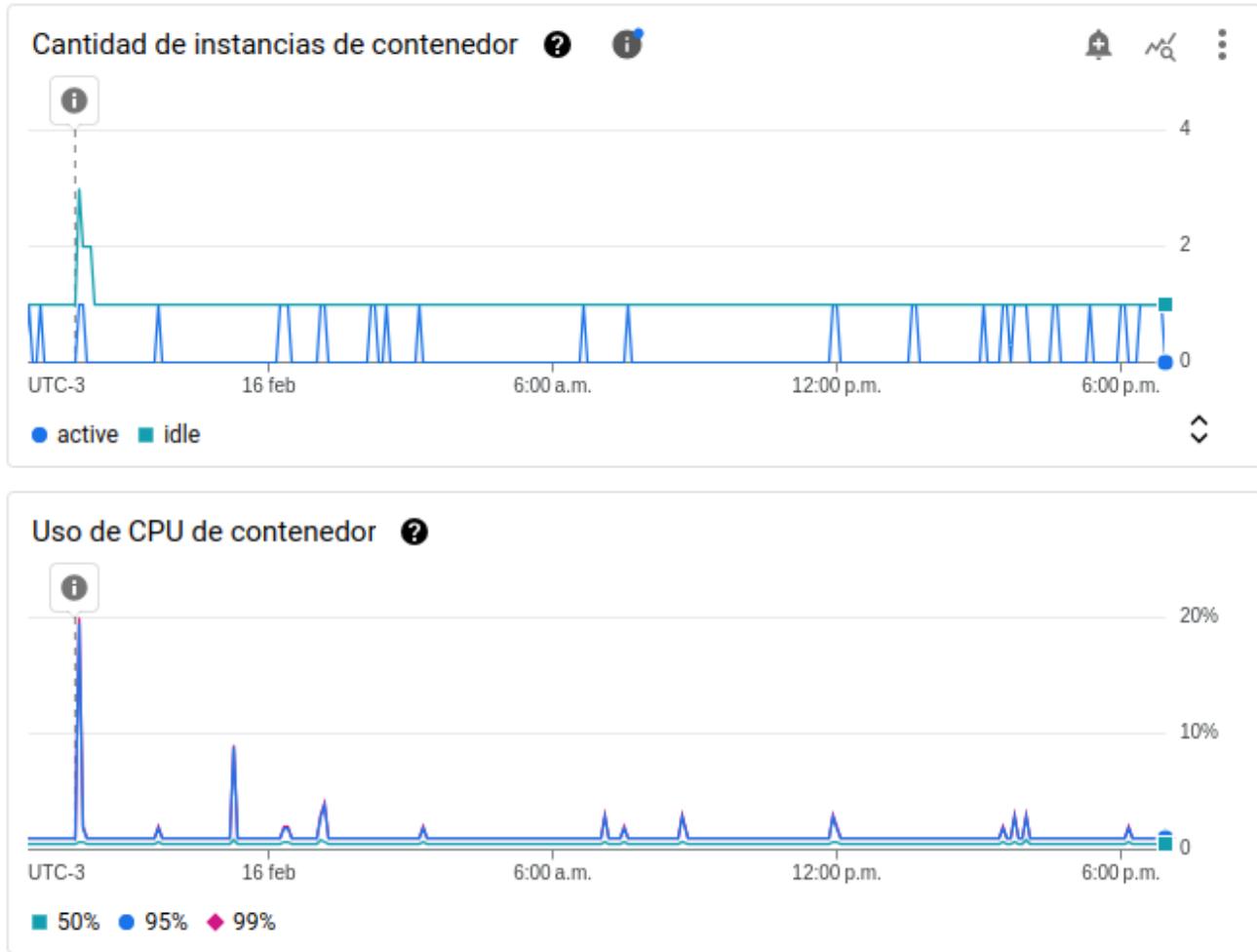


Figura 9: Captura de Google Cloud Run, instancias y uso de CPU.

7.1. Compra del dominio

El nuevo dominio de la asociación es cultivandoconocimiento.org.ar. A pesar que la asociación ya contaba con un dominio propio (cultivandoconocimiento.com), se les recomendó presentar los documentos necesarios para poder registrar un dominio .org.ar, principalmente porque siempre hicieron énfasis en que no eran una asociación comercial y este nuevo dominio de nivel superior (TDL o Top Level Domain en inglés) refleja mejor su situación. Además, les otorga reconocimiento y confianza, identidad propia y es más barato que un dominio .com.

A través de nic.ar se los asistió en la presentación de la documentación, en la realización del registro del nuevo dominio y en cambiar los registros DNS en el registrador de dominios(NIC.ar).

7.2. Bases de datos

Como se utilizaron servicios serverless se buscaron proveedores de bases de datos relacionales que sean confiables y a su vez accesibles en precios. Debido a esto, se utilizaron los servicios Render Postgres, una empresa que ofrece bases de datos relacionales en la nube. Se contrataron tres instancias de bases de datos para los servicios Usuarios, Pedidos y Productos.

Para la gestión de estructura de las bases de datos se utilizaron *migrations*[10] de Active Record. Es una funcionalidad que provee el framework de Ruby para cambiar la estructura de la base de datos sin afectar los datos almacenados. Los cambios se ejecutan cuando se despliegan las aplicaciones. Además, en caso de errores, permite revertir los cambios realizados en las bases de datos sin necesidad de eliminar toda la base de datos y migrarla desde cero, minimizando el impacto en la disponibilidad de la aplicación.

7.3. Almacenamiento de documentos e imágenes

Para el almacenamiento de documentos e imágenes se utilizó el servicio de Firebase Cloud Storage, una empresa de Google Inc. que ofrece un servicio de almacenamiento de archivos en la nube. Debido al tamaño de los archivos, se optó por utilizar el plan gratuito.

7.4. Servicios

Los servicios de Usuarios, Productos y Pedidos fueron desplegados en Google Cloud Run, el costo de los servicios se basa en la cantidad de recursos utilizados. Como se mencionó anteriormente, el despliegue es automático mediante GitHub Actions.

Los servicios están accesibles desde las siguientes URLs:

- <https://usuarios-1008730422331.us-central1.run.app>
- <https://productos-1008730422331.us-central1.run.app>
- <https://pedidos-1008730422331.us-central1.run.app>

8. Dificultades encontradas

Extensiones de las imágenes

Siendo que el almacenamiento de las imágenes de Productos se realiza en el componente Storage de Firebase, es necesario indicar el tipo de archivo a subir mediante el header “Content-Type” de HTTP. Surgió un problema con el tipo de archivo “jpeg”, debido a que el tipo de archivo es ese, pero la extensión del archivo puede ser “jpeg” o “jpg”.

La solución encontrada fue simplemente verificar si la extensión era jpeg o jpg y en ambos casos establecer header de la solicitud HTTP como jpeg.

¿Dónde ensamblar los Pedidos?

Antes de implementar el servicio de Pedidos, surgió el debate de dónde ensamblar a los pedidos, los cuales también tienen información de usuarios y de productos. Se pensó en ensamblar en la Web app directamente, debido a que desacoplaba a los microservicios entre ellos; y entre ensamblarlos en pedidos, debido a que dicha responsabilidad parecía corresponder a Pedidos. Luego de varias charlas entre los integrantes del grupo y una charla con el Tutor, se decidió por la última.

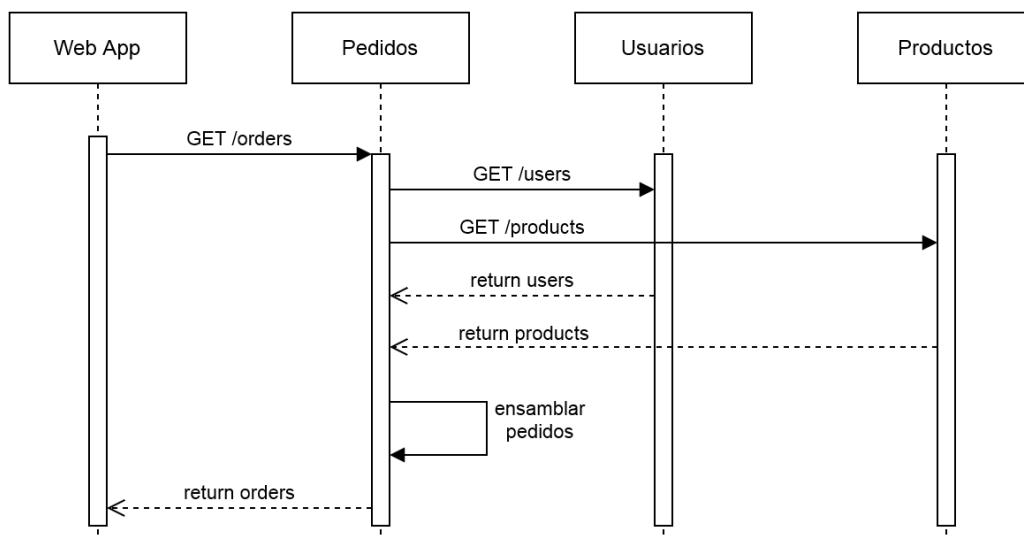


Figura 10: Diagrama de secuencia de ensamblaje de los pedidos en el servicio Pedidos.

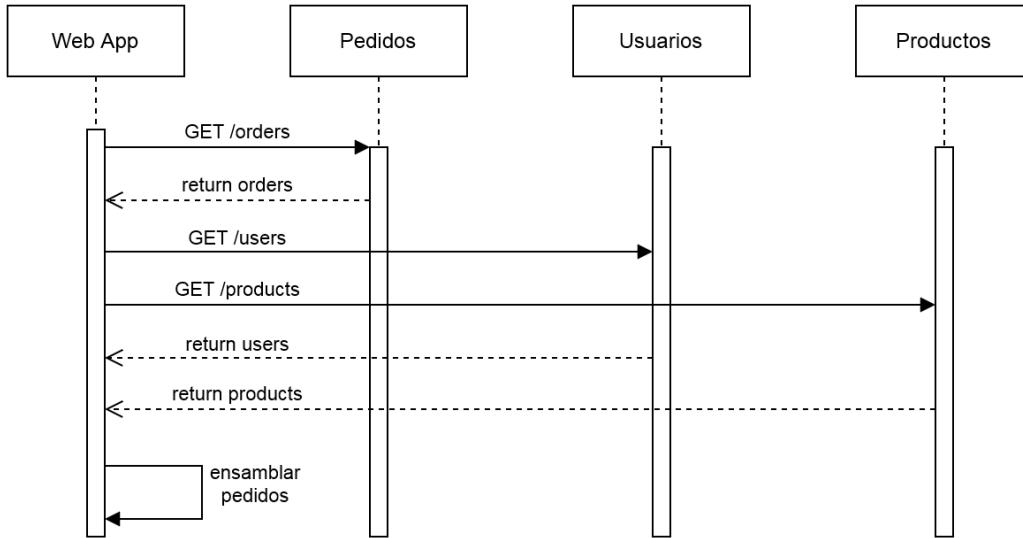


Figura 11: Diagrama de secuencia de ensamblaje de los pedidos en la aplicación web.

Comunicación entre microservicios

La comunicación entre los distintos servicios se llevó a cabo mediante APIs. Documentar estas APIs fue un proceso que requirió una cantidad de tiempo considerable, ya sea por escribir pruebas (especificaciones) o por utilizar herramientas como OpenAPI, con el fin de definir con precisión los datos que cada servicio debía recibir.

Si bien este enfoque resultó más lento en comparación con la comunicación interna de un sistema monolítico, se logró implementar de manera efectiva. Se considera que la escalabilidad y la flexibilidad de la arquitectura elegida justifica el esfuerzo requerido en su implementación.

Realizar los ambientes de pruebas

Durante la configuración de los entornos de prueba, se utilizaron *mocks* para evaluar la interacción entre los distintos componentes del sistema, dado que este incluye un cliente, tres servicios propios y dos componentes de Firebase. Esto implicó que, ante cualquier modificación en la funcionalidad de un servicio fuese necesario actualizar las pruebas de integración de los demás para reflejar dichos cambios.

Inicialmente, establecer un entorno de pruebas local presentó dificultades, especialmente entre la comunicación del servicio Pedidos con Usuarios y Productos. La solución adoptada fue la creación de una red de Docker en la cual se conectaron los tres servicios mencionados, permitiendo así evaluar su interacción de manera efectiva en un entorno controlado.

Para la simulación de los componentes de Firebase, en lugar de realizar *mocks* directamente sobre

su API, se optó por aplicarlos a las clases conectoras que encapsulan su interacción, facilitando así la abstracción y el mantenimiento del código de prueba.

TailwindCSS + PrimeNG

Para el desarrollo de la aplicación web, se decidió utilizar TailwindCSS para la definición de estilos, complementándolo con diversos componentes de PrimeNG. Inicialmente se presentaron dificultades al intentar personalizar los estilos de PrimeNG mediante TailwindCSS, dado que se sobreescritían estilos entre sí. La solución adoptada consistió en modificar directamente las clases de PrimeNG utilizando CSS y dando más importancia a TailwindCSS, permitiendo así una integración adecuada entre ambas tecnologías.

Este problema de integración fue resuelto finalmente por PrimeNG en su última versión, pero no en la que se utilizó en el proyecto.

Servicios de hosting

A la hora de hacer el despliegue se realizaron configuraciones de DNS en los registradores de dominios y demás configuraciones en el proveedor del servicio de hosting, como registros de tipo A y AAAA para asociar los nombres de dominio con las direcciones IP. La intención era que el proyecto esté desplegado en el dominio cultivandoconocimiento.org.ar y no con el nombre de dominio poco legible como el que provee Google Cloud Run.

Agotamiento de la cuota diaria de Storage de Firebase

Durante las pruebas destinadas a verificar la compatibilidad del sistema con archivos de extensión .jpg y .jpeg, se alcanzó el límite diario de carga de archivos permitido por el plan gratuito de Firebase Storage.

Para mitigar esta limitación, se optó por utilizar archivos de menor tamaño en las pruebas y se esperó un período de veinticuatro horas antes de continuar con las evaluaciones.

Siendo que esta situación podría ocurrirle a los administradores, se les informó acerca de cómo comprimir las imágenes, la recomendación de utilizar archivos .jpeg debido a su mayor comprensión con menor pérdida de calidad y que en caso que quieran aumentar el límite de subida deben contratar una versión paga.

Cross Origin Requests

Durante el desarrollo de la aplicación web, se presentó un problema cuando la aplicación intentaba realizar peticiones a los servicios, debido a que estas son consideradas *Cross Origin Requests*. La mayoría de navegadores web como Firefox, Chrome o Edge validan que los servicios las acepten. En principio estas validaciones fallaban al no haber sido configuradas. La solución adoptada consistió en configurar los servicios Usuarios, Productos y Pedidos para permitir solicitudes desde cualquier origen.

9. Seguridad

Se optó por utilizar varios servicios de Firebase que ya cuentan con algunas medidas de seguridad sofisticadas, como la autenticación de usuarios, la gestión de roles y la protección contra ataques de falsificación de solicitudes. También se utilizaron protocolos HTTPS para garantizar la seguridad de la comunicación entre los distintos servicios.

La comunicación entre los microservicios y las bases de datos utilizan una comunicación cifrada utilizando el protocolo SSL/TLS[11], un requisito estándar en la industria.

9.1. JWT (JSON Web Tokens)

- **Seguridad:** Los JWT[12] están firmados digitalmente, lo que garantiza que la información contenida en el token no ha sido alterada.
- **Portabilidad:** Son pequeños y pueden ser transmitidos fácilmente a través de parámetros POST, URL o cabeceras HTTP.
- **Autenticación y autorización:** Permiten verificar la identidad de los usuarios y autorizar el acceso a recursos específicos sin necesidad de almacenar información sensible.
- **Escalabilidad:** Son ideales para aplicaciones distribuidas, ya que no requieren almacenamiento de estado en el servidor.

9.2. OAuth de Firebase

- **Autenticación:** OAuth[13] Proporciona una solución fácil y segura para la autenticación de usuarios.
- **Seguridad mejorada:** Gestiona la autenticación y la autorización, reduciendo la carga de seguridad en el lado del desarrollador.
- **Protección contra ataques:** Ofrece medidas de seguridad adicionales, como la protección contra ataques de falsificación de solicitudes entre sitios (CSRF).
- **Actualizaciones automáticas:** Se mantiene actualizado con las últimas medidas de seguridad, asegurando que la aplicación esté protegida contra las amenazas más recientes.

10. Accesibilidad

Uno de los puntos claves del proyecto fue elegir una paleta de colores adecuada para poder diseñar una interfaz web accesible y una experiencia de usuario inclusiva.

Para esto se utilizaron herramientas como Color-Hex [14] y Adobe Color[15] para encontrar una paleta de colores que se ajustara a las necesidades de la ONG y que a su vez fuera accesible para todos los usuarios. Algunos de los puntos más destacados son:

- Contraste de colores: se buscó un contraste adecuado entre texto (negro) y fondo (blanco/gris suave) tomando como color principal en algunas secciones al verde dado que este era el color que representaba a la ONG.
- No depender únicamente del color: los enlaces o botones también incluyen subrayados, iconos, etiquetas o cambios de opacidad y animación para indicar su función.
- Consistencia y percepción: mantener una paleta de colores consistente ayuda a los usuarios a navegar por el sitio de manera más intuitiva y mejora la percepción de los elementos interactivos, facilitando la diferenciación entre secciones y funciones.

Así mismo, se tuvo en cuenta el uso de los íconos que proveía PrimeNG para identificar botones clásicos (como el de carrito de compras y otros) para que el cliente pueda navegar más rápido con la sola identificación visual.

Otro de los puntos importantes del proyecto fue el diseño responsive, para que de esta manera usuarios con distintos equipos puedan acceder a la web y abastecerse de productos. Entre los beneficios de este tipo de diseño se encuentran:

- Mejora la experiencia de usuario con una navegación óptima en distintos dispositivos.
- Captura y retiene usuarios que acceden desde dispositivos *mobile*.
- Mantiene una única versión del sitio web.
- Google favorece los sitios web responsive en sus resultados de búsqueda.

11. Futuro del proyecto

Se espera que la ONG Cultivando Conocimiento se pueda beneficiar directamente con la solución desarrollada. Se continuará brindándole el soporte tecnológico necesario para mantener la plataforma funcionando y con la posibilidad de expandir las funcionalidades durante los primeros meses.

11.1. Entrevista con los directores de Cultivando Conocimiento

Luego del despliegue del proyecto y de la capacitación del personal de Cultivando Conocimiento, se realizó una entrevista con los directores de la ONG para conocer sus expectativas y cómo esperan que el proyecto impacte en su organización.

¿Cómo esperan que los afecte este proyecto?

Esperamos que el proyecto tenga un impacto positivo en la eficiencia operativa de la ONG. Al automatizar la generación de pedidos y profesionalizar la comunicación con los pacientes, podremos reducir el margen de error y mejorar la experiencia para el usuario brindando respuestas rápidas y precisas y generando una plataforma clara y accesible con un mayor caudal de información disponible para el usuario en todo momento.

¿Creen que van a minimizar los tiempos?

Sí, creemos que se podrán minimizar considerablemente los tiempos relacionados con tareas repetitivas como responder mensajes, gestionar pedidos y hacer seguimiento con los pacientes a la hora de generar los pedidos, lo que nos permitirá centrarnos en la mejora continua del proceso de cultivo y en la investigación para mejorar la calidad del cannabis medicinal.

¿Cuál es el mayor problema actual en el manejo de usuarios? ¿Creen que se solucionó?

El mayor problema actual es la gestión manual de la comunicación con los pacientes, lo cual genera retrasos y riesgo de errores, además de ser difícil de escalar.

Creemos que la implementación de esta solución automatizada enmendará en gran medida este problema, al garantizar una comunicación más organizada y eficiente, permitiendo un mejor control y atención personalizada cuando sea necesario.

¿Qué van a hacer con el tiempo y trabajo que les ahorraría esta app?

El tiempo y trabajo que se ahorre se invertirá en mejorar los procesos relacionados con el cultivo y en incrementar la calidad del cannabis medicinal. También se destinará a la investigación para nuevos tratamientos y a la optimización de la logística para asegurar que los pacientes reciban sus tratamientos de manera puntual.

Además, se podrá dedicar más tiempo a la formación del personal y a la creación de alianzas con otras entidades para expandir nuestra capacidad de atender a más pacientes.

11.2. Soporte técnico

Para garantizar el correcto funcionamiento de la aplicación y la satisfacción de los clientes, se ofrecerán tres meses de soporte técnico como garantía. Este soporte incluirá:

- Correcciones de errores.
- Ajustes menores en la interfaz de usuario.

En caso que los directores decidan continuar con el soporte técnico, se ofrecerá un contrato de mantenimiento.

Por otro lado, si los directores decidieran seguir expandiendo la aplicación, se ofrecerán servicios de desarrollo, soporte técnico y consultoría que incluirán:

- Implementación de nuevas funcionalidades.
- Soporte técnico ilimitado.
- Asesoramiento en la implementación de nuevas funcionalidades.

11.3. Oferta de la aplicación a otras organizaciones

Una vez que la plataforma esté en funcionamiento y se haya demostrado su efectividad, se la ofrecerá a otras organizaciones similares que necesiten una solución para gestionar la comunicación con sus pacientes, adaptando la interfaz a la visión de cada organización.

Luego de una breve investigación se observó que la mayoría de las organizaciones que ofrecen cannabis medicinal cuentan con una *landing page* pero no con una plataforma virtual como la desarrollada. Se espera que la aplicación sea bien recibida en el mercado y que se pueda expandir a otras organizaciones en el futuro.

12. Conclusiones

La realización del presente Trabajo Profesional fue un proceso largo y demandante, en el cual cada uno de los integrantes dedicó más de cuatrocientas horas entre programar, configurar los repositorios, desplegar los servicios, investigar las mejores opciones para el hosting, investigar sobre las distintas herramientas disponibles para implementar la solución, generar documentación, reuniones entre el equipo de desarrollo, reuniones con los directores de Cultivando Conocimiento, realización de documentación a la comisión curricular del Departamento de Informática de la Facultad de Ingeniería de la Universidad de Buenos Aires y su seguimiento, entre otras tareas.

A lo largo de este proceso se pudieron aprender y profundizar conocimientos en diversas herramientas, verificar empíricamente los beneficios o problemas de ciertas implementaciones, comprender mejor el funcionamiento de servidores web y sobre técnicas de desarrollo.

Las conclusiones principales que se obtuvieron a partir de este proyecto son las siguientes:

- La arquitectura orientada a microservicios cuenta con el beneficio de la escalabilidad, pero ralentiza el proceso de implementar nuevas funcionalidades debido principalmente a no poder comunicar objetos de distintos servicios directamente, sino que se tienen que realizar mediante una API, la cual hay que implementar y documentar, ya sea con pruebas o herramientas externas.
- La arquitectura orientada a microservicios permite una mayor separación de las funciones y la comunicación entre los distintos servicios, lo cual facilita el mantenimiento y el desarrollo de nuevas funcionalidades. Si en un futuro se necesita un nuevo servicio se pueden implementar sin afectar al resto de la arquitectura.
- El proceso de planificación de un proyecto de desarrollo de software es una tarea compleja. Dadas las etapas del proceso como el descubrimiento de los requisitos o necesidades de la asociación, el planeamiento de la arquitectura de la aplicación, desarrollo, documentación, cambios relacionados con el alcance, elección de la infraestructura, capacitación, etc. hacen que sea difícil establecer la cantidad de horas totales que se deben dedicar a un proyecto. En el caso de este proyecto se dedicaron más horas de las esperadas.
- La utilización de pruebas unitarias y de integración permitió un avance más veloz debido a no tener que perder tiempo realizando pruebas manuales ante las nuevas funcionalidades implementadas y daban seguridad de no haber implementado errores nuevos al funcionar como pruebas de regresión.
- El desarrollo de pruebas automatizadas End to End entre varios microservicios, a pesar de ser útiles, no justificaba el tiempo invertido en crearlas. En cambio la creación de pruebas de integración

utilizando mocks y las eventuales pruebas End to End manuales resultaron más eficientes.

- Siendo que la arquitectura era orientada a microservicios y los integrantes del equipo de programación son tres, de las diversas maneras de planificar y organizar el desarrollo, la utilización de historias de usuarios, cada una con sus casos de uso resultó la más conveniente.

La división por historias de usuario se realizaba entre los distintos integrantes durante las reuniones y luego cada uno implementaba los casos de uso que luego eran validados por el resto del equipo.

- Hay que ser paciente, saber escuchar y comunicarse con quienes establecen los requisitos, en este caso los directores de la ONG. Surgieron situaciones en las que fue difícil entender qué era lo que realmente querían y luego de algunas reuniones se llegó a un acuerdo sobre qué implementar.

Un ejemplo de esto fue la implementación del cobro de la cuota mensual, ya que en principio se pensó en implementar un sistema de pagos con tarjeta de crédito, pero luego se decidió implementar un sistema de pagos con transferencia bancaria y suba de comprobantes.

- La búsqueda sobre la mejor oferta de dónde desplegar los servicios es una tarea ardua, siendo que lleva tiempo comparar los diferentes precios de los distintos proveedores y sus beneficios.
- A veces encontrar la solución óptima no justifica el tiempo que se tarda en llegar a ella y conviene implementar una solución más genérica pero que igualmente funciona.

Por ejemplo, en el servicio Productos se implementaron las bases de datos considerando que existen genéticas y otros productos, cuando implementarlo teniendo únicamente productos hubiese tardado menos tiempo a pesar de no ajustarse perfectamente a las reglas de negocio explicadas (por ejemplo, los otros productos no hace falta obtenerlos en múltiplos de 5).

- Desarrollar correctamente una aplicación web requiere mucho más esfuerzo del pensado, debido a que hay que lidiar con cuestiones de seguridad como las Cross Origin Requests, devolver los códigos de mensaje establecidos en las RFCs, entre otros.
- El diseño de una interfaz de usuario útil y atractiva demora más de lo previsto, siendo que hay que elegir una paleta de colores, tener en cuenta los distintos tamaños de ventana, que sea utilizable por lectores de pantalla para ofrecer accesibilidad a personas con problemas de visión, entre otros.

13. Referencias

- [1] Joaquin Herrero. Transformación digital en la sociedad civil de américa latina: desafíos, oportunidades y el rol de las herramientas digitales, 2023.
<https://www.lacnic.net/innovaportal/file/7215/1/version-final-joaquin-herrero.pdf>.
- [2] Ministerio de Salud Argentino Dirección Nacional de Medicamentos y Tecnología Sanitaria. Pre-guntas frecuentes reprocann.
<https://www.argentina.gob.ar/salud/cannabis-medicinal/frecuentes>.
- [3] GitHub Inc. Github projects, 2025.
<https://docs.github.com/en/issues/planning-and-tracking-with-projects/learning-about-projects/about-projects>.
- [4] Código civil y comercial de la nación [ccyc], artículo 168, octubre 2014. Argentina.
- [5] Lokesh Gupta. What is rest?, 2023. <https://restfulapi.net/>.
- [6] M. Nottingham R. Fielding and J. Reschke. Http semantics. RFC 9110, RFC Editor, junio 2022.
<https://datatracker.ietf.org/doc/html/rfc9110>.
- [7] Ian Smalley Stephanie Susnjara. What is docker?, 2024.
<https://www.ibm.com/think/topics/docker>.
- [8] IdeaBlade. What is angular?, 2023. <https://v17.angular.io/guide/what-is-angular>.
- [9] Google Inc. Signed urls, 2025.
<https://cloud.google.com/storage/docs/access-control/signed-urls>.
- [10] Ruby on Rails. Active record migrations, 2025.
https://guides.rubyonrails.org/active_record_migrations.html.
- [11] E. Rescorla. The transport layer security (tls) protocol version 1.3. RFC 8446, RFC Editor, agosto 2018. <https://datatracker.ietf.org/doc/html/rfc8446>.
- [12] Firebase. Create custom tokens, 2025.
<https://firebase.google.com/docs/auth/admin/create-custom-tokens>.
- [13] Firebase. Firebase authentication, 2025. <https://firebase.google.com/docs/auth?hl=es-419>.
- [14] HbApps. Color hex, 2025. <https://color-hex.com>.
- [15] Adobe. Adobe color, 2025. <https://color.adobe.com/es/create/color-wheel>.

A. Wireframes

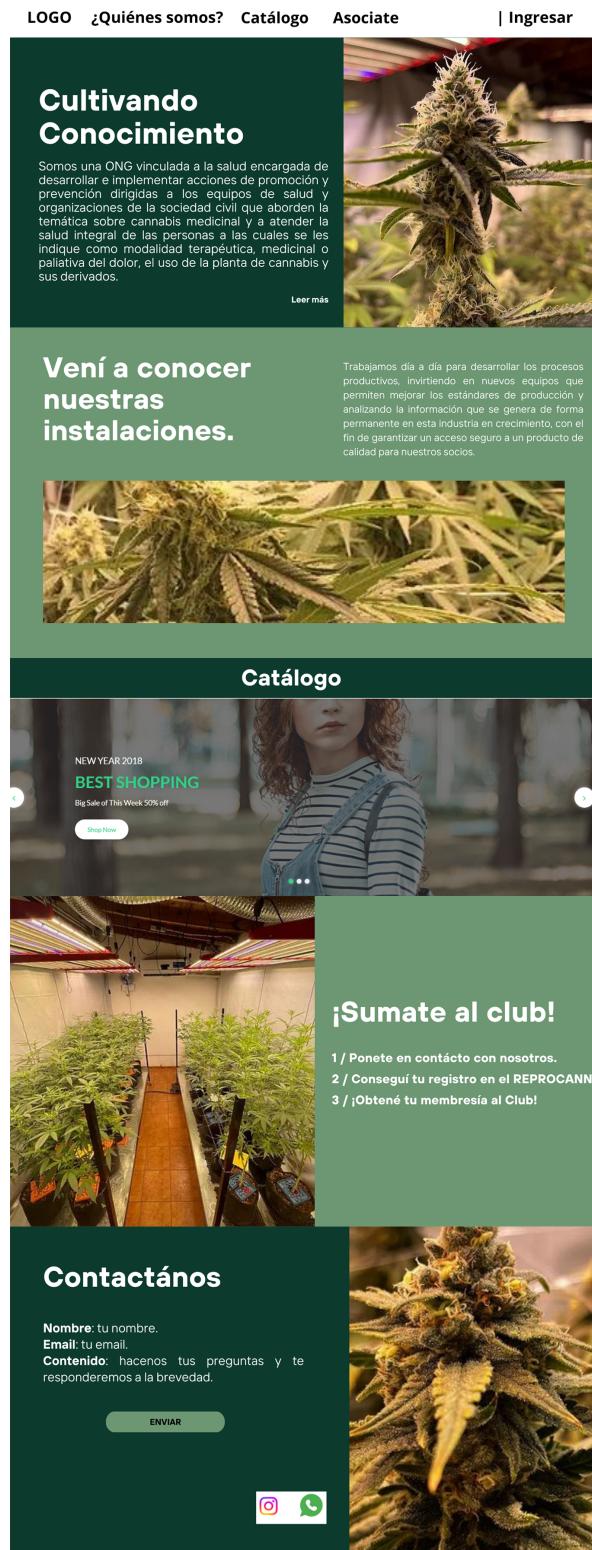


Figura 12: Wireframe de la *landing page*.

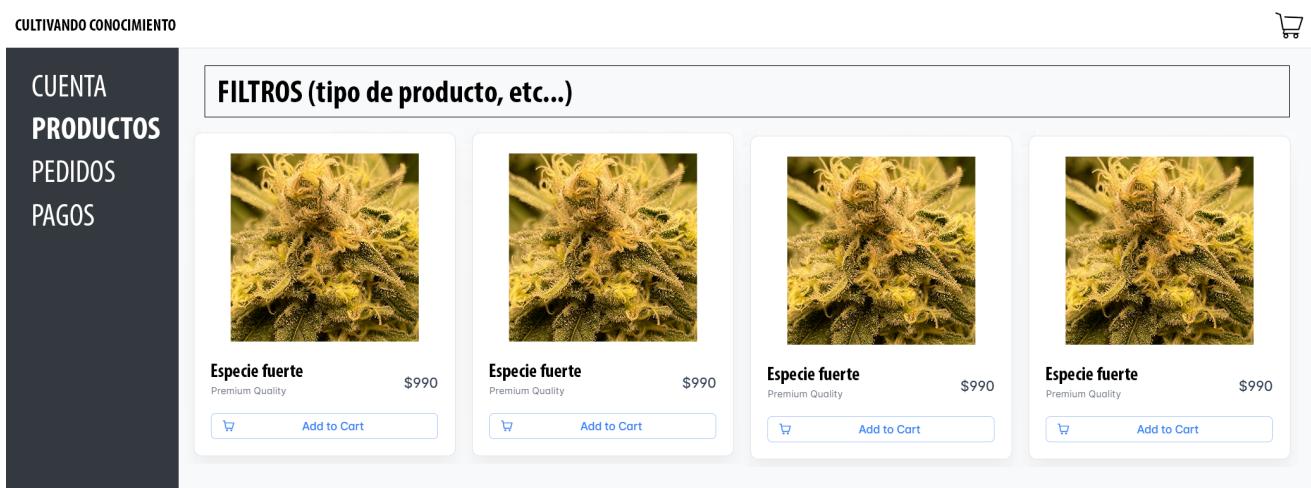


Figura 13: Wireframe del catálogo de productos.

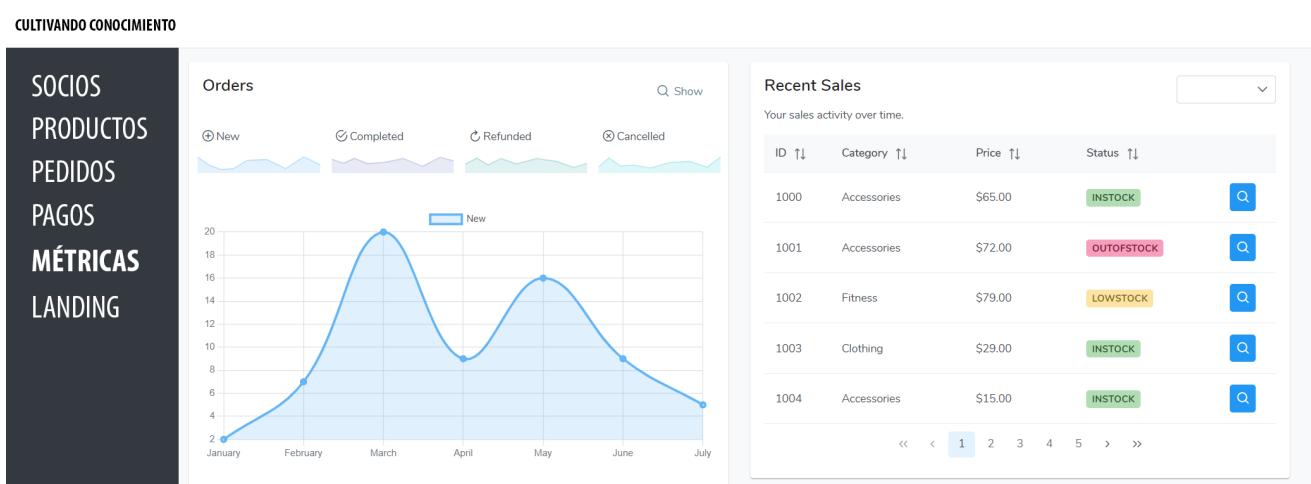


Figura 14: Wireframe de métricas visto desde la página web.

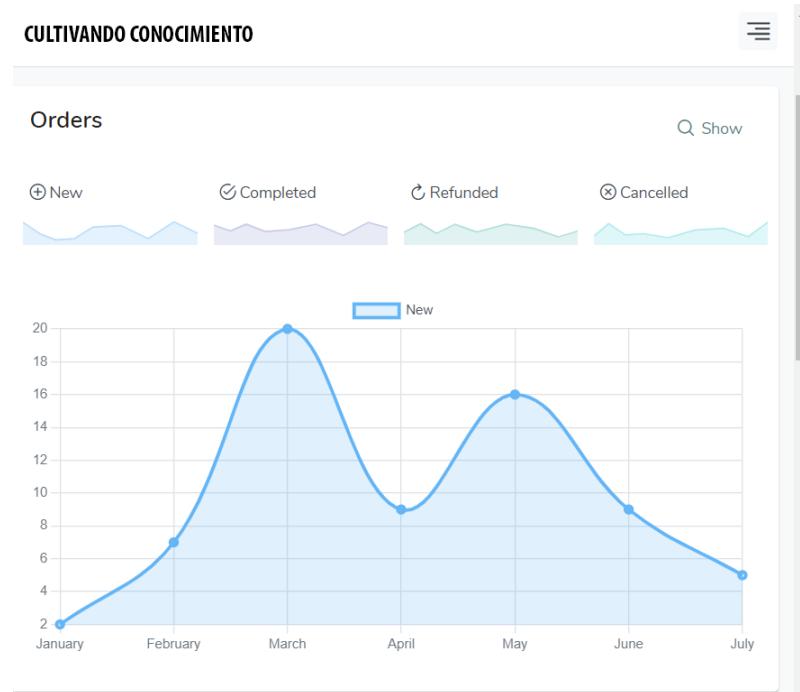


Figura 15: Wireframe de la vista *mobile* de métricas.