

**Universidad ORT Uruguay**

**Facultad de Ingeniería**

**Gestor de reclamos para Mercado Libre**

Entregado como requisito para la obtención del título de Analista Programador

**Antonieta Álvarez – 237628**

**Florencia Fernández – 233908**

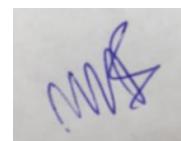
**Tutor: Susana Abulafia**

**2020**

## Declaración de autoría

Nosotros, Antonieta Álvarez y Florencia Fernández, declaramos que el trabajo que se presenta en esa obra es de nuestra propia mano. Podemos asegurar que:

- La obra fue producida en su totalidad mientras realizábamos el Proyecto final de carrera para la obtención del título Analista Programador.
- Cuando hemos consultado el trabajo publicado por otros, lo hemos atribuido con claridad;
- Cuando hemos citado obras de otros, hemos indicado las fuentes. Con excepción de estas citas, la obra es enteramente nuestra;
- En la obra, hemos acusado recibo de las ayudas recibidas;
- Cuando la obra se basa en trabajo realizado conjuntamente con otros, hemos explicado claramente qué fue contribuido por otros, y qué fue contribuido por nosotros;
- Ninguna parte de este trabajo ha sido publicada previamente a su entrega, excepto donde se han realizado las aclaraciones correspondientes.



Antonieta Álvarez

Florencia Fernández

11/05/2020

11/05/2020

## Abstract

El siguiente documento describe el desarrollo del sistema Gestor de reclamos para Mercado Libre, el mismo es una API REST que se desarrolló con el fin de integrar el manejo de los reclamos sobre las ventas realizadas desde la plataforma de Mercado Libre (ML) al *software* existente del cliente (Sherlock Assistant). Principalmente se cumplió con la necesidad que tiene el cliente de tener una API para integrar a dicho *software*, además se realizó la construcción de un sistema de *frontend* inicialmente para prueba de la misma y con fines académicos. Posiblemente el *frontend* en un futuro evolucione para integrarlo también a la herramienta del cliente.

El usuario final de esta aplicación es cualquier vendedor que utilice la plataforma de Mercado Libre (ML) como medio de venta de sus artículos.

Este *software* permite al usuario gestionar sus reclamos de ML, cuenta con el listado de los reclamos ordenados de tal forma que le permite al usuario tomar acciones sobre los reclamos que es prioritario resolver, a su vez cuenta con filtros para poder encontrar los reclamos de forma más rápida. También puede realizar todas las acciones que se pueden hacer desde la plataforma de ML a la hora de gestionar los reclamos que se encuentran abiertos. Por último, pero no menos importante, le ofrece al usuario la posibilidad de ver datos relevantes sobre el historial de sus reclamos.

Las tecnologías utilizadas para el desarrollo del mismo son .NET Core, MVC, XUnitTest, Bootstrap, SQL Server sobre Azure, GitHub y Swagger.

Se optó por una metodología ágil para la gestión del proyecto, utilizando el modelo iterativo e incremental.

## Palabras clave

- ML (Mercado Libre)
- Reclamo
- Reputación
- Mediación
- Xpert Solution
- Sherlock Solutions
- Sherlock Assistant
- PNR (pagado y no recibido)
- PDD (producto defectuoso o no es lo que esperaba)
- C#.NET
- MVC
- API
- Azure
- .NET Core
- SQL Server
- OAUTH2

# Índice

1.	Introducción.....	9
2.	Anteproyecto .....	10
2.1.	Introducción .....	10
2.2.	Presentación del cliente .....	10
2.3.	Presentación del problema .....	11
2.3.1.	Introducción.....	11
2.3.2.	¿Cómo gestionan hoy los reclamos a través de ML? .....	11
2.3.3.	Análisis del problema y solución propuesta .....	13
2.4.	Lista de necesidades.....	15
2.5.	Objetivos .....	16
2.6.	Actores involucrados .....	17
2.7.	Lista de requerimientos.....	18
2.7.1.	Requerimientos funcionales .....	18
2.7.2.	Requerimientos no funcionales .....	20
2.8.	Alcance y limitaciones.....	21
2.9.	Arquitectura - Particularidades .....	22
2.9.1.	Diagrama de arquitectura.....	22
2.9.2.	Diagrama conceptual .....	23
2.10.	Identificación de riesgos.....	24
2.11.	Plan de proyecto .....	26
2.11.1.	Metodología .....	26
2.11.2.	Descripción y selección de herramientas .....	27
2.11.3.	Plan de calidad .....	27

2.11.4.	Plan de configuración de software.....	27
2.11.5.	Plan de capacitación.....	27
2.11.6.	Sprint de planificación .....	28
2.11.7.	Cronograma de trabajo.....	29
2.11.8.	Diagrama de Gantt (actividades planificadas) .....	31
2.12.	Compromiso de trabajo .....	32
3.	Proyecto .....	33
3.1.	Introducción .....	33
3.2.	Pila del producto final.....	33
3.3.	Diseño final de la solución.....	34
3.3.1.	Documentos de análisis .....	34
3.3.2.	Documentos de diseño.....	35
3.3.3.	Arquitectura final de la solución .....	36
3.3.4.	Diagrama de tablas .....	37
3.4.	Desarrollo del proyecto.....	38
3.4.1.	Sprint 3 (18/05/2020 - 31/05/2020).....	38
3.4.2.	Sprint 4 (01/06/2020 - 14/06/2020).....	42
3.4.3.	Sprint 5 (15/06/2020 - 28/06/2020).....	46
3.4.4.	Sprint 6 (29/06/2020 - 12/07/2020).....	50
3.4.5.	Sprint 7 (13/07/2020 - 26/07/2020).....	54
3.4.6.	Sprint 8 (27/07/2020 - 09/08/2020).....	57
3.4.7.	Sprint 9 (10/08/2020 - 23/08/2020).....	61
3.4.8.	Sprint 10 (24/08/2020 - 06/09/2020).....	65
3.4.9.	Sprint de cierre (07/09/2020 - 20/09/2020) .....	69
4.	Evidencia de ejecución de los planes .....	70
4.1.	Introducción .....	70

4.2.	Plan de calidad .....	70
4.3.	Plan de configuración de software .....	71
4.4.	Plan de capacitación.....	72
4.5.	Plan de riesgos .....	73
5.	Conclusiones.....	74
5.1.	Introducción .....	74
5.2.	Grado de satisfacción del cliente .....	74
5.3.	Lecciones aprendidas .....	75
5.4.	Posibles mejoras y desarrollos futuros.....	76
5.5.	Reflexiones finales.....	77
6.	Glosario .....	78
7.	Referencias .....	80
8.	Bibliografía .....	82
9.	Anexos .....	83
	Anexo 1 - Definición objetivos con el cliente .....	83
	Anexo 2 - Recursos para ver el diagrama de Gantt .....	83
	Anexo 3 – Evidencia de <i>testing</i> de la API .....	84
	Anexo 4 – Casos de prueba .....	87
	Anexo 5 – Validación del cliente sobre avance del producto ( <i>sprint 7</i> ) .....	123
	Anexo 6 – Validación del cliente sobre avance del producto ( <i>sprint 8</i> ) .....	124
	Anexo 7 – Validación del cliente sobre avance del producto ( <i>sprint 9</i> ) .....	125
	Anexo 8 – Validación del cliente sobre avance del producto ( <i>sprint 10</i> ) .....	126
	Anexo 9 – Evaluación de satisfacción del cliente .....	127
	Anexo 10 – Manual de usuario.....	129
	Anexo 11 – Manual de <i>deploy</i> .....	151
	Anexo 12 – Datos de prueba para correctores.....	154

Anexo 13 – Documentación de la API: ..... 156

## **1. Introducción**

El objetivo de este documento es presentar el detalle del desarrollo de una aplicación Web para la gestión de reclamos para Mercado Libre. Se presenta toda la documentación de la instancia de análisis y planificación del mismo, así como un detalle de la documentación generada en el transcurso de desarrollo e implementación del proyecto. Además, se incluye la evidencia de la ejecución de los planes, las conclusiones, lecciones aprendidas y posibles mejoras a implementar en un futuro.

## **2. Anteproyecto**

### **2.1. Introducción**

El objetivo de este capítulo es presentar en detalle las características del proyecto Gestor de reclamos para Mercado Libre (ML), que se va a desarrollar como proyecto final de la carrera Analista en Programación. Se presenta el cliente, las particularidades del negocio, el alcance del proyecto, los diferentes roles, así como los requisitos funcionales y no funcionales que se identificaron junto al cliente para poder cumplir con las necesidades y objetivos propuestos. Además, se detalla las metodologías a utilizar, el diseño de la solución, alcance del proyecto, riesgos con su posible mitigación y plan de contingencia, herramientas a utilizar y la planificación del mismo.

### **2.2. Presentación del cliente**

Los clientes de este proyecto son Hugo Olivera y Ricardo Melo socios y cofundadores de dos marcas distintas una es Xpert Solution y la otra es Sherlock Solutions desde el punto de vista legal es la misma empresa. Sus propietarios las manejan como “marcas diferentes” porque realizan diferente tipo de trabajo bajo cada una de estas marcas. Xpert Solution se dedica al desarrollo de *software* y consultoría en tecnologías de la información mientras que Sherlock Solutions se dedica específicamente a la creación de soluciones basadas en inteligencia artificial (IA).

En la actualidad cuentan con múltiples soluciones y proyectos en diferentes rubros, pero es importante destacar uno de ellos por estar relacionado con el proyecto que se va a desarrollar; Sherlock Assistant, es un *software* para la gestión de la venta y postventa de productos a través de ML. Esta herramienta hoy en día ya está comercializada en Uruguay, Argentina, Brasil, Chile y Méjico. Esta aplicación tiene integración con IA, es decir cuentan con una plataforma que con gestión automatizada permite manejar todo lo que es comunicación, referente a preguntas y respuestas en la preventa de un producto y luego una vez que se concreta la venta todo lo que es mensajería sobre los detalles de cuando se entrega el producto, la forma de entrega del mismo, si es necesario un número de *tracking*.

Tener una herramienta que gestione los reclamos que recibe un usuario vendedor de ML, los cuales se producen por parte del comprador después de una venta cuando hay algún problema o desconformidad, es de mucho interés para nuestros clientes. Esto se debe a que ML califica a los integradores (herramientas de software que ofrecen capacidad de trabajar sobre ML) y abarcar esta etapa del negocio sumado a las que ya tienen, hace que sea mayor su puntaje. Es de suma importancia tener una buena calificación como integrador ya que es uno de los pilares para aumentar la confianza de los vendedores de ML para comenzar a utilizar la herramienta que brindan.

## **2.3.Presentación del problema**

### **2.3.1. Introducción**

Actualmente existe la necesidad de desarrollar una solución para la gestión de los reclamos que puedan surgir de las ventas realizadas en ML, de esta manera el cliente lograría cubrir con todas las etapas o ciclos que podrían surgir de la venta de un producto. Cabe destacar que en la actualidad no cuentan con experiencia ni con una herramienta para la gestión de reclamos de ML.

El proceso de postventa de cualquier tienda que vende por medio de ML se encuentra con la necesidad de manejar los reclamos realizados por sus clientes. ML últimamente incorporó las políticas de devolución o cambio sin costo para el comprador por diversos motivos como ser entregas que no llegan, productos que llegan en mal estado, productos que no son lo que el cliente compró o no cumplen con sus expectativas. Estas políticas ya existen hace años en mercados de Estados Unidos o Europa, pero para los vendedores en Latinoamérica esto es un problema, porque genera costos de traslados además cuando los clientes solicitan un cambio de producto muchas veces el mismo ya está abierto y no siempre llega en buen estado al vendedor, por lo tanto, no lo puede volver a vender o lo debe vender a un precio menor que el original. Estos nuevos beneficios ML los gestiona a través de los reclamos que inician los compradores luego de la postventa. En los últimos tiempos los reclamos se multiplicaron para la mayoría de las tiendas, haciendo que cada vez sean más los usuarios que piden una solución a esta gestión integrada en la herramienta que ya cuentan nuestros clientes.

La gestión de reclamos para vendedores particulares o pequeños no conlleva un gran problema y lo manejan por medio de la plataforma de ML. Sin embargo, en tiendas o multi tiendas la gestión de estos reclamos sin contar con una herramienta más completa que ofrezca más funcionalidades a las que tiene ML hace que no sea una solución adecuada ni escalable. Esto puede llevar a que los reclamos queden perdidos o no sean resueltos en el tiempo adecuado, esta mala gestión de los reclamos puede producir pérdidas monetarias significativas y mala reputación para los vendedores. Para los vendedores la reputación (su calificación dentro de la plataforma de ML) es una de las cosas que más debe cuidar ya que ésta es uno de los factores que afecta positiva o negativamente sus ventas. Es importante destacar que para que ML no les baje este puntaje a los vendedores, las ventas que tengan reclamos deben ser menores al 2% del total de sus ventas en 3 meses.

### **2.3.2. ¿Cómo gestionan hoy los reclamos a través de ML?**

Hoy en día los usuarios gestionan sus reclamos a través de la plataforma de ML. Una vez que el vendedor confirma una venta como entregada, al comprador se le habilita la posibilidad de solicitar ayuda a ML e iniciar un reclamo. Los motivos para iniciar un reclamo son diversos: quiere cancelar la compra, aún no recibió el producto, recibió el producto con un problema, quiere devolver o cambiar el producto. Dependiendo el tipo el reclamo que inicie el comprador como éste repercute en el vendedor, ya que algunos bajan la reputación del vendedor dentro de ML. Los que afectan la reputación del

vendedor son todos excepto: cuando el comprador se arrepiente de la compra o cuando quiere un cambio.

Existen dos tipos de reclamos en ML estos pueden ser del tipo PNR (pagado y no recibido) o PDD (producto defectuoso o no es lo que esperaba) [1]. Y las resoluciones solicitadas por el comprador cuando inicia el reclamo pueden ser:

- devolución del importe (*refund*): espera que se devuelva el dinero (en cualquiera de los dos tipos) [1].
- quiere el producto (*product*): espera que le llegue el producto (en el tipo PNR) [1].
- cambio del producto (*change\_product*): espera que le cambien el producto (en el tipo PDD) [1].
- devolución del producto (*return\_product*): espera que se devuelva el producto con la posterior devolución del dinero (en el tipo PDD) [1].

Luego de que el comprador completa el flujo de información solicitada por ML, especificando el motivo por el cual inicia el reclamo el último paso es escribir un mensaje para el vendedor. En este momento el vendedor recibe el reclamo pudiendo ver el detalle del mismo y el mensaje, a su vez se le informa el tiempo que tiene para contestarle al comprador, de no cumplir el plazo el reclamo será cerrado por ML a favor del comprador, esto implica una pérdida monetaria para el vendedor ya que pierde el dinero de la venta y el producto. En el momento que el vendedor recibe un reclamo se inicia una etapa de negociación e intercambio entre él y el comprador para llegar a un acuerdo y cerrar el reclamo, el vendedor a través de esto procura tener la menor pérdida económica o de reputación.

Mantener una buena reputación dentro de ML es muchas veces más importante para los vendedores que las pérdidas económicas de una venta. El vendedor puede aceptar la solución propuesta por el comprador o proponer una nueva resolución, pero siguiendo estas reglas:

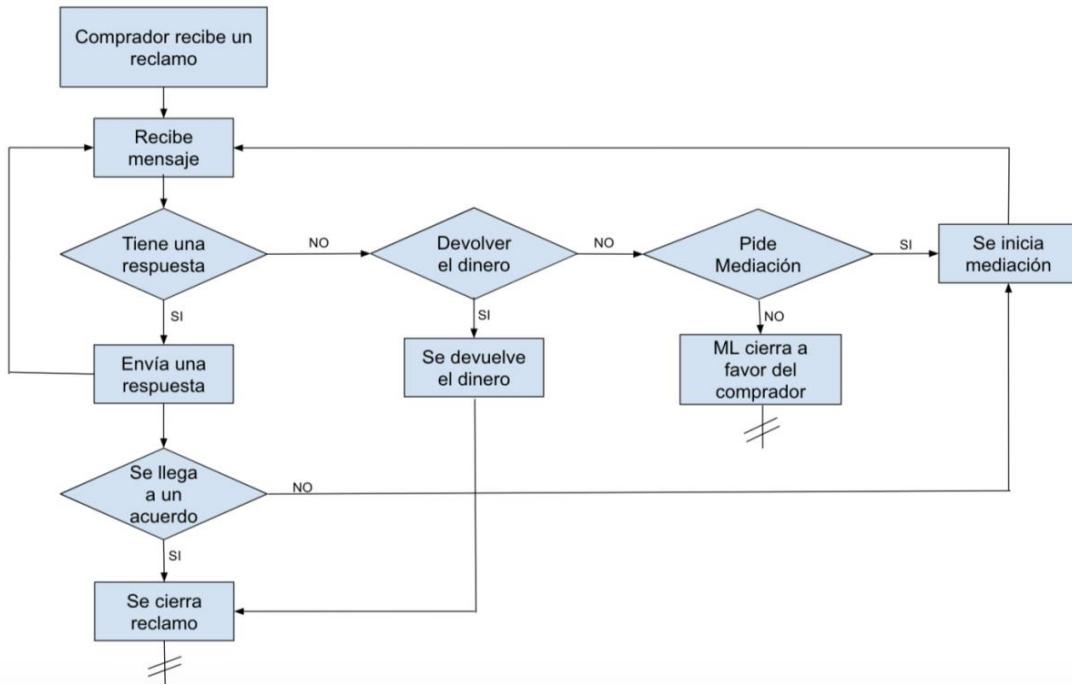
- Si el comprador elige *product*, el vendedor puede elegir *product* o *refund* [1].
- Si el comprador elige *refund*, el vendedor debe aceptar esa resolución [1].
- Si el comprador elige *change\_product*, el vendedor puede elegir *change\_product* o *return\_product* [1].
- Si el comprador elige *return\_product* debe aceptar esa resolución [1].

Las resoluciones *refund* y *return\_product* hacen referencia a que el comprador quiere la devolución del dinero, por este motivo sólo puede aceptar esa resolución. Se debe aclarar que ML hoy en día solo permite la gestión de la opción *refund* a través de su plataforma, no por medio de ningún integrador.

En caso de que esta negociación por mensajes directos (comprador - vendedor) no sea fructífera y no se acepte una resolución, cualquiera de las dos partes puede solicitar a ML la intervención como mediador, lo que significa llegar a la etapa de mediación. En este momento el comprador y vendedor ya no se puede enviar mensajes directamente,

sino que ML es quien lee e interpreta los mensajes buscando una resolución entre ambos, la mayoría de las veces lo hace a favor del comprador.

A continuación, se presenta un diagrama del flujo de la gestión de los reclamos en la plataforma de ML.



### 2.3.3. Análisis del problema y solución propuesta

Desde el punto de vista de análisis y reportes de estado de los reclamos, la plataforma de ML es muy precaria cuando no existente para estas capacidades. Poder tener estas funcionalidades sería muy valioso para que los vendedores puedan ver métricas sobre los reclamos que tienen sus ventas, también identificar de forma clara cuáles de esos reclamos repercuten en su reputación y tener la gestión de estos de una forma más clara ya que ML es muy precario en la presentación, orden y alerta de los reclamos.

Mediante este proyecto se busca crear una herramienta para gestionar y analizar reclamos recibidos por usuarios que venden por medio de la plataforma de ML, permitiendo a nuestros clientes tener una primera solución de gestión de reclamos la cual en un futuro puedan integrar a su sistema (Sherlock Assistant). La misma va a permitir a los usuarios ver de forma clara todos los reclamos abiertos y cerrados, brindando una herramienta de comunicación entre vendedores, compradores y mediadores. A su vez permitir al vendedor aceptar una resolución o proponer una nueva. Por otro lado, va a facilitar a los usuarios ver reportes sobre los reclamos y tener un *dashboard* donde ver las métricas de los mismos.

Esta herramienta es de suma importancia para nuestros clientes ya que podrán abarcados de los motivos principales que condiciona las ventas de sus usuarios; estas son su reputación, la comunicación e interacción con los clientes que implica la respuesta a los

mensajes de manera rápida y la calidad de las publicaciones y posicionamiento de las mismas.

## **2.4.Lista de necesidades**

- 2.4.1.** Contar con una herramienta que permita realizar la gestión integral de reclamos de ventas de ML con las mismas capacidades y flujos en sus distintas variantes al igual que en la plataforma de ML.
- 2.4.2.** Poder identificar el país del vendedor que está usando la herramienta y seguir los flujos o reglas de negocio definidas para los mismos en este proceso por ML.
- 2.4.3.** Obtener un conjunto de reportes sobre la gestión de reclamos.
- 2.4.4.** Permitir a los usuarios ver un *dashboard* con indicadores de reclamos que permitan el análisis en tiempo real.
- 2.4.5.** Permitir la interacción con la plataforma de ML a través de la conexión que ellos disponen para intercambiar información de los reclamos y sus usuarios.
- 2.4.6.** Contar con una arquitectura que permita interactuar con el sistema existente de Sherlock Assistant. Tener una estructura de datos compatible con la actual.

## **2.5.Objetivos**

- 2.5.1.** Disponibilizar las mismas capacidades de gestión de reclamos que se pueden hacer por medio del portal de ML, respetando los distintos flujos definidos por ML en base al tipo de reclamo generado por el comprador.
- 2.5.2.** Gestionar los reclamos con sus variantes para todos los países donde opera ML.
- 2.5.3.** Crear un conjunto de reportes que permita el análisis de los reclamos.
- 2.5.4.** Crear un *dashboard* con indicadores de reclamos que permitan el análisis en tiempo real de los mismos.
- 2.5.5.** Integración con la plataforma actual (Sherlock Assistant).

## **2.6. Actores involucrados**

- 2.6.1.** Clientes: principales interesados, su participación en el proyecto es constante, validan los requerimientos y la prioridad de los requerimientos.
- 2.6.2.** Potenciales usuarios: todos los vendedores que utilizan la plataforma de ML como medio de venta.
- 2.6.3.** Usuarios: personas registradas en nuestro sitio.
- 2.6.3.1.** Gerente: dueños de tiendas y multitiendas que utilizan la plataforma de ML como medio de venta. Van a ser afectados positivamente con el resultado de la ejecución del proyecto. Su desempeño comercial se ve involucrado directamente con el nuevo *software*. Tienen acceso a más funcionalidades del sistema.
- 2.6.3.2.** Operario: funcionarios de las tiendas y multitiendas que se encargan de la gestión del proceso de venta a través de ML. Su desempeño laboral se va a ver afectado de forma positiva ya que va a tener una optimización en la resolución de los reclamos con esta herramienta.
- 2.6.4.** Desarrolladores del proyecto: se encargan de llevar el proyecto adelante, gestionando los tiempos, stakeholders, riesgos y recursos.
- 2.6.5.** Consultor: Encargado de ventas de la empresa de nuestros clientes, conoce al detalle el negocio.
- 2.6.6.** Tutora: Interés en el proyecto desde el rol académico, acompañando en el proceso para poder realizarlo de forma exitosa logrando un buen producto, una buena gestión y una experiencia real aprovechable para la vida laboral.

## 2.7. Lista de requerimientos

### 2.7.1. Requerimientos funcionales

A continuación, se detallan los requerimientos funcionales, los mismos fueron priorizados por el cliente.

ID	NOMBRE	DESCRIPCIÓN	PRIORIDAD
R-01	<i>Login</i>	El sistema debe contar con un <i>login</i> para la autenticación de los usuarios, la información que se solicita a los usuarios es nombre de usuario y contraseña.	Baja
R-02	Autenticación del usuario en ML	El sistema debe permitir conectar al usuario con sus cuentas de ML y persistir en la base de datos todos los reclamos existentes de este usuario.	Alta
R-03	Manejo de Roles	El sistema debe contar con distintos roles que sean escalables y poder hacer el manejo de funcionalidades acorde a la categoría del rol.	Baja
R-04	Registro de nuevos usuarios	El sistema debe permitir el registro de nuevos usuarios, los datos que se deben solicitar son país, nombre, apellido, email, nombre de usuario, contraseña y verificación de contraseña. Todos los campos deben ser de carga obligatoria.	Baja
R-05	Recibir notificaciones de ML	El sistema debe recibir las notificaciones enviadas por la API de ML ante algún cambio del estado del reclamo o un nuevo reclamo y persistirá este cambio o el reclamo en la base de datos.	Alta
R-06	Validar reclamo	La herramienta debe verificar que el reclamo que llega pertenece al usuario.	Baja
R-07	Listado de todos los reclamos	Se debe desplegar un listado con todos los reclamos de todas las tiendas. Primero los abiertos y luego los cerrados, ordenados por mayor precio y antigüedad descendente y ascendente respectivamente.	Alta
R-08	Identificación visual	Los reclamos que estén abiertos y esperando por la respuesta del vendedor deben tener un borde de color rojo	Alta
R-09	Filtrar por tiendas	El <i>software</i> debe permitir filtrar por tiendas en caso de que el vendedor tenga multitiendas.	Media
R-10	Búsqueda por filtro	El sistema debe permitir filtrar los reclamos por los siguientes criterios precargados: estado (abierto o cerrado), por estado de la respuesta (esperando tu respuesta,	Alta

		esperando al comprador o esperando a mercado libre) y por tipo (PDD o PNR).	
R-11	Ver detalle del reclamo	La herramienta debe permitir ver el detalle de un reclamo incluyendo los mensajes enviados por todas las partes especificando el remitente, detalle de la venta: nombre, URL, precio y foto, el nombre del comprador y un indicador de si afecta la reputación.	Alta
R-12	Comunicación entre las partes	El sistema debe tener la opción de responder los mensajes de un reclamo abierto con y sin archivo adjunto.	Media
R-13	Ver resolución propuesta por el comprador	El sistema debe mostrar en el detalle del reclamo la resolución propuesta por el comprador.	Alta
R-14	Resolución de reclamos	La herramienta debe permitir al usuario visualizar en el detalle de un reclamo abierto todas las acciones que ML le permite realizar para gestionar el reclamo.	Alta
R-15	Resolución de reembolso	En los casos que la resolución a tomar o a proponer sea la opción de reembolso no se podrá gestionar desde la herramienta, se deberá indicar un link a la URL para gestionar esto.	Baja
R-16	Comprobante de envío	El sistema debe permitir subir la evidencia de envío de un paquete ya enviado teniendo en cuenta las posibilidades de envío existentes. O subir una probable fecha de envío del paquete.	Media
R-17	Visualizar mediación	El detalle del reclamo contará con la información de si está en mediación o no.	Alta
R-18	Iniciar mediación	En el detalle del reclamo se debe brindar al vendedor la opción de abrir una mediación con ML.	Media
R-19	Obtención de reportes	Se debe permitir obtener reportes de operaciones, foto del tablero día a día de todas las variables de datos que esta muestra, se podrá seleccionar un rango de fechas para filtrar.	Media
R-20	Exportar reportes	El sistema debe permitir exportar los reportes del requerimiento anterior (R-19) en formato csv.	Baja
R-21	<i>Dashboard</i>	Se contará con una ventana que oficiará de Tablero ( <i>dashboard</i> ) con la indicación de los valores de variables en tiempo real. En el <i>dashboard</i> se indicará las siguientes variables: el porcentaje de reportes que afectaron la reputación del vendedor en los	Alta

		últimos 3 meses, la cantidad de reclamos abiertos y la cantidad de reclamos pendientes a respuesta del vendedor.	
R-22	Dashboard ampliado	Se debe considerar la posibilidad de también mostrar los siguientes valores en el <i>dashboard</i> : la cantidad de reclamos por tienda, la cantidad de reclamos cerrados ( <i>nice to have</i> )	Media
R-23	Cerrar reclamos de alto monto	El gerente debe tener el poder de cerrar los reclamos de alto valor de dinero (rol con acceso a más funcionalidades).	Baja

### 2.7.2. Requerimientos no funcionales

ID	NOMBRE	DESCRIPCIÓN
R-24	Tipo de solución	Aplicación Web con MVC Nativo
R-25	Navegadores	El sistema correrá en los navegadores Chrome y Firefox.
R-26	Tecnología de <i>backend</i>	Se debe desarrollar una WebAPI Nativa para Azure con NET Core.
R-27	Persistencia de datos	El sistema debe persistir todos los datos de los reclamos y usuarios, la tecnología a utilizar debe ser SQL Server sobre Azure
R-28	Tecnología a usar	C#.NET
R-29	Usabilidad	Debe ser una aplicación <i>responsive</i>
R-30	Integración con ML	El sistema se debe integrar con la API de ML
R-31	Autenticación con ML	Integrar el <i>login</i> vía OAUTH2 con la plataforma de ML.
R-32	Futura integración con SS	El sistema debe tener una estructura de datos compatible con la actual de Sherlock Solutions
R-33	Lenguaje	El lenguaje debe de ser español latinoamericano.
R-34	Localización	Manejar localización ( <i>nice to have</i> )

## 2.8. Alcance y limitaciones

Para definir el alcance y limitaciones de este proyecto se consideró que el mismo es un proyecto académico de tiempo limitado, a partir de esto y luego de relevar los requerimientos junto con el cliente se realizó la estimación detallada del tiempo de trabajo de cada uno de los mismos, dicha estimación se detalla en la siguiente tabla.

ID	NOMBRE	TIEMPO EN HORAS
R-01	<i>Login</i>	10
R-02	Autenticación del usuario en ML	45
R-03	Manejo de Roles	45
R-04	Registro de nuevos usuarios	20
R-05	Recibir notificaciones de ML	50
R-06	Validar reclamo	10
R-07	Listado de todos los reclamos	30
R-08	Identificación visual	8
R-09	Filtrar por tiendas	20
R-10	Búsqueda por filtro	30
R-11	Ver detalle del reclamo	50
R-12	Comunicación entre las partes	25
R-13	Ver resolución propuesta por el comprador	10
R-14	Resolución de reclamos	40
R-15	Resolución de reembolso	12
R-16	Comprobante de envío	30
R-17	Visualizar mediación	10
R-18	Iniciar mediación	10
R-19	Obtención de reportes	40
R-20	Exportar reportes	25
R-21	<i>Dashboard</i>	40
R-22	<i>Dashboard</i> ampliado	15
R-23	Cerrar reclamos de alto monto	15
Total		600

A partir de esta estimación, se observó que el tamaño del proyecto excede los recursos de tiempo con los que se cuenta para el desarrollo ya que el total de horas estimadas para la totalidad de los requerimientos es de 600 horas, sin embargo, disponemos de 505 horas de desarrollo. Es así que en conjunto con el cliente se determinó el alcance del proyecto quedando los siguientes requerimientos fuera del alcance del mismo:

- R-03 - Manejo de Roles.
- R-20 - Exportar reportes
- R-23 - Cerrar reclamos de alto monto.
- R-34 - Localización

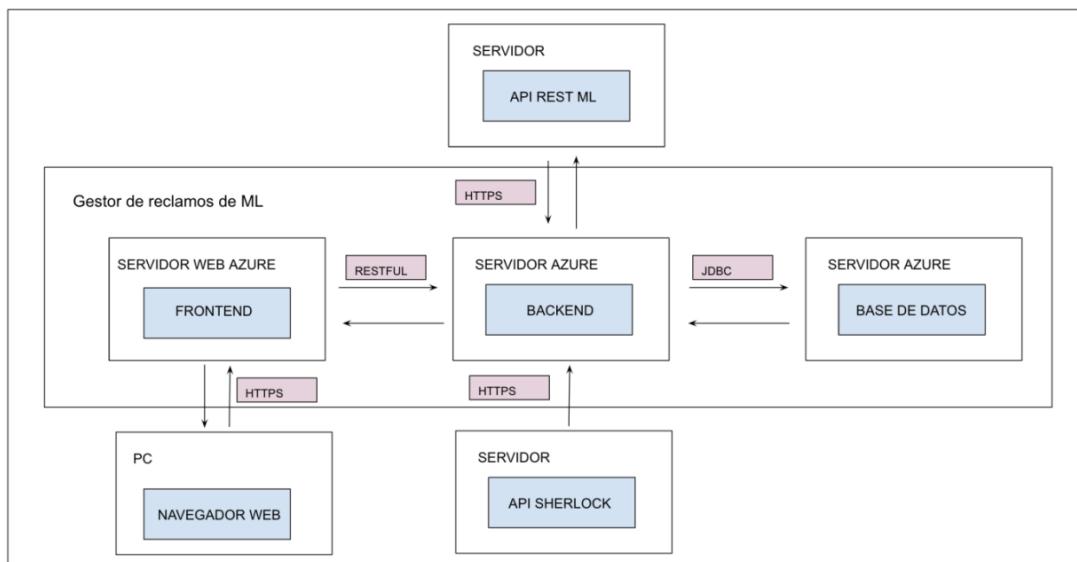
## 2.9.Arquitectura - Particularidades

### 2.9.1. Diagrama de arquitectura

Para este proyecto se acordó con el cliente implementar una arquitectura que se va a dividir en *Front-End* y *Back-End*. Permitiendo de esta forma seguir el modelo de distribución SaaS (*software as a service*), teniendo como *Back-End* una API REST que dejará comunicarnos con otras plataformas como la API REST de ML y la de Sherlock Solutions a través del protocolo *HTTPS*. Esto hace a la solución sumamente escalable y permite que fácilmente se pueda integrar nuestra API a los servicios que ya brinda Sherlock Assistant.

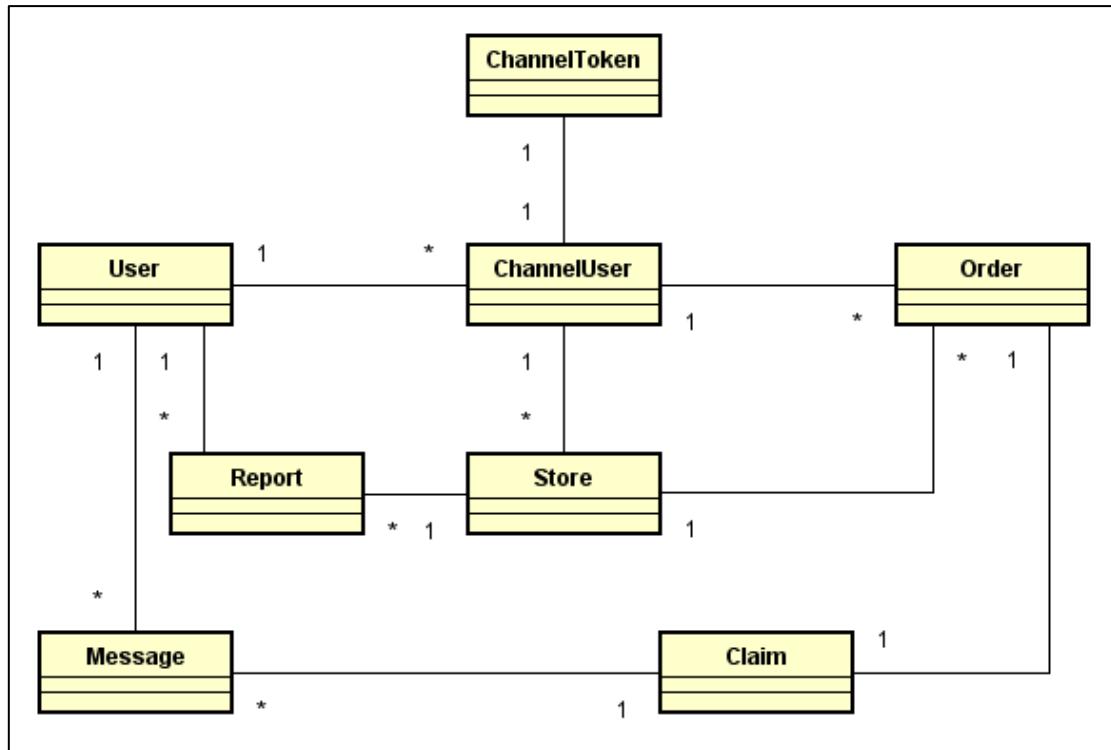
La base de datos se conectará con el *Back-End* pero será independiente a la misma; ésta será publicada en un servidor de Azure desde el comienzo del desarrollo para evitar inconsistencias en las estructuras de los datos, entre los dos desarrolladores mientras se está desarrollando de manera local.

De la misma forma, el resto del proyecto será publicado en su totalidad (*Front-End* y *Back-End*) en Azure para el ambiente de producción. La publicación del *Back-End* se hará a medida que se avance con el trabajo de desarrollo, ya que es un requisito tener nuestra API publicada para mantener la conexión con ML y que se pueda recibir notificaciones desde la API de ML.



## 2.9.2. Diagrama conceptual

Luego de realizar un estudio del sistema existente de Sherlock Assistant y de las necesidades que se abarcarán en este proyecto se realizó el diagrama conceptual que se detalla a continuación. El mismo puede llegar a sufrir algún cambio a medida que avance el proceso de desarrollo documentándose oportunamente para la entrega final.



## 2.10. Identificación de riesgos

En este capítulo se encuentra el análisis de los riesgos que pueden surgir durante el proyecto, a su vez se realizó el análisis cuantitativo de la ocurrencia del mismo y se clasificó en una escala de alta, moderada y baja. Además, se estimó el impacto que puede llegar a tener en el proyecto categorizándose en catastrófico, grave y tolerable. También se especifica la mitigación y contingencia ante la ocurrencia de cada uno de los riesgos.

NOMBRE DEL RIESGO	OCURRENCIA	IMPACTO
Disponibilidad horaria del equipo de desarrollo	Media	Tolerable
Estimación de esfuerzos en comparación al tamaño del producto	Alta	Tolerable
Enfermedad de alguno de los desarrolladores o indisponibilidad por fuerza mayor	Media	Grave - Catastrófica
Cambio en los requerimientos o existencia de nuevos	Baja	Tolerable
Cambio de las reglas de negocio de ML para la gestión de reclamos	Baja	Grave
Curva de aprendizaje	Media	Tolerable

- **Disponibilidad horaria del equipo de desarrollo:**

Descripción: Los dos integrantes del equipo trabajan a tiempo completo sumado a actividades curriculares de inglés, lo cual implica que las horas dedicadas al proyecto durante los días laborales sean reducidas.

Mitigación: A la hora de calcular las horas dedicadas al proyecto se subestimó con el fin de asegurar que esas horas si se van a utilizar.

Contingencia: Se prevé que no van a ser menores las horas dedicadas al proyecto en el caso que no se cuente con las mismas se replanificarán los *sprints*.

- **Estimación de esfuerzos en comparación al tamaño del producto:**

Descripción: Los integrantes del equipo poseen menos de un año de experiencia en el desarrollo de *software* sumado a la inexperiencia en la estimación de tiempos y el poco conocimiento en la complejidad del uso de la herramienta de ML, puede ocurrir que las estimaciones de tiempos no sean las adecuadas para cada uno de los requerimientos.

Mitigación: A la hora de estimar el esfuerzo de los requerimientos más complejos se sobreestima el mismo, contemplando el tiempo de investigación y capacitación que requerirán.

Contingencia: En el caso de no haber estimado adecuadamente los tiempos se replanificarán los *sprints* junto con el cliente, a su vez en la planificación del

cronograma de trabajo se dejó una semana libre al final con el fin de disponer de la misma ante la ocurrencia de este riesgo.

- **Enfermedad de alguno de los desarrolladores o indisponibilidad por fuerza mayor:**

Descripción: Dada la circunstancia de pandemia del COVID-19 u otra patología, puede existir el riesgo de enfermedad de alguno de los desarrolladores o allegados de los mismos viéndose afectada la disponibilidad horaria.

Mitigación: Tomar las medidas de precaución propuestas para tratar de evitar el contagio.

Contingencia: En la planificación del cronograma del proyecto se dejó una semana libre al final para contar con este recurso en caso de materializarse el riesgo.

- **Cambio en los requerimientos o existencia de nuevos:**

Descripción: El cliente cambia o propone nuevos requerimientos.

Mitigación: Se llevaron a cabo varias reuniones con los clientes para realizar un análisis exhaustivo de las necesidades y requerimientos para que no queden necesidades sin contemplar.

Contingencia: Se evaluará cuento impacta en el proyecto para definir si es posible integrarlo.

- **Cambio de las reglas de negocio de ML para la gestión de reclamos:**

Descripción: Las reglas de negocio de ML van cambiando a lo largo del tiempo pudiendo ocurrir durante el transcurso de desarrollo del proyecto.

Mitigación: Pensar una solución escalable permitiendo que se pueda modificar la misma de manera fácil, teniendo una capa de lógica desacoplada.

Contingencia: Se realizarán los cambios necesarios en la capa de lógica si se dispone del tiempo suficiente.

- **Curva de aprendizaje:**

Descripción: Actualmente los desarrolladores cuentan con conocimiento curricular en las tecnologías propuestas por el cliente, sin embargo, no cuentan con experiencia en el Framework .NET Core ni en la integración con la API de ML.

Mitigación: Se consideró destinar tiempo al inicio del *sprint* 3 para la configuración de ambientes de desarrollo y capacitación.

Contingencia: Se replanificarán los *sprints* en caso de que no se llegue a obtener todos los conocimientos necesarios.

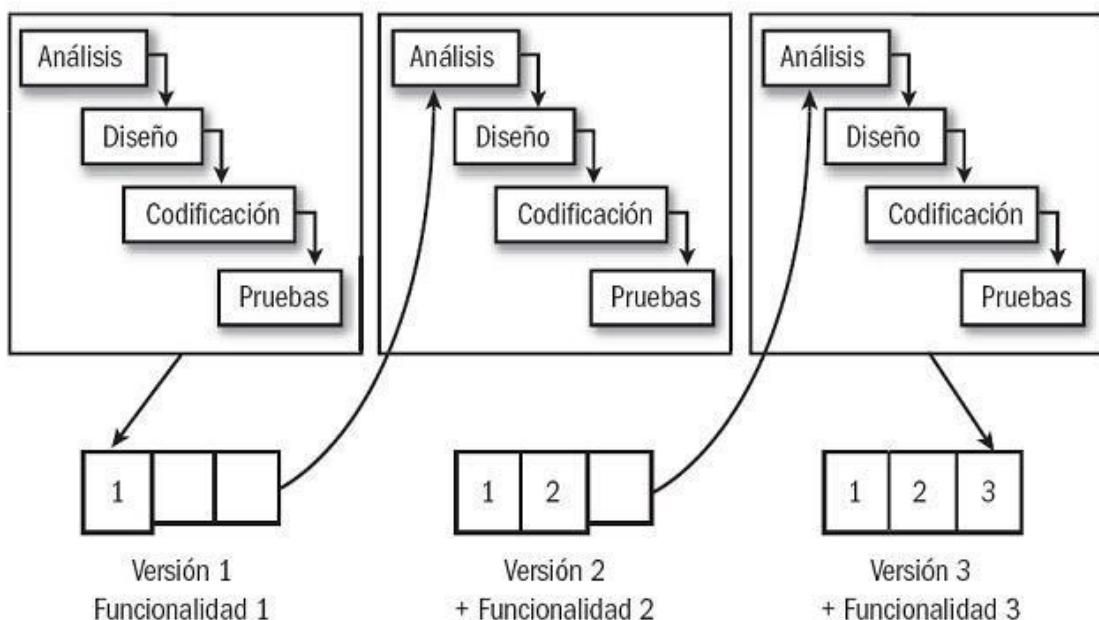
## 2.11. Plan de proyecto

### 2.11.1. Metodología

La metodología seleccionada para la ejecución del proyecto va a ser el modelo iterativo e incremental. A partir del análisis de la complejidad del proyecto y los *stakeholders* optamos por utilizar un método ágil que divide el desarrollo en pequeños subproductos entregables que va incrementando a partir de los resultados completos de las iteraciones anteriores y creando así un versionado del producto hasta obtener el producto final.

Los requerimientos no son estáticos pero intentaremos definirlos lo más exhaustivamente posible para que no sufran cambios o estos sean solo ajustes mínimos a lo largo del proyecto, por lo tanto el modelo iterativo e incremental es la metodología que mejor se adapta a esto, ya que combina el modelo de cascada donde se definen los requisitos al comienzo y se mantienen a lo largo del proyecto y a su vez la metodología de prototipación donde se va agregando funcionalidad de a caso de uso y entregando al cliente los avances. Esta metodología además brinda la ventaja de que el cliente puede obtener resultados importantes y usables ya desde las primeras iteraciones, se puede ir gestionando los cambios que van apareciendo en el proyecto y realizando los ajustes pertinentes, además tras la finalización de cada iteración el cliente puede ir realizando un *feedback* tras ver los resultados. Esta metodología permite que en cada iteración se desarrolle un caso de uso lo que implica una simplificación y organización del trabajo dividiendo requisitos complejos en problemas más pequeños.

También permite corregir los errores en etapas iniciales ya que en cada iteración se realizan pruebas de *testing* y calidad de cada una de las funcionalidades agregadas.



## **2.11.2. Descripción y selección de herramientas**

Se planifica trabajar con el sistema operativo Windows. A partir del requisito planteado por el cliente de desarrollar en el lenguaje C#.NET Core y por ser una herramienta conocida por los desarrolladores, el entorno de desarrollo (IDE) seleccionado es Visual Studio.

El motor de la base de datos elegido es SQL Server y la herramienta para el manejo de base de datos es SQL Management Studio debido a que es un software libre y ya conocido por los desarrolladores.

Para la gestión de los *sprints* y *backlog* se va a utilizar Trello, se optó por esta herramienta ya que se va a utilizar Kanban como metodología para el proceso de desarrollo.

La herramienta elegida como medio de comunicación entre el equipo es Teams.

## **2.11.3. Plan de calidad**

La gestión de SQA se realizará una vez que se completa la fase de desarrollo dentro de cada iteración; con el objetivo de encontrar errores en el *software* y corregirlos para medir la calidad del producto desde el punto de vista funcional. Las pruebas las realizará el desarrollador y posteriormente el cliente. Las formas de hacer estas revisiones por parte del desarrollador serán casos de prueba de cada una de las funcionalidades/requerimientos acordados con el cliente y *testing* unitarios automatizados. El cliente por otra parte luego de cada iteración realizará pruebas funcionales de aceptación, para validar que cumpla los requisitos especificados.

## **2.11.4. Plan de configuración de software**

Para el manejo de configuración del código (SCM) se va a utilizar GitHub por ser una herramienta conocida por los desarrolladores y compatible con las tecnologías a usar, a su vez se consideró el uso de Team Foundation Service por estar integrada en Visual Studio y ser una herramienta de .NET, pero implicaba capacitación en la misma, por lo que se descartó ya que GitHub proporciona las mismas soluciones. Para el versionado de la documentación se está usando Teams.

## **2.11.5. Plan de capacitación**

### **2.11.5.1. Capacitación del equipo de proyecto**

Dado que las herramientas y lenguajes son conocidos para los desarrolladores, no es necesario en principio una capacitación previa para comenzar la etapa de desarrollo. Pero sí al momento de preparar el ambiente de desarrollo se hará una inducción al Framework .NET Core e instalación, el cual según lo que se ha investigado no es muy diferente a .NET y ambas adquirimos conocimientos de este lenguaje durante las materias Programación II y Programación III en la Universidad ORT Uruguay.

Se planea seguir capacitándonos a medida del desarrollo, particularmente se hará un estudio y capacitación para la conexión e integración de la API REST de ML a través de la documentación que la plataforma brinda.

#### **2.11.5.2. Capacitación del cliente**

Se realizará una transferencia tecnológica a los clientes a partir de la documentación de la API y un manual de usuario. La API se documentará con la herramienta Swagger creando una *Apiary*, en la misma se detallarán todos los *end points*, los campos que espera recibir y las posibles respuestas de cada solicitud. La documentación de usabilidad de la aplicación web se realizará a través de un manual de usuario donde se detallarán todos los flujos y funcionalidades del sistema.

### **2.11.6. Sprint de planificación**

#### **2.11.6.1. Sprint 1 (13/04/2020 – 26/04/2020)**

El primer *sprint* se trató de conocer al cliente, las particularidades del negocio, establecer los objetivos por medio de las siguientes tareas:

- 13/04 - Primera reunión de *kick off* con Ricardo y Hugo con el objetivo de conocer al cliente y escuchar su problema.
- 17/04 - Reunión con Agustina encargada del departamento comercial la finalidad de la misma es conocer la herramienta con la que ya cuentan de Sherlock Assistant. También se especificó el proceso de venta y post venta en ML.
- 22/04 - Reunión con Agustina (se siguieron los distintos flujos de la gestión de reclamos en ML)  
23/04 - Se identificó que no estaba bien definida la problemática que tenía el cliente, por lo que se envió un mail para definirla y conocer las necesidades que tenían, contestaron con los objetivos que esperaban del proyecto (ver Anexo 1 - Definición objetivos con el cliente).

También se comenzó a escribir el anteproyecto abarcando los siguientes capítulos introducción, descripción del cliente y definición de las herramientas a utilizar.

#### **2.11.6.2. Sprint 2 (27/04/2020 – 10/05/2020)**

En el segundo *sprint* se continuó escribiendo el anteproyecto. Aplicando la ingeniería de requerimientos para poder establecer de forma clara con el cliente el alcance del proyecto. Se realizaron las siguientes tareas:

- Se escribió el problema y gestión actual del proceso de reclamos.
- Se realizó el análisis y documentación de requerimientos.
- 27/04 - Reunión con Ricardo y Hugo donde se logró establecer claramente el problema y definición de requerimientos.
- Se realizó y documento el análisis de riesgos.
- Se realizó y documento el análisis de esfuerzos.
- Se realizó y documento la planificación del proyecto.

## **2.11.7. Cronograma de trabajo**

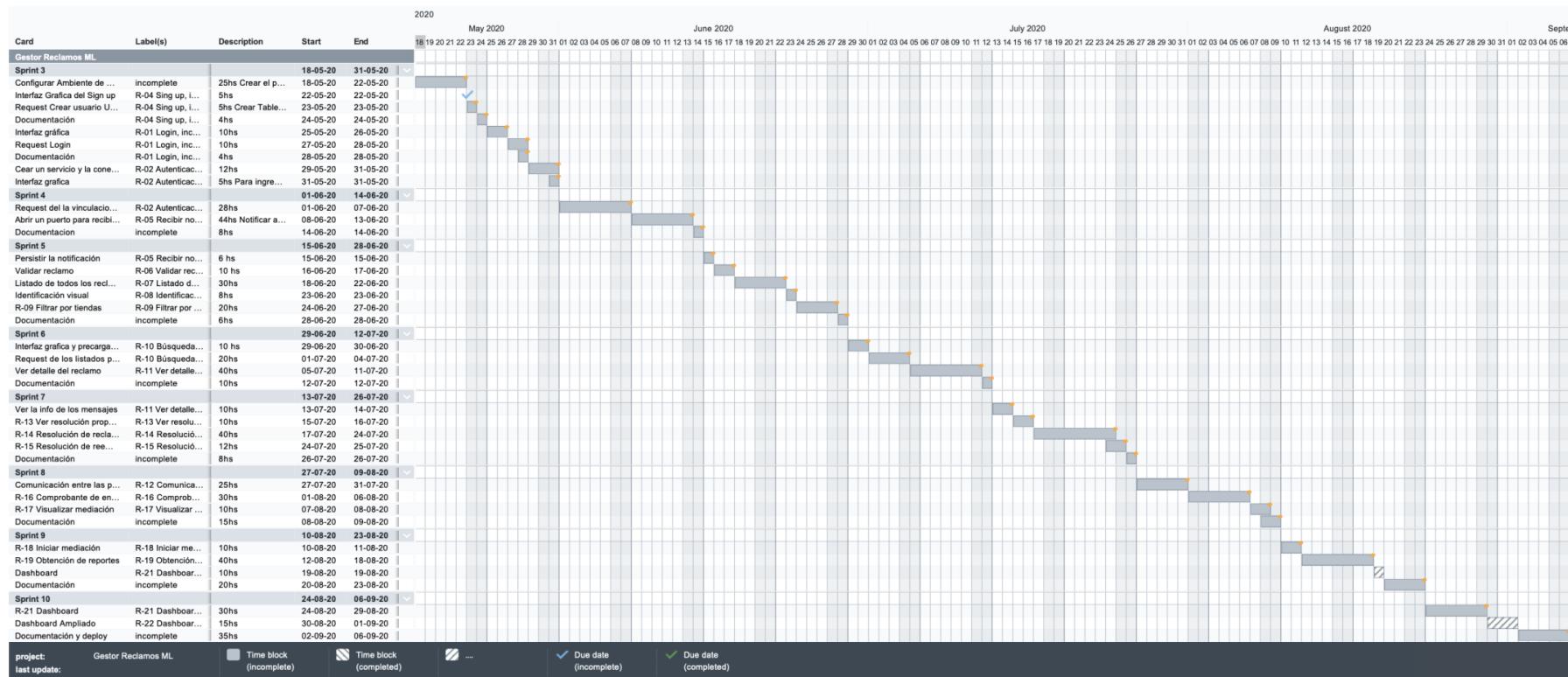
A continuación, se presenta la estimación del tiempo y el cronograma de trabajo. El mismo se va a dividir en 10 *sprints* de un tiempo de 2 semanas cada uno. A raíz de que ambos integrantes del proyecto trabajan a tiempo completo se decidió trabajar un total de 20 horas semanales por persona, haciendo una sumatoria de 640 horas para los *sprints* del 3 al 10. El total de horas se distribuirá de la siguiente manera 505 horas destinadas al desarrollo de *software*, 25hs para capacitación y configuración de ambiente de desarrollo y 110 horas para la documentación y *deploy* del proyecto. En la siguiente tabla se detallan todas las tareas con el tiempo estimado y el *sprint* en el que se realizará la misma.

Card	Label(s)	Description	Start	End
<b>Gestor Reclamos ML</b>				
<b>Sprint 3</b>			<b>18-05-20</b>	<b>31-05-20</b>
Configurar Ambiente de ...	incomplete	25hs Crear el p...	18-05-20	22-05-20
Interfaz Grafica del Sign up	R-04 Sing up, i...	5hs	22-05-20	22-05-20
Request Crear usuario U...	R-04 Sing up, i...	5hs Crear Table...	23-05-20	23-05-20
Documentación	R-04 Sing up, i...	4hs	24-05-20	24-05-20
Interfaz gráfica	R-01 Login, inc...	10hs	25-05-20	26-05-20
Request Login	R-01 Login, inc...	10hs	27-05-20	28-05-20
Documentación	R-01 Login, inc...	4hs	28-05-20	28-05-20
Cesar un servicio y la cone...	R-02 Autenticac...	12hs	29-05-20	31-05-20
Interfaz grafica	R-02 Autenticac...	5hs Para ingre...	31-05-20	31-05-20
<b>Sprint 4</b>			<b>01-06-20</b>	<b>14-06-20</b>
Request del la vinculacio...	R-02 Autenticac...	28hs	01-06-20	07-06-20
Abrir un puerto para recibi...	R-05 Recibir no...	44hs Notificar a...	08-06-20	13-06-20
Documentacion	incomplete	8hs	14-06-20	14-06-20
<b>Sprint 5</b>			<b>15-06-20</b>	<b>28-06-20</b>
Persistir la notificación	R-05 Recibir no...	6 hs	15-06-20	15-06-20
Validar reclamo	R-06 Validar rec...	10 hs	16-06-20	17-06-20
Listado de todos los recl...	R-07 Listado d...	30hs	18-06-20	22-06-20
Identificación visual	R-08 Identificac...	8hs	23-06-20	23-06-20
R-09 Filtrar por tiendas	R-09 Filtrar por ...	20hs	24-06-20	27-06-20
Documentación	incomplete	6hs	28-06-20	28-06-20
<b>Sprint 6</b>			<b>29-06-20</b>	<b>12-07-20</b>
Interfaz grafica y precarga...	R-10 Búsqueda...	10 hs	29-06-20	30-06-20
Request de los listados p...	R-10 Búsqueda...	20hs	01-07-20	04-07-20
Ver detalle del reclamo	R-11 Ver detalle...	40hs	05-07-20	11-07-20
Documentación	incomplete	10hs	12-07-20	12-07-20
<b>Sprint 7</b>			<b>13-07-20</b>	<b>26-07-20</b>
Ver la info de los mensajes	R-11 Ver detalle...	10hs	13-07-20	14-07-20
R-13 Ver resolución prop...	R-13 Ver resolu...	10hs	15-07-20	16-07-20
R-14 Resolución de recla...	R-14 Resolució...	40hs	17-07-20	24-07-20
R-15 Resolución de ree...	R-15 Resolució...	12hs	24-07-20	25-07-20
Documentación	incomplete	8hs	26-07-20	26-07-20
<b>Sprint 8</b>			<b>27-07-20</b>	<b>09-08-20</b>
Comunicación entre las p...	R-12 Comunica...	25hs	27-07-20	31-07-20
R-16 Comprobante de en...	R-16 Comprob...	30hs	01-08-20	06-08-20
R-17 Visualizar mediación	R-17 Visualizar ...	10hs	07-08-20	08-08-20
Documentación	incomplete	15hs	08-08-20	09-08-20
<b>Sprint 9</b>			<b>10-08-20</b>	<b>23-08-20</b>
R-18 Iniciar mediación	R-18 Iniciar me...	10hs	10-08-20	11-08-20
R-19 Obtención de reportes	R-19 Obtención...	40hs	12-08-20	18-08-20
Dashboard	R-21 Dashboar...	10hs	19-08-20	19-08-20
Documentación	incomplete	20hs	20-08-20	23-08-20
<b>Sprint 10</b>			<b>24-08-20</b>	<b>06-09-20</b>
R-21 Dashboard	R-21 Dashboar...	30hs	24-08-20	29-08-20
Dashboard Ampliado	R-22 Dashboar...	15hs	30-08-20	01-09-20
Documentación y deploy	incomplete	35hs	02-09-20	06-09-20

Además de este calendario también están planificadas en conjunto con el cliente las fechas de las entregas parciales de los avances del proyecto, la primera se hará al final del *sprint 4* y luego las siguientes al final de cada uno de los *sprints*.

## 2.11.8. Diagrama de Gantt (actividades planificadas)

Contamos con dos recursos para poder ver de forma más nítida el diagrama de Gantt (ver Anexo 2 - Recursos para ver el diagrama de Gantt)



## **2.12. Compromiso de trabajo**

Los integrantes del proyecto nos comprometemos a dedicarle un total de 20 horas semanales por persona al proyecto con el fin de cumplir con las actividades planificadas en cada uno de los *sprints*. Las horas se distribuirán aproximadamente como indica la tabla a continuación:

DIA	HORARIO	CANTIDAD DE HORAS
Lunes	19:00 - 21:00	2 horas
Martes	19:00 - 21:00	2 horas
Viernes	19:00 - 21:00	2 horas
Sábado	14:00 - 21:00	7 horas
Domingo	14:00 - 21:00	7 horas

A si mismo nos comprometemos a cumplir con las fechas estipuladas por la Universidad ORT Uruguay para cada una de las entregas. También nos comprometemos a respetar las fechas acordadas con el cliente para las entregas parciales del producto, estas fechas se especifican en el último párrafo del cronograma de trabajo que se encuentra en el capítulo 2.11.7.

## 3. Proyecto

### 3.1. Introducción

En este capítulo se detalla el proceso de desarrollo y ejecución del proyecto. Se documenta el abordaje de los requerimientos en cada uno de los *sprints*, especificando el esfuerzo estimado en el anteproyecto vs el esfuerzo real, los desvíos que pudo haber surgido y los nuevos requerimientos incluidos en el producto final. A su vez, se incluye los documentos y diagramas finales de la solución. También se especifica las muestras realizadas al cliente.

### 3.2. Pila del producto final

Durante el transcurso del desarrollo del proyecto al utilizar una metodología de trabajo ágil surgieron algunos requerimientos nuevos, los cuales fueron evaluados y estimados en cada *sprint* para definir si era posible incluirlos en esta etapa y en qué *sprint* era conveniente desarrollarlos. Algunos de estos requerimientos nuevos fueron solicitados por el cliente luego de las muestras realizadas mientras que otros fueron propuestos por las desarrolladoras con el objetivo de brindar un producto de mejor calidad y usabilidad. A continuación, se incluye la pila del producto final:

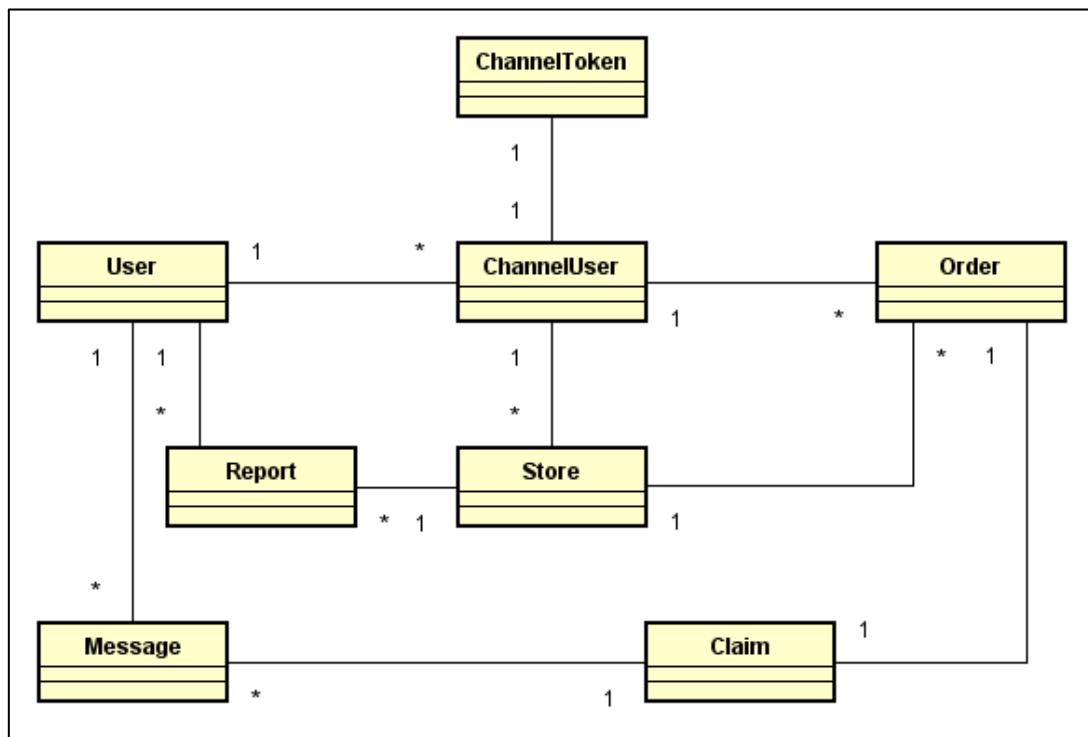
- R-01: *Login*
- R-02: Autenticación del usuario en ML
- R-04: Registro de nuevos usuarios
- R-05: Recibir notificaciones de ML
- R-06: Validar reclamo
- R-07: Listado de todos los reclamos
- R-08: Identificación visual
- R-09: Filtrar por tiendas
- R-10: Búsqueda por filtro
- R-11: Ver detalle del reclamo
- R-12: Comunicación entre las partes
- R-13: Ver resolución propuesta por el comprador
- R-14: Resolución de reclamos
- R-15: Resolución de reembolso
- R-16: Comprobante de envío
- R-17: Visualizar mediación
- R-18: Iniciar mediación
- R-19: Obtención de reportes
- R-21: *Dashboard*
- R-22: *Dashboard* ampliado
- R-35: Incorporación visual en listado de los reclamos de si el reclamo afecta la reputación del vendedor y del identificador de la orden de compra en el detalle del reclamo.
- R-37: *Dashboard* ampliado 2
- R-38: Filtrar reclamos en mediación con ML
- R-39: Persistencia del log de excepciones

- R-40: *Logout*
- R-42: Visualizar detalle de evidencia de envío
- *Deploy*
- Configuración de ambientes de desarrollo

### 3.3. Diseño final de la solución

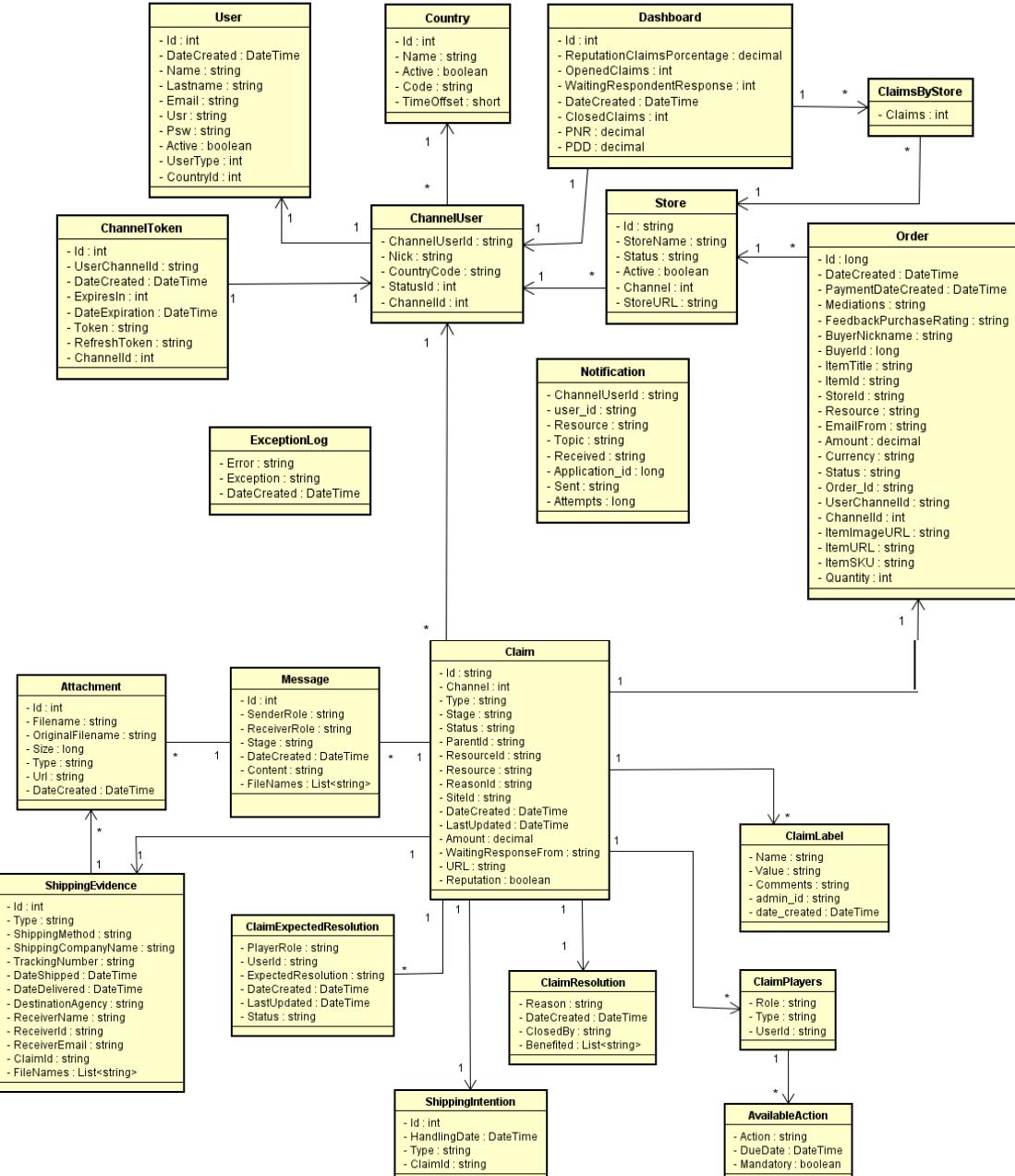
#### 3.3.1. Documentos de análisis

Se utilizó el mismo diagrama conceptual realizado en el anteproyecto y a medida que se fue avanzando en el desarrollo del proyecto se hizo el diagrama de clases presentando en el siguiente capítulo (3.3.2 Documentos de diseño) y el mismo se actualizaba a mediada que iba creciendo el proyecto y se iban incorporando clases.



### **3.3.2. Documentos de diseño**

A continuación, se presenta el diagrama de clases de implementación:

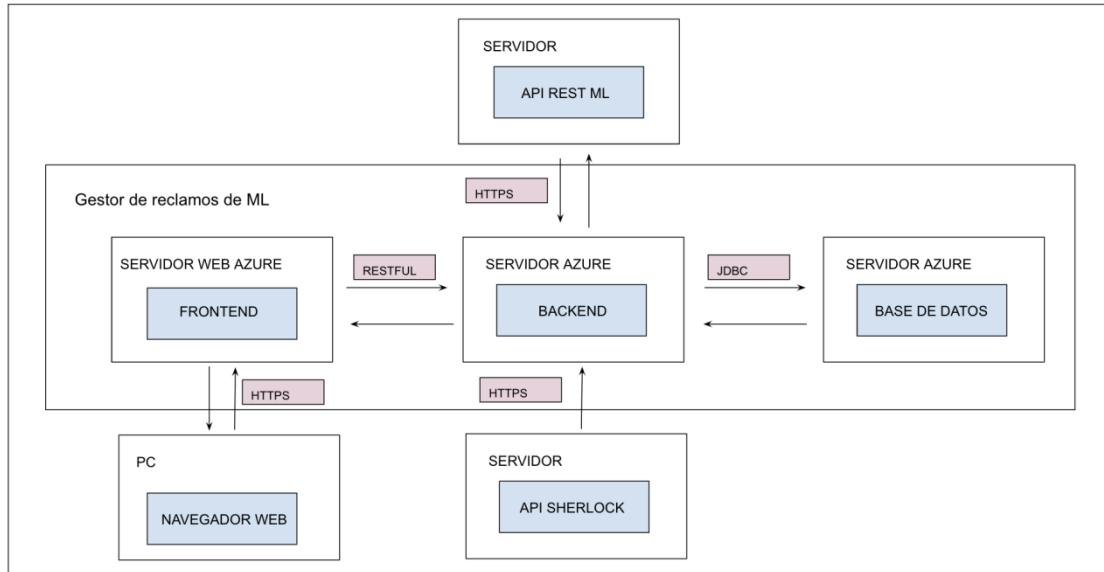


### 3.3.3. Arquitectura final de la solución

En este capítulo se presenta la arquitectura final de la solución especificando el diagrama de arquitectura implementado y el manual de *deploy*.

#### 3.3.3.1. Diagrama de arquitectura implementado

Se mantuvo el mismo diagrama realizado en el anteproyecto ya que el mismo era adecuado y fue el que se utilizó para la solución final.



#### 3.3.3.2. Manual de *deploy*

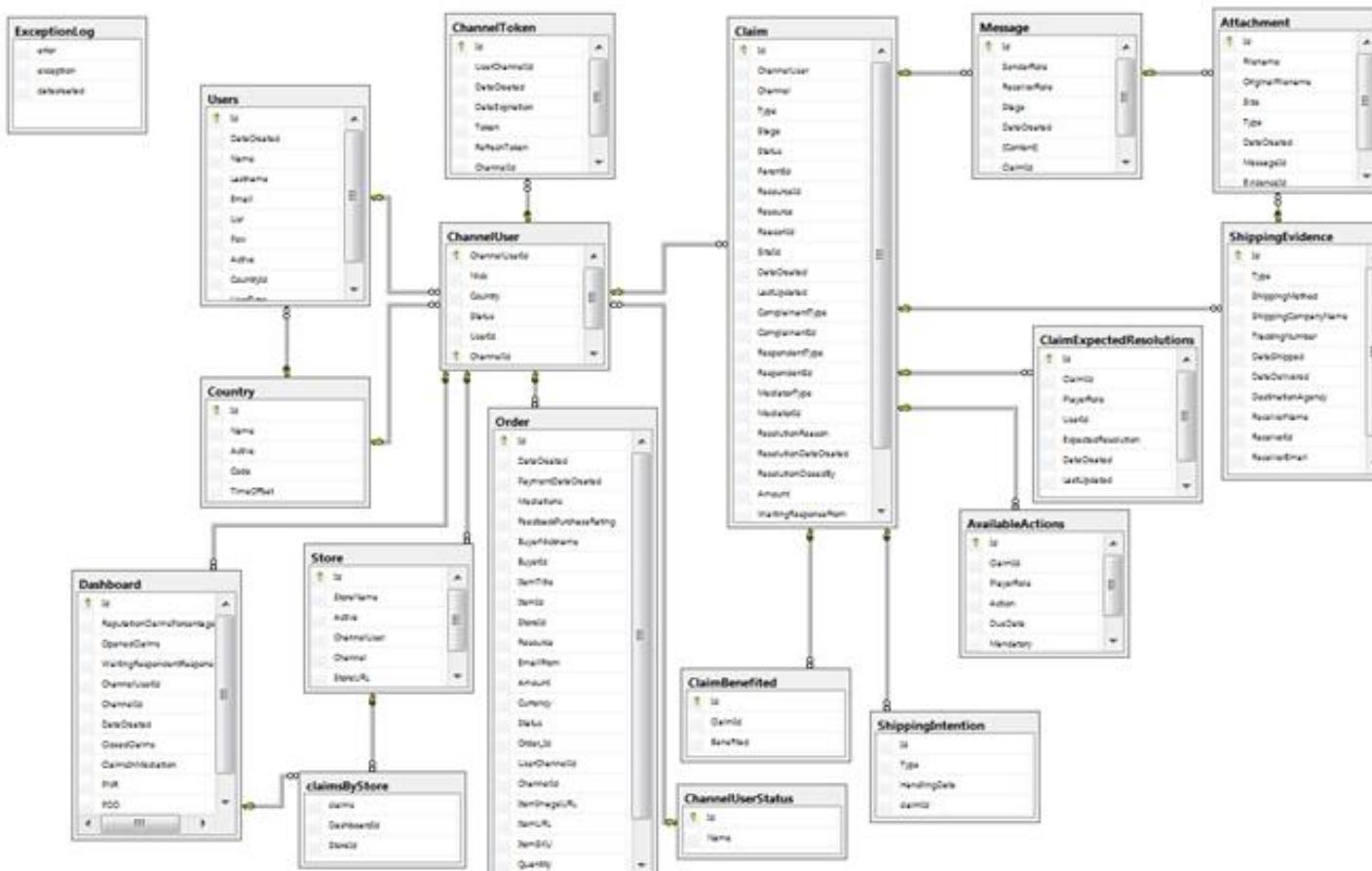
Tanto el *frontend* como el *backend* y la base de datos se encuentran *hosteados* en Azure y cada uno está alojado de manera independiente. Para realizar el *deploy* se utilizó la cuenta creada por la Universidad ORT Uruguay con créditos para utilizar en Azure. Para ver el detalle del *deploy* ver en Anexo 11 - Manual de *deploy*.

##### Recent resources

Name	Type	Last Viewed
GRML API	App Service	4 days ago
GRML (grml/GRML)	SQL database	4 days ago
GRMLWWW	App Service	4 weeks ago

### 3.3.4. Diagrama de tablas

Se trabajó con una base de datos relacional, se utilizó SQL Server como motor de base de datos. A continuación, se presenta el modelo entidad relación implementado en la solución:



## 3.4. Desarrollo del proyecto

Al comenzar el *sprint* 3 vimos que, en el Anteproyecto, en el capítulo 2.11.7 cronograma de trabajo, las horas especificadas para el requerimiento de *login* y registro de nuevos usuarios quedaron invertidas. Sin embargo, en el capítulo 2.6 Alcance y limitaciones en la tabla que se especifica la estimación en tiempo de cada uno de los requerimientos sí están correctamente especificadas las horas. En la documentación de este capítulo y en la etapa de desarrollo del proyecto se tuvo en cuenta las horas especificadas en el capítulo 2.6, siendo 10 horas estimadas para el requerimiento del *login* y 20 horas para el registro de nuevos usuarios.

Durante el desarrollo del proyecto se fue realizando el *testing* de la API, la documentación de estos *tests* se encuentra en el Anexo 3 – Evidencia de *testing* de la API. También se realizaron casos de prueba para validar el correcto funcionamiento del *frontend* (ver en Anexo 4 – Casos de prueba)

### 3.4.1. Sprint 3 (18/05/2020 - 31/05/2020)

#### 3.4.1.1. Planificación del sprint

En el listado a continuación se detallan las tareas planificadas para este *sprint*.

- Configurar ambientes de desarrollo
- *Login* (R-01)
- Registro de nuevos usuarios (R-04)
- Autenticación del usuario en ML (R-02)
- Documentación

#### 3.4.1.2. Tareas Completadas/Realizadas

Las horas de trabajo estimadas para el *sprint* son de 80 horas, finalizado el mismo se totalizan 93,5 horas reales dedicadas. Dentro de estas horas se incluyen diversos tipos de tareas ya sea de desarrollo, configuración, *testing* o documentación. En esta tabla se detalla la distribución horaria entre las diferentes tareas.

OBJETIVO	HORAS		ESTADO
	ESTIMADAS	REALES	
Configurar ambientes de desarrollo	25	25	Completado
<i>Login</i> (R-01)	10	11	Completado
Registro de nuevos usuarios (R-04)	20	20	Completado
<i>Logout</i> (R-40)	-	6	Completado
Documentación de la API	-	3.5	Completado
Deploy	-	6	Completado
Autenticación del usuario en ML (R-02)	17	17	En progreso
Documentación	8	5	Completado

Uno de los principales objetivos de este *sprint* es la configuración de ambientes de desarrollo, dentro de esta gran tarea se incluyó:

- Capacitación
- Instalación de Visual Studio 2019
- Creación de la API en .NET Core.
- Creación de un proyecto web en .NET con MVC.
- Creación de un proyecto XUnitTest.
- Creación de la base de datos.
- Configuración de los proyectos para conectarlos.
- Creación del repositorio en GitHub para la gestión de SCM y conexión con el mismo.
- Configuración de Swashbuckle y Swagger para la documentación de la API.

Cabe destacar que dentro de las horas estimadas y reales del requerimiento *login* y registro de nuevos usuarios se incluyen las horas dedicadas al *testing* de estas tareas siendo 2 y 3 horas respectivamente. Específicamente se dedicaron 9 horas para el desarrollo del *login* y 2 horas para el *testing* del mismo y el registro de nuevos usuarios se desarrolló en 17 horas y 3 horas fueron dedicadas para realizarle *testing*.

El estado del requerimiento de autenticación del usuario en ML es “en progreso” ya que el mismo está planificado por cronograma (ver capítulo 2.0.7 Cronograma de trabajo) para ser continuado en el siguiente *sprint*.

Si bien las horas dedicadas a la documentación del proyecto son menores a las estimadas, se incluyó en este *sprint* la documentación de los servicios de la API desarrollados hasta la fecha, utilizando la herramienta de Swagger.

Otra tarea que se incorporó y no estaba planificada es el *deploy*, logrando ya tener en este *sprint* el *deploy* en Azure de la base de datos, la API y el proyecto web. Si bien estaba planificado crear la base de datos alojada en Azure desde un inicio para no tener inconsistencias, el *deploy* de la API y el proyecto web se planificaba para futuros *sprints*. Cabe destacar que nos encontramos con algunas dificultades dado que una de las desarrolladoras no contaba con créditos en su cuenta de Azure brindada por parte de la universidad esto implicó horas de comunicación y coordinación con diferentes integrantes de la universidad aún no contamos con una resolución. El *deploy* se realizó en la cuenta de la otra integrante, pero esperamos poder solucionar este inconveniente en los próximos días para contar con esa cuenta de respaldo por si surgiera algún problema con los *deploy* existentes.

#### **3.4.1.3. Tareas no completadas**

En este *sprint* se logró cumplir con todas las tareas planificadas a realizarse en el mismo.

#### **3.4.1.4. Entrega del trabajo a los interesados**

Al finalizar este *sprint* estaba acordado en conjunto con el cliente no realizar una entrega parcial del producto, esto se detalla en el capítulo 2.11.7 Cronograma de trabajo. La finalidad de esto es poder mostrar mayores avances al cliente, por lo que se planificó no realizar una entrega al final de este *sprint*, pero sí al finalizar cada uno de los siguientes *sprints*.

### **3.4.1.5. Cierre/Revisión del Sprint**

#### **3.4.1.5.1. Desvíos y gestión de riesgos**

Si bien se había estimado una menor cantidad de horas de trabajo para este *sprint* la mayor disponibilidad horaria de los desarrolladores permitió dedicarle más horas de las previstas. Esto permitió incluir más tareas que si bien no estaban planificadas dentro del *sprint* se pudieron llevar a cabo permitiendo adelantar trabajo que estaba planificado para futuros *sprints*.

#### **3.4.1.5.2. Nuevos requerimientos surgidos**

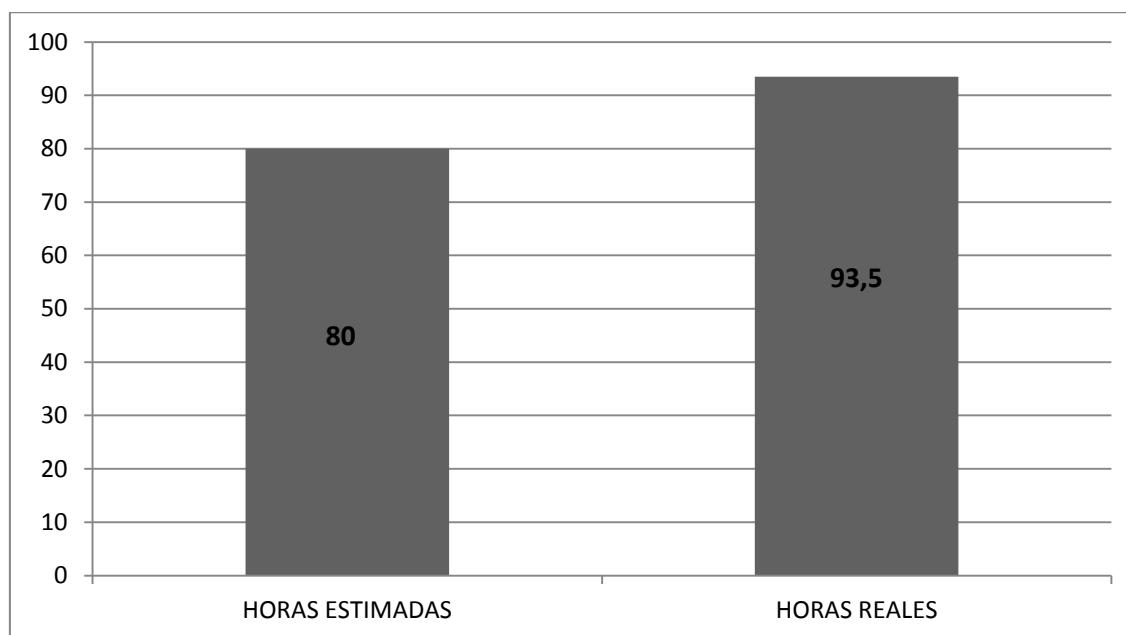
Durante la ejecución del *sprint* se incorporó un nuevo requerimiento el *logout* de los usuarios, el mismo no estaba planificado dentro de los requerimientos detallados en el anteproyecto. Si bien cada nuevo requerimiento que surge puede implicar un problema en el proyecto logramos superarlo con éxito al contar con una mayor disponibilidad de horas en la semana para dedicarle al proyecto.

#### **3.4.1.5.3. Ajuste del plan de trabajo del próximo sprint**

No se realizarán ajustes en el trabajo del próximo *sprint* por lo que se planifica continuar con el cronograma especificado en el anteproyecto.

#### **3.4.1.5.4. Resumen del esfuerzo dedicado en el sprint**

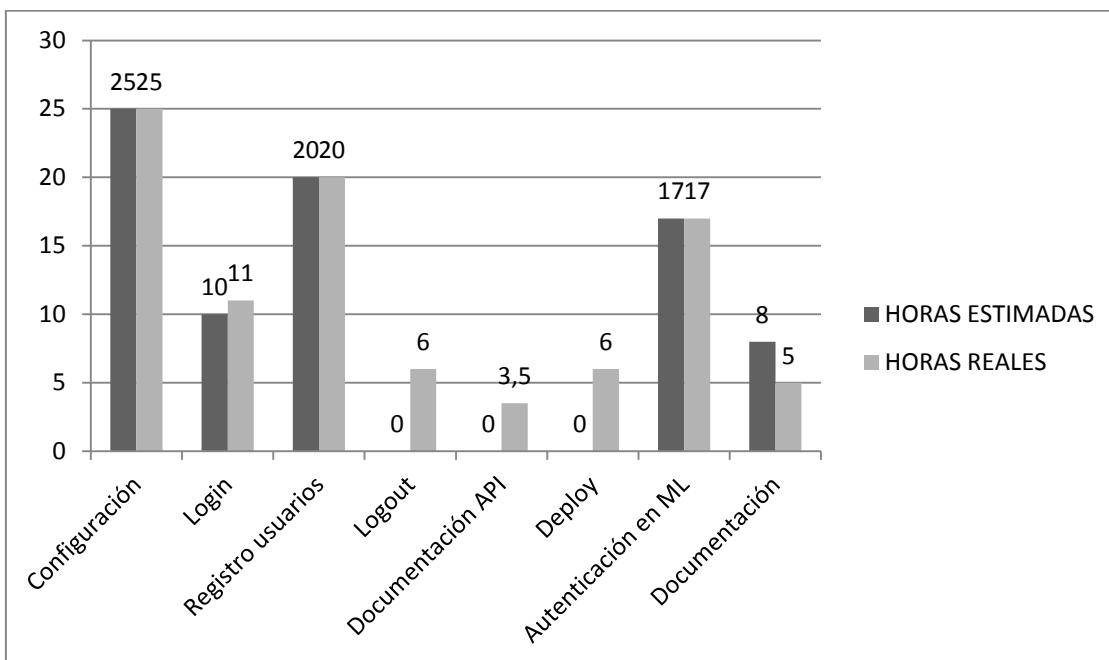
A continuación, se presenta de manera gráfica los datos del esfuerzo en tiempo estimado y el real.



Gráfica 1 - Comparación del total de horas estimadas y reales

Si bien en esta gráfica se observa una diferencia de 13,5 horas entre las horas estimadas y las horas reales cabe destacar que este desvío se debió a la incorporación de nuevas

tareas que no estaban planificadas para este *sprint*. Se decidió incorporar estas tareas ya que las desarrolladoras contamos con más disponibilidad horaria en este *sprint* y se consideró oportuno y de gran aporte para los siguientes *sprints*. En el siguiente gráfico se puede observar en detalle las estimaciones y horas reales de cada una de las tareas implementadas.



Gráfica 2 - Comparación del total de horas estimadas y reales por tarea

### **3.4.2.Sprint 4 (01/06/2020 - 14/06/2020)**

#### **3.4.2.1. Planificación del sprint**

En el listado a continuación se detallan las tareas planificadas para este *sprint*.

- Autenticación del usuario en ML (R-02)
- Recibir notificaciones de ML (R-05)
- Documentación

#### **3.4.2.2. Tareas Completadas/Realizadas**

Las horas de trabajo estimadas para el *sprint* son de 80 horas, finalizado el mismo se totalizan 87 horas reales dedicadas. Dentro de estas horas se incluyen diversos tipos de tareas ya sea de desarrollo, configuración, *testing* y documentación. En esta tabla se detalla la distribución horaria entre las diferentes tareas.

OBJETIVO	HORAS		ESTADO
	ESTIMADAS	REALES	
Autenticación del usuario en ML (R-02)	28	32.5	Completado
Recibir notificaciones de ML (R-05)	44	46.5	En progreso
Documentación	8	8	Completado

Los principales objetivos de este *sprint* son la integración de nuestra API con la API de ML y la creación de las tablas en nuestra base de datos, con la finalidad de obtener toda la información que se necesita de la API de ML y persistir dicha información en nuestra base de datos. Si bien la mayor parte de desarrollo de este *sprint* se centró en el *backend*, también se hizo desarrollo en el proyecto de *frontend*, permitiendo a los usuarios de nuestra aplicación asociarse con su usuario de ML, esto se realizó a través de OAUTH2. Luego que se obtiene el *token* de autorización de ML, con el mismo se accede a la información de dicho usuario en ML lo que permite persistir todos los datos que luego necesitaremos en nuestra aplicación.

Otro de los objetivos de este *sprint* es el recibir notificaciones de ML, esto también implicó trabajo de configuración y desarrollo. Recibir notificaciones de ML nos permite conocer cada vez que se actualiza o genera un nuevo reclamo perteneciente a alguno de nuestros usuarios registrados en ML, una vez que llega la notificación podemos solicitar todos los datos del reclamo con la finalidad de persistir dicha información y poder en un futuro entre otras cosas realizar la gestión del mismo. El estado de este requerimiento es “en progreso” ya que el mismo está planificado por cronograma (ver capítulo 2.11.7 Cronograma de trabajo) para ser continuado en el siguiente *sprint*.

Cabe destacar que tanto para el requerimiento de autenticación del usuario en ML como el recibir notificaciones de ML implicó dedicar tiempo de capacitación ya que al integrarse con una API externa implica primero conocerla al detalle para luego poder conectarse con la misma. Si bien durante los *sprints* 1 y 2 dedicados al desarrollo del anteproyecto se había realizado un estudio de la API de ML fue en el *sprint* actual

donde se implementó la integración inicial por lo que implicó continuar con la capacitación. Estas horas dedicadas a la capacitación ya estaban contempladas en las horas estimadas y fueron sumadas a las horas reales de cada uno de los requerimientos.

Durante este *sprint* luego de varias instancias de comunicación con personal de la universidad, también se logró solucionar el problema de activación en la cuenta de Azure de una de las desarrolladoras. Por lo que actualmente se tiene esta cuenta de respaldo para realizar el *deploy* de la aplicación en caso de que surja algún problema con el actual *deploy*.

Por último, otro de los objetivos de este *sprint*, así como de los siguientes *sprints*, es el documentar los avances que se van realizando en el proyecto, de esta manera se puede detallar de forma más precisa en lo que se fue trabajando sin perder información del proceso.

#### **3.4.2.3. Tareas no completadas**

En este *sprint* se logró cumplir con todas las tareas planificadas a realizarse en el mismo.

#### **3.4.2.4. Entrega del trabajo a los interesados**

Al finalizar este *sprint* estaba acordado en conjunto con el cliente realizar una muestra de los avances en el proyecto para lo cual se agendó una reunión por video llamada donde se realizó la primera presentación del producto. En la misma se presentó la implementación de los requerimientos *login*, registro de nuevos usuarios, *logout*, autenticación del usuario en ML y recibir notificaciones de ML. El cliente se mostró conforme y satisfecho con todos los avances que se mostraron, ya que consideraban que estos dos *sprints* era donde estaba la mayor curva de aprendizaje a nivel tecnológico. Destacaron positivamente el ya haber realizado la integración con la API de ML por los desafíos que esto implica.

#### **3.4.2.5. Cierre/Revisión del Sprint**

##### **3.4.2.5.1. Desvíos y gestión de riesgos**

Al finalizar el *sprint* podemos observar una pequeña desviación en la estimación del esfuerzo en los requerimientos de autenticación del usuario en ML y recibir notificaciones de ML. Se había estimado una menor cantidad de horas de trabajo para cada una de estas tareas y para el total del *sprint*, esta desviación no implicó un problema ya que se contó con una mayor disponibilidad horaria para dedicarle más horas de las previstas. Los motivos de esta desviación se pueden deber a la poca experiencia con que se cuenta en la estimación de horas, sumado a la dificultad que implica estimar el esfuerzo de integrarse con una API externa.

##### **3.4.2.5.2. Nuevos requerimientos surgidos**

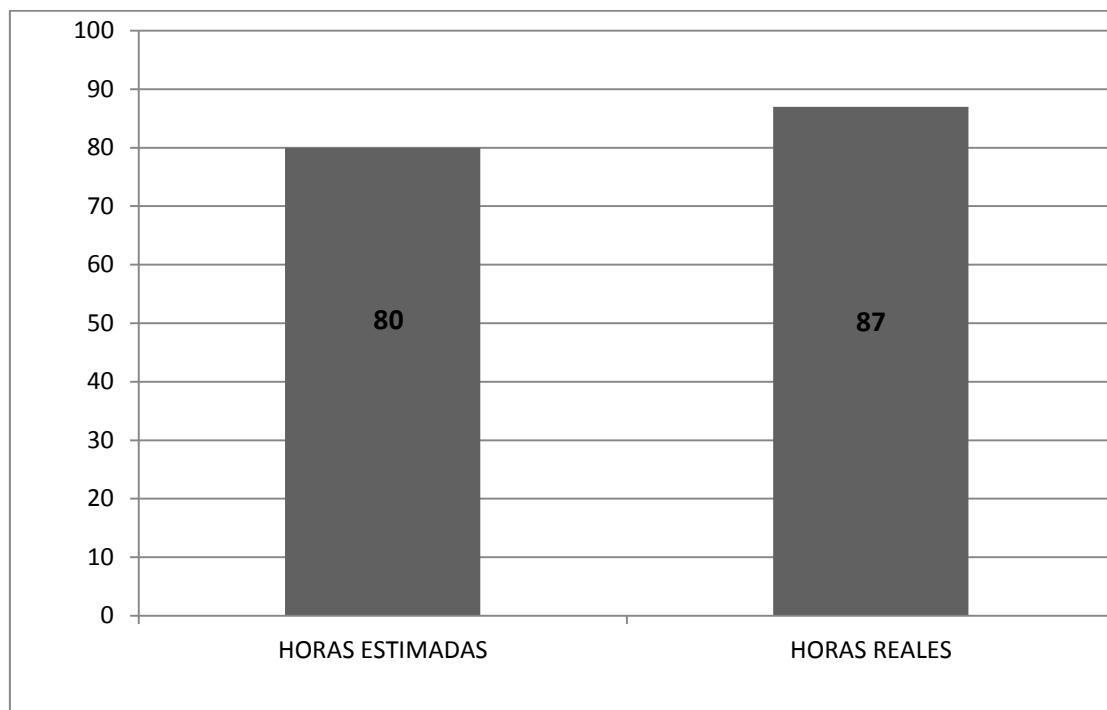
Durante este *sprint* no se incorporaron nuevos requerimientos.

### **3.4.2.5.3. Ajuste del plan de trabajo del próximo sprint**

No se realizarán ajustes en el trabajo del próximo *sprint* por lo que se planifica continuar con el cronograma especificado en el anteproyecto.

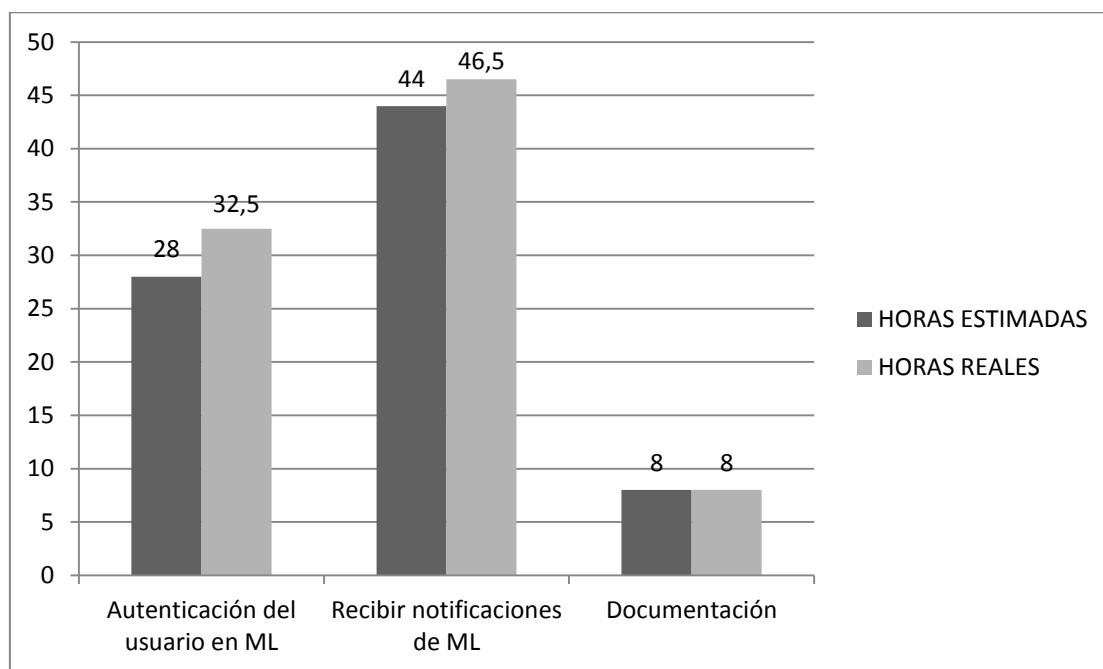
### **3.4.2.5.4. Resumen del esfuerzo dedicado en el sprint**

A continuación, se presenta de manera gráfica los datos del esfuerzo en tiempo estimado y el real.



Gráfica 1 - Comparación del total de horas estimadas y reales

En el siguiente gráfico se puede observar en detalle las estimaciones y horas reales de cada una de las tareas implementadas.



Gráfica 2 - Comparación del total de horas estimadas y reales por tarea

### **3.4.3.Sprint 5 (15/06/2020 - 28/06/2020)**

#### **3.4.3.1. Planificación del sprint**

En el listado a continuación se detallan las tareas planificadas para este *sprint*.

- Recibir notificaciones de ML (R-05)
- Validar reclamo (R-06)
- Listado de todos los reclamos (R-07)
- Identificación visual (R-08)
- Filtrar por tiendas (R-09)
- Documentación

#### **3.4.3.2. Tareas Completadas/Realizadas**

Las horas de trabajo estimadas para el *sprint* son de 80 horas, finalizado el mismo se totalizan 73 horas reales dedicadas. Dentro de estas horas se incluyen diversos tipos de tareas ya sea de desarrollo, configuración y documentación. En esta tabla se detalla la distribución horaria entre las diferentes tareas.

OBJETIVO	HORAS		ESTADO
	ESTIMADAS	REALES	
Recibir notificaciones de ML (R-05)	6	13.5	Completado
Validar reclamo (R-06)	10	11	Completado
Listado de todos los reclamos (R-07)	30	24	Completado
Identificación visual (R-08)	8	4.5	Completado
Filtrar por tiendas (R-09)	20	14	Completado
Documentación	6	6	Completado

Durante este *sprint* se llevaron a cabo diversas tareas, completándose el desarrollo de varios de los requerimientos funcionales de este proyecto.

En un principio se continuó y se pudo finalizar con la tarea recibir notificaciones de ML la cual había comenzado en el *sprint* 4. Este requerimiento se desarrolló en un mayor tiempo que el estimado, pero se consideró que era fundamental dedicarle este tiempo, ya que es un pilar importante en el proyecto, sumado a que el cliente lo había calificado como un requerimiento de prioridad alta.

Otra de las tareas que se llevó a cabo fue la validación de los reclamos. En una de las reuniones que se realizaron con el cliente, habían advertido a raíz de su experiencia en el trabajo con ML, que algunas veces ML puede enviar una notificación que no es perteneciente a uno de sus usuarios o no es sobre una venta realizada por uno de sus usuarios. Por lo que previamente a guardar la información del reclamo en la base de datos, se realiza esta validación para asegurarse de guardar solo los reclamos pertenecientes a usuarios de nuestro *software* y de órdenes existentes.

En este *sprint* se comenzó con la construcción del *home* de la aplicación, la cual contiene el listado de todos los reclamos del usuario con algunos datos importantes de

los mismos. Se listan primero los reclamos abiertos luego los cerrados y ordenados por precio y antigüedad descendente y ascendente respectivamente. Esta tarea de listar todos los reclamos se implementó en un menor tiempo que el estimado. Cabe destacar que del total de horas dedicadas a este requerimiento la mayor carga horaria estuvo a nivel del desarrollo del *frontend*.

Otro de los objetivos de este *sprint* fue el desarrollo del requerimiento identificación visual, el cual implicaba remarcar con un borde rojo los reclamos abiertos que estén esperando por la respuesta del vendedor. El mismo se completó en un menor tiempo que el estimado.

Durante este *sprint* también se llevó a cabo la tarea para permitir al usuario filtrar los reclamos por sus tiendas, en el caso que el usuario tenga multi tiendas. Esta tarea se pudo completar y las horas reales dedicadas a la misma fue menor que las estimadas. Cabe destacar que no fue posible probar este requerimiento con datos reales, ya que se solicitó al cliente si nos podían brindar los datos de un usuario con multi tiendas de *testing* y nos explicaron que no es posible crear usuario de *testing* con multi tiendas, ya que las tiendas las tiene que aprobar ML. Para comprobar el correcto funcionamiento de este requerimiento se cargaron datos de prueba en la base de datos y a partir de esto se realizaron las pruebas.

Por último, otro de los objetivos de este *sprint*, así como de los siguientes *sprints*, es el documentar los avances que se van realizando en el proyecto.

#### **3.4.3.3. Tareas no completadas**

En este *sprint* se logró cumplir con todas las tareas planificadas a realizarse en el mismo.

#### **3.4.3.4. Entrega del trabajo a los interesados**

Al finalizar este *sprint* se realizó la segunda muestra de los avances en el proyecto, se agendó una reunión con el cliente por video llamada, en la misma se mostró todo lo trabajado del *sprint 5*. Se presentó la implementación de los requerimientos listado de todos los reclamos, validar reclamo, identificación visual y filtrar por tiendas. El cliente se mostró conforme y satisfecho con todos los avances que se mostraron.

#### **3.4.3.5. Cierre/Revisión del Sprint**

##### **3.4.3.5.1. Desvíos y gestión de riesgos**

Al finalizar el *sprint* podemos llegar a la conclusión de que existió un mayor desvío en la estimación del esfuerzo que en los *sprints* anteriores, este desvío no implicó ningún perjuicio para el proyecto ya que el saldo de horas fue a favor. Si bien la estimación del esfuerzo fue de 80 horas, las horas reales de trabajo fueron 73 lográndose realizar todas las tareas y objetivos planteados para el *sprint* en un menor tiempo. Las diferencias en la estimación del tiempo de trabajo fueron tanto por una subestimación como una sobreestimación del tiempo para cada una de las diferentes tareas que comprenden este *sprint*. Se puede observar que la integración con la API de ML implicó dedicar una mayor cantidad de horas que las estimadas en un principio, mientras que las otras tareas

se desarrollaron en un menor tiempo. El menor tiempo en que se desarrollaron estas tareas se puede deber a la curva de aprendizaje, la cual en un principio fue más alta y a este punto ya se cuenta con un mayor dominio de las herramientas y tecnologías con las que se está trabajando. Como cierre del sprint se puede destacar que a pesar de que una de las tareas implicó un mayor tiempo de desarrollo se logró completar todas las tareas planificadas para el mismo. Por último, se subraya como positivo este proceso de aprendizaje, el cual es sumamente valioso para nosotras ya que nos permite adquirir mayor experiencia en la estimación de horas y conocer nuestra productividad.

#### **3.4.3.5.2. Nuevos requerimientos surgidos**

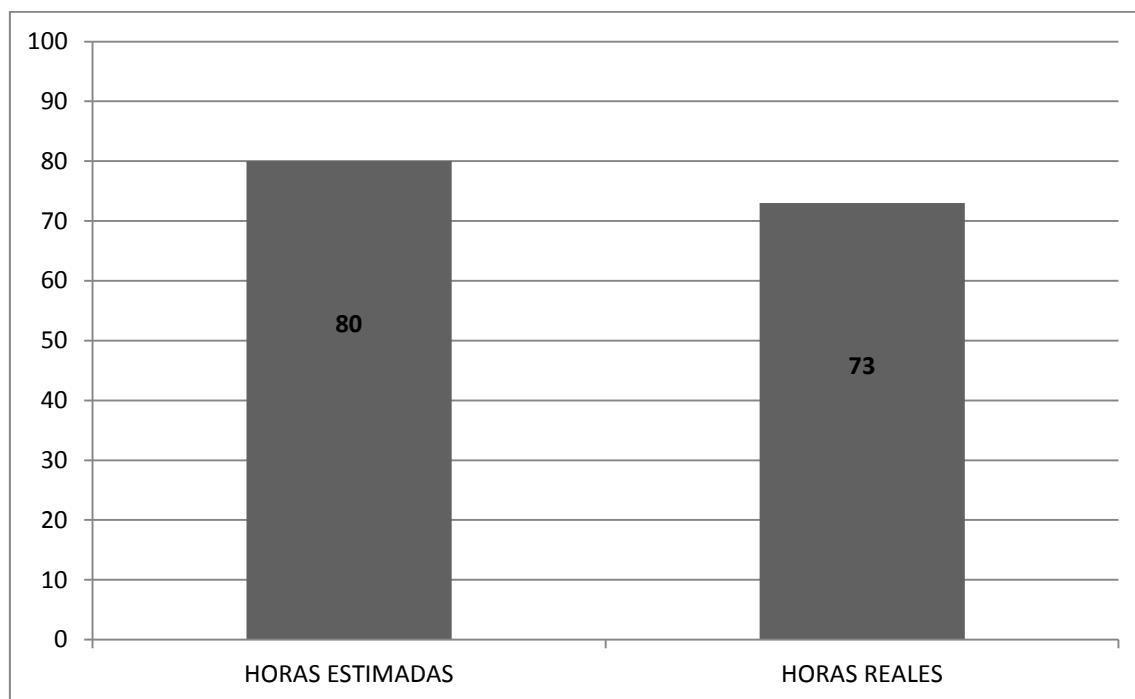
Durante este *sprint* no se incorporaron nuevos requerimientos.

#### **3.4.3.5.3. Ajuste del plan de trabajo del próximo sprint**

No se realizarán ajustes en el trabajo del próximo sprint por lo que se planifica continuar con el cronograma especificado en el anteproyecto.

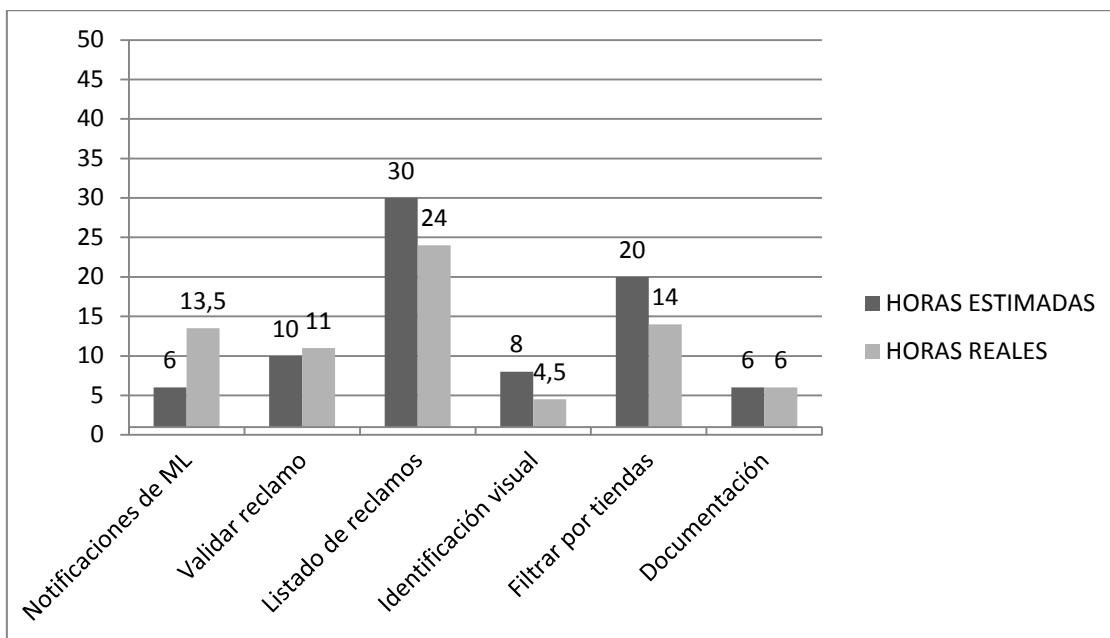
#### **3.4.3.5.4. Resumen del esfuerzo dedicado en el sprint**

A continuación, se presenta de manera gráfica los datos del esfuerzo en tiempo estimado y el real.



Gráfica 1 - Comparación del total de horas estimadas y reales

En el siguiente gráfico se puede observar en detalle las estimaciones y horas reales de cada una de las tareas implementadas.



Gráfica 2 - Comparación del total de horas estimadas y reales por tarea

### **3.4.4.Sprint 6 (29/06/2020 - 12/07/2020)**

#### **3.4.4.1. Planificación del sprint**

En el listado a continuación se detallan las tareas planificadas para este *sprint*.

- Búsqueda por filtro (R-10)
- Ver detalle del reclamo (R-11)
- Documentación

#### **3.4.4.2. Tareas Completadas/Realizadas**

Las horas de trabajo estimadas para el *sprint* son de 80 horas, finalizado el mismo se totalizan 79.5 horas reales dedicadas. Dentro de estas horas se incluyen diversos tipos de tareas ya sea de desarrollo, configuración, *testing* y documentación. En esta tabla se detalla la distribución horaria entre las diferentes tareas.

OBJETIVO	HORAS		ESTADO
	ESTIMADAS	REALES	
Búsqueda por filtro (R-10)	30	28	Completado
Ver detalle del reclamo (R-11)	40	41	Completado
Documentación	10	10.5	Completado

Uno de los objetivos de este *sprint* fue permitir al usuario filtrar los reclamos por los siguientes criterios de búsqueda: por estado (abierto o cerrado), por estado de la respuesta (esperando tu respuesta, esperando respuesta del comprador o esperando respuesta de mercado libre) o por tipo (PDD o PNR).

Durante este *sprint* se comenzó a desarrollar el requerimiento que permite al usuario ver el detalle del reclamo. En el detalle del reclamo se puede visualizar el id del mismo, fecha de iniciado el reclamo, un indicador de si afecta la reputación, de quien se espera la respuesta, etapa, estado y si la compra fue recibida o no por el comprador; también se puede ver el detalle de la venta (nombre del artículo, URL de la publicación, precio, cantidad vendida, foto, fecha de la venta, el nombre y apodo del comprador). La tarea de ver la información de los mensajes del reclamo se desarrollará en el siguiente *sprint* ya que está planificado por cronograma (ver capítulo 2.11.7 Cronograma de trabajo) para ser realizado en el siguiente *sprint*.

Por último, otro de los objetivos de este *sprint*, así como de los siguientes *sprints*, es el documentar los avances que se van realizando en el proyecto y la documentación de la API en Swagger.

#### **3.4.4.3. Tareas no completadas**

En este *sprint* se logró cumplir con todas las tareas planificadas a realizarse en el mismo.

#### **3.4.4.4. Entrega del trabajo a los interesados**

Luego de finalizar el *sprint* se realizó la tercera muestra al cliente con los avances del proyecto. Se realizó una reunión por video llamada, en la misma se presentó la implementación de los requerimientos búsqueda por filtro y ver el detalle del reclamo. El cliente se mostró conforme y satisfecho con todos los avances que se mostraron. Solicitaron la incorporación visual en listado de los reclamos de si el reclamo afecta la reputación del vendedor y el identificador de la orden de compra en el detalle del reclamo. Estas incorporaciones solicitadas se estimarán para ser incluidas en el siguiente *sprint*.

#### **3.4.4.5. Cierre/Revisión del Sprint**

##### **3.4.4.5.1. Desvíos y gestión de riesgos**

Al finalizar el *sprint* se puede llegar a la conclusión de que no existió un gran desvío con las estimaciones de esfuerzo.

En este *sprint* se enfrentó dos situaciones que no estaban planificadas y las mismas implicaron que se requiriera un mayor esfuerzo para poder solucionarlas. Sin embargo, estos inconvenientes se pudieron sortear satisfactoriamente sin que existiera un desvío en las estimaciones del tiempo.

La primera complejidad que surgió fue mientras se trabajaba en el detalle del reclamo, luego de modificar un atributo en el modelo reclamo, volvimos a utilizar los servicios que se conectan con la API de ML para traer y persistir en la base de datos los reclamos del usuario de *testing*; esta funcionalidad ya estaba aprobada y testeada en *sprints* anteriores. Al utilizarla, se pudo detectar que no estaba funcionando de la manera correcta al no cargarse los reclamos de ese usuario, la primera conclusión que se tuvo fue que probablemente durante el desarrollo del nuevo requerimiento habíamos modificado algo en ese servicio. Después de un par de horas *debugueando* el código sin encontrar el error, se decidió probar el método que devuelve todos los reclamos en la API de ML utilizando una herramienta externa como Postman, donde se pudo ver que este método no estaba devolviendo los reclamos que sabíamos existían. Esto permitió descartar la primera teoría de un error en nuestro código, pero aún no se tenía una idea clara de qué estaba ocurriendo en la API de ML. Al día siguiente se volvió a probar con Postman llamar a este método de la API de ML y devolvió los reclamos esperados, por lo que concluimos que la noche anterior la API de ML estuvo con fallas o con algún problema de conexión a su base de datos. También se arribó a esta conclusión ya que el cliente en una de las reuniones nos mencionó que en varias oportunidades habían detectado inconvenientes con la API de ML y el cliente cuenta con gran experiencia trabajando con ML. Lamentablemente requirió dedicar horas buscando en el código un error que no existía, sin embargo, se destaca como positivo el identificar este problema externo a nosotras ya que puede volver a existir en un futuro. Nos permitió obtener el aprendizaje de verificar el estado la API de ML con una herramienta externa ante un problema similar. También nos permitió evaluar y planificar un plan de contingencia por la posibilidad de que esto vuelva a ocurrir el día de la defensa del proyecto, donde sin la API de ML no se podría realizar una completa demostración de las funcionalidades del proyecto. La forma que se planifica implementar el plan de

contingencia es grabando previamente una presentación mostrando las funcionalidades del *software*.

La segunda dificultad que se presentó implicó dedicar un total de 10 horas de trabajo, esta dificultad se debió a un error que mostraba el entorno de desarrollo utilizado (Visual Studio) luego de desarrollar una nueva funcionalidad. El error que se mostraba decía que estaba tardando más tiempo del esperado para su iniciación, por lo que no se podía probar esa funcionalidad, ni ninguna otra. Se comenzó a buscar información sobre el error y no se encontró información muy clara. Quizás por la poca experiencia que teníamos seguimos lo que se encontró para solucionarlo, los mismos adjudicaban ese error a una incompatibilidad de Visual Studio con el Sistema Operativo Windows 7 (el mismo estaba instalado en la máquina). Se comenzó desinstalando y volviendo a instalar Visual Studio sin poder solucionar el error, por lo que se continuó buscando información sobre el error y probando las sugerencias de solución encontradas. Luego de probar por varias horas sin lograr ningún tipo de avance, se decidió probar el funcionamiento de la aplicación con otra versión del proyecto que sabíamos funcionaba. Esto fue posible ya que el código se está manejando con un sistema de versionado y se cuenta con diferentes ramas. Se decidió probar ejecutando el código que teníamos en la rama “*master*”, ya que ese código estaba testeado y guardado un par de versiones antes a la que se estaba trabajando. Cuando se probó pudimos identificar que el *debugger* funcionaba sin problemas, lo que nos permitió saber que el error se debía al código de la versión actual y no al programa. Después de una intensa búsqueda se logró encontrar que el error se debía a una referencia cíclica entre las clases (una clase estaba llamando a otra y ésta a la primera), lo que causaba un bucle infinito y no permitía iniciar el *debugger*.

Tanto en el primer caso como en el segundo fue algo frustrante pensar en el tiempo y recursos destinados a la solución de un problema tan sencillo, por otro lado, se destaca que se pudo ganar experiencia y todos los aprendizajes obtenidos para el manejo de futuros errores. Pese a estos contra tiempos, se logró alcanzar la meta y terminar todas las tareas planificadas para este *sprint*.

#### **3.4.4.5.2. Nuevos requerimientos surgidos**

Durante este *sprint* no se incorporaron nuevos requerimientos.

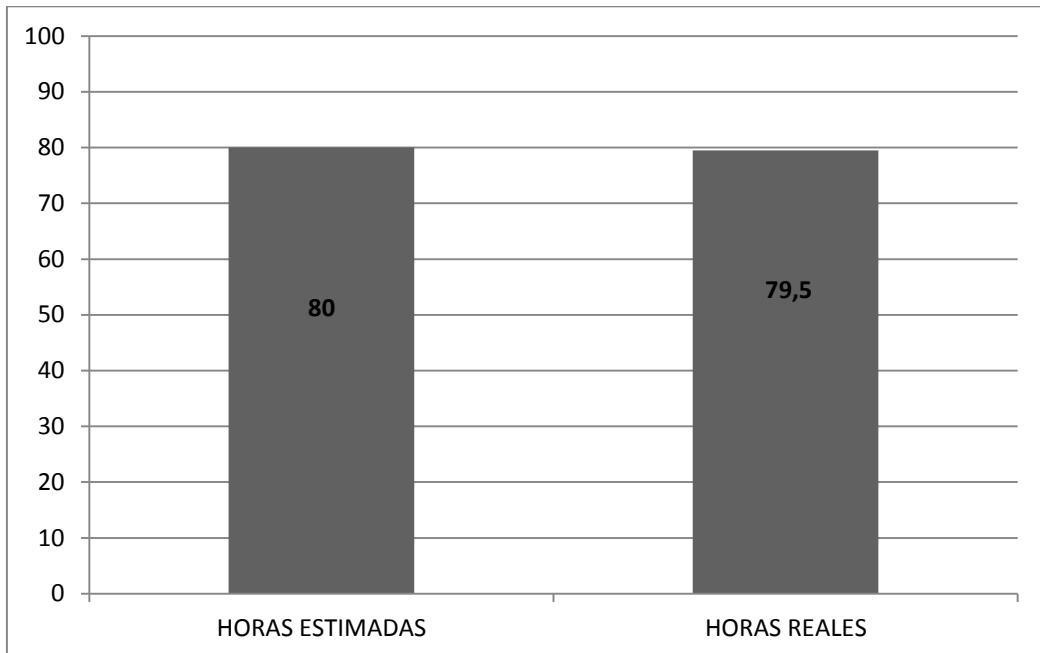
#### **3.4.4.5.3. Ajuste del plan de trabajo del próximo sprint**

Se realizarán ajustes en el trabajo del próximo *sprint*, ya que luego de la muestra realizada al cliente el mismo solicitó agregar dos datos en la información que se muestra en la vista del listado de los reclamos y en el detalle del reclamo. Se estima un esfuerzo de 2 horas para esta tarea. En el próximo *sprint* se incluirán todas las tareas planificadas durante el anteproyecto para el *sprint* 7 sumado a la siguiente tarea:

R-35 Arreglos visuales solicitados por el cliente: incorporación visual en listado de los reclamos de si el reclamo afecta la reputación del vendedor y del identificador de la orden de compra en el detalle del reclamo.

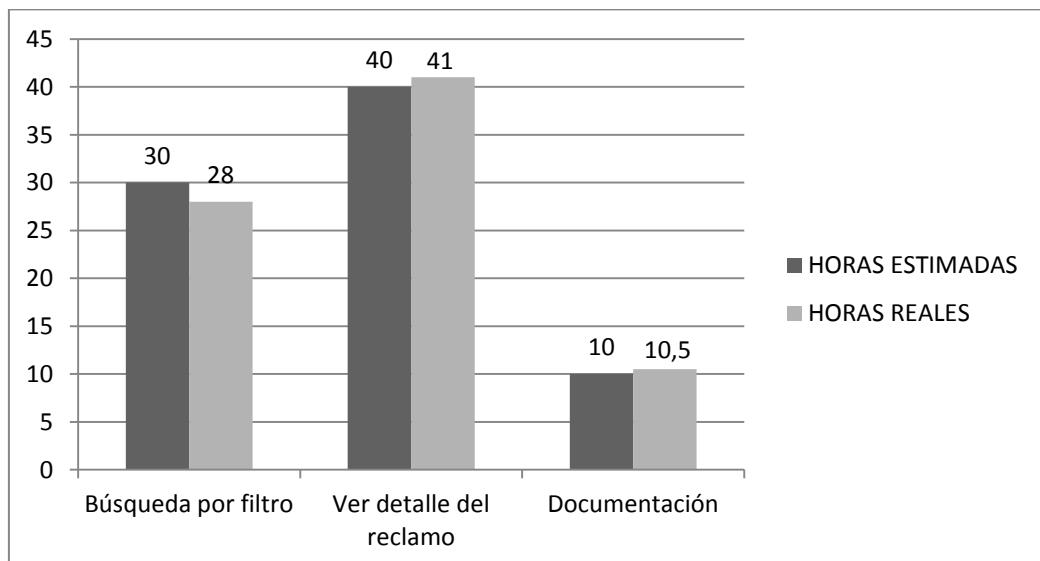
#### **3.4.4.5.4. Resumen del esfuerzo dedicado en el sprint**

A continuación, se presenta de manera gráfica los datos del esfuerzo en tiempo estimado y el real.



Gráfica 1 - Comparación del total de horas estimadas y reales

En el siguiente gráfico se puede observar en detalle las estimaciones y horas reales de cada una de las tareas implementadas.



Gráfica 2 - Comparación del total de horas estimadas y reales por tarea

### **3.4.5.Sprint 7 (13/07/2020 - 26/07/2020)**

#### **3.4.5.1. Planificación del sprint**

En el listado a continuación se detallan las tareas planificadas para este *sprint*.

- Ver la información de los mensajes - Ver detalle del reclamo (R-11)
- Ver resolución propuesta por el comprador (R-13)
- Resolución de reclamos (R-14)
- Resolución de reembolso (R-15)
- Incorporación visual en listado de los reclamos de si el reclamo afecta la reputación del vendedor y del identificador de la orden de compra en el detalle del reclamo. (R-35)
- Documentación

#### **3.4.5.2. Tareas Completadas/Realizadas**

Las horas de trabajo estimadas para el *sprint* son de 88 horas, finalizado el mismo se totalizan 79.5 horas reales dedicadas. Dentro de estas horas se incluyen diversos tipos de tareas ya sea de desarrollo, *testing* y documentación. En esta tabla se detalla la distribución horaria entre las diferentes tareas.

OBJETIVO	HORAS		ESTADO
	ESTIMADAS	REALES	
Ver la información de los mensajes (R-11)	10	10	Completado
Ver resolución propuesta por el comprador (R-13)	10	10	Completado
Resolución de reclamos (R-14)	40	38	Completado
Resolución de reembolso (R-15)	12	5.5	Completado
Arreglos visuales solicitados por el cliente (R-35)	2	2	Completado
Eliminación de todos los datos del usuario si se dispara una excepción (R-41)	6	6	Completado
Documentación	8	8	Completado

#### **3.4.5.3. Tareas no completadas**

En este *sprint* se logró cumplir con todas las tareas planificadas a realizarse en el mismo.

#### **3.4.5.4. Entrega del trabajo a los interesados**

Luego de la reunión donde se presentaron los cambios implementados en el *sprint* 6, cuando se realizaron los cambios visuales solicitados por el cliente, se le envió un mail al cliente con una demostración visual de la implementación de los mismos. En ese mail además se les pidió por escrito algún tipo de devolución de cómo se han sentido

trabajando con nosotras y su impresión sobre el avance del proyecto (ver respuesta del cliente en Anexo 5 – Validación del cliente sobre avance del producto (*sprint* 7)).

Al finalizar el *sprint* se realizó una muestra de los requerimientos trabajados, el cliente se mostró conforme con los avances mostrados.

### **3.4.5.5. Cierre/Revisión del Sprint**

#### **3.4.5.5.1. Desvíos y gestión de riesgos**

Al finalizar el *sprint* se puede llegar a la conclusión de que no existió un desvío con las estimaciones de esfuerzo.

En este *sprint* surgieron dos nuevos requerimientos, uno pedido por el cliente y el otro contemplado por las desarrolladoras causando un esfuerzo adicional de 8 horas para cumplir los mismos. Por otra parte, en dos de los requerimientos que estaban planificados para este *sprint* se tuvo un desempeño mejor al planificado, lo que hizo que al finalizarlo se compensaran las horas que requirieron estos nuevos requerimientos. Sin embargo, cuando se incluyeron en este *sprint*, no se sabía que se iba a tener un balance positivo al final, pero como el esfuerzo que conllevaban era de pocas horas, se decidió incluirlos en este *sprint* sin tener que hacer una reprogramación del cronograma.

#### **3.4.5.5.2. Nuevos requerimientos surgidos**

Durante este *sprint* se incorporaron dos nuevos requerimientos.

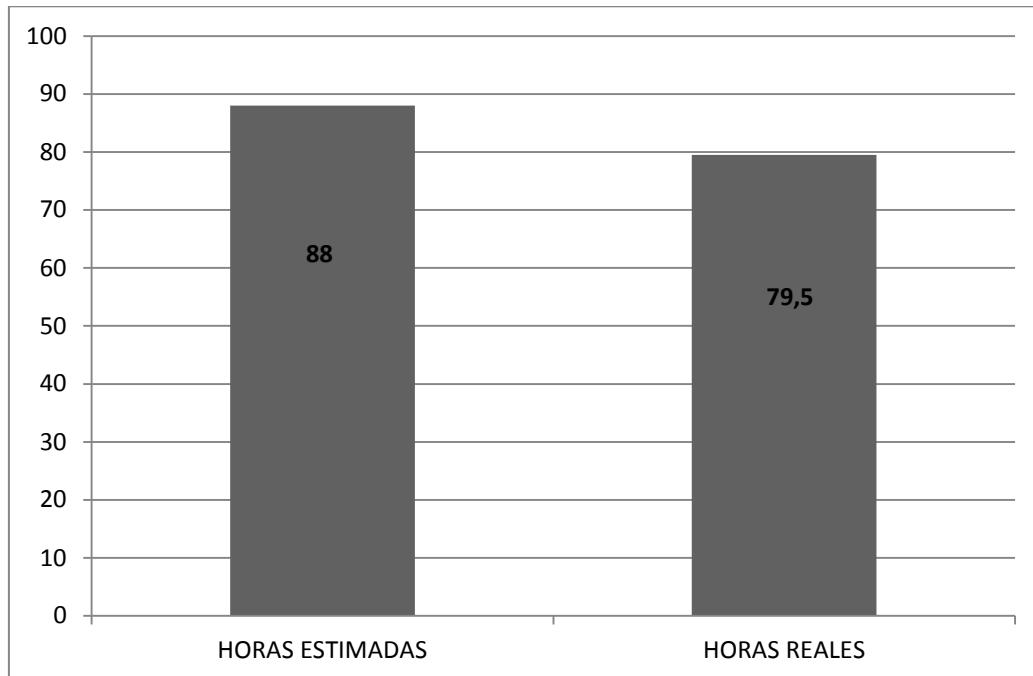
- R-35 Arreglos visuales solicitados por el cliente: incorporación visual en listado de los reclamos de si el reclamo afecta la reputación del vendedor y del identificador de la orden de compra en el detalle del reclamo.
- Eliminación de todos los datos del usuario si se dispara una excepción. El equipo de desarrollo detectó que existe un riesgo de que ocurran fallas vinculadas a la API de ML. Ante una eventual excepción durante la vinculación de la cuenta del usuario de ML con nuestra base de datos parte de la información de este podría quedar persistida y otra no ya que esto se hace en más de una transacción. Esto provocaría que se insertara información no consistente en la base de datos y cuando el usuario vuelva a intentar asociar su cuenta se generarán conflictos. Para mitigar el riesgo de que esto ocurra, ante una excepción que cancele alguna de estas transacciones se decidió eliminar toda la información que se pudiera haber persistido en la vinculación. De esta forma cuando el usuario lo vuelva a intentar la información será consistente.

#### **3.4.5.5.3. Ajuste del plan de trabajo del próximo sprint**

No se realizarán ajustes en el trabajo del próximo *sprint* por lo que se planifica continuar con el cronograma especificado en el anteproyecto.

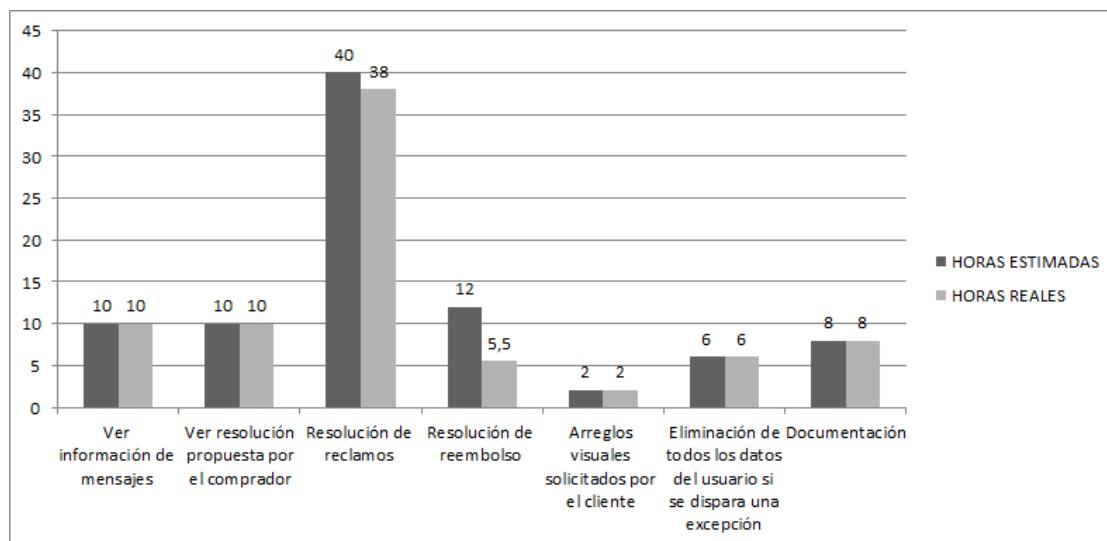
### 3.4.5.5.4. Resumen del esfuerzo dedicado en el sprint

A continuación, se presenta de manera gráfica los datos del esfuerzo en tiempo estimado y el real.



Gráfica 1 - Comparación del total de horas estimadas y reales

En el siguiente gráfico se puede observar en detalle las estimaciones y horas reales de cada una de las tareas implementadas.



Gráfica 2 - Comparación del total de horas estimadas y reales por tarea

### **3.4.6.Sprint 8 (27/07/2020 - 09/08/2020)**

#### **3.4.6.1. Planificación del sprint**

En el listado a continuación se detallan las tareas planificadas para este *sprint*.

- Comunicación entre las partes (R-12)
- Comprobante de envío (R-16)
- Visualizar mediación (R-17)
- Documentación

#### **3.4.6.2. Tareas Completadas/Realizadas**

Las horas de trabajo estimadas para el *sprint* son de 80 horas, finalizado el mismo se totalizan 91 horas reales dedicadas. Dentro de estas horas se incluyen diversos tipos de tareas ya sea de desarrollo, configuración y documentación. En esta tabla se detalla la distribución horaria entre las diferentes tareas.

OBJETIVO	HORAS		ESTADO
	ESTIMADAS	REALES	
Comunicación entre las partes (R-12)	25	32	Completado
Comprobante de envío (R-16)	30	35	Completado
Visualizar mediación (R-17)	10	9	Completado
Documentación	15	15	Completado

En este *sprint* se llevaron a cabo diversas tareas, completándose el desarrollo de tres grandes requerimientos funcionales del proyecto.

Durante este *sprint* se implementó el requerimiento que le permite al usuario enviar mensajes al comprador o a ML en un reclamo abierto. Si el reclamo está en mediación con ML los mensajes se envían a ML ya que la comunicación deja de ser vendedor-comprador y es ML quien actúa como intermediario. Los mensajes se pueden enviar con o sin archivos adjuntos.

También se desarrolló el requerimiento que permite al usuario subir el comprobante de envío. Una vez que el comprador inicia un reclamo porque quiere recibir el producto que compró, si el vendedor ya realizó el envío del producto y tiene evidencias, puede subir la evidencia a través de la aplicación. A su vez, tiene la opción de subir una probable fecha de envío del producto en caso de que aún no lo haya enviado. En esta funcionalidad también se le brinda la opción al usuario de adjuntar un archivo como evidencia.

Otra de las tareas que se llevó a cabo fue el mostrar dentro del detalle del reclamo si el mismo está en mediación con ML (Visualizar mediación R-17).

Por último, otro de los objetivos del *sprint*, es el documentar los avances que se van realizando en el proyecto.

### **3.4.6.3. Tareas no completadas**

En este *sprint* se logró cumplir con todas las tareas planificadas a realizarse en el mismo.

### **3.4.6.4. Entrega del trabajo a los interesados**

Al finalizar el *sprint* se realizó la quinta muestra al cliente con los avances del proyecto. La misma se realizó por video llamada y se presentó la implementación de los requerimientos comunicación entre las partes, subir comprobante de envío y visualizar si un reclamo está en mediación con ML. El cliente aprobó los avances mostrados, mostrándose conforme y satisfecho con todas las funcionalidades presentadas. En el Anexo 6 - Validación del cliente sobre avance del producto (*sprint* 8) se puede ver la respuesta del cliente validando el avance del proyecto mostrado en este *sprint*.

### **3.4.6.5. Cierre/Revisión del Sprint**

#### **3.4.6.5.1. Desvíos y gestión de riesgos**

Durante este *sprint* se enfrentaron varios desafíos, uno de los más complejos fue el manejo de los archivos adjuntos tanto desde el *backend* como del *frontend*. Esto implicó que se tuviera que dedicar más horas que las planificadas al desarrollo del requerimiento de envío de mensajes y subir comprobante de envío. Para poder cumplir con el desarrollo de estos requerimientos se tuvo que dedicar más horas de trabajo total en el *sprint* lo cual no fue un problema ya que se disponía de estas horas extras. Se dispone de estas horas ya que durante la planificación del proyecto se estimó dedicarle menos horas de trabajo en los días laborales y contar con esas horas como forma de contingencia en caso de ser necesario utilizarlas.

Estos dos requerimientos implicaron un mayor tiempo de trabajo que el planificado, lo que se puede atribuir a una falta de experiencia en la estimación del esfuerzo. Sin embargo, esto no implicó un perjuicio para el proyecto gracias a las previsiones que se realizaron en el análisis de riesgos, pudiéndose completar todas las tareas que estaban planificadas.

#### **3.4.6.5.2. Nuevos requerimientos surgidos**

Durante el desarrollo de este *sprint* se encontró un nuevo requerimiento que no estaba planificado en el anteproyecto. El mismo es mostrar al usuario dentro del detalle del reclamo los datos de cuando se sube la evidencia de envío del producto. Se planifica incorporar este nuevo requerimiento en el próximo *sprint*.

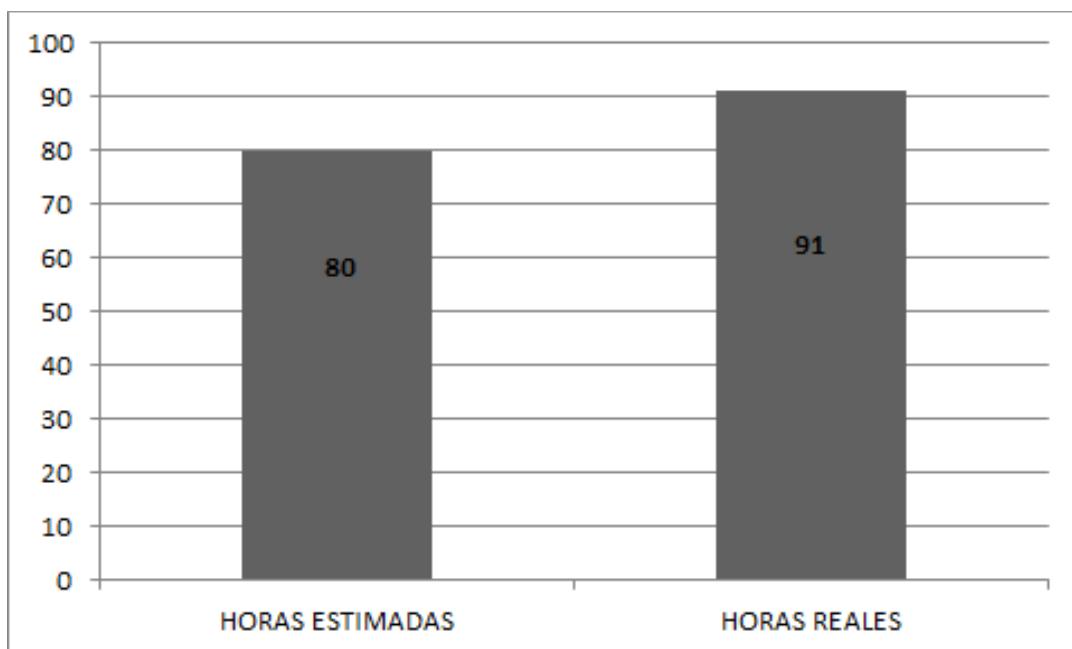
#### **3.4.6.5.3. Ajuste del plan de trabajo del próximo sprint**

Se realizarán ajustes en el trabajo del próximo *sprint*, se planifica desarrollar los requerimientos iniciar mediación (R-18), *dashboard* (R-21) y comenzar con la obtención de reportes (R-19). Se invertirá el orden del desarrollo comenzando con el requerimiento de *dashboard* antes que la obtención de reportes ya que es necesario desarrollar primero el requerimiento R-21 para luego comenzar con el R-19. A su vez se incorporará el desarrollo del nuevo requerimiento que surgió en este *sprint* (visualizar

detalle de evidencia de envío). Se estima un esfuerzo de 7 horas para este nuevo requerimiento.

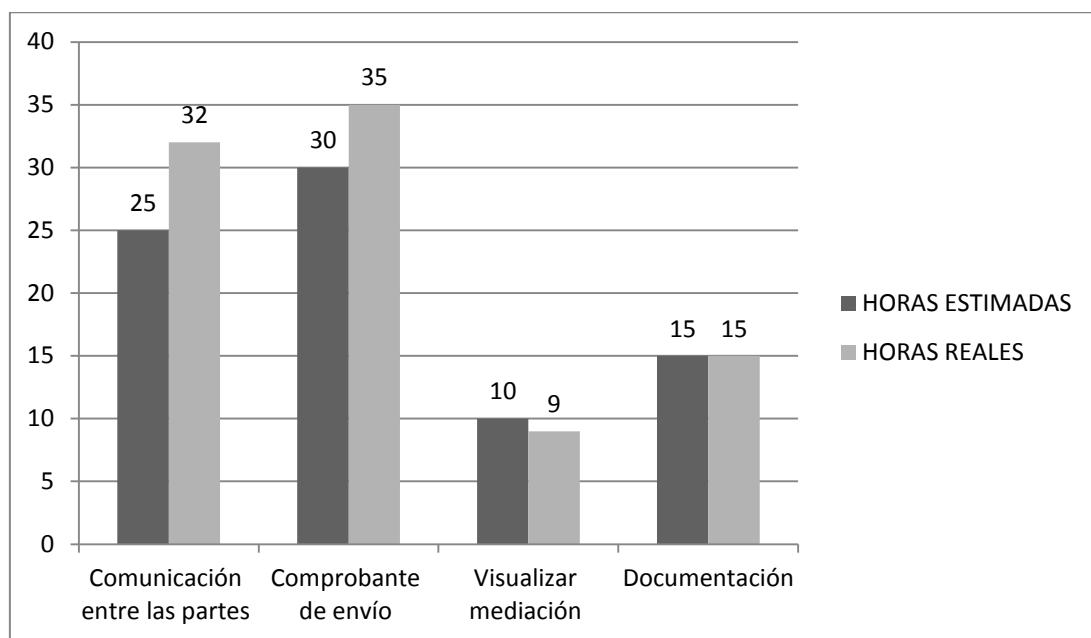
#### **3.4.6.5.4. Resumen del esfuerzo dedicado en el sprint**

A continuación, se presenta de manera gráfica los datos del esfuerzo en tiempo estimado y el real.



Gráfica 1 - Comparación del total de horas estimadas y reales

En el siguiente gráfico se puede observar en detalle las estimaciones y horas reales de cada una de las tareas implementadas.



Gráfica 2 - Comparación del total de horas estimadas y reales por tarea

### **3.4.7.Sprint 9 (10/08/2020 - 23/08/2020)**

#### **3.4.7.1. Planificación del sprint**

En el listado a continuación se detallan las tareas planificadas para este *sprint*.

- Visualizar detalle de evidencia de envío (R-42)
- Iniciar mediación (R-18)
- Obtención de reportes (R-19)
- *Dashboard* (R-21)
- Documentación

#### **3.4.7.2. Tareas Completadas/Realizadas**

Las horas de trabajo estimadas para el *sprint* son de 87 horas, finalizado el mismo se totalizan 80 horas reales dedicadas. Dentro de estas horas se incluyen diversos tipos de tareas ya sea de desarrollo, configuración, *testing* y documentación. En esta tabla se detalla la distribución horaria entre las diferentes tareas.

OBJETIVO	HORAS		ESTADO
	ESTIMADAS	REALES	
Visualizar detalle de evidencia de envío (R-42)	7	7.5	Completado
Iniciar mediación (R-18)	10	8	Completado
Obtención de reportes (R-19)	10	10	Completado
<i>Dashboard</i> (R-21)	40	34.5	Completado
Documentación	20	20	Completado

Durante este *sprint* se llevó a cabo el desarrollo de varios requerimientos funcionales del proyecto, completándose en su totalidad todas las tareas planificadas para el *sprint*. Como se mencionó en el capítulo anterior (3.5.6. Sprint 8) se realizaron ajustes al plan de trabajo de este *sprint* incorporándose un requerimiento nuevo y desarrollándose primero el requerimiento de *dashboard* antes que la obtención de reportes.

En este *sprint* se implementó el requerimiento que le permite al usuario visualizar el detalle de evidencia de envío. Si bien es un requerimiento nuevo que no estaba contemplado en el anteproyecto se decidió incorporarlo ya que el mismo le aporta valor al *software* y es de utilidad para el usuario. A su vez, se evaluó la viabilidad de incluirlo, estimándose el esfuerzo que implicaría desarrollarlo y finalmente se decidió incluirlo en este *sprint*.

Durante el *sprint* se desarrolló el requerimiento iniciar mediación, esto le permite al usuario solicitar mediación a ML, a partir de ese momento ML actúa como intermediario entre el vendedor y el comprador.

También se desarrolló el requerimiento de *Dashboard*, permitiéndole al usuario contar con un tablero donde puede visualizar de forma clara algunos datos útiles sobre sus reclamos en tiempo real.

Otra de las tareas que se llevó a cabo fue la obtención de reportes, este requerimiento le permite al usuario poder obtener los datos del *dashboard* a lo largo del tiempo, pudiendo filtrar por un rango de fechas.

Por último, otro de los objetivos del *sprint*, es el documentar los avances que se van realizando en el proyecto.

#### **3.4.7.3. Tareas no completadas**

En este *sprint* se logró cumplir con todas las tareas planificadas a realizarse en el mismo.

#### **3.4.7.4. Entrega del trabajo a los interesados**

Luego de finalizado el *sprint* se realizó una muestra al cliente con los avances del producto, presentándose los requerimientos desarrollados en el mismo. Se obtuvo un muy buen *feedback* por parte del cliente el cual se mostró muy conforme con los avances mostrados (ver en Anexo 7- Validación del cliente sobre avance del producto (*sprint* 9)). A su vez, se habló sobre incorporaciones que se podrían realizar al *software* los cuales serían de gran utilidad para el cliente, algunas de estas incorporaciones fueron planteadas por las desarrolladoras y otras por el cliente. Se planifica incorporar estos nuevos requerimientos en el próximo *sprint*.

#### **3.4.7.5. Cierre/Revisión del Sprint**

##### **3.4.7.5.1. Desvíos y gestión de riesgos**

En este *sprint* se logró completar y realizar todas las tareas que estaban planificadas para el mismo. Esto se logró en un menor tiempo que el estimado, ya que se había estimado 87 horas de trabajo y el esfuerzo real fue de 80 horas. Se puede concluir que, si bien existieron algunas desviaciones pequeñas en el tiempo, estas fueron a favor, completándose todas las tareas en un menor tiempo que el estimado. Este balance positivo en las horas se puede atribuir a una mayor eficiencia en el desarrollo, ya que se cuenta con mayor experiencia en las tecnologías utilizadas en el proyecto.

##### **3.4.7.5.2. Nuevos requerimientos surgidos**

Durante este *sprint* surgieron nuevos requerimientos que son R-37: *Dashboard* ampliado 2 y R-38: Filtrar reclamos en mediación con ML. En el *dashboard* se deberá mostrar la información del porcentaje de reclamos de tipo PDD y PNR en los últimos 3 meses y la cantidad de reclamos en mediación con ML. En el listado de todos los reclamos se debe permitir al usuario filtrar aquellos que se encuentran en mediación con ML. Se estimará el esfuerzo que implica incorporar estos nuevos requerimientos en el próximo *sprint*.

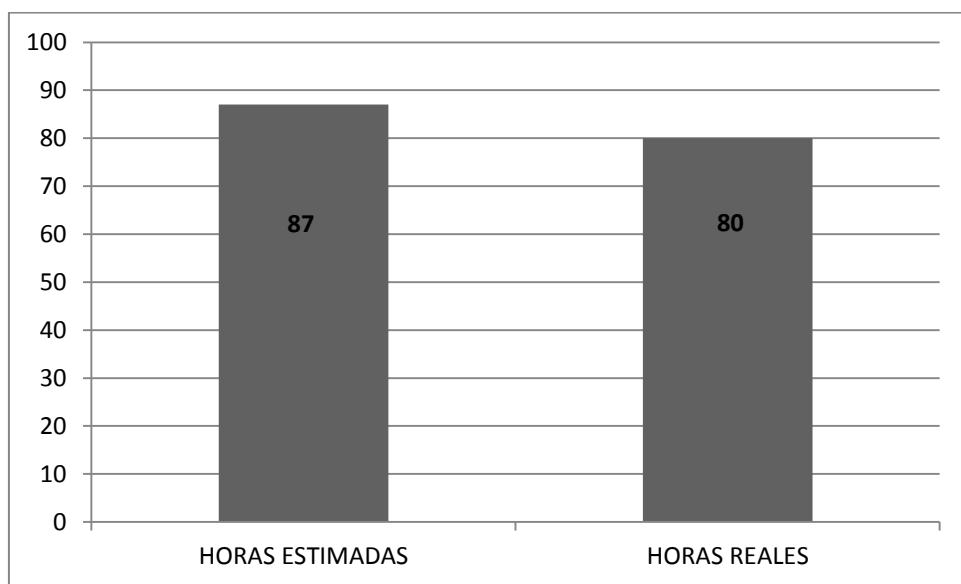
##### **3.4.7.5.3. Ajuste del plan de trabajo del próximo sprint**

Durante este *sprint* (*sprint* 9) al invertirse el orden en el desarrollo del requerimiento *dashboard* (R-21) y obtención de reportes (R-19), se planifica para el próximo *sprint* continuar con el desarrollo del requerimiento obtención de reportes, ya que esta tarea

estaba estimada en 40 horas de desarrollo. Sumado a la implementación de las tareas ya planificadas para el próximo *sprint* que son *Dashboard* ampliado, documentación y los nuevos requerimientos surgidos. Se estima un esfuerzo de 5 horas para el requerimiento R-37 *Dashboard* ampliado 2 y de 3 horas para el requerimiento R-38 Filtrar reclamos en mediación con ML. A su vez, dada la incorporación de estos nuevos requerimientos se realizará un ajuste en la estimación de horas dedicadas a la documentación siendo de 27 horas de documentación para el siguiente *sprint*. Por lo que a las horas que se habían estimado en el anteproyecto para el *sprint* 10 de documentación (que era 35 horas) se restaron 8 horas que es el esfuerzo estimado para los nuevos requerimientos.

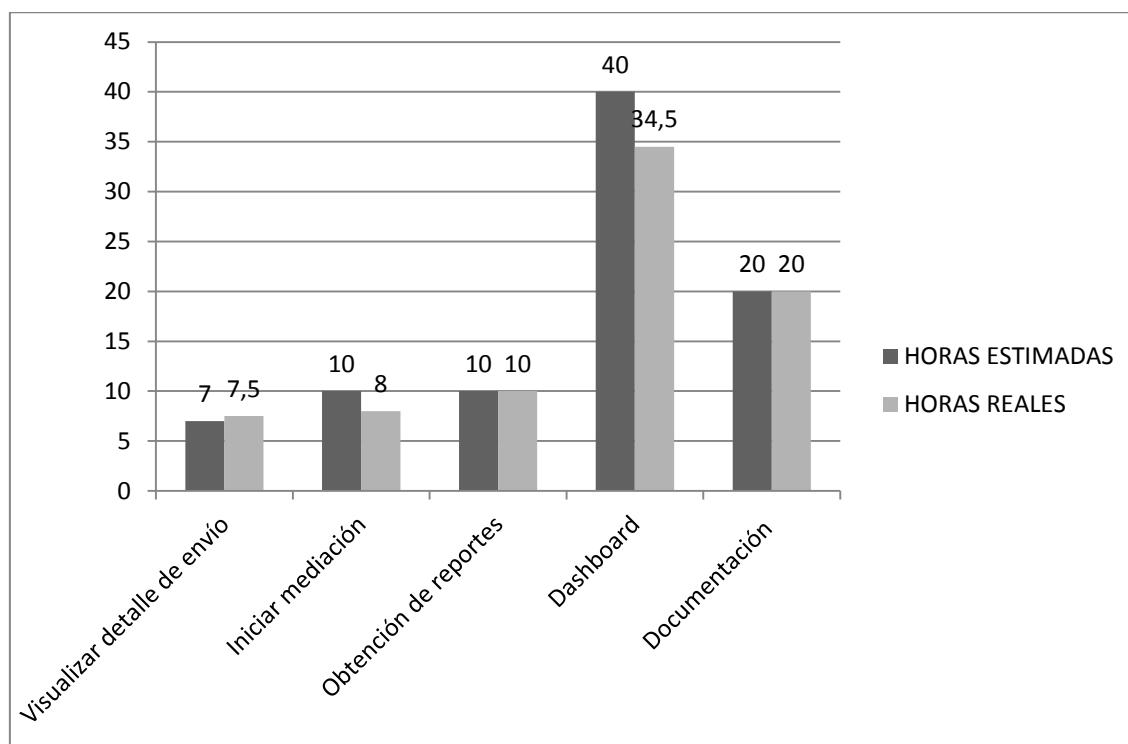
#### **3.4.7.5.4. Resumen del esfuerzo dedicado en el sprint**

A continuación, se presenta de manera gráfica los datos del esfuerzo en tiempo estimado y el real.



Gráfica 1 - Comparación del total de horas estimadas y reales

En el siguiente gráfico se puede observar en detalle las estimaciones y horas reales de cada una de las tareas implementadas.



Gráfica 2 - Comparación del total de horas estimadas y reales por tarea

### **3.4.8.Sprint 10 (24/08/2020 - 06/09/2020)**

#### **3.4.8.1. Planificación del sprint**

En el listado a continuación se detallan las tareas planificadas para este *sprint*.

- Obtención de reportes (R-19)
- *Dashboard* ampliado (R-22)
- *Dashboard* ampliado 2 (R-37)
- Filtrar reclamos en mediación con ML (R-38)
- Persistencia del log de excepciones (R-39)
- Documentación

#### **3.4.8.2. Tareas Completadas/Realizadas**

Las horas de trabajo estimadas para el *sprint* son de 85 horas, finalizado el mismo se totalizan 81 horas reales dedicadas. Dentro de estas horas se incluyen diversos tipos de tareas ya sea de desarrollo, configuración y documentación. En la tabla que se presenta a continuación se detalla la distribución horaria entre las diferentes tareas.

OBJETIVO	HORAS		ESTADO
	ESTIMADAS	REALES	
Obtención de reportes (R-19)	30	30	Completado
<i>Dashboard</i> ampliado (R-22)	15	11	Completado
<i>Dashboard</i> ampliado 2 (R-37)	5	5	Completado
Filtrar reclamos en mediación con ML (R-38)	3	3	Completado
Persistencia del log de excepciones (R-39)	5	5	Completado
Documentación	27	27	Completado

En este *sprint* se llevaron a cabo varias tareas, completándose el desarrollo de diferentes requerimientos funcionales. Algunos de estos requerimientos fueron planificados y estimados en el anteproyecto, mientras que otros surgieron durante el transcurso de desarrollo del proyecto.

Durante este *sprint* se completó la tarea de obtención de reportes, el desarrollo de este requerimiento se comenzó en el *sprint* 9 culminándose la implementación del mismo en este *sprint*. Este requerimiento le permite al usuario obtener los datos del *dashboard* a lo largo del tiempo, pudiendo filtrar los reportes por un rango de fechas.

También se desarrolló el requerimiento de *Dashboard* ampliado y *Dashboard* ampliado 2, estos requerimientos son un complemento del requerimiento *Dashboard* (R-21) ya que, si bien la funcionalidad es la misma, se le incorporaron nuevos datos al tablero, permitiéndole al usuario contar con más información sobre sus reclamos.

Otra de las tareas que se llevó a cabo fue filtrar reclamos en mediación con ML, este requerimiento le permite al usuario filtrar sus reclamos por un criterio nuevo, este filtro se suma a los filtros que ya se habían implementado anteriormente en la aplicación. El cliente consideró que era importante brindarle al usuario la posibilidad de filtrar todos

sus reclamos que están en mediación con ML, así como los reclamos que se encuentran abiertos y en mediación con ML.

En este *sprint* surgió un requerimiento nuevo persistencia del *log* de excepciones, este requerimiento no fue solicitado por el cliente, sino que nosotras consideramos importante contar con el registro de las excepciones que pudieran surgir.

Por último, otro de los objetivos de este *sprint*, es el documentar los avances que se van realizando en el proyecto y la documentación de la API en Swagger. Como se mencionó en el capítulo anterior se hizo un ajuste en la estimación de horas de esta tarea siendo 27 horas las estimadas.

#### **3.4.8.3. Tareas no completadas**

En este *sprint* se logró cumplir con todas las tareas planificadas a realizarse en el mismo.

#### **3.4.8.4. Entrega del trabajo a los interesados**

Durante este *sprint* le hicimos la entrega del código fuente al cliente, nos solicitaron el código para evaluar la manera que van a realizar la integración con su software. Si bien el *frontend* no estaba pensado para integrarse a su aplicación se interesaron en ver el código para evaluar la viabilidad de integrarlo.

También, se coordinó una reunión por video llamada con el cliente para hacer la última muestra con los requerimientos implementados en el *sprint* 10, el cliente destacó el buen trabajo, estando satisfecho con la muestra realizada. En el Anexo 8 - Validación del cliente sobre avance del producto (*sprint* 10) se puede ver la respuesta del cliente validando el avance del proyecto mostrado en este *sprint*.

#### **3.4.8.5. Cierre/Revisión del Sprint**

##### **3.4.8.5.1. Desvíos y gestión de riesgos**

En el transcurso de este *sprint* se logró completar todas las tareas planificadas y a su vez se incorporó un nuevo requerimiento. Si se compara las horas estimadas con las horas reales se puede observar que existió una pequeña desviación y que nuevamente esta desviación resultó en un balance positivo ya que se pudo completar todas las tareas planificadas para el *sprint* en un menor tiempo que el estimado. Una de las razones se puede deber a que se contaba con la experiencia del *sprint* anterior (*sprint* 9) en la realización del *dashboard* ya que la tarea que se realizó en menor tiempo que el estimado fue el requerimiento *Dashboard* ampliado.

##### **3.4.8.5.2. Nuevos requerimientos surgidos**

Durante este *sprint* surgió un requerimiento nuevo R-39: Persistencia del *log* de excepciones. Cuando se evaluaron los riesgos que podía tener este proyecto, se identificó que uno de ellos podía ser los algo frecuentes cambios en la API de ML, que podía ser el cambió de tipo de dato que espera un atributo, o la baja de un servicio de forma temporal. Por lo que creímos que sería muy importante poder persistir en la base de datos todo tipo de excepciones generadas cuando las respuestas a las *request* de ML

no son las esperadas en nuestra API. Se guarda la función donde se produjo, la descripción del error, la fecha, y el usuario al que se le disparó la excepción. Para luego chequear periódicamente los datos en la tabla correspondiente, con el fin de poder identificar eventuales fallas a las llamadas de los servicios de ML. Para poder chequear lo mismo creamos un *end point* donde se puede hacer una *request* y obtener todas las excepciones guardadas, en un rango de fecha brindado como parámetro.

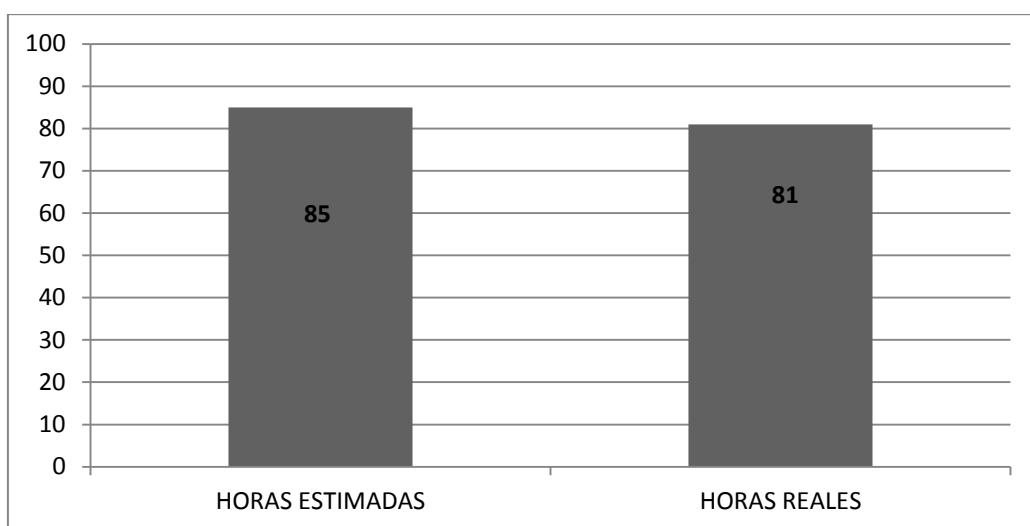
Se estimó el esfuerzo que implica incorporar este nuevo requerimiento en 5 horas y se decidió incorporar la implementación del mismo en este *sprint*.

#### **3.4.8.5.3. Ajuste del plan de trabajo del próximo sprint**

Este es el último *sprint* por lo que no se realizarán ajustes. Las próximas semanas serán destinadas a continuar con la documentación y *testing* de la aplicación.

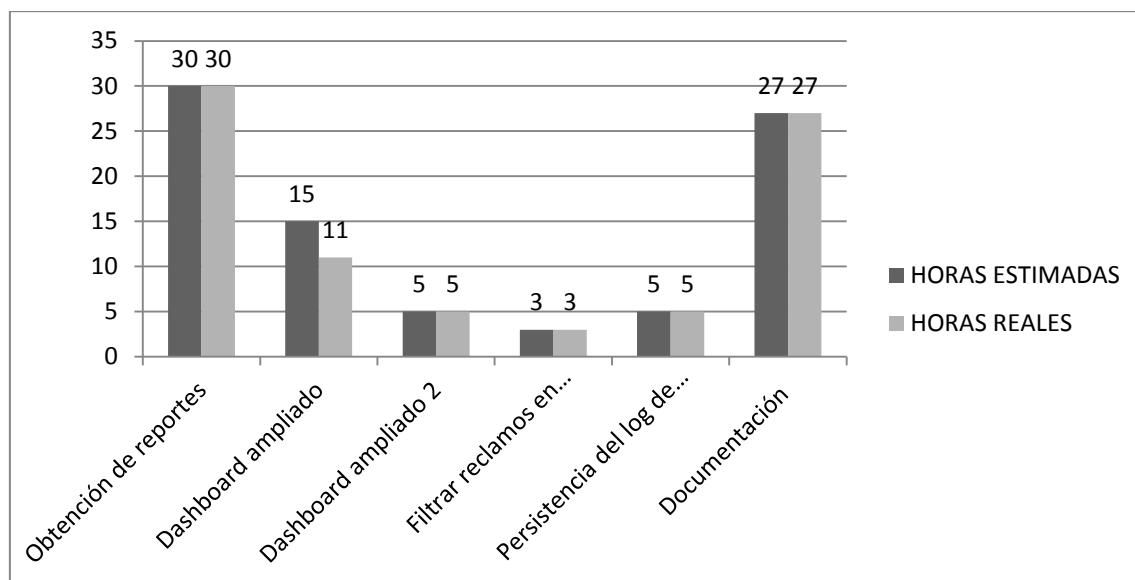
#### **3.4.8.5.4. Resumen del esfuerzo dedicado en el sprint**

A continuación, se presenta de manera gráfica los datos del esfuerzo en tiempo estimado y el real.



Gráfica 1 - Comparación del total de horas estimadas y reales

En el siguiente gráfico se puede observar en detalle las estimaciones y horas reales de cada una de las tareas implementadas.



Gráfica 2 - Comparación del total de horas estimadas y reales por tarea

### **3.4.9. Sprint de cierre (07/09/2020 - 20/09/2020)**

Este *sprint* se dedicó al cierre de la documentación incluyendo la realización de los diagramas de implementación. También se tuvo la reunión de cierre con el cliente y se relevó el grado de satisfacción a través de un cuestionario. Por otro lado se siguió con el plan de *testing* de la aplicación. A su vez, se aprovechó el tiempo para realizar algunos cambios visuales del *frontend*, esto no fue solicitado por el cliente, ya que su interés está puesto en el *backend*, sino que consideramos que puede ser valiosos para la instancia de corrección y defensa del proyecto académico.

## **4. Evidencia de ejecución de los planes**

### **4.1. Introducción**

A lo largo del proyecto se documentó de forma detallada en cada *sprint*, la ejecución de los planes y las variantes a la planificación inicial en caso de que existiera. Sin embargo, este capítulo tiene el objetivo de dar un cierre y presentar de manera resumida la evidencia de la implementación de los planes realizados en el anteproyecto.

### **4.2. Plan de calidad**

Se pudo cumplir con lo previsto para el plan de calidad, ya que la gestión de SQA se realizó al final de cada *sprint*, probando cada una de las funcionalidades implementadas en el mismo.

Se realizaron pruebas manuales con usuarios reales de la plataforma de ML, comprobando que el comportamiento era el esperado y que coincidía con el flujo y las acciones que se brindan desde la plataforma de ML para la gestión de reclamos. Ante eventuales fallas que surgían, las mismas eran corregidas dentro del mismo *sprint*, antes de la presentación del avance del producto al cliente. Luego, durante la muestra del producto al cliente se realizaban las mismas pruebas. En el Anexo 4 - Casos de prueba queda la información y datos utilizados para las pruebas realizadas de cada uno de los requerimientos.

También, como se había planeado, se realizaron pruebas unitarias automatizadas para cada uno de los *end points*, queda evidencia en el Anexo 3 - Evidencia de *testing* de la API. Dada las características de este proyecto es de sumo interés hacer foco en la calidad de la API y de cada uno de sus *end points*, ya que el cliente está interesado en conectar el producto que ya tiene con nuestro *backend*. Mientras que el *frontend* es utilizado principalmente con fines académicos, para poder mostrar el uso y funcionalidades básicas de la API generada.

## 4.3.Plan de configuración de software

Para el manejo de configuración de *software* (SCM) finalmente se utilizó GitHub, esta herramienta permitió tener un control de las versiones trabajadas, ya que muchas veces se trabajó en conjunto desarrollando el mismo requerimiento. Se contó con dos ramas principales: *develop* y *master*. En cada *sprint* durante la etapa de desarrollo se iba subiendo todo el trabajo a la rama *develop*. Se iba realizando el *test* necesario y una vez finalizado el *sprint* se subía todo a la rama *master*. Esto permitía saber que el trabajo que se encontraba en la rama *master* funcionaba adecuadamente ya que había pasado previamente por el proceso de *testing*.

The screenshot shows the GitHub repository page for 'florfer / GRML'. The repository is private. At the top, there are buttons for 'Add Repo', 'Watch', 'Star' (0), and 'Fork' (0). Below the header, there are tabs for 'Code', 'Issues', 'Pull requests', 'Actions', 'Projects', 'Security', and 'Insights'. The 'Code' tab is selected. It shows a dropdown for 'master' branch, '3 branches', and '0 tags'. There are buttons for 'Go to file', 'Add file', and 'Code'. A message says 'This branch is 3 commits ahead, 102 commits behind develop.' with links for 'Pull request' and 'Compare'. Below this, a list of commits is shown:

Author	Commit Message	Date	Commits
fiorfer and antoalconti	Sprint 10 (#36)	6 days ago	92 commits
ApiUnitTestProject	Sprint 10 (#36)	6 days ago	
GRML	Sprint 10 (#36)	6 days ago	
GRML_WWW	Sprint 10 (#36)	6 days ago	
.gitattributes	Agregar .gitignore y .gitattributes.	4 months ago	
.gitignore	Agregar .gitignore y .gitattributes.	4 months ago	
GRML.sln	Sprint 5 (#19)	3 months ago	

At the bottom, there's a note 'Help people interested in this repository understand your project by adding a README.' with a 'Add a README' button. On the right side, there are sections for 'About' (no description, website, or topics provided), 'Releases' (no releases published, Create a new release), 'Packages' (no packages published, Publish your first package), and 'Contributors' (2 contributors: fiorfer and antoalconti).

## 4.4.Plan de capacitación

Siguiendo lo planificado en el anteproyecto, para hacer la transferencia tecnológica al cliente, se realizó la documentación de la API y un manual de usuario.

Se utilizó la herramienta Swagger para la documentación de la API, la misma se encuentra *hosteada* junto con el proyecto de *backend*, se puede ingresar en este enlace <https://grmlapi.azurewebsites.net/swagger/index.html> para ver dicha documentación. El *apiary* se creó en el *sprint 3* y se fue actualizando a lo largo del desarrollo del proyecto cada vez que se creaba un nuevo *end point*. Para cada *end point* se detalla la funcionalidad del mismo, los campos y tipos de datos que espera recibir y las posibles respuestas de cada solicitud.

A continuación, se incluye una imagen de la documentación, para ver en detalle el contenido de la misma se puede ingresar en el enlace mencionado anteriormente o ver el Anexo 13 – Documentación de la API.

The screenshot shows the Swagger UI for the GRML\_API version 1. The URL in the browser is grmlapi.azurewebsites.net/swagger/index.html. The page title is "GRML\_API v1 OAS3". Below the title, it says "/swagger/v1/swagger.json". The interface is organized into sections: "Claim", "Dashboard", and "Exception".

- Claim:**
  - GET /Claim/GetAll** Returns a complete list of user's claims ordered by status (opened - closed), purchase price and claim opened date.
  - GET /Claim/GetByUserStore** Returns a list of user's claims filtered by store and ordered by status (opened - closed), purchase price and claim opened date.
  - GET /Claim/GetByUserStoreAndFilter** Returns a list of user's claims filtered by store and other params (status, type or waiting for response of) ordered by status (opened - closed), purchase price and claim opened date.
  - GET /Claim/GetById** Returns the claim corresponded to the id provided.
  - GET /Claim/GetAttachment** Returns the URL to download the attached file.
- Dashboard:**
  - GET /Dashboard/GetDashboard** Returns a dashboard.
  - GET /Dashboard/GetAllByDates** Returns a complete list of user's Dashboard in the specified time period.
- Exception:**

La documentación de la usabilidad de la aplicación web se realizó como estaba planificado a través de un manual de usuario donde se detallan todos los flujos y funcionalidades (ver en Anexo 10 - Manual de usuario).

## 4.5.Plan de riesgos

A continuación, se detallan los riesgos identificados antes de comenzar el proceso de desarrollo y si los mismos se materializaron o no durante la ejecución del proyecto.

NOMBRE DEL RIESGO	OCURRENCIA	MATERIALIZACION	COMENTARIOS
Disponibilidad horaria del equipo de desarrollo	Media	NO	Este riesgo no se materializó, en parte por el poco error de estimación que hubo en general.
Estimación de esfuerzos en comparación al tamaño del producto	Alta	SI	Si bien en alguno de los <i>sprints</i> hubo una pequeña desviación en los tiempos, esto no implicó un perjuicio para el proyecto, ya que en su mayoría fue positivo lo que permitió equiparar las desviaciones negativas que existieron. Por lo que no se tuvo que aplicar el plan de contingencia.
Enfermedad de alguno de los desarrolladores o indisponibilidad por fuerza mayor	Media	NO	Este riesgo no se materializó, en parte se puede deber a que ambas desarrolladoras continuaron trabajando remoto durante la pandemia, lo que hizo que se tuviera menos exposición a la misma.
Cambio en los requerimientos o existencia de nuevos	Baja	SI	Este riesgo se materializó, ya que existieron nuevos requerimientos. Se recurrió al plan de contingencia y se evaluó si era posible incluir cada uno de los nuevos requerimientos. Cabe destacar que se pudo cumplir con la implementación de los mismos. Sin embargo, para lograr esto, en algunos <i>sprints</i> se tuvo que dedicar una mayor carga horaria que la estimada en el anteproyecto para cada <i>sprint</i> .
Cambio de las reglas de negocio de ML para la gestión de reclamos	Baja	NO	No se materializó durante el desarrollo.
Curva de aprendizaje	Media	NO	No se materializó, ya que se pudo alcanzar los aprendizajes de las nuevas herramientas y tecnologías en los tiempos estipulados.

## **5. Conclusiones**

### **5.1. Introducción**

Finalizado el proyecto se puede concluir que en su mayoría se lograron cumplir los objetivos establecidos en un inicio con el cliente. Se logró cumplir completamente el objetivo principal del proyecto de gestionar los reclamos desde nuestra herramienta, permitiéndole a los usuarios realizar todas las acciones que ML ofrece en su plataforma.

El segundo objetivo se cumplió parcialmente ya que si bien en un inicio se planteó la necesidad de contemplar todas las variantes de flujo que puede tener la gestión de un reclamo en los distintos países donde opera ML, cuando se acordó con el cliente el alcance y limitaciones del proyecto, siendo este una primera versión de la herramienta se acotó a la gestión de los reclamos para usuarios de ML en Uruguay.

Los objetivos de contar con un *dashboard* y obtener reportes se cumplieron en su totalidad e incluso durante el desarrollo se incorporaron otros datos los cuales se creyó eran de valor mostrar para ampliar los valores para el análisis del manejo de la gestión de los reclamos.

Por último, el objetivo respecto a la integración con la plataforma actual de Sherlock Assistant si bien aún no se implementó por parte del cliente, desde el lado de las desarrolladoras se cumplió, ya que se implementaron los requerimientos que permitían realizar esta integración. Se mantuvo la misma estructura de datos que utiliza el cliente en su plataforma y se diseñó una arquitectura acordé, creando una API REST que permite la conexión con terceros y por último se entregó al cliente un *apiary* para facilitar la conexión a la misma.

### **5.2. Grado de satisfacción del cliente**

Se puede destacar que durante todo el proceso se mantuvo una excelente comunicación con el cliente, siempre se mostraron accesibles y con gran disposición para responder las dudas que surgieron y estuvieron involucrados en los avances que se fueron realizando. A su vez, el cliente también destacó la buena comunicación por parte de las desarrolladoras, esto nos lo hicieron saber de manera verbal en cada una de las instancias de muestra que se realizaron al finalizar los *sprints*, así como de manera escrita.

Para el equipo fue una gran satisfacción saber que no solo se cumplió con los objetivos esperados por el cliente, sino que también se mostraron muy conformes en las muestras que se realizaron de cada uno de los avances del producto. Esto no solo nos lo hicieron saber de manera verbal, sino que también a partir del *sprint* 7 luego de las muestras, les enviamos un mail para dejar por escrito la evidencia del cumplimiento de los requerimientos. Recibiendo en la respuesta de los mismos la conformidad y el grado de satisfacción de los avances que se iban mostrando. Esta evidencia se encuentra documentada en los anexos (Anexo 5 - Validación del cliente sobre avance del producto (*sprint* 7), Anexo 6 - Validación del cliente sobre avance del producto (*sprint* 8), Anexo

7 - Validación del cliente sobre avance del producto (*sprint 9*) y Anexo 8 - Validación del cliente sobre avance del producto (*sprint 10*)).

También, se realizó una evaluación final con el cliente a modo de cierre, para conocer de manera global el grado de aprobación del mismo. Se recibió una muy buena devolución respecto al desempeño en diferentes áreas, ya sea en la gestión del proyecto, desarrollo del mismo, cumplimiento de requerimientos, comunicación y adaptación a los cambios que fueron surgiendo. En el Anexo 9 - Evaluación de satisfacción del cliente, se puede ver la respuesta que se tuvo.

### **5.3. Lecciones aprendidas**

Se puede concluir que durante el desarrollo del proyecto se han logrado innumerables instancias de aprendizaje. Se obtuvo un mayor conocimiento tanto a nivel tecnológico como en la gestión de proyectos, en particularidades del negocio y en la integración con una API externa.

Para ambas desarrolladoras fue la primera experiencia en la gestión de un proyecto, abarcando todos los roles que se necesitan para llevar a cabo el mismo. Si bien durante la carrera Analista Programador como equipo se había enfrentado al desarrollo de soluciones tecnológicas a problemas ficticios presentados en obligatorios, lo cual nos dio herramientas tecnológicas, estas instancias nunca fueron con clientes y necesidades reales. Por lo que al enfrentar esta instancia nos permitió por primera vez hacer el análisis para brindar una solución a esas necesidades, evaluando el esfuerzo de la misma, adecuándola a los tiempos y alcance del Proyecto Integrador. Lo que nos permitió lograr aprendizajes en cuanto a la realización de estimaciones con la complejidad de analizar el esfuerzo que implica integrarse con una herramienta externa nueva, ajena a nuestro conocimiento, como lo es la API de ML y la necesidad de realizar ajustes a la planificación para contemplar cambios en los requerimientos. Así mismo, a lo largo del proyecto el equipo fue ganando un mayor conocimiento de las tecnologías involucradas y de la propia productividad, lo que permitió lograr una mayor exactitud a la hora de estimar los nuevos requerimientos que fueron surgiendo. Por todo lo expuesto sentimos que hemos ganado un valioso conocimiento referente a la evolución de nuestra productividad lo que redunda en una mayor confianza en nuestra capacidad de estimación y gestión de los futuros proyectos que tomemos a nuestro cargo.

Por otro lado, el utilizar una metodología ágil para la gestión del proyecto fue de gran valor, reforzando el aprendizaje de la importancia de evaluar e implementar la metodología adecuada para cada proyecto, de manera de poder realizarlo de forma organizada como se logró en este proyecto.

También se destaca los aprendizajes adquiridos en el manejo de las diferentes herramientas tecnológicas utilizadas, brindándonos mayor experiencia en el lenguaje .NET lo que es una gran herramienta para nuestra carrera profesional. A su vez se logró realizar con éxito el *deploy* y *releases* de la aplicación brindándonos mayor experiencia en esta área.

Por último, se logró el aprendizaje de la importancia de definir desde un inicio quién se responsabiliza de los costos económicos que implica el desarrollo del proyecto, así como evaluar correctamente los mismos. Dado que ML no tiene un ambiente de prueba, durante este proyecto cada uno de los reclamos que se creó para probar las funcionalidades fue a partir de la compra de un artículo en la plataforma de ML lo que implicó que las desarrolladoras tuvieran que desembolsar y hacerse cargo de los costos. En un principio cuando nos mostraron los flujos de ventas y reclamos de ML nos aclararon que para probar teníamos que realizar compras y las mismas debían correr por nuestra cuenta. Hoy nos queda el aprendizaje de la importancia de hablar sobre esto en etapas iniciales, para que no recaiga en el equipo de desarrollo la responsabilidad de cubrir estos costos.

Todos estos conocimientos que se obtuvieron son de suma importancia y aportan en nuestra formación académica y profesional, permitiéndonos ganar mayor experiencia. Incluso de los inconvenientes que surgieron destacamos que de los mismos se obtuvo muchos aprendizajes.

#### **5.4. Posibles mejoras y desarrollos futuros**

Finalizado el proyecto se considera que existen determinadas mejoras que se pueden realizar en el futuro, estas mejoras son principalmente en el proyecto de *frontend*, cabe destacar como se dijo anteriormente, que el principal foco de este proyecto está puesto en el *backend*. Ya que el cliente tiene principal interés en conectarse con el *backend* desarrollado en el proyecto e integrarlo a su herramienta. Por lo que en el proyecto también se priorizó esto. A continuación, se detallan las posibles mejoras que se pueden realizar en un futuro.

- Cuando se realizó la implementación del requerimiento (R-39) Persistencia del log de excepciones en el *sprint* 10. Se creó un *end point* para obtener un listado de las mismas filtrado por un rango de fechas, pero no se implementó en el *frontend* ninguna vista para poder ver esta información. Sería una buena mejora a futuro implementar la vista para mostrar los resultados de la consulta de este servicio, donde se muestre en un listado el log de cada una de esas excepciones. Así mismo, también una mejora de este requerimiento podría ser crear un *job* que cada cierto tiempo elimine todas las excepciones persistidas hasta cierta fecha.
- Otra posible mejora a implementar en una siguiente versión podría ser tener la posibilidad de exportar en archivos csv los datos desplegados en el reporte de los reclamos. Si bien éste era un requerimiento contemplado desde un inicio, cuando se evaluó el alcance y limitaciones del proyecto, se decidió en conjunto con el cliente no incluirlo dentro de los requerimientos a desarrollar, esto se encuentra documentado en el anteproyecto en el capítulo 2.8 Alcance y limitaciones.
- También se cree que podría ser valioso para el usuario la posibilidad de poder ver su perfil de usuario con la información del mismo y permitirle cambiar y recuperar su contraseña. Pero debido a que el cliente no tenía interés en el *frontend* y el mismo fue desarrollado principalmente con fines académicos para poder mostrar el trabajo desarrollado en el *backend*, no tenía valor para el cliente implementar esta funcionalidad en esta instancia.

- Otra mejora que se podría realizar y sería de gran valor para el usuario, es el obtener y mostrar su reputación como vendedor en ML, así el usuario podría ver como ésta cambia si mejora la gestión de las ventas. A partir de la información que puede recaudar de los reclamos en nuestra herramienta y conociendo los motivos por los que le inician los reclamos podría tomar acciones a futuro como vendedor que le permitirían disminuir la cantidad de reclamos que le inician y así ayudaría a mantener o mejorar su reputación como vendedor. Esto no se implementó porque en un principio no fue de interés para el cliente.
- Actualmente ML no permite realizar la devolución del dinero de la compra a través de su API, por lo que esta funcionalidad se resolvió con un link al reclamo en la plataforma de ML. En un futuro creemos que ML va a incorporar esta funcionalidad y ahí sería de gran valor integrarlo a nuestra aplicación.

## **5.5. Reflexiones finales**

Se puede concluir que culminado el proyecto se lograron cumplir en tiempo los requerimientos planteados, logrando la satisfacción y validación por parte de los clientes. Sentimos que durante todo el proceso fuimos creciendo e incorporando nuevos conocimientos y percepciones tanto en lo que refiere a la gestión del proyecto, como en el manejo de las tecnologías, trabajo en equipo y en la comunicación con los clientes.

Finalmente creemos que esta instancia de formación académica fue de gran valor y aprendizaje, enriqueciendo nuestra formación como estudiantes y profesionales, dejándonos herramientas que sin dudas aplicaremos en el futuro.

## 6. Glosario

- Integradores: herramienta de *software* que ofrecen capacidad de trabajar sobre ML
- API (Interfaz de Programación de Aplicaciones): “funciones y procedimientos que cumplen una o muchas funciones con el fin de ser utilizadas por otro *software*” [2].
- MVC (Modelo-Vista-Controlador): “es un patrón en el diseño de *software* comúnmente utilizado para implementar interfaces de usuario, datos y lógica de control. Enfatiza una separación entre la lógica de negocios y su visualización” [3].
- Azure: conjunto de servicios en la nube para crear, administrar e implementar aplicaciones en una red global [4].
- C#.NET: “lenguaje de programación desarrollado por Microsoft que se ejecuta en .NET Framework” [5].
- .NET Core: “versión multiplataforma de .NET para crear sitios web, servicios y aplicaciones de consola” [6].
- SQL Server: plataforma para la gestión de datos.
- Aplicación *responsive*: “consiste en crear páginas web que se vean bien en todos los dispositivos” [7].
- OAUTH2: protocolo de autorización estándar de la industria. proporciona flujos de autorización específicos para aplicaciones web, aplicaciones de escritorio, teléfonos móviles y dispositivos [8].
- *Front-End*: parte de un sitio web que interactúa con los usuarios, por eso se dice que está del lado del cliente [9].
- *Back-End*: aplicación que se conecta con la base de datos y el servidor que utiliza dicho sitio web [9].
- *Software as a service* (Software como servicio (SaaS)): “permite a los usuarios conectarse a aplicaciones basadas en la nube a través de Internet y usarlas” [10].
- IDE (Entorno de desarrollo integrado): “conjunto de procedimientos y herramientas que se utilizan para desarrollar un código fuente o programa” [11].
- Visual Studio: entorno de desarrollo integrado para crear aplicaciones. Compatible con múltiples lenguajes de programación [12].
- SQL Management Studio: “entorno integrado para administrar cualquier infraestructura SQL” [13].
- Trello: *software* para la administración de proyectos, permite que los equipos trabajen de forma colaborativa y sean más productivos [14].
- Kanban: metodología para gestionar el trabajo.
- Microsoft Teams: centro para el trabajo en equipo, reúne todo lo que un equipo necesita: conversaciones de chat y subprocesos, reuniones y videoconferencias, llamadas y colaboración de contenido [15].
- SCM: gestión de configuración de *software*.
- SQA: aseguramiento de calidad de *software*.
- Git: “sistema de control de versiones distribuido gratuito y de código abierto” [16].

- *Swagger*: herramienta que brinda una gama de soluciones para generar, visualizar y mantener documentos de API, eliminando el trabajo manual de la documentación de API [17].
- *Apiary*: permite documentar las APIs fácilmente [18].
- *End points* (punto final): “permite que dos programas de *software* se comuniquen entre sí” [19].
- *Kick off*: primera reunión, es una oportunidad para establecer objetivos y propósitos comunes al completar el trabajo [20].
- *Sprint*: “período breve de tiempo fijo en el que un equipo de scrum trabaja para completar una cantidad de trabajo establecida” [21].
- *Deploy*: “todos los pasos, procesos y actividades necesarios para que un sistema de software o una actualización estén disponibles para los usuarios” [22].
- Proyecto XUnitTest: herramienta para realizar pruebas unitarias gratuita y de código abierto para .NET Framework [23].
- Swashbuckle: paquete Nuget para habilitar Swagger [24].
- Postman: herramienta que principalmente permite crear peticiones sobre APIs de una forma muy sencilla y poder, de esta manera, probar las APIs [25].
- *Release*: versión del software implementado en la culminación de una iteración [26].

## 7. Referencias

- [1] Mercado Libre S.R.L., “Trabajar con reclamos”. [Online]. Available: [https://developers.mercadolibre.com.ar/es\\_ar/trabajar-con-reclamos](https://developers.mercadolibre.com.ar/es_ar/trabajar-con-reclamos)
- [2] Hipertextual, “¿Qué es una API?”, mayo 2014. [Online]. Available: <https://hipertextual.com/archivo/2014/05/que-es-api/#:~:text=Una%20API%20es%20un%20conjunto,Interfaz%20de%20Programaci%C3%83n%20de%20Aplicaciones.&text=En%20t%C3%A9rminos%20de%20programaci%C3%83n%20es%20una%20capa%20de%20abstracci%C3%83n>
- [3] Mozilla and individual contributors, “MVC”, enero 2020. [Online]. Available: <https://developer.mozilla.org/es/docs/Glossary/MVC>
- [4] Microsoft, “¿Qué es Azure?”, 2020. [Online]. Available: <https://azure.microsoft.com/es-es/overview/what-is-azure/>
- [5] W3Schools, “C# Tutorial”. [Online]. Available: <https://www.w3schools.com/cs/>
- [6] .NET, “Descarga .NET”. [Online]. Available: <https://dotnet.microsoft.com/download>
- [7] W3Schools, “Diseño Web Responsive HTML”. [Online]. Available: [https://www.w3schools.com/html/html\\_responsive.asp](https://www.w3schools.com/html/html_responsive.asp)
- [8] OAuth2, “OAuth 2.0”. [Online]. Available: <https://oauth.net/2/>
- [9] Platzi, “Qué es Frontend y Backend”. [Online]. Available: <https://platzi.com/blog/que-es-frontend-y-backend/>
- [10] Microsoft, “¿Qué es SaaS?”. [Online]. Available: <https://azure.microsoft.com/es-es/overview/what-is-saas/>
- [11] Arimetrics, “Entorno de desarrollo”. [Online]. Available: <https://www.arimetrics.com/glosario-digital/entorno-de-desarrollo>
- [12] Espacio Honduras, “Concepto de Microsoft Visual Studio, ¿Qué es y para qué sirve Microsoft Visual Studio?”, agosto 2020. [Online]. Available: <https://www.espaciohonduras.net/microsoft-visual-studio-concepto-y-que-es-y-para-que-sirve-microsoft-visual-studio>
- [13] Microsoft, “Download SQL Server Management Studio (SSMS)”, julio 2020. [Online]. Available: <https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15>
- [14] Atlassian, “Trello”. [Online]. Available: [https://trello.com/?&aceid=&adposition=&adgroup=105703213888&campaign=9843285526&creative=437184392305&device=c&keyword=trello&matchtype=e&network=g&placement=&ds\\_kids=p53016482445&ds\\_e=GOOGLE&ds\\_eid=700000001557344&](https://trello.com/?&aceid=&adposition=&adgroup=105703213888&campaign=9843285526&creative=437184392305&device=c&keyword=trello&matchtype=e&network=g&placement=&ds_kids=p53016482445&ds_e=GOOGLE&ds_eid=700000001557344&)

ds\_e1=GOOGLE&gclid=EAIaIQobChMI\_df53uXx6wIViIiRCh0oww17EAA Yasaa EgJzm\_D\_BwE&gclsrc=aw.ds

[15] Google, “Microsoft Teams”. [Online]. Available: [https://play.google.com/store/apps/details?id=com.microsoft.teams&hl=es\\_UY](https://play.google.com/store/apps/details?id=com.microsoft.teams&hl=es_UY)

[16] Git, “git”. [Online]. Available: <https://git-scm.com/>

[17] SmartBear Software, “Documentación de API”. [Online]. Available: <https://swagger.io/solutions/api-documentation/>

[18] Adictos al trabajo, “Documenta tu API con Apiary”, agosto 2017. [Online]. Available: <https://www.adictosaltrabajo.com/2017/08/17/documenta-tu-api-con-apiary/>

[19] TechTarget, “API endpoint”. [Online]. Available: <https://searchapparchitecture.techtarget.com/definition/API-endpoint>

[20] The Digital Project Manager, “Kickoff Meeting: The Complete Guide To Starting Projects Right”, diciembre 2016. [Online]. Available: <https://thedigitalprojectmanager.com/project-kickoff-meeting/>

[21] Atlassian, “¿Qué son los sprints?”. [Online]. Available: Scrum: <https://www.atlassian.com/es/agile/scrum/sprints#:~:text=son%20los%20sprints%3F-.Un%20sprint%20es%20un%20per%C3%A1odo%20breve%20de%20tiempo%20fijo%20en,con%20menos%20quebraderos%20de%20cabeza.>

[22] Sumo Logic, “What is Software Deployment”. [Online]. Available: <https://www.sumologic.com/glossary/software-deployment/>

[23] Wikimedia Foundation, “xUnit.net”, junio 2020. [Online]. Available: <https://en.wikipedia.org/wiki/XUnit.net>

[24] SomosTechies, “Agregando Swagger a ASP.NET Web Api 2”, setiembre 2016. [Online]. Available: <https://somostechies.com/agregando-swagger-a-asp-net-web-api-2/>

[25] Arquitecto IT, “¿Qué es Postman?”, febrero 2019. [Online]. Available: <http://www.arquitectoit.com/postman/que-es-postman/#:~:text=Postman%20nace%20como%20una%20herramienta,una%20extensi%C3%B3n%20de%20Google%20Chrome.>

[26] TechTarget, “release”. [Online]. Available: <https://searchsoftwarequality.techtarget.com/definition/release>

## **8. Bibliografía**

Mercado Libre S.R.L., “Trabajar con reclamos”. [Online]. Available: [https://developers.mercadolibre.com.ar/es\\_ar/trabajar-con-reclamos](https://developers.mercadolibre.com.ar/es_ar/trabajar-con-reclamos)

Mercado Libre S.R.L., “Registra tu aplicación”. [Online]. Available: [https://developers.mercadolibre.com.ar/es\\_ar/registra-tu-aplicacion](https://developers.mercadolibre.com.ar/es_ar/registra-tu-aplicacion)

Mercado Libre S.R.L., “Conoce los últimos recursos disponibles”. [Online]. Available: [https://developers.mercadolibre.com.ar/es\\_ar/api-docs-es](https://developers.mercadolibre.com.ar/es_ar/api-docs-es)

Mercado Libre S.R.L., “Usuarios y Aplicaciones”. [Online]. Available: [https://developers.mercadolibre.com.uy/es\\_ar/usuarios-y-aplicaciones#close](https://developers.mercadolibre.com.uy/es_ar/usuarios-y-aplicaciones#close)

Microsoft, “Get started with Swashbuckle and ASP.NET Core”, junio 2020. [Online]. Available: <https://docs.microsoft.com/en-us/aspnet/core/tutorials/getting-started-with-swashbuckle?view=aspnetcore-3.1&tabs=visual-studio>

## 9. Anexos

### Anexo 1 - Definición objetivos con el cliente

Hugo Olivera para Antonieta, Ricardo, mí ▾

22 abr. 2020 19:08 (hace 10 días) ☆ ↶ ⋮

Hola como andan?

Objetivos

1. Disponibilizar como mínimo las mismas capacidades de gestión de reclamos que se pueden hacer por medio del portal de ML.

2. Crear un tablero de Indicadores de la gestión de reclamos (no existe en ML)

3. Crear reportes de gestión de reclamos (no existe en ML)

4. Integración con la plataforma actual, es un **nice to have** pero no es requisito obligatorio.

Saludos,

Hugo Olivera Alonso  
Consultor IA  
  
[www.sherlock-assistant.com](http://www.sherlock-assistant.com) /Cureim 1147- Montevideo - Uruguay / (598) 092 200 905

### Anexo 2 - Recursos para ver el diagrama de Gantt

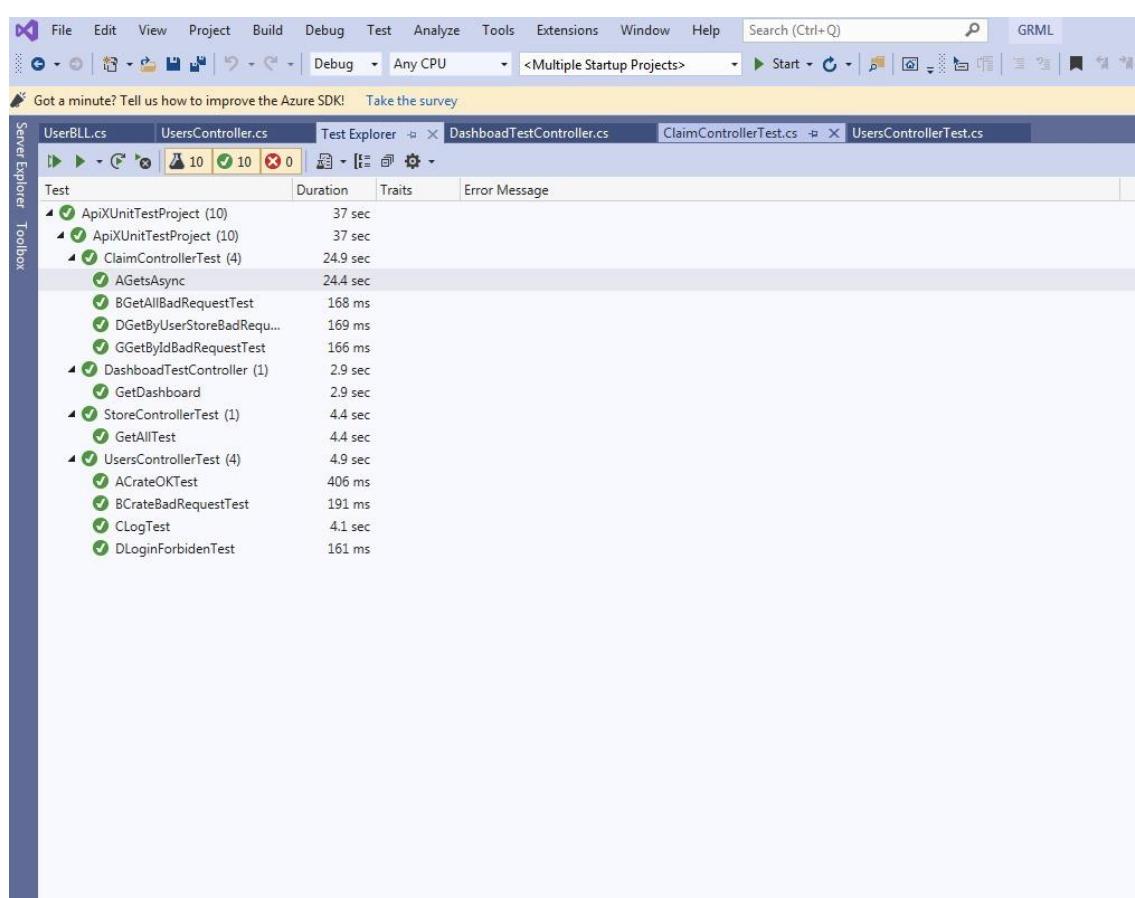
El diagrama de Gantt se puede ver de forma más nítida en el archivo Diagrama de Gantt.pdf el cual se encuentra dentro de la carpeta de entrega de este proyecto o en el siguiente link:

<https://drive.google.com/file/d/10xK8c38M76sRSu0-DO1wshEfos-oW8tR/view>

## Anexo 3 – Evidencia de testing de la API

En el *backend*, se realizaron pruebas unitarias automatizadas, de cada uno de los *end points* que brindan un servicio que ejecuta la lógica de nuestra API. Mientras que no se realizó a los que directamente hacen una *request* a la API de ML. Los *test* se codificaron en un *ApiXUnitTestProject* dentro de la solución. En cada uno de los *test* se creó el contexto de la prueba, luego se ejecuta la llamada al *end point* correspondiente y se obtiene la respuesta esperada para cada uno de los escenarios de contexto creados para el mismo.

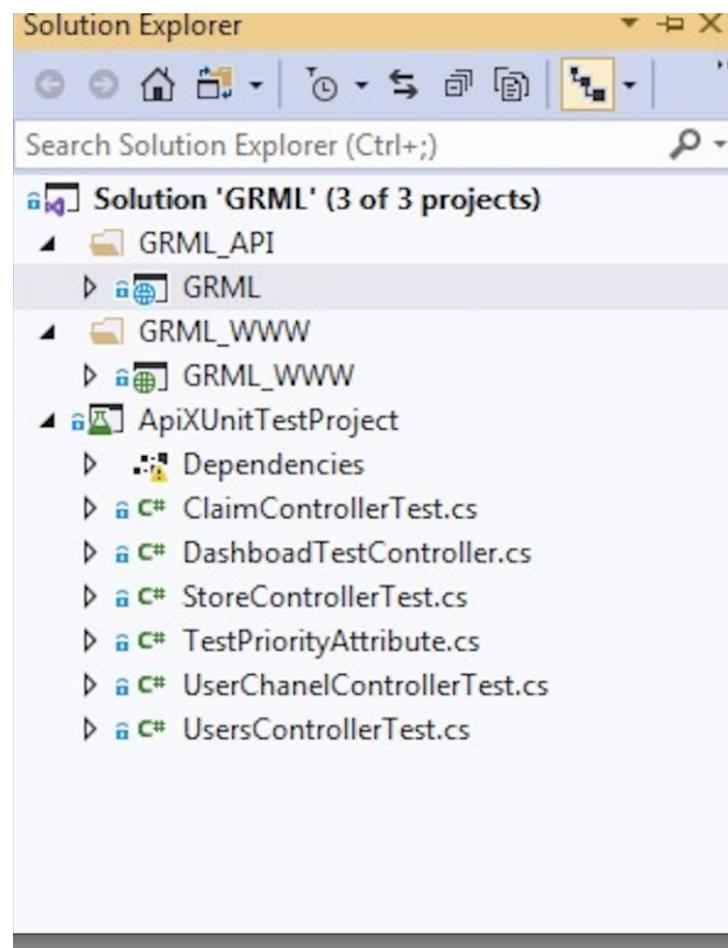
Para cada contexto de prueba se generan datos, se persisten y se destruyen los mismos luego de pasar el test, no dejando ningún tipo de dato “*basura*” en la base de datos.



A medida que se iba avanzando en el desarrollando del proyecto, creándose nuevos *end points* en los controladores, se fueron creando los *test*. En cada *test* se fue contemplando los distintos casos de respuesta que podía tener el mismo, para comprobar que el comportamiento fuera el esperado. A continuación, se detalla en qué *sprint* se desarrolló cada uno de los *test*.

- *ClaimControllerTest*:
  - AGetAsync - *Sprint 9*
  - B GetAllBadRequest - *Sprint 9*
  - D GetUserStoreRequest - *Sprint 9*
- *DashboardTestController*:
  - GetDashboard - *Sprint 10*
- *StoreTestController*:
  - GetAllTest - *Sprint 5*
- *UserControllerTest*:
  - ACreateTest - *Sprint 3*
  - BCreateBadRequestTest - *Sprint 3*
  - CLogTest - *Sprint 4*
  - DLoginForbidenTest - *Sprint 4*

A continuación, se detalla la arquitectura donde se creó el proyecto de *testing*:



## Anexo 4 – Casos de prueba

### Casos de prueba para requerimientos asociados al usuario

#### R-01: *Login*

Precondiciones: El formulario de inicio de sesión está visible ya que no hay ningún usuario autenticado en el sitio. (URL: <https://grmlwww.azurewebsites.net/Users/Login>)

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R01-T01	El usuario ingresa correctamente al sitio.	El usuario: 1. Ingresa su nombre de usuario correctamente 2. Ingresa su contraseña correctamente 3. Presiona el botón “Ingresar”.	Nombre de usuario: usuarioTesting Contraseña: 123123	El usuario ingresa al sitio y visualiza la página principal. Si tiene un usuario de ML asociado y tiene reclamos visualiza el listado de todos los reclamos. Si tiene un usuario de ML asociado y no tiene reclamos visualiza el mensaje “No tiene reclamos”. Si no tiene un usuario de ML asociado visualiza el mensaje “Primero debe asociar su cuenta de Mercado Libre”.	P
R01-T02	El usuario intenta ingresar al sitio con su nombre de usuario correcto y contraseña incorrecta.	El usuario: 1. Ingresa su nombre de usuario correctamente 2. Ingresa su contraseña incorrectamente 3. Presiona el botón “Ingresar”.	Nombre de usuario: usuarioTesting Contraseña: 123456	El usuario sigue visualizando la pantalla de <i>login</i> , con el mensaje “Nombre de usuario o contraseña incorrectos”. Se le permite el reingreso de los datos. El nombre de usuario ingresado anteriormente no se borra, mientras que la contraseña sí.	P

<b>ID</b>	<b>Escenario de test</b>	<b>Pasos</b>	<b>Datos utilizados</b>	<b>Resultado esperado</b>	<b>Estado (F=Falla, P=pasa)</b>
R01-T03	El usuario intenta ingresar al sitio con su nombre de usuario incorrecto y contraseña correcta.	El usuario: 1. Ingresa su nombre de usuario incorrectamente 2. Ingresa su contraseña correctamente 3. Presiona el botón “Ingresar”.	Nombre de usuario: usuarioIncorrecto Contraseña: 123123	Ídem R01-T02.	P
R01-T04	El usuario intenta ingresar al sitio con su nombre de usuario incorrecto y contraseña incorrecta.	El usuario: 1. Ingresa su nombre de usuario incorrectamente 2. Ingresa su contraseña incorrectamente 3. Presiona el botón “Ingresar”.	Nombre de usuario: usuarioIncorrecto Contraseña: 123456	Ídem R01-T02.	P

## R-02: Autenticación del usuario en ML

Precondiciones: El usuario inició sesión con su usuario y se encuentra en la pantalla de asociar cuenta de ML. (URL: <https://grmlwww.azurewebsites.net/ChannelUsers/AddAccount>)

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R02-T01	El usuario asocia correctamente su usuario con su cuenta de ML.	<p>El usuario:</p> <ol style="list-style-type: none"> <li>Presiona el botón “Asociar cuenta de ML”</li> <li>Es redireccionado al sitio de autenticación de ML</li> <li>Ingresa su usuario y contraseña de ML (en caso de no estar <i>logueado</i> previamente con su cuenta de ML)</li> <li>Presiona el botón “Permitir” para autorizar la conexión entre sus cuentas.</li> </ol>		El usuario es redireccionado a nuestro sitio, a la pantalla de asociación de cuenta de ML y visualiza el mensaje “Su cuenta de Mercado Libre se asoció correctamente”.	P
R02-T02	El usuario cancela la asociación de usuario con su cuenta de ML.	<p>El usuario:</p> <ol style="list-style-type: none"> <li>Presiona el botón “Asociar cuenta de ML”</li> <li>Es redireccionado al sitio de autenticación de ML</li> <li>Ingresa su usuario y contraseña de ML (en caso de no estar <i>logueado</i> previamente con su cuenta de ML)</li> <li>Presiona el botón “Por ahora no”</li> </ol>		El usuario es redireccionado a nuestro sitio, a la pantalla de asociación de cuenta de ML y visualiza el mensaje “No se pudo asociar correctamente su cuenta de Mercado Libre”. Se le permite volver a intentar asociar su cuenta de ML.	P

		para denegar la conexión entre sus cuentas.			
R02-T03	El usuario ya tiene una cuenta de ML asociada e intenta nuevamente asociar su usuario con una cuenta de ML.	<p>El usuario:</p> <ol style="list-style-type: none"> <li>1. Presiona el botón “Asociar cuenta de ML”</li> <li>2. Es redireccionado al sitio de autenticación de ML</li> <li>3. Ingresa su usuario y contraseña de ML (en caso de no estar logueado previamente con su cuenta de ML)</li> <li>4. Presiona el botón “Permitir” para autorizar la conexión entre sus cuentas.</li> </ol>		El usuario es redireccionado a nuestro sitio, a la pantalla de asociación de cuenta de ML y visualiza el mensaje “Ya tiene un usuario de ML asociado a tu cuenta con el nombre de UYUPDATE”	P

#### R-06: Validar reclamo

Precondiciones: El usuario inició sesión con su usuario y se encuentra en la pantalla principal de la aplicación, luego de haber asociado su cuenta de ML. (URL: <https://grmlwww.azurewebsites.net/Home/Index>)

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R06-T01	El usuario no vera reclamos, que no pertenezcan a una orden que no esté	- Desde la herramienta <i>Postman</i> , enviamos notificaciones, con la misma estructura que la que nos envía ML y al mismo <i>end point</i> que	Hicimos la llamada a la URL: <a href="https://grmlapi.azurewebsites.net/Notification/Create">https://grmlapi.azurewebsites.net/Notification/Create</a> , con el siguiente json: <pre>{"resource":"/v1/claims/5037292111", "user_id":"645077240", "topic":"claims",}</pre>	El usuario no ve ningún reclamo nuevo cuando actualiza la página.	P

	asociada a su cuenta de ML.	las recibimos, con números de reclamos que no le pertenecían al usuario.	<pre>"application_id":5503910054141466,   "attempts":1,   "sent":"2019-10-29T17:46:24.606Z",   "received":"2019-10-29T17:46:24.616"}</pre>		
R06-T02	El usuario no vera reclamos, que pertenecen a una orden que no sobre su cuenta de ML.	- Desde la herramienta <i>Postman</i> , enviamos notificaciones, con la misma estructura que la que nos envía ML y al mismo <i>end point</i> que las recibimos, con números de reclamos que si le pertenecían al usuario de prueba.	<p>Hicimos la llamada a la URL: <a href="https://grmlapi.azurewebsites.net/Notification/Create">https://grmlapi.azurewebsites.net/Notification/Create</a>, con el siguiente <i>json</i>:</p> <pre>{"resource":"/v1/claims/5037292716", "user_id":645077240", "topic":"claims", "application_id":5503910054141466, "attempts":1, "sent":"2019-10-29T17:46:24.606Z", "received":"2019-10-29T17:46:24.616"}</pre>	El usuario ve un reclamo nuevo, cuando actualiza la página.	P

R-04: Registro de nuevos usuarios

Precondiciones: El formulario de registro está visible. (URL: <https://grmlwww.azurewebsites.net/Users/Registration>)

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Fallá, P=pasa)
R04-T01	El usuario se registra correctamente en la aplicación.	<p>El usuario:</p> <ol style="list-style-type: none"> <li>1. Selecciona su país</li> <li>2. Escribe su nombre con un largo entre 3 y 20 caracteres</li> <li>3. Escribe su apellido con un largo entre 3 y 20 caracteres</li> <li>4. Ingresa su email con formato válido</li> <li>5. Escribe su nombre de usuario con un largo entre 3 y 20 caracteres</li> <li>6. Ingresa su contraseña con un largo mínimo de 6 caracteres</li> <li>7. Confirma su contraseña ingresando el mismo valor que ingresó en el paso 6</li> <li>8. Presiona el botón “Registrar”.</li> </ol>	País: Uruguay Nombre: Usuario Apellido: Testing Email: usutest@gmail.com Nombre de usuario: usuarioTesting Contraseña: 123123 Confirmar Contraseña: 123123	<p>El usuario queda registrado en la aplicación, sigue visualizando la pantalla de registro con el mensaje “Quedaste registrado, comienza a utilizar la aplicación ingresando en el link que se encuentra al final del formulario”.</p>	P
R04-T02	El usuario intenta registrarse ingresando el nombre en un formato no válido.	<p>El usuario:</p> <ol style="list-style-type: none"> <li>1. Selecciona su país</li> <li>2. Escribe su nombre con un largo mayor a 20 caracteres</li> <li>3. Escribe su apellido con un largo entre 3 y 20 caracteres</li> <li>4. Ingresa su email con formato válido</li> <li>5. Escribe su nombre de usuario con un largo entre 3 y 20 caracteres</li> </ol>	País: Uruguay Nombre: nombreMuuuuuyyyyy Largo Apellido: Testing Email: usutest@gmail.com Nombre de usuario: usuarioTesting	<p>El usuario no queda registrado en la aplicación, sigue visualizando la pantalla de registro y debajo del campo para el ingreso del nombre se muestra el mensaje “El nombre debe tener entre 3 y 20 caracteres”. Se le permite el</p>	P

		<ol style="list-style-type: none"> <li>6. Ingresa su contraseña con un largo mínimo de 6 caracteres</li> <li>7. Confirma su contraseña ingresando el mismo valor que ingresó en el paso 6</li> <li>8. Presiona el botón “Registrar”.</li> </ol>	Contraseña: 123123 Confirmar Contraseña: 123123	reingreso de los datos.	
R04-T03	El usuario intenta registrarse ingresando el apellido en un formato no válido.	<p>El usuario:</p> <ol style="list-style-type: none"> <li>1. Selecciona su país</li> <li>2. Escribe su nombre con un largo entre 3 y 20 caracteres</li> <li>3. Escribe su apellido con un largo menor a 3 caracteres</li> <li>4. Ingresa su email con formato válido</li> <li>5. Escribe su nombre de usuario con un largo entre 3 y 20 caracteres</li> <li>6. Ingresa su contraseña con un largo mínimo de 6 caracteres</li> <li>7. Confirma su contraseña ingresando el mismo valor que ingresó en el paso 6</li> <li>8. Presiona el botón “Registrar”.</li> </ol>	País: Uruguay Nombre: Usuario Apellido: Te Email: usutest@gmail.com Nombre de usuario: usuarioTesting Contraseña: 123123 Confirmar Contraseña: 123123	El usuario no queda registrado en la aplicación, sigue visualizando la pantalla de registro y debajo del campo para el ingreso del apellido se muestra el mensaje “El apellido debe tener entre 3 y 20 caracteres”. Se le permite el reingreso de los datos.	P
R04-T04	El usuario intenta registrarse ingresando el email en un formato no válido.	<p>El usuario:</p> <ol style="list-style-type: none"> <li>1. Selecciona su país</li> <li>2. Escribe su nombre con un largo entre 3 y 20 caracteres</li> <li>3. Escribe su apellido con un largo entre 3 y 20 caracteres</li> <li>4. Ingresa su email con formato inválido</li> <li>5. Escribe su nombre de usuario con un largo entre 3 y 20 caracteres</li> <li>6. Ingresa su contraseña con un largo mínimo de 6 caracteres</li> </ol>	País: Uruguay Nombre: Usuario Apellido: Testing Email: usutest@gmail.com Nombre de usuario: usuarioTesting Contraseña: 123123 Confirmar Contraseña: 123123	El usuario no queda registrado en la aplicación, sigue visualizando la pantalla de registro y debajo del campo para el ingreso del email se muestra el mensaje “Debe ingresar un correo electrónico válido”. Se le permite el reingreso de los datos.	P

		<p>7. Confirma su contraseña ingresando el mismo valor que ingresó en el paso 6</p> <p>8. Presiona el botón “Registrar”.</p>			
R04-T05	El usuario intenta registrarse ingresando el nombre de usuario en un formato no válido.	<p>El usuario:</p> <ol style="list-style-type: none"> <li>1. Selecciona su país</li> <li>2. Escribe su nombre con un largo entre 3 y 20 caracteres</li> <li>3. Escribe su apellido con un largo entre 3 y 20 caracteres</li> <li>4. Ingresa su email con formato válido</li> <li>5. Escribe su nombre de usuario con un largo menor a 3 caracteres</li> <li>6. Ingresa su contraseña con un largo mínimo de 6 caracteres</li> <li>7. Confirma su contraseña ingresando el mismo valor que ingresó en el paso 6</li> <li>8. Presiona el botón “Registrar”.</li> </ol>	País: Uruguay Nombre: Usuario Apellido: Testing Email: usutest@gmail.com Nombre de usuario: us Contraseña: 123123 Confirmar Contraseña: 123123	<p>El usuario no queda registrado en la aplicación, sigue visualizando la pantalla de registro y debajo del campo para el ingreso del nombre de usuario se muestra el mensaje “El nombre de usuario debe tener entre 3 y 20 caracteres”. Se le permite el reingreso de los datos.</p>	P
R04-T06	El usuario intenta registrarse ingresando la contraseña en un formato no válido.	<p>El usuario:</p> <ol style="list-style-type: none"> <li>1. Selecciona su país</li> <li>2. Escribe su nombre con un largo entre 3 y 20 caracteres</li> <li>3. Escribe su apellido con un largo entre 3 y 20 caracteres</li> <li>4. Ingresa su email con formato válido</li> <li>5. Escribe su nombre de usuario con un largo entre 3 y 20 caracteres</li> <li>6. Ingresa su contraseña con un largo menor de 6 caracteres</li> <li>7. Confirma su contraseña ingresando el mismo valor que ingresó en el paso 6</li> </ol>	País: Uruguay Nombre: Usuario Apellido: Testing Email: usutest@gmail.com Nombre de usuario: usuarioTesting Contraseña: 123 Confirmar Contraseña: 123123	<p>El usuario no queda registrado en la aplicación, sigue visualizando la pantalla de registro y debajo del campo para el ingreso de la contraseña se muestra el mensaje “La contraseña debe tener como mínimo 6 caracteres”. Se le permite el reingreso de los datos.</p>	P

		8. Presiona el botón “Registrar”.			
R04-T07	El usuario intenta registrarse ingresando la confirmación de contraseña diferente a la contraseña.	<p>El usuario:</p> <ol style="list-style-type: none"> <li>1. Selecciona su país</li> <li>2. Escribe su nombre con un largo entre 3 y 20 caracteres</li> <li>3. Escribe su apellido con un largo entre 3 y 20 caracteres</li> <li>4. Ingresa su email con formato válido</li> <li>5. Escribe su nombre de usuario con un largo entre 3 y 20 caracteres</li> <li>6. Ingresa su contraseña con un largo mínimo de 6 caracteres</li> <li>7. Confirma su contraseña ingresando un valor diferente que el ingresado en el paso 6</li> <li>8. Presiona el botón “Registrar”.</li> </ol>	País: Uruguay Nombre: Usuario Apellido: Testing Email: usutest@gmail.com Nombre de usuario: usuarioTesting Contraseña: 123123 Confirmar Contraseña: 321321	<p>El usuario no queda registrado en la aplicación, sigue visualizando la pantalla de registro y debajo del campo para el ingreso de la confirmación de contraseña se muestra el mensaje “Los datos contraseña y confirmar contraseña no coinciden”. Se le permite el reingreso de los datos.</p>	P
R04-T08	El usuario intenta registrarse ingresando un nombre de usuario que ya existe en el sistema.	<p>El usuario:</p> <ol style="list-style-type: none"> <li>1. Selecciona su país</li> <li>2. Escribe su nombre con un largo entre 3 y 20 caracteres</li> <li>3. Escribe su apellido con un largo entre 3 y 20 caracteres</li> <li>4. Ingresa su email con formato válido</li> <li>5. Escribe su nombre de usuario con un largo entre 3 y 20 caracteres</li> <li>6. Ingresa su contraseña con un largo mínimo de 6 caracteres</li> <li>7. Confirma su contraseña ingresando un valor diferente que el ingresado en el paso 6</li> </ol>	País: Uruguay Nombre: Usuario Apellido: Testing Email: usutest@gmail.com Nombre de usuario: usuario Contraseña: 123123 Confirmar Contraseña: 123123	<p>El usuario no queda registrado en la aplicación, sigue visualizando la pantalla de registro y se muestra el mensaje “No quedaste registrado, el nombre de usuario ya existe”. Se le permite el reingreso de los datos.</p>	P

		8. Presiona el botón “Registrar”.		
--	--	-----------------------------------	--	--

R-04: *Logout*

Precondiciones: El usuario inició sesión con su usuario

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R04-T01	El usuario cierra su sesión.	El usuario: Presiona en el menú botón “Cerrar sesión”.		Se cierra la sesión del usuario y es redireccionado a la pantalla de formulario de inicio de sesión.	P

**Casos de prueba para requerimientos asociados al listado de reclamos**

R-05: Recibir notificaciones de ML

Precondiciones: El usuario inició sesión con su usuario, y se encuentra en la pantalla principal de la aplicación. (URL: <https://grmlwww.azurewebsites.net/Home/Index>)

Para poder ejecutar la prueba de este requerimiento, el comprador debió crear un reclamo a nuestro usuario de prueba, desde la plataforma de ML, para probar que llega la notificación a nuestra API.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R05-T01	El usuario va a tener un nuevo reclamo	El comprador en ML: 1. Crea un reclamo sobre una		El usuario visualizara un nuevo reclamo abierto en el listado de	P

	abierto.	compra que le había realizado a nuestro usuario. El usuario: 2. Actualiza la pagina		reclamos.	
R05-T02	El usuario va a tener un reclamo cerrado.	El comprador en ML: 1. Cierra el reclamo que tenía abierto con nuestro usuario. El usuario: 2. Actualiza la pagina		El usuario visualizara como en el listado el reclamo cambia de abierto a cerrado.	P

#### R-07: Listado de todos los reclamos

Precondiciones: El usuario inició sesión con su usuario y se encuentra en la pantalla principal de la aplicación. (URL: <https://grmlwww.azurewebsites.net/Home/Index>)

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R07-T01	El usuario tiene reclamos abiertos o cerrados.			<p>El usuario visualiza el título “Reclamos”, dos filtros (uno para filtrar por tienda y otro por diferentes criterios de búsqueda) y el listado de todos sus reclamos primero los abiertos y luego los cerrados, ordenados por mayor precio y antigüedad descendente y ascendente respectivamente. De cada uno de los reclamos listados debe ver:</p> <ul style="list-style-type: none"> <li>- La imagen del artículo que vendió.</li> <li>- El id del reclamo y estado (abierto o cerrado).</li> <li>- La cantidad, nombre del artículo que vendió y monto total de la venta.</li> <li>- La información si el reclamo afecta o no su reputación.</li> <li>- El tipo: mediación: “Reclamo abierto/cerrado”, compra cancelada, devolución del producto o vendedor canceló.</li> </ul>	P

				- El nombre del comprador. - La cantidad de días desde que se abrió el reclamo hasta la fecha. - Un botón con el texto “Ver reclamo”.	
R07-T02	El usuario no tiene reclamos abiertos o cerrados.			El usuario visualiza el título “Reclamos”, dos filtros (uno para filtrar por tienda y otro por diferentes criterios de búsqueda) y el mensaje “No tiene reclamos”	P
R07-T03	El usuario no tiene asociada su cuenta de ML.			El usuario visualiza el título “Reclamos”, dos filtros (uno para filtrar por tienda y otro por diferentes criterios de búsqueda) y el mensaje “Primero debe asociar su cuenta de Mercado Libre”	P

#### R-08: Identificación visual

Precondiciones: El usuario inició sesión con su usuario, tiene reclamos abiertos y/o cerrados y se encuentra en la pantalla principal de la aplicación. (URL: <https://grmlwww.azurewebsites.net/Home/Index>)

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Fallá, P=pasa)
R08-T01	El usuario tiene al menos un reclamo abierto esperando por su respuesta.			El usuario visualiza la pantalla principal de la aplicación con el listado de todos sus reclamos. Los reclamos que se encuentran abiertos y esperando por su respuesta presentan un borde de color rojo.	P
R08-T02	El usuario tiene reclamos, pero ninguno está abierto.			El usuario visualiza la pantalla principal de la aplicación con el listado de todos sus reclamos y ningún reclamo presenta un borde rojo.	P
R08-T03	El usuario tiene reclamos, pero ninguno está esperando por su respuesta.			El usuario visualiza la pantalla principal de la aplicación con el listado de todos sus reclamos y ningún reclamo presenta un borde rojo.	P

## R-09: Filtrar por tiendas

**Precondiciones:** El usuario inició sesión con su usuario, tiene multi tiendas, tiene reclamos abiertos y/o cerrados y se encuentra en la pantalla principal de la aplicación. (URL: <https://grmlwww.azurewebsites.net/Home/Index>)

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R09-T01	El usuario selecciona una de sus tiendas para filtrar la misma tiene reclamos asicados.	El usuario: 1. Selecciona en el selector de tiendas la opción “Tienda testing”.		El usuario visualiza la pantalla principal de la aplicación con el listado de los reclamos que pertenecen a la tienda seleccionada.	P
R09-T02	El usuario selecciona una de sus tiendas para filtrar la misma no tiene reclamos asicados.	El usuario: 1. Selecciona en el selector de tiendas a la opción “Tienda testing sin reclamos”.		El usuario visualiza la pantalla principal de la aplicación con el mensaje “No tiene reclamos”.	P
R09-T03	El usuario selecciona la opción para visualizar todos los reclamos.	El usuario: 1. Selecciona en el selector de tiendas la opción “Todos los reclamos”		El usuario visualiza la pantalla principal de la aplicación con el listado de todos sus reclamos.	P

## R-10: Búsqueda por filtro

Precondiciones: El usuario inició sesión con su usuario, tiene reclamos abiertos y/o cerrados y se encuentra en la pantalla principal de la aplicación. (URL: <https://grmlwww.azurewebsites.net/Home/Index>)

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R10-T01	El usuario busca los reclamos abiertos.	El usuario: 1. Selecciona en el selector por filtros la opción “Abiertos”.		El usuario visualiza la pantalla principal de la aplicación con el listado de los reclamos que se encuentran abiertos.	P
R10-T02	El usuario busca los reclamos cerrados.	El usuario: 1. Selecciona en el selector por filtros la opción “Cerrados”.		El usuario visualiza la pantalla principal de la aplicación con el listado de los reclamos que se encuentran cerrados.	P
R10-T03	El usuario busca los reclamos en mediación ML.	El usuario: 1. Selecciona en el selector por filtros la opción “En mediación ML”.		El usuario visualiza la pantalla principal de la aplicación con el listado de los reclamos abiertos o cerrados que se encuentran en mediación con ML.	P
R10-T04	El usuario busca los reclamos abiertos en mediación con ML.	El usuario: 1. Selecciona en el selector por filtros la opción “En mediación ML abiertos”.		El usuario visualiza la pantalla principal de la aplicación con el listado de los reclamos abiertos que se encuentran en mediación con ML.	P
R10-T05	El usuario busca los reclamos que están esperando su respuesta.	El usuario: 1. Selecciona en el selector por filtros la opción “Tu respuesta”.		El usuario visualiza la pantalla principal de la aplicación con el listado de los reclamos que se encuentran esperando por su respuesta.	P
R10-T06	El usuario busca los reclamos que están esperando la respuesta del comprador.	El usuario: 1. Selecciona en el selector por filtros la opción “Respuesta del comprador”.		El usuario visualiza la pantalla principal de la aplicación con el listado de los reclamos abiertos que se encuentran esperando por la respuesta del comprador.	P

				comprador.	
R10-T07	El usuario busca los reclamos que están esperando la respuesta de ML.	El usuario: 1. Selecciona en el selector por filtros la opción “Respuesta del Mercado Libre”.		El usuario visualiza la pantalla principal de la aplicación con el listado de los reclamos abiertos que se encuentran esperando por la respuesta de ML.	P
R10-T08	El usuario busca los reclamos de tipo PDD (producto defectuoso).	El usuario: 1. Selecciona en el selector por filtros la opción “PDD”.		El usuario visualiza la pantalla principal de la aplicación con el listado de los reclamos de tipo PDD.	P
R10-T09	El usuario busca los reclamos de tipo PNR (pagado y no recibido).	El usuario: 1. Selecciona en el selector por filtros la opción “PNR”.		El usuario visualiza la pantalla principal de la aplicación con el listado de los reclamos de tipo PNR.	P
R10-T10	El usuario selecciona la opción para visualizar todos los reclamos.	El usuario: 1. Selecciona en el selector de filtros la opción “Todos los reclamos”		El usuario visualiza la pantalla principal de la aplicación con el listado de todos sus reclamos.	P
R10-T11	Búsqueda por filtro combinado.	El usuario: 1. Selecciona en el selector por filtros la opción “Abiertos”. 2. Selecciona en el selector de tiendas la opción “Tienda testing”.		El usuario visualiza la pantalla principal de la aplicación con el listado de los reclamos abiertos pertenecientes a la tienda seleccionada.	P

R-35: Incorporación visual en listado de los reclamos de si el reclamo afecta la reputación del vendedor y del identificador de la orden de compra en el detalle del reclamo.

Precondiciones: El usuario inició sesión con su usuario, tiene reclamos abiertos y/o cerrados y se encuentra en la pantalla principal de la aplicación. (URL: <https://grmlwww.azurewebsites.net/Home/Index>)

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R35-T01	Visualizar en el listado de los reclamos si afecta la reputación del vendedor.			El usuario visualiza la pantalla principal de la aplicación con el listado de los reclamos. En cada uno de los reclamos debajo de la información del producto vendido puede visualizar si el mismo afecta su reputación o no. En caso de que afecte la reputación esta información se muestra en color rojo.	P
R35-T02	Visualizar en el detalle del reclamo el identificador de la orden de compra.	El usuario: 1. Presiona el botón de “Ver reclamo” en uno de sus reclamos y se encuentra en la pantalla del reclamo seleccionado. (URL <a href="https://grmlwww.azurewebsites.net/Claim/Detail?id=[id del reclamo seleccionado]">https://grmlwww.azurewebsites.net/Claim/Detail?id=[id del reclamo seleccionado]</a> )		El usuario visualiza la pantalla con el detalle del reclamo y en la información del detalle de la venta ve el Id de la orden de compra.	P

R-38: Filtrar reclamos en mediación con ML

Precondiciones: El usuario inició sesión con su usuario, tiene reclamos abiertos y/o cerrados y se encuentra en la pantalla principal de la aplicación. (URL: <https://grmlwww.azurewebsites.net/Home/Index>)

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R38-T01	El usuario tiene reclamos en mediación con ML y busca los reclamos en mediación con ML.	El usuario: 1. Selecciona en el selector por filtros la opción “En mediación ML”.		El usuario visualiza la pantalla principal de la aplicación con el listado de los reclamos que se encuentran abiertos o cerrados y en mediación con ML.	P
R38-T02	El usuario tiene reclamos en mediación con ML y busca los reclamos abiertos en mediación con ML.	El usuario: 1. Selecciona en el selector por filtros la opción “En mediación ML abiertos”.		El usuario visualiza la pantalla principal de la aplicación con el listado de los reclamos que se encuentran y en mediación con ML.	P
R38-T03	El usuario no tiene reclamos en mediación con ML.	El usuario: 1. Selecciona en el selector por filtros la opción “En mediación ML”.		El usuario visualiza la pantalla principal de la aplicación con el mensaje “No tiene reclamos”.	P

## Casos de prueba para requerimientos asociados al detalle del reclamo

R-11: Ver detalle del reclamo

**Precondiciones:** El usuario inició sesión con su usuario, tiene reclamos abiertos y/o cerrados, en el listado de los reclamos presionó el botón “Ver reclamo” y se encuentra en la pantalla del reclamo seleccionado. (URL [https://grmlwww.azurewebsites.net/Claim/Detail?id=\[id del reclamo seleccionado\]](https://grmlwww.azurewebsites.net/Claim/Detail?id=[id del reclamo seleccionado]))

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R11-T01	Visualizar el detalle de un reclamo cerrado.	El usuario: Presiona el botón “Ver reclamo” en un reclamo que se encuentra cerrado.		<p>El usuario visualiza la pantalla del detalle del reclamo con los siguientes datos:</p> <ol style="list-style-type: none"> <li>1. Id del reclamo, fecha de abierto el reclamo, botón “Ver en mercadolibre”.</li> <li>2. La resolución esperada por el comprador.</li> <li>3. En el caso que el reclamo está en mediación con ML una leyenda con la información de que el reclamo se encuentra en mediación con ML.</li> <li>4. La información de si el reclamo afecta o no la reputación del vendedor.</li> <li>5. Todos los mensajes enviados por todas las partes especificando fecha y hora de envío del mensaje, remitente, el texto del mensaje y los archivos adjuntos (en caso de que tenga).</li> <li>6. La información del detalle del reclamo: etapa, estado y si la compra fue o no recibida</li> <li>7. La información del detalle de la venta: Id de la orden, fecha de venta, ítem vendido cantidad, precio y URL de la publicación del artículo.</li> <li>8. El detalle del comprador: nombre y apodo.</li> </ol>	P

				9. Un cartel indicando que el reclamo se encuentra cerrado.	
R11-T02	Visualizar el detalle de un reclamo abierto en el que el comprador tiene habilitada la acción de mandar mensaje al comprador.	El usuario: Presiona el botón “Ver reclamo” en un reclamo que se encuentra abierto con la acción de enviar mensaje al comprador habilitada.		El usuario visualiza la pantalla del detalle del reclamo con los siguientes datos: Punto 1, 2, 3, 4, 5, 6, 7, 8, ídem R11-T01. 9. Un botón con la imagen de un <i>clip</i> que le permite al usuario seleccionar un archivo para adjuntar al mensaje. 10. Un cuadro que le permite escribir un mensaje, indicando que el receptor del mensaje es el comprador 11. Un botón con la imagen de flecha para realizar el envío del mensaje.	P
R11-T03	Visualizar el detalle de un reclamo abierto en mediación con mercado libre en el que el comprador tiene habilitada la acción de mandar mensaje a ML.	El usuario: Presiona el botón “Ver reclamo” en un reclamo que se encuentra abierto con la acción de enviar mensaje a ML habilitada.		El usuario visualiza la pantalla del detalle del reclamo con los siguientes datos: Punto 1, 2, 3, 4, 5, 6, 7, 8, ídem R11-T01. 9. Un botón con la imagen de un <i>clip</i> que le permite al usuario seleccionar un archivo para adjuntar al mensaje. 10. Un cuadro que le permite escribir un mensaje, indicando que el receptor del mensaje es ML 11. Un botón con la imagen de flecha para realizar el envío del mensaje.	P
R11-T04	Visualizar el detalle de un reclamo abierto en el que el comprador tiene habilitada la acción de devolver el dinero al comprador.	El usuario: Presiona el botón “Ver reclamo” en un reclamo que se encuentra abierto con la acción de devolver el dinero al comprador habilitada.		El usuario visualiza la pantalla del detalle del reclamo con los siguientes datos: Punto 1, 2, 3, 4, 5, 6, 7, 8, ídem R11-T01. 9. El texto “También puedes” 10. Un enlace con el texto “Devolver el dinero” para ver el reclamo con la herramienta de ML para devolver el dinero al comprador desde la aplicación de ML.	P
R11-T05	Visualizar el	El usuario:		El usuario visualiza la pantalla del detalle del reclamo con los	P

	detalle de un reclamo abierto en el que el comprador tiene habilitada la acción de pedir mediación a ML.	Presiona el botón “Ver reclamo” en un reclamo que se encuentra abierto con la acción de pedir mediación a ML habilitada.		siguientes datos: Punto 1, 2, 3, 4, 5, 6, 7, 8, ídem R11-T01. 9. El texto “También puedes” 10. Un botón con el texto “Pedir ayuda a Mercado Libre” para solicitar mediación a ML.	
R11-T06	Visualizar el detalle de un reclamo abierto en el que el comprador tiene habilitada la acción de enviar evidencia de envío.	El usuario: Presiona el botón “Ver reclamo” en un reclamo que se encuentra abierto con la acción de enviar evidencia de envío habilitada.		El usuario visualiza la pantalla del detalle del reclamo con los siguientes datos: Punto 1, 2, 3, 4, 5, 6, 7, 8, ídem R11-T01. 9. Un recuadro con el texto “¿Ya enviaste el paquete?” 10. Un botón con el texto “Subir evidencia de envío”.	P
R11-T07	Visualizar el detalle de un reclamo abierto en el que el comprador tiene habilitada la acción de enviar una probable fecha de envío.	El usuario: Presiona el botón “Ver reclamo” en un reclamo que se encuentra abierto con la acción de enviar una probable fecha de envío habilitada.		El usuario visualiza la pantalla del detalle del reclamo con los siguientes datos: Punto 1, 2, 3, 4, 5, 6, 7, 8, ídem R11-T01. 9. Un recuadro con el texto “¿Cuándo le enviarás el paquete?” 10. Un botón con el texto “Subir fecha de envío”.	P
R11-T08	Visualizar el detalle de un reclamo abierto en el que el comprador subió	El usuario: Presiona el botón “Ver reclamo” en un reclamo que se encuentra abierto		El usuario visualiza la pantalla del detalle del reclamo con los siguientes datos: Punto 1, 2, 3, 4, 5, 6, 7, 8, ídem R11-T01. 9. Un recuadro con el texto “¡Gracias! Le avisamos a tu comprador que ya enviaste el paquete. En cuanto	P

	la evidencia de envío.	en el que el comprador subió la evidencia de envío.		Mercado Libre confirme que está en camino, cierra el reclamo.” 10. El detalle de todos los datos que subió el vendedor como evidencia de envío del paquete.	
--	------------------------	---	--	--	--

R-13: Ver resolución propuesta por el comprador

Precondiciones: El usuario inició sesión con su usuario, tiene reclamos, en el listado de los reclamos presionó el botón “Ver reclamo” de un reclamo abierto o cerrado y se encuentra en la pantalla del reclamo seleccionado. (URL [https://grmlwww.azurewebsites.net/Claim/Detail?id=\[id del reclamo seleccionado\]](https://grmlwww.azurewebsites.net/Claim/Detail?id=[id del reclamo seleccionado]))

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R13-T01	Ver resolución propuesta por el comprador cuando quiere que se le devuelva el dinero.			En el detalle del reclamo abajo del id del reclamo el usuario visualiza el texto: “Tu comprador quiere que le devuelvas el dinero”	P
R13-T02	Ver resolución propuesta por el comprador cuando quiere que le llegue el producto.			En el detalle del reclamo abajo del id del reclamo el usuario visualiza el texto: “Tu comprador quiere que le llegue el producto”	P
R13-T03	Ver resolución propuesta por el comprador cuando quiere cambiar el producto.			En el detalle del reclamo abajo del id del reclamo el usuario visualiza el texto: “Tu comprador quiere cambiar el producto”	P
R13-T04	Ver resolución propuesta por el comprador cuando quiere devolver el producto y que le devuelvan el dinero.			En el detalle del reclamo abajo del id del reclamo el usuario visualiza el texto: “Tu comprador quiere devolverte el producto y obtener su dinero”	P

### R-17: Visualizar mediación

**Precondiciones:** El usuario inició sesión con su usuario, tiene reclamos, en el listado de los reclamos presionó el botón “Ver reclamo” de un reclamo abierto o cerrado y se encuentra en la pantalla del reclamo seleccionado. (URL [https://grmlwww.azurewebsites.net/Claim/Detail?id=\[id del reclamo seleccionado\]](https://grmlwww.azurewebsites.net/Claim/Detail?id=[id del reclamo seleccionado]))

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R17-T01	El reclamo está en mediación con ML.			El usuario visualiza una leyenda con la información de que el reclamo se encuentra en mediación con ML en el detalle del reclamo debajo de la resolución esperada por el comprador.	P
R17-T02	El reclamo no está en mediación con ML.			El usuario visualiza el detalle del reclamo sin la leyenda “Reclamo en mediación con Mercado Libre”.	P

### R-05: Recibir notificaciones de ML

**Precondiciones:** El usuario inició sesión con su usuario, tiene reclamos, en el listado de los reclamos presionó el botón “Ver reclamo” de un reclamo abierto y se encuentra en la pantalla del reclamo seleccionado. (URL [https://grmlwww.azurewebsites.net/Claim/Detail?id=\[id del reclamo seleccionado\]](https://grmlwww.azurewebsites.net/Claim/Detail?id=[id del reclamo seleccionado]))

Para poder ejecutar la prueba de este requerimiento, el comprador debió solicitar mediación, en un reclamo abierto a nuestro usuario de prueba, desde la plataforma de ML.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R05-T05	El comprador solicita que se abra	El comprador en ML: 1. Solicita la mediación, desde		El usuario visualizara que el reclamo seleccionado ahora está en	P

	mediación.	la plataforma de ML, en un reclamo que tiene abierto con nuestro usuario de prueba. El usuario: 1. Actualiza la página.		mediación.	
--	------------	---	--	------------	--

### Casos de prueba para requerimientos asociados a la gestión de los reclamos

R-12: Comunicación entre las partes

Precondiciones: El usuario inició sesión con su usuario, tiene reclamos, en el listado de los reclamos presionó el botón “Ver reclamo” de un reclamo abierto y se encuentra en la pantalla del reclamo seleccionado y tiene la opción de enviar un mensaje al comprador o a ML. (URL [https://grmlwww.azurewebsites.net/Claim/Detail?id=\[id del reclamo seleccionado\]](https://grmlwww.azurewebsites.net/Claim/Detail?id=[id del reclamo seleccionado]))

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R12-T01	Enviar mensaje con texto y sin archivo adjunto	El usuario: 1. Escribe un mensaje. 2. Presiona el botón con la imagen de flecha para realizar el envío del mensaje.	Mensaje: “Estimado: lamento mucho el inconveniente pronto tendrá su paquete. Saludos.”	El mensaje es enviado por el vendedor y recibido por el destinatario (comprador o ML). El usuario visualiza un texto con fondo verde que dice: “Mensaje enviado con éxito”.	P
R12-T02	Enviar mensaje con texto y con archivo adjunto	El usuario: 1. Presiona el botón con la imagen de un <i>clip</i> . 2. Selecciona el archivo que desea adjuntar. 3. Escribe un mensaje.	Nombre del archivo adjunto: boleta.jpg Tipo de archivo adjunto: jpeg Mensaje: “Estimado: lamento mucho el	El mensaje con el archivo adjunto es enviado por el vendedor y recibido por el destinatario (comprador o ML). El usuario visualiza un texto con fondo verde que dice: “Mensaje enviado con éxito”.	P

		<p>4. Presiona el botón con la imagen de flecha para realizar el envío del mensaje.</p>	<p>inconveniente pronto tendrá su paquete. Saludos.”</p>		
R12-T03	Enviar mensaje con archivo adjunto y sin texto	<p>El usuario:</p> <ol style="list-style-type: none"> <li>1. Presiona el botón con la imagen de un <i>clip</i>.</li> <li>2. Selecciona el archivo que desea adjuntar.</li> <li>3. Presiona el botón con la imagen de flecha para realizar el envío del mensaje.</li> </ol>	<p>Nombre del archivo adjunto: boleta.jpg Tipo de archivo adjunto: jpeg</p>	<p>El mensaje con el archivo adjunto no es enviado. El usuario visualiza un texto con fondo rojo que dice: “Error! Error al enviar el mensaje”.</p>	P

#### R-05: Recibir notificaciones de ML

**Precondiciones:** El usuario inició sesión con su usuario, tiene reclamos, en el listado de los reclamos presionó el botón “Ver reclamo” de un reclamo abierto y se encuentra en la pantalla del reclamo seleccionado. (URL [https://grmlwww.azurewebsites.net/Claim/Detail?id=\[id del reclamo seleccionado\]](https://grmlwww.azurewebsites.net/Claim/Detail?id=[id del reclamo seleccionado]))

Para poder ejecutar la prueba de este requerimiento, el comprador debió enviar el mensaje, a un reclamo abierto a nuestro usuario de prueba, desde la plataforma de ML, para probar que llega la notificación a nuestra API.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R05-T03	Visualizar los mensajes que llegan desde una notificación.	<p>El comprador en ML:</p> <ol style="list-style-type: none"> <li>1. Envía un mensaje al vendedor en el reclamo abierto con</li> </ol>		<p>El usuario visualizará un mensaje nuevo del comparador.</p>	P

		<p>el id seleccionado.</p> <p>El usuario:</p> <ol style="list-style-type: none"> <li>2. Actualiza la página.</li> </ol>			
R05-T04	Visualizar los mensajes con archivos adjuntos, que llegan desde una notificación.	<p>El comprador en ML:</p> <ol style="list-style-type: none"> <li>1. Envía un mensaje con archivo adjunto al vendedor en el reclamo abierto con el id seleccionado.</li> </ol> <p>El usuario:</p> <ol style="list-style-type: none"> <li>3. Actualiza la página.</li> </ol>		<p>El usuario visualizará un nuevo mensaje del comprador con el archivo adjunto correspondiente.</p>	P

#### R-14: Resolución de reclamos

Precondiciones: El usuario inició sesión con su usuario, tiene reclamos, en el listado de los reclamos presionó el botón “Ver reclamo” de un reclamo abierto y se encuentra en la pantalla del reclamo seleccionado. (URL [https://grmlwww.azurewebsites.net/Claim/Detail?id=\[id del reclamo seleccionado\]](https://grmlwww.azurewebsites.net/Claim/Detail?id=[id del reclamo seleccionado]))

Ver desde R11-T02 a R11-T07

## R-15: Resolución de reembolso

**Precondiciones:** El usuario inició sesión con su usuario, también está logueado en ML con su cuenta que tiene asociada al usuario logueado en nuestra aplicación, tiene reclamos, en el listado de los reclamos presionó el botón “Ver reclamo” de un reclamo abierto y se encuentra en la pantalla del reclamo seleccionado. (URL [https://grmlwww.azurewebsites.net/Claim/Detail?id=\[id del reclamo seleccionado\]](https://grmlwww.azurewebsites.net/Claim/Detail?id=[id del reclamo seleccionado]))

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R15-T01	Devolver el dinero al comprador.	El usuario: 1. Presiona el enlace con el texto “Devolver el dinero”.		El usuario es redireccionado al sitio web de ML y visualiza el mismo reclamo que estaba mirando en nuestra aplicación. Allí tiene la opción para devolver el dinero al comprador.	P

R-16: Comprobante de envío

Precondiciones: El usuario inició sesión con su usuario, tiene reclamos, en el listado de los reclamos presionó el botón “Ver reclamo” de un reclamo abierto, tiene como posibles acciones subir evidencia de envío y/o enviar una probable fecha de envío y se encuentra en la pantalla del reclamo seleccionado. (URL [https://grmlwww.azurewebsites.net/Claim/Detail?id=\[id del reclamo seleccionado\]](https://grmlwww.azurewebsites.net/Claim/Detail?id=[id del reclamo seleccionado]))

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R16-T01	Subir evidencia de envío de tipo correo.	El usuario: 1. Presiona el botón “Subir evidencia de envío”. 2. Se abre un cuadro de diálogo. 3. Selecciona la opción correo. 4. Escribe el nombre de la empresa de correo. 5. Escribe el código de seguimiento. 6. Selecciona la fecha de envío. 7. Presiona el botón “Confirmar”.	¿Qué medio usaste?: Correo Empresa: El Correo Código de seguimiento: 123123 Fecha de envío: 02/09/2020	La evidencia de envío se sube correctamente, se cierra el cuadro de diálogo. Se muestra el mensaje “Evidencia de envío subida con éxito”. Se muestra el detalle de los datos enviados (ver detalle de caso de prueba R42-T01).	P
R16-T02	Subir evidencia de envío sin dato en alguno de los campos requeridos.	El usuario: 1. Presiona el botón “Subir evidencia de envío”. 2. Se abre un cuadro de diálogo. 3. Selecciona la opción correo. 4. Escribe el nombre de la	¿Qué medio usaste?: Correo Empresa: El Correo Código de seguimiento: 123123	La evidencia de envío no se sube, el cuadro de diálogo no se cierra y se muestra un mensaje de error debajo del campo Fecha de envío con el mensaje “Campo requerido”.	P

		<p>empresa de correo.</p> <p>5. Escribe el código de seguimiento.</p> <p>Presiona el botón “Confirmar”.</p>			
R16-T03	Subir evidencia de envío de tipo encomienda.	<p>El usuario:</p> <ol style="list-style-type: none"> <li>1. Presiona el botón “Subir evidencia de envío”.</li> <li>2. Se abre un cuadro de diálogo.</li> <li>3. Selecciona la opción encomienda.</li> <li>4. Escribe el nombre de la empresa de transporte.</li> <li>5. Escribe la terminal o sucursal de entrega.</li> <li>6. Selecciona la fecha de envío.</li> <li>7. Escribe el nombre del destinatario.</li> <li>8. Presiona el botón “Confirmar”.</li> </ol>	<p>¿Qué medio usaste?: Encomienda Empresa: Turil Terminal o sucursal de entrega: Tres cruces Fecha de envío: 02/09/2020 Nombre del destinatario: Juan Pérez</p>	<p>La evidencia de envío se sube correctamente, se cierra el cuadro de diálogo. Se muestra el mensaje “Evidencia de envío subida con éxito”. Se muestra el detalle de los datos enviados (ver detalle de caso de prueba R42-T02).</p>	P
R16-T04	Subir evidencia de envío de tipo entrega personal.	<p>El usuario:</p> <ol style="list-style-type: none"> <li>1. Presiona el botón “Subir evidencia de envío”.</li> <li>2. Se abre un cuadro de diálogo.</li> <li>3. Selecciona la opción entrega personal.</li> <li>4. Selecciona la fecha de envío.</li> <li>5. Presiona el botón “Confirmar”.</li> </ol>	<p>¿Qué medio usaste?: Entrega personal Fecha de envío: 02/09/2020</p>	<p>La evidencia de envío se sube correctamente, se cierra el cuadro de diálogo. Se muestra el mensaje “Evidencia de envío subida con éxito”. Se muestra el detalle de los datos enviados (ver detalle de caso de prueba R42-T03).</p>	P

R16-T05	Subir evidencia de envío de tipo email.	<p>El usuario:</p> <ol style="list-style-type: none"> <li>1. Presiona el botón “Subir evidencia de envío”.</li> <li>2. Se abre un cuadro de diálogo.</li> <li>3. Selecciona la opción entrega email.</li> <li>4. Escribe una dirección de email con formato válido de email.</li> <li>5. Selecciona la fecha de envío.</li> <li>6. Presiona el botón “Confirmar”.</li> </ol>	<p>¿Qué medio usaste?: Email            ¿A qué dirección de email lo enviaste?: shippingTesting@gmail.com            Fecha de envío: 02/09/2020</p>	<p>La evidencia de envío se sube correctamente, se cierra el cuadro de diálogo. Se muestra el mensaje “Evidencia de envío subida con éxito”. Se muestra el detalle de los datos enviados (ver detalle de caso de prueba R42-T04).</p>	P
R16-T06	Subir una probable fecha de envío del paquete	<p>El usuario:</p> <ol style="list-style-type: none"> <li>1. Presiona el botón “Subir fecha de envío”.</li> <li>2. Se abre un cuadro de diálogo.</li> <li>3. Selecciona la fecha que lo enviará.</li> <li>4. Presiona el botón “Confirmar”.</li> </ol>	<p>Fecha que lo enviarás: 10/09/2020</p>	<p>La probable fecha de se sube correctamente, se cierra el cuadro de diálogo. Se muestra el mensaje “Fecha de envío subida con éxito”.</p>	P

## R-18: Iniciar mediación

**Precondiciones:** El usuario inició sesión con su usuario, tiene reclamos, en el listado de los reclamos presionó el botón “Ver reclamo” de un reclamo abierto, tiene como posibles acciones solicitar mediación a ML y se encuentra en la pantalla del reclamo seleccionado. (URL [https://grmlwww.azurewebsites.net/Claim/Detail?id=\[id del reclamo seleccionado\]](https://grmlwww.azurewebsites.net/Claim/Detail?id=[id del reclamo seleccionado]))

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R18-T01	Iniciar mediación.	El usuario: 1. Presiona en el botón con el texto “Pedir ayuda a Mercado Libre”. 2. Se abre un cuadro de diálogo. 3. Presiona el botón “Confirmar”.		La solicitud de mediación se envía correctamente, se cierra el cuadro de diálogo. Se muestra el mensaje “Mediación solicitada con éxito”. En el detalle del reclamo se ve la leyenda “Reclamo en mediación con Mercado Libre”.	P
R18-T02	El usuario se arrepiente de iniciar mediación.	El usuario: 1. Presiona en el botón con el texto “Pedir ayuda a Mercado Libre”. 2. Se abre un cuadro de diálogo. 3. Presiona el botón “Cancelar”.		La solicitud de mediación no se envía y se cierra el cuadro de diálogo.	P

R-42: Visualizar detalle de evidencia de envío

**Precondiciones:** El usuario inició sesión con su usuario, tiene reclamos, en el listado de los reclamos presionó el botón “Ver reclamo” de un reclamo abierto del cual anteriormente subió la evidencia de envío y se encuentra en la pantalla del reclamo seleccionado. (URL [https://grmlwww.azurewebsites.net/Claim/Detail?id=\[id del reclamo seleccionado\]](https://grmlwww.azurewebsites.net/Claim/Detail?id=[id del reclamo seleccionado]))

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R42-T01	Ver evidencia de envío de tipo correo.		En la evidencia de envío el usuario uso estos datos: ¿Qué medio usaste?: Correo Empresa: El Correo Código de seguimiento: 123123 Fecha de envío: 02/09/2020	En el detalle del reclamo se visualiza un recuadro con la siguiente información: “¡Gracias! Le avisamos a tu comprador que ya enviaste el paquete. En cuanto Mercado Libre confirme que está en camino cierra el reclamo. <ul style="list-style-type: none"> <li>- Forma de envío: Correo</li> <li>- Fecha de envío: 02/09/2020”</li> <li>- Empresa: El Correo</li> <li>- Número de seguimiento o de factura: 123123”</li> </ul>	P
R42-T02	Ver evidencia de envío de tipo encomienda.		En la evidencia de envío el usuario uso estos datos: ¿Qué medio usaste?: Encomienda Empresa: Turil Terminal o sucursal de entrega: Tres cruces Fecha de envío: 02/09/2020 Nombre del destinatario: Juan Pérez	En el detalle del reclamo se visualiza un recuadro con la siguiente información: “¡Gracias! Le avisamos a tu comprador que ya enviaste el paquete. En cuanto Mercado Libre confirme que está en camino cierra el reclamo. <ul style="list-style-type: none"> <li>- Forma de envío: Encomienda</li> <li>- Fecha de envío: 02/09/2020”</li> <li>- Empresa: Turil</li> <li>- Terminal o sucursal de entrega: Tres cruces</li> <li>- Nombre del destinatario: Juan Pérez</li> </ul>	P
R42-T03	Ver evidencia de		En la evidencia de envío el	En el detalle del reclamo se visualiza un recuadro con	P

	envío de tipo entrega personal.		usuario uso estos datos: ¿Qué medio usaste?: Entrega personal Fecha de envío: 02/09/2020	la siguiente información: “¡Gracias! Le avisamos a tu comprador que ya enviaste el paquete. En cuanto Mercado Libre confirme que está en camino cierra el reclamo. - Forma de envío: En persona, mensajería o medio propio - Fecha de envío: 02/09/2020”	
R42-T04	Ver evidencia de envío de tipo email.		En la evidencia de envío el usuario uso estos datos: ¿Qué medio usaste?: Email ¿A qué dirección de e-mail lo enviaste?: shippingTesting@gmail.com Fecha de envío: 02/09/2020	En el detalle del reclamo se visualiza un recuadro con la siguiente información: “¡Gracias! Le avisamos a tu comprador que ya enviaste el paquete. En cuanto Mercado Libre confirme que está en camino cierra el reclamo. - Forma de envío: Vía e-mail - Fecha de envío: 02/09/2020” - Dirección de e-mail: shippingTesting@gmail.com	P

## Casos de prueba para requerimientos asociados a los reportes

### R-19: Obtención de reportes

Precondiciones: El usuario inició sesión con su usuario y se encuentra en la pantalla de reporte. (URL <https://grmlwww.azurewebsites.net/Report/Index>)

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R19-T01	El usuario tiene asociada su cuenta de ML, tiene reclamos y tiendas.			Visualiza la tabla con los campos fecha, % afecta tu reputación, abiertos, cerrados, esperando tu respuesta, en mediación con ML, %PNR y %PDD con los datos del último mes. Y la tabla de los reclamos por tienda también con los datos del último mes.	P
R19-T02	Filtrar por rango de fechas.	El usuario: 1. Selecciona la fecha desde 2. Selecciona la fecha hasta 3. Presiona en el botón “Actualizar”.	Desde: 20/09/2020 Hasta: 22/09/2020	Visualiza la tabla con los campos fecha, % afecta tu reputación, abiertos, cerrados, esperando tu respuesta, en mediación con ML, %PNR y %PDD con los datos en el rango de fecha seleccionado 3 filas con datos. Y la tabla de los reclamos por tienda con los datos en el rango de fecha seleccionado.	P
R19-T03	Filtrar por rango de fechas valor desde mayor que hasta.	El usuario: 1. Selecciona la fecha desde 2. Selecciona la fecha hasta 3. Presiona en el	Desde: 22/09/2020 Hasta: 20/09/2020	Visualiza la tabla con los campos fecha, % afecta tu reputación, abiertos, cerrados, esperando tu respuesta, en mediación con ML, %PNR y %PDD sin datos. Y la tabla de los reclamos por tienda también sin datos. Se muestra la información “La fecha hasta debe ser	P

		botón “Actualizar”.		mayor que la fecha desde”.	
R19-T04	El usuario no tiene asociada su cuenta de ML.			Visualiza la tabla con los campos fecha, % afecta tu reputación, abiertos, cerrados, esperando tu respuesta, en mediación con ML, %PNR y %PDD sin datos. Y la tabla de los reclamos por tienda también sin datos. Se muestra la información “Primero debe asociar su cuenta de ML”.	P

## Casos de prueba para requerimientos asociados al Dashboard

R-21: *Dashboard*, R-22: *Dashboard* ampliado, R-37: *Dashboard* ampliado 2

Precondiciones: El usuario inició sesión con su usuario y se encuentra en la pantalla de *dashboard*. (URL <https://grmlwww.azurewebsites.net/Dashboard/Index>)

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R21-T01 R22-T01 R37-T01	El usuario tiene asociada su cuenta de ML, tiene reclamos y tiendas.			Visualiza el tablero con la información en tiempo real de % afecta tu reputación en los últimos 3 meses, cantidad de reclamos abiertos, reclamos cerrados, esperando tu respuesta, en mediación con ML, %PNR en los últimos 3 meses y %PDD en los últimos 3 meses. A su vez un recuadro con la información de la cantidad de reclamos por tienda.	P
R21-T02 R22-T02 R37-T02	El usuario no tiene asociada su cuenta de ML.			Visualiza la información “Primero debe asociar su cuenta de ML”.	P

## Casos de prueba para requerimientos asociados a las excepciones

R-39: Persistencia del log de excepciones

Este requerimiento no tiene una visualización en el *frontend*. Por lo que, para realizar las pruebas, se cambiaron a mano atributos de los objetos que devolvían las respuestas a las llamadas de la API de ML, logrando así que se *loguearan* excepciones.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Estado (F=Falla, P=pasa)
R39-T01	Se obtiene un listado de excepciones en un rango de fecha determinado.	Desde la herramienta <i>Postman</i> : 1. Se debe realizar un <i>get</i> a la url: <a href="https://grmlapi.azurewebsites.net/Exception/GetByDates">https://grmlapi.azurewebsites.net/Exception/GetByDates</a> .	Los parámetros fueron: <ul style="list-style-type: none"><li>• <i>from</i>: 06/08/2020</li><li>• <i>to</i>: 10/09/2020</li></ul>	Se obtienen un listado con todas las excepciones <i>logueadas</i> en ese rango de fechas.	P

## Anexo 5 – Validación del cliente sobre avance del producto (*sprint 7*)



Hugo Olivera <hugo.olivera@sherlock.solutions>  
para mí, Antonieta, Ricardo

lun., 27 jul. 12:09

Hola como están?  
Perdón la demora, quedé yo de responder esto y se me pasó totalmente.

Los cambios están perfectos, es exactamente lo que se precisa si y la visualización es la correcta.

Respecto a como viene el proyecto nos parece que vienen muy bien y venimos teniendo una comunicación fluida. Nos parece que viene todo muy alineado con nuestras expectativas y no tenemos dudas que vamos a llegar a buen puerto con este proyecto.

Estamos muy conformes por como vienen trabajando.

Hugo Olivera Alonso

Consultor IA



[www.sherlock-assistant.com](http://www.sherlock-assistant.com) /Cureim 1147- Montevideo - Uruguay / (598) 092 200 905

## Anexo 6 – Validación del cliente sobre avance del producto (*sprint 8*)

Gestor reclamos ML sprint 8 ➔ Recibidos x



**Maria Florencia Fernández Martínez** <florenciafermarti@gmail.com>  
para Antonieta, Hugo, Ricardo ▾

mar., 18 ago. 19:42 ⭐ ↗ :

Buenas tardes, cómo están?

Queríamos confirmar con ustedes que los siguientes requerimientos desarrollados en el último sprint (*sprint 8*), presentados el Miércoles 12/08 se adecuan a sus necesidades y cumplen con lo solicitado.

Requerimientos presentados del sprint 8:

- Comunicación entre las partes: El sistema debe tener la opción de responder los mensajes de un reclamo abierto con y sin archivo adjunto.
- Comprobante de envío: El sistema debe permitir enviar la evidencia de envío
- Visualizar mediación: El detalle del reclamo contará con la información de si está en mediación o no.

Desde ya muchas gracias!

Saludos!



**Hugo Olivera**  
para mí, Antonieta, Ricardo ▾

18 ago. 2020 21:58 ⭐ ↗ :

Si perfecto, seguimos avanzado según lo acordado.

Hugo Olivera Alonso

Consultor IA



[www.sherlock-assistant.com](http://www.sherlock-assistant.com) /Cureim 1147- Montevideo - Uruguay / (598) 092 200 905

## Anexo 7 – Validación del cliente sobre avance del producto (sprint 9)

Gestor reclamos ML sprint 9 ➔ Recibidos x

 María Florencia Fernández Martínez <florenciatermari@gmail.com>  
para Antonieta, Hugo, Ricardo

vie., 28 ago. 21:05 (hace 5 días) ⭐ 🤵 :

Buenas noches,

Queríamos confirmar con ustedes que los siguientes requerimientos desarrollados en el último sprint (sprint 9), que presentamos hoy se adecuan a sus necesidades y cumplen con lo solicitado.

Requerimientos presentados del sprint 9:

- Iniciar mediación con ML.
- Obtención de reportes
- Dashboard

Estos son los nuevos requerimientos que hablamos para incorporar:

- Dashboard: mostrar % de reclamos de tipo PDD y PNR en los últimos 3 meses
- Dashboard: mostrar cantidad de reclamos en mediación con ML
- Filtrar reclamos que están en mediación con ML

También tenemos una consulta respecto a los reclamos que están en mediación con ML tanto en el dashboard como en el filtro, deberíamos tomar solo los reclamos que se encuentran abiertos en mediación con ML o sería sobre el total es decir reclamos abiertos y cerrados en mediación con ML?

Desde ya muchas gracias!

Saludos!

 Hugo Olivera  
para mí, Antonieta, Ricardo

1 sept. 2020 9:36 (hace 14 horas) ⭐ 🤵

Hola, como están?

Respecto a los requerimientos presentados en el Sprint 9, todo estaba correcto.

Los 3 nuevos requerimientos son los que surgieron a partir de la demo realizada el viernes.

Respecto a los reclamos en mediación entendemos que es importante poder filtrar tanto por el total (abiertos + cerrados )como por los abiertos.

Saludos,

Hugo Olivera Alonso  
Consultor IA  
  
[www.sherlock-assistant.com](http://www.sherlock-assistant.com) /Cureim 1147- Montevideo - Uruguay / (598) 092 200 905

## Anexo 8 – Validación del cliente sobre avance del producto (*sprint 10*)

Cierre sprint 10  Recibidos 

 **María Florencia Fernández Martínez** <florencefermarti@gmail.com>  
para Antonieta, Hugo, Ricardo  mié., 9 sept. 22:02 (hace 4 días)   

Buenas noches,

**viados** Queríamos confirmar con ustedes que los siguientes requerimientos desarrollados en el último sprint (sprint 10), que presentamos hoy se adecuan a sus necesidades y cumplen con lo solicitado.

Requerimientos presentados del sprint 10:

- Obtención de reportes
- Dashboard ampliado
- Filtrar reclamos en mediación con ML

Desde ya muchas gracias!

Saludos!

---

 Libre de virus. [www.avast.com](http://www.avast.com)

---

 **Hugo Olivera**  
para mí, Antonieta, Ricardo  mié., 9 sept. 22:07 (hace 4 días)   

Todo en tiempo y forma, de acuerdo a lo definido.

Muchas gracias.

**Hugo Olivera Alonso**  
Consultor IA  
  
[www.sherlock-assistant.com](http://www.sherlock-assistant.com) /Cuaréim 1147- Montevideo - Uruguay / (598) 092 200 905

## Anexo 9 – Evaluación de satisfacción del cliente

En la imagen que se presenta a continuación se muestra el mail enviado por el cliente respondiendo a las diferentes preguntas que se realizaron para conocer el grado de satisfacción del mismo. Luego se transcribieron las preguntas y respuestas para poder ver de forma clara el contenido del mail.

Hugo Olivera  
para mí, Antonieta, Ricardo +

Estimadas,

Les respondemos abajo en cada punto.

Hugo Olivera Alonso  
Consultor IA  
  
www.sherlock-assistant.com /Cureim 1147- Montevideo - Uruguay / (598) 092 200 905

---

De: María Florencia Fernández Martínez <[florenciatercero@gmail.com](mailto:florenciatercero@gmail.com)>  
Fecha: sábado, 12 de setiembre de 2020, 17:39  
Para: Antonieta Alvarez Conti <[antonieta.alvarezconti@gmail.com](mailto:antonieta.alvarezconti@gmail.com)>, Hugo Olivera <[hugo.olivera@sherlock.solutions](mailto:hugo.olivera@sherlock.solutions)>, Ricardo Melo <[ricardo.melo@sherlock.solutions](mailto:ricardo.melo@sherlock.solutions)>  
Asunto: Cuestionario de cierre de proyecto académico

Hola, cómo están?

Finalizado el proyecto académico, nos gustaría hacerles algunas preguntas sobre el grado de satisfacción a lo largo del mismo en diferentes áreas.

1. En cuanto a la comunicación por parte del equipo, ¿considera que la misma fue buena respecto a frecuencia, medios utilizados y efectividad en las respuestas?  
SH – La comunicación fue adecuada y fluida a lo largo de todo el proyecto.

2. ¿Cómo considera que el equipo gestionó el proyecto desde sus inicios, etapa de desarrollo y el final?  
SH – Se cumplió con los tiempos y alcances definidos por lo cual consideramos que la gestión fue adecuada.

3. ¿Las entregas se realizaron según el tiempo estipulado?  
SH – Todas las entregas fueron en tiempo y forma.

4. ¿Considera que el equipo cumplió con todos los requerimientos acordados en el anteproyecto y con los que surgieron durante la etapa de desarrollo?  
SH – Sin lugar a dudas se alcanzaron todos los requerimientos dentro de los objetivos así como algunos que fueron surgiendo sobre la marcha en el proyecto.

5. ¿El proyecto logró satisfacer sus necesidades y logró abarcar la resolución del problema propuesto?  
SH – Se cumplieron todos los objetivos definidos en el alcance.

6. ¿Considera que el equipo se mostró receptivo y flexible a las sugerencias y solicitudes de cambios?  
SH – En todo momento fueron un equipo receptivo y enfocado en lograr los objetivos.

7. ¿Cómo evalúa el desempeño en la gestión de los cambios?  
SH – Los cambios que fueron surgiendo en el proceso fueron manejados de forma correcta, con impactos mínimos en cronogramas y entregas.

8. Comentarios finales:  
SH – Estamos más que conformes con respecto al desempeño del equipo, y que no solo se lograron los objetivos del proyecto en tiempo informa, sino que tanto en lo personal como en lo profesional fue una muy buena experiencia colaborar a lo largo de todo el proyecto.

Finalizado el proyecto académico, nos gustaría hacerles algunas preguntas sobre el grado de satisfacción a lo largo del mismo en diferentes áreas.

1. En cuanto a la comunicación por parte del equipo, ¿considera que la misma fue buena respecto a frecuencia, medios utilizados y efectividad en las respuestas?

SH – La comunicación fue adecuada y fluida a lo largo de todo el proyecto.

2. ¿Cómo considera que el equipo gestionó el proyecto desde sus inicios, etapa de desarrollo y el final?

SH – Se cumplió con los tiempos y alcances definidos por lo cual consideramos que la gestión fue adecuada.

3. ¿Las entregas se realizaron según el tiempo estipulado?

SH – Todas las entregas fueron en tiempo y forma.

4. ¿Considera que el equipo cumplió con todos los requerimientos acordados en el anteproyecto y con los que surgieron durante la etapa de desarrollo?

SH – Sin lugar a dudas se alcanzaron todos los requerimientos dentro de los objetivos así como algunos que fueron surgiendo sobre la marcha en el proyecto.

5. ¿El proyecto logró satisfacer sus necesidades y logró abarcar la resolución del problema propuesto?

SH – Se cumplieron todos los objetivos definidos en el alcance.

6. ¿Considera que el equipo se mostró receptivo y flexible a las sugerencias y solicitudes de cambios?

SH – En todo momento fueron un equipo receptivo y enfocado en lograr los objetivos.

7. ¿Cómo evalúa el desempeño en la gestión de los cambios?

SH – Los cambios que fueron surgiendo en el proceso fueron manejados de forma correcta, con impactos mínimos en cronogramas y entregas.

8. Comentarios finales:

SH – Estamos más que conformes con respecto al desempeño del equipo, y que no solo se lograron los objetivos del proyecto en tiempo y forma, sino que tanto en lo personal como en lo profesional fue una muy buena experiencia colaborar a lo largo de todo el proyecto.

## Anexo 10 – Manual de usuario

### 1. Introducción

En este documento se especifica el uso que se puede hacer de la aplicación, el fin de este documento es permitirle al usuario conocer las diferentes funcionalidades con las que cuenta la aplicación y cómo debería ser su uso. El usuario final de esta aplicación es cualquier vendedor que utilice la plataforma de Mercado Libre (ML) como medio de venta de sus artículos.

Este *software* permite al usuario gestionar sus reclamos de ML, cuenta con el listado de los reclamos ordenados de tal forma que le permite al usuario tomar acciones sobre los reclamos que es prioritario resolver, a su vez cuenta con filtros para poder encontrar los reclamos de forma más rápida. También puede realizar todas las acciones que se pueden hacer desde la plataforma de ML a la hora de gestionar los reclamos que se encuentran abiertos. Por último, pero no menos importante, le ofrece al usuario la posibilidad de ver datos relevantes sobre el historial de sus reclamos.

### 2. Uso de la aplicación

#### Acceder a la aplicación:

Ingresé en el navegador de su computadora a la siguiente URL:  
<https://grmlwww.azurewebsites.net/>



1. Puede observar en la pantalla el formulario para acceder a la aplicación con un usuario previamente registrado en el sitio.
2. O un enlace que lo redireccionará al formulario para el registro de nuevos usuarios.

Ingresé su nombre de usuario en el campo “Nombre de usuario” y su contraseña en el campo “Contraseña” Presione el botón “INGRESAR”.

En caso de que no ingrese de manera correcta su usuario o su contraseña vera el siguiente mensaje de error:

The image shows a login interface with a red header "LOGIN DE USUARIO". Below it, a green-bordered box contains the error message "1 Nombre de usuario o contraseña incorrectos". The form fields are labeled "NOMBRE DE USUARIO" and "CONTRASEÑA", each with an input field containing "usuarioTesting" and "Contraseña" respectively. A large red button at the bottom is labeled "INGRESAR".

¿No tiene una cuenta? [Registrarse](#)

## Registro de nuevos usuarios:

Si aún no tiene un usuario registrado en la aplicación lo puede realizar desde el formulario de registro el mismo se puede acceder desde la pantalla del *login* (como se explicó previamente) o directamente desde ésta URL: <https://grmlwww.azurewebsites.net/Users/Registration>

The screenshot shows a registration form titled "REGISTRO". The form includes a dropdown menu set to "Uruguay", and six text input fields for "Nombre", "Apellido", "Email", "Nombre de usuario", "Contraseña", and "Confirmar Contraseña". A large red "REGISTRAR" button is located at the bottom of the form.

¿Ya tienes una cuenta? [Entre aquí](#)

Si se encuentra en esta pantalla, pero ya cuenta con un usuario registrado puede presionar en el enlace con nombre “Entre aquí” que lo redireccionará al formulario para ingresar con su usuario.

A continuación, se explica cómo debe completar el formulario:

The image shows a registration form titled "REGISTRO". The form consists of eight fields, each with a corresponding number from 1 to 8 indicating the order of completion. The fields are arranged vertically. Step 1 is a dropdown menu for selecting a country, showing "Uruguay" as the selected option. Step 2 is a text input field for "Nombre". Step 3 is a text input field for "Apellido". Step 4 is a text input field for "Email". Step 5 is a text input field for "Nombre de usuario". Step 6 is a text input field for "Contraseña". Step 7 is a text input field for "Confirmar Contraseña". Step 8 is a red button labeled "REGISTRAR".

¿Ya tienes una cuenta? [Entre aquí](#)

Todos los campos son de ingreso obligatorio.

1. Selecciona de la lista de países el que corresponda.
2. Ingrese su nombre, el mismo debe contener entre 3 y 20 caracteres.
3. Ingrese su apellido, el mismo debe contener entre 3 y 20 caracteres.
4. Ingrese su email, el mismo debe ser con el formato adecuado de un correo electrónico.
5. Ingrese su nombre de usuario, el mismo debe contener entre 3 y 20 caracteres. (Este nombre de usuario es el que luego utilizará cada vez que ingrese a la aplicación en el *login*)
6. Ingrese una contraseña la cual debe recordar, la misma debe contener como mínimo 6 caracteres.
7. Vuelva a escribir la contraseña, la misma debe ser exactamente igual a la que ingresó en el paso anterior (paso 6).
8. Presione el botón “REGISTRAR”.

Si ingresó todos los datos de manera correcta su usuario quedó registrado, va a ver un mensaje indicando que quedó registrado y podrá ingresar a la aplicación desde el formulario de *login*.

En caso de haber ingresado un usuario que ya existe en el sistema será notificado:

## REGISTRO

No quedaste registrado, el nombre de usuario ya existe

Uruguay ▾

Florencia

Fernandez

flor@gmail.com

florfer

Contraseña

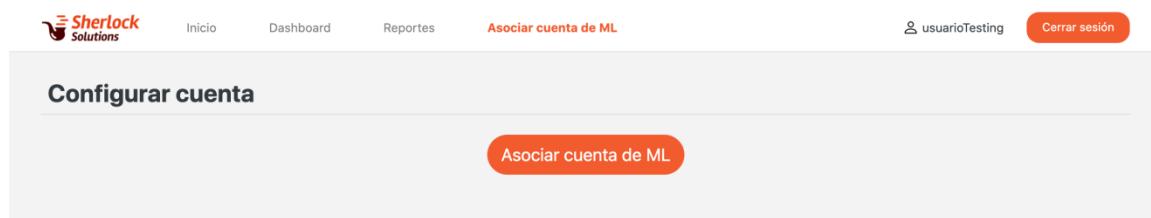
Confirmar Contraseña

**REGISTRAR**

¿Ya tienes una cuenta? [Entre aquí](#)

Puede editar los datos que había ingresado previamente y seguir el flujo de registro.

## Asociar su cuenta de ML a su usuario de aplicación:



The screenshot shows a navigation bar with links for Inicio, Dashboard, Reportes, and Asociar cuenta de ML. The user is logged in as 'usuarioTesting'. Below the navigation bar, the page title is 'Configurar cuenta'. A prominent orange button labeled 'Asociar cuenta de ML' is centered on the page.

Puede acceder a ver esta pantalla desde el menú de navegación en la opción "Asociar cuenta de ML".

Presione el botón "Asociar cuenta de ML", será redireccionado al sitio de ML. Si no está *logueado* con su cuenta de ML debe ingresar su usuario y contraseña, si ya está *logueado* no debe realizar este paso.



The screenshot shows a yellow header bar with the Mercado Libre logo and a user profile icon. Below it, a white dialog box is titled 'Autorizar conexión'. It features two circular icons: one with a square and another with a handshake. The text inside the box reads: 'Hola uy, autoriza a que la aplicación "GRML UY" se conecte con Mercado Libre'. Below this, there are three permission statements with corresponding icons: 'Podrá acceder a información de tu cuenta hasta que cancelés la autorización.', 'Podrá ver datos privados de tu cuenta.', and 'Podrá operar con tu cuenta.' At the bottom, there are two buttons: a blue 'Permitir' button and a grey 'Por ahora no' button.

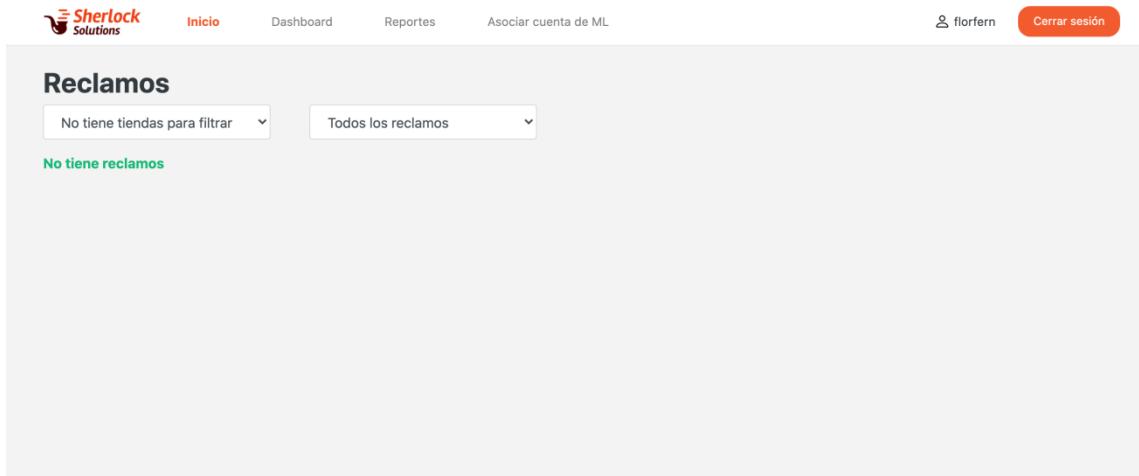
Luego deberá autorizar la conexión de la aplicación con su usuario presionando el botón "Permitir.". Finalmente, será redireccionado nuevamente a la aplicación donde se le informara si se pudo asociar su cuenta de manera exitosa o no.

## Información y usos en la página principal:

Si aún no tiene su cuenta de ML asociado a su usuario puede realizarlo desde el ítem “Asociar cuenta de ML” que se encuentra en el menú de la aplicación (1) y va a ver el mensaje que debe asociar su cuenta de Mercado Libre (2). Se presenta una imagen a modo ilustrativo:



Si tiene su cuenta de ML asociada y no tiene reclamos verá la información que no tiene reclamos.



Si tiene su cuenta de ML asociada y tiene reclamos verá la información que se muestra a continuación:

**Reclamos**

No tiene tiendas para filtrar ▾ Todos los reclamos ▾

Imagen	Detalles del Reclamo	Estado	Ultima actualización	Opciones
	Reclamo #5036014631 Reclamo abierto 1 x Hoja De Paltoro Monto total \$50 ● No afecta tu reputación	Reclamo abierto	Maria Alvarez hace 8 días	<a href="#">Ver reclamo</a>
	Reclamo #5036932061 Reclamo abierto 1 x Afilar Monto total \$50 ● Afecta tu reputación	Reclamo abierto	Maria Alvarez hace 0 días	<a href="#">Ver reclamo</a>
	Reclamo #5036935224 Reclamo abierto 1 x Limon Monto total \$50 ● No afecta tu reputación	Reclamo abierto	Maria Alvarez hace 0 días	<a href="#">Ver reclamo</a>

© 2020 - Proyecto final - GRML

A continuación, se detalla cada parte de esta pantalla y la información que allí se ve:

No tiene tiendas para filtrar ▾ Todos los reclamos ▾

1. Filtro de tiendas, le permite filtrar el listado de reclamos por cada una de sus tiendas en el caso que tenga un usuario de ML con multitiendas.

2. Filtro, le permite filtrar el listado de reclamos aplicando diferentes criterios de filtro.

© 2020 - Proyecto final - GRML

1. Filtro de tiendas, le permite filtrar el listado de reclamos por cada una de sus tiendas en el caso que tenga un usuario de ML con multitiendas.
2. Filtro, le permite filtrar el listado de reclamos aplicando diferentes criterios de filtro.

## Reclamos

No tiene tiendas para filtrar

	Reclamo #5036014631 Reclamo 1 x Hoja De Paltero Monto total \$50 <b>i</b> No afecta tu reputación
	Reclamo #5036932061 Reclamo 1 x Alfiler Monto total \$50 <b>i</b> Afecta tu reputación
	Reclamo #5036935224 Reclamo 1 x Limon Monto total \$50 <b>i</b> No afecta tu reputación

✓ Todos los reclamos

Estado

- Abiertos
- Cerrados
- En mediación ML
- En mediación ML abiertos

Esperando...

- Tu respuesta
- Respuesta del comprador
- Respuesta de Mercado Libre

Tipo

- PDD
- PNR

Los criterios de filtro son:

- Según el estado:
    - Abiertos: le permite ver todos los reclamos que se encuentran abiertos.
    - Cerrados: le permite ver todos los reclamos que se encuentran cerrados.
    - En mediación con ML: le permite ver todos los reclamos en mediación con ML y que se encuentran abiertos o cerrados.
    - En mediación ML abiertos: le permite ver todos los reclamos en mediación con ML y que se encuentran abiertos.
  - Esperando la respuesta de:
    - Tu respuesta: le permite ver todos los reclamos que están esperando por su respuesta.
    - Respuesta del comprador: le permite ver todos los reclamos que están esperando por la respuesta del comprador que inició el reclamo.
    - Respuesta de Mercado Libre: le permite ver todos los reclamos que están esperando por la respuesta de ML en el caso de los reclamos que se encuentran en mediación.
  - Tipo:
    - PDD: le permite ver todos los reclamos abiertos o cerrados que son de tipo producto defectuoso.
    - PNR: le permite ver todos los reclamos abiertos o cerrados que son de tipo pagado y no recibido.
3. Listado de los reclamos primero los abiertos y luego los cerrados. Los puede ver ordenados por mayor precio y por antigüedad descendente y ascendente

respectivamente. Los reclamos que ve con un borde rojo y fondo de color son los reclamos que se encuentran abiertos y esperando por su respuesta.

A continuación, se detalla la información del reclamo que se ve en la vista del listado de reclamos:

The screenshot shows a list of two open claims (reclamos abiertos) in a user interface. Each claim is represented by a card with the following details:

- Claim 1 (Left):**
  - Thumbnail: A small image of a product.
  - ID: Reclamo #5036014631
  - Status: **Reclamo abierto**
  - Item Details: 1x Hoja De Paltoro, Monto total \$50
  - Impact: **No afecta tu reputación**
- Claim 2 (Right):**
  - Thumbnail: A small image of a product.
  - ID: Reclamo #5036932061
  - Status: **Reclamo abierto**
  - Item Details: 1x Alfiler, Monto total \$50
  - Impact: **Afecta tu reputación**
  - Buyer: Maria Alvarez
  - Age: hace 9 días
  - Ver reclamo** button

1. La imagen del artículo que vendió.
2. El id del reclamo y el estado (abierto o cerrado).
3. La cantidad, nombre del artículo que vendió y monto total de la venta.
4. La información si el reclamo afecta o no su reputación.
5. El tipo: mediación: “Reclamo abierto/cerrado”, compra cancelada, devolución del producto o vendedor canceló.
6. El nombre del comprador.
7. La cantidad de días desde que se abrió el reclamo hasta la fecha.
8. Un botón con el texto “Ver reclamo”, presionando este botón puede acceder a ver el detalle del reclamo y tomar acciones en caso de que el reclamo este abierto.

## Ver detalle de un reclamo:

Una vez que ingrese a ver el detalle de uno de los reclamos, se le abrirá la siguiente pantalla, pudiendo ver toda la información que hay en el mismo y ejecutar acciones que brinda ML para gestionarlo.

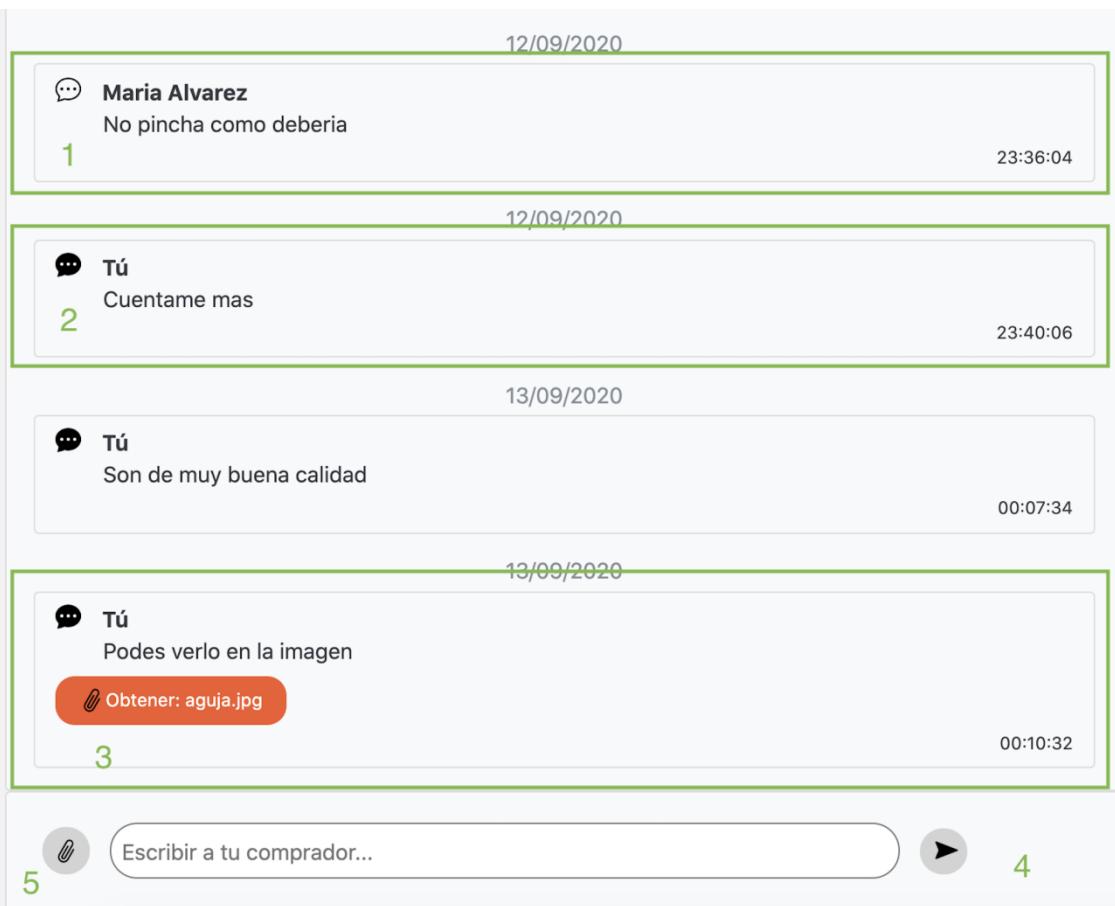
The screenshot shows the Sherlock interface for Mercado Libre. At the top, there are navigation links: Inicio, Dashboard, Reportes, Asociar cuenta de ML, userTesting, and Cerrar sesión. The main title is "Reclamo #5036932061". Below it, a message says "Tu comprador quiere cambiar el producto". A note indicates "Este reclamo afecta tu reputación". The main area displays a conversation between a buyer ("Maria Alvarez") and a seller ("Tú"). The buyer asks for more information and provides a link to a photo of an item. The seller responds with a detailed description of the item. To the right, there are three sections: "Detalle del reclamo" (claim details), "Detalle de la venta" (sale details) which includes a product image of a safety pin, and "Detalle del comprador" (buyer details). The buyer's name is Maria Alvarez and her alias is ALMA548186.

A continuación, se detalla cada parte de esta pantalla y la información que allí se ve:

This screenshot shows the same claim detail page with numbered callouts:

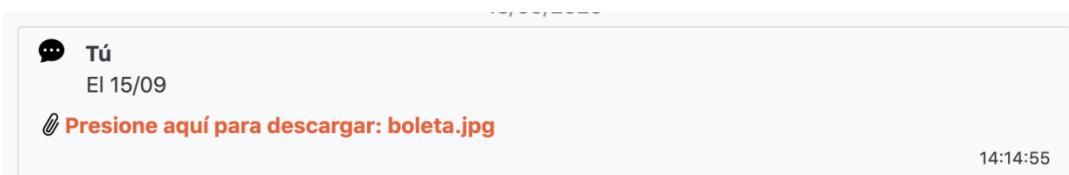
1. The claim ID: Reclamo #5036932061
2. The message: Tu comprador quiere cambiar el producto
3. The date the claim was created: 12/09/2020
4. The "Ver en mercadolibre" button
5. The note: Este reclamo afecta tu reputación

4. Id del Reclamo, es el número que ML identifica internamente cada reclamo.
5. Detalle de la resolución esperada por el comprador que abrió el reclamo.
6. La fecha que se creó el mismo.
7. Link a la vista del reclamo en la plataforma de ML (para que este link funcione exitosamente debe estar logeado a su cuenta en ML)
8. Información de si el reclamo afecta o no tu reputación.



En el área de mensajería con el comprador o con ML en caso de estar en mediación se podrá ver y accionar lo siguiente:

1. Ver los mensajes que el comprador haya enviado, identificándolos con el nombre del mismo.
2. Ver los mensajes que ha enviado.
3. Ver los mensajes tuyos o del comprador que tengan un archivo adjunto, seleccionando el botón “Obtener: nombre.png” se podrá acceder al *link* de descarga del archivo y posteriormente descargarlo.



4. En esta área el usuario podrá escribir y luego mandar el mensaje directamente al comprador o mediador en caso de estar en mediación. También podrá adjuntar un archivo a su mensaje con la selección del botón (5). Los archivos deben solo ser de tipo jpg, png o pdf y tener un peso menor a 5Mb. Una vez enviado el mensaje, verá un cartel verde en la parte superior de la pantalla donde se confirmará que el mensaje fue enviado con éxito.

En el caso que el reclamo este en mediación se verá un mensaje que advierte de este estado, y los mensajes serán intercambiados con ML, viéndose estos con un color de fondo distinto a los mensajes que envió el comprador.

## Reclamo #5018067785

22/04/2020

### Tu comprador quiere que le llegue el producto

Reclamo en mediación con Mercado Libre

Este reclamo no afecta tu reputación

22/04/2020

Martin Zanetti

Hola no me llegó

21:52:18

23/04/2020

**Mercado Libre**

Buenos días, esperamos que te encuentres bien.

Tu comprador nos comentó que no recibió la lampara led que te compró.

Teniendo en cuenta tu buen historial como vendedor, cerramos el reclamo y el dinero permanecerá en Mercado Pago. En el detalle del cobro podrás ver si tu dinero ya está disponible y en caso de que no lo esté, la fecha en la que lo estará.

Para próximas oportunidades, te recomendamos enviar los productos lo antes posible. De esta forma estarás brindando una buena experiencia de compra, que podrá verse reflejada en la calificación que tu comprador deje sobre tu atención.

¡Éxito en tus próximas ventas!

Saludos.

16:38:59

En la parte Inferior se encuentran las acciones que el usuario puede tomar frente al reclamo, las mismas pueden variar dependiendo el tipo de reclamo:



Escribir a tu comprador...



#### También puedes

[Devolver el dinero](#)

[Pedir ayuda a Mercado Libre](#)

1. Devolver el dinero: esta acción es un *link* al reclamo en la plataforma de ML, donde puedes acceder a la opción de devolver el dinero, al igual que el *link* en la parte superior, debe *loguearse* con su cuenta de ML para poder usar su plataforma.
2. Pedir ayuda a ML: seleccionando esta acción el usuario solicita a ML que intervenga en el reclamo y encuentre una resolución para las dos partes del reclamo, una vez que elija esta opción, no se podrá volver a enviar mensajes directos con el comprador, sino solo a su mediador (operadores de ML).

En la parte derecha de la pantalla, se podrá ver en un cuadro toda la información descriptiva y de interés del reclamo y la orden de compra que tiene asociada.

### ☰ Detalle del reclamo

Esperando a: **Respuesta de comprador** 1  
Etapa: **Reclamo** 2  
Estado: **Abierto** 3  
Compra: **No recibida** 4

### ⓘ Detalle de la venta

Id de la orden: **#4040938672** 5  
Fecha: **15/09/2020**  
Item:  
**Limon** 6  
1 x \$50  
[https://articulo.mercadolibre.com.uy/MLU-474929712-limon-\\_JM](https://articulo.mercadolibre.com.uy/MLU-474929712-limon-_JM)

### 👤 Detalle del comprador

Nombre: **Maria Alvarez**  
Apodo: **ALMA548186**

1. Este campo especifica cuál de las dos partes tiene pendiente contestar los mensajes y solo es visible cuando el reclamo está abierto.
2. Etapa en la que se encuentra el reclamo: puede tomar el valor de 'Reclamo' o de 'Disputa'. Este último se da cuando el reclamo se encuentra en mediación.
3. Estado en el que está el reclamo, puede tomar valores de 'Abierto' o 'Cerrado'.
4. Estado de envío del producto, puede tener valores de 'Recibida' o 'No recibida'.
5. Estos dos campos refieren a la orden a la que se le creó el reclamo, y los valores son el id de la misma y la fecha en la que se creó.
6. Producto que se compró en la orden, donde se detalla la imagen, el nombre, precio y un *link* a la publicación en ML.

7. Estos campos contienen el nombre y el apodo (en ML) del comprador que creó el reclamo.

Dependiendo el tipo de reclamo y las acciones que le habilite ML a realizar respecto a la gestión del reclamo, podrá subir una probable fecha de envío del paquete o la evidencia de que ya lo envío. En la siguiente foto se muestra la manera en que puede visualizar si tiene habilitadas estas acciones y cuáles son los pasos a seguir en cada caso.

The screenshot shows a web-based application for managing claims. At the top, there's a navigation bar with links for 'Inicio', 'Dashboard', 'Reportes', 'Asociar cuenta de ML', 'usuario', and 'Cerrar sesión'. The main content area is titled 'Reclamo #5037157692' and shows a message from a customer named 'Martin Zanetti' with the text 'Quiero!!!!'. Below this, there are two green-highlighted sections: one for 'Subir evidencia de envío' (Upload delivery evidence) and another for 'Subir fecha de envío' (Upload delivery date). To the right, there's a sidebar with 'Detalle del reclamo' (Claim detail) and 'Detalle de la venta' (Sale detail), which lists the item as a 'Lampara Led'.

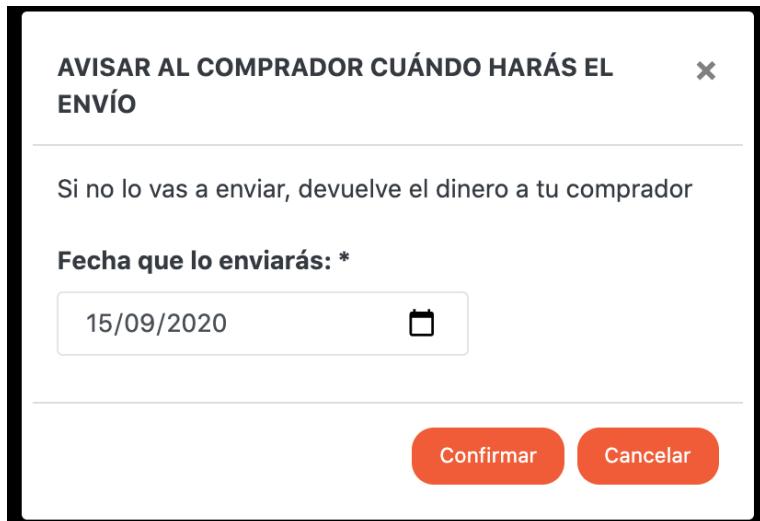
1. En el caso de que haya enviado el paquete, presione el botón "Subir evidencia de envío", para subir el comprobante. Los detalles de los pasos a seguir se detallan más adelante.
2. Si aún no envío el paquete, pero lo va a enviar, presione el botón "Subir fecha de envío". Los detalles de los pasos a seguir se detallan a continuación.

A continuación, se detallan los pasos a seguir para subir una fecha probable de envío:

1. Presione en el botón "Subir fecha de envío"

¿Cuándo le enviarás el paquete? Subir fecha de envío

2. Se abre un cuadro de diálogo donde debe seleccionar la fecha que enviará el paquete y presionar en el botón "Confirmar".



3. Se cierra el cuadro de diálogo y verá en la parte superior un cartel verde indicando que se subió correctamente o un cartel en rojo indicando que existió un error.
4. En la parte inferior verá la información con la fecha que indicó que iba a enviar el paquete y un botón que le permite subir la evidencia de envío del paquete una vez que realice el envío.

También puede tener disponible la opción para subir la evidencia de envío de un paquete. A continuación, se detallan los pasos que debe seguir para subir la evidencia de que envío el paquete:

1. Presione en el botón "confirma los datos del envío" o "Subir evidencia de envío" dependiendo el caso.

2. Se abre un cuadro de diálogo donde debe seleccionar el medio de envío que usó, dependiendo el que seleccione el formulario que debe completar:

a. Correo: todos los campos son de carga obligatoria, excepto subir un archivo de comprobante de envío. Complete todos los campos y luego presione el botón "Confirmar".

The dialog box has a title bar 'CÓMO ENTREGASTE EL PAQUETE' and a close button 'X'. It contains the following fields:

- ¿Qué medio usaste?**: A dropdown menu set to 'Correo'.
- Empresa:** \* A text input field containing 'El Correo'.
- Código de seguimiento:** \* A text input field containing '123123'.
- Fecha de envío:** \* A date input field set to '02/09/2020' with a calendar icon.
- Comprobante de envío:** A text input field with a file icon and the text 'Subir Archivo'.
- Buttons:** 'Confirmar' (orange) and 'Cancelar' (orange).

b. Encomienda: todos los campos son de carga obligatoria, excepto el documento del destinatario, código de seguimiento y subir un archivo de comprobante de envío. Complete todos los campos y luego presione el botón "Confirmar".

**CÓMO ENTREGASTE EL PAQUETE**

¿Qué medio usaste?

Empresa: \*

Terminal o sucursal de entrega: \*

Fecha de envío: \*

Nombre del destinatario: \*

Documento del destinatario:

Código de seguimiento:

Comprobante de envío:

- c. Entrega personal: Seleccione la fecha en la cual envío el paquete, opcionalmente puede subir un archivo como comprobante de envío. Finalmente presione el botón "Confirmar".

**CÓMO ENTREGASTE EL PAQUETE**

¿Qué medio usaste? Entrega personal

Fecha de envío: \* 02/09/2020

Comprobante de envío: Subir Archivo

Confirmar Cancelar

- d. Email: Todos los campos son de carga obligatoria, excepto subir un archivo de comprobante de envío. Complete todos los campos y luego presione el botón "Confirmar".

**CÓMO ENTREGASTE EL PAQUETE**

×

¿Qué medio usaste?

¿A qué dirección de e-mail lo enviaste?: \*  
shippingTesting@gmail.com

Fecha de envío: \*  
02/09/2020

Comprobante de envío:

3. Se cierra el cuadro de diálogo y vera el parte superior un cartel verde indicando que la evidencia de envío se subió correctamente.
4. En la parte inferior verá la información con la fecha que indicó que iba a enviar el paquete y un botón que le permite subir la evidencia de envío del paquete una vez que realice el envío.

Evidencia de envío subida con éxito

Reclamo #5036958149 13/09/2020 Ver en mercadolibre

Tu comprador quiere que le llegue el producto

Este reclamo no afecta tu reputación

Maria Alvarez Cuando lo mandas 13/09/2020 13:18:37

Tú El 15/09 13/09/2020 14:14:55

Obtener: boleita.jpg

Escribir a tu comprador...

¡Gracias! Le avisamos a tu comprador que ya enviste el paquete. En cuanto Mercado Libre confirme que está en camino, cierra el reclamo.

Forma de envío: Vía e-mail  
Fecha de envío: 02/09/2020  
Dirección de e-mail: shippingTesting@gmail.com

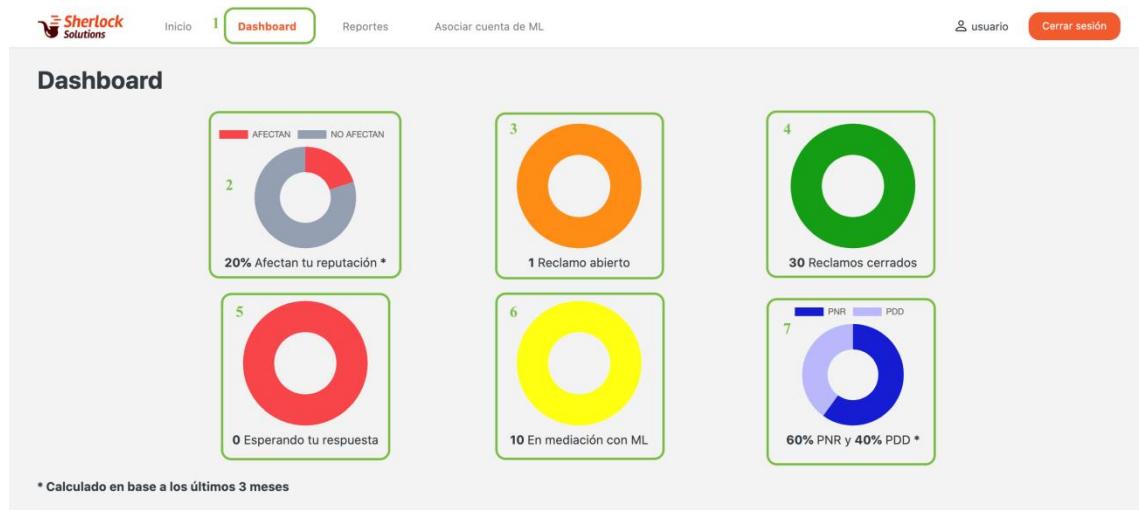
Detalle del reclamo  
Esperando a: Respuesta de comprador  
Etapa: Reclamo  
Estado: Abierto  
Compra: No recibida

Detalle de la venta  
Id de la orden: #4035683077  
Fecha: 13/09/2020  
Item:  
 Lámpara Led  
1 x \$10  
https://articulo.mercadolibre.com.uy/MLU-471291865-lámpara-led-...JM

Detalle del comprador

## Ver *dashboard*:

En esta pantalla podrá visualizar algunos datos importantes sobre sus reclamos en tiempo real. A continuación, se especifica la información que puede observar.



1. Puede acceder a ver esta pantalla desde el menú de navegación en la opción “*Dashboard*”.
2. El porcentaje de reclamos que afectan su reputación en los últimos 3 meses.
3. Cantidad de reclamos que se encuentran abiertos.
4. Cantidad de reclamos que se encuentran cerrados.
5. Cantidad de reclamos que esperan por su respuesta.
6. Cantidad total de reclamos en mediación con ML.
7. Porcentaje de reclamos de tipo PNR (pagado y no recibido) y PDD (producto defectuoso) calculado en base a los últimos 3 meses.

En el caso que tenga un usuario con multi tiendas podrá ver la cantidad de reclamos en cada una de sus tiendas.

## Ver reportes:

En esta pantalla podrá visualizar el histórico de la información de su *dashboard*. A su vez podrá filtrar por un rango de fechas. A continuación, se especifica la información que puede observar.

The screenshot shows the Sherlock Solutions dashboard interface. At the top, there are navigation links: Inicio, Dashboard, Reportes (which is highlighted in green), Asociar cuenta de ML, and user-related links. Below the header, the title "Reportes" is displayed. Two input fields are present: "Desde:" with the value "03/09/2020" and "Hasta:" with the value "11/09/2020". To the right of these fields is a red button labeled "Actualizar". A large green box, labeled with the number 5, encloses a table with the following data:

Fecha	% Afectan tu reputación*	Abiertos	Cerrados	Esperando tu respuesta	En mediación con ML	% PNR*	% PDD*
03/09/2020	29	2	29	1	9	59	41
04/09/2020	29	2	29	1	9	59	41
05/09/2020	29	2	29	0	9	59	41
06/09/2020	32	2	30	0	9	57	43
07/09/2020	32	1	31	0	10	57	43
08/09/2020	32	1	31	0	10	57	43
10/09/2020	36	1	33	1	11	53	47
11/09/2020	23	0	30	0	10	58	42

1. Puede acceder a ver esta pantalla desde el menú de navegación en la opción "Reportes".
2. Si quiere filtrar por un rango de fechas, seleccione la fecha desde, la misma debe ser menor que la fecha hasta.
3. Si quiere filtrar por un rango de fechas, seleccione la fecha hasta, la misma debe ser mayor que la fecha desde.
4. Si quiere filtrar por un rango de fechas, luego de haber elegido la fecha desde y hasta presione el botón "Actualizar".
5. Puede ver la tabla con todos los datos en las fechas seleccionadas

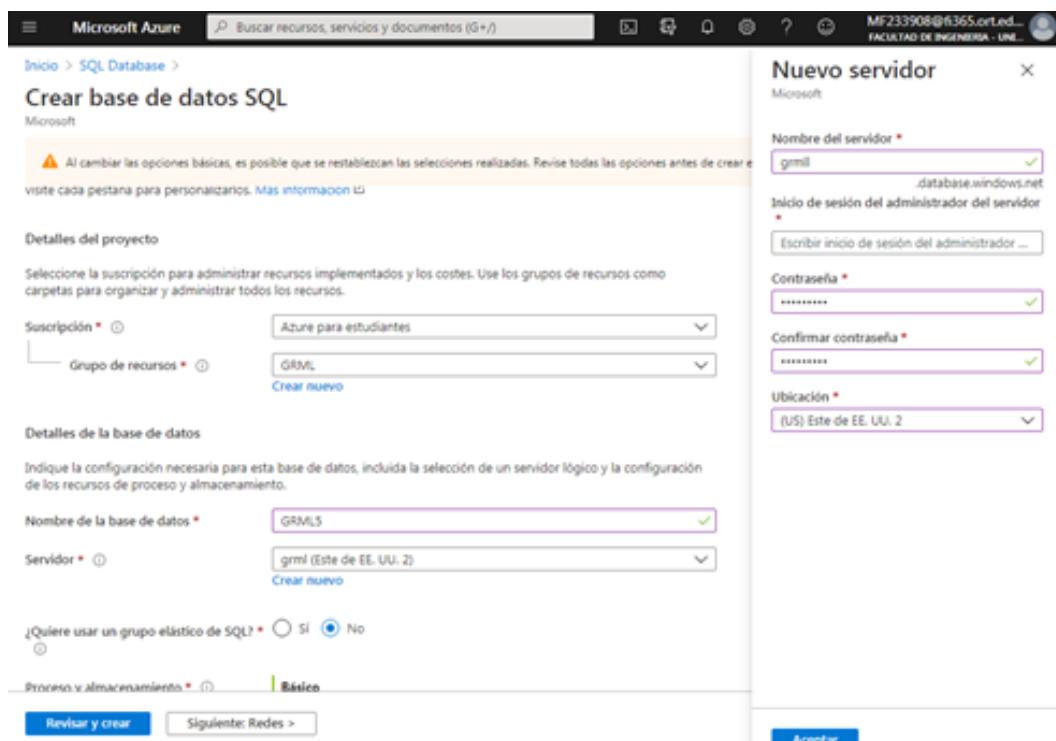
## Anexo 11 – Manual de *deploy*

### Primer *deploy* en producción:

A continuación, se presenta el proceso y los pasos seguidos para realizar el primer *deploy* de cada uno de los proyectos. El mismo se realizó durante el *sprint* 3. Luego, se consultaba periódicamente el estado de la cuenta brindada por la universidad para constatar que las configuraciones habían sido las adecuadas y que se contaba con créditos.

#### Base de datos:

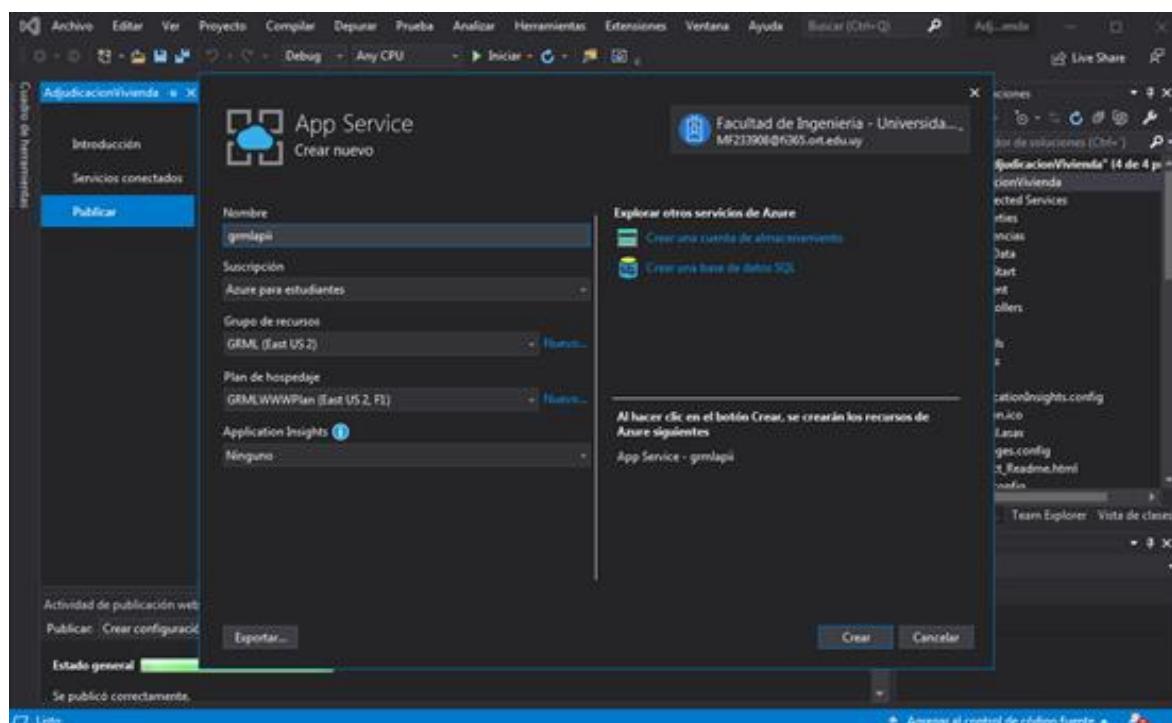
1. Se inició sesión en el portal de Azure con la cuenta brindada por la Universidad ORT Uruguay.
2. Se accedió al *dashboard* de SQL Database.
3. Se accedió a crear SQL data base, desde el botón “+ Agregar”.
4. Se abre una ventana emergente.
5. Se creó un nuevo recurso desde la opción de grupo de recursos “Crear Nuevo”, se eligió el nombre del recurso (GRML).
6. Se escribió el nombre de la base de datos (GRML).
7. Se creó un servidor desde el campo Servidor con la opción “Crear nuevo”. Se usó el nombre del servidor: grml.database.windows.net, se escribió el nombre de usuario y la contraseña, se eligió la ubicación del servidor: (US) Este de EE. UU 2.
8. Se seleccionó la opción no en el grupo elástico de SQL.
9. Se seleccionó configurar base de datos y se optó por la opción “Básica”.
10. Se finalizó la creación de la base de datos con la opción “Revisar y crear”.



### Backend / Frontend:

La primera publicación se realizó a través de la herramienta Visual Studio, donde se vinculó con la cuenta de Azure. Se repitieron los mismos pasos para publicar tanto el proyecto de *backend* (GRMLAPI) como para el de *frontend* (GRMLWWW).

1. En el Explorador de la solución, sobre el proyecto, se presionó el botón derecho, seleccionando la opción “Publicar”.
2. Luego en la ventana emergente, en el Target se seleccionó Azure > Azure App Service.
3. Se inició sesión en Azure con la cuenta brindada por la Universidad ORT Uruguay, en la esquina superior derecha de la ventana emergente.
4. Se seleccionó “Crear nuevo” en Servicio de aplicaciones de Azure.
5. Se configuró el servicio eligiendo el nombre, suscripción, grupo de recursos y plan de hospedaje.
6. Se seleccionó el plan gratuito.



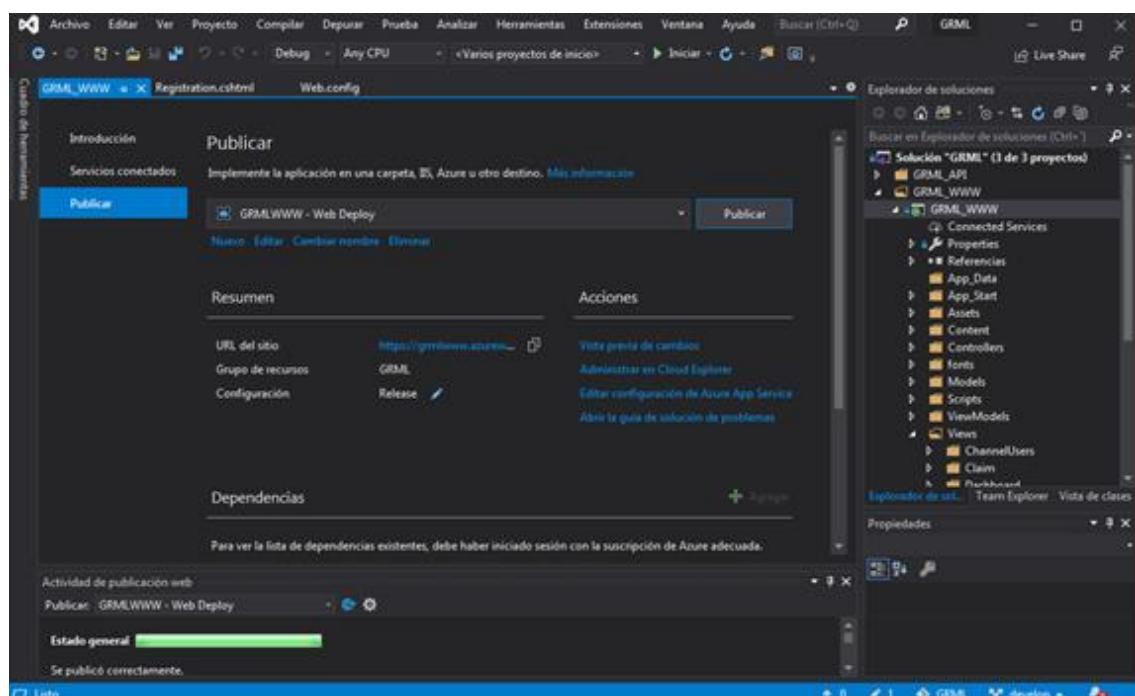
## ***Releases de deploy en producción:***

### **Base de datos:**

Durante el desarrollo del proyecto se trabajó localmente con la base de datos conectada en Azure, por lo que todo cambio que se ejecutaba era automáticamente implementado en la misma.

### **Backend / Frontend:**

Los *releases* se realizaron, tanto del proyecto de *backend* como el de *frontend*, al finalizar cada *sprint*. También se hicieron a través de Visual Studio, el mismo guarda el perfil de la cuenta creada en Azure y al seleccionar publicar sobre el proyecto, se abre una ventana emergente con el App Service correspondiente, y al seleccionar Publicar, se realiza la publicación del mismo.



## Anexo 12 – Datos de prueba para correctores.

Desde el *sprint 3* el sistema se encuentra alojado en la nube, en esta URL <https://grmlwww.azurewebsites.net>. El mismo ha sido actualizado en nuevos *releases* al finalizar cada *sprint*. Para prueba de los correctores, se crearon los siguientes usuarios.

USUARIO	CONTRASEÑA	ACLARACIONES
usuarioSinML	pass123	Esta cuenta de prueba no tiene aún un usuario de ML asociado.
usuarioTesting	123123	Esta cuenta tiene un usuario de ML asociado con reclamos abiertos y cerrados. En los abiertos se podrán accionar todas las funcionalidades que la herramienta brinda para la gestión del reclamo. En el mismo se encuentran ejemplos de reclamos, esperando evidencia de envío, queriendo la devolución del producto, devolución del dinero, con mensajes con y sin archivos adjuntos.
usuarioTesting2	pass123	Esta cuenta tiene también un usuario de ML con reclamos, donde se puede visualizar un reclamo en mediación.
usuarioSinReclamos	pass123	Esta cuenta tiene un usuario de ML asociado, pero éste no tiene ningún reclamo.

Es necesario aclarar que hubiese sido de mucho valor dar las credenciales de las cuentas asociadas de ML a los correctores, para que puedan ver en paralelo como se visualiza un reclamo en ML y en nuestro *frontend*. Pero esto no lo podemos realizar ya que ML pide segunda autenticación (con un SMS o llamada) para acceder a sus cuentas desde nuevos dispositivos.

Una prueba que si se puede ejecutar es la de generar un reclamo sobre algún artículo que se venda en ML por las cuentas de los usuarios de prueba. En la siguiente URL

<https://articulo.mercadolibre.com.uy/MLU-474929712-limon- JM?quantity=1> se encuentra la publicación de un artículo a la venta por el usuario de prueba usuarioTesting. Ese mismo artículo se puede comprar y posteriormente abrir un reclamo en el mismo, desde cualquier cuenta de ML, asimilando el costo de \$50 del mismo. Una vez creado el reclamo por ML, se podrá visualizar y gestionar desde el lado del vendedor en nuestra plataforma.

**URL para acceder a carpeta en Teams:**

Se puede acceder a ver el código fuente y documentación del proyecto a través de la siguiente URL: [https://teams.microsoft.com/\\_#/school/files/Py%2002%20-%20Alvarez%20-%20Fernandez%20-%20\(Abulafia\)?threadId=19%3A031168ee18ab40a29153637a9e1f1c60%40thread.tacv2&ctx=channel&context=0%2520-%2520ENTREGA%2520PROYECTO&rootfolder=%252Fsites%252FProyectosAPATI%2520Marzo2020-Py02-Alvarez-FernandezAbulafia%252FDocumentos%2520compartidos%252FPy02%2520-%2520Alvarez%2520-%2520Fernandez%2520\(Abulafia\)%252F0%2520-%2520ENTREGA%2520PROYECTO](https://teams.microsoft.com/_#/school/files/Py%2002%20-%20Alvarez%20-%20Fernandez%20-%20(Abulafia)?threadId=19%3A031168ee18ab40a29153637a9e1f1c60%40thread.tacv2&ctx=channel&context=0%2520-%2520ENTREGA%2520PROYECTO&rootfolder=%252Fsites%252FProyectosAPATI%2520Marzo2020-Py02-Alvarez-FernandezAbulafia%252FDocumentos%2520compartidos%252FPy02%2520-%2520Alvarez%2520-%2520Fernandez%2520(Abulafia)%252F0%2520-%2520ENTREGA%2520PROYECTO)

## Anexo 13 – Documentación de la API:

9/16/2020      Swagger UI

{...}      Select a definition      My API V1

### GRML\_API v1 OAS3

/swagger/v1/swagger.json

#### Claim

▼

**GET /Claim /GetAll**      Returns a complete list of user's claims ordered by status (opened - closed), purchase price and claim opened date.

**Parameters**      Try it out

Name	Description
userId integer (query)	<input type="text" value="userId"/>

**Responses**

Code	Description	Links

<https://grmlapi.azurewebsites.net/swagger/index.html>      1/23

Code	Description	Links
200	Returns an ordered list of all claims Sample response: [ { "id": "5025622237", "channelUser": { "channelUserId": "460680950", "nick": "UYUPDATE", "countryCode": null, "country": { "id": 1, "name": "Uruguay", "active": true, "code": "UY", "timeOffset": 0 }, "statusId": 1, "user": { "id": 1, "dateCreated": "2020-05-23T18:40:04", "name": "hola", "lastName": "apellido", "email": "apellido@gmail.com", "usr": "usuario", "psw": "Anto2019", "active": true, "countryId": 1, "userType": 0 }, "channelId": 1, "channel": 1, "order": { "id": 13, "dateCreated": "2020-06-14T22:35:06", "paymentDateCreated": "2020-06-14T22:35:11", "mediations": "5024851494", "feedbackPurchaseRating": "", "buyerNickname": "ALMA548186", "buyerId": 311950252, "itemTitle": "Lampara Led", "itemId": "MLU471291665", "storeId": "", "store": { "id": null, "storeName": null, "status": null, "active": false, "channelUser": null, "channel": 0, "storeURL": null }, "resource": "", "emailFrom": "", "amount": 50, "currency": "UYU", "status": "paid", "order_id": "2507492897", "userChannelId": "460680950", "channelId": 1, "itemImageURL": "http://mlu-s2-p.mlstatic.com/900684-MLU41512393075_042020-l.jpg", "itemURL": "https://articulo.mercadolibre.com.uy/MLU-471291665-lampara-led-_JM", "itemSKU": "", "quantity": 1 }, "type": "mediations", "stage": "claim", "status": "opened", "parentId": "", "resourceId": "2507492897", "resource": "order", "reasonId": null, "siteId": "MLU", "dateCreated": "2020-06-21T15:53:42", "lastUpdated": "2020-06-22T22:39:53", "amount": 50, "players": [ { "role": "Complaint", "type": "buyer", "userId": "311950252", "actions": [] }, { "role": "Respondent", "type": "seller", "userId": "460680950", "actions": [ { "action": "refund", "dueDate": null, "mandatory": false }, { "action": "send_message_to_complainant", "dueDate": "2020-06-25T02:48:59", "mandatory": true } ] }, { "role": "Mediator", "type": "", "userId": "", "actions": [] }, "resolution": null } ]	No links
400	If the request has invalid params, the user doesn't have linked his ML account or if there is a server error	No links

**GET** /Claim /GetByUserStore

Returns a list of user's claims filtered by store and ordered by status (opened - closed), purchase price and claim opened date.

Parameters		Try it out
Name	Description	
userId	integer (query)	<input type="text" value="userId"/>
storeId	string (query)	<input type="text" value="storeId"/>

**Responses**

## responses

Code	Description	Links
200	Returns a filtered and ordered list of claims Sample response: [ { "id": "5025622237", "channelUser": { "channelUserId": "460680950", "nick": "UYUPDATE", "countryCode": null, "country": { "id": 1, "name": "Uruguay", "active": true, "code": "UY", "timeOffset": 0 }, "statusId": 1, "user": { "id": 1, "dateCreated": "2020-05-23T18:40:04", "name": "holo", "lastname": "apellido", "email": "apellido@gmail.com", "usr": "usuario", "psw": "Anto2019", "active": true, "countryId": 1, "userType": 0 }, "channelId": 1, "channel": 1, "order": { "id": 13, "dateCreated": "2020-06-14T22:35:06", "paymentDateCreated": "2020-06-14T22:35:11", "mediations": "5024851494", "feedbackPurchaseRating": "", "buyerNickname": "ALMA548186", "buyerId": 311950252, "itemTitle": "Lampara Led", "itemId": "MLU471291665", "storeId": "", "store": { "id": null, "storeName": null, "status": null, "active": false, "channelUser": null, "channel": 0, "storeURL": null }, "resource": "", "emailFrom": "", "amount": 50, "currency": "UYU", "status": "paid", "order_Id": "2507492897", "userChannelId": "460680950", "channelId": 1, "itemImageURL": "http://mlu-s2-p.mlstatic.com/900684-MLU41512393075_042020-l.jpg", "itemURL": "https://articulo.mercadolibre.com.uy/MLU-471291665-lampara-led-_JM", "itemSKU": "", "quantity": 1 }, "type": "mediations", "stage": "claim", "status": "opened", "parentId": "", "resourceId": "2507492897", "resource": "order", "reasonId": null, "siteId": "MLU", "dateCreated": "2020-06-21T15:53:42", "lastUpdated": "2020-06-22T22:39:53", "amount": 50, "players": [ { "role": "Complaint", "type": "buyer", "userId": "311950252", "actions": [] }, { "role": "Respondent", "type": "seller", "userId": "460680950", "actions": [ { "action": "refund", "dueDate": null, "mandatory": false } ], "action": "send_message_to_complainant", "dueDate": "2020-06-25T02:48:59", "mandatory": true } ], { "role": "Mediator", "type": "", "userId": "", "actions": [] }, "resolution": null } ]	No links
400	If the request has invalid params, the user doesn't have linked his ML account or if there is a server error	No links

**GET** /Claim /GetByUserStoreAndFilter

Returns a list of user's claims filtered by store and other params (status, type or waiting for response of) ordered by status (opened - closed), purchase price and claim opened date.

**Parameters**

**Try it out**

Name	Description
userId	userId
storeId	storeId

Name	Description	Links
filter string (query)	filter	
<b>Responses</b>		
Code	Description	Links
200	Returns a filtered and ordered list of claims Sample response: [ { "id": "5025622237", "channelUser": { "channelUserId": "460680950", "nick": "UYUPDATE", "countryCode": null, "country": { "id": 1, "name": "Uruguay", "active": true, "code": "UY", "timeOffset": 0 }, "statusId": 1, "user": { "id": 1, "dateCreated": "2020-05-23T18:40:04", "name": "hola", "lastname": "apellido", "email": "apellido@gmail.com", "usr": "usuario", "psw": "Anto2019", "active": true, "countryId": 1, "userType": 0 }, "channelId": 1, "channel": { "id": 13, "order": { "id": 13, "dateCreated": "2020-06-14T22:35:06", "paymentDateCreated": "2020-06-14T22:35:11", "mediations": "5024851494", "feedbackPurchaseRating": "", "buyerNickname": "ALMA548186", "buyerId": 311950252, "itemTitle": "Lampara Led", "itemId": "MLU471291665", "storeId": "", "store": { "id": null, "storeName": null, "status": null, "active": false, "channelUser": null, "channel": 0, "storeURL": null }, "resource": "", "emailFrom": "", "amount": 50, "currency": "UYU", "status": "paid", "order_Id": "2507492897", "userChannelId": "460680950", "channelId": 1, "itemImageURL": "http://mlu-s2-p.mlstatic.com/900684-MLU41512393075_042020-l.jpg", "itemURL": "https://articulo.mercadolibre.com.uy/MLU-471291665-lampara-led-_JM", "itemSKU": "", "quantity": 1 } } } ] "type": "mediations", "stage": "claim", "status": "opened", "parentId": "", "resourceId": "2507492897", "resource": "order", "reasonId": null, "siteId": "MLU", "dateCreated": "2020-06-21T15:53:42", "lastUpdated": "2020-06-22T22:39:53", "amount": 50, "players": [ { "role": "Complaint", "type": "buyer", "userId": "311950252", "actions": [] }, { "role": "Respondent", "type": "seller", "userId": "460680950", "actions": [ { "action": "refund", "dueDate": null, "mandatory": false } ] }, { "role": "Mediator", "type": "", "userId": "", "actions": [] } ], "resolution": null } ] }	No links
400	If the request has invalid params, the user doesn't have linked his ML account or if there is a server error	No links

<b>GET</b>	/Claim/GetById	Returns the claim corresponded to the id provided.		
<b>Parameters</b>		<b>Try it out</b>		
<table border="1"> <thead> <tr> <th>Name</th><th>Description</th></tr> </thead> </table>		Name	Description	
Name	Description			

Name	Description	Links
claimId <small>string (query)</small>	claimId	No links
<b>Responses</b>		
Code	Description	Links
200	Returns the claim Sample response: { "id": "5025622237", "channelUser": { "channelUserId": "460680950", "nick": "UYUPDATE", "countryCode": null, "country": { "id": 1, "name": "Uruguay", "active": true, "code": "UY", "timeOffset": 0 }, "statusId": 1, "user": { "id": 1, "dateCreated": "2020-05-23T18:40:04", "name": "hola", "lastname": "apellido", "email": "apellido@gmail.com", "usr": "usuario", "psw": "Anto2019", "active": true, "countryId": 1, "userType": 0 }, "channelId": 1, "channel": { "id": 13, "dateCreated": "2020-06-14T22:35:06", "paymentDateCreated": "2020-06-14T22:35:11", "mediations": "5024851494", "feedbackPurchaseRating": "", "buyerNickname": "ALMA548186", "buyerId": 311950252, "itemTitle": "Lampara Led", "itemId": "MLU471291665", "storeId": "", "store": { "id": null, "storeName": null, "status": null, "active": false, "channelUser": null, "channel": 0, "storeURL": null }, "resource": "", "emailFrom": "", "amount": 50, "currency": "UYU", "status": "paid", "order_Id": "2507492897", "userChannelId": "460680950", "channelId": 1, "itemImageURL": "http://mlu-s2.p.mlstatic.com/900684-MLU41512393075_042020-I.jpg", "itemURL": "https://articulo.mercadolibre.com.uy/MLU-471291665-lampara-led-_JM", "itemSKU": "", "quantity": 1 }, "type": "mediations", "stage": "claim", "status": "opened", "parentId": "", "resourceId": "2507492897", "resource": "order", "reasonId": null, "siteId": "MLU", "dateCreated": "2020-06-21T15:53:42", "lastUpdated": "2020-06-22T22:39:53", "amount": 50, "players": [ { "role": "Complaint", "type": "buyer", "userId": "311950252", "actions": [] }, { "role": "Respondent", "type": "seller", "userId": "460680950", "actions": [ { "action": "refund", "dueDate": null, "mandatory": false }, { "action": "send_message_to_complainant", "dueDate": "2020-06-25T02:48:59", "mandatory": true } ] }, { "role": "Mediator", "type": "", "userId": "", "actions": [] }, "resolution": null }	No links
400	If there is not a claim with this id.	No links

<b>GET</b>	<b>/Claim/GetAttachment</b> Returns the URL to download the attached file.
<b>Parameters</b>	<b>Try it out</b>
Name	Description

9/16/2020

Swagger UI

Name	Description
claimId string (query)	<input type="text" value="claimId"/>
fileName string (query)	<input type="text" value="fileName"/>
userId integer (query)	<input type="text" value="userId"/>

### Responses

Code	Description	Links
200	Returns the URL	No links
400	There are no files for those attributes.	No links

## Dashboard



GET	/Dashboard/GetDashboard	Returns a dashboard.				
<b>Parameters</b>		<a href="#">Try it out</a>				
<table border="1"><thead><tr><th>Name</th><th>Description</th></tr></thead><tbody><tr><td>userId integer (query)</td><td><input type="text" value="userId"/></td></tr></tbody></table>		Name	Description	userId integer (query)	<input type="text" value="userId"/>	
Name	Description					
userId integer (query)	<input type="text" value="userId"/>					
<b>Responses</b>						
<table border="1"><thead><tr><th>Code</th><th>Description</th><th>Links</th></tr></thead></table>		Code	Description	Links		
Code	Description	Links				

<https://grmlapi.azurewebsites.net/swagger/index.html>

6/23

Code	Description	Links
200	Sample response: { "id": 0, "reputationClaimsPorcentage": 33, "openedClaims": 0, "waitingRespondentResponse": 0, "closedClaims": 30, "claimsInMediation": 0, "PNR": 20, "PDD": 80, "claimsByStores": [ { "store": { "id": 1, "storeName": "name", "status": "", "active": true, "channelUser": { "channelUserId": "460680950", "nick": "UYUPDATE", "countryCode": null, "country": { "id": 1, "name": "Uruguay", "active": true, "code": "UY", "timeOffset": 0 }, "statusId": 1, "user": { "id": 1, "dateCreated": "2020-05-23T18:40:04", "name": "hola", "lastname": "apellido", "email": "apellido@gmail.com", "usr": "usuario", "psw": "Anto2019", "active": true, "countryId": 1, "userType": 0 }, "channelId": 1, "channel": { "storeURL": "store url" } "claims": 34, } ], "channelUser": { "channelUserId": "460680950", "nick": "UYUPDATE", "countryCode": null, "country": { "id": 1, "name": "Uruguay", "active": true, "code": "UY", "timeOffset": 0 }, "statusId": 1, "user": { "id": 1, "dateCreated": "2020-05-23T18:40:04", "name": "hola", "lastname": "apellido", "email": "apellido@gmail.com", "usr": "usuario", "psw": "Anto2019", "active": true, "countryId": 1, "userType": 0 }, "channelId": 1, "DateCreated": "2020-08-19T00:45:02", } }	No links
400	If the request has invalid params, the user doesn't have linked his ML account or if there is a server error	No links

**GET    /Dashboard/GetAllByDates** Returns a complete list of user's Dashboard in the specified time period.

Parameters	Try it out
Name	Description
userId <small>integer (query)</small>	<input type="text" value="userId"/>
from <small>string (query)</small>	<input type="text" value="from"/>
to <small>string (query)</small>	<input type="text" value="to"/>

**Responses**

Code	Description	Links
------	-------------	-------

Code	Description	Links
200	Returns a list of all dashboards for this user in the given period of time  Sample response: [ { "id": 0, "reputationClaimsPercentage": 33, "openedClaims": 0, "waitingRespondentResponse": 0, "closedClaims": 30, "claimsInMediation": 0, "PNR": 20, "PDD": 80, "claimsByStores": [ { "store": { "id": 1, "storeName": "name", "status": "", "active": true, "channelUser": { "channelUserId": "460680950", "nick": "UYUPDATE", "countryCode": null, "country": { "id": 1, "name": "Uruguay", "active": true, "code": "UY", "timeOffset": 0 }, "statusId": 1, "user": { "id": 1, "dateCreated": "2020-05-23T18:40:04", "name": "hola", "lastname": "apellido", "email": "apellido@gmail.com", "usr": "usuario", "psw": "Anto2019", "active": true, "countryId": 1, "userType": 0 }, "channelId": 1, "channel": { "storeURL": "store url" } "claims": 34, } ], "channelUser": { "channelUserId": "460680950", "nick": "UYUPDATE", "countryCode": null, "country": { "id": 1, "name": "Uruguay", "active": true, "code": "UY", "timeOffset": 0 }, "statusId": 1, "user": { "id": 1, "dateCreated": "2020-05-23T18:40:04", "name": "hola", "lastname": "apellido", "email": "apellido@gmail.com", "usr": "usuario", "psw": "Anto2019", "active": true, "countryId": 1, "userType": 0 }, "channelId": 1, "DateCreated": "2020-08-19T00:45:02", } ] }	No links
400	If the request has invalid params, the user doesn't have linked his ML account or if there is a server error	No links

## Exception

GET	/Exception/GetByDates	Returns the exception logs from the given range of dates.					
Parameters		<a href="#">Try it out</a>					
<table> <thead> <tr> <th>Name</th><th>Description</th></tr> </thead> <tbody> <tr> <td>from string (query)</td><td><input type="text" value="from"/></td></tr> <tr> <td>to string (query)</td><td><input type="text" value="to"/></td></tr> </tbody> </table>		Name	Description	from string (query)	<input type="text" value="from"/>	to string (query)	<input type="text" value="to"/>
Name	Description						
from string (query)	<input type="text" value="from"/>						
to string (query)	<input type="text" value="to"/>						
Responses							
Code	Description	Links					

9/16/2020

Swagger UI

Code	Description	Links
200	Returns a list of exceptions Sample response: [ { "error": "MLServices GetOrder() - channelUser: 460680950 - - orderId: 4027849852 - ", "exception": " - Ex: Index was out of range. Must be non-negative and less than the size of the collection. (Parameter 'index') - ", "dateCreated": "2020-09-09T00:03:38" }, { "error": "MLServices GetOrder() - channelUser: 460680950 - - orderId: 4027849852 - ", "exception": " - Ex: Index was out of range. Must be non-negative and less than the size of the collection. (Parameter 'index') - ", "dateCreated": "2020-09-09T00:03:39" } ]	No links
400	If the request has invalid params, the user doesn't have linked his ML account or if there is a server error	No links

## Mediation

**POST** /Mediation/AskMediation It request ML for mediation.

Parameters		Try it out
Name	Description	
claimId	string (query)	claimId
userId	integer (query)	userId

**Responses**

Code	Description	Links
200		No links
400	If there is not a claim with this id or if there is a server error.	No links

## Message

**POST** /Message/SendMessage

<https://grmlapi.azurewebsites.net/swagger/index.html> 9/23

Parameters		Try it out
Name	Description	
claimId string (query)	claimId	
text string (query)	text	
attachment string (query)	attachment	
userId integer (query)	userId	
Responses		
Code	Description	Links
200		No links

**POST** /Message/CreateMessage Create a message with attachments or with out message.

---

**Parameters**

[Try it out](#)

Name	Description
claimId <small>string (query)</small>	<input type="text" value="claimId"/>
userId <small>integer (query)</small>	<input type="text" value="userId"/>
<b>Request body</b>	
<b>application/json</b>	
<b>Example Value</b>	
<pre>{   "id": 0,   "senderRole": "string",</pre>	

9/16/2020

Swagger UI

```

"receiverRole": "string",
"attachments": [
  {
    "id": 0,
    "filename": "string",
    "originalFilename": "string",
    "size": 0,
    "type": "string",
    "url": "string",
    "dateCreated": "2020-09-16T10:24:18.147Z"
  }
],
"stage": "string",
"dateCreated": "2020-09-16T10:24:18.147Z",
"content": "string",
"claim": {
  "id": "string",
  "channelUser": {
    "channelUserId": "string",
    "nick": "string",
    "countryCode": "string",
    "country": {
      "id": 0,
      "name": "string",
      "...."
    }
  }
}

```

### Responses

Code	Description	Links
200		No links
400	If the request has invalid params, the user doesn't have linked his ML account or if there is a server error	No links

POST /Message/AddAttachment Create the request to store attachments.

#### Parameters

[Try it out](#)

Name	Description
claimId string (query)	claimId
userId integer (query)	userId

#### Request body

[multipart/form-data](#)

#### Files

array

<https://grmlapi.azurewebsites.net/swagger/index.html>

11/23

**Responses**

Code	Description	Links
200	Returns the attachments file names Sample response: ["d15b7f92-7284-4a47-a4d4-91d491e130b4.jpeg", "d15b7f92-7284-4a47-a4d4-91d4937sh7.jpeg"]	No links
400	If the request has invalid params, the user doesn't have linked his ML account or if there is a server error	No links

**Notification**

<b>POST</b>	/Notification /Create	Receives ML notifications and takes the necessary actions to process and save the data.
-------------	--------------------------	---

**Parameters**
[Try it out](#)

No parameters

Request body

application/json

**Example Value** Schema

```
{
  "user_id": "string",
  "resource": "string",
  "topic": "string",
  "received": "string",
  "application_id": 0,
  "sent": "string",
  "attempts": 0
}
```

**Responses**

Code	Description	Links
200		No links



## Shipping

<b>POST</b> <code>/Shipping /SendShippingEvidence</code>	It send a shipping evidence when the user has already sent the product.						
<div style="display: flex; justify-content: space-between;"> <span><b>Parameters</b></span> <span><a href="#" style="border: 1px solid black; padding: 2px 10px; border-radius: 5px;">Try it out</a></span> </div> <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 10px;"> <thead> <tr> <th>Name</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>claimId <small>string (query)</small></td> <td><input type="text" value="claimId"/></td> </tr> <tr> <td>userId <small>integer (query)</small></td> <td><input type="text" value="userId"/></td> </tr> </tbody> </table> <div style="display: flex; justify-content: space-between; margin-top: 20px;"> <span>Request body</span> <span><a href="#" style="border: 1px solid black; padding: 2px 10px; border-radius: 5px;">application/json</a></span> </div> <div style="margin-top: 20px;"> <b>Example Value</b>   <a href="#">Schema</a> <pre>{   "id": 0,   "type": "string",   "shippingMethod": "string",   "shippingCompanyName": "string",   "trackingNumber": "string",   "dateShipped": "2020-09-16T10:24:18.155Z",   "dateDelivered": "2020-09-16T10:24:18.155Z",   "destinationAgency": "string",   "receiverName": "string",   "receiverId": "string",   "receiverEmail": "string",   "claimId": "string",   "fileNames": [     "string"   ],   "attachments": [     {       "id": 0,       "filename": "string",       "originalFilename": "string",       "size": 0,       "type": "string",       "url": "string",       "dateCreated": "2020-09-16T10:24:18.155Z",       "message": {         "id": 0,         "senderRole": "string",         "content": "string"       }     }   ] }</pre> </div>		Name	Description	claimId <small>string (query)</small>	<input type="text" value="claimId"/>	userId <small>integer (query)</small>	<input type="text" value="userId"/>
Name	Description						
claimId <small>string (query)</small>	<input type="text" value="claimId"/>						
userId <small>integer (query)</small>	<input type="text" value="userId"/>						
<b>Responses</b> <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 10px;"> <thead> <tr> <th>Code</th> <th>Description</th> <th>Links</th> </tr> </thead> <tbody> <tr> <td>200</td> <td></td> <td><a href="#" style="border: 1px solid black; padding: 2px 10px; border-radius: 5px;">No links</a></td> </tr> </tbody> </table>		Code	Description	Links	200		<a href="#" style="border: 1px solid black; padding: 2px 10px; border-radius: 5px;">No links</a>
Code	Description	Links					
200		<a href="#" style="border: 1px solid black; padding: 2px 10px; border-radius: 5px;">No links</a>					

9/16/2020

Swagger UI

Code	Description	Links
400	If the request has invalid params, the user doesn't have linked his ML account or if there is a server error	No links

POST	/Shipping /SendShippingIntention	It send a shipping intention date when the user has not sent the product yet.
<b>Parameters</b>		<b>Try it out</b>
Name	Description	
claimId string (query)	<input type="text" value="claimId"/>	
userId integer (query)	<input type="text" value="userId"/>	
Request body		application/json
Example Value	Schema	
<pre>{   "id": 0,   "handlingDate": "2020-09-16T10:24:18.156Z",   "type": "string",   "claimId": "string" }</pre>		
<b>Responses</b>		
Code	Description	Links
200		No links
400	If the request has invalid params, the user doesn't have linked his ML account or if there is a server error	No links

## Store



GET    /Store/GetAll    Returns a complete list of user's stores.

<https://grmlapi.azurewebsites.net/swagger/index.html>

14/23

9/16/2020

Swagger UI

**Parameters**

Name	Description
userId integer (query)	<input type="text" value="userId"/>

**Responses**

Code	Description	Links
200	Returns a list of stores Sample response: [{ "Id": "1", "StoreName": "Store name", "Status": "active", "Active": true, "ChannelUser": ChannelUser, "Channel": 1, "StoreURL": "Store URL", }]	No links
400	If the request has invalid params, the user doesn't have linked his ML account or if there is a server error	No links

## UserChannel

**GET** /UserChannel /Authenticate Set the Auth code: associates the user with the channelUser and fetch ML user information.

**Parameters**

Name	Description
code string (query)	<input type="text" value="code"/>
userId integer (query)	<input type="text" value="userId"/>

**Responses**

Code	Description	Links
------	-------------	-------

<https://grmlapi.azurewebsites.net/swagger/index.html>

15/23

Code	Description	Links
200	Returns the ChannelUser Sample response: { "ChannelUserId": "1", "Nick": "User nick", "Country": Country, "StatusId": 1, "User": User, "ChannelId": 1, }	No links
400	If the request has invalid params	No links

## Users

**POST** /Users/Create Creates a user.

Parameters
Try it out

No parameters

Request body application/json

Example Value Schema

```
{
  "id": 0,
  "dateCreated": "2020-09-16T10:24:18.160Z",
  "name": "string",
  "lastname": "string",
  "email": "string",
  "usr": "string",
  "psw": "string",
  "active": true,
  "countryId": 0,
  "userType": 0
}
```

**Responses**

Code	Description	Links
201	Returns the newly created user Sample response: { "id": 1, "dateCreated": "2020-05-26T23:50:22.037Z", "name": "User name", "lastname": "User lastname", "email": "User email", "usr": "User", "psw": "User psw", "active": true, "countryId": 1, "userType": 1 }	No links
400	If the request has invalid params in body	No links

Code	Description	Links
404	If there is a server error	No links

**POST /Users/Login** Login an existing user.

Parameters		Try it out
Name	Description	
USR string (query)	USR	
PSW string (query)	PSW	

**Responses**

Code	Description	Links
200	Returns the user logged Sample response: { "id": 1, "dateCreated": "2020-05-26T23:50:22.037Z", "name": "User name", "lastname": "User lastname", "email": "User email", "usr": "User", "psw": "User psw", "active": true, "countryId": 1, "userType": 1 }	No links
400	If the request has invalid params	No links
403	Bad credentials	No links

**POST /Users/Logout** Log out a logged user.

Parameters		Try it out
Name	Description	
id integer (query)	id	

Responses		
Code	Description	Links
200	If the user is logged out	No links
400	If the request has invalid params	No links
404	If there is a server error	No links

Schemas	▼
<pre>Attachment {     id           integer(\$int32)     filename     string     nullable: true     originalFilename string     nullable: true     size          integer(\$int64)     type          string     nullable: true     url           string     nullable: true     dateCreated   string(\$date-time)     message       Message {...} }</pre>	
Country	<pre>{     id           integer(\$int32)     name         string     nullable: true     active        boolean     code          string     nullable: true     timeOffset    integer(\$int32) }</pre>

```
User {
    id           integer($int32)
    dateCreated string($date-time)
    name*        string
    lastname*    string
    email*       string
    usr*         string
    psw*         string
    active       boolean
    countryId*   integer($int32)
    userType     integer($int32)
}
```

```
ChannelUser {
    channelUserId  string
    nullable: true
    nick          string
    nullable: true
    countryCode   string
    nullable: true
    country       Country {...}
    statusId      integer($int32)
    user          User {...}
    channelId     integer($int32)
}
```

```
Store {
    id           string
    nullable: true
    storeName    string
    nullable: true
    status       string
    nullable: true
    active       boolean
    channelUser  ChannelUser {...}
    channel      integer($int32)
    storeURL    string
    nullable: true
}
```

```

Order  {
    id          integer($int32)
    dateCreated string($date-time)
    paymentDateCreated string($date-time)
    mediations   string
    nullable: true
    feedbackPurchaseRating string
    nullable: true
    buyerNickname  string
    nullable: true
    buyerId        integer($int64)
    buyerFirstName string
    nullable: true
    buyerLastName  string
    nullable: true
    itemTitle      string
    nullable: true
    itemId         string
    nullable: true
    storeId        string
    nullable: true
    store          Store  {...}
    resource       string
    nullable: true
    emailFrom     string
    nullable: true
    amount         number($double)
    currency       string
    nullable: true
    status         string
    nullable: true
    order_id       string
    nullable: true
    userChannelId string
    nullable: true
    channelId       integer($int32)
    itemImageURL  string
    nullable: true
    itemURL        string
    nullable: true
    itemSKU         string
    nullable: true
    quantity        integer($int32)
}

```

```

ShippingEvidence  {
    id          integer($int32)
    type        string
    nullable: true
    shippingMethod string
    nullable: true
    shippingCompanyName string
    nullable: true
    trackingNumber string
    nullable: true
    dateShipped  string($date-time)
    dateDelivered string($date-time)
    destinationAgency string
    nullable: true
    receiverName  string
    nullable: true
    receiverId    string
    nullable: true
    receiverEmail string
    nullable: true
    claimId       string
    nullable: true
    fileNames     [...]
    attachments   [...]
}

```

```
ShippingIntention {
    id           integer($int32)
    handlingDate string($date-time)
    type         string
    nullable: true
    claimId      string
    nullable: true
}
```

```
AvailableAction {
    action        string
    nullable: true
    dueDate       string($date-time)
    nullable: true
    mandatory     boolean
}
```

```
ClaimPlayers {
    role          string
    nullable: true
    type          string
    nullable: true
    userId        string
    nullable: true
    actions       [...]
}
```

```
ClaimResolution {
    reason        string
    nullable: true
    dateCreated   string($date-time)
    nullable: true
    closedBy      string
    nullable: true
    benefited     [...]
}
```

```
ClaimLabel {
    name          string
    nullable: true
    value         string
    nullable: true
    comments      string
    nullable: true
    admin_id      string
    nullable: true
    date_created  string($date-time)
}
```

```

ClaimExpectedResolution  {
    playerRole      string
                        nullable: true
    userId          string
                        nullable: true
    expectedResolution string
                        nullable: true
    dateCreated    string($date-time)
    lastUpdated     string($date-time)
    status          string
                        nullable: true
    claim           Claim   {...}
}

Claim  {
    id              string
                        nullable: true
    channelUser     ChannelUser   {...}
    channel         integer($int32)
    order           Order   {...}
    type            string
                        nullable: true
    stage           string
                        nullable: true
    status          string
                        nullable: true
    parentId        string
                        nullable: true
    resourceId      string
                        nullable: true
    resource        string
                        nullable: true
    reasonId        string
                        nullable: true
    siteId          string
                        nullable: true
    dateCreated    string($date-time)
    lastUpdated     string($date-time)
    amount          number($double)
    waitingResponseFrom string
                        nullable: true
    url             string
                        nullable: true
    se               ShippingEvidence   {...}
    si               ShippingIntention   {...}
    reputation      boolean
    players          [...]
    messages         [...]
    resolution       ClaimResolution   {...}
    labels           [...]
    expectedResolutions [...]
}

```

```
Message  {
    id          integer($int32)
    senderRole  string
    nullable: true
    receiverRole string
    nullable: true
    attachments  [...]
    stage       string
    nullable: true
    dateCreated string($date-time)
    content     string
    nullable: true
    claim        Claim  {...}
    fileNames   [...]
}

Notification  {
    user_id      string
    nullable: true
    channelId    string
    nullable: true
    readonly: true
    resource     string
    nullable: true
    topic        string
    nullable: true
    received     string
    nullable: true
    application_id integer($int64)
    sent         string
    nullable: true
    attempts    integer($int64)
}
```