Escuela de Ciencias de la Ingeniería
Ingeniería Civil en Computación

Proceso de desarrollo de un WMS minimalista con infraestructura mínima

Vicente Antonio González Saldivia
Profesor(a) guía: David Rivas Galdames

Memoria para optar al título de Ingeniero Civil en Computación

Rancagua, Chile
Enero, 2024

Agradecimientos

En primer lugar, agradezco a mi familia por estar ahí para apoyarme y hablar siempre que los necesité, además de por su colaboración con las tareas cotidianas que permitieron mantenerme con vida durante esta experiencia y las anteriores, los quiero muchísimo infinito mil y más. Aparte, quiero agradecerles por permitirme tener esta oportunidad, ni de lejos pensé que podría aprender tanto y conocer tantas cosas nuevas, estoy mucho más feliz y satisfecho de lo que lo estaba cuando empecé, así que en serio se los agradezco mucho. Además de a mi familia, quiero agradecer a mis amigos por soportarme, quererme y por dejarme ser parte de sus vidas, me han dado un gran apoyo para seguir, incluso cuando estaba más nublado. A todos ustedes, los quiero mucho, y espero que sigamos compartiendo el tiempo en los años que vienen. Me gustaría también agradecer a mis compañeros de carrera, tanto dentro como fuera del aula me han permitido mejorar en muchos aspectos y honestamente tengo muchas expectativas en varios de ustedes, así que espero que sigamos en contacto en el futuro.

Ahora quiero hacer algunos agradecimientos más centrados en lo logrado en este trabajo, quiero agradecer al profesor Rodrigo Delgado, fue uno de mis primeros compañeros de docencia, además de quién me entregó la mayoría de las herramientas que me llevaron a desear aprender más sobre el desarrollo de software, aparte de ser un muy buen amigo. Lo mismo puedo decir del profesor Waldo Gálvez, ha sido un compañero excelente impartiendo docencia, y el primero en sacudirme con sus altas expectativas en mí y mis compañeros en ese gran curso de 4 personas. Quiero agradecer también a los profesores Cristóbal Quiñinao, Alfonso Ehijo e Ignacio Bugueño, por mostrarme lo que es la ingeniería a un nivel técnico e incluso como estilo de vida, creo en serio en muchas de sus palabras, y espero poder desempeñarme siguiendo sus enseñanzas en el futuro. Gracias por su conocimiento compartido.

Saliendo de la universidad, quiero agradecer también a Héctor Villarroel por guiarme durante mi práctica con su notable honestidad. La experiencia que tuve en Synaptic me cambió la vida, por lo mismo agradezco también a Eduardo, Esteban, Jeannette, Matías, Rolando y Sebastián por su paciencia y apoyo en mi aprendizaje. Quiero agradecer a Esteban Arriagada, por distraerme cuando lo necesité, que no fueron pocas veces, te quiero bro. Agradezco también a Ricardo Castillo, Felipe Torres y en general a las comunidades de Proin Chile y JavaScript Chile, son lo máximo, me han enseñado muchísimo y espero seguir formando parte de sus comunidades en el futuro. Finalmente, quiero agradecer a Felipe Gómez, Hernán Moreno y Camila Rapu por ser parte primordial de este trabajo, ustedes me acompañaron día a día, literalmente, y me permitieron seguir avanzando, incluso en los peores días, gracias, aunque a veces no lo demuestre o sea medio pesado, los quiero mucho y les deseo lo mejor de lo mejor.

¡MUCHAS GRACIAS A TODOS, ESTE TRABAJO NO SERÍA POSIBLE SIN USTEDES!

Índice

Índice de Figuras

Resumen

En el mercado actual, los softwares WMS, o sistemas de administración de bodega, presentan una gran diversidad de requerimientos. Esta situación impulsa a las empresas, que lo necesitan, a contratar equipos de desarrollo con la intención de crear un sistema a medida. Por otro lado, las soluciones preestablecidas disponibles en el mercado suelen incluir muchos servicios, lo que suele resultar en la compra de servicios no utilizados. Además, estas soluciones en general requieren de infraestructura adicional y capacitación del personal. Los desafíos comentados generan grandes costos de implementación, inversiones que pueden ser inviables para las operaciones de las MiPymes chilenas.

Mediante una metodología ágil adaptada, se propone diseñar un software basado en una arquitectura de microservicios. Esto pues, los microservicios bien orquestados pueden reducir los costos y aumentar la escalabilidad del sistema gracias al cloud computing. Además, esta decisión favorece el uso de dispositivos inteligentes para comunicarse con el sistema, lo que da pie a un software con un mínimo costo de infraestructura. Por otro lado, se buscará mitigar el costo de la capacitación del personal mediante el diseño en función de la experiencia de usuario e introducir la gamificación con el fin de simplificar la interacción con este.

En base a lo anterior, el objetivo de este trabajo es diseñar un software WMS minimalista, que contenga solo lo esencial, para reducir costos, pero que además priorice la calidad de la experiencia que brindará a sus usuarios en todos los eslabones de la cadena de suministros. En un contexto de MiPymes con al menos 5 personas en la operación logística, estas necesidades se resumen generalmente en la disponibilidad de un inventario en tiempo real, el control sobre la operación en la bodega, la economía de la empresa, y la satisfacción de la demanda de sus clientes de forma continua.

Palabras clave: Software WMS, Ingeniería de software, Experiencia de usuario, Logística, Arquitectura de microservicios

Introducción

Como se anticipa en el inciso anterior, este trabajo aborda la problemática de la ausencia de un software WMS que sea asequible para Micro, Pequeñas y Medianas Empresas (MiPymes) y que ofrezca las ventajas competitivas que un WMS de clase mundial puede brindar. Ventajas como optimizar los tiempos del proceso, aumentar el control sobre la operación y mejorar la calidad de servicio. Obviando que el costo de un software WMS adquirido por una empresa líder del mercado es excesivo para las MiPymes en general, existen soluciones minimalistas, pero algunas no incluyen el picking, otras no incluyen la recepción, y en general las interfaces no suelen ser amigables para un usuario nuevo, este último punto se debe principalmente a que el software no es actualizado continuamente.

Para aterrizar este problema, se contemplarán los costos de los WMS ofrecidos por las empresas que han sido revisadas en el Cuadrante Mágico del grupo de investigación Gartner para el estudio de mercado de los softwares WMS que superan ciertos umbrales de uso y ventas. Según la búsqueda de precios oficiales para las empresas revisadas en el trabajo de Tunstall et al. (2023), se tiene que el precio aproximado de compra de los WMS más baratos que cumplan todas las necesidades básicas de una operación de bodega están alrededor de los $4000 USD. Por otro lado, las empresas que ofrecen planes mensuales suelen indicar un precio por usuario, dónde los más baratos suelen cobrar aproximadamente $100 USD por cada uno.

Para resolver esta problemática se ha elegido solucionarlo mediante el diseño y desarrollo de un software de WMS, por lo tanto, en este escrito existe una dualidad marcada, lo cual suele ocurrir en proyectos asociados a la ingeniería de software. Por un lado, se tiene la teoría computacional asociada al proceso de desarrollo, lo que permite justificar las decisiones tomadas sobre cómo implementar la solución. Por otro lado, se observa la teoría tras del tema que se busca resolver para definir qué se debe desarrollar. En este caso, ese tema se ve representado por la cadena de suministro, o más precisamente, la logística, esto bajo el contexto de las operaciones realizadas por MiPymes. Antes de comenzar, se realizará una contextualización general de ambos temas, y cómo estos influyen sobre el problema a resolver.

Desde la perspectiva computacional este escrito trata del desarrollo de un Software como Servicio (SaaS). Esto permite explorar paradigmas de desarrollo conocidos, tales como las Arquitecturas Orientadas a Servicios (SOA), patrones arquitecturales como la Transferencia de Estado Representacional (REST) o los message brokers. También los modelos relacionales (SQL) y no relacionales de datos (NoSQL), la usabilidad en aplicaciones de dispositivos móviles y protocolos de autenticación para la seguridad del sistema. Para aplicar estos paradigmas en la futura implementación del sistema, se utilizarán lenguajes de programación y librerías de alto nivel que reduzcan la complejidad del desarrollo. Adicionalmente, se trabajará con métodos como la Infraestructura como Código (IaC) para desplegar el sistema con un Proveedor de

Servicios en la Nube (CSP) utilizando contenedores, de manera que el sistema sea agnóstico a los proveedores sobre los cuáles se despliegue.

Para un mayor detalle sobre las decisiones tomadas sobre el diseño, se puede revisar el Marco Teórico, en dónde se discuten las ventajas que tiene desarrollar una arquitectura de microservicios frente a otras arquitecturas, en particular por el hecho de implementar el sistema en la nube, también se explicará a mayor detalle la razón tras de ubicar el despliegue del sistema en la nube. En el Marco Metodológico se profundizará en las temáticas más orientadas a la experiencia de usuario y su importancia, dando origen además a una metodología ágil con la cual se realizará el diseño completo del sistema. Finalmente, en los Resultados del Proyecto se podrá ver aplicado todo el conocimiento y decisiones tomadas de forma anticipada en los incisos anteriores respecto al diseño de la arquitectura, del modelo de datos y de las vistas de las aplicaciones a desarrollar, además de realizar una introducción al proyecto open source que se origina del planteamiento de este trabajo.

Por el lado de la teoría de cadena de suministros se abordará el desarrollo del software WMS como una herramienta que conecte los componentes de un proceso logístico de bodega. Entendiendo que un proceso logístico en este contexto consta de 4 etapas, estas son, abastecimiento, producción, distribución, y post venta; el software WMS modela virtualmente estas 4 etapas como parte de la operación del almacén con la definición de logística de entrada y de salida, lo que conlleva un mayor control sobre el flujo de la operación. Esto permite la optimización del proceso en base a la toma de decisiones informadas de los encargados de la logística de la operación, información que obtendrían del cálculo periódico de indicadores de desempeño por parte del sistema. Esta optimización se puede llevar a distintos niveles, pero se hará énfasis en la reducción de tiempos, costos y errores en la cadena de suministros, de forma que se reduzcan las pérdidas, dando impulso al crecimiento de las MiPymes chilenas.

Durante el desarrollo de este trabajo, se organizó una visita a la bodega del complejo médico Clínica Dávila para interiorizar la teoría investigada sobre el proceso. Cabe mencionar que la operación completa no pertenece al sector de MiPymes, pero la bodega funcionaba con 10 a 15 personas, por lo tanto, se utilizará esta experiencia con fines ilustrativos sobre cómo podría ser una operación que se beneficie de este proyecto. En la visita a la bodega, se observó una metodología que consistía en mantener las repisas de picking con cajas que contenían de un solo tipo de producto a la vez. El sistema utilizará esta metodología internamente, puesto que es posible ponerla en práctica aún sin cajas, utilizando la forma de las repisas convenientemente. Otra observación valiosa de la visita fue notar cómo los trabajadores hacían sus labores y los requisitos que esas distintas labores tenían. De ello, se determinó que todas las labores podrían ser interpretadas como una "misión", y que mientras los trabajadores tuvieran asistencia para ubicar el lugar al que debían ir y la acción que debían realizar, podrían cumplir con sus objetivos exitosamente.

Objetivos

Objetivo General

- Diseñar un WMS minimalista que cubra las necesidades de una operación logística de bodega en el contexto de las MiPymes chilenas de forma oportuna, siendo estas la disponibilidad de un inventario actualizado, control sobre la operación en la bodega, así como sobre la economía del negocio, además de la satisfacción continua de sus clientes.

Objetivos Específicos

- Analizar el proceso logístico para detectar oportunidades de mejora a incluir en el sistema a desarrollar.
- Diseñar la interfaz gráfica de la aplicación para teléfonos inteligentes de forma que se maximice su usabilidad.
- Diseñar modelos de datos y arquitecturas de software para su posterior análisis y despliegue en la nube.
- Analizar posibles integraciones al sistema, así como el futuro crecimiento del software con el proyecto open source.

Objetivo Futuro

- Implementar y documentar el software a través de un proyecto open source para llevar el diseño del Objetivo General a la realidad, insinuando la masificación de los cambios en la línea de la digitalización de la logística en las MiPymes chilenas.

Alcances y Limitaciones

- En este escrito solo se diseñará el proyecto, principalmente porque el tiempo es insuficiente para implementar el sistema completo con solo un miembro en el equipo de desarrollo y sin un previo diseño del software.
- En este trabajo se ha concebido el concepto para una primera versión del producto, por ello se despreciará la implementación de tópicos ajenos al área de responsabilidades principales de un WMS, como lo es el transporte o el área de post venta.
- En la misma línea anterior, no se abordarán temas que sean muy específicos a la producción de alta envergadura, como lo son parcel manifesting, wave planning o 3PL billing pues son problemas mayores y además se escapan del contexto de MiPymes.
- De igual modo, tampoco se abordarán temas muy particulares dentro de los contextos abordados, como lo son operaciones con manejo inteligente de la temperatura o la humedad de los espacios de almacenamiento.

Marco Teórico

En este capítulo se desarrollarán en mayor profundidad los temas presentados en la Introducción, para ello se recurrió a la búsqueda de material bibliográfico de carácter académico para definir los conceptos más relevantes de cada tópico. Además, como se trata de un problema actual, se acompañarán estas definiciones con antecedentes contemporáneos. Adicionalmente, se expondrán los temas a la vez que se discute su aplicabilidad al proyecto.

Definición Teórica y Práctica del software WMS

En la tesis de León (2018), un Sistema de Gestión de Almacenes (WMS) "se define como un sistema de planificación de los recursos y de gestión de la información que, de una forma estructurada, satisface la demanda de necesidades de la gestión de un almacén". Además, en la misma tesis declara que "el objetivo principal de un WMS es controlar el movimiento y almacenamiento de artículos dentro y fuera de las operaciones y procesos del almacén", este último comentario se explica más a fondo en los incisos posteriores. Ahora bien, la definición anterior no deja de ser bastante global, por lo que para tener una noción más práctica de los componentes de un WMS, se revisará el Cuadrante Mágico (MQ) de WMS elaborado por Gartner. El MQ es un tipo de artículo elaborado por la compañía de investigaciones Gartner, y a grandes rasgos tiene por objetivo hacer un estudio de mercado sobre algún servicio, todo bajo una estructura que busca la imparcialidad desde la independencia que tiene la compañía.

Las capacidades principales de un WMS abarcan una amplia gama de funciones, según la investigación del equipo de Gartner, entre las más habituales del mercado se incluyen la recepción, la ubicación de productos, la gestión de inventario, el cycle counting, la intercalación de tareas, el wave planning, la asignación de pedidos, la preparación de pedidos, el reabastecimiento, el packing y el shipping. Adicionalmente, en el mismo estudio del equipo de Gartner se observan funcionalidades "extendidas", pero igualmente presentes en el mercado, entre estas se tiene la gestión de recursos humanos, slotting, gestión de patios, voice picking, parcel manifesting y 3PL billing, además de servicios que facilitan la agregación de valor, como el kitting o la manufactura ligera.

Entre estos últimos servicios también se podría considerar el cross docking, que en palabras de Zenteno (2017), "corresponde a un sistema de distribución donde las unidades logísticas son recibidas en una plataforma de alistamiento o centro de distribución y no son almacenadas, sino que son preparadas para ser enviadas de manera inmediata". Si bien, en el estudio del equipo de Gartner no se menciona esta funcionalidad de forma explícita, al menos los líderes del MQ, como Manhattan Associates, sí contemplan esta operación, e incluso como un recurso valioso al que una operación puede apuntar, pues tienen un costo relativamente bajo y un reducido impacto en la operación como tal.

Logística y la Cadena de Suministros

El software a desarrollar se encuentra fuertemente enmarcado en la logística, razón por la cuál se definirá este concepto a continuación. Como es natural en las infografías hechas por los proveedores del mercado de WMS, la logística se define como la Gestión de la Cadena de Suministros (SCM) de una operación. Lo mismo ocurre en la literatura académica, incluso en escritos que definen la logística y la SCM por separado, como ocurre en la tesis de Cristi (2003), donde se cierra la definición de logística con el siguiente comentario, "el concepto de Logística Integral se convierte en otra forma de denominar la SCM, puesto que se refiere a esa coordinación en integración de actividades a lo largo de la cadena de suministros". Pero aún resta por precisar sobre la cadena de suministro como tal.

El alcance que tiene la cadena de suministro se diversifica de autor en autor. Ahora bien, un punto en común para todos es que la cadena de suministros contempla al menos las actividades que ocurren allí, tales como los procesos que se mencionan en el próximo inciso. En el artículo de Qi, Huo, Wang y Yeung (2017) se define la cadena de suministro como una colección de empresas conectadas a través de personas, actividades, información y recursos. Colección que tiene por objetivo mover un producto o servicio desde su proveedor original hasta el cliente final. Si bien esta última definición es mucho más amplia, es la más adecuada para este trabajo, pues no solo indica claramente el objetivo del proceso como tal, sino que además lo denota como un conjunto de muchas piezas que dan cuerpo a un sistema complejo.

Para terminar con la definición de logística de este documento, se hará la diferencia entre logística de entrada y logística de salida. Este último enfoque es algo que ha tomado fuerza en la comunidad, pero aún más importante, es que este enfoque es el que toman las empresas líderes actuales del MQ de proveedores de WMS, tales como Manhattan Associates o Blue Yonder. La diferencia radica en lo que se explica en la tesis de Arias (2022), donde se indica que "la logística de entrada, que abarca el aprovisionamiento, la agilización y la recepción de mercancías que llegan a la organización, mientras que la logística de salida se encarga del almacenamiento, embalaje y transporte de mercancías que salen de la organización".

Definición de los Procesos en una Bodega

En un inciso anterior se introdujo el nombre de algunas de las operaciones más comunes abordadas por los proveedores de WMS del mercado actual, en este inciso se explicarán algunas de ellas al clasificarlas como parte de los procesos logísticos de una bodega. Como se define en la tesis de Mellado, la logística de bodega se puede dividir en el abastecimiento, la recepción, el almacenamiento y la distribución. Antes de comenzar a especificar sobre cada etapa, a continuación, se hará el uso del término unidad logística para

referirse a los productos, kits de productos o bienes en general que se almacenan durante la operación. Para el abastecimiento de nuevas unidades logísticas generalmente se utilizan otros softwares, más orientados a la planificación que a la producción en sí, como lo son los Planificadores de Recursos para Empresas (ERP). Es por esa razón que en el MQ de Gartner no se contempla ninguna operación de esta etapa.

Ahora bien, en la recepción si hay algunas tareas frecuentes en el estudio de mercado mencionado, principalmente se contempla la recepción y ubicación de productos en el almacén. La recepción en particular es un proceso que requiere confiabilidad, puesto que son los números que ingresan al sistema, por lo que un error en esta etapa es crítico para la operación. Por su parte, la ubicación en el mercado actual consiste en sistemas automáticos que definen el lugar óptimo para guardar una unidad logística en el almacén. Una estrategia complementaria a esto es el slotting, que como se define en el trabajo de Kofler et al. (2014), es un proceso de dos pasos, en el que primero se define una clase para la unidad logística y luego en función de esta clase se almacenan en un lugar del almacén. Por último, en el abastecimiento también se contempla el kitting y la manufactura ligera, que consisten en modificar levemente los productos recibidos para la venta de kits o nuevos productos respectivamente.

Por otro lado, en el almacenamiento se considera la gestión de inventario en general, que se resume a la planificación y control de la manipulación, almacenamiento y preservación de las unidades logísticas. Además, se contemplan actividades de mantenimiento con el mismo fin, como el cycle counting, que según la definición de Wijffels et al. (2016), corresponde al conteo de una selección de artículos a la vez, permitiendo contar los productos del almacén luego de un número conocido de iteraciones. Además, en esta etapa se siguen estrategias para optimizar los recursos de la operación. Para la aplicación de estas estrategias generalmente se requiere que el software presente una plataforma que facilite la metodología, entre algunas de estas estrategias se encuentra la intercalación de tareas, el wave planning, o el parcel manifesting. En el próximo capítulo se revisará la posibilidad de incorporar estas estrategias.

Finalmente, en la distribución se trabaja con el cumplimiento de los pedidos recibidos, desde la asignación de la orden a un operador, pasando por la preparación del pedido por parte del empleado, continuando en una etapa de packing opcional, hasta finalmente despachar el pedido fuera del almacén con destino a otro centro de distribución, o hasta el mismo comprador. En esta etapa destaca la preparación de pedido, conocida en el rubro como picking, que se trata de un problema ampliamente estudiado, en palabras de Dujmešić et al. (2018), esta etapa no solo es una de las que más tiempo consume en la operación logística, sino que también el costo operacional asociado a la preparación de pedido se puede elevar hasta el 55% del costo total. Son datos como este los que han dado pie al desarrollo de estrategias particulares como el voice picking o light picking.

Indicadores de Desempeño para la SCM

Para la inclusión de indicadores que permitan a cargos administrativos la toma de decisiones informadas sobre el proceso, se utilizará la definición de indicadores presentada en la tesis de Rodríguez (2022), "un indicador es una magnitud asociada a una actividad o proceso que permite realizar una comparación en un periodo definido, para así obtener información sobre una situación de este". Utilizando esta definición, Rodríguez dividió los indicadores en 4, el indicador de eficacia que refleja el grado de cumplimiento de los objetivos ignorando los costos en general. El indicador de eficiencia que implica la relación entre la eficacia lograda con un costo de tiempo y/o recursos. El indicador de calidad se relaciona con la rapidez y consistencia de respuesta a las necesidades de los usuarios. Y finalmente, el indicador de economía que alude a la capacidad de recursos financieros que es posible mover con un fin.

Las definiciones anteriores son muy compatibles con el concepto de este proyecto, esto pues se busca satisfacer las necesidades de información financiera y operacional de los procesos en la cadena de suministro para que los emprendedores puedan tomar decisiones informadas. Además, la intención del trabajo de Rodríguez es la propuesta de una herramienta financiera, y en dicho trabajo se describe que la importancia estratégica de disponer de reportes de estado sobre la operación se encuentra en que la transparencia de la información entre distintos sectores a través de sistemas de TI no sólo mejora la confiabilidad de los datos, sino que también la productividad en sectores como la mejora continua de la operación. Por último, la tesis de Arias introducida en el inciso anterior, tiene por objetivo la definición de indicadores que permitan evaluar el desempeño de la SCM a un nivel operacional, por lo que se podrán revisar dichos indicadores a modo de base de la selección.

Desarrollo Ágil Orientado a los Usuarios

Este problema, así como muchos, tiene sus bases en las personas. Es por esta razón, que a la hora de comenzar un desarrollo de software resulta conveniente tener un enfoque orientado a la resolución de problemas que las personas involucradas tienen. Ahora bien, es relevante también ajustar los requerimientos que se obtengan a la principal limitación de este trabajo, el tiempo. Es por esto que para combinar ambos intereses se pueden utilizar metodologías ágiles orientadas a la Experiencia de Usuario (UX). Las metodologías ágiles tienen su origen con el Manifiesto Ágil, el cual da cuenta de medidas que buscan reducir la burocracia del proceso con el objetivo de entregar el software con antelación, permitiendo una mayor flexibilidad en el desarrollo.

Según los resultados de Version One (2022), el 87% de las organizaciones participantes utilizan Scrum, que es una de las metodologías ágiles más exitosas a lo largo del tiempo. Como esta metodología tiene tanto éxito en equipos de clase mundial, resulta valioso analizarla para

este trabajo, el problema es que, así como la gran mayoría de metodologías ágiles, Scrum está pensado precisamente para desarrollarse con un equipo. Aun así, hay algunas componentes que se pueden extrapolar al formato de este trabajo, tomando la definición en la tesis de Cho (2010), Scrum se compone de 3 principales partes, el Control del Proceso Empírico (EPC), el framework y el workflow, teniendo en cuenta que el principal objetivo de los dos últimos es la coordinación del equipo, se revisará más a fondo el EPC.

Según Schwaber (1997), cofundador de Scrum, el EPC depende de 3 características, la visibilidad, que se refiere a que el proceso debe estar disponible en su totalidad para cada participante del equipo. La inspección, que alude a la revisión continua del proceso para mantener los objetivos siempre como prioridad. Y finalmente la adaptación, la cual implica que en base a la inspección se puedan tomar decisiones en el momento. Esta última característica se manifiesta también bajo la perspectiva de que Scrum es un método iterativo. Para el desarrollo de este proyecto, estas 3 características son aplicables y cooperan en el objetivo de definir la metodología con la que desarrollar el software WMS. Sin embargo, hay un punto que el equipo de IBM da a conocer en su artículo de metodologías de desarrollo, y es que es de suma importancia considerar la reproducibilidad del método, de manera que se reduzca la incertidumbre sobre la finalización del proyecto.

La forma en que Lucena et al. (2016) en IBM abordan este desafío, es mediante la introducción de estándares sobre la metodología que abordaron en su escrito, el Design Thinking (DT). Estos estándares corresponden a una forma de tomar los requerimientos, que consiste en responder a quién quiere qué y cómo se puede resolver. Un rol adicional al de los miembros del equipo de desarrollo, que corresponde a un potencial usuario y/o experto en el área, de forma que pueda ofrecer su punto de vista sobre las soluciones. Finalmente, en conjunto con este nuevo rol se celebran reuniones con diversos objetivos, por ejemplo, revisar los prototipos alcanzados, o la misma confirmación de los requerimientos ya tomados. Este estándar se revisará más a fondo en el Marco Metodológico.

Usabilidad para la Experiencia de Usuario

"Considerar la experiencia de usuario es una tarea imprescindible en todo proyecto digital", de esta forma comienza Morales (2023) su artículo y, como se podrá intuir del inciso anterior, en este proyecto se trabaja con la misma ideología. Como mencionan Bevan et al. (2015) en su revisión de la ISO 9241-11, la investigación en la Experiencia del Usuario (UX) se centra en medir las preferencias, las percepciones, las emociones y las respuestas físicas y psicológicas de los usuarios, las que pueden ocurrir antes, durante y después de la navegación del sistema. Esta definición es bastante cercana a lo que para efectos de este proyecto se considera como UX, puesto que contempla la evaluación de todo el proceso.

El concepto de usabilidad permite evaluar una interfaz con el fin de mejorar la UX desde un punto de vista principalmente operacional, lo que se presta para una visión más práctica de cómo analizar la UX en el proyecto. La usabilidad generalmente se trabaja en conjunto con otras características deseables en el producto final, según se menciona en el artículo de Rahmat et al. (2017), en dispositivos móviles estas características se pueden resumir en la interactividad y aprendibilidad, en la transparencia de permisos, la inmediatez del flujo principal, la completitud de las notificaciones, la flexibilidad, la accesibilidad en gestos, el diseño responsive, además de la seguridad de los datos y la confiabilidad para el usuario. Todos estos conceptos se revisarán y definirán individualmente en el Marco Metodológico.

Arquitecturas Orientadas a Servicios

Según Papazoglou y Van Den Heuvel (2006), una Arquitectura Orientada a Servicios (SOA) provee lineamientos, principios y técnicas que permiten la reorganización y despliegue de sus componentes para soportar y habilitar la toma de decisiones sobre el negocio a un nivel que sea competitivo en el mercado. Esta definición se puede interpretar como el diseño de una solución que funciona ágilmente y complementa las decisiones de negocio del producto mismo en base a la flexibilidad de la arquitectura. Como comentan Denisov y Sorokin (2021) en su artículo, habitualmente para implementar una SOA, se revisan dos acercamientos, la utilización de web services o microservicios como componentes principales. Estas implementaciones se complementan con otros patrones arquitecturales que beneficien la lógica de negocio, como lo pueden ser las APIs RESTful por ejemplo.

Como se comentó en la introducción, este proyecto utiliza una arquitectura de microservicios, la razón de que se prefiera esta por sobre la basada en web services es principalmente por la influencia de Baresi y Garriga (2020), pues en su artículo comentan que los microservicios a diferencia de los web services se basan en la comunicación sobre protocolos de red livianos y en la modularización de sus componentes, características que permiten el despliegue independiente de cada componente, lo que da pie a múltiples beneficios como la automatización y agilidad en el desarrollo. Además, los microservicios tienen la heterogeneidad de tecnología y resiliencia que comenta Newman (2021) en su libro, características que permiten libertad a los ingenieros de elegir la herramienta adecuada para el problema a abordar con cada requerimiento y seguridad de que el sistema no se vea comprometido por completo con el fallo de alguna de las implementaciones desplegadas.

Cabe mencionar que sobre el uso de microservicios se pueden implementar fácilmente las Interfaces de Programación de Aplicaciones (API) basadas en la Transferencia de Estado Representacional (REST), que en palabras de Laranjeiro et al. (2021) corresponde a la comunicación, por medio de mensajes en el protocolo HTTP, entre los componentes a través

del uso de Identificadores Uniformes de Recursos (URI) para el traspaso de información, generalmente bajo la Notación de Objetos de JavaScript (JSON). La simpleza y robustez de este patrón de diseño lo hace particularmente flexible para el desarrollo de microservicios simples que satisfagan los requerimientos rápidamente. Además, el uso de estándares que son agnósticos a los distintos lenguajes de programación hace que este patrón de diseño sea muy cómodo para trabajos colaborativos, como es el caso de un proyecto open source.

Servicios y Bases Datos en la Nube

No se podría comenzar a ahondar sobre el desarrollo de este proyecto sin mencionar las plataformas sobre las que se encontraría desplegado. Bajo la perspectiva de generar una SOA utilizando microservicios, es natural contemplar la idea de utilizar proveedores de servicios en la nube, pues ambas tecnologías comparten la ambición de modularizar el sistema y fomentar la flexibilidad. Similar a la estrategia del inciso sobre softwares WMS, se revisarán las opciones a través del estudio del MQ de Gartner pero de Proveedores de Servicios en la Nube (CSP). En principio, los líderes son claros y lo han sido durante años, se trata de Amazon, Microsoft y Google, Wright et al. (2023) denotan a Amazon como el líder indiscutible de los servicios en general. De igual forma, en el MQ sobre gestión de contenedores, que corresponde a, posiblemente, la pieza más importante de la arquitectura, Smith et al. (2023) destacan a Google por su desempeño con la tecnología.

Finalmente, cabe mencionar que dentro de los servicios en la nube ofrecidos por los proveedores también se contempla la contratación de servicios de bases de datos, y es buscando la modularidad deseada para el proyecto que se ha considerado una base de datos NoSQL, pues, como menciona Copeland (2013) en su libro, uno de sus principales beneficios es la escalabilidad horizontal. Ahora bien, es razonable notar que dicho libro ya no es actual bajo el ritmo de crecimiento de la tecnología, y si bien, la horizontalidad sigue siendo una característica emblemática de las bases de datos NoSQL, el panorama ha cambiado significativamente en estos últimos años. Como comenta Valduriez et al. (2021) en su artículo, la diferencia se ha vuelto difusa, pues las bases de datos NoSQL alcanzan altos niveles de cumplimiento de las características ACID, anteriormente exclusivas de SQL, y los modelos relacionales también han adquirido la escalabilidad horizontal gracias a variantes de NewSQL, como lo es Spanner de Google. De lo anterior se puede concluir que, en el estado actual, existe un alto grado de libertad para la elección entre bases de datos relacionales y no relacionales.

Deuda Técnica y la Importancia de las Buenas Prácticas

Para terminar el Marco Teórico, es relevante mencionar por qué este escrito contiene un alto énfasis en la organización del proyecto, incluso por sobre el desarrollo de este. La principal

razón son las condiciones bajo las cuales se desarrollará este producto, es decir, un desarrollo llevado a cabo por solo una persona en principio y con un límite de tiempo bastante ajustado de menos de 4 meses incluyendo la investigación acerca del problema como tal. Con ello en mente, la manera elegida para mitigar este problema es mediante un gran primer esfuerzo en proponer la estrategia a seguir, seguido por iteraciones cortas que concluyan en prototipos de cada uno de los módulos por desarrollar. Pero estas iteraciones cortas significan un riesgo no menor de que el proyecto acumule una gran "deuda técnica", provocando la pérdida de la flexibilidad que se buscaba originalmente para el producto.

En palabras de Tonin et al. (2017), la "deuda técnica" es una metáfora que se asocia a que, con la primera entrega rápida de un módulo, sobre esta se acumula una deuda de tiempo que se saldará en tiempos de mantenimiento, y es con el exceso de este "préstamo de tiempo" que el posterior mantenimiento o reestructuración del código puede ser inviable, creando un producto inflexible y propenso a riesgos de seguridad con el crecimiento de la tecnología. Pero esto va más allá, gracias al trabajo de Tornhill y Borg (2022) con su artículo Code Red, dónde con CodeScene analizaron 39 proyectos en desarrollo activo y alcanzaron resultados uniformes en el transcurso de un año, resultados que demuestran que la acumulación de esta deuda técnica puede significar hasta 9 veces más tiempo en el desarrollo con archivos afectados por la deuda. Además, en su conferencia asociada al mismo artículo, Tornhill estimó que al menos entre el 23% y 42% del tiempo invertido por los desarrolladores se consume en buscar soluciones a los problemas que la deuda técnica genera en su producto.

Con la perspectiva de buscar un desarrollo rápido, pero sin sacrificar la calidad del código, es que utilizarán herramientas automáticas, como la integración continua con tests unitarios y la estandarización de código mediante linters, todo esto para prevenir la deuda técnica en el menor tiempo posible. Lamentablemente, como comentan Wiese et al. (2022) en su trabajo, la investigación acerca de la deuda técnica se ha orientado a la reacción cuando se ha diagnosticado el problema en lugar de su prevención. De igual modo, en la investigación de literatura del mismo trabajo algunos de los tópicos más populares eran el uso de herramientas para el diagnóstico de code smells y la automatización con el fin de estandarizar. Por otro lado, el uso de estas herramientas coincide con las prácticas habituales en los proyectos open source de GitHub, por lo que el uso de estas "buenas prácticas" abre la posibilidad de un trabajo colaborativo a través de la plataforma de GitHub, por consiguiente, se revisarán las posibilidades de este tipo de iniciativas en un capítulo posterior.

Marco Metodológico

En este capítulo se utilizan los antecedentes adquiridos en la etapa anterior para la toma de decisiones iniciales sobre el proyecto, por lo tanto, en cada inciso se estará explicando el razonamiento tras de las decisiones tomadas. El propósito de estas primeras decisiones es cimentar la metodología del Objetivo Futuro, así como contextualizar la etapa de diseño comenzada en este escrito.

Selección de los Servicios Fundamentales para el Software

En base a los procesos nombrados en el Marco Teórico, se contempla que los procesos de compra o abastecimiento del almacén se trabajen con el sistema que la empresa disponga de antemano. Con ello en mente, el software WMS a implementar ofrece soporte integral en las etapas de recepción, almacenamiento y distribución. Adicionalmente a las funcionalidades directamente relacionadas a la operación del proceso logístico y sus respectivos datos de entrada y salida, el software trabajará con datos inferidos a partir del proceso, brindando soporte a los administrativos para realizar un análisis de la operación tanto a nivel operacional como financiero. Los indicadores disponibles se tratan en el próximo inciso, por lo que a continuación se revisará a mayor detalle los servicios a implementar en función de la operación objetivo. Para ello, se presentará la operación en las tres etapas mencionadas y se ilustrará el proceso con la operación del complejo médico Clínica Dávila con sede en Santiago de Chile.

Figura 1: Fotografía de racks de recepción en la bodega de la Clínica Dávila. Elaboración propia.

Comenzando por la recepción, el WMS dispondrá de un portal web sobre el cual el operador encargado de recibir insumos podrá indicar o confirmar las cantidades recibidas. Posteriormente, el cargamento se recibe para ser almacenado, con ese fin el sistema deberá brindar instrucciones acerca del lugar en el cual deben ser guardados los elementos recibidos. Dependiendo de si el proceso contempla la realización de cross docking, el sistema también deberá indicar si los insumos deben ser desempacados o no. Bajo el caso particular de una operación que contempla el cross docking, se recomienda mantener algunas ubicaciones del almacén destinadas a este propósito. Adicionalmente, estas ubicaciones dedicadas pueden servir como almacenamiento de reposición para el área de picking, como ocurre en la operación del complejo médico. Como se observa en la fotografía anterior, hay un área completa designada para la recepción sin desempaque, que sirve para delimitar las distintas etapas del recorrido que sigue el producto, manteniendo un mejor flujo de los operadores.

Figura 2: Fotografía de repisas de picking en la bodega de la Clínica Dávila. Elaboración propia.

Como parte del almacenamiento, el WMS organizará órdenes de reposición que tomen los insumos recibidos, ya sea desde el área de recepción o desde el área dedicada que se explicó anteriormente, y se envíen a las zonas de picking en dónde sean requeridos. Tomando como referencia la fotografía de las repisas de picking, el sistema se organizará mediante cajas, las cuales tendrán diversos tamaños dependiendo del contenido que vayan a albergar. Con esto en mente, el modelo virtual de la bodega tendrá esa representación, y como estas cajas deben ser contenidas por repisas, el sistema también deberá contar con un gestor de la disposición del almacén, de manera que se pueda adaptar a las distintas condiciones en las que se llevan las operaciones. Volviendo sobre los procesos, el sistema debe tener una calendarización

automática de las iteraciones del cycle counting para el mantenimiento del inventario de forma que se asegure el conteo completo.

Por otro lado, respecto a los procesos de optimización de recursos nombrados en el Marco Teórico, el análisis es simple, tanto el wave planning como el parcel manifesting no serán implementados, principalmente pues se escapan de la lógica de una operación en contexto de MiPymes. Sin embargo, la intercalación de tareas resulta más interesante de analizar, puesto que, si bien requiere que el personal pueda hacer otras tareas, esta estrategia les permite ser más productivos. La intercalación de tareas aprovecha los tiempos de regreso de un picking para llevar a cabo otras tareas, tales como buscar productos de reposición o contar inventario. Como de todas maneras el sistema creará estas tareas para que sean cumplidas en el transcurso del día, agregar este servicio no requiere un mayor esfuerzo adicional, lo que fomenta la idea de implementar esta estrategia. De igual modo, esta implementación se mantendrá como una configuración opcional, pues el funcionamiento de la estrategia depende de los operadores.

Por último, respecto a los procesos en distribución, como en el Marco Teórico se comentaba sobre el proceso de la orden de picking hasta el despacho, hace sentido que sea una sola interfaz la que se encargue de guiar al operador por el proceso completo, haciendo una recepción automática de la orden, orientando al operador a través de la bodega para que prepare el pedido, finalmente llevándolo a entregar el pedido a un área designada, dónde otro operador confirme la recepción del pedido. Adicionalmente en la misma interfaz se podrá integrar la estrategia anterior. Luego de que la orden de picking ha sido confirmada, otro operador deberá comenzar a hacer el packing de ser necesario para finalmente entregar el pedido ya completo al operador designado para despachar la orden, este paso al igual que el de compra y abastecimiento mantendrán las estrategias con las que la empresa ya esté trabajando.

Selección de Indicadores Financieros y Operacionales

Recordando las necesidades indicadas en el inciso anterior, es lógico comprender que el sistema debe responder a las necesidades del personal administrativo de la operación, tanto para que tengan una terminal centralizada dónde acceder a los indicadores financieros y operacionales del proceso logístico, como para que puedan efectuar cambios sobre el sistema, permitiendo actuar en el momento oportuno para mejorar el funcionamiento de la operación. Con esto en mente, se contempla un gestor de bodega, que cumpla las funciones de ajustar el layout de la bodega, observar y administrar los datos del sistema, acceder a los indicadores propuestos en este inciso y otras configuraciones sobre el sistema. Cabe mencionar que lo anterior debe estar sujeto a algún tipo de jerarquía en el personal logístico, puesto que, dependiendo de la operación, no todos los miembros del equipo deberían tener acceso a

absolutamente todas estas funciones. Ya descrito el contexto general, se pasará a revisar el trabajo de Arias para declarar los indicadores a considerar.

La tesis de Arias (2022), se contextualiza en los datos del Banco Central de Chile en el período de los años 2016 hasta el 2020, considerando solo las actividades económicas con mayor aporte al PIB y que tengan una cadena de suministros integrada en el proceso. Las actividades seleccionadas fueron industria manufacturera, minería y comercio al por mayor y al por menor, de estas no se considerará el análisis de minería puesto que es bastante más específico y los indicadores de la tesis así lo demuestran. Adicionalmente, en la tesis en revisión existen tablas dedicadas para los indicadores de cada una de las actividades mencionadas, las tablas contemplan varias columnas que aluden al objetivo que tenía Arias con su trabajo, por ejemplo, la columna de "categoría del indicador", responde a una catalogación que hizo la tesista previamente. Las tablas por considerar se encuentran en el Anexo 1, donde estas han sido modificadas para mostrar qué indicadores se conservarán en la primera versión.

Para efectos de este trabajo, se podrán revisar los indicadores considerados en una tabla presentada en los Resultados Adicionales haciendo uso de la definición de indicador de Rodríguez. Por otro lado, cabe mencionar que todos los indicadores propuestos en la investigación de Arias se definen en función de una recopilación mensual. Como el marco de este trabajo se basa en incrementar la agilidad de una operación, ofreciendo flexibilidad y transparencia a través de un sistema de TI, se buscará integrar cada uno de esos indicadores en una lógica diaria, de forma que puedan ser calculados mientras la operación esté activa. Por supuesto, esto se deberá poder configurar, ya que el generar los reportes financiero-operacionales no es necesariamente despreciable en términos de costo computacional, lo que puede impactar en el costo económico del sistema completo.

Análisis de Usabilidad para la Optimización del Proceso

Antes de comenzar con el detalle de las características nombradas en el Marco Teórico, resta una aclaración importante, y esa es la importancia de la UX en los distintos puestos de trabajo del proceso logístico. En principio se tiene ya la experiencia de entregar sistemas portables a los operadores de procesos ejecutivos, como lo son quienes aceptan y construyen los pedidos de picking, o quienes realizan labores de mantenimiento o reposición. Pero un nuevo concepto que ha ido escalando al respecto es la introducción de componente lúdicas a estos sistemas, o gamificación del término en inglés. Como describen Bahr et al. (2022) en su estudio, la inclusión de sistemas con interfaces lúdicas produce beneficios en la eficiencia del proceso, como lo es incorporar modelos que inducen la competitividad activa entre el personal.

Ahora bien, este último beneficio es algo que quizás no sea del todo aplicable en operaciones a baja escala, pero incorporar sistemas con la capacidad de aumentar la

productividad, que mejoren el estado anímico de los trabajadores y a su vez permita simplificar la labor para reducir el número de errores es algo, cuanto menos, deseable para una operación. Incluso el líder del MQ de WMS en las últimas 14 ediciones, Manhattan Associates, contempla esta lógica como algo valioso, al punto de producir una colección de infografías que avalan la utilización de esta estrategia, colección que se puede revisar a mayor detalle en el Anexo 2. Y para finalizar la aclaración, hay muchas cosas que por medio de la virtualización del proceso se acercan a las oficinas de administrativos, pero en una operación en crecimiento, con un personal limitado, hay problemas y oportunidades que solo se pueden diagnosticar estando en el lugar de la operación. De la última observación se desprende que la portabilidad del sistema es una característica prioritaria, razón por la que se preferirán vistas en portarretrato.

Ahora se explicará conceptualmente cada una de las características de usabilidad mencionadas. La interactividad es una de las principales, los teléfonos inteligentes en la modernidad cuentan con un gran número de sensores, lo que da pie a que puedan inferir el contexto del usuario, llevando la interacción entre ambos a un nivel mucho más profundo, la interactividad mide exactamente eso, que tan simple y disfrutable es la interacción entre dispositivo y usuario. La aprendibilidad también es sumamente relevante, en particular pues esta contiene a su vez la familiaridad, que corresponde a qué tanto aprovecha la aplicación de la experiencia que tiene un usuario con ese tipo de aplicaciones. La aprendibilidad por sí sola alude a qué tan simple es que un usuario pueda utilizar y seguir los distintos flujos que la aplicación tiene para satisfacer la necesidad bajo la cual abrió la aplicación en primer lugar.

Similar a la aprendibilidad, la accesibilidad, en particular asociada a gestos, alude a que el usuario pueda seguir el flujo de la aplicación cómodamente, ahora bien, el enfoque es levemente distinto, la accesibilidad apunta a que las acciones que realiza el usuario sean naturales de hacer, o en su defecto permisivas, por ejemplo, si quisiera pasar página en un libro digital, es razonable pensar que pudiera hacerlo deslizando como se hace con un libro físico. Por su parte, la transparencia de permisos es técnicamente una necesidad en dispositivos móviles, esto por cómo funcionan los sistemas operativos de IOS y Android, pero no deja de ser importante, puesto que mejora la confiabilidad de la aplicación. La confiabilidad se puede observar en otros aspectos también, pero en general corresponde al acto de hacer saber al usuario que una acción que tiene consecuencias está por ocurrir, de forma que el proceso interno sea transparente cuando corresponda. Por otro lado, la seguridad alude a que el sistema aborde las potenciales vulnerabilidades cibernéticas de la aplicación como tal, así como también de la comunicación que la aplicación tiene a través de Internet.

La inmediatez alude a que la resolución de las necesidades del usuario por parte del sistema se lleve a cabo en el tiempo adecuado, e independientemente de si la resolución de una petición es instantánea el usuario debe recibir algún refuerzo que demuestre que su acción ha tenido un impacto en el sistema. En el contexto de notificaciones, la completitud se refiere a

que éstas apoyen el progreso del usuario a lo largo del flujo principal, como una especie de guía paso a paso. Sobre las notificaciones, vale la pena mencionar que las hay de varios tipos, el término no se utiliza como solo las notificaciones de tipo push. Sobre la misma línea de guiar al usuario, la flexibilidad se basa en que el sistema cuente con alternativas para que el usuario pueda resolver problemas mientras no se escapen de la lógica de su necesidad. Finalmente, el diseño responsive es algo inherente a la tecnología móvil, y consiste en que el sistema se adapte rápido y correctamente al entorno, si el dispositivo está en vertical y tiene una resolución específica, la aplicación debería verse tan correctamente como si estuviera en una orientación horizontal y con una resolución diferente.

Propuesta de la Metodología a Seguir Durante el Desarrollo

Como se comenzó a desarrollar en el Marco Teórico, se pretende seguir una metodología que se enmarque en el Manifiesto Ágil, haciendo uso de los principios propuestos como parte del EPC de Scrum, pero incrementando la reproducibilidad del proceso a través de las modificaciones que hizo el equipo de IBM sobre una metodología ágil como lo es el DT. A mayor detalle, las modificaciones sugeridas por Lucena et al. (2016), corresponden a una forma de tomar requerimientos conocida como hills, integrar a los usuarios expertos al desarrollo bajo el cargo de usuario sponsor y orquestar reuniones que operen como un punto de revisión del trabajo realizado denominadas playbacks. En esencia son definiciones similares a algunos de los artefactos ya presentes en las definiciones más prácticas de Scrum, pero con algunas leves diferencias que se discuten a continuación.

En términos generales las hills no tienen mayores diferencias con una historia de usuario, pero las hills tienen una característica fundamental que las hace perfectas para el desarrollo ágil, y es que son simples. Su definición consta solo de seguir un estándar de acuerdo con cómo están escritas, nada más. Una hill debe responder a tres propiedades, la primera es que debe tener un sujeto objetivo, es decir, quién tiene la necesidad o requerimiento que se está documentando. La segunda es el requerimiento como tal, es decir, qué es lo que se necesita por parte del sistema para satisfacer la necesidad de ese usuario. Y la tercera, responde a una intuición temprana de la manera en que se resolverá, pero no bajo un enfoque técnico, sino que debe responder a cómo el sistema será capaz de ofrecer el servicio que responde a la necesidad del usuario en cuestión.

El usuario sponsor es bastante diferente al product owner de Scrum, puesto que si bien ambos tienen el mismo objetivo de representar una especie de fiscalizador del cumplimiento de los requerimientos levantados en el proceso de desarrollo, el Product Owner es parte del equipo y por tanto parte de la empresa desarrolladora, el usuario sponsor en cambio, es un tercero que funciona como un experto en la materia del problema que la implementación de software busca

resolver, o en su defecto, un usuario que utilizará eventualmente ese software. El profesor David Rivas de la Universidad de O'Higgins actuará como usuario sponsor de este proyecto durante la duración del trabajo de título, para asegurar la definición y el cumplimiento de los requerimientos asociados a las necesidades de las MiPymes de un software de WMS.

Terminando de revisar las diferencias, las playbacks son equivalentes en objetivos a las reuniones de hitos presentes en Scrum, pero no contemplan una calendarización estricta, sino que se asocian al estado de desarrollo del proyecto, esto se debe a que esta metodología no funciona a base a sprints periódicos. Sobre este mismo punto, se puede observar que la única documentación generada a partir de la metodología son las hills y cualquier registro asociado a los playbacks, registros que podrán ser encontrados en el Anexo 3. Para finalizar, a continuación, se observa un diagrama elaborado en Figma que resume el desarrollo de un módulo, donde se comienza y termina con playbacks que aseguren el mutuo acuerdo entre el equipo de desarrollo y los Usuarios Sponsor, mientras que el desarrollo ocurre en medio con una estructura flexible y constante comunicación con los Usuarios Sponsor.

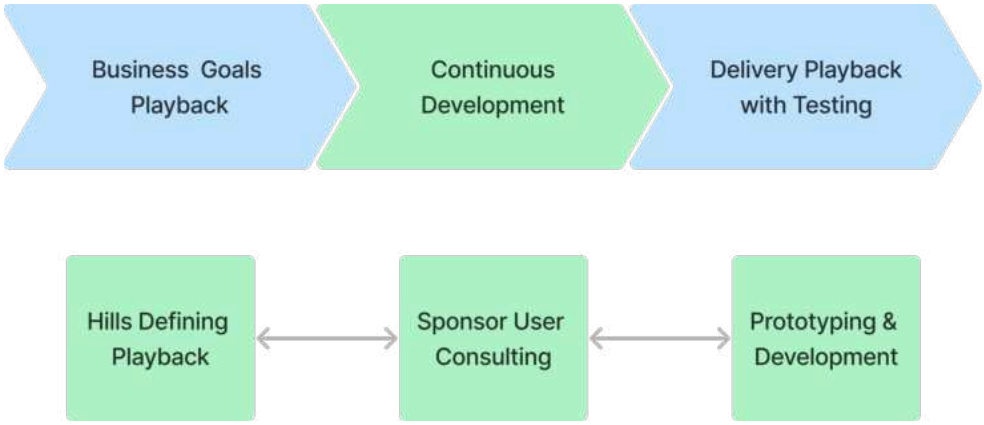

Figura 3: Diagrama sobre la aplicación de la metodología en un módulo. Elaboración propia.

Finalmente, cabe mencionar que para llevar a cabo las implementaciones se estarán utilizando tecnologías de integración continua y generación de código, de manera que los requerimientos puedan ser resueltos eficazmente. Además, se utilizarán herramientas que mejoren la consistencia del código, para que sea más sencilla la colaboración con otros desarrolladores del proyecto open source. Igualmente, para facilitar la comunicación entre desarrolladores, se hará uso de un tablero Kanban en la plataforma de GitHub Projects que tendrá registradas las hills a desarrollar y se dispondrá de plantillas para documentar los aportes a los distintos repositorios de código fuente. Para formar parte del equipo de desarrollo se requerirá de una cuenta de GitHub y realizar alguna contribución al proyecto como mínimo, los miembros actuales del equipo se reservan el derecho de aceptar nuevos miembros.

Resultados del Proyecto

A partir de las selecciones y comentarios realizados en el capítulo anterior, en este capítulo se definen las hills principales que representan la primera versión del software. Para mayores detalles acerca de la planificación y el estado de avance del proyecto, siempre se puede consultar en la ventana de proyectos de la organización en GitHub. Dicho esto, las hills en cuestión se observan a continuación:

- Hill Nº1: El empleado a cargo de recibir el abastecimiento de productos necesita verificar y confirmar la cantidad recibida comparándola con la cantidad comprada. Para ello se desarrollará una vista en la que se debe completar un formulario de confirmación, el cual se envía a los empleados encargados de logística. Además, se tendrá como prioridad la alta seguridad y el control de confirmaciones para reducir los errores.
- Hill Nº2: Un administrador o el empleado responsable de definir el layout del almacén necesita ingresar esa información al sistema para asignar los productos a sus ubicaciones. Para ello se desarrollará una vista sencilla para configurar el layout del almacén, este debe ser altamente configurable para las diversas operaciones y debe contar con confirmaciones, pues es una acción de alto impacto en la operación.
- Hill Nº3: El empleado de labores logísticas que monitorea la operación necesita revisar el rendimiento de la operación para optimizar el trabajo realizado. Para ello se desarrollará una vista de análisis de datos que muestra el rendimiento a lo largo del tiempo, guardando continuamente los cálculos realizados para los indicadores de desempeño financiero y operacional.
- Hill Nº4: El empleado que realiza los pedidos de picking necesita orientación de manera didáctica para completar sus tareas. Para ello se desarrollará una aplicación de teléfonos inteligentes que sea capaz de recibir tareas de picking y de mantenimiento. Durante el picking, dará instrucciones para ir de repisa en repisa, indicando que caja revisar y limitando el progreso con la necesidad de validar el producto por tomar.
- Hill Nº5: El empleado que realiza los pedidos de picking necesita recibir otras tareas entre cada pedido para mejorar su productividad y optimizar su tiempo. Para ello se desarrollará la aplicación antes mencionada que, después de que se confirme el pedido, recibirá tareas de mantenimiento como cycle counting, o movimientos en el almacén, tales como la reposición de las repisas de picking.
- Hill Nº6: El empleado en la zona de packing o shipping que recibe un pedido de picking necesita validar la completación del pedido y virtualizar su trabajo para contribuir a los datos recopilados por el sistema. Para ello se desarrollará una aplicación simple para validar el trabajo del empleado de picking y que disponga de los pedidos en camino de antemano, para así preparar materiales para el posterior envío de los productos.

- Hill Nº7: Un administrador o el empleado responsable del sistema necesita impactar directamente en las configuraciones. Para ello se desarrollará un panel de control que muestra las configuraciones actuales con widgets para realizar los cambios. Además de otro panel para la creación y gestión de usuarios que pueden tener diferentes permisos o niveles de acceso.

Ahora bien, las hills descritas corresponden a la planificación del proyecto hasta la primera versión. Para llevar a cabo estas hills primero se harán iteraciones sobre los conceptos introducidos en este trabajo para llegar a prototipos en base a los cuales comenzar a desarrollar el proyecto. En los incisos siguientes se comenta acerca de cada uno de los prototipos, y sus respectivos análisis que validen las decisiones tomadas mediante el uso de las definiciones entregadas en capítulos anteriores.

Arquitectura de Microservicios

Para definir la arquitectura del proyecto se hará uso de la notación del modelo C4, una razón de que se vaya a utilizar esta notación es porque corresponde a un conjunto de reglas bastante generales y simples. Según Brown (2017), el "C4" alude a las capas de contexto, contenedores, componentes y código. Un contexto corresponde a un sistema tecnológico involucrado en el proceso que se quiere representar. Las demás capas son las partes que se encuentran al interior de cada una de las demás capas de forma recursiva, esto quiere decir que el contexto se compone de contenedores, los contenedores de componentes y así sucesivamente. Otra razón, es que esta notación tiene herramientas para la creación de diagramas como código, en particular la herramienta Structurizr que utiliza un lenguaje de programación, permitiendo el desarrollo colaborativo de los diagramas para una toma de decisiones más horizontal en el desarrollo del proyecto, lo que beneficia el open source.

Luego, las definiciones de arquitecturas para cada una de las hills declaradas al comienzo de este capítulo se representarán individualmente mediante los diagramas elaborados con la herramienta Structurizr a lo largo de este inciso. Comenzando entonces por la vista de contextos en la Hill Nº1, el punto más relevante son las relaciones que hay. Como se observa en la imagen a continuación, un usuario llena un formulario en la vista designada, y luego de que la información sea recibida por el sistema, el módulo de recepción comenzará a distribuir estos datos mediante el uso de un message broker a los microservicios que consumen estos datos posteriormente. Un message broker, o intermediario de mensajes, es un patrón arquitectónico que permite la transferencia segura y balanceada de mensajes en una plataforma distribuida, como lo es la infraestructura en la nube que se utilizará en el proyecto. Los intermediarios de mensajes pueden ser representados por un contenedor, ya que existen implementaciones de este patrón previas a la masificación del cloud computing, pero en la

actualidad, este tipo de servicios son ofrecidos por los CSP, por ejemplo, en Google se tiene Pub/Sub, en Amazon está SNS y en Microsoft el Service Bus.

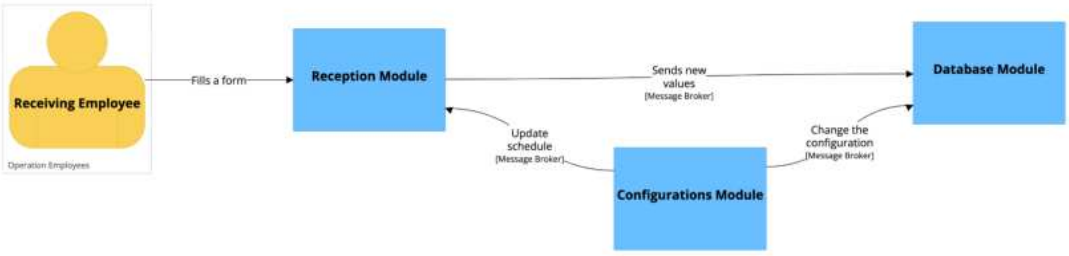

Figura 4: Diagrama de arquitectura sobre los contextos en el módulo de recepción. Elaboración propia.

Profundizando más en la vista anterior, se pueden revisar los contenedores definidos al interior del módulo de recepción. En las figuras a continuación se observan varios principios que serán reutilizados en otras partes de la arquitectura. El primero de ellos siendo la presencia de un paso intermedio entre el frontend y el backend, como el proyecto es una arquitectura en la nube, el sistema se encuentra abierto a Internet por defecto, lo que supone un gran riesgo de seguridad, por ello se configurarán una firewall para un control estándar del flujo de red, seguido de un sistema de autenticación como OAuth o el IAM del proveedor de nube, para terminar con un balanceador de carga que distribuya de forma equitativa las solicitudes de los clientes al resto del sistema. Lo anterior se representa con el contenedor de autenticación. El otro principio es la validación de esquema, se trata de un servicio inicial que se encarga de validar que la solicitud recibida sigue la estructura requerida para el posterior procesamiento.

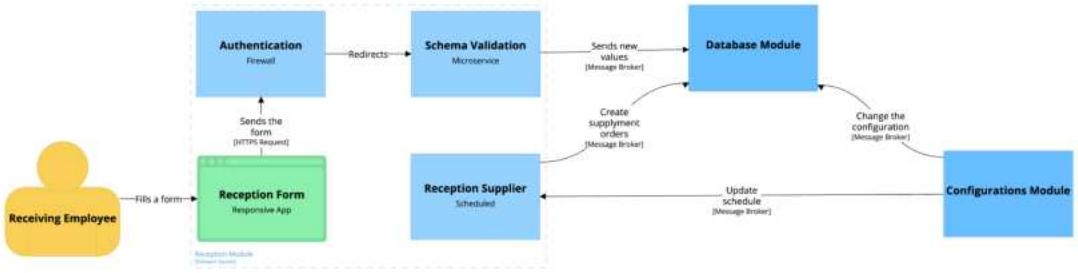

Figura 5: Diagrama de arquitectura sobre los contenedores en el módulo de recepción. Elaboración propia.

Continuando con los requerimientos del sistema, en la Hill Nº2 no hay una diferencia sustancial con la arquitectura presentada para el primer requerimiento, el cambio radica en que el sistema de creación y edición de layouts tiene un impacto de mayor nivel en la operación, lo que se observa en que este módulo crea nuevas misiones de movimiento para reorganizar el almacén de acuerdo con la construcción del administrador del layout. En la imagen a continuación se observa la reutilización de los principios comentados anteriormente, haciendo

referencia además a la posibilidad de utilizar el servicio de validación de esquema como un servicio enrutador para los demás servicios disponibles en el sistema, en este caso envía requerimientos tanto al servicio que se encarga de generar las misiones de movimiento, así como para un servicio del módulo de bases de datos asociado al almacenamiento del layout.

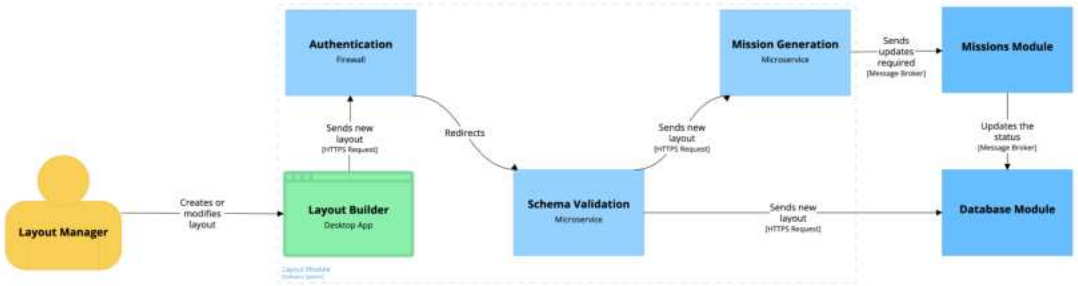

Figura 6: Diagrama de arquitectura sobre los contenedores en el módulo de layout. Elaboración propia.

Respecto a la Hill Nº3 del analista del área logística, la arquitectura se vuelve algo más compleja, y es que se requiere que los indicadores de desempeño sean calculados de forma periódica, lo que da pie a servicios que se encuentran sujetos a un itinerario. Este tipo de procesos es relativamente común en productos modernos, razón por la cual es un servicio ofrecido por distintos CSP. En el siguiente diagrama se muestra la lógica de un servicio que calcula los indicadores de forma periódica y almacena esos datos en una base de datos particular del módulo de analíticas, de forma que pueda ser consultado más tarde. Adicionalmente, se puede observar que la base de datos tiene un servicio intermediario, esta práctica se conoce informalmente como "independencia de datos", y supone la implementación de un sistema agnóstico a proveedores de almacenamiento para facilitar la migración, estandarizando y limitando las operaciones a una API RESTful.

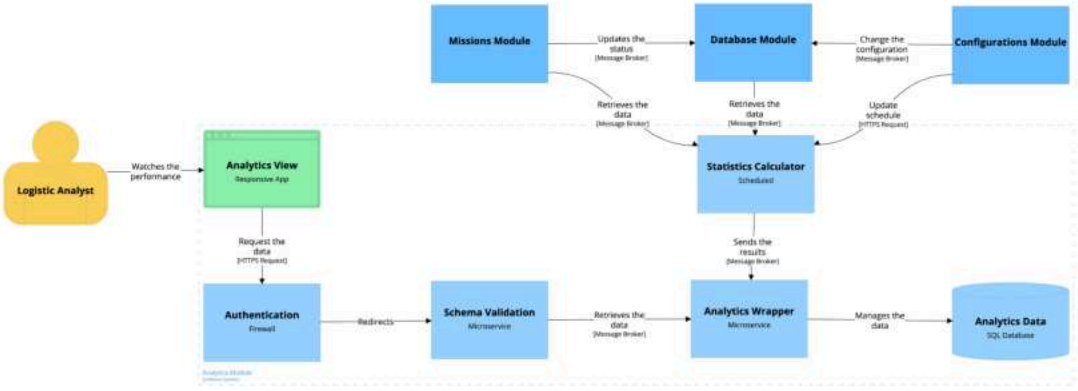

Figura 7: Diagrama de arquitectura sobre los contenedores en el módulo de analíticas. Elaboración propia.

Para las hills Nº4, Nº5 y Nº6 se reitera lo explicado hasta ahora, además representan un proceso conjunto, que consiste en el picking, el shipping y los procesos intermedios como lo es el packing y las tareas de mantenimiento que se solicitarán al empleado de picking en su camino de regreso. Por parte del cuarto requerimiento, la arquitectura a utilizar corresponde al módulo de picking, respecto al quinto requerimiento se ve representado en el módulo de Inventario, ya que allí se originan las tareas de mantenimiento para el intercalado de tareas. Finalmente, el sexto requerimiento consiste en el módulo de shipping, dónde el operador tendrá acceso a las órdenes de picking en curso para preparar el embalaje adecuado con relación al pedido, y dará su validación para el término de un proceso de picking y el comienzo de una próxima misión para el empleado de picking. A continuación, se ven las arquitecturas comentadas en sus respectivos diagramas.

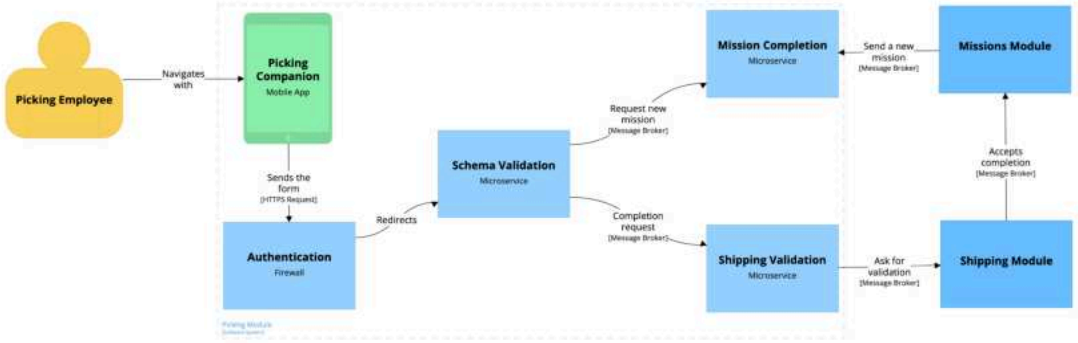

Figura 8: Diagrama de arquitectura sobre los contenedores en el módulo de picking. Elaboración propia.

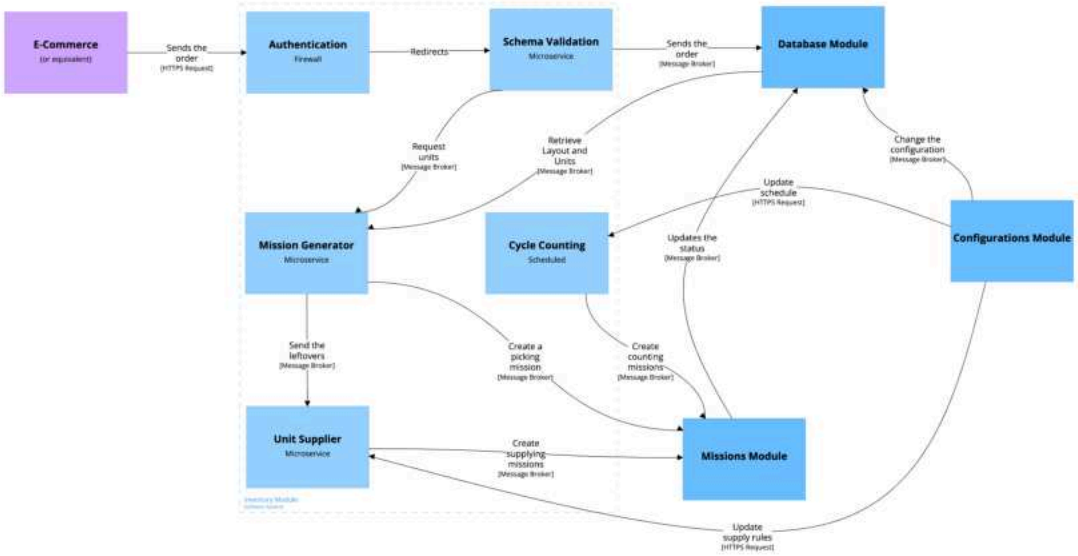

Figura 9: Diagrama de arquitectura sobre los contenedores en el módulo de inventario. Elaboración propia.

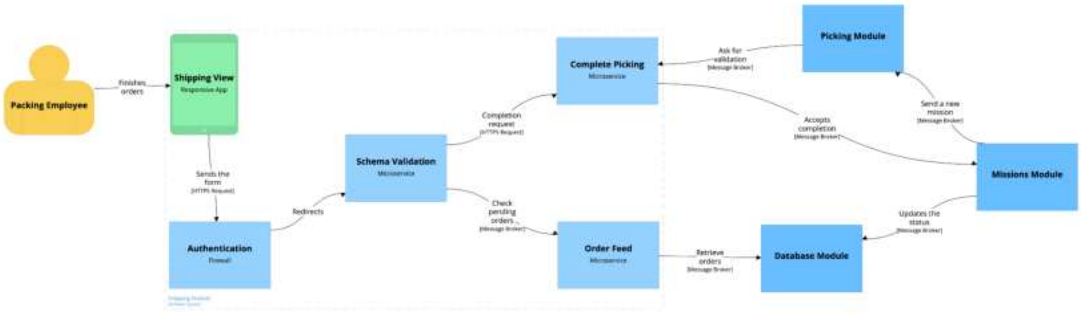

Figura 10: Diagrama de arquitectura sobre los contenedores en el módulo de shipping. Elaboración propia.

El último requerimiento consiste en el acceso a configurar el sistema mientras este se encuentra en funcionamiento, entre algunas de las configuraciones que ya han sido tomadas en cuenta, se encuentran las configuraciones respecto de los servicios que se gatillan mediante un itinerario, también, como se pudo ver en el diagrama del módulo de inventario, se ha considerado la modificación de los índices que denotan el límite antes de que se requiera una reposición de producto y también la agregación o eliminación de personal para trabajar en la operación. Además, como se observa en el siguiente diagrama, las configuraciones definidas por el administrador se mantienen almacenadas en el módulo de bases de datos, por lo que en caso de que alguno de los servicios actualizados fuera apagado, actualizado o se desactive a causa de un error, estos podrán recuperar la configuración luego de restablecerse.

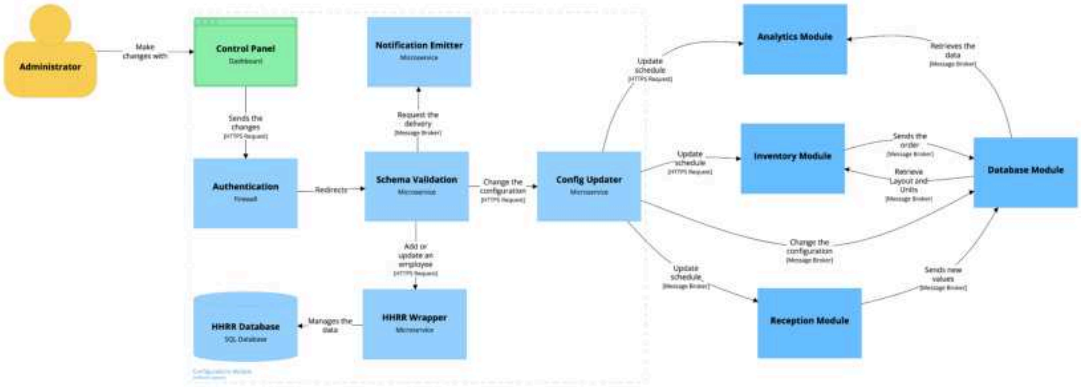

Figura 11: Diagrama de arquitectura sobre los contenedores en el módulo de administración. Elaboración propia.

Para profundizar en el funcionamiento de la arquitectura presentada, se pueden revisar los diagramas de contenedores de los módulos de misiones y bases de datos en el Anexo 4, en ambos casos se reutilizan los patrones y principios descritos en este inciso, para una mejor visualización también se pueden aprovechar las características de diagramas dinámicos que ofrece Structurizr y navegar los distintos contextos. Para acceder a la revisión dinámica se

pueden seguir las instrucciones presentes en el mismo Anexo 4. Antes de terminar este inciso, es valioso comentar algunas de las facultades que ofrece el diseño actual de la arquitectura. En resumen, la principal ventaja que ofrecen los principios comentados tiene un punto en común, el cuál es la seguridad de red y por consiguiente, de los datos del sistema. La protección y control sobre los endpoints del sistema marcan la diferencia en un sistema en la nube, dado que se puede gestionar cuáles son los endpoints expuestos a las redes de dominio público.

Planteamiento del Modelo de Datos

Anteriormente se ha concluido que respecto al diseño de una base de datos, la elección entre un modelo relacional y uno no relacional, radica mayoritariamente en cuál se ajuste mejor a la naturaleza de los datos, y es allí donde hay un comentario en el libro de Copeland (2013) que se mantiene vigente para esta decisión, si los datos tienen una estructura jerárquica, es decir, que un elemento contenga a otros, es recomendable utilizar una lógica no relacional. Esto no quiere decir que datos con dicha estructura no puedan ser representados en una lógica relacional, pero si simplifica algunos tipos de consulta, llevando a un funcionamiento más eficiente. Cabe mencionar que la elección de uno u otro modelo en la arquitectura actual sólo puede representar un problema si los proveedores de almacenamiento elegidos generan un gasto mayor al necesario, esto es por el uso de microservicios que manejen las bases de datos, lo que crea una capa de abstracción que elimina la necesidad de que los demás servicios tuvieran que interactuar con dos tecnologías distintas.

Como se pudo observar en la arquitectura del inciso anterior, hay al menos 2 instancias en las que se utilizará un modelo de datos relacional, esto pues no hacen uso de la estructura jerárquica comentada anteriormente. Además de estas instancias, se ha contemplado otra instancia relacional para el manejo de productos, es decir, el inventario, y las demás instancias utilizarán una lógica no relacional que opera con documentos basados en documentos JSON de manera que admitan propiedades que presenten la jerarquía mencionada. En función de la observación del párrafo anterior, se tiene en consideración que, si el aumento de proveedores y tecnologías representa un problema financiero, esto se solucionará cambiando los modelos relacionales, para que las instancias se manejen como documentos JSON. Esto no sería un problema, pues las llaves foráneas entre modelos no serán una regla a nivel de base de datos, sino que se implementará esa lógica en los microservicios dedicados, externalizando las relaciones del sistema al nivel de la arquitectura.

Con todo lo anterior en mente, en la imagen a continuación se observa una primera parte del modelo de datos, que representa el funcionamiento de las unidades de producto. Todos los diagramas fueron elaborados con la herramienta online dbdiagram.io, en particular los 3 esquemas observados en la primera parte del modelo presentan la relación que podría ser

más problemática del proyecto, ya que existen muchas unidades de un mismo producto y estas unidades deben ser guardadas en cajas. Ahora, respecto a la distribución de unidades entre las cajas disponibles se pretende dejar libertad de uso a los distintos clientes, pero es en esta misma figura que se observa por qué la práctica observada en la Clínica Dávila es deseable. Al generar exclusividad de productos en una misma caja, no solo se simplifica el proceso de picking, sino que también la relación de estos dos esquemas en el modelo de datos, brindando un mejor escalamiento desde la etapa de diseño.

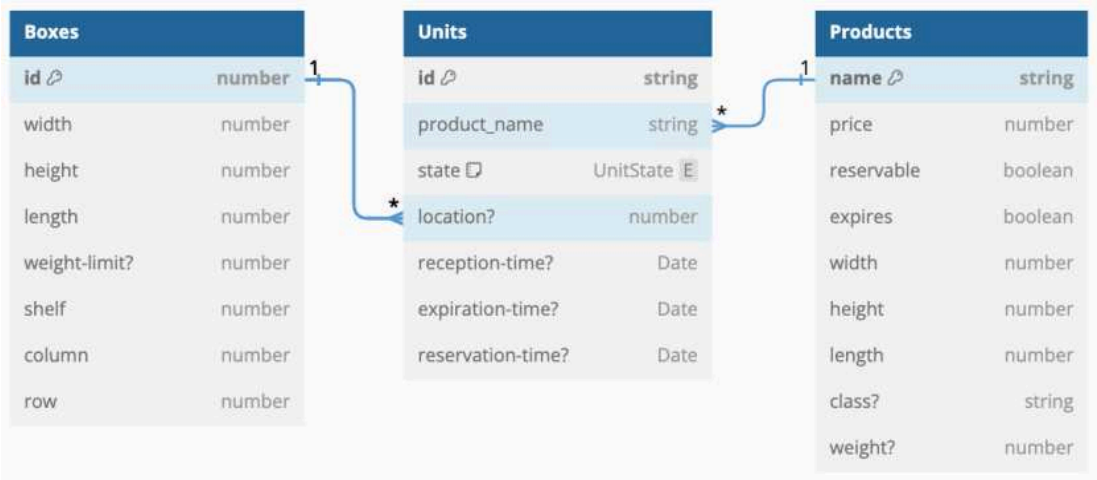

Figura 12: Los esquemas de cajas, unidades y productos del modelo de datos. Elaboración propia.

Además de las relaciones, en el diagrama anterior se pudo observar una notación diferente en la propiedad de estado del esquema de unidades, esta notación representa un tipo de valor que extiende las cadenas de texto conocida como enum, esta estructura se utiliza para establecer un comportamiento constante entre un número finito de opciones para el valor, en la imagen a continuación se pueden observar las enumeraciones utilizadas en todo el modelo para simplificar las explicaciones posteriores. En el caso de la propiedad de estado comentada se contemplan todos los estados por los que pasa una unidad de producto, desde que ésta ha sido comprada o creada para abastecer la bodega, hasta que ha sido vendida a un cliente.

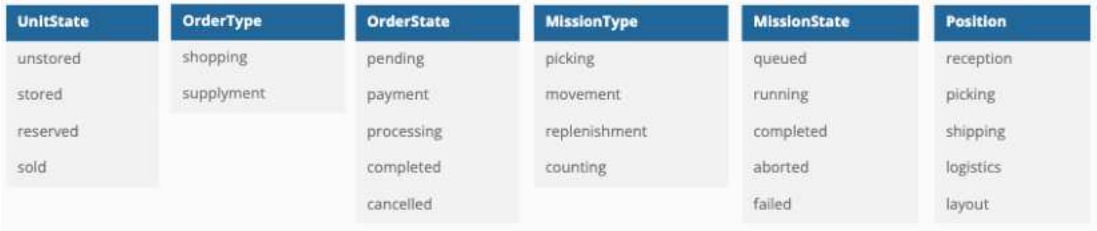

Figura 13: Todas las definiciones de enum del modelo datos. Elaboración propia.

El resto de las definiciones de enum se utilizan en la segunda parte del modelo de datos que se presenta en el diagrama a continuación. Lo primero que se puede observar es que hay dos tipos de órdenes, las de abastecimiento y las de compra. Las primeras representan las órdenes listadas como itinerario por el administrador para reponer el inventario, mientras que las segundas permiten originar misiones de picking, por lo que la relación uno a uno entre ambos esquemas se da sólo cuando el tipo de orden es de compra. Luego, las misiones asignadas a un empleado de picking pueden ser originadas por una órden, o en su defecto ser una misión automática del inventario para hacer efectivo los cambios de administración o mantener la bodega. Dependiendo del tipo de misión, la propiedad del "camino de cajas" así como la de "productos" serán utilizadas de distinta forma en la aplicación, para más detalles sobre esta implementación revisar el Anexo 5.

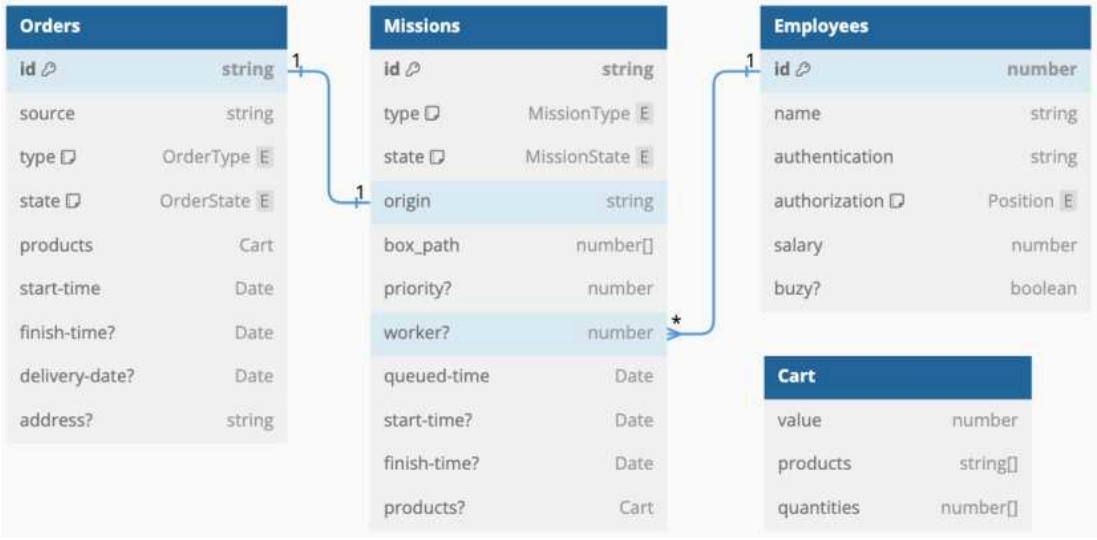

Figura 14: Los esquemas de órdenes, misiones y empleados del modelo de datos. Elaboración propia.

Por último, los empleados pueden tener distinto grado de acceso al sistema a raíz de sus responsabilidades, por ello la propiedad de autorización enumera las posiciones disponibles, dando la posibilidad de diferenciar los permisos a nivel administrativo. Por otro lado se tiene también la autenticación, propiedad que se explicará más adelante. Antes de continuar se hará una revisión sobre la notación. Como se ha visto hasta ahora, hay algunas propiedades con un signo de interrogación o una llave al final del nombre. Estas son formas de representar características de las propiedades, por ejemplo, la llave hace alusión a las características de una llave primaria, por lo que se considera como un elemento no nulo y único que cumple la función de identificar una entrada. Por otro lado, el signo de interrogación da cuenta de que la propiedad puede ser nula, lo que implica a su vez, que el resto del esquema no debe ser nulo, es decir, que todos los valores deben ser inicializados.

Para finalizar el modelo, en el siguiente diagrama se muestra la última parte del modelo, que corresponde a los datos más administrativos, como lo son los Indicadores de desempeño y las configuraciones del sistema. A primera vista se puede notar que ambos esquemas están abreviados, esto es necesario por un tema de espacio en los indicadores, pues son muchos. Pero respecto a las configuraciones la razón es otra, el sistema aún no ha sido probado, y como tal, no se tiene conocimiento sobre qué aspectos del sistema podrían ser dinamizados para mejorar la experiencia de los clientes, razón por la cual se ha decidido no ahondar en este punto hasta que se tengan más datos sobre el uso que el sistema tiene. Finalmente, las configuraciones contienen también el layout del almacén, el cual se ve representado por dos estructuras de datos numéricos, las repisas y los pisos, los pisos son instancias rectangulares de 2 dimensiones sobre las cuáles almacenar repisas, repisas sobre las que se pueden ubicar las cajas definidas anteriormente.

Figura 15: Los esquemas de indicadores y configuraciones del modelo de datos. Elaboración propia.

Respecto a la autenticación de usuarios, como se pudo ver en la arquitectura, el administrador maneja el esquema de empleados, razón por la cual tiene un método de autenticación distinto, la idea en cuestión es limitar la forma de acceder al sistema a que solo el administrador pueda acceder por su cuenta mediante una autenticación con un token secreto, y es el administrador quien permite accesos limitados a la infraestructura que requieran de un factor presencial, como podría ser un código QR. Como comentario final, se observa que el modelo asume su funcionamiento para una única empresa, se ha elegido seguir este acercamiento, puesto que aún no hay certeza de que podría ser más barato en infraestructura, si crear un proyecto por empresa o un proyecto para todas. De igual manera, para incorporar más empresas simplemente se debe agregar la empresa a las llaves primarias de los esquemas.

Diseño y Análisis de las Vistas del Sistema

En este inciso se presentarán las vistas que permiten la interacción entre los usuarios y el sistema. La forma en que se presentarán las vistas es mediante wireframes, que son un diagrama que describe sólo la disposición de la información en el marco visible, es decir, la posición de los distintos elementos de la interfaz, de forma que sea explícita la forma en que se mostrará la información al usuario. Los wireframes mencionados, no contemplan el uso de colores u otros componentes del diseño de interfaces, pero como posteriormente se deberá trabajar sobre esos apartados, se ha elegido desarrollar estas vistas en una aplicación para el diseño de interfaces, la aplicación en cuestión es Figma. De antemano, el detalle sobre el análisis de usabilidad para cada vista se podrá encontrar en el Anexo 6, para abreviar, en este inciso se comentará sobre cuál fue la característica principal en cada caso.

Para comenzar, se retomará una decisión tomada en el inciso anterior, la cuál corresponde al modelamiento de permisos de acceso para los distintos usuarios. En particular con respecto a los empleados administrativos de la operación, como lo son el analista de la logística, el encargado del layout de la bodega o el mismo administrador. Estos deberán contar con la información relevante para desempeñarse en su trabajo, lo que lleva a observar que todos necesitan la información del módulo de analíticas. A razón de esta observación, se ha optado por la creación de una dashboard equivalente para todos estos usuarios, dónde en función de los permisos habilitados el usuario podrá acceder a distintas ventanas de la dashboard. Con eso en mente, la dashboard del usuario administrador tendrá acceso a todas las vistas como se observa en la siguiente imagen.

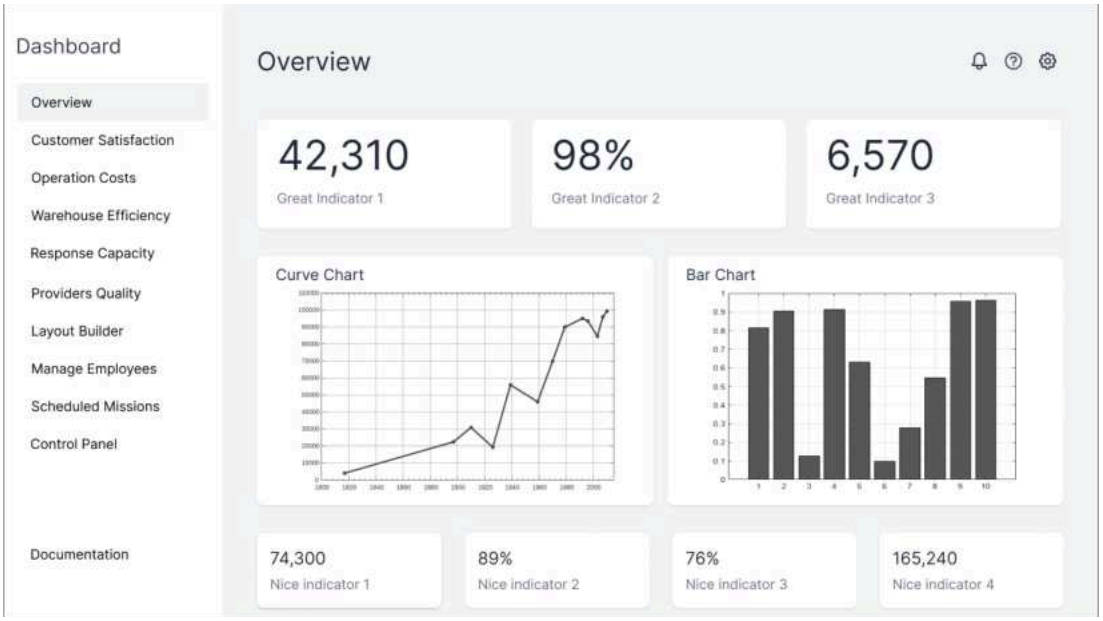

Figura 16: Wireframe de la vista inicial de la dashboard para administrativos. Elaboración propia.

En el wireframe anterior considera de manera prioritaria la aprendibilidad, puesto que reutiliza elementos gráficos ampliamente conocidos en aplicaciones web, como lo son las barras laterales de navegación o las notificaciones, o también la ayuda rápida disponible en el ícono de interrogación en la esquina superior derecha. Este foco se dió con la intención de que la capacitación en el uso de esta aplicación sea lo más acompañada y simple posible. El diseño responsivo también será contemplado en el desarrollo de la aplicación final, pero en este caso su implementación supone un diseño reinventado, puesto que los componentes actuales aprovechan la orientación horizontal de un navegador en la computadora del usuario. Para una idea más clara, a continuación, se observan vistas en orientación vertical de otras vistas de la dashboard.

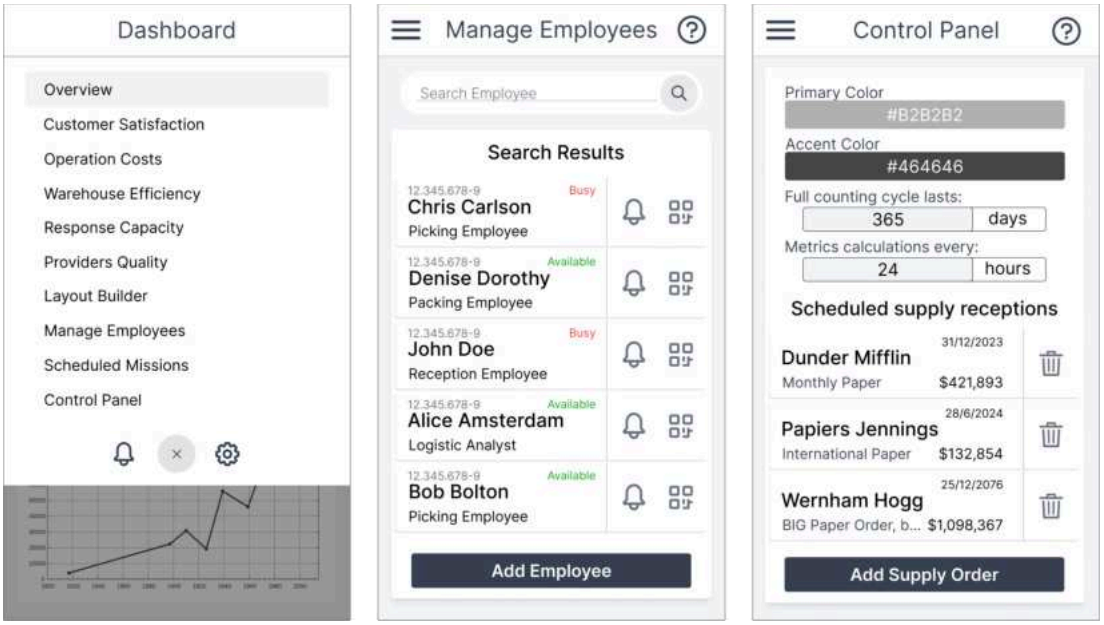

Figura 17: Wireframes de algunas vistas de la dashboard en portarretrato. Elaboración propia.

Como se puede observar, la transformación para el diseño responsive supone un esfuerzo adicional, casi al punto de diseñar dos aplicaciones. La mayor complicación es el hecho de que algunos botones deben reubicarse para que la interfaz aproveche la familiaridad del usuario con el sistema en el que observa la vista, como es el caso de la barra de navegación lateral, que fue completamente oculta tras el ícono de menú para ser reemplazada por un menú desplegable. Por otro lado, se observan las funcionalidades de las que dispone el usuario administrador, por ejemplo, en la vista de administración de recursos humanos se permite buscar entre los empleados contratados, notificarlos con un mensaje personalizado, editar o borrar su información, darles acceso al WMS mediante un método de autenticación presencial con códigos QR, además de agregar nuevos empleados en medida de que sean contratados.

Dado el enfoque decidido en el Marco Metodológico, se dará prioridad a las vistas en portarretrato de las aplicaciones del sistema. Ahora bien, la excepción a esta regla sería el constructor de layouts, que como su nombre indica, se trata de un editor sobre el cual modelar la disposición física de las repisas en la bodega y las medidas de la bodega misma. En el wireframe del constructor de layouts se puede observar cómo se ha puesto el mayor énfasis en la flexibilidad del editor, esto se debe a que en las distintas operaciones a las cuales el sistema deberá poder integrarse son distintas a varios niveles, no solo las dimensiones de la bodega pueden ser diferentes, también lo es la forma en que el área de la bodega se distribuye, incluso las repisas pueden tener distintas formas y capacidades. Un ejemplo de esto se ve en la fotografía a continuación, donde se observa como algunas operaciones requieren de repisas con cajones pequeños para algunos de sus productos.

Figura 18: Fotografía de cajones pequeños para picking en la bodega de la Clínica Dávila. Elaboración propia.

En el wireframe a continuación se observa la vista del constructor de layouts, retomando la observación anterior, en la vista se puede notar la incorporación de botones para agregar tipos de repisas y pisos distintos. El modelo incluye un punto de inicio para el picking y un punto de término para el packing y shipping, también bloques de obstrucción para hacer inutilizable las celdas que sean una pared o un pilar y por último bloques de tránsito, para modelar el paso de uno a otro piso. Los pisos que fueron parte del esquema de configuraciones en el inciso anterior representan áreas de la bodega que sea beneficioso de separar. Retomando el ejemplo de la operación de la Clínica Dávila, en esa bodega se separaba la bodega de cross

docking, de una bodega de picking que no requería desempacar las entregas de recepción y a su vez se separaban de la bodega de picking con cajas de plástico observada anteriormente.

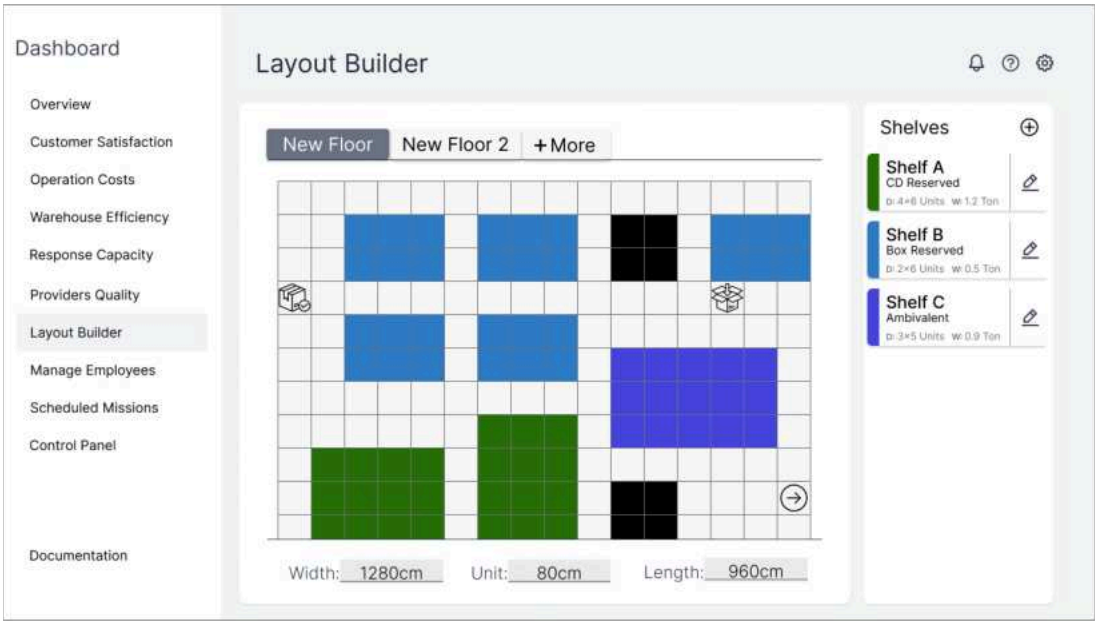

Figura 19: Wireframe de la vista del constructor de layout en la dashboard. Elaboración propia.

Cabe mencionar que las repisas podrán ser asignadas a cross docking, al uso de cajas o ambos, lo que significa que estos tipos de repisas deberán estar etiquetados para su posterior navegación con la aplicación que se muestra en los párrafos siguientes. La aplicación mencionada es la aplicación móvil para empleados de picking, que es donde se incorporó la noción de agregar la gamificación, esto mediante el uso de una interfaz amigable y dinámica que guíe a su operador a través de la misión que debe completar, reforzándolo a continuar en cada paso. Cómo se eligió este objetivo, las características de usabilidad en las que se hará mayor énfasis son la interactividad y la accesibilidad, esto pues son las que guardan mayor relación con brindar una experiencia similar a la de un juego.

En las siguientes vistas se observa la navegación de un paso a seguir por parte del operador para completar la misión que se le ha asignado. El flujo observado es cíclico en función del paso descrito. Este consta de una etapa de navegación, la cual termina cuando el usuario está cerca de la caja con la que va a interactuar. El siguiente paso es la validación de la caja, donde el dispositivo solicitará confirmar con la cámara que se está interactuando con la caja correcta. Finalmente, aparece una descripción de qué se espera que se haga con la caja o su interior, una vez el operador ha completado esa solicitud, podrá presionar de forma breve la pantalla para completar este paso, y así continuar con el próximo paso hasta terminar con la misión. La presión breve es parte de los elementos de accesibilidad que han sido considerados, esto para evitar una finalización del paso mediante un toque accidental en la pantalla.

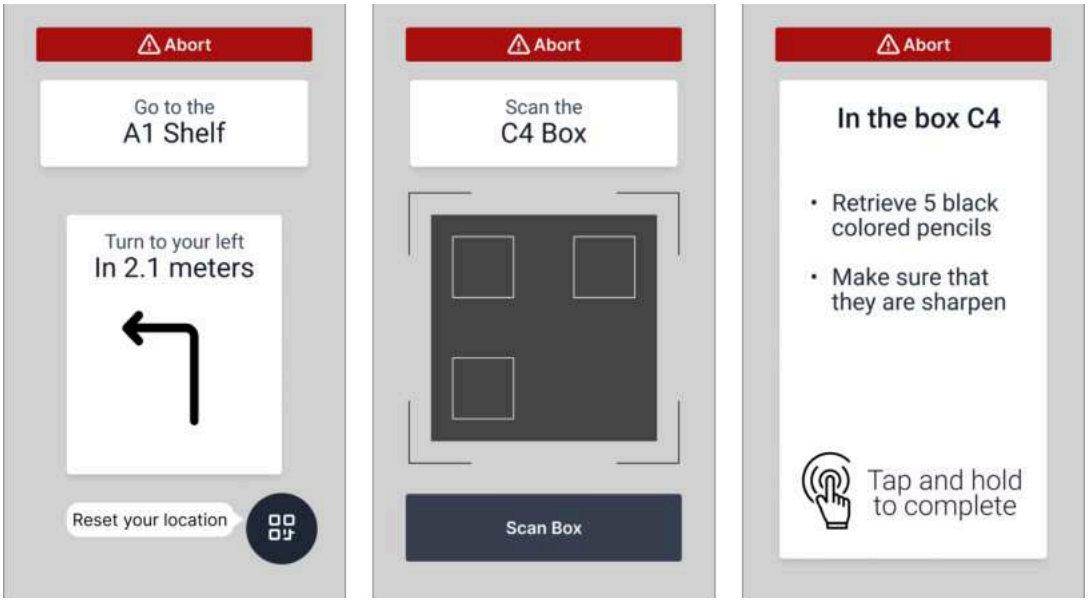

Figura 20: Wireframes de la navegación de un paso de la app de picking. Elaboración propia.

Para finalizar, a continuación, se observan las vistas en portarretrato de las interfaces que los encargados de shipping y de recepción utilizan. En este punto solo se replican las técnicas y explicaciones anteriores, donde el encargado de shipping presenta la vista de confirmación del término de una misión de picking y el listado de órdenes en curso para su preparación del despacho. Por otro lado, en la recepción se requiere llenar un formulario que confirme la recepción de los insumos previamente solicitados por el administrador. Ambas aplicaciones tienen su enfoque en la aprendibilidad, esto pues no son vistas que permitan completar todo el trabajo de quien la usa, por lo que se busca que su integración sea sencilla.

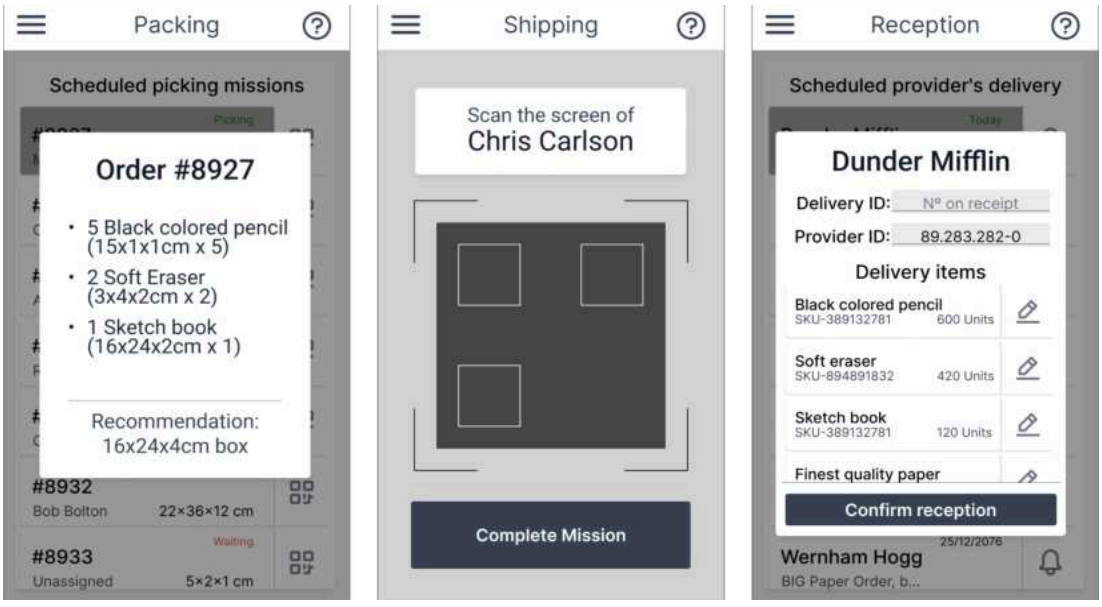

Figura 21: Wireframes de las vistas de shipping y recepción. Elaboración propia.

Proyecto Open Source y Contribuciones

Antes de terminar este capítulo, cabe mencionar que los avances y contribuciones hasta ahora han sido realizados en su totalidad solo por el tesista. Ahora bien, en un esfuerzo de concretar el proyecto más rápido, además de mejorar los cimientos ya establecidos en el proceso, se ha creado una organización pública de GitHub bajo el nombre de SLC-wms, nombre proveniente de la S de Smart y LC de Low-Cost. Por su parte, la comunidad desarrolladora ha establecido una lista de estándares a seguir en orden de que el proyecto cuente con el grado de organización y seriedad necesaria para que sea seguro y valioso contribuir en él. En la imagen a continuación, se puede ver una captura de pantalla que muestra el grado de cumplimiento de estos estándares por parte de la organización SLC-wms en sus repositorios finales.

Figura 22: Los "Community Standards" del proyecto en GitHub. Extraído del apartado de analíticas del repositorio principal en GitHub.

Como se puede observar, todos los documentos y requisitos se han completado, para ello se ha hecho una investigación informal del contenido esperado en los documentos y se han llenado de forma personalizada para que cumplan con su propósito. El código de conducta utiliza el estándar del "Acuerdo de Contribuidores" sugerido por GitHub. La licencia de software es GPL versión 3.0, que es una de las licencias más restrictivas acorde al principio de Copyleft, para asegurar que el proyecto se mantenga open source después de su publicación. La política de seguridad presenta la intención de un software seguro acorde a la protección ante la Enumeración de Debilidades Comunes (CWE) y un protocolo de comunicación segura para el reporte de vulnerabilidades. Finalmente, en los lineamientos de contribución se establece el estándar bajo el cual se trabajará en el proyecto, además de un acuerdo de autenticidad de la contribución elaborado por la comunidad de Linux.

Resultados Adicionales

Entre los resultados obtenidos en la duración de este trabajo, se encuentran los diagramas presentados en el capítulo anterior como parte del Resultados del Proyecto, en dicho desarrollo se pudo observar que el software aún se encuentra solo en una etapa teórica. Pero, aún si este es el caso, se han creado cimientos robustos para el posterior desarrollo del sistema por medio de un estudio teórico con bases comprobadas, así como en la comunicación y distribución del proyecto mismo a través del cumplimiento de los estándares de la comunidad open source en GitHub. En este inciso, se observará el último resultado que no formó parte del resto del proceso de desarrollo, ya que significó el estudio individual del trabajo encontrado en la tesis de Arias. A continuación se observa una tabla que resume los indicadores a considerar, separados mediante la definición de la tesis de Rodríguez sobre los tipos de indicadores.

Eficacia	Eficiencia	Calidad	Economía
Flexibilidad de Entrega (FLE)	Stock Fuera de Fecha (SFF)	Tiempo de Respuesta a Consultas (TRC)	Costos Totales de la Operación (CTO)
Frecuencia de Entrega (FRE)	Utilización de la Capacidad (UC)	Desabastecimiento en el Tiempo (DT)	Costos de Almacenamiento (CA)
Tiempo de Procesamiento (TP)	Cobertura de Inventario (CI)	Número de Pedidos Pendientes (NPP)	Valor del Inventario (VI)
Tiempo de Logística por Pedido (TLP)	Rotación de Inventario (RI)	Número de Entregas a Tiempo (NET)	Costo por Salarios (CS)
Tiempo Total por Pedido (TTP)	Demanda Atendida sin Retrasos (DAR)	Tasa de Fallo de Entregas (TFE)	Costos de Compra al Proveedor (CCP)
Tiempo por Compra del Proveedor (TCP)	Tiempo Promedio por Empleado (TPE)	Tasa de Fallo del Proveedor (TFP)	Valor a Favor con el Proveedor (VFP)
Oferta Disponible del Proveedor (ODP)	Tasa de Utilización del Almacén (TUE)	Evaluación del Proveedor (EP)	

Figura 23: Tabla de indicadores de desempeño que se integrarán en el software WMS. Elaboración propia.

Para más detalles sobre los indicadores, se pueden revisar aquellos indicadores con los mismos acrónimos en las tablas del Anexo 1. En el siguiente capítulo se realizará un análisis sobre el diseño desarrollado, incluyendo tópicos como las oportunidades que provoca el proyecto, las debilidades y riesgos actuales que han sido contemplados y los beneficios de las decisiones tomadas hasta ahora. Para ello, se contempla el capítulo anterior como centro de la discusión, puesto que corresponde al diseño completo del producto principal.

Discusión

El proyecto verdaderamente presenta una oportunidad de resolver el problema planteado al inicio del escrito, en la tesis de Cornejo y Gómez (2023) observan con fuente en el Servicio de Impuestos Internos (SII), que "las MiPymes en Chile que representan el 98,6% de las empresas y el 65,3% del empleo formal", y profundizando sobre los datos del SII, al menos el 42.6% de las MiPymes operan en un rubro que se relaciona a la SCM. Esto se traduce en aproximadamente 500000 empresas que podrían acceder a una mejora en la calidad de su servicio y su experiencia laboral con este proyecto. Otra oportunidad no menor que ofrece el llevar a cabo este proyecto es que, al seguir una lógica open source, se da acceso a desarrolladores menos experimentados a aprender mediante sus contribuciones al código fuente del sistema y también permite a los desarrolladores experimentados pulir sus habilidades con las tecnologías utilizadas.

Cabe mencionar que el costo de mantención del sistema ya representa un ahorro respecto a la problemática presentada en la Introducción, para más detalles se puede revisar el Anexo 7. Pero un gran problema que tiene este proyecto es que aún no está implementado. De tener código funcional, no solo se podrían hacer las pruebas necesarias para determinar con más seguridad los costos, sin depender en gran medida de supuestos respecto a la operación, sino que además el acceso a una implementación permite la obtención de retroalimentación de múltiples fuentes, como lo puede ser hacer una encuesta para validar empíricamente algunas características de usabilidad que están enfocadas en la percepción subjetiva de las distintas vistas, o encontrar detalles de implementación que indiquen la necesidad de arreglar algún componente en el diseño original de la arquitectura y finalmente, pero no menos importante, al tener múltiples repositorios con código funcional, hay más posibilidades de que otros desarrolladores encuentren el proyecto y se unan a la colaboración.

Aun así, se pueden revisar aspectos dentro de las decisiones de diseño tomadas, hay 2 conceptos críticos que avalan la priorización que se tuvo sobre el proceso de picking y la manera en que se abordó la construcción de la arquitectura de microservicios. Según Anđelković y Radosavljević (2018), el 55% de los costos de una operación logística corresponden al área de picking, esto hace evidente que cualquier esfuerzo en mejorar el proceso, permite la reducción de costos de forma significativa en la operación. Pero esto ya lo comentan los mismos autores, y es que el trabajo que escribieron es acerca del impacto que tiene un WMS sobre el proceso de picking. En dicho documento concluyen que cuando el WMS tiene un gran impacto positivo sobre el proceso mencionado, se tenía que el 90% de las operaciones que así lo experimentaban, eran SME, que es el equivalente en inglés para la MiPymes, lo que prueba el valor de un WMS para las MiPymes que se relacionan con la SCM.

Como parte del proceso de picking, se buscó la integración del primer concepto, la gamificación, integración que se hizo primordialmente en las vistas y el proceso. Como describen Kiselichki y Josimovski (2022), en la implementación de la gamificación en entornos productivos existen una serie de riesgos, y de hecho la gran mayoría están ligados a una pobre ejecución en la incorporación de KPIs que evalúen a los empleados de forma competitiva. Con un único indicador referente al tiempo promedio por orden de cada empleado (TPE), estos riesgos se reducen bastante en medida que este KPI no se maneje irresponsablemente, de igual modo, en el mismo trabajo dan a conocer un conjunto de estrategias para prevenir estos riesgos, pero este requiere contar con las herramientas de software implementadas. Por lo tanto, dicha estrategia se preservará en la documentación del proyecto, espacio que presentará información técnica del software, así como un manual de usuario con sugerencias como esta.

El último concepto se explica en el trabajo de Gannon et al. (2017), la arquitectura de software diseñada sigue los principios "cloud native", puesto que cumple con que todos los nodos de la arquitectura son microservicios, los cuales se gestionan independientemente, se despliegan de forma continua mediante el uso de contenedores como Docker o Containerd, y que el método de despliegue para los servicios que no sean nativos de cloud, es decir, aquellos microservicios implementados, pueden ser desplegados mediante servicios de tipo serverless. Adicionalmente, sobre estos principios existe una regla adicional considerada una buena práctica. Esta consiste en emplear la cultura serverless desde la etapa de diseño, estableciendo que los microservicios deban tener una única responsabilidad, para que puedan ser modeladas como una función libre de estado que recibe una petición, responde luego de procesar y libera los recursos utilizados.

Las funciones mencionadas, son el servicio que dio origen al concepto serverless, en Amazon se les llama Lambdas, en Google son las Cloud Functions y en Azure son Functions. En el caso de la arquitectura presentada, no se sigue esta regla de forma explícita, pero los elementos si están interconectados en base al flujo de datos, lo que permite separar las responsabilidades adoptadas por los microservicios en tantas funciones como flechas entrantes haya. Para hacerlo explícito solo habría que agregar la capa de componentes, que es algo que formará parte de la documentación. Ahora, respecto a la ventaja que supone que un sistema de software sea implementado bajo los principios de cloud native, ésta va más allá de una mejor escalabilidad y menor costo, que es lo que ofrecen los CSP. Según Nandhini et al. (2020), una arquitectura que se basa en cloud native presenta una mejora en la seguridad del sistema, aumenta la agilidad para la mejora continua y vuelve al sistema más resiliente a errores.

Conclusiones

Adicionalmente a los objetivos propuestos, en la Introducción se presentó una problemática respecto al bajo acceso que tienen las MiPymes a los software WMS para sus operaciones. Con el desarrollo del software libre SLC-wms, se lograría un ahorro para estas empresas, ya que el costo de licencias y desarrollo sería nulo, mientras que el gasto mensual por mantener el sistema en línea sería menor al costo mensual por usuario que ofrecen los líderes del mercado de WMS. Esto requeriría un poco más de trabajo para los usuarios, puesto que tendrían que hacer el despliegue del software en lugar de solo pagar, pero con esto ganarán la soberanía de sus datos de operación, además de la opción de extender el sistema de forma manual para adquirir funcionalidades más específicas a su operación en caso de ser necesario. La planificación lograda agrega un caso de éxito de la metodología ágil propuesta, puesto que se alcanzaron los objetivos en el tiempo establecido. Con el tiempo, esta metodología podrá probar su utilidad en la implementación de este proyecto open source.

Finalmente, cabe mencionar que el proyecto aún se encuentra en una etapa teórica, puesto que se han realizado avances sólo en la etapa de diseño y planificación del software, pero ha sido un buen comienzo. Para lograr este avance se han hecho investigaciones respecto al mercado actual para definir las necesidades de las operaciones de bodega y posibilidades de mejora que no extendieran significativamente el tiempo de implementación. Además, se han investigado las tecnologías contemporáneas para construir un sistema distribuido centrado en la escalabilidad y el ahorro de recursos, basándose en lograr una arquitectura nativa en el ecosistema cloud. De la discusión del capítulo anterior se tiene también que existen estudios que avalan el enfoque en la gamificación del picking y la bodega en general como un esfuerzo útil para mejorar la utilidad entregada por el WMS. Por lo tanto, en función de lo planteado en los Objetivos, se tiene que el diseño del software ha sido completado hasta una primera versión, en la que satisfaga las necesidades básicas que las operaciones de las MiPymes tienen.

Referencias

Anđelković, A., & Radosavljević, M. (2018). *Improving order-picking process through implementation of warehouse management system*. Strategic Management-International Journal of Strategic Management and Decision Support Systems in Strategic Management, 23(1).

Arias, M. G. (2022). *Análisis y definición de indicadores para la evaluación del desempeño de cargos directivos dentro de áreas dedicadas a la gestión de la cadena de suministro para las principales industrias en Chile*. Universidad de Chile.

Bahr, W., Mavrogenis, V., & Sweeney, E. (2022). *Gamification of warehousing: exploring perspectives of warehouse managers in the UK*. International Journal of Logistics Research and Applications, 25(3), 247-259.

Baresi, L., & Garriga, M. (2020). *Microservices: The evolution and extinction of web services?*. Microservices: Science and Engineering, 3-28.

Beck, Beedle, Van Bennekum, Cockburn, Cunningham, Fowler, ... & Thomas. (2001). *The agile manifesto*. Agile Alliance.

Bevan, N., Carter, J., & Harker, S. (2015). *ISO 9241-11 revised: What have we learnt about usability since 1998?*. In Human-Computer Interaction: Design and Evaluation: 17th International Conference, HCI International 2015, Los Angeles, CA, USA, August 2-7, 2015, Proceedings, Part I 17 (pp. 143-151). Springer International Publishing.

Brown, S. (2017). *Software Architecture for Developers-Volume 2: The C4 model for visualising software architecture*. Ebook.

Cho, J. J. (2010). *An exploratory study on issues and challenges of agile software development with scrum*. All Graduate theses and dissertations, 599.

Copeland, R. (2013). *MongoDB Applied Design Patterns: Practical Use Cases with the Leading NoSQL Database*. O'Reilly Media, Inc.

Cornejo Escobar, J., & Gómez Alegría, R. (2023). *FINALIB*. Universidad de Chile.

Cristi Laurich, P. A. (2003). *La administración de la cadena de suministros y la aplicación de innovaciones tecnológicas*. Universidad de Chile.

Denisov, S. A., & Sorokin, A. A. (2021). *Model of a management system for deterministic scientific services of digital platform*. Procedia Computer Science, 186, 1-10.

Dujmešić, N., Bajor, I., & Rožić, T. (2018). *Warehouse processes improvement by pick by voice technology*. Tehnički vjesnik, 25(4), 1227-1233.

Gannon, D., Barga, R., & Sundaresan, N. (2017). *Cloud-native applications*. IEEE Cloud Computing, 4(5), 16-21.

Kiselichki, M., & Josimovski, S. (2022). *Risks of Implementing Gamification: A Literature Review*. ESD Conference.

Kofler, Beham, Wagner, Affenzeller (2014). *Affinity based slotting in warehouses with dynamic order patterns*. In Advanced methods and applications in computational intelligence (pp. 123-143). Heidelberg: Springer International Publishing.

Laranjeiro, N., Agnelo, J., & Bernardino, J. (2021). *A black box tool for robustness testing of REST services*. IEEE Access, 9, 24738-24754.

León Santibáñez, V. A. (2018). *Rediseño del sistema de control y manejo multibodega para Farmoquímica del Pacífico*. Universidad de Chile.

Lucena, Braz, Chicoria, Tizzei (2016, Nov). *IBM design thinking software development framework*. In Agile Methods: 7th Brazilian Workshop, WBMA 2016, Curitiba, Brazil, November 7-9, 2016, Revised Selected Papers 7 (pp. 98-109). Springer International Publishing.

Mellado Loch, B. H. (2015). *Análisis del estado actual de gestión de bodega en obras de construcción de edificación en altura*. Universidad de Chile.

Morales Vargas, A. (2023). *Entre lo que los usuarios dicen y lo que hacen: métodos de investigación UX más útiles para evaluar la calidad web*. Anuario ThinkEPI, 17.

Nandhini, S., Joseph, A., & Ajay, S. (2020). *Impact of Implementing cloud native Applications in Replacement to on-Premise Applications*. IJERT, Vol 9. 1621-1625.

Newman, S. (2021). *Building microservices: Designing Fine-Grained Systems*. O'Reilly Media, Inc.

One, V. (2022). *16th annual state of agile development survey results*.

Papazoglou, M. P., & Van Den Heuvel, W. J. (2006). *Service-oriented design and development methodology*. International Journal of Web Engineering and Technology, 2(4), 412-442.

Qi, Huo, Wang, Yeung (2017). *The impact of operations and supply chain strategies on integration and performance*. International Journal of Production Economics, 185, 162-174.

Rahmat, Zulzalil, Ghani, Kamaruddin (2017, Sep). *Usability evaluation checklist for smartphone app*. Journal of Engineering and Applied Sciences, 12, 4127-4131.

Rodríguez Sandoval, M. A. (2020). *Propuesta de una herramienta de control financiero para la línea de negocios industrial de la Empresa SGS SA*. Universidad de Chile.

Schwaber, K. (1997). *Scrum development process*. In Business Object Design and Implementation: OOPSLA'95 Workshop Proceedings 16 October 1995, Austin, Texas (pp. 117-134). Springer London.

Smith, Katsurashima, Warrilow, Iams, Watson (2023, Sep). *Magic Quadrant for Container Management*. Gartner Core Research.

Tonin, Goldman, Seaman, Pina (2017). *Effects of technical debt awareness: A classroom study*. In Agile Processes in Software Engineering and Extreme Programming: 18th International Conference, XP 2017, Cologne, Germany, May 22-26, 2017, Proceedings 18 (pp. 84-100). Springer International Publishing.

Tornhill, A., & Borg, M. (2022, May). *Code red: The business impact of code quality-A quantitative study of 39 proprietary production codebases*. In Proceedings of the International Conference on Technical Debt (pp. 11-20).

Tunstall, Klappich, Narang, Stufano (2023, May). *Magic Quadrant for Warehouse Management Systems*. Gartner Core Research.

Valduriez, P., Jiménez-Peris, R., & Özsu, M. T. (2021). *Distributed database systems: The case for NewSQL*. In Transactions on Large-Scale Data-and Knowledge-Centered Systems XLVIII: Special Issue In Memory of Univ. Prof. Dr. Roland Wagner (pp. 1-15). Berlin, Heidelberg: Springer Berlin Heidelberg.

Wright, Katsurashima, Warrilow, Iams, Watson (2023, Dic). *Magic Quadrant for Strategic Cloud Platform Services*. Gartner Core Research.

Wiese, Rachow, Riebisch, Schwarze (2022). *Preventing technical debt with the TAP framework for Technical Debt Aware Management*. Information and Software Technology, 148, 106926.

Wijffels, Giannikas, Woodall, McFarlane, Lu (2016). *An enhanced cycle counting approach utilising historical inventory data*. IFAC-PapersOnLine, 49(12), 1347-1352.

Zenteno Fouilloux, E. J. (2017). *Propuesta de rediseño del proceso de pedidos y despacho de alimentos del cliente COMPASS, para mejorar la calidad de servicio y optimizar recursos utilizados en el proceso*. Universidad de Chile.

Anexos

Para más información del estado del proyecto, visite: https://github.com/SLC-wms.

Anexo 1: Tablas de Indicadores

En las siguientes tablas se observan los indicadores presentados en el trabajo de Arias, las tablas originales contaban con 4 columnas adicionales que fueron eliminadas para simplificar la lectura, es decir, en favor de más espacio para la descripción. De esas 4 columnas, 2 de ellas podrían ser un poco más relevantes para este trabajo, la métrica y las consideraciones, para complementar la lectura se pueden revisar las tablas desde la fuente original. Finalmente, en las tablas se indicará destacando con color rojo cuáles indicadores no serán contemplados para una primera versión, la razón de ello se explicará de forma particular al final de la tabla.

Categoría	Nombre del Indicador	Descripción
Calidad de la logística de salida	Entregas a Tiempo (NET)	Porcentaje de cumplimiento en la entrega de los pedidos/órdenes de venta en cuanto a la fecha o periodo de tiempo acordado con el cliente.
Calidad de la logística de salida	Tiempo de Respuesta a Consultas (TRC)	Tiempo promedio en responder las consultas al cliente.
Calidad de la logística de salida	Tiempo de Respuesta a Reclamos	Tiempo promedio en solucionar los reclamos al cliente.
Calidad de la logística de salida	Tasa de Fallo (TFP)	Porcentaje de pedidos/órdenes que no llegan de acuerdo con las especificaciones de calidad y servicio definidas por presentar fallos durante la entrega al cliente final.
Costos	Costo Total de la Operación (CTO)	Costo total de procesar, almacenar y enviar pedidos al cliente final.
Costos	Costo de Distribución o Transporte	Porcentaje de impacto que tienen los costos de distribución sobre las ventas de la compañía durante el periodo, es decir, representan el porcentaje de los ingresos invertidos en el proceso de distribución.
Eficiencia	Tasa de Llenado (DAR)	Porcentaje de la demanda atendida con el inventario existente sin retraso.

Figura 24: Tabla de indicadores de desempeño comunes en manufactura y comercio. Adaptación propia de la tabla 9 de Arias.

Respecto al Tiempo de Respuesta a Reclamo, no se implementaría en primera instancia, puesto que el software propuesto no contempla la gestión de reclamos/devoluciones de la operación, es decir, el área de postventa. Si bien se presentan los ajustes necesarios que permitan agregar un módulo de postventa, se priorizará la entrega de un software que cumpla con las responsabilidades estrictas de un WMS primero. De igual manera, el Costo de Distribución o Transporte, no se calculará de momento, pues no pertenece a las responsabilidades de la gestión de bodega que el WMS ofrece, lo más seguro es que no aumente demasiado la complejidad del software, por lo que es considerable para el futuro.

Categoría	Nombre del Indicador	Descripción
Costos	Costo de Almacenamiento (CA)	Costo promedio de mantener inventario, es decir mide el impacto del costo por m^3 almacenado o unidad almacenada.
Costos	Valor del Inventario (VI)	Valor total actual de los productos en inventario.
Capacidad de respuesta	Ciclo de la Cadena de Suministro (TLP)	Tiempo promedio entre la emisión de la orden de compra de materias primas al proveedor hasta que la orden de venta es despachada al cliente final.
Capacidad de respuesta	Adaptabilidad	Porcentaje de adaptabilidad cuando el cliente cambia las especificaciones de su pedido inicial o su orden de venta inicial.
Capacidad de respuesta	Flexibilidad de Entrega (FLE)	Porcentaje de pedidos/órdenes de venta con modificaciones en la entrega, solicitadas por parte del cliente que son entregados a tiempo.
Capacidad de respuesta	Tiempo Promedio de Entrega de los Pedidos al Cliente (TTP)	Tiempo promedio desde que se recibe una orden de venta por parte del cliente hasta que la orden es despachada al cliente final.
Capacidad de respuesta	Tiempo Promedio Procesamiento (TP)	Tiempo promedio en convertir las especificaciones del cliente en información ejecutable, es decir en una orden de trabajo.
Capacidad de respuesta	Tiempo Promedio de Planificación	Tiempo promedio en convertir una orden de trabajo en un plan de manufactura, es decir en una orden de fabricación.
Capacidad de respuesta	Tiempo en Tránsito	Corresponde al tiempo promedio empleado únicamente en el recorrido durante el despacho del producto o de la orden de venta. Es decir, la suma de todos los tiempos iniciales y finales del despacho.

Capacidad de respuesta	Tiempo Promedio de Despacho	Tiempo promedio desde que la orden de venta está completa o lista para ser despachada hasta su entrega al cliente final.
Eficiencia	Merma (SFF)	Porcentaje de productos del total de productos del periodo que no se lograron vender a tiempo, porque se sobreestimó su venta y que originaron pérdidas a la empresa, o simplemente se dañaron.
Eficiencia	Utilización de la Capacidad (UC)	Porcentaje de utilización del espacio de almacenamiento sobre su capacidad (m^2 o las posiciones).
Eficiencia	Días de Inventario (CI)	Tiempo que la cantidad de inventario existente permite cubrir las necesidades de los clientes.
Proveedor	Tasa de Fallo (TFP)	Porcentaje de órdenes entregadas por el proveedor que llegan incorrectamente tanto en cantidad, tiempo y calidad.
Proveedor	Cumplimiento de la Demanda (ODP)	Porcentaje de cumplimiento de la demanda de órdenes solicitadas al proveedor durante el periodo.
Proveedor	Evaluación del Proveedor (EP)	Se refiere a la puntuación promedio de los proveedores de la empresa.
Proveedor	Ciclo de la Orden de Compra (TCP)	Tiempo promedio entre la emisión de la orden de compra de materias primas al proveedor y su recibimiento en las instalaciones de la empresa.

Figura 25: Tabla de indicadores de desempeño para manufactura. Adaptación propia de la tabla 10 de Arias.

La Adaptabilidad, así como el Tiempo Promedio de Planificación son irrelevantes en un contexto que no sea Manufactura, por lo que son descartados. El Tiempo en Tránsito junto al Tiempo Promedio de Despacho no se contemplarán, pues los asuntos de despacho no forman parte de las responsabilidades que se han asumido para el software WMS en construcción, de igual modo que en la tabla anterior, estos podrían ser contemplados para una segunda versión del software. No se considera prioritario agregar integraciones con otros sistemas, pero esto se debe a que existen otras opciones, como lo es agregar la funcionalidad, similar a lo que ofrecen los sistemas de logística unificada, que son los productos principales de los líderes de mercado como Manhattan Associates o Blue Yonder.

Categoría	**Nombre del Indicador**	**Descripción**
Calidad de la logística de salida	Tasa de Desabastecimiento (DT)	Promedio de ventas perdidas por el desabastecimiento de producto durante el periodo.

Calidad de la logística de salida	Número de Pedidos Pendientes (NPP)	Porcentaje de pedidos que están pendientes sobre el total de los pedidos del periodo.
Costos	Costo de Almacenamiento (CA)	Costo promedio de mantener inventario, es decir, mide el impacto del costo por m^3 almacenado o unidad almacenada.
Costos	Costo por Ajuste de Venta	Representa el porcentaje de lo que la empresa deja de ganar por vender productos a un precio de venta inferior al estipulado durante el periodo
Capacidad de respuesta	Ciclo de la Cadena de Suministro (TLP)	Tiempo promedio entre la emisión de la orden de compra de productos al proveedor hasta que los productos salen del almacén o son despachados al cliente final.
Capacidad de respuesta	Flexibilidad de Entrega (FLE)	Porcentaje de pedidos y/o órdenes con modificaciones en la entrega, solicitadas por parte del cliente que son entregados a tiempo.
Capacidad de respuesta	Flexibilidad del Producto	Tasa de adaptabilidad para medir el tiempo necesario en ajustarse a los cambios ocurridos en los gustos y necesidades de los clientes.
Capacidad de respuesta	Tiempo de Carga y Descarga	Tiempo promedio que transcurre mientras se carga el producto hasta que se descarga, se inspecciona y se registra la información en el sistema.
Capacidad de respuesta	Frecuencia de Entrega o Número de Entregas Diarias (TTP)	Promedio de pedidos entregados por día por trabajador con el objetivo de analizar su eficiencia en el desarrollo de sus tareas.
Capacidad de respuesta	Tiempo Promedio de Entrega de los Pedidos al Cliente (TE)	Tiempo promedio desde que se recibe una orden de venta por parte del cliente hasta que la orden es despachada al cliente final.
Eficiencia	Stock Fuera de Fecha (SFF)	Porcentaje de productos del total de productos del periodo que no se lograron vender a tiempo, porque se sobreestimó su venta y que originaron pérdidas a la empresa.
Eficiencia	Utilización de la Capacidad (UC)	Porcentaje de utilización del espacio de almacenamiento sobre su capacidad (m^2 o posiciones).
Eficiencia	Cobertura del Inventario (CI)	Tiempo que la cantidad de inventario existente permite cubrir las necesidades de los clientes.
Eficiencia	Rotación Inventario (RI)	Días de inventario o número de veces que se han renovado las existencias de un artículo durante el periodo.

Proveedor	Tasa de Fallo en la Entrega (TFP)	Porcentaje de órdenes entregadas por el proveedor que llegan incorrectamente tanto en cantidad, tiempo y calidad.
Proveedor	Cumplimiento de la Demanda (ODP)	Porcentaje de cumplimiento de la demanda de producto solicitada al proveedor durante el periodo.
Proveedor	Evaluación del Proveedor (EP)	Se refiere a la puntuación promedio de los proveedores de la empresa.
Proveedor	Ciclo de la Orden de Compra (TCP)	Tiempo promedio entre la emisión de la orden de compra de productos al proveedor y su recibimiento en las instalaciones de la empresa.

Figura 26: Tabla de indicadores de desempeño para comercio. Adaptación propia de la tabla 12 de Arias.

Por un lado, el Costo por Ajuste de Venta representa la posibilidad de crear ofertas por tiempo limitado y otro tipo de variaciones de precio, este tipo de ajustes no se han contemplado en el desarrollo aún, puesto que corresponde a un cambio crítico, pero que debe ser accesible para los usuarios administradores. Con esto en mente, este indicador puede ser implementado junto con esa funcionalidad. Por otro lado, las dos restantes como unidades de tiempo se escapan de la lógica presente en el software, puesto que sólo se admitirán cambios en el pedido mientras este no haya sido atendido aún, es decir, que no corresponde a un tiempo en la bodega, sino que es parte de la planificación y por tanto, un tiempo anterior.

Categoría	**Nombre del Indicador**	**Descripción**
Costos	Costo por Salarios (CS)	Costo financiero asociado al pago de los salarios de los empleados de la empresa
Eficiencia	Tasa de Utilización del Almacén (TUE)	Porcentaje de espacio utilizado por el layout en el espacio disponible en el almacén.
Eficiencia	Tiempo Promedio por Empleado (TPE)	Promedios de tiempo de la atención de una orden por parte de los empleados.
Proveedor	Costos de Abastecimiento (CCP)	Costo financiero de la creación de una orden de compra hacia un proveedor.
Proveedor	Valor a Favor (VFP)	Valor financiero excedente en función de los precios de venta con los precios de compra

Figura 27: Tabla de indicadores de desempeño adicionales para el WMS. Elaboración propia.

Anexo 2: Infografías de Manhattan Associates

Figura 28: Portada de las infografías "The Engaged Workforce". Extraído de los repositorios de Manhattan Associates.

Figura 29: Infografía corporativa sobre el propósito de la compañía. Extraído de los repositorios de Manhattan Associates.

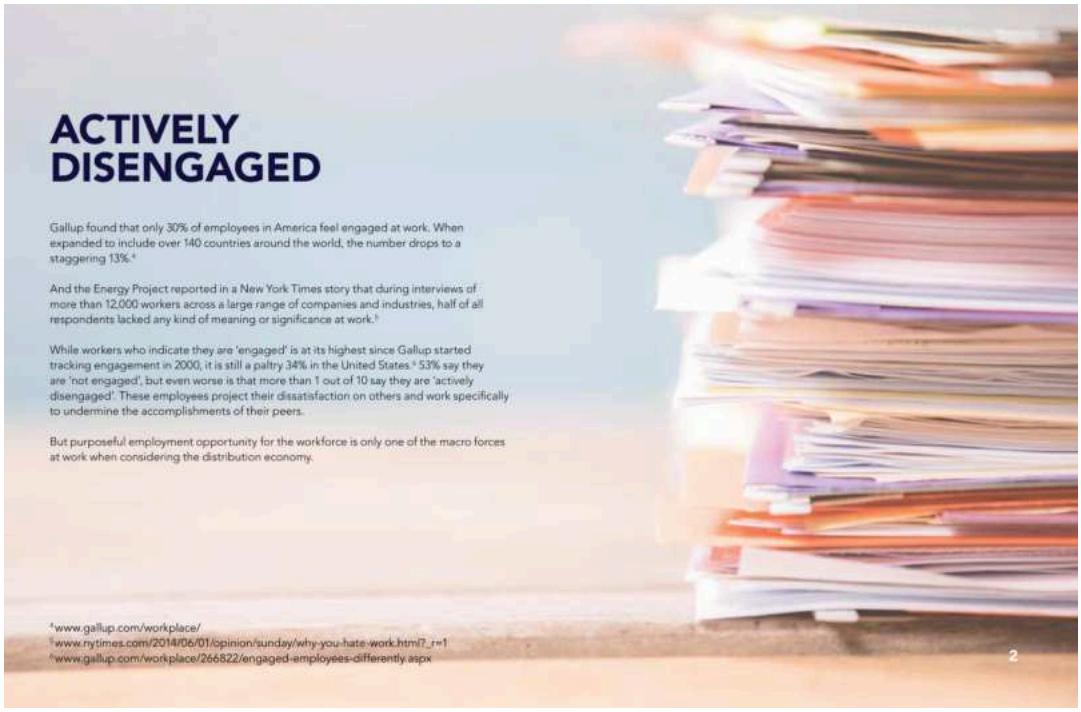

Figura 30: Infografía de antecedentes sobre la disposición laboral en América. Extraído de los repositorios de Manhattan Associates.

Figura 31: Infografía de antecedentes sobre las operaciones de bodega. Extraído de los repositorios de Manhattan Associates.

Figura 32: Infografía sobre los beneficios de la portabilidad del sistema. Extraído de los repositorios de Manhattan Associates.

Figura 33: Infografía sobre los beneficios de la gamificación del sistema. Extraído de los repositorios de Manhattan Associates.

Anexo 3: Playbacks Realizadas en el Plazo

- Con una primera reunión extraoficial el 1 de septiembre, se dió inicio a la iniciativa del desarrollo de un software de WMS, dada la necesidad de un software que ofreciera un kit de servicios más personalizado a las operaciones de menor envergadura.
- En la primera playback, el día 15 de septiembre, se hizo una presentación entre el equipo de desarrollo y el usuario Sponsor con el fin de comparar información y establecer un punto de inicio para investigar sobre las necesidades de una operación logística en contexto de MiPymes, así como del proceso y los intereses involucrados con una buena SCM en general. Con la muestra de un primer prototipo de la aplicación de picking, se generó un enfoque particular en apoyo a esta idea.
- Con la investigación concretada el 4 de octubre, se hizo la presentación de una primera iteración sobre la arquitectura y el modelo de datos para el problema. Desde esta playback se decidió el inicio del escrito, ambas iteraciones fueron aprobadas parcialmente, sobre lo cual se solicitó la esquematización de los procesos mediante diagramas de flujo, para una mayor comprensión de la solución propuesta.
- En la playback del 10 de octubre, el usuario Sponsor coincidió en la solución propuesta, solicitando una refactorización de los esquemas en función de la definición de hills que establecieran los objetivos para una primera versión de salida para el software. En este punto se decidieron 10 hills que contemplaban las 7 del escrito condensadas en 6, junto con otras 4 asociadas a mantenimiento, abastecimiento y post venta.
- El 18 de octubre, el equipo de desarrollo solicitó una reunión para aterrizar los requerimientos mediante la definición de un prototipo de MiPymes a la cual considerar como cliente final del software luego de revisar el estudio de mercado de Gartner. Posteriormente, en la playback del 23 de Octubre, se designó la operación de MiPymes objetivo en una sociedad con al menos 5 empleados luego de la revisión de algunos ejemplos revisados de la experiencia del usuario sponsor y la tesis de Mellado (2015) disponible en las referencias.
- En la playback del 30 de octubre, se definieron las 7 hills finales para ser diseñadas en el plazo de un mes y una semana restantes. Hasta antes de esta reunión se tenía un diseño preliminar de todos los diseños presentes en el escrito final, además de que se había iniciado la estrategia de utilizar el open source para la futura implementación.
- En la playback del 8 de noviembre, se presentó el proyecto open source ya iniciado con toda la documentación necesaria, además de comentar las tecnologías a utilizar para la implementación y el control de la calidad del software. También se hizo una revisión sobre el estado de avance en el diseño de la arquitectura final del escrito. Se validaron los progresos expuestos por parte del usuario sponsor.

- En la playback del 22 de noviembre, se revisó la arquitectura completada y se confirmó su inclusión al escrito y al proyecto. Se revisó el progreso del modelo de datos y de las vistas del software por medio de las explicaciones tras las decisiones tomadas en la arquitectura, en este punto se aterrizó más la idea respecto al constructor de layouts.
- En la playback del 4 de diciembre, se revisó el modelo de datos completado y se confirmó su inclusión al escrito y al proyecto. Se validaron más aspectos referentes a las vistas del software con las explicaciones sobre el modelo de datos, se observó la existencia de múltiples alternativas de despliegue y se optó por postergar esa decisión. Durante el análisis del modelo de datos, se consideró agregar un par de detalles adicionales a la arquitectura para mejorar las explicaciones de las propiedades de algunos esquemas del modelo de datos.
- En la playback del 11 de diciembre, se revisaron todas las vistas terminadas junto al escrito terminado, se confirmó la inclusión ya realizada de las vistas al escrito y se validó el análisis tras del cual se decidió el diseño, así como las características consideradas como principales en las distintas vistas.

Anexo 4: Módulos de la Arquitectura Adicionales

En este anexo se presentan las dos vistas de módulos que fueron omitidas en el inciso de la arquitectura de los Resultados del Proyecto, estas igualmente fueron elaboradas con la herramienta Structurizr. Las siguientes vistas cumplen un propósito funcional en la arquitectura, pues conectan los distintos módulos para que la información se transmita en base a eventos, que es la forma en que los proveedores de servicios en la nube facilitan seguir un plan Pay As You Use. Estos módulos corresponden al módulo de misiones, que se encarga de enviar y completar las misiones de los operadores del área de picking, y el módulo de bases de datos, que en realidad es solo el conjunto de servicios relacionados a una base de datos que deben ser compartidos por varios módulos a la vez. La excepción a estos es el servicio de misiones, que requiere la persistencia de las misiones recibidas para poder distribuirlas.

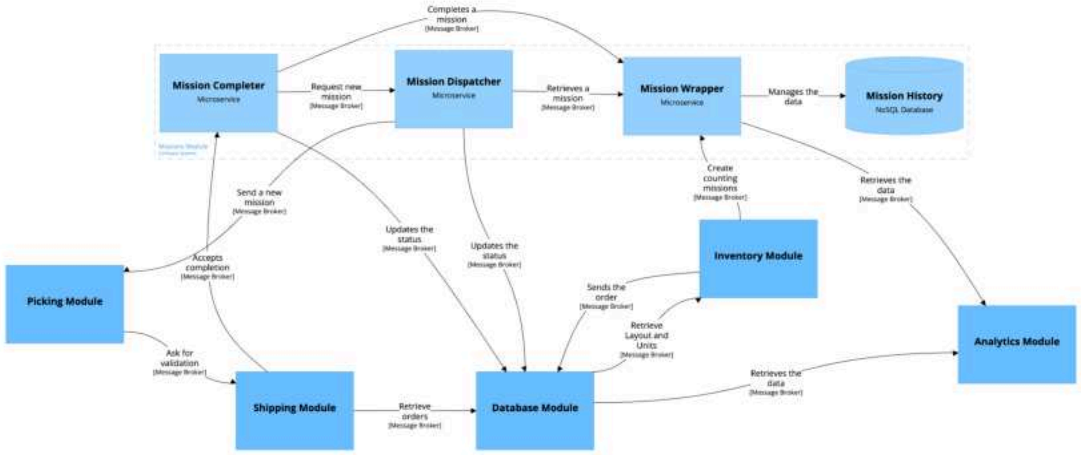

Figura 34: Diagrama de arquitectura de los contenedores en el módulo de misiones. Elaboración propia.

Este primer módulo, utiliza las nociones presentadas en el inciso del diseño de arquitectura respecto a la independencia de datos y los métodos de comunicación, pero igualmente introduce una lógica particular a este módulo. Haciendo caso omiso a la lógica de cómo la base de datos del historial de misiones es poblada, se tiene que este módulo recibe solicitudes de terminación de parte del módulo de shipping, cuando este es notificado por el módulo de picking. Luego, en este módulo se hacen efectivos los cambios necesarios para indicar al resto del sistema el cumplimiento de la misión, y por lo tanto, el despacho de la orden de compra que originó la misión en primer lugar. Una vez el cambio es efectuado, este mismo módulo se encarga de entregar su próxima misión al operador de picking que ha completado la misión, recuperando la siguiente misión en términos de prioridad. En un futuro, se podría requerir un servicio recurrente que mantenga la lista de prioridades actualizada,

según qué tanto haya escalado la operación, pero esto va más allá de lo que una operación de MiPymes podría necesitar.

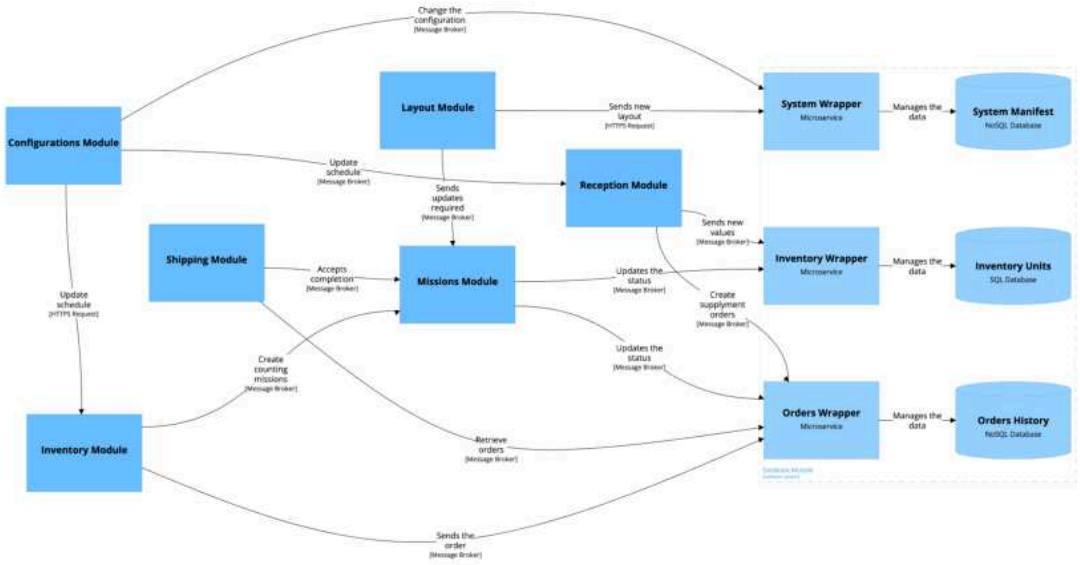

Figura 35: Diagrama de arquitectura de los contenedores en el módulo de bases de datos. Elaboración propia.

Este último diagrama solo presenta la lógica de independencia de datos que se explicó anteriormente, ya que no hay más lógica que simple almacenamiento en este módulo, pues funciona como un punto de conexión para el resto de la arquitectura y ya. Para observar la vista de la arquitectura completa con todos los módulos conectados, además de las vistas de contexto para todos los módulos, se puede revisar el despliegue en GitHub Pages siguiendo el siguiente enlace https://slc-wms.github.io/DSL-Architecture/main/. También se pueden seguir las indicaciones del repositorio DSL-Architecture en la organización para una instalación en local, la cual permite modificar las vistas a revisar, pudiendo explorar aún más la arquitectura modelada. Para acceder al repositorio desde dónde se realiza el despliegue, se puede visitar el siguiente enlace https://github.com/SLC-wms/DSL-Architecture.

Anexo 5: Detalles de Implementación de las Misiones

En este anexo se comenta más sobre el esquema de los documentos que representan una misión en el sistema, y como este se utiliza para satisfacer el requerimiento impuesto en la Hill Nº5 de incorporar la intercalación de tareas. Como se pudo observar en el enum de "Tipo de Misión", se contempla utilizar 4 tipos de misiones, las de reposición, las de movimiento, las de picking y las de conteo. Las de recepción serían equivalentes a las de reposición, por lo que no tienen un tipo dedicado, pero esto puede cambiar en el futuro si se requiere. Como se comentaba en el inciso del planteamiento del modelo de datos, los distintos tipos de misiones utilizan de manera diferente las propiedades de box_path y products. Estos corresponden a una forma de representar las cajas involucradas y los productos involucrados respectivamente, en todos los casos la propiedad box_path contempla el camino que debe seguir el operador en función de las cajas con las que debe interactuar.

La única excepción al caso anterior es la misión de movimiento, en dicho caso, la propiedad box_path siempre será una lista de largo par, puesto que los números impares representan las cajas que debe recoger, mientras que los pares el lugar al que las cajas que ha recogido deben ser reubicadas. Por esta misma razón, en este caso la propiedad products no contendría información. Por otro lado, en las demás misiones la propiedad products se utiliza para determinar los elementos que deben ser extraídos, contados o agregados en la caja involucrada dependiendo del objetivo de la misión. Volviendo al caso de recepción, habitualmente la propiedad de products podría ser irrelevante por el hecho de utilizar repisas dedicadas a la recepción como se recomendó en el Marco Metodológico. La manera en que estas misiones serán generadas, será mediante un sistema experto dedicado por cada tipo, como se observa con los microservicios que inicializan las misiones en algunos módulos de la arquitectura, como el módulo de inventario, o los microservicios presentes en el módulo de misiones observados en el Anexo 4.

Finalmente, estas misiones se priorizarán localmente por parte de la aplicación utilizada por los operadores de picking para hacer efectiva la intercalación de tareas, pero además se contempla una propiedad priority en el modelo de datos, lo que da pie a la incorporación de modelos inteligentes o inclusive wave planning desde la terminal del administrador. Por defecto, dada la naturaleza cloud native del proyecto, la cola de misiones se maneja con una lógica FIFO, de forma que se atiendan los pedidos de picking de forma ordenada, es decir, que la propiedad priority no se considerará por defecto, pero en función de los requisitos de la operación estas opciones pueden ser integradas por medio del uso de esta propiedad en la implementación del microservicio Mission Dispatcher del módulo de misiones presente en el Anexo 4.

Anexo 6: Análisis de Usabilidad de las Vistas

En este anexo se presenta el análisis individual de cada una de las aplicaciones o vistas propuestas como puntos de acceso para el sistema, de forma que se distinga cómo se han considerado las distintas características de usabilidad en cada uno de estos diseños. Cabe mencionar que los diseños se presentan en un formato de wireframe, por lo que algunas de las características podrían no observarse de forma clara, para ello se proveerá de una descripción acerca de la funcionalidad y navegación presentes en la vista. Antes de comenzar, hay algunas características que se cumple de forma innata en todos los diseños y de la misma manera, por lo que en los primeros párrafos se abordarán esas características en detalle, y luego se irá caso por caso. En general, la forma en que se resuelven las características de la misma manera tiene que ver con la interacción en base a notificaciones, pero eso se podrá observar a continuación.

Para los asuntos más cercanos a la comodidad del usuario con el sistema en términos de que el proceso es seguro y no tiene componentes maliciosas, es que se manejan las características como la transparencia de permisos o la confiabilidad de forma exhaustiva cuando hay datos de por medio, por ejemplo, como se ha mencionado en capítulos anteriores, cuando una operación está a punto de tener un impacto crítico en la operación, como ocurre con un cambio de layout, el usuario será notificado mediante una ventana de diálogo que el cambio que está por realizar tendrá un determinado impacto. Además, como se comentaba respecto de la transparencia de permisos en términos del sistema operativo, en las aplicaciones que leen códigos QR se solicitará el permiso de cámara, y para la aplicación de picking se solicitará acceso al magnetómetro y acelerómetro para la navegación en el almacén.

Sobre el mismo punto anterior, como parte de la confiabilidad del sistema, los distintos puntos de acceso cuentan con un método de autenticación con un factor presencial, lo que genera una capa de seguridad a nivel de las vistas. Además, se podrá asegurar el acceso a la aplicación mediante algún PIN en conjunto con sensores biométricos, que certifiquen la identidad del usuario. Adicionalmente, respecto a seguridad hay bastantes contramedidas planificadas para el sistema que consumen las vistas, por lo que en general es una característica que se ha tomado en cuenta exhaustivamente en el escrito. Para concluir las revisiones generales, la inmediatez se aborda de la misma manera, mediante alguna forma de notificación se dará a entender que existe una comunicación con el sistema en la nube, de forma que con cada interacción del usuario, los puntos de acceso o vistas, siempre tengan una respuesta predeterminada con el fin de esperar los cambios globales.

Comenzando por la dashboard, la completitud se presenta en la forma de que el sistema completo se encuentra documentado en la vista, lo que crea una experiencia del tipo conversación con su usuario, y es esta experiencia la que en conjunto con una estructura apta para la accesibilidad en gestos proporcionados por los sistemas operativos la que da una

sensación de interactividad, pero como además se trata de una herramienta productiva, se han considerado temas de accesibilidad más relacionados a la flexibilidad del sistema por medio de accesos rápidos en el teclado, una navegación independiente del ratón y las configuraciones locales de la dashboard. En dichas configuraciones se pretende dar la posibilidad de editar el layout de los elementos en pantalla, además de editar o agregar accesos del teclado. Respecto a la aprendibilidad y al diseño responsive ya se ha comentado en el capítulo de Resultados del Proyecto. Este mismo análisis aplica para las vistas de shipping y recepción, pues tendrán el mismo tipo de construcción, como se podrá haber visto en los wireframes.

Finalmente, la aplicación de picking tiene su foco en la gamificación, y es mediante el diseño focalizado en la interactividad que la vista presenta una interfaz sencilla con instrucciones directas y sencillas en cada una de sus ventanas, lo que favorece su aprendibilidad. Adicionalmente, para mantener ese flujo único se hace uso de accesibilidad en gestos, por ejemplo, mantener presionada la pantalla para completar un paso. Pero es con ese mismo flujo único que hay un detalle, puesto que se considera como que el sistema no puede fallar, ahí es donde entra la flexibilidad, no solo la vista permite abortar la misión en cualquier punto, sino que el cierre de la aplicación determinará la cancelación de la misión, de esta forma el sistema puede recuperarse de ello. El diseño responsive se contempla para distintas resoluciones, aunque solo para vistas en portarretrato, pues el diseño así lo requiere. Finalmente, la completitud es parte de la interactividad lograda, con las instrucciones mencionadas, por lo que también se encuentra considerada en el diseño.

Anexo 7: Estimación de Costos de Infraestructura

Recopilando los requerimientos del sistema desde el capítulo de Arquitectura de Microservicios, donde se presenta la infraestructura necesaria para satisfacer las hills del proyecto, se tiene que el sistema completo utiliza aproximadamente 1 servicio de firewall, 1 servicio de balanceo de carga, 25 colas de mensajes, 3 servicios de evento periódico, 3 instancias de bases de datos no relacionales, 2 instancias de bases de datos relacionales y otra instancia pequeña, 3 instancias de ejecución de uso moderado, 9 instancias de uso general, y 13 de uso intensivo. Esta aproximación se basa en que aún no se ha hecho la separación de los microservicios en funciones cloud como se comenta en los párrafos finales de la Discusión. Considerando el uso que le dará cada usuario, se estima que los operadores de picking deberían tener una interacción más continua con el sistema, por lo que se los considerará como una cota superior respecto al uso del sistema en términos del número de operaciones.

Asumiendo que un operador completa una orden de picking en 4 minutos, y una misión de mantenimiento en 2 minutos, da un total de 10 ciclos por hora. Con la jornada laboral de 45 horas semanales que hay en Chile, se tendrían 450 ciclos en la semana, es decir, aproximadamente 1800 ciclos mensuales. Redondeando hacia arriba para asegurar, se podrían esperar 2000 ciclos mensuales por operador como máximo. De los 6 minutos del ciclo, parte del tiempo la aplicación estará a la espera de la interacción del usuario, haciendo otro supuesto pesimista, se asume que 4 de los 6 minutos estará accediendo al sistema de forma activa. Con esto en mente, hay un total de 240 segundos de cómputo, lo que con un muy bajo grado de optimización se puede considerar como 480 operaciones por ciclo, redondeando hacia arriba nuevamente, se tienen 500 operaciones por ciclo. Esto completa la aproximación a un total de 1 millón de operaciones por usuario mensualmente como máximo.

Bajo la misma línea de esta consideración, se han asumido 500 Kb en comunicaciones por operación, resultando un total de 5 Gb por usuario para el sistema. A modo de referencia, esos 500 Kb representan un total de 500 mil caracteres, capaces de almacenar más de 300 registros de todos los indicadores del sistema, por lo que hay un buen margen de error en esta estimación también. Adicionalmente, se estiman 500 Gb de almacenamiento para el sistema completo, entendiendo que salvo por el módulo de misiones y el de analíticas, el resto de esquemas son mucho más dinámicos, lo que quiere decir que, de esos 500 Gb, casi todo es utilizado por un solo tipo de esquema, y ninguno es particularmente pesado. Recordando el ejercicio anterior respecto a los 500 Kb, en 1 Gb caben más de 500 mil millones de registros con todos los indicadores del sistema, que es el esquema más pesado que se va a guardar en estas bases de datos, por lo que esos 500 Gb darán una persistencia muy flexible a las futuras necesidades del negocio.

En conjunto con todas estas estimaciones, existen valores fuera de alcance hasta que el sistema no se encuentre en línea, además de que las calculadoras de precios de los servicios cloud contemplan un márgen de error, lo que implica hacer la observación de que el costo final de este ejemplo puede diferir en gran medida del costo real. Ahora bien, asumiendo el cálculo para una empresa de 10 personas con todas las consideraciones anteriores y que elige al proveedor Google Cloud Platform (GCP) para alojar su infraestructura hoy 10 de enero de 2024, obtiene los siguientes costos mensuales aproximando desfavorablemente:

- $125 USD por el servicio de firewall
- $30 USD por el servicio de balanceo de carga
- $300 USD por las 25 colas de mensajes
- $0 USD por los 3 servicios de evento periódico
- $100 USD por las 3 instancias grandes NoSQL (300 Gb)
- $95 USD por las 2 instancias grandes y la pequeña de SQL (205 Gb)
- $0 USD por las 3 instancias computacionales de uso moderado
- $27 USD por las 9 instancias computacionales de uso general
- $156 USD por las 13 instancias computacionales de uso intensivo

Los precios son obtenidos de la siguiente cotización en la calculadora de precios de GCP https://cloud.google.com/products/calculator/#id=baa4a2d4-c3fd-44c3-ae6e-9b19e7947b96, además cabe mencionar que el costo $0 de los servicios de evento periódico y de las instancias de uso moderado, se debe a los tier gratuitos que tiene GCP, los cuales existen igualmente bajo otras condiciones en los demás proveedores. Con esto, el total de mantener el sistema con unos requerimientos bastante por encima de como funcione la operación real se resumen a $833 USD mensuales para 10 personas, lo que en términos de la problemática planteada en la introducción, representa una reducción del costo por usuario en un 10% como mínimo. Y es relevante hacer énfasis en la aclaración anterior, puesto que con el modelo Pay As You Use, este porcentaje crece en función de cuánto se utilice el sistema en realidad, lo que dependiendo de la operación, se podría incrementar hasta incluso al 100%, aunque esto es seguramente improbable en cualquier empresa con 5 o más empleados.

El mismo ejercicio pero en una empresa de 5 empleados se observa en esta cotización https://cloud.google.com/products/calculator/#id=f08b5afd-2fa1-4505-a276-0ee34895bee6. Y haciendo los mismos cálculos de redondear y multiplicar, se obtiene un costo máximo de $480 USD, que representa un ahorro menor al del ejemplo anterior, pero sigue significando un ahorro y el pasar a tener la soberanía sobre los datos de la operación. Además de que este costo seguramente será más bajo en función del modelo de negocio que tienen los CSP líderes del mercado y de que el uso estimado en estos ejemplos está muy por encima del uso que le darán las MiPymes.