UNIVERSIDAD DE ANTIOQUIA

CLIENTE INTERMEDIO ENTRE E-COMMERCE Y ERP PARA @PC

Autora

Estefany Muriel Cano

Universidad de Antioquia

Facultad de Ingeniería

Departamento de Ingeniería de Sistemas

Medellín, Colombia

2021

CLIENTE INTERMEDIO ENTRE E-COMMERCE Y ERP PARA @PC

Estefany Muriel Cano

Tesis o trabajo de investigación presentada(o) como requisito parcial para optar al título de:
Ingeniera de Sistemas

Asesores (a):
Astrid Duque Ramos, Doctora en Ingeniería Informática
Jonathan Diosa Giraldo, Magíster en Ingeniería de Sistemas

Universidad de Antioquia
Facultad de Ingeniería
Departamento de Ingeniería de Sistemas
Medellín, Colombia
2021

Índice

Índice de figuras

Índice de tablas

1. Resumen

Actualmente el número de personas que prefieren hacer sus compras por internet incrementa constantemente debido a la facilidad de compra que esta modalidad presenta, muchas empresas y personas han optado por este modelo de e-commerce, una de estas empresas es el grupo empresarial @PC, quien además de tener puntos físicos para la venta de sus productos tiene diferentes tiendas virtuales para la distribución de los mismos. El manejo de las tiendas en Mercado Libre y Vtex es un proceso que requiere de tiempo, ya que al ser tiendas independientes se tiene que procesar todas las órdenes de cada tienda antes de cambiar a la siguiente tienda para ser procesada. Es por esto que en @PC surgió la necesidad de una plataforma que permita manejar todas las tiendas e-commerce que tienen desde el mismo lugar, sin necesidad de entrar a cada tienda de Mercado Libre o de Vtex en sus respectivas plataformas independientes. Centralizar el manejo de las órdenes hace la gestión de estas más rápida y fácil de administrar a través de los diferentes estados que tienen desde la compra hasta la facturación y entrega del producto, además de realizar la conexión del ERP de la empresa con la plataforma para crear los terceros y las facturas de forma casi automática. Aunque la plataforma ya se adquirió hace algunos años y está funcionando actualmente, está desarrollada con tecnologías antiguas y técnicas de programación de baja calidad, debido a esto, se decide hacer la implementación de un nuevo sistema que permita la gestión de órdenes, pero esta vez con tecnologías que van más a la vanguardia y siguiendo diferentes arquitecturas y patrones que permitan la mantenibilidad y escalabilidad del sistema, además hacer mejoras respecto a algunos flujos de negocio que se manejaban en la aplicación y un rediseño de los componentes visuales. En este proyecto se realiza el frontend para el sistema que permite gestionar las órdenes de los e-commerces Vtex y Mercado Libre desde el servidor local de la empresa, desplegado en dos ambientes: desarrollo y producción, para facilitar las futuras implementaciones.

Palabras clave: Frontend, React, CircleCI, Docker, E-commerce

2. Introducción

El aprovechamiento de la tecnología ha permitido compartir y entregar información, además de contribuir al desarrollo de diferentes herramientas que virtualiza algunas actividades cotidianas que solíamos hacer presencialmente, es por esto que actividades como la compra y la venta de productos han logrado quedar totalmente virtualizadas para quienes así lo quieran. Los continuos incrementos de ventas indican que los e-commerce tienen un gran potencial, el éxito de compañías como Amazon, Alibaba y Groupon es un ejemplo de la implementación de este método [18]. La popularidad de las ventas online ha causado que empresas con instalaciones físicas busquen ampliar sus servicios a la virtualidad, convirtiéndose en empresas online y offline [11], esto ha hecho que muchas compañías tengan que adecuar sus procesos y recursos internos para afrontar esta demanda online y poder suplir las necesidades de sus clientes en tiempos considerables.

El grupo empresarial @PC se compone de diferentes tipos de empresas como Experimentality, @PC, entre otras. @PC es una compañía dedicada a la distribución de tecnología, software y hardware, esta compañía empezó hace algunos años con ventas en puntos físicos pero gradualmente se fue uniendo a la era de las compras online y actualmente uno de sus canales de distribución más importante es el online, en plataformas e-commerce como Vtex, Linio, Mercado Libre, etc. Para poder combatir la alta demanda de los pedidos que ingresan por estos canales y facilitar la gestión por parte de las personas encargadas en esta labor, la empresa cuenta en la actualidad con un sistema de gestión de tiendas online en el que se pueden visualizar todas las órdenes que se generan y gestionarlas a través de los diferentes estados por los que pasa, como son: en proceso, lista para entrega, facturada, entre otros. Adicionalmente, desde la plataforma se podían actualizar automáticamente los inventarios de las tiendas, pero esto ha dejado de funcionar, así como la facturación automática con el ERP.

El sistema se encuentra integrado con Mercado Libre,Vtex y Linio, pero en Linio sólo se permiten ver las órdenes, aún no se tiene la función de gestión de órdenes desde el sistema para este canal. El equipo de e-commerce de @PC usa constantemente la herramienta que está desarrollada en tecnologías muy antiguas y presenta constantes impedimentos en el acoplamiento de nuevas tiendas, la concurrencia o actualización del mismo. Por lo anterior, surge la necesidad de hacer una nueva implementación de la plataforma tanto frontend como backend para actualizar las tecnologías, rediseñar la arquitectura, hacerla escalable y mantenible, además de hacer un cambio en el diseño de la interfaz que se tiene actualmente para mejorar la usabilidad y entendimiento de las funcionalidades del sistema.

El alcance planeado del sistema para esta práctica empresarial es realizar el frontend para la gestión de las órdenes de compra para mercadolibre y vtex, desde que ingresan como nueva hasta que son facturadas y entregadas al cliente, sin embargo, se tuvo un impedimento ligado al ERP de la empresa por lo que no se pudo hacer el paso de facturación automática para las órdenes. Para la implementación del frontend de la aplicación, se utilizó el desarrollo por compo-

nentes que ofrece React y que permite tener el código más modularizado para así tener un mejor desempeño y un mejor mantenimiento a largo plazo [**react**].

3. Objetivos

3.1. Objetivo general

Optimizar el proceso de gestión, trazabilidad y administración de las tiendas online, llevado a cabo por el equipo ecommerce de @PC, disminuyendo así los tiempos de gestión para cada tienda, al igual que la mano de obra requerida para la administración.

3.2. Objetivos específicos

- Analizar y definir los nuevos y antiguos flujos de procesos del sistema dentro del negocio.
- Desarrollar el frontend para un sistema que servirá de cliente intermedio para la gestión de órdenes de las tiendas e-commerce (Vtex - Mercado Libre) en @PC.
- Validar los requerimientos técnicos y de negocio.
- Desplegar y entregar el sistema con los resultados finales.

4. Marco teórico

En esta sección se explicarán los conceptos utilizados para el desarrollo del proyecto, como React que fue la librería javascript utilizada para el frontend, Docker que fue el contenedor en donde se encapsuló la aplicación para que por medio de integración y despliegue continuo con la ayuda de Circle CI y Gitflow, se alojara el proyecto en los servidores de la empresa, todo esto siguiendo el marco de trabajo SCRUM el cuál se administró por medio de la plataforma Jira:

4.1. React

Es una biblioteca de javascript que permite crear interfaces de usuario de forma sencilla, está basado en componentes encapsulados que manejan su propio estado, al estar la lógica de los componentes escrita en javascript y no en plantillas, se puede pasar datos de forma sencilla a través de la aplicación y mantener el estado fuera del DOM(Document Object Model) [8]. React fue creado por ingenieros de facebook para resolver algunos retos de creación de interfaces complejas con data que cambia a través del tiempo e introdujo nuevos paradigmas, cambiando la forma de crear aplicaciones e interfaces javascript escalables y mantenibles [7]. La filosofía de react es trabajar por medio de componentes,

y cada componente tiene el método render() que genera un DOM intermedio el cual luego se convierte en el HTML DOM [1], en la figura 2 podemos ver como la conversión del DOM intermedio queda totalmente igual al DOM HTML.

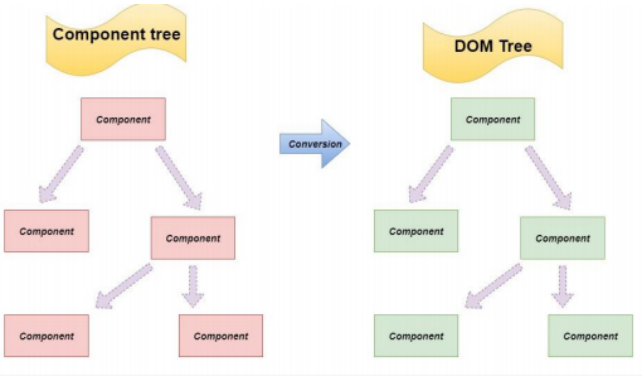

Figura 1: Conversión del DOM intermedio al DOM HTML
[1]

4.2. Docker

Es una herramienta que permite la creación de contenedores para empaquetar el código y las dependencias de una aplicación y que se ejecute rápidamente en el entorno que deseemos. Cada contenedor tiene todo lo necesario como código, herramientas del sistema, bibliotecas y configuraciones. Los contenedores aíslan el software y garantizan que funcione de manera uniforme sin importar donde se ubique [6].

4.3. Integración Continua

Es una práctica de desarrollo de Software donde los miembros de un equipo integran su desarrollo constantemente, usualmente cada persona fusiona su código con los otros al menos una vez al día. Cada integración es verificada por un despliegue automático que incluye tests para detectar errores de integración lo más rápido posible y esto contribuye a una reducción importante de problemas en los despliegues [13]

4.4. Despliegue Continuo

Es un proceso de despliegue de sotfware que utiliza pruebas automatizadas para validar que los cambios en un código son correctos y estables para la implementación inmediata en un entorno de producción. Esto ha solucionado los problemas que se tenían antes de poder mover el código a otra máquina para desplegar en los ambientes de producción o desarrollo y verificar si las tareas se

cumplieron a cabalidad[15], en la figura 3 vemos las fases por las cuales pasa cada ciclo del despliegue continuo, que incluye desde los builds hasta los test y el deploy a los diferentes ambientes que se tienen.

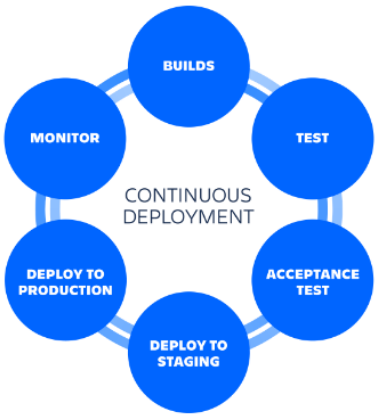

Figura 2: Proceso del despliegue continuo
[15]

4.5. Jira Software

Jira es una herramienta que sirve para ayudar a equipos a gestionar el trabajo en todo tipo de casos de uso, desde la gestión de requisitos hasta el desarrollo de software de forma ágil, por ejemplo, para el caso de equipos que usan metodologías ágiles. Jira proporciona tableros de scrum y kanban para la gestión de tareas y flujos personalizables de trabajo[2], estos tableros ofrecen visibilidad de cada parte del trabajo y de cada integrante del equipo, para llevar a cabo un registro del progreso, obstáculos encontrados y así supervisar la productividad

4.6. CircleCI

Es una herramienta que permite la integración continua de nuestro proyecto, además de optimizar las compilaciones para aumentar la velocidad, se puede configurar el paralelismo en los proyectos para ejecutar jobs más rápidos y configurar el almacenamiento en caché para utilizar datos de jobs anteriores en los jobs actuales, Circle CI ejecuta cada job en un contenedor o máquina virtual independiente, cada cambio de código desencadena las pruebas automatizadas en un nuevo contenedor [5], en la figura 4 se puede ver como CircleCI se integra con el repositorio y posteriormente realiza la orquestación del proyecto para desencadenar en el despliegue.

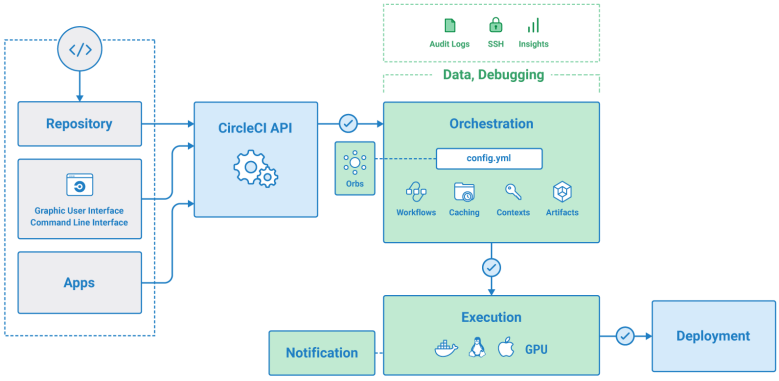

Figura 3: Flujo de despliegue con CircleCI
[5]

4.7. Metodología ágil

En esta metodología los proyectos suelen subdividirse en proyectos más pequeños mediante una lista de características, cada subproyecto es tratado de forma independiente y se desarrolla un subconjunto de características durante un periodo de tiempo, se tiene una comunicación constante con el cliente. Los proyectos realizados bajo esta modalidad son colaborativos y son abiertos a cambios, se realizan entregas constantes y debido a esto se puede obtener una retroalimentación que permiten hacer mejoras frecuentemente.En el desarrollo de software ya es muy común que se utilice este tipo de metodología, pues es flexible y pueden ser modificada para que se ajuste a la realidad de cada equipo y proyecto. Hay varias características que hacen que las metodologías ágiles se diferencien de las metodologías tradicionales[4](ver Tabla 1).

Metodologías tradicionales	**Metodologías ágiles**
Predictivos	Adaptativos
Orientados a procesos	Orientados a personas
Proceso rígido	Proceso flexible
Se concibe como un proyecto	Un proyecto es subdividido en varios proyectos más pequeños
Poca comunicación con el cliente	Comunicación constante con el cliente
Entrega de software al finalizar el desarrollo	Entregas constantes de software
Documentación extensa	Poca documentación

Tabla 1: Diferencias entre metodologías tradicionales y metodologías ágiles

Si bien el SDLC(Software Development Life Cycle) puede convertirse en la metodología Cascada para desarrollar software cuando sus pasos son ejecutados de forma secuencial, la verdad es que al ser un marco de trabajo puede adaptarse a diferentes necesidades y desde diferentes puntos de vista. Algunas metodologías derivadas del SDLC son por ejemplo la ya mencionada metodología en cascada, la metodología en espiral, el Modelo-V, el prototipado rápido y el método incremental [4].

Existe un manifiesto ágil, donde se presentan algunos principios que diferencian un proceso ágil de uno normal, los dos primeros principios resumen gran parte del espíritu ágil, el resto se refieren al equipo de desarrollo y las metas a seguir, a continuación se presentan los 12 principios [9]:

1. La prioridad es satisfacer al cliente mediante tempranas y continuas entregas de software que le aporte un valor.
2. Dar la bienvenida a los cambios. Se capturan los cambios para que el cliente tenga una ventaja competitiva.
3. Entregar frecuentemente software que funcione desde un par de semanas a un par de meses, con el menor intervalo de tiempo posible entre entregas.
4. La gente del negocio y los desarrolladores deben trabajar juntos a lo largo del proyecto.
5. Construir el proyecto en torno a individuos motivados. Darles el entorno y el apoyo que necesitan y confiar en ellos para conseguir finalizar el trabajo.
6. El diálogo cara a cara es el método más eficiente y efectivo para comunicar información dentro de un equipo de desarrollo.
7. El software que funciona es la medida principal de progreso.
8. Los procesos ágiles promueven un desarrollo sostenible. Los promotores, desarrolladores y usuarios deberían ser capaces de mantener una paz constante.
9. La atención continua a la calidad técnica y al buen diseño mejora la agilidad.
10. La simplicidad es esencial.
11. Las mejores arquitecturas, requisitos y diseños surgen de los equipos organizados por sí mismos.
12. En intervalos regulares, el equipo reflexiona respecto a cómo llegar a ser más efectivo, y según esto ajusta su comportamiento.

4.8. SCRUM

Es un marco de trabajo creado por Ikujiro Nonaka e Hirotaka Takeuchi y hace referencia a un tipo de formación que se realiza en el rugby para aumentar la productividad y flexibilidad del equipo[16], Scrum fue desarrollado a principios de la década de 1990 y ayuda a las personas, equipos y organizaciones a generar valor a través de soluciones que se adaptan a un problema complejo, básicamente SCRUM hace uso de un Scrum Master para que mantenga los siguientes elementos en el entorno donde se está aplicando el marco de trabajo:

1. Un Product Owner ordena el trabajo de un problema complejo en un Product Backlog.
2. El Scrum Team convierte una selección del trabajo en un incremento de valor durante un Sprint.
3. El Scrum Team y sus interesados inspeccionan los resultados y se adaptan para el próximo Sprint.
4. Repita.

Scrum emplea un enfoque iterativo e incremental para optimizar los posibles riesgos y controlarlos, para esto se combinan cuatro eventos formales para la inspección y adaptación dentro de un evento contenedor llamado Sprint [10], como se muestra en la figura 1 en SCRUM se sigue un ciclo iterativo en donde se tiene la planeación, la review y la restrospeciva del trabajo realizado por el equipo.

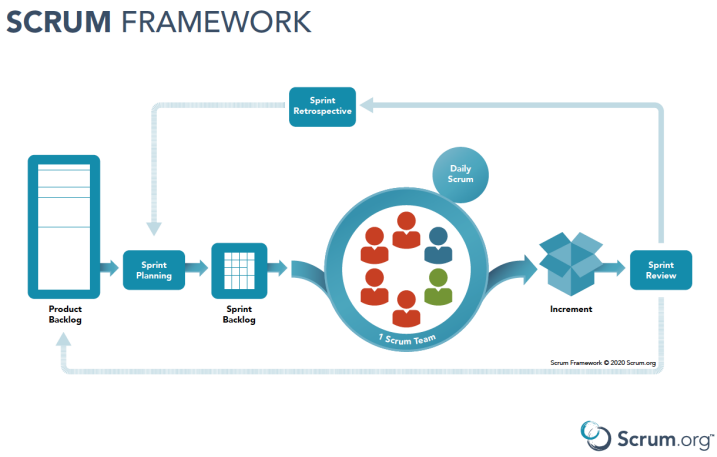

Figura 4: Ciclo Scrum
[17]

4.9. Gitflow

Es un flujo de trabajo de git que es muy útil cuando se hace desarrollo continuo de software y prácticas DevOps. Se basa en un modelo de ramificación donde se asignan roles específicos a las ramas y se decide como interactúan entre ellas, se tienen dos ramas principales, una llamada master que es la que va a producción y una llamada develop, la cual va para el ambiente de desarrollo, de la rama develop se sacan otras ramas llamadas features para las nuevas funcionalidades y releases para sacar lo que ya se tiene listo de develop a producción. Cuando se producen errores en producción, se crean ramas llamadas hotfix para solucionarlos [3].

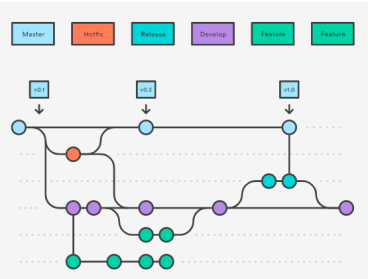

Figura 5: Gitflow
[3]

5. Metodología

Para el desarrollo del proyecto se hizo el levantamiento de requisitos con los implicados de @PC y se definieron 5 grandes fases para desarrollar durante las prácticas, las cuales se dividieron en el diseño y análisis del sistema, en donde se definieron los requisitos funcionales y no funcionales del sistema y se plantearon las decisiones arquitectónicas para el software. La siguiente fase fue la codificación y desarrollo, en donde se hizo la implementación de las funcionalidades de la aplicación. Luego siguió la fase de integración, en donde se conectó la plataforma con los servicios externos necesarios para su correcto funcionamiento, por último se hicieron dos fases finales de validación y despliegue para dejar todo a disposición del cliente.

La implementación del frontend se hizo con React, la figura 6 presenta el flujo general del sistema, donde se puede observar que la conexión del back se realizó por medio de la interfaz de programación REST, que fue la encargada de hacer la conexión con los e-commerce Vtex y Mercado Libre a través de las API de cada uno de estos e-commerce, dejando centralizado todo su manejo en una misma API, el ERP de la empresa y una base de datos en el servidor de la empresa.

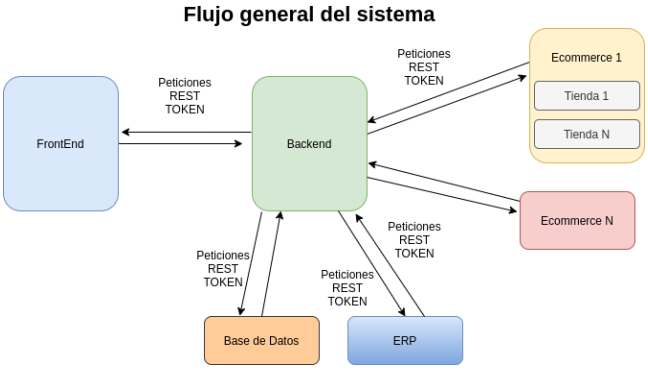

Figura 6: Flujo general del sistema

La arquitectura del frontend se presenta en la figura 7 donde se muestra el uso de React como librería de javascript frontend y se utilizó el concepto de desarrollo por componentes y contenedores de React, en donde un contenedor abarca diferentes componentes para formar una pantalla de la aplicación. Se desarrollaron diferentes componentes de uso único y compartido entre los contenedores, en el manejo de data entre componentes se hizo uso del context de React para crear el state de la aplicación.

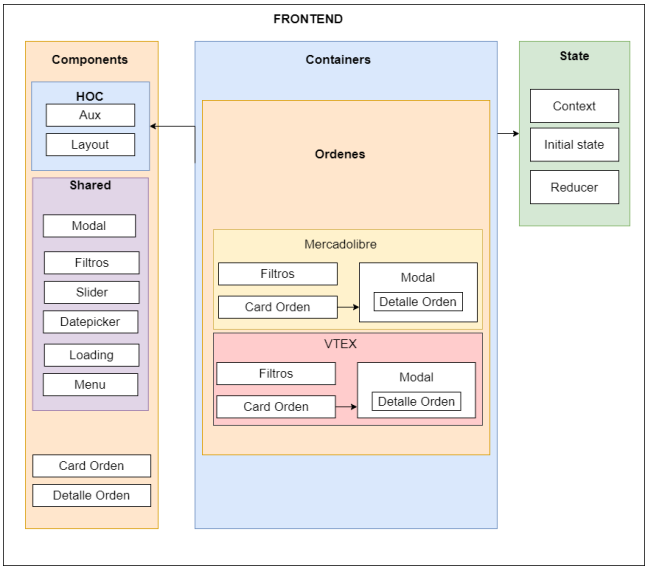

Figura 7: Diagrama de arquitectura del frontend

Se decidió trabajar con React para aprovechar las bondades que ofrece el desarrollo por componentes, ya que cada componente se encarga de su renderizado

y esto hace que el performance de la aplicación aumente[1], además de hacer más fácil el entendimiento de la aplicación y su mantenimiento al estar todo modularizado y con poca dependencia entre sí.Para la gestión del repositorio del proyecto, se hizo uso de Github como herramienta de control de versiones y alojamiento de código, además se hizo uso de Gitflow para la gestión de ramas y poder hacer el desarrollo continuo de software y así poder desplegar de forma automática con la rama master a producción y con la rama develop al ambiente de desarrollo.

El marco de trabajo mediante el cual se desarrolló el proyecto fue una adaptación de SCRUM que es una metodología ideal para los entornos en los que se tiene incertidumbre, autoorganización, control moderado y transmisión del conocimiento. Una de las características de SCRUM es la creación de ciclos breves para el desarrollo, los cuales se conocen como iteraciones, pero para nuestro marco de trabajo fueron denominados como "Sprints" [14]. En el desarrollo del proyecto se tuvieron 8 sprints de 11 días hábiles cada uno en donde cada que se iniciaba un sprint se realizaba una planning con el fin de ver qué objetivos se iban a cumplir el próximo sprint y se realizaba una review para mirar cuáles objetivos se cumplieron en el sprint anterior.

En busca de gestionar el flujo de trabajo, avances, impedimentos y metas cumplidas se hizo uso de la herramienta para gestión de proyectos Jira, la cual es usada comúnmente en proyectos ágiles ya que nos provee una alta configuración y creación de ítems [12], como requerimientos del sistema, requerimientos de software o requerimientos de cualquier tipo, además de poderle asignar un status a cada tarea, una prioridad, criterio de aceptación, etc.

El proyecto del cliente intermedio para @PC se hizo en conjunto con un desarrollador backend, se sacaban grandes épicas de los desarrollos que se iban a hacer en el sprint y luego se creaban las subtareas de front y de back para llevar a cabo el objetivo señalado en la épica y además poder tener un mapeo del trabajo de cada desarrollador y de las tareas faltantes para culminar el objetivo propuesto y que el product owner pudiera gestionar las prioridades en el proyecto.

6. Resultados y análisis

Como resultado de estas prácticas se logró realizar el frontend del cliente intermedio entre e-commerce y ERP en la empresa @PC para el manejo de los estados de órdenes de Vtex y Mercado Libre. La aplicación cuenta con una pantalla de inicio sencilla en donde se describe brevemente el propósito de la misma y con un menú lateral en donde se permite elegir la opción de ingreso a las órdenes, el menú es expansible, ya que por el momento solo se desarrolló un módulo de la aplicación, pero faltan más módulos que se desarrollaran en trabajo futuro con otros practicantes o colaboradores de la empresa.

En la figura 8 vemos la pantalla inicial de la aplicación, en donde los colores que se utilizaron fueron los del manual de marca de la empresa, el cual fue facilitado para el desarrollo.

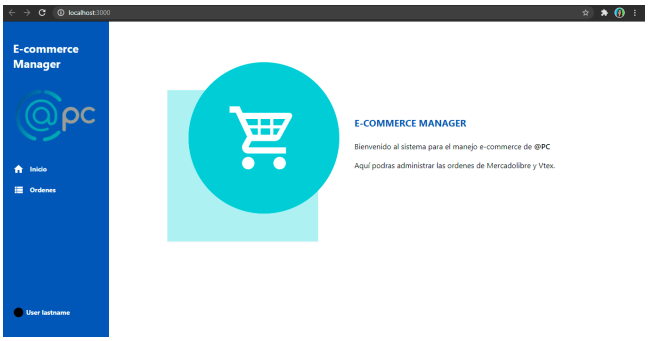

Figura 8: Pantalla inicial de la plataforma

En la pantalla de órdenes por defecto se muestran las órdenes de Vtex con unos filtros iniciales los cuales buscan las órdenes del día actual que tengan estado "nueva" y en todos los seller actuales. Cuando la búsqueda arroja resultados, la información más relevante de la orden y que genera valor para el negocio aparece en una card como vista previa, se puede ver en la figura 9.

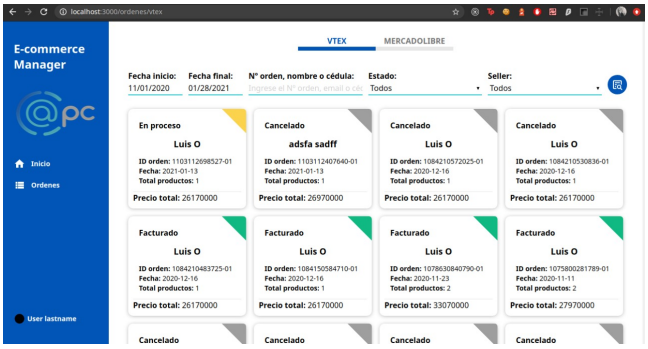

Figura 9: Pantalla de órdenes vtex

En Vtex y mercadolibre los estados de las órdenes y su manejo son diferentes, por lo que algunas cosas como el flujo de estados y filtros difieren de como están en Vtex, para esto se utiliza el mismo componente pero con una variable en donde se le indica cual e-commerce se va a renderizar para que oculte o muestre información según sea el caso. Para cambiar entre e-commerces se dispuso de unas tabs en la parte superior, en donde se puede hacer un switch rápidamente al e-commerce requerido y dentro de cada uno existe un filtro de seller o tienda según corresponda. Para gestionar una orden basta con darle click a la card para que se despliegue un modal con la información del usuario, de la compra y el flujo de estado en los cuales está la orden y a los cuales puede ser administrada, como se muestra en la figura 10. Cuando no se encuentran resultados que coincidan con los filtros en la figura 11 podemos observar la pantalla que se muestra.

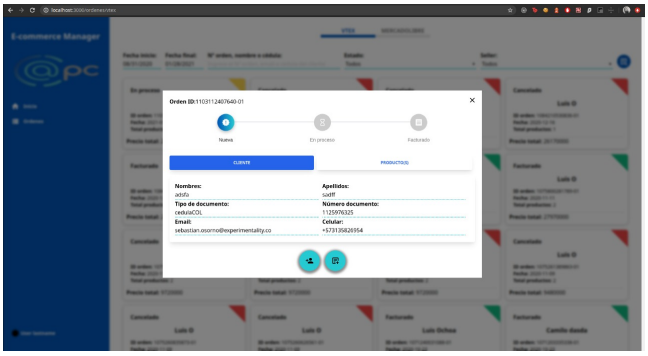

Figura 10: Detalle de la orden

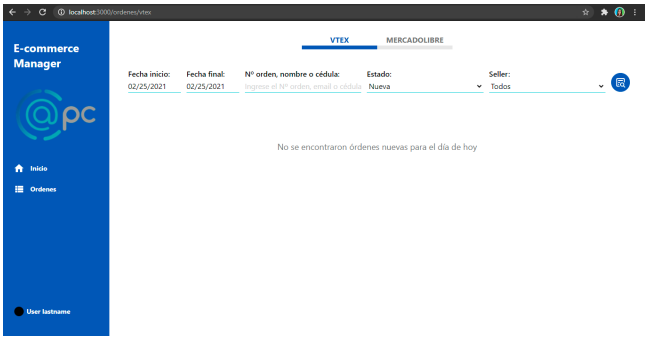

Figura 11: Pantalla cuando no se encuentran órdenes

Se designaron dos ambientes de desarrollo; uno para producción y otro para desarrollo. Esto con el fin de no mezclar ambos ambientes y poder hacer los desarrollos a futuro sin intervenir en el ambiente real. Hacer esto fue posible gracias a que desde el inicio se quiso tener un enfoque de integración y despliegue continuo, se logró por medio de la herramienta CircleCI desplegando todo en el servidor de la empresa. Haciendo esto se pudo cumplir con uno de los requerimientos de la empresa que era tener esta plataforma desplegada en sus servidores y que solo fuera accesibles desde su red local.

Para desplegar el proyecto se hizo uso de Docker, por lo que se creó un archivo Dockerfile dentro del repositorio con las configuraciones necesarias para ejecutar el proyecto, como la herramienta utilizada fue CircleCI se creó un archivo config.yml con las configuraciones necesarias y unos archivos .bash para cada uno de los ambientes que serán los encargados de preparar el servidor para el despliegue de los contenedores. Al realizar todas estas configuraciones, el proyecto quedó listo para desplegar en development cuando se hace un merge con la rama develop y para desplegar en producción cuando se hace merge con la rama master. En la figura 11 vemos un despliegue de la aplicación exitoso desde el dashboard de CircleCI.

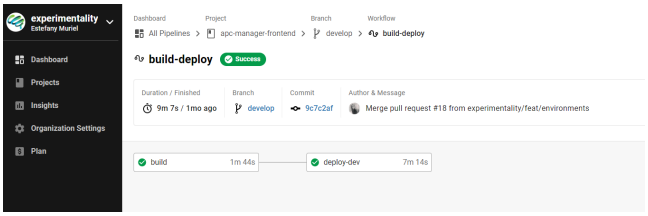

Figura 12: Despliegue en CircleCI

Durante el desarrollo del proyecto se tuvieron algunos impedimentos que no permitieron el desarrollo de unas funcionalidades que se habían considerado inicialmente como la creación de cliente en el ERP de la empresa y la facturación electrónica automática, esto debido a que durante el desarrollo algunas reglas de negocio cambiaron y se actualizó el ERP de la empresa con algunos cambios que no fue posible automatizar y que el equipo e-commerce está haciéndolos manual actualmente.

7. Conclusiones

El uso de React para el desarrollo de aplicaciones web es muy enriquecedor, ya que su filosofía del trabajo por componentes y de su renderizado por cada uno ayuda a mejorar el performance de las aplicaciones y a modularizar bastante la misma, facilitando así el mantenimiento a futuro.

Realizar aplicaciones con integración y despliegue continuo incrementa la productividad y satisfacción con el negocio, ya que se puede hacer en cada sprint

pequeños entregables que son reflejados inmediatamente en los ambientes de desarrollo, con la facilidad de probar las features desde un ambiente controlado.

Usar metodologías ágiles en el desarrollo de la aplicación permitió estar sincronizados en el equipo y además se pudo realizar ajustes sobre la marcha de funcionalidades para lograr el resultado esperado.

Como trabajo a futuro y si el ERP de la empresa lo permite, sería ideal conectar el inventario de la empresa con el sistema para actualizar los productos en las tiendas virtuales y así mismo, actualizar el ERP cuando se hace la compra de productos, además de la conexión para crear el cliente y la factura electrónica que no fue posible hacerla en esta práctica debido a que el ERP se actualizó durante el desarrollo de la misma y no se logró hacer la conexión con este para cumplir con las funcionalidades mencionadas anteriormente.

Muchos conceptos de la universidad fueron utilizados para el razonamiento y análisis que se hace frente al problema presentado por la compañía, sin embargo en el momento de iniciar la implementación del sistema hay muchas prácticas que se tienen en la industria que no son mostradas por la academia y que sería demasiado útil implementar en algunos cursos de desarrollo de software.

Referencias

[1] Sanchit Aggarwal. «Modern web-development using reactjs». En: *International Journal of Recent Research Aspects* 5.1 (2018), págs. 133-137.

[2] Atlassian. URL: `https://www.atlassian.com/es/software/jira/guides/use-cases/what-is-jira-used-for#jira-for-agile-teams`.

[3] Atlassian. URL: `https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow`.

[4] S. Balaji y M. Murugaiyan. «WATEERFALLVs V-MODEL Vs AGILE: A COMPARATIVE STUDY ON SDLC». En: 2012.

[5] circleCI. URL: `https://circleci.com/docs/2.0/about-circleci/`.

[6] docker. URL: `https://www.docker.com/resources/what-container`.

[7] Cory Gackenheimer. *Introduction to React*. Apress, 2015, pág. 1.

[8] Facebook Inc. URL: `https://es.reactjs.org/`.

[9] Mª Carmen Penadés José H. Canós Patricio Letelier. «Metodologías Ágiles en el Desarrollo de Software». En: (2003), págs. 1-8.

[10] Jeff Sutherland Ken Schwaber. *La Guía de Scrum*. Creative Commons, 2020, págs. 1-3.

[11] Amstrong Gary Kotler Philip. *Fundamentos de Marketing*. 11.ª ed. Pearson Education Limited, 2014, págs. 433-434.

[12] Nicolas Daviot Luc Filion. «Using Atlassian Tools for Efficient Requirements Management». En: (2017).

[13] M Foemmel Martin Fowler. «Continuous integration». En: *Thought-Works* 14 (2006), págs. 1-7.

[14] Torreão P Pereira P. «Entendiendo Scrum para gerenciar proyectos de forma ágil». En: 1 (2007), págs. 3-11.

[15] Sten Pittet. URL: https://www.atlassian.com/continuous-delivery/continuous-deployment.

[16] Nonaka I Takeuchi H. «The new product development game». En: *International Journal of Recent Research Aspects* 64.1 (1986), págs. 137-146.

[17] *The Scrum Framework Poster*. 2021. URL: https://www.scrum.org/resources/scrum-framework-poster.

[18] Shahrul Nizam Salahuddin Yi Jin Lim Abdullah Osman. «Factors Influencing Online Shopping Behavior: The Mediating Role of Purchase Intention». En: *Procedia Economics and Finance* 35.4 (2016), págs. 401-410.