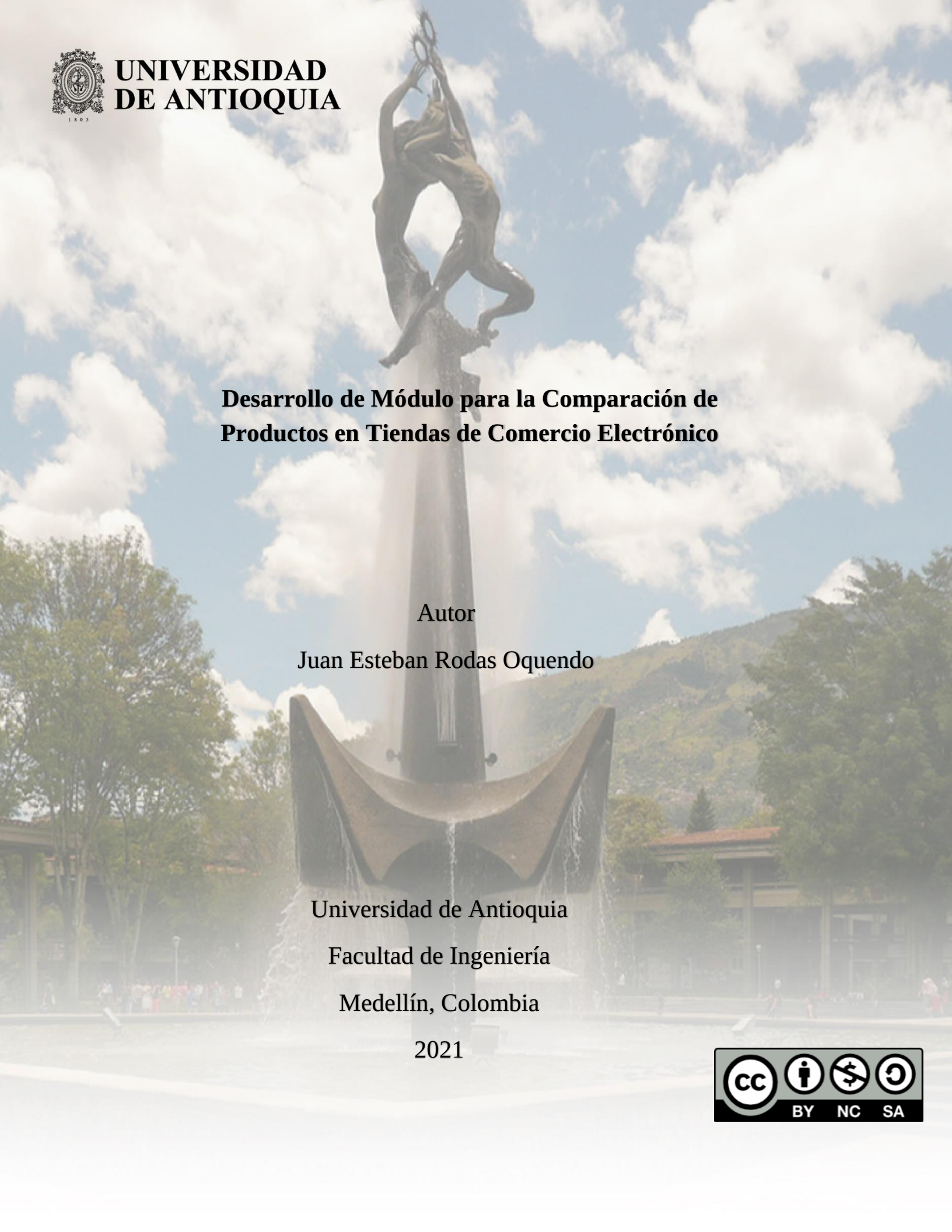

UNIVERSIDAD DE ANTIOQUIA

Desarrollo de Módulo para la Comparación de Productos en Tiendas de Comercio Electrónico

Autor

Juan Esteban Rodas Oquendo

Universidad de Antioquia

Facultad de Ingeniería

Medellín, Colombia

2021

DESARROLLO DE MÓDULO PARA LA COMPARACIÓN DE PRODUCTOS EN TIENDAS DE COMERCIO ELECTRÓNICO

Juan Esteban Rodas Oquendo

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

Asesores (a):
Javier Fernando Botia Valderrama - Doctor en Ingeniería Electrónica
Docente del Departamento de Ingeniería de Sistemas de la Universidad de Antioquia

Alejandro Granada Restrepo - Ingeniero de Sistemas
Líder de Desarrollo de SAP Commerce - Asesor Externo Talos Digital

Universidad de Antioquia
Facultad de Ingeniería
Medellín, Colombia
2021.

Tabla de Contenido

Tabla de Abreviaciones

Palabra	**Abreviación**
Business to client	B2C
Business to business	B2B
Progressive Web Application	PWA
SAP Commerce Cloud	SAP CC

1. Resumen

El *framework Spartacus* es un conjunto de librerías que se lanzó oficialmente en el 2019, esto con el fin de implementar de una manera más eficiente los avances de las tiendas de comercio electrónico que se desarrollarán a futuro en la empresa Talos Digital. Este proyecto se realizó debido a que *Spartacus* es una herramienta relativamente nueva, no trae varios módulos que son muy comunes en los desarrollos de la empresa, es por esto que es necesario crearlos desde cero, específicamente el módulo de comparación de productos, utilizando esta nueva tecnología, lo cual implica que se utilicen *frameworks* como *Angular* y librerías como NgRx para sacar el máximo provecho a *Spartacus*.

Uno de los principales resultados de la elaboración de este proyecto, fue que mediante el uso de la librería NgRx, se logró llevar a cabo de una forma más eficaz la comunicación de los componentes que se vieron involucrados en el desarrollo de la solución. Los datos fueron almacenados en un *store*, en este caso los productos, los cuales pudieron ser comparados, dado que fue posible enviar cada uno de ellos a los componentes y así realizar la función requerida. De igual manera, gracias a los múltiples beneficios de la librería, los estados de los componentes fueron guardados y modificados, los cuales fueron requeridos en el momento en el cual se navegaba a través de la tienda virtual y así verificar que las acciones realizadas quedan registradas sin importar las modificaciones en los otros componentes.

Es necesario resaltar que mediante este proyecto se evidenció la facilidad que provee *Spartacus* para personalizar y agregar nuevas funcionalidades a las tiendas de comercio virtual sin la necesidad de realizar grandes cambios en el código que viene predeterminado, esto permitió que la realización de una nueva interfaz de usuario no tuviera ningún inconveniente al momento de ser acoplada en el *framework*, lo que llevó a que todos los objetivos planteados se lograran.

Palabras clave

E-commerce, Spartacus, Angular, Frameworks, SAP Commerce

2. Introducción

Dada la pandemia actual, muchas empresas han tenido que buscar nuevas alternativas para llegar a sus usuarios, pues no solo los precios bajos y buenos productos es suficiente, es necesario ofrecer una experiencia de compra amigable para cada uno de los clientes (Thomsen, 2020). Es por esto que, desde marzo del año 2020, ha crecido de gran manera el desarrollo de tiendas de comercio electrónico. La empresa Talos Digital que se especializa en este tipo de desarrollos no estuvo exenta de esta realidad, debido al incremento de nuevos proyectos ya sea para nuevos desarrollos a empresas o para mejorar o robustecer algunas páginas de empresas que estaban en este tipo de comercios.

En vista de la gran demanda que se tiene actualmente y a los nuevos requerimientos que se exigen en los desarrollos de software actuales Talos Digital se vio en la necesidad de buscar nuevas tecnologías que incrementaran el valor del producto y que al mismo tiempo facilitara el desarrollo de estos. Es por ello que, para los nuevos productos hechos por la empresa, se empezó a utilizar *Spartacus* un *framework* basado en *Angular*, el cual trae como primicia el poder desacoplar el *storefront* (interfaz de usuario de una *e-commerce*) de la herramienta actual (*SAP Commerce Cloud*) donde actualmente se hace todo el desarrollo del *software*. *Spartacus* trae muchas funcionalidades de caja, lo que facilita el desarrollo y permite que para los diferentes proyectos se pueda hacer la personalización de las características que necesite el usuario, aun así hay algunas que son muy comunes y no vienen con el *framework*, es por esto que se ve en la necesidad de crear nuevos módulos los cuales suplen las diferentes funciones que no se han implementado y así poder utilizar cada una de estas en cualquier desarrollo de la empresa utilizando *Spartacus*.

Uno de estos módulos, el cual es muy pedido en los desarrollos de la empresa, es la comparación de productos, pues el 65% de los usuarios que van a adquirir cualquier producto buscan comparar el precio, sea con otro producto similar o con el mismo producto en otra tienda (The truth about online consumers, 2017). Es por esto que con este nuevo módulo podemos evitar que el usuario salga de la tienda virtual y pueda realizar sus compras con una mayor confianza. Dado lo anterior, vemos la

importancia de desarrollar e implementar dicho módulo para *Spartacus*, pues en la empresa los desarrollos de este tipo se hacen mediante *Spring Boot* utilizando el lenguaje de programación Java, el cual no permite realizar una migración a *Angular*, por esta razón se hará código completamente nuevo, haciendo uso de las grandes librerías como RxJs y NgRx, puesto que nos permitirán hacer una programación reactiva de la cual se obtendrán beneficios para las llamadas asíncronas al *backend* y para el manejo de estados, tanto de la aplicación como para la comunicación y envío de datos entre componentes.

Por otro lado, para el desarrollo del proyecto se dividió en tres fases, cada una de las cuales se enfocó en darle solución a un objetivo específico. Para la realización de cada fase se utilizó la metodología ágil *Scrum* la cual se caracteriza por realizar entregas parciales de lo que sería el producto final, y así se vería el avance del proyecto entregando código funcional fase por fase.

3. Objetivos

3.1. Objetivo General

Construir una interfaz de usuario para el *framework Spartacus*, la cual permite a los usuarios hacer una comparativa específica entre productos.

3.2. Objetivos Específicos

- Analizar el uso del *framework Spartacus* a partir de su implementación y utilidad para la empresa Talos Digital.
- Proponer una interfaz de usuario para comparar el comportamiento de las compras que realiza el cliente a partir del *framework Spartacus*.
- Evaluar el desempeño de la interfaz por medio de su integración con la GUI principal.

4. Marco Teórico

Para el desarrollo de una tienda e-commerce de gran magnitud es necesario tener herramientas disponibles para que los desarrolladores puedan hacer un despliegue mucho más fácil y rápido.

4.1. **SAP Commerce Cloud** es una plataforma *e-commerce*, anteriormente llamada SAP Hybris, basada en Java que permite crear multiplataformas de comercio electrónico, tanto para *B2C* como para *B2B* ("Choosing Which Storefront to Use for Your SAP Commerce Cloud Solution", 2021). Es necesario aclarar que SAP CC no es simplemente un *framework* para el desarrollo del backend, es un ambiente de desarrollo que integra todos los componentes necesarios para el negocio omnicanal de las e-commerce, la cual posee una arquitectura muy completa (ver figura 1).

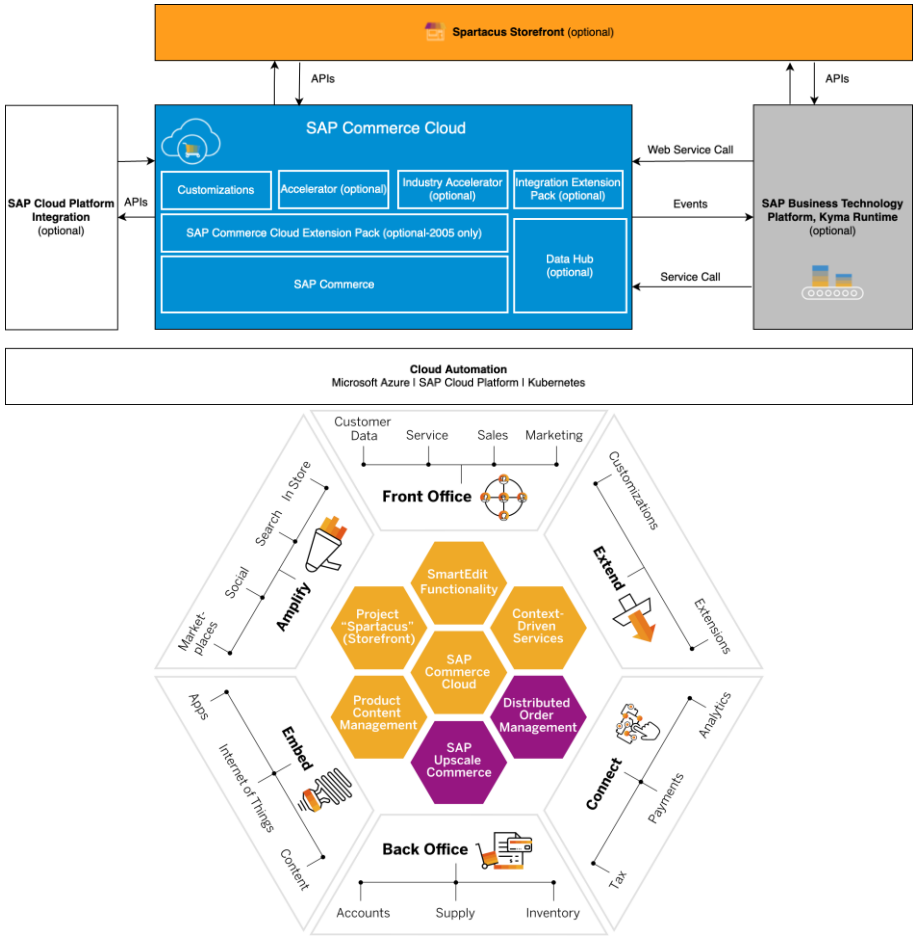

Figura 1- Arquitectura SAP Commerce Cloud [Tomado de ("SAP Commerce Cloud Architecture", 2021)]

4.2. **SAP Commerce Accelerator**. Para el desarrollo de las interfaces de usuario de estos comercios electrónicos, la primera herramienta que desarrolló SAP, la cual se integró muy bien con el *SAP Hybris* fue *SAP Commerce Accelerator* cuya plantilla de implementación web permite implementar su propia interfaz fácilmente y con muchas funcionalidades que vienen de caja ("SAP Accelerators", 2021). Este *Storefront* para SAP CC ha tenido muchas actualizaciones a través del tiempo incluyendo el módulo para plataformas tipo *B2B*. Dada la demanda del mercado en ese momento SAP se vio en la necesidad de crear Aceleradores Industriales (*Industry Accelerators*) los cuales se enfocaron en otros campos como los son los viajes, las telecomunicaciones y los medios, servicios financieros, entre otros, los cuales tienen ciclos de desarrollo diferentes a un *e-commerce* pero dependen de SAP CC para su implementación.

4.3. **Spartacus**. Dado que hasta el momento los recursos que se tenían disponibles estaban estrictamente entrelazados para cualquier desarrollo, SAP se vio en la necesidad de crear un *framework* para desacoplar el *storefront* del *SAP Commerce Core*, llamado *Spartacus*, el cual es desarrollado con el lenguaje de programación JavaScript que aprovecha las múltiples modernas librerías en este lenguaje y convertirlo en una completa aplicación web progresiva (PWA). Este *framework* crea una rápida y atractiva tienda electrónica, permitiendo al cliente tener una experiencia de usuario agradable ("Getting Started with SAP Commerce Cloud Project Spartacus", 2021) Algunas de las características más importantes de *Spartacus* y que lo diferencian de los demás son:

- Es configurable, muchas de las características del *storefront* son personalizables, pero la principal son los estilos. Se pueden utilizar variables globales y selectores muy específicos que permiten configurar ítems en *storefront.*

- Desacoplado, por lo tanto **escalable y flexible**. *Spartacus* se ejecuta en su propio servidor y realiza llamadas API REST a la plataforma o a cualquier otra librería externa (*ver figura 2*).

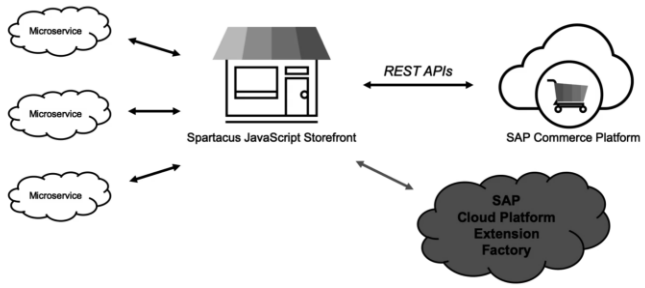

Figura 2 - Arquitectura Desacoplada [("Getting Started with Spartacus - openSAP Microlearning", 2021)]

- Los desarrollos en ***Spartacus son mucho más rápido que en otros frameworks***, lo que significa un tiempo de comercialización más rápido ("Getting Started with Spartacus - openSAP Microlearning", 2021). Debido a que Spartacus viene con características integradas, es un gran puntapié inicial para su *e-commerce* y no se estará construyendo desde cero.

- *Spartacus* hace uso de uno de los *frameworks* más utilizados para administrar el estado global de una aplicación, como lo es **NgRx**, cuya tarea es construir aplicaciones reactivas. **NgRx** trae grandes ventajas para el rendimiento, una mejor capacidad de prueba y facilidad de resolución de problemas ("Spartacus Documentation", 2021).

En algunos casos es posible que los *storefronts* mencionados anteriormente no se acomoden al equipo de trabajo, ya sea porque están familiarizados con un lenguaje de programación diferente, como Ruby, o porque las plantillas no se acomodan a la necesidad. Es por esto que para algunas implementaciones se necesita crear todo desde cero y ahí es cuando el equipo de desarrollo crea su propio *storefront* con cada una de las funcionalidades que necesita.

Para elegir una de las herramientas de desarrollo anteriormente mencionadas, es importante tener en cuenta tanto las capacidades del equipo técnico, como las necesidades del proyecto. Cabe aclarar que cada uno de los *frameworks* disponibles

difieren en algunos aspectos, lo primero es que están desarrollados en tecnologías completamente diferentes y por esto es imposible migrar de uno al otro, pues no hay alguna librería que lo haga o ayude a hacerlo y lo más relevante es que *Spartacus* es un *framework* basado en las últimas versiones de *Angular* lo que lo hace moderno mientras los otros vienen desde las primeras versiones de SAP CC.

5. Metodología

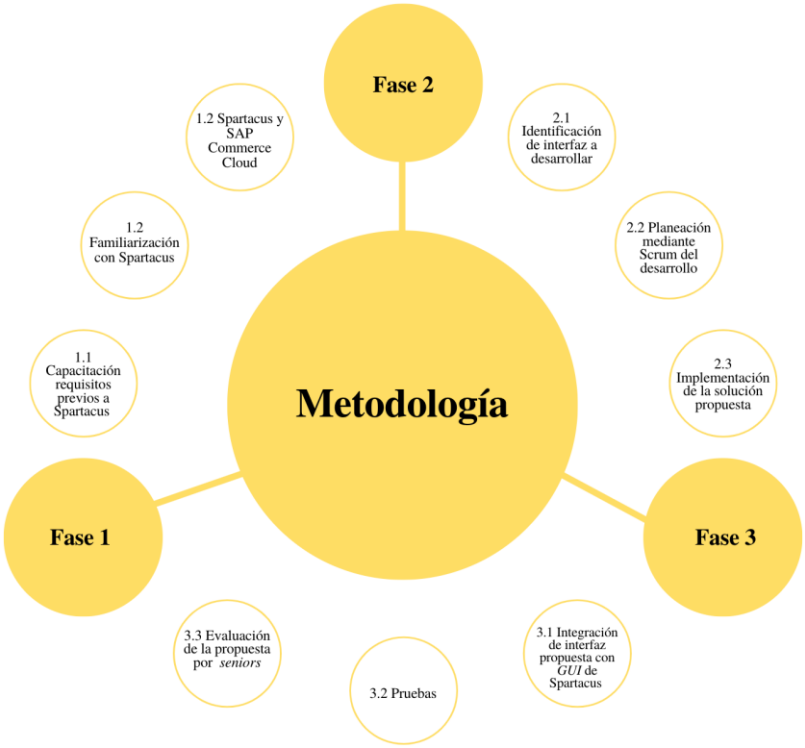

Figura 3 - Diagrama de Ideas [Propio]

Para llevar a cabo cada uno de los objetivos planteados, se dividió el desarrollo en las siguientes tres fases:

5.1. **Fase 1**: Esta primera fase consistió en la familiarización con *Spartacus*, en la cual la empresa puso a mi disposición una capacitación completa con todos los temas previos y necesarios para empezar con la herramienta, tales como CSS, HTML, JavaScript, TypeScript, entre otros. Luego de esto se hizo el entrenamiento en *Spartacus* y como es su conexión con SAP CC y de esta

manera podemos ver como esta fase está conectada con el primer objetivo específico del proyecto.

5.2. **Fase 2**: A partir de un conocimiento más profundo de las diferentes herramientas y lenguajes de programación a utilizar, se procedió a identificar la forma que se llevaría a cabo el desarrollo del módulo de comparación, el cual está estrictamente unido con el segundo objetivo específico. En esta fase del proyecto, se empezó a utilizar la metodología *Scrum* debido a que se inició con el desarrollo de cada uno de los *sprints*, con una duración de dos semanas para su implementación. Además, se realizaron pequeñas entregas funcionales de la interfaz de usuario que evidenciaban el cumplimiento de los objetivos de cada *sprint*, y de esta forma, se observan los posibles cambios que se llegaran a requerir en caso de mejoras del desarrollo.

5.3. **Fase 3**: Después de tener una interfaz de usuario definida, se procedió a hacer las acciones necesarias para que esta nueva interfaz se integrara de la mejor manera con el *storefront*. Mediante la modificación de los componentes involucrados, se realizó la integración debido a la necesidad de hacer algunos cambios en la interfaz de visualización del desarrollo. Por otra parte, se hicieron las pruebas para verificar que la implementación no presentara ningún inconveniente a la hora de un futuro desarrollo en la empresa con este nuevo módulo. Debido a que en la empresa se tiene previsto que los nuevos proyectos sean desarrollados utilizando *Spartacus*, se está capacitando a gran parte de los empleados en este *framework*. Por lo tanto, en este momento solo pocas personas tienen un amplio conocimiento en esta nueva herramienta. Es por esta razón que lo realizado en el proyecto se le presentó a un desarrollador *Spartacus Senior*, el cual examinó y testeo toda la implementación realizada.

6. Resultados y análisis

Durante el desarrollo de este proyecto, se efectuaron las actividades descritas a continuación:

- **Capacitación en las herramientas necesarias para el uso de *Spartacus***: A través de diferentes cursos que fueron puestos a disposición por la empresa, más precisamente en el desarrollo web *frontend*, hubo un entrenamiento en los diferentes lenguajes de programación y *frameworks* que son requeridos para un buen uso de *Spartacus* como lo son, HTML, CSS, JavaScript, TypeScript, *Angular*, entre otros. Además, se hizo un especial énfasis en las librerías más utilizadas en *Angular* para los desarrollos web, como son RxJs y NgRx, estas dos para construir aplicaciones reactivas, que son aquellas aplicaciones que se basan en el paradigma de programación declarativo el cual se ocupa de los flujos de datos, mayormente asíncronos. La librería NgRx es el popular patrón de gestión de estados llamado ***Redux*** hecho a la medida para *Angular* ("NgRx Docs", 2021). El patrón Redux nos ayuda a administrar el estado de la aplicación, al proporcionar un flujo de datos unidireccional en esta y así facilitar la comunicación entre los diferentes componentes de la aplicación ("Redux Fundamentals, Part 1: Redux Overview | Redux", 2021). Una vez familiarizado con todas las herramientas necesarias para cualquier implementación en *Spartacus*, fue preciso conocer sobre SAP CC para crear un proyecto en *Spartacus*, lo cual es necesario tener una copia de cualquiera de las últimas versiones de SAP CC. Con esto claro y teniendo en cuenta que ambas tecnologías son desarrolladas por SAP para tiendas electrónicas, podemos ver como cumplido el análisis del *framework* y su gran utilidad para Talos Digital.

- **Identificar interfaz de usuario a desarrollar**: Luego de entender e interiorizar la manera en que se pueden personalizar algunas de las funcionalidades que vienen “de caja” en *Spartacus*, se den las diferentes interfaces de usuario que se podían desarrollar, teniendo en cuenta que se estaba utilizando la tienda de electrónicos que viene con el *framework*, por lo tanto fue necesario que la interfaz seleccionada se acoplara de una manera muy natural con el *storefront*. Después de descartar dos opciones por no ser viables, se procedió al cambio de interfaz de usuario de *Spartacus* en diferentes componentes que hacian que el desarrollo se desviara a su objetivo.

Por consiguiente, se eligió crear un modal para que el usuario agregue los productos (Entre 2 y 4) que desea comparar. Para que el usuario pudiera acceder a este modal, fue necesario agregar un *checkbox* en la interfaz actual, el cual permitió agregar el producto cada vez que era seleccionado y eliminarlo de los productos a comparar cada vez que se deshacía la selección.

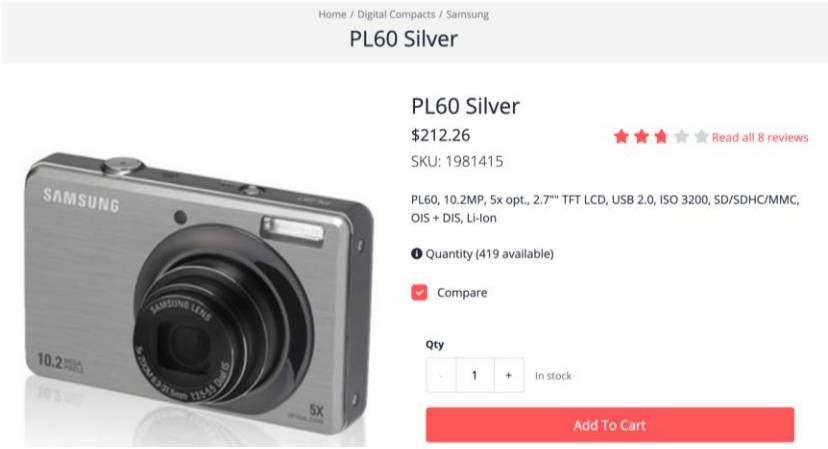

Figura 4 - Checkbox para agregar producto y abrir modal [Propio]

Una vez seleccionado el producto, se abría el modal con los elementos agregados y este contaba con dos botones, uno con el que podría seguir navegando en la página seleccionando más productos, y otro que lo redireccionaba a una nueva página en la que el usuario podía ver los productos que desea comprar juntos con sus especificaciones técnicas.

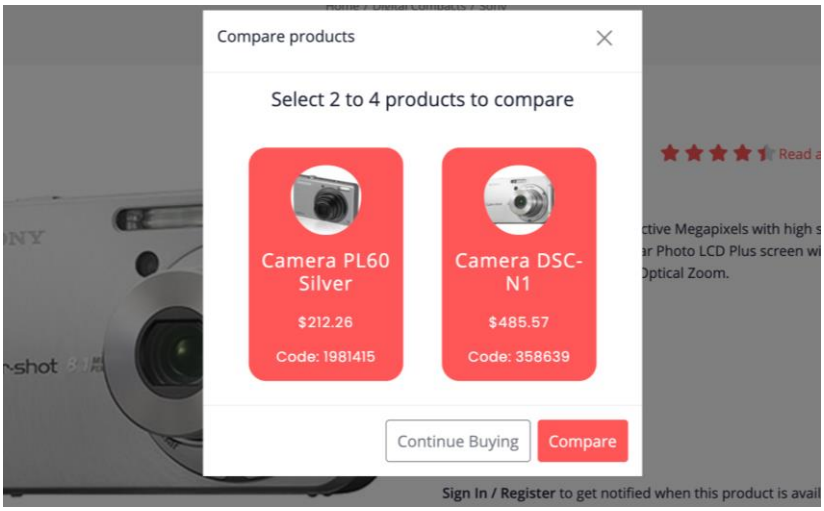

Figura 5 - Modal con productos para comparar [Propio]

- **Integración de interfaz propuesta con la GUI de *Spartacus***: Como se ha mencionado anteriormente este *framework* viene con una *e-commerce* predeterminada en la cual se pueden utilizar muchas de sus funcionalidades ya desarrolladas. Al crear la interfaz de usuario, se realizó de una manera mucho más cómoda en el momento de la implementación y se observaba como era su integración con la GUI principal. De igual modo se tuvieron que hacer varios cambios en algunos componentes, puesto que fue indispensable crear un nuevo componente el cual se extendia o heredaba de uno ya creado, ya que no era simplemente agregar el *checkbox* que abría el modal, si no que en este nuevo componente se necesitó recrear toda la información disponible, pero con la personalización requerida. Por otra parte, en la implementación de la nueva página de comparación fue necesario crearla en SAP CC, la cual nos posibilitó el acceso mediante el *routing* de *Angular*. Una vez creada la página, se personalizaron algunos atributos, como el nombre, el título, los *links* de las rutas, entre otros, además, se crearon los componentes que se requerían, los cuales permitieron enviar y mostrar los productos que se tenían en el modal. Por último, para que todo el manejo de estados se hiciera acorde a como lo hace *Spartacus*, en el *store* que maneja el *framework*, se crearon las diferentes acciones para el cambio de estados y administración de productos, todo esto con la ayuda de la librería NgRx implementada en *Spartacus*.

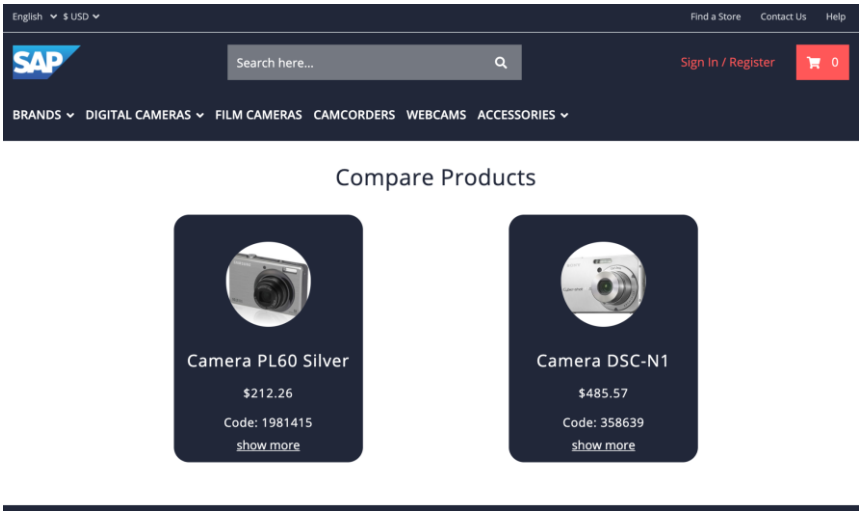

Figura 6 - Nueva página con productos para comparar [Propio]

6.1. Discusión

Con los resultados obtenidos, podemos deducir que los desarrollos hechos con *Spartacus* proveen muchas más herramientas al desarrollador, pues éstas permiten que las diferentes implementaciones realizadas tengan todas las ventajas que ofrece *Angular*, por ejemplo, como es el desarrollo de aplicaciones tipo PWA, las cuales generan una mejor experiencia de usuario, así como un menor costo en los desarrollos de la empresa. De igual manera, se puede evidenciar que tanto modificar, personalizar como agregar nuevas funcionalidades, no significó un gran reto como se estimó al inicio del proyecto, pues existen unos pasos muy simples a seguir que te indican cómo debe ser el proceso para llevar estos a cabo; además, estos pasos se deben complementar teniendo tanto un amplio conocimiento como un buen manejo de *Angular*. Esto brinda la posibilidad que al desarrollo hecho se le puedan hacer modificaciones a futuro, en caso de ser necesario o según lo considere el cliente.

Como se ha mencionado, *Spartacus* hace uso del patrón Redux mediante la librería NgRx, la cual está hecha para Angular, esta librería permitió el manejo de los estados del modal, así como el envío de datos a través de los componentes utilizados. Dado que *Spartacus* necesita SAP CC para su uso, mediante este último, se facilita la creación de nuevas páginas, componentes y *layouts* personalizables en *Angular*, además de permitir la modificación de las páginas a través de *SmartEdit*, esta herramienta nos da la posibilidad de cambiar todos los atributos de los componentes tipo CMS sin necesidad de codificar. Así mismo se pueden agregar nuevos catálogos de productos, entre otras facilidades que provee *SmartEdit*.

7. Conclusiones

- Para cualquier desarrollo con *Spartacus* es necesario tener un buen conocimiento de *Angular* pues toda la codificación se hace mediante este *framework.*
- Es muy importante conocer el uso de las librerías para *Angular*, RxJs y NgRx, ambas encargadas en la implementación del paradigma de programación reactiva.
- Como *Spartacus* implementa el patrón redux mediante la librería NgRx, es necesario para cualquier desarrollo que se siga este mismo patrón para no desentonar en el manejo de estados de la aplicación.
- Aunque este proyecto era principalmente desarrollo *frontend* se necesita tener bases o aprender sobre el *backend* en Java pues se requirió manejo de SAP CC para algunas tareas.
- Para el desarrollo de la interfaz propuesta es necesario tener en cuenta la interfaz de usuario del *storefront* pues estas dos se deben complementar y crear una buena sensación al usuario.
- La personalización de una *e-commerce* usando *Spartacus* se facilita, pues para hacer modificaciones en las funcionalidades que se requieran se puede heredar de las ya hechas y hacer los cambios. Teniendo en cuenta que en *Angular* se utiliza el lenguaje de programación TypeScript se aprovechó que este provee un alto grado de tipificado el cual evita que se pierda tiempo en la búsqueda de errores. Finalmente, cualquier código que se haga, sin importar si es en la etapa de desarrollo o en las pruebas es necesario que cumpla con algún estándar de buenas prácticas de programación.

Por último, es necesario resaltar toda la experiencia adquirida en el mundo empresarial, pues en este campo se aprende demasiado rápido en la parte técnica ya que se está constantemente en contacto con ello, así mismo en la parte personal se terminan de desarrollar algunas habilidades que desde la universidad se empiezan a inculcar, como lo es el trabajo en equipo y las llamadas habilidades blandas.

8. Referencias Bibliográficas

- KPMG International. (2017). The truth about online consumers [Ebook] (pp. 12-13). Retrieved 23 May 2021, from https://assets.kpmg/content/dam/kpmg/xx/pdf/2017/01/the-truth-about-online-consumers.pdf.
- Thomsen, R. (2020). 19 datos estadísticos sobre e-commerce que todos los vendedores deben conocer [2020]. Sleeknote. Retrieved 23 May 2021, from https://sleeknote.com/es/blog/datos-estadisticos-e-commerce.
- Choosing Which Storefront to Use for Your SAP Commerce Cloud Solution. SAP. (2021). Retrieved 15 April 2021, from https://www.sap.com/cxworks/article/435949087/choosing_which_storefront_to_use_for_your_sap_commerce_cloud_solution.
- SAP Commerce Cloud Architecture. SAP.com. (2021). Retrieved 16 April 2021, from https://www.sap.com/cxworks/article/435954690/sap_commerce_cloud_architecture.
- SAP Accelerators. Help.sap.com. (2021). Retrieved 15 April 2021, from https://help.sap.com/viewer/4c33bf189ab9409e84e589295c36d96e/2011/en-US/8adca7a186691014bd31f1d2d96624f5.html.
- Getting Started with SAP Commerce Cloud Project Spartacus. SAP. (2021). Retrieved 15 April 2021, from https://www.sap.com/cxworks/article/455575580/getting_started_with_sap_commerce_cloud_project_spartacus.
- Getting Started with Spartacus - openSAP Microlearning. Microlearning.opensap.com. (2021). Retrieved 15 April 2021, from https://microlearning.opensap.com/playlist/dedicated/178318381/1_sokuy66v/1_6dln57h9.
- Spartacus Documentation. Spartacus Documentation. (2021). Retrieved 23 May 2021, from https://sap.github.io/spartacus-docs/.
- NgRx Docs. Ngrx.io. (2021). Retrieved 24 May 2021, from https://ngrx.io/docs.

- Redux Fundamentals, Part 1: Redux Overview | Redux. Redux.js.org. (2021). Retrieved 24 May 2021, from https://redux.js.org/tutorials/fundamentals/part-1-overview.