PLATAFORMA WEB INTERMEDIARIA PARA LA COMERCIALIZACIÓN DE PRODUCTOS AGRÍCOLAS EN LA ZONA PERIFÉRICA DE BOGOTÁ.



UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

JULY TATIANA JIMENEZ ROZO
BRAIN ALEJANDRO TRIBALDOS VALERO

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS
FACULTAD TECNOLÓGICA
TECNOLOGÍA EN SISTEMATIZACIÓN DE DATOS
BOGOTÁ D.C., COLOMBIA
2019

PLATAFORMA WEB INTERMEDIARIA PARA LA COMERCIALIZACIÓN DE PRODUCTOS AGRÍCOLAS EN LA ZONA PERIFÉRICA DE BOGOTÁ.



UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

JULY TATIANA JIMENEZ ROZO 20151578036
BRAIN ALEJANDRO TRIBALDOS VALERO 20151578093

Investigación e Innovación
Trabajo de grado presentado como requisito para optar al título de
TECNÓLOGO EN SISTEMATIZACIÓN DE DATOS

Tutora: SONIA ALEXANDRA PINZON NUÑEZ
Ingeniera de Sistemas
spinzon@udistril.edu.co

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS
FACULTAD TECNOLÓGICA
TECNOLOGÍA EN SISTEMATIZACIÓN DE DATOS
BOGOTÁ D.C., COLOMBIA
2019

Nota de Aceptación

_____
_____
_____
_____
_____
_____

_____

Tutora del Proyecto
Ing. Sonia Alexandra Pinzón Núñez

_____

Jurado
Ing. Mariluz Romero García

Bogotá D.C, agosto de 2019

CONTENIDO

RESUMEN	11
ABSTRACT	12
INTRODUCCION	13
1. FASE 1: PLANEACION, DEFINCIÓN, ORGANIZACIÓN	14
1.1 TITULO	14
1.2 TEMA	14
1.3 PLANTEAMINETO DEL PROBLEMA	14
1.3.1 Descripción	14
1.3.2 Identificación	15
1.4 ALCANCES Y DELIMITACIONES	15
1.5 OBJETIVOS	16
1.5.1 Objetivo General	16
1.5.2 Objetivos Específicos	16
1.6 JUSTIFICACIÓN	17
1.7 MARCO REFERENCIAL	17
1.7.1 Marco Teórico	17
1.7.1.1 Estado del arte	17
7.1.1.2 Comercialización agrícola	18
7.1.1.3 Herramientas de desarrollo	18
1.7.2 Metodología de Desarrollo	20
1.7.2.1 Descripción de Scrum	20
1.7.2.2 Roles	20
1.7.2.3 Desarrollo de las fases	21
1.7.2.4 Herramientas	22
1.8 RESULTADOS ESPERADOS	23
1.9 IMPACTO	23
1.9.1 Social	23
1.9.2 Ambiental	23
1.9.3 Tecnológico	23
1.9.4 Innovación	24
1.9.5 Económico	24
1.10 CRONOGRAMA	25
1.11 PRESUPUESTO Y FUENTES DE FINANCIACIÓN	28
2. FASE 2: ANALISIS DEL PROYECTO	28
2.1 IDENTIFICACIÓN DE LOS ROLES	28
2.2 LISTA DE ACTIVIDADES POR ROL	29
2.3 PRODUCTO BACKLOG	31
2.4 SPRINT O ITERACIONES	33
3. FASE 3: DISEÑO DE LOS SPRINT	34
3.1 SPRINT 1 LOGIN	34
3.1.1 Definición del Sprint 1	34
3.1.2 Tareas del Sprint 1	34
3.1.3 Diseño del Sprint 1	35

3.1.4 Proceso de desarrollo de software del Sprint 1 35
3.1.5 Diagrama de contexto del Sprint 1 36
3.1.6 Prueba del Sprint 1 36
3.2 SPRINT 2 CARRITO DE COMPRAS 37
3.2.1 Definición del Sprint 2 37
3.2.2 Tareas del Sprint 2 37
3.2.3 Diseño del Sprint 2 38
3.2.4 Proceso de desarrollo de software del Sprint 2 40
3.2.5 Diagrama de contexto del Sprint 2 41
3.2.6 Prueba del Sprint 2 41
3.3 SPRINT 3 GESTION DE PRODUCTOS 42
3.3.1 Definición del Sprint 3 42
3.3.2 Tareas del Sprint 3 43
3.3.3 Diseño del Sprint 3 43
3.3.4 Proceso de desarrollo de software del Sprint 3 47
3.3.5 Diagrama de contexto del Sprint 3 54
3.3.6 Prueba del Sprint 3 55
3.4 SPRINT 4 GESTION DE TRANSPORTE 55
3.4.1 Definición del Sprint 4 55
3.4.2 Tareas del Sprint 4 56
3.4.3 Diseño del Sprint 4 57
3.4.4 Proceso de desarrollo de software del Sprint 4 60
3.4.5 Diagrama de contexto del Sprint 4 67
3.4.6 Prueba del Sprint 4 68
3.5 SPRINT 5 GESTION DE PRODUCTOS POSTULADOS 68
3.5.1 Definición del Sprint 5 68
3.5.2 Tareas del Sprint 5 69
3.5.3 Diseño del Sprint 5 70
3.5.4 Proceso de desarrollo de software del Sprint 5 71
3.5.5 Diagrama de contexto del Sprint 5 75
3.5.6 Prueba del Sprint 5 76
3.6 SPRINT 6 GESTION DE TRANSPORTE POSTULADOS 76
3.6.1 Definición del Sprint 6 77
3.6.2 Tareas del Sprint 6 77
3.6.3 Diseño del Sprint 6 78
3.6.4 Proceso de desarrollo de software del Sprint 6 79
3.6.5 Diagrama de contexto del Sprint 6 83
3.6.6 Prueba del Sprint 6 83
3.7 SPRINT 7 GESTION DE INQUIETUDES 84
3.7.1 Definición del Sprint 7 84
3.7.2 Tareas del Sprint 7 84
3.7.3 Diseño del Sprint 7 85
3.7.4 Proceso de desarrollo de software del Sprint 7 86
3.7.5 Diagrama de contexto del Sprint 7 87
3.7.6 Prueba del Sprint 7 87

3.8 SPRINT 8 INFORME 87
3.8.1 Definición del Sprint 8 88
3.8.2 Tareas del Sprint 8 88
3.8.3 Diseño del Sprint 8 89
3.8.4 Proceso de desarrollo de software del Sprint 8 91
3.8.5 Diagrama de contexto del Sprint 8 93
3.8.6 Prueba del Sprint 8 93
4. FASE 4: IMPLEMENTACION 94
4.1 DIAGRAMA DE PAQUETES 94
4.2 PRUEBAS EN DIFERENTES RESOLUCIONES Y NAVEGADORES 94
5. CONCLUSIONES 98
6. RECOMENDACIONES 99
7. BIBLIOGRAFIA 100

LISTA DE TABLAS

Tabla 1: Alcances y delimitaciones. 16
Tabla 2: Factibilidad económica recursos humanos. 28
Tabla 3: Factibilidad económica recursos técnicos. 28
Tabla 4: Factibilidad económica costo total. 28
Tabla 5: Identificación de roles. 29
Tabla 6: Lista de actividades por rol. 30
Tabla 7: Producto Backlog. 33
Tabla 8: Sprint o iteración. 34
Tabla 9: Definición del Sprint 1 34
Tabla 10: Tareas del Sprint 1. 35
Tabla 11: Prueba del Sprint 1. 37
Tabla 12: Definición del Sprint 2. 37
Tabla 13: Tareas del Sprint 2. 38
Tabla 14: Prueba del Sprint 2. 42
Tabla 15: Definición del Sprint 3. 42
Tabla 16: Tareas del Sprint 3. 43
Tabla 17: Prueba del Sprint 3. 55
Tabla 18: Definición del Sprint 4. 56
Tabla 19: Tareas del Sprint 4. 56
Tabla 20: Prueba del Sprint 4. 68
Tabla 21: Descripción del Sprint 5. 69
Tabla 22: Tareas del Sprint 5. 70
Tabla 23: Prueba del Sprint 5. 76
Tabla 24: Definición del Sprint 6. 77
Tabla 25: Tareas del Sprint 6. 78
Tabla 26: Prueba del Sprint 6. 84
Tabla 27: Definición del Sprint 7. 84
Tabla 28: Tareas del Sprint 7. 85
Tabla 29: Prueba del Sprint 7. 87
Tabla 30: Definición del Sprint 8. 88
Tabla 31: Tareas del Sprint 8. 88
Tabla 32: Prueba del Sprint 8. 93
Tabla 33: Pruebas en diferentes resoluciones y navegadores. 97

LISTA DE ILUSTRACIONES

Ilustración 1: Cronograma. 25
Ilustración 2: Cronograma 2. 26
Ilustración 3: Cronograma. 27
Ilustración 4: Diseño del Sprint 1. 35
Ilustración 5: Desarrollo del Sprint 1. 35
Ilustración 6: Desarrollo del Sprint 1. 36
Ilustración 7: Diagrama del Sprint 1. 36
Ilustración 8: Diseño del Sprint 2. 38
Ilustración 9: Diseño del Sprint 2. 39
Ilustración 10: Diseño del Sprint 2. 39
Ilustración 11: Diseño del Sprint 2. 39
Ilustración 12: Desarrollo del Sprint 2. 40
Ilustración 13: Desarrollo del Sprint 2. 40
Ilustración 14: Diagrama del Sprint 2. 41
Ilustración 15: Diseño del Sprint 3. 44
Ilustración 16: Diseño del Sprint 3. 44
Ilustración 17: Diseño del Sprint 3. 45
Ilustración 18: Diseño del Sprint 3. 45
Ilustración 19: Diseño del Sprint 3. 46
Ilustración 20: Diseño del Sprint 3. 46
Ilustración 21: Diseño del Sprint 3. 47
Ilustración 22: Desarrollo del Sprint 3. 47
Ilustración 23: Desarrollo del Sprint 3. 48
Ilustración 24: Desarrollo del Sprint 3. 48
Ilustración 25: Desarrollo del Sprint 3. 49
Ilustración 26: Desarrollo del Sprint 3. 49
Ilustración 27: Desarrollo del Sprint 3. 50
Ilustración 28: Desarrollo del Sprint 3. 50
Ilustración 29: Desarrollo del Sprint 3. 51
Ilustración 30: Desarrollo del Sprint 3. 51
Ilustración 31: Desarrollo del Sprint 3. 52
Ilustración 32: Desarrollo del Sprint 3. 52
Ilustración 33: Desarrollo del Sprint 3. 53
Ilustración 34: Desarrollo del Sprint 3. 53
Ilustración 35: Desarrollo del Sprint 3. 54
Ilustración 36: Desarrollo del Sprint 3. 54
Ilustración 37: Diagrama del Sprint 3. 54
Ilustración 38: Diseño del Sprint 4. 57
Ilustración 39: Diseño del Sprint 4. 57
Ilustración 40: Diseño del Sprint 4. 58
Ilustración 41: Diseño del Sprint 4. 58
Ilustración 42: Diseño del Sprint 4. 59
Ilustración 43: Diseño del Sprint 4. 59

Ilustración 44: Diseño del Sprint 4. 60
Ilustración 45: Desarrollo del Sprint 4. 60
Ilustración 46: Desarrollo del Sprint 4. 61
Ilustración 47: Desarrollo del Sprint 4. 61
Ilustración 48: Desarrollo del Sprint 4. 62
Ilustración 49: Desarrollo del Sprint 4. 62
Ilustración 50: Desarrollo del Sprint 4. 63
Ilustración 51: Desarrollo del Sprint 4. 63
Ilustración 52: Desarrollo del Sprint 4. 64
Ilustración 53: Desarrollo del Sprint 4. 64
Ilustración 54: Desarrollo del Sprint 4. 65
Ilustración 55: Desarrollo del Sprint 4. 65
Ilustración 56: Desarrollo del Sprint 4. 66
Ilustración 57: Desarrollo del Sprint 4. 67
Ilustración 58: Desarrollo del Sprint 4. 67
Ilustración 59: Diagrama del Sprint 4. 67
Ilustración 60: Diseño del Sprint 5. 70
Ilustración 61: Diseño del Sprint 5. 70
Ilustración 62: Diseño del Sprint 5. 71
Ilustración 63: Desarrollo del Sprint 5. 71
Ilustración 64: Desarrollo del Sprint 5. 72
Ilustración 65: Desarrollo del Sprint 5. 73
Ilustración 66: Desarrollo del Sprint 5. 74
Ilustración 67: Desarrollo del Sprint 5. 75
Ilustración 68: Desarrollo del Sprint 5. 75
Ilustración 69: Diagrama del Sprint 5. 75
Ilustración 70: Diseño del Sprint 6. 78
Ilustración 71: Diseño del Sprint 6. 78
Ilustración 72: Diseño del Sprint 6. 79
Ilustración 73: Desarrollo del Sprint 6. 79
Ilustración 74: Desarrollo del Sprint 6. 80
Ilustración 75: Desarrollo del Sprint 6. 81
Ilustración 76: Desarrollo del Sprint 6. 81
Ilustración 77: Desarrollo del Sprint 6. 82
Ilustración 78: Diagrama del Sprint 6. 83
Ilustración 79: Diseño del Sprint 7. 85
Ilustración 80: Desarrollo del Sprint 7. 86
Ilustración 81: Desarrollo del Sprint 7. 86
Ilustración 82: Diagrama del Sprint 7. 87
Ilustración 83: Diseño del Sprint 8. 89
Ilustración 84: Diseño del Sprint 8. 89
Ilustración 85: Diseño del Sprint 8. 90
Ilustración 86: Diseño del Sprint 8. 90
Ilustración 87: Desarrollo del Sprint 8. 91
Ilustración 88: Desarrollo del Sprint 8. 91

Ilustración 89: Desarrollo del Sprint 8. 92
Ilustración 90: Desarrollo del Sprint 8. 92
Ilustración 91: Diagrama del Sprint 8. 93
Ilustración 92: Diagrama de paquetes. 94

RESUMEN

En este proyecto se plantea una alternativa para la comercialización de productos de los sectores periféricos de Bogotá, en la cual se abordaron diferentes aspectos de la comercialización como lo son el transporte, los aportes de los usuarios, entre otros. La planificación de la página identifico diferentes aspectos que se pudieron abordar desde la página para que este medio ser más eficiente al momento de realizar los objetivos propuestos con mayor facilidad y eficiencia donde los usuarios ven que el proceso de comercialización se puede ver simplificado, donde el proyecto fortalece el proceso buscar fortalecer la comercialización de los productos de aquellas personas que buscan una incorporación de sus productos y servicios al área de las telecomunicaciones.

ABSTRACT

This project proposes an alternative for the marketing of products of the peripheral sectors of Bogotá, in which different aspects of the commercialization were addressed such as the transport, the contributions of the users, among others. The planning of the page identified different aspects that could be addressed from the page so that this medium is more efficient at the moment of achieving the proposed objectives more easily and efficiently where users see that the marketing process can be simplified, where the project strengthens the process of seeking to strengthen the marketing of the products of those who seek to incorporate their products and services into the telecommunications area.

INTRODUCCION

En la actualidad de nuestro diario vivir la tecnología está tomando un papel muy importante ya que facilita nuestras actividades y mejora la manera en como estas se realizan. Por ello la importancia de estar al día con las herramientas tecnológicas que se están desarrollando en todo el mundo. Ahora bien, si tenemos en cuenta que la agricultura es parte fundamental para nuestro desarrollo y más en nuestro país que es una rama que no está siendo explotada a su máximo nivel, con ese fin se está empezando a relacionar la tecnología con la agricultura.

Es por eso que se desarrolló este proyecto con el objetivo de facilitar la comercialización agrícola que se realiza especialmente en los sectores periféricos de Bogotá, debido que allí no cuentan con los recursos tecnológicos que les permitan ofrecer sus productos directamente a sus clientes o consumidores y debido a esta situación recurren a negociar con terceros para así poder vender sus productos.

Para resolver esta situación se desarrolló una plataforma web que ofrece los servicios de venta, compra y transporte de productos agrícolas para poder comercializar sin terceras personas involucradas, lo cual generara una mejora tanto económica, de calidad y eficiencia. Por lo cual en la solución tecnológica se implementó las herramientas actuales de desarrollo de software como lo son Angular, y FireBase para el almacenamiento de los datos.

1. FASE 1: PLANEACION, DEFINCIÓN, ORGANIZACIÓN

1.1 TITULO

PLATAFORMA WEB INTERMEDIARIA PARA LA COMERCIALIZACIÓN DE PRODUCTOS AGRÍCOLAS EN LA ZONA PERIFÉRICA DE BOGOTÁ.

1.2 TEMA

Plataforma web intermediaria que gestione los servicios de venta, compra y transporte de productos agrícolas de las zonas periféricas de Bogotá.

1.3 PLANTEAMINETO DEL PROBLEMA

1.3.1 Descripción

La comercialización agrícola de los sectores periféricos de Bogotá muestra un panorama negativo en relación con la adaptación a las nuevas tecnologías y la forma en la cual comercializan sus productos[1]; llegando a estar en situación de desventaja al intentar competir con diversas empresas a nivel nacional [2].

En la parte de comercialización se ven problemas dada la expansión urbana por la cual se ha visto afectado el sector agrícola[3], dada la mala ubicación del sector si hablamos en el sentido de comercializar sus productos, dados antecedentes en los cuales han fracasado los sectores que buscaban la comercialización de sus productos como lo fue llanoabastos que no funcionó por este factor en específico. Se necesita que la comercialización de sus productos se mejore para que los recursos correspondientes a este sector no les sean arrebatados a los campesinos por el hecho de que sus productos no pueden ser comercializados fácilmente y requieren de intermediarios lo cual no garantiza el sustento suficiente para mantener sus tierras y así mismos.

---

[1] Cámara de comercio de Bogotá. (2016). La cadena de comercialización de alimentos en Colombia. Julio, 2016, de Cámara de comercio de Bogotá Sitio web: https://www.ccb.org.co/Sala-de-prensa/Noticias-sector-Agricola-y-Agroindustrial/Noticias-2016/La-cadena-de-comercializacion-de-alimentos-en-Colombia

[2] Perletti, J; Botero, J; Oviedo, S; Forero, F; Higuera, S; Correa, M; García, J. (2017). Política comercial agrícola: nivel, costos y efectos de la protección en Colombia. Septiembre 21, 2018, Sitio web: https://www.repository.fedesarrollo.org.co/bitstream/handle/11445/3443/Repor_Agosto_2017_Perfetti_et_al.pdf?sequence=3

[3] Mendoza, G. (2002). Diagnóstico del mercadeo agrícola y agroindustrial en Colombia. Bogotá: U. Jorge Tadeo Lozano

Aunque existen muchos proyectos que se quieren realizar en el sector como lo son el parque Minero Industrial[4]. Se requiere que el sector sea reconocido a nivel nacional por sus productos agrícolas para que estos proyectos se lleven a cabo en otro lugar para que la zona permanezca como sector de agricultura, ganadería, entre otros[5].

Por otra parte, Bogotá posee zonas destinadas al trabajo agrícola y ciudad bolívar [6] es una de las localidades que contiene comunidades rurales cuyo sustento se basa en este tipo de trabajo, la alcaldía de Bogotá[7], junto con la gobernación han creado programas que permitan comercializar a los productos a los habitantes de estas zonas, creando espacios para vender directamente sin intermediarios, pero estos programas aún tienen muchas falencias respecto al proceso de contacto y transporte de dichos productos[8].

1.3.2 Identificación

¿Es posible generar una solución tecnológica que permita apoyar los procesos de comercialización de productos agrícolas en sectores periféricos de la ciudad?

Surge la necesidad de dar una solución tecnológica a los sectores periféricos de Bogotá debido a que su ingreso principal es el sector agrícola y este se ve afectado por la falta de métodos apropiados para ofrecer sus productos, ya que los servicios de plataformas virtuales actuales que tiene el país no son brindados para estos sectores si no para empresas a nivel nacional.

1.4 ALCANCES Y DELIMITACIONES

En la tabla 1 se muestra la descripción de los alcances y delimitaciones según los diferentes aspectos como los son el temático, temporal y geográfico.

---

[4] Secretaria distrital de ambiente. (2018). Parque-minero-industrial. Septiembre 26, 2018, de Secretaria distrital de ambiente. Sitio web: http://ambientebogota.gov.co/nb/parque-minero-industrial

[5] Alcaldía Local de Ciudad Bolívar. (2013-2016). PLAN AMBIENTAL LOCAL LOCALIDAD 19 CIUDAD BOLÍVAR. 2017, de Alcaldía Local de Ciudad Bolívar Sitio web: http://www.ambientebogota.gov.co/documents/10157/2883179/PAL+CBol%C3%ADvar+2013-2016.pdf

[6] Secretaria Distrital de Ambiente. (2018). Sistema Agropecuario Distrital (SISADI). Septiembre 30, 2018, de Secretaria Distrital de Ambiente Sitio web: http://ambientebogota.gov.co/bg/sistema-agropecuario-distrital-sisadi

[7] Secretaria de ciencia y tecnología. (2017). Proyectos de la Secretaría de Ciencia, Tecnología e Innovación. Septiembre 30, 2018, de Secretaria de ciencia y tecnología Sitio web: http://www.cundinamarca.gov.co/Home/SecretariasEntidades.gc/SecdeCienyTec/SecdeCienyTecDespliegue/aspropgramasyproyectos_contenidos/proyectos

[8] CIUDAD & TERRITORIO. (2016). Pasquilla, una vereda de Bogotá, presionada por crecimiento urbano. Mayo 23, 2016, de Agencia de Noticias UN Sitio web: http://agenciadenoticias.unal.edu.co/detalle/article/pasquilla-una-vereda-de-bogota-presionada-por-crecimiento-urbano.html

	Alcances	Delimitaciones
Temático	Esta plataforma web va a contener los temas de comercialización de productos agrícolas que permitirá comprar y ofrecer sus productos.	La plataforma se enfocara en la comercialización de los productos agrícolas del sector y en los servicios al usuario tales como reportes o consultas de los productos y la disponibilidad del servicio de transporte.
Temporal	El proyecto se desea realizar en un tiempo de 6 meses (septiembre-febrero).	Se espera tener la plataforma funcionando en un tiempo determinado de 6 meses (septiembre-febrero).
Geográfico	Este proyecto se realizará en los sectores periféricos de Bogotá.	Este proyecto se realizará en los sectores periféricos de Bogotá, específicamente en la localidad 19 ciudad bolívar de donde se obtendrán los datos de los productos agrícolas que allí cultivan.

Tabla 1: Alcances y delimitaciones.

1.5 OBJETIVOS

1.5.1 Objetivo General

Desarrollar una plataforma web intermediaria que permita la comercializar los productos agrícolas que se ofrecen en sectores periféricos de Bogotá.

1.5.2 Objetivos Específicos

- Realizar un diagnóstico de los procesos de comercialización en la periferia para identificar los datos que debe procesar el sistema.
- Desarrollar e implementar un módulo que permita generar un catálogo de productos.
- Fortalecer la gestión de la información de los productos mediante la apropiación de las nuevas tecnologías de información y comunicación.

- Articular un sistema horizontal de información sobre los productos que garantice la organización, conservación y difusión para la futura comercialización de estos.
- Implementar módulos de gestión de usuarios, comercialización que de reportes o consultas para quienes desean ofrecer sus productos a quienes deseen comprar y quienes deseen ofrecer el servicio de transporte.

1.6 JUSTIFICACIÓN

Este proyecto se realizará con el fin de diseñar una plataforma web que facilite la comercialización de productos agrícolas de los sectores periféricos de Bogotá, ya que estos sectores no cuentan con los recursos tecnológicos como lo son plataformas web.

1.7 MARCO REFERENCIAL

1.7.1 Marco Teórico

1.7.1.1 Estado del arte

Nuestra temática se relaciona con algunos proyectos que ya se han implementado en el campo en el cual estamos desarrollando el proyecto, como los siguientes:

- Agronet
  Es una red de información y de comunicación del sector agropecuario colombiano, que brinda la información para el apoyo de decisiones sobre el sector y brinda los servicios de (producción y agronegocios, capacitación, estadísticas, desarrollo rural y agroclima), es liderada por el ministerio de agricultura y minagricultura entre otras entidades gubernamentales[9].

- Comproagro
  Es una plataforma web desarrollada para la comercialización de productos agrícolas con el objetivo de dinamizar la relación entre los consumidores urbanos, nuevas tecnologías y trabajo campesino[10].

[9] Agronet. (2016). Agronet. Septiembre 21, 2018, de MINAGRICULTURA Sitio web: http://www.agronet.gov.co/Paginas/default.aspx

[10] Comproagro. (2018). Comproagro. Septiembre 30, 2018, de Comproagro Sitio web: https://www.comproagro.com/

- Agrosavia
  Es una plataforma web de la corporación de investigación agropecuaria de Colombia antes llamada corpoica, cuyo objetivo es desarrollar y ejecutar actividades de ejecución, que permitan la implementación de nuevas tecnologías en este sector, esta plataforma permite la compra de productos agrícolas entre otros servicios, es liderada por minagricultura[11].

7.1.1.2 Comercialización agrícola

La comercialización agrícola cubre los servicios que se ocupan de hacer llegar el producto agrícola de la granja al consumidor. De esta manera existen numerosas actividades interconectadas implicadas en este proceso durante el cual los cultivadores pueden vender sus cultivos para poder conservar dinero y volver a sembrar y cultivar.

La comercialización agrícola es realizada más bien por el sector privado que por los gobiernos y todos los pasos de la cadena deben mostrar un beneficio para los participantes. El apoyo a los países en desarrollo para el progreso agrícola es llevado a cabo por organizaciones como la FAO1 y varias organizaciones donantes. Las actividades incluyen el desarrollo de la información de mercado, extensión en comercialización, capacitación agrícola y desarrollo de infraestructura. Las tendencias recientes muestran el aumento de la cuota de mercado de los supermercados y un creciente interés en la agricultura por contrato y en las actividades de comercialización colectiva.

Dadas los avances tecnológicos que se viven en nuestros días es más que una ventaja una desventaja el no tener incorporadas estas tecnologías en algunas empresas o microempresas del sector agrícola como lo son las páginas mostradas en el punto anterior.

7.1.1.3 Herramientas de desarrollo

Hoy en día existe un gran mercado en el sector del desarrollo web, y con esto se han creado distintas tecnologías para la creación de plataformas en línea. Una de las más utilizadas en este tiempo vendría siendo Angular por Las Single Page Application son páginas que cargan únicamente un archivo HTML en el cliente y

[11] MINAGRICULTURA. (2018). Agrosavia. Septiembre 30, 2018, de MINAGRICULTURA Sitio web: http://www.corpoica.org.co/noticias/generales/somos-agrosavia/

que su contenido cambia dinámicamente a medida el usuario interactúa con ella, lo cual vemos con los diferentes servicios ofrecidos por nuestra plataforma.[12]

JavaScript es uno de los referentes internacionales en la programación de proyectos, el cual utilizaremos para nuestra plataforma web, para generar acciones de back-end donde utilizaremos varias características fundamentales que en cierta manera ofrece este lenguaje como lo son Sistema bidireccional de data-binding, patrón de Inyección de Dependencias, entre otros que podremos ver más adelante, dada la arquitectura de nuestra aplicación es la mejor opción ya que tiene una alta cohesión con las otras herramientas utilizadas en nuestro aplicativo web.[13]

Todos los datos de Firebase Realtime Database se almacenan como objetos JSON. La base de datos puede conceptualizarse como un árbol JSON alojado en la nube. A diferencia de una base de datos de SQL, no hay tablas ni registros. Cuando le agregas datos al árbol JSON, estos se convierten en un nodo de la estructura JSON existente con una clave asociada.[14]

- Productos: En este nodo se registrará los datos de los productos.
- Transportes: En este nodo se registrará los datos de los transportes.
- Productos Postulados: En este nodo se registrarán los productos postulados por los clientes.
- Transportes Postulados: En este nodo se registrarán los transportes postulados por los clientes.

HTML es el lenguaje que se utiliza para crear las páginas web a las que se accede mediante internet. Más concretamente, HTML será el lenguaje que trabajaremos para la vista de nuestra plataforma web generando la vista de los elementos propuestos. HTML será utilizado mayormente para generar las vistas o el front-end de nuestra plataforma utilizando algunos recursos del lenguaje.[15]

CSS (Del inglés Cascading Style Sheets) permite crear páginas web con un diseño agradable, es utilizado mayormente para los estilos de la vista o lo que solemos

[12] Garcia A. «Medium. ¿Que es Angular?,» 16 Febrero, 2019. [En línea]. Disponible en: https://medium.com/ngesyfirebase/qu%C3%A9-es-angular-877c99a607f4. [Último acceso: 15 Marzo 2019].

[13] Bbvaopen4u, Herramientas básicas para desarrolladores de JavaScript. 11 septiembre, 2015 Disponible en: https://bbvaopen4u.com/es/actualidad/herramientas-basicas-para-desarrolladores-en-javascript. [Último acceso: 12 Marzo 2019].

[14] Firebase. (2019) Estructura tu base de datos | Firebase Realtime Database | Base de fuego. [en línea]. Disponible en: https://firebase.google.com/docs/database/web/structure-data?hl=es-419. [revisado 25 de marzo de 2018].

[15] Universidad de Murcia, Introducción a HTML y CSS. Desarrollo de Aplicaciones Web. Febrero 11, 2019. [en línea]. Disponible en: https://www.um.es/docencia/barzana/DAWEB/2017-18/daweb-tema-1-introduccion-html-css.html. [Último acceso: 27 febrero 2019].

front-end volviendo en cierta manera más agradable a la vista de los usuarios generando algunos diseños interesantes e interactivos para el agrado del usuario.[16]

PHP (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML, lo cual nos brinda una mayor ayuda en la elaboración de nuestro aplicativo web. PHP lo utilizaremos para algunas funciones o el back-end de la plataforma web.[17]

1.7.2 Metodología de Desarrollo

1.7.2.1 Descripción de Scrum

La metodología Scrum es para el desarrollo de software ágil y flexible ya que busca la forma de optimizar los recursos del proyecto al máximo, permite la retroalimentación continua, de modo que si se cambia alguna parte del proyecto este se pueda ajustar sin dificultad, permitiendo también un buen mantenimiento al proyecto, se trabaja por medio de interacciones en equipo, cuyo objetivo principal es satisfacer los requerimientos del cliente[18].

1.7.2.2 Roles

1.7.2.2.1 Roles centrales

Son las personas que están involucradas debido a que son las personas a cargo de la creación del proyecto, estos se dividen en tres equipos:

- El Product Owner es el responsable de desarrollar, mantener y priorizar las tareas.
- El Scrum Master es responsable de asegurarse que el trabajo del equipo tenga una óptima comunicación. Además, se encarga de resolver cualquier problema que pueda encontrar el equipo de desarrollo.
- Los Miembros del Equipo de desarrollo son los encargados de escribir y probar el código[19].

---

[16] MDN web docs. Cómo funciona CSS. marzo 18, 2019. [en línea]. Disponible en: https://developer.mozilla.org/es/docs/Learn/CSS/Introduction_to_CSS/Como_funciona_CSS. [Último acceso: 27 marzo 2019].

[17] php. ¿Qué es PHP? [en línea]. Disponible en: https://www.php.net/manual/es/intro-whatis.php. [Último acceso: 16 marzo 2019].

[18] TRIGAS GALLEGO, Manuel. GESTIÓN DE PROYECTOS INFORMÁTICOS. Septiembre 21, 2018, de GESTIÓN DE PROYECTOS INFORMÁTICOS Sitio web: http://openaccess.uoc.edu/webapps/o2/bitstream/10609/17885/1/mtrigasTFC0612memoria.pdf

[19] Yesica Lizeth. ¿Qué es Scrum y los roles en Scrum? Septiembre 21, 2018, de Platzi Sitio web: https://platzi.com/blog/que-es-scrum-y-los-roles-en-scrum/

1.7.2.2.2 Roles no centrales

Son importantes para el proyecto, pero de ellos no depende el éxito o el fracaso del proyecto, ya que no se involucran directamente con el desarrollo del proyecto, estos son:

- Stakeholders son las personas que se verán beneficiadas de alguna manera por el proyecto.
- Usuarios son los últimos en manejar el proyecto ya finalizado.
- Managers son los encargados de verificar los objetivos y requerimientos finales del proyecto.

1.7.2.3 Desarrollo de las fases

Para el desarrollo de las fases de la metodología SCRUM es necesario saber un concepto básico sobre que es Sprint, el cual es una iteración realizada por los miembros del equipo, para el desarrollo de un Sprint se lleva a cabo mediante las fases de la metodología[20].

1.7.2.3.1 Planeación del sprint

En esta fase todos los miembros del equipo de trabajo del proyecto se reúnen para definir los requerimientos, grupos de trabajo y las tareas que a cada grupo le corresponde, definiendo también los tiempos de entrega de cada tarea asignada.

1.7.2.3.2 Reunión de equipo

Las reuniones se realizan con el fin de que los miembros del equipo de un reporte de su avance y si se le ha presentado algún problema para que así entre todos se pueda llegar alguna solución.

1.7.2.3.3 Refinamiento del Backlog

Esta fase se realiza con el fin de aclarar cualquier duda que tenga algún miembro del equipo y para revisar si el tiempo y recursos invertidos en cada tarea y requerimiento son los necesarios.

[20] GARCIA DE LEON, Alicia. (2002). ETAPAS DE LA CREACIÓN DE UN SITIO WEB. Lima: Biblios.

1.7.2.3.4 Revisión del sprint

En esta fase se realiza una reunión con el cliente donde se le demostrara la finalización de cada uno de los requerimientos solicitados, mediante una explicación por uno de los miembros del equipo encargado de cada uno de los requerimientos.

1.7.2.3.5 Retrospectiva del sprint

En esta fase se hace una reunión final con todos los miembros del equipo para determinar qué errores se cometieron, qué soluciones se dieron, que cosas se hicieron bien y qué problemas encontraron.

1.7.2.4 Herramientas

1.7.2.4.1 Backlog de producto

Es una herramienta que ofrece información general del proyecto como un inventario de todos los requerimientos del proyecto de una manera ordenada.

1.7.2.4.2 Historias de usuario

Tiene la información de cómo funciona los requerimientos que se están manejando en el proyecto y es el que le informa al cliente en caso de algún cambio sobre ellos.

1.7.2.4.3 Backlog del Sprint

Son los elementos aceptados que se tomaron de las reuniones del equipo y fueron ordenados, terminan siendo los requerimientos del proyecto.

1.7.2.4.4 El panel de Tareas

Es una herramienta diseñada como tabla con la función de mostrar los estados de las tareas asignadas y el tiempo que tomó realizar dicha tarea y a quien fue asignada.

1.7.2.4.5 Definición de “Listo”

Esta herramienta sirve para verificar que todas las tareas se hayan cumplido, que las pruebas se realizarán correctamente, la revisión por parte de los clientes, y qué condiciones se cumplieran.

1.8 RESULTADOS ESPERADOS

Este proyecto se ideó con el fin de realizar una plataforma web para los sectores periféricos de Bogotá con el fin de que se permita una comercialización de los productos agrícolas que beneficie a los habitantes del sector como lo son los campesinos y se realice una negociación directa entre el proveedor, quien es el que cultiva los productos y el consumidor sin la intervención de terceras personas. Se espera alcanzar con este proyecto:

- Crear, modificar y eliminar los productos agrícolas que hay en los sectores periféricos de Bogotá.
- Brindar el servicio de transporte si está disponible.
- Por medio de un catálogo mostrar los productos disponibles.
- Generar un reporte de los productos agrícolas vendidos mediante la plataforma.

1.9 IMPACTO

1.9.1 Social

Este proyecto beneficiara a los habitantes de los sectores periféricos y al consumidor promedio ya que no habría terceros intermediarios para obtener el producto.

1.9.2 Ambiental

Con este proyecto se podría explotar más la fuente de ingresos principal de estos sectores ya que se lograría una fuerte demanda y así evitar los proyectos por parte de la alcaldía que se quieren realizar allí que amenazan el medio ambiente.

1.9.3 Tecnológico

En estos sectores periféricos no tienen acceso a los avances tecnológicos que se brindan a las empresas nacionales en el país, realizando este proyecto ya no estarían en una gran desventaja en nivel tecnológico y cambiaría la manera tradicional que tienen los habitantes de comercializar los productos que cultivan.

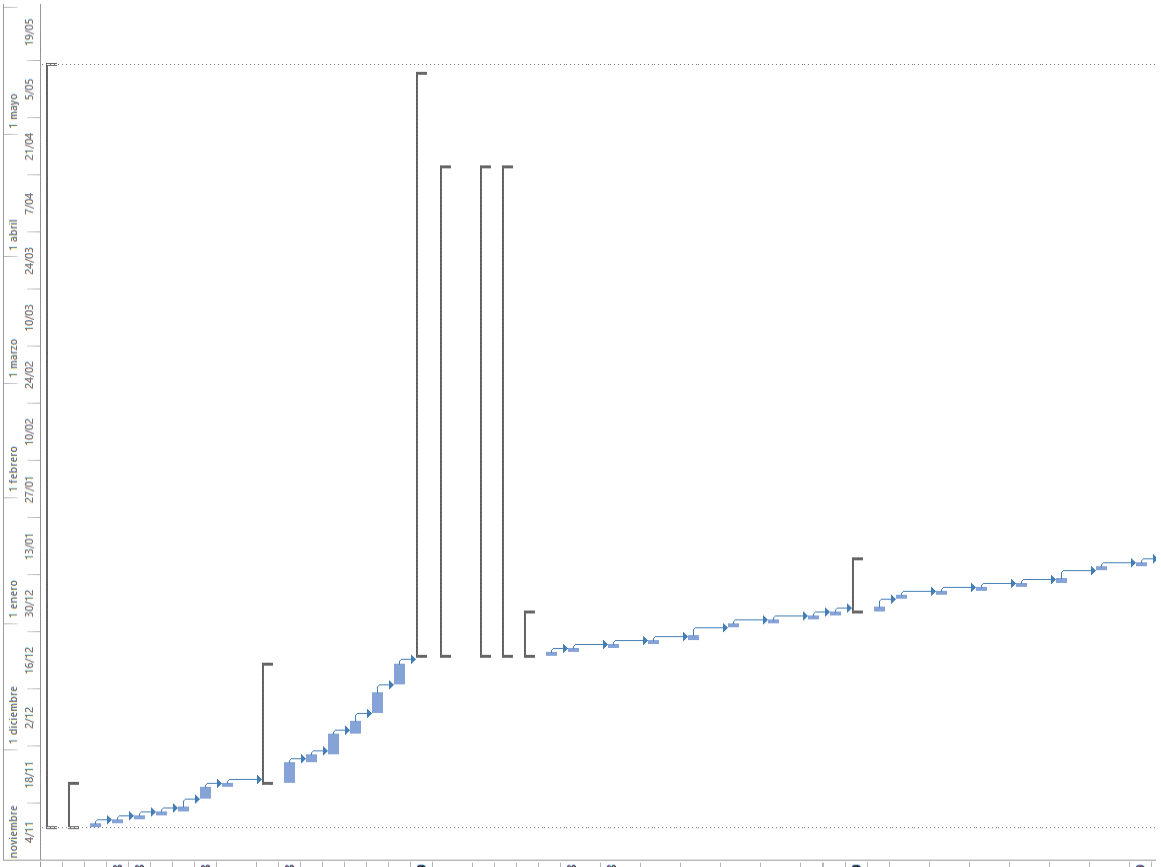
1.9.4 Innovación

Con este proyecto se mejoraría la actividad principal de estos sectores y se abriría la puerta para que se implementen nuevas tecnologías que permitan un manejo productivo de las tierras.

1.9.5 Económico

El proyecto permitiría a los habitantes de estos sectores obtener mayor ganancia ya que no tendrían terceros involucrados en la comercialización de los productos.

1.10 CRONOGRAMA



	Nombre de tarea	Duración	Comienzo	Fin
1	**Proyecto**	**27 sem.**	**lun 12/11/18**	**vie 17/05/19**
2	**Incio**	**1,8 sem.**	**lun 12/11/18**	**jue 22/11/18**
3	Planteamiento del problema	1 día	lun 12/11/18	lun 12/11/18
4	Alcances y limitaciones	1 día	mar 13/11/18	mar 13/11/18
5	Objetivos	1 día	mié 14/11/18	mié 14/11/18
6	Resultados esperados	1 día	jue 15/11/18	jue 15/11/18
7	Impacto	1 día	vie 16/11/18	vie 16/11/18
8	Cronograma	3 días	lun 19/11/18	mié 21/11/18
9	Presupuesto y fuentes de financiacion	1 día	jue 22/11/18	jue 22/11/18
10	**Planeacion del Sprint**	**4,2 sem.**	**vie 23/11/18**	**vie 21/12/18**
11	Definir roles	3 días	vie 23/11/18	mar 27/11/18
12	Definir actividades por rol	2 días	mié 28/11/18	jue 29/11/18
13	Definir requerimientos	3 días	vie 30/11/18	mar 4/12/18
14	Produc Blacklog	3 días	mié 5/12/18	vie 7/12/18
15	Sprint o Iteraciones	5 días	lun 10/12/18	vie 14/12/18
16	Designar tareas	5 días	lun 17/12/18	vie 21/12/18
17	**Reunion del Equipo**	**20,6 sem.**	**lun 24/12/18**	**mié 15/05/19**
18	**Refinamiento del Backlog**	**17,2 sem.**	**lun 24/12/18**	**lun 22/04/19**
19	**Panel de tareas**	**17,2 sem.**	**lun 24/12/18**	**lun 22/04/19**
20	**Diseñar los Sprint**	**17,2 sem.**	**lun 24/12/18**	**lun 22/04/19**
21	**Definicion de los Sprint**	**1,8 sem.**	**lun 24/12/18**	**jue 3/01/19**
22	Sprint 1: Login	1 día	lun 24/12/18	lun 24/12/18
23	Sprint 2: Carrito de compras	1 día	mar 25/12/18	mar 25/12/18
24	Sprint 3: Gestión de productos	1 día	mié 26/12/18	mié 26/12/18
25	Sprint 4: Gestión de Transporte	1 día	jue 27/12/18	jue 27/12/18
26	Sprint 5: Gestión productos postulados	1 día	vie 28/12/18	vie 28/12/18
27	Sprint 6: Gestión transportes postulados	1 día	lun 31/12/18	lun 31/12/18
28	Sprint 7: Gestión de inquietudes	1 día	mar 1/01/19	mar 1/01/19
29	Sprint 8: Informe	1 día	mié 2/01/19	mié 2/01/19
30	Reunion 1	1 día	jue 3/01/19	jue 3/01/19
31	**Tareas de los Sprint**	**1,8 sem.**	**vie 4/01/19**	**mié 16/01/19**
32	Sprint 1: Login	1 día	vie 4/01/19	vie 4/01/19
33	Sprint 2: Carrito de compras	1 día	lun 7/01/19	lun 7/01/19
34	Sprint 3: Gestión de productos	1 día	mar 8/01/19	mar 8/01/19
35	Sprint 4: Gestión de Transporte	1 día	mié 9/01/19	mié 9/01/19
36	Sprint 5: Gestión productos postulados	1 día	jue 10/01/19	jue 10/01/19
37	Sprint 6: Gestión transportes postulados	1 día	vie 11/01/19	vie 11/01/19
38	Sprint 7: Gestión de inquietudes	1 día	lun 14/01/19	lun 14/01/19
39	Sprint 8: Informe	1 día	mar 15/01/19	mar 15/01/19

Ilustración 1: Cronograma.

51	◢ **Proceso de desarrollo de los Sprint**	**5 sem.**	**lun 11/02/19**	**vie 15/03/19**
52	Sprint 1: Login	3 días	lun 11/02/19	mié 13/02/19
53	Sprint 2: Carrito de compras	3 días	jue 14/02/19	lun 18/02/19
54	Sprint 3: Gestión de	3 días	mar 19/02/19	jue 21/02/19
55	Sprint 4: Gestión de Transporte	3 días	vie 22/02/19	mar 26/02/19
56	Sprint 5: Gestión productos postulados	3 días	mié 27/02/19	vie 1/03/19
57	Sprint 6: Gestión transportes postulados	3 días	lun 4/03/19	mié 6/03/19
58	Sprint 7: Gestión de inquietudes	3 días	jue 7/03/19	lun 11/03/19
59	Sprint 8: Informe	3 días	mar 12/03/19	jue 14/03/19
60	Reunion 4	1 día	vie 15/03/19	vie 15/03/19
61	◢ **Diagrama de contexto de los Sprint**	**1,8 sem.**	**lun 18/03/19**	**jue 28/03/19**
62	Sprint 1: Login	1 día	lun 18/03/19	lun 18/03/19
63	Sprint 2: Carrito de compras	1 día	mar 19/03/19	mar 19/03/19
64	Sprint 3: Gestión de productos	1 día	mié 20/03/19	mié 20/03/19
65	Sprint 4: Gestión de Transporte	1 día	jue 21/03/19	jue 21/03/19
66	Sprint 5: Gestión productos postulados	1 día	vie 22/03/19	vie 22/03/19
67	Sprint 6: Gestión transportes postulados	1 día	lun 25/03/19	lun 25/03/19
68	Sprint 7: Gestión de inquietudes	1 día	mar 26/03/19	mar 26/03/19
69	Sprint 8: Informe	1 día	mié 27/03/19	mié 27/03/19
70	Reunion 5	1 día	jue 28/03/19	jue 28/03/19
71	◢ **Prueba de los Sprint**	**3,4 sem.**	**vie 29/03/19**	**lun 22/04/19**
72	Sprint 1: Login	2 días	vie 29/03/19	lun 1/04/19
73	Sprint 2: Carrito de compras	2 días	mar 2/04/19	mié 3/04/19
74	Sprint 3: Gestión de productos	2 días	jue 4/04/19	vie 5/04/19
75	Sprint 4: Gestión de Transporte	2 días	lun 8/04/19	mar 9/04/19
76	Sprint 5: Gestión productos postulados	2 días	mié 10/04/19	jue 11/04/19
77	Sprint 6: Gestión transportes postulados	2 días	vie 12/04/19	lun 15/04/19
78	Sprint 7: Gestión de inquietudes	2 días	mar 16/04/19	mié 17/04/19
79	Sprint 8: Informe	2 días	jue 18/04/19	vie 19/04/19
80	Reunion 6	1 día	lun 22/04/19	lun 22/04/19
81	◢ **Revision del Sprint**	**3,4 sem.**	**mar 23/04/19**	**mié 15/05/19**
82	◢ **Pruebas tecnicas**	**3,2 sem.**	**mar 23/04/19**	**mar 14/05/19**
83	Ejecucion de las prubas según los requerimientos establecidos	15 días	mar 23/04/19	lun 13/05/19
84	Reunion 7	1 día	mar 14/05/19	mar 14/05/19
85	◢ **Pruebas cliente**	**0,2 sem.**	**mié 15/05/19**	**mié 15/05/19**
86	Realizar demostracion al cliente	1 día	mié 15/05/19	mié 15/05/19
87	◢ **Retrospectiva del Sprint**	**0,4 sem.**	**jue 16/05/19**	**vie 17/05/19**
88	Conclusiones del proyecto	2 días	jue 16/05/19	vie 17/05/19
89	Fin	0 sem.	vie 17/05/19	vie 17/05/19

Ilustración 2: Cronograma 2.

	Nombre de tarea	Duración	Comienzo	Fin
1	**Proyecto**	**27 sem.**	**lun 12/11/18**	**vie 17/05/19**
2	**Incio**	**1,8 sem.**	**lun 12/11/18**	**jue 22/11/18**
3	Planteamiento del problema	1 día	lun 12/11/18	lun 12/11/18
4	Alcances y limitaciones	1 día	mar 13/11/18	mar 13/11/18
5	Objetivos	1 día	mié 14/11/18	mié 14/11/18
6	Resultados esperados	1 día	jue 15/11/18	jue 15/11/18
7	Impacto	1 día	vie 16/11/18	vie 16/11/18
8	Cronograma	3 días	lun 19/11/18	mié 21/11/18
9	Presupuesto y fuentes de financiacion	1 día	jue 22/11/18	jue 22/11/18
10	**Planeacion del Sprint**	**4,2 sem.**	**vie 23/11/18**	**vie 21/12/18**
11	Definir roles	3 días	vie 23/11/18	mar 27/11/18
12	Definir actividades por rol	2 días	mié 28/11/18	jue 29/11/18
13	Definir requerimientos	3 días	vie 30/11/18	mar 4/12/18
14	Produc Blacklog	3 días	mié 5/12/18	vie 7/12/18
15	Sprint o Iteraciones	5 días	lun 10/12/18	vie 14/12/18
16	Designar tareas	5 días	lun 17/12/18	vie 21/12/18
17	**Reunion del Equipo**	**20,6 sem.**	**lun 24/12/18**	**mié 15/05/19**
18	**Refinamiento del Backlog**	**17,2 sem.**	**lun 24/12/18**	**lun 22/04/19**
19	**Panel de tareas**	**17,2 sem.**	**lun 24/12/18**	**lun 22/04/19**
20	**Diseñar los Sprint**	**17,2 sem.**	**lun 24/12/18**	**lun 22/04/19**
21	**Definicion de los Sprint**	**1,8 sem.**	**lun 24/12/18**	**jue 3/01/19**
22	Sprint 1: Login	1 día	lun 24/12/18	lun 24/12/18
23	Sprint 2: Carrito de compras	1 día	mar 25/12/18	mar 25/12/18
24	Sprint 3: Gestión de productos	1 día	mié 26/12/18	mié 26/12/18
25	Sprint 4: Gestión de Transporte	1 día	jue 27/12/18	jue 27/12/18
26	Sprint 5: Gestión productos postulados	1 día	vie 28/12/18	vie 28/12/18
27	Sprint 6: Gestión transportes postulados	1 día	lun 31/12/18	lun 31/12/18
28	Sprint 7: Gestión de inquietudes	1 día	mar 1/01/19	mar 1/01/19
29	Sprint 8: Informe	1 día	mié 2/01/19	mié 2/01/19
30	Reunion 1	1 día	jue 3/01/19	jue 3/01/19
31	**Tareas de los Sprint**	**1,8 sem.**	**vie 4/01/19**	**mié 16/01/19**
32	Sprint 1: Login	1 día	vie 4/01/19	vie 4/01/19
33	Sprint 2: Carrito de compras	1 día	lun 7/01/19	lun 7/01/19
34	Sprint 3: Gestión de productos	1 día	mar 8/01/19	mar 8/01/19
35	Sprint 4: Gestión de Transporte	1 día	mié 9/01/19	mié 9/01/19
36	Sprint 5: Gestión productos postulados	1 día	jue 10/01/19	jue 10/01/19
37	Sprint 6: Gestión transportes postulados	1 día	vie 11/01/19	vie 11/01/19
38	Sprint 7: Gestión de inquietudes	1 día	lun 14/01/19	lun 14/01/19
39	Sprint 8: Informe	1 día	mar 15/01/19	mar 15/01/19
40	Reunion 2	1 día	mié 16/01/19	mié 16/01/19
41	**Diseño de los Sprint**	**3,4 sem.**	**jue 17/01/19**	**vie 8/02/19**
42	Sprint 1: Login	2 días	jue 17/01/19	vie 18/01/19
43	Sprint 2: Carrito de compras	2 días	lun 21/01/19	mar 22/01/19
44	Sprint 3: Gestión de productos	2 días	mié 23/01/19	jue 24/01/19
45	Sprint 4: Gestión de Transporte	2 días	vie 25/01/19	lun 28/01/19
46	Sprint 5: Gestión productos postulados	2 días	mar 29/01/19	mié 30/01/19
47	Sprint 6: Gestión transportes postulados	2 días	jue 31/01/19	vie 1/02/19
48	Sprint 7: Gestión de inquietudes	2 días	lun 4/02/19	mar 5/02/19
49	Sprint 8: Informe	2 días	mié 6/02/19	jue 7/02/19
50	Reunion 3	1 día	vie 8/02/19	vie 8/02/19
51	**Proceso de desarrollo de los Sprint**	**5 sem.**	**lun 11/02/19**	**vie 15/03/19**
52	Sprint 1: Login	3 días	lun 11/02/19	mié 13/02/19
53	Sprint 2: Carrito de compras	3 días	jue 14/02/19	lun 18/02/19
54	Sprint 3: Gestión de	3 días	mar 19/02/19	jue 21/02/19
55	Sprint 4: Gestión de Transporte	3 días	vie 22/02/19	mar 26/02/19
56	Sprint 5: Gestión productos postulados	3 días	mié 27/02/19	vie 1/03/19
57	Sprint 6: Gestión transportes postulados	3 días	lun 4/03/19	mié 6/03/19
58	Sprint 7: Gestión de inquietudes	3 días	jue 7/03/19	lun 11/03/19
59	Sprint 8: Informe	3 días	mar 12/03/19	jue 14/03/19
60	Reunion 4	1 día	vie 15/03/19	vie 15/03/19
61	**Diagrama de contexto de los Sprint**	**1,8 sem.**	**lun 18/03/19**	**jue 28/03/19**
62	Sprint 1: Login	1 día	lun 18/03/19	lun 18/03/19
63	Sprint 2: Carrito de compras	1 día	mar 19/03/19	mar 19/03/19
64	Sprint 3: Gestión de productos	1 día	mié 20/03/19	mié 20/03/19
65	Sprint 4: Gestión de Transporte	1 día	jue 21/03/19	jue 21/03/19
66	Sprint 5: Gestión productos postulados	1 día	vie 22/03/19	vie 22/03/19
67	Sprint 6: Gestión transportes postulados	1 día	lun 25/03/19	lun 25/03/19
68	Sprint 7: Gestión de inquietudes	1 día	mar 26/03/19	mar 26/03/19
69	Sprint 8: Informe	1 día	mié 27/03/19	mié 27/03/19
70	Reunion 5	1 día	jue 28/03/19	jue 28/03/19
71	**Prueba de los Sprint**	**3,4 sem.**	**vie 29/03/19**	**lun 22/04/19**
72	Sprint 1: Login	2 días	vie 29/03/19	lun 1/04/19
73	Sprint 2: Carrito de compras	2 días	mar 2/04/19	mié 3/04/19
74	Sprint 3: Gestión de productos	2 días	jue 4/04/19	vie 5/04/19
75	Sprint 4: Gestión de Transporte	2 días	lun 8/04/19	mar 9/04/19
76	Sprint 5: Gestión productos postulados	2 días	mié 10/04/19	jue 11/04/19
77	Sprint 6: Gestión transportes postulados	2 días	vie 12/04/19	lun 15/04/19
78	Sprint 7: Gestión de inquietudes	2 días	mar 16/04/19	mié 17/04/19
79	Sprint 8: Informe	2 días	jue 18/04/19	vie 19/04/19
80	Reunion 6	1 día	lun 22/04/19	lun 22/04/19
81	**Revision del Sprint**	**3,4 sem.**	**mar 23/04/19**	**mié 15/05/19**
82	**Pruebas tecnicas**	**3,2 sem.**	**mar 23/04/19**	**mar 14/05/19**
83	Ejecucion de las pruebas según los requerimientos establecidos	15 días	mar 23/04/19	lun 13/05/19
84	Reunion 7	1 día	mar 14/05/19	mar 14/05/19
85	**Pruebas cliente**	**0,2 sem.**	**mié 15/05/19**	**mié 15/05/19**
86	Realizar demostracion al cliente	1 día	mié 15/05/19	mié 15/05/19
87	**Retrospectiva del Sprint**	**0,4 sem.**	**jue 16/05/19**	**vie 17/05/19**
88	Concluciones del proyecto	2 días	jue 16/05/19	vie 17/05/19
89	Fin	0 sem.	vie 17/05/19	vie 17/05/19

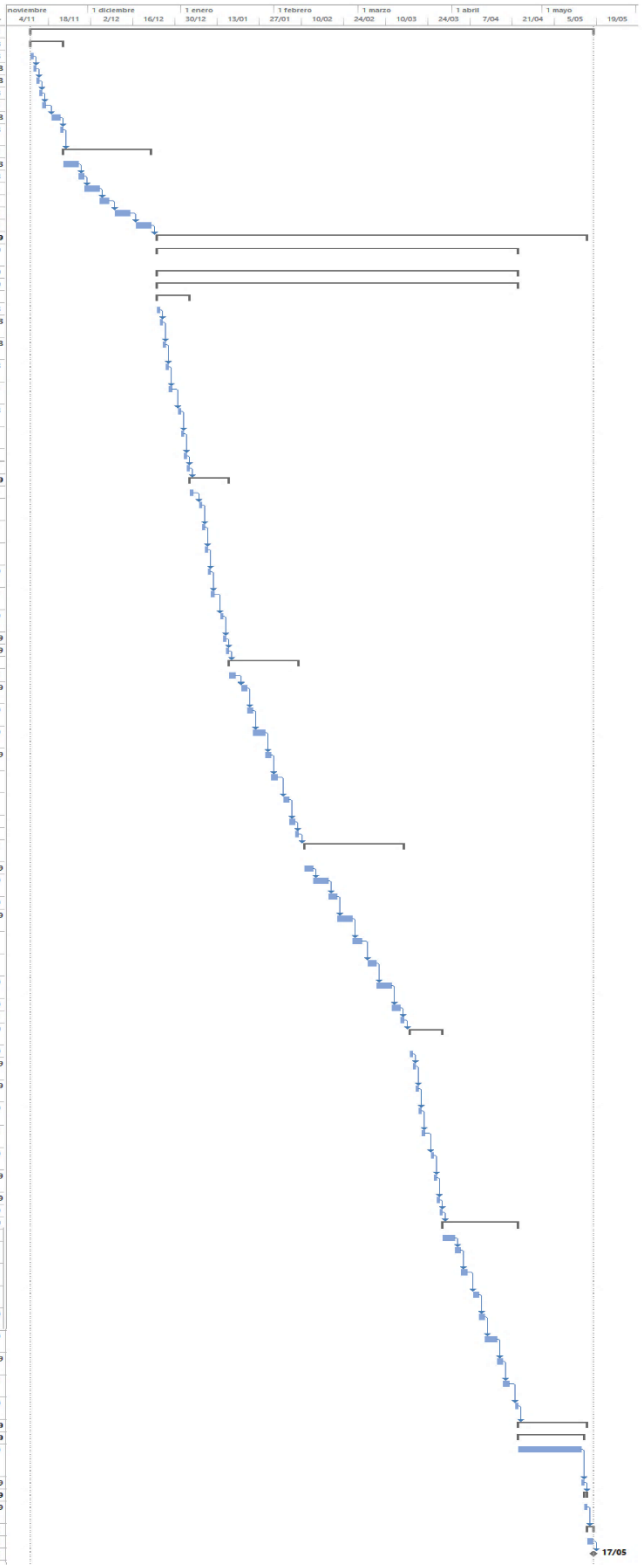


Ilustración 3: Cronograma.

1.11 PRESUPUESTO Y FUENTES DE FINANCIACIÓN

En la siguiente tabla se presenta las asesorías que se tendrán y los gastos de los desarrolladores.

Tipo	Descripción	Valor-Hora	Cantidad	Total
Analista	Un analista encargado para diseño y obtención de los algoritmos que se requieran.	$ 8333	8 horas semanales	$ 5.000.000
Desarrolladores	Dos programadores que realicen la implementación del software.	$ 20.832	8 horas semanales	$ 9.374.400
Total Recursos Humanos				$ 14.374.400

Tabla 2: Factibilidad económica recursos humanos.

A continuación, en la Tabla 3 se presentarán los gastos de los recursos que se ostentan en el desarrollo del proyecto.

Recurso	Descripción	Valor Unitario	Cantidad	Total
Computadores (Servidor - clientes)	Equipos de escritorio para el desarrollo y las pruebas del sistema.	$ 1.700.000	2	$ 3.400.000
Total Recursos Técnicos				$ 3.400.000

Tabla 3: Factibilidad económica recursos técnicos.

Adicionalmente se muestran los gastos adicionales en la Tabla 4 que serán solventados por desarrolladores del proyecto.

Recurso	Valor
Total Recursos Humanos	$ 14.374.400
Total Recursos Técnicos	$ 5.100.000
Total Otros recursos	$50.000
Costos imprevistos (10%)	$ 1.827.000
TOTAL COSTO	$21.351.400

Tabla 4: Factibilidad económica costo total.

2. FASE 2: ANALISIS DEL PROYECTO

2.1 IDENTIFICACIÓN DE LOS ROLES

En el desarrollo de la plataforma web identificamos unos requerimientos, para lo cual también se hace una identificación de los roles, a los que se les determina con

el nombre de la o las personas responsables y una descripción acertada de sus funciones.

ROL	RESPONSABLE	DESCRIPCION
Product Owner	Brain Alejandro Tribaldos Valero	Es el responsable de desarrollar, mantener y priorizar las tareas.
Scrum Master	July Tatiana Jimenez Rozo	Es responsable de asegurarse que el trabajo del equipo tenga una óptima comunicación. Además, se encarga de resolver cualquier problema que pueda encontrar el equipo de desarrollo.
Team	Brain Alejandro Tribaldos Valero, July Tatiana Jimenez Rozo	Es el grupo de personas que se encargan de la realización del proyecto (escribir y probar el código).

Tabla 5: Identificación de roles.

2.2 LISTA DE ACTIVIDADES POR ROL

Se asignan las actividades que debe realizar cada uno de los responsables asignados a cada rol.

ROL	RESPONSABLE	ACTIVIDAD
Product Owner	Brain Alejandro Tribaldos Valero	• Participar como un miembro más del Team. • Define los objetivos del proyecto. • Detecta riesgos que pueda haber durante la fase de sprint. • Busca actualizaciones para las tareas a desarrollar. • Demostrar y explicar los elementos del Product Blacklog.

Scrum Master	July Tatiana Jimenez Rozo	• Asegurarse que todos los integrantes del grupo sigan las reglas y procesos del Scrum. • Tener conocimiento del proceso de Scrum. • Aportar a las reuniones de Scrum. • Resolver cualquier problema que se le presenten al equipo, con el fin de conseguir el objetivo de cada sprint.
Team	Brain Alejandro Tribaldos Valero, July Tatiana Jimenez Rozo	• Trabajar de manera grupal para conseguir los objetivos del sprint o iteración. • Estimar la complejidad de cada requisito en la lista de requisitos priorizada del producto o proyecto. • Identificar todas las tareas necesarias para completar cada requisito. • Estimar el esfuerzo necesario para realizar cada tarea.

Tabla 6: Lista de actividades por rol.

2.3 PRODUCTO BACKLOG

En la siguiente tabla se mostrará la lista de requerimientos del proyecto.

ID	DESCRIPCION	ALIAS	PRIORIDAD	ESTIMACION DE ESFUERZO (1-6)	RESPONSABLE	SPRINT
1	Validar la información de los formularios.	Validar formulario.	Alta	6	Brain Tribaldos, July Jimenez.	3,4,5,6
2	Validar inicio de sesión de usuarios.	Validar usuario.	Alta	6	Brain Tribaldos, July Jimenez.	1,3,4,5,6
3	Encriptar datos de información del cliente e información del pedido.	Encriptar datos.	Alta	6	Brain Tribaldos, July Jimenez.	
4	Crear producto por el administrador.	Crear productos.	Alta	6	Brain Tribaldos, July Jimenez.	3
5	Modificar producto por el administrador.	Modificar producto.	Alta	6	Brain Tribaldos, July Jimenez.	3
6	Eliminar producto por el administrador.	Eliminar producto.	Alta	6	Brain Tribaldos, July Jimenez.	3
7	Crear transporte por el administrador.	Crear transporte.	Alta	6	Brain Tribaldos, July Jimenez.	4
8	Modificar transporte por el administrador.	Modificar transporte.	Alta	6	Brain Tribaldos, July Jimenez.	4
9	Eliminar transporte por el administrador.	Eliminar transporte.	Alta	6	Brain Tribaldos, July Jimenez.	4
10	Creación de la interfaz para todos los módulos de la plataforma .	Crear interfaz.	Alta	6	Brain Tribaldos, July Jimenez.	1, 2,3,4,5,6, 7,8
11	Confirmar datos del usuario y	Confirmar datos	Alta	6	Brain Tribaldos, July Jimenez.	2

	cantidad de productos en el carrito de compras.	carrito.				
12	Modificar producto del carrito de compras.	Modificar carrito.	Media	5	Brain Tribaldos, July Jimenez.	2
13	Eliminar producto del carrito de compras.	Eliminar carrito.	Media	5	Brain Tribaldos, July Jimenez.	2
14	Vaciar carrito de compras.	Vaciar carrito.	Media	5	Brain Tribaldos, July Jimenez.	2
15	Información de pedidos realizados por los cliente.	Informe de pedidos.	Media	5	Brain Tribaldos, July Jimenez.	8
16	Selección de los productos por el cliente.	Selección productos.	Alta	6	Brain Tribaldos, July Jimenez.	2
17	Almacenamiento de información necesaria para el funcionamiento de la plataforma.	Almacenamiento de información.	Alta	6	Brain Tribaldos, July Jimenez.	1
18	Permitir el envío de información a los usuarios de la plataforma.	Envió de información.	Media	5	Brain Tribaldos, July Jimenez.	2
19	Crear producto propuesto por el cliente.	Crear producto propuesto.	Media	5	Brain Tribaldos, July Jimenez.	5
20	Modificar producto propuesto por el cliente.	Modificar producto propuesto.	Media	5	Brain Tribaldos, July Jimenez.	5
21	Eliminar producto propuesto por el cliente.	Eliminar producto propuesto.	Media	5	Brain Tribaldos, July Jimenez.	5
22	Crear transporte	Crear transporte	Media	5	Brain Tribaldos, July Jimenez.	6

	propuesto por el cliente.	propuesto.				
23	Modificar transporte propuesto por el cliente.	Modificar transporte propuesto.	Media	5	Brain Tribaldos, July Jimenez.	6
24	Eliminar transporte propuesto por el cliente.	Eliminar transporte propuesto.	Media	5	Brain Tribaldos, July Jimenez.	6
25	Módulo de preguntas y sugerencias por los usuarios.	Módulo de inquietudes.	Alta	6	Brain Tribaldos, July Jimenez.	7

Tabla 7: Producto Backlog.

2.4 SPRINT O ITERACIONES

Listado de tareas de modo sprint o iteración para el desarrollo del proyecto.

Sprint	Nombre	Descripción
Sprint 1	Login.	Ingreso del usuario con cuentas de Google, Twitter o Facebook y validación para su ingreso a la plataforma.
Sprint 2	Carrito de compras.	Permite confirmar, modificar, eliminar y vaciar el carrito de compras.
Sprint 3	Gestión de productos.	Permite crear, modificar y eliminarlos productos.
Sprint 4	Gestión de transporte.	Permite crear, modificar y eliminar transporte.
Sprint 5	Gestión productos postulados.	Permite crear, modificar y eliminarlos productos postulados.
Sprint 6	Gestión transportes postulados.	Permite crear, modificar y eliminar transportes postulados.
Sprint 7	Gestión de inquietudes.	Permite a los usuarios tener una buena comunicación y resolver preguntas.

Sprint 8	Informe.	Permite mostrar consultas sobre los pedidos, productos y transporte.

Tabla 8: Sprint o iteración.

3. FASE 3: DISEÑO DE LOS SPRINT

3.1 SPRINT 1 LOGIN

Esta fase de la metodología Scrum, el Sprint se basa en validar el ingreso del usuario con Google, Twitter o Facebook.

3.1.1 Definición del Sprint 1

SPRINT 1	NOMBRE	Login
OBJETIVO	Validar el ingreso del usuario a la plataforma por medio de Google, Twitter o Facebook.	
DETALLES		
Se implementará una interfaz en la cual se le mostraran las opciones posibles con las cuales el usuario puede ingresar a la plataforma, teniendo en cuenta que se manejaran dos tipos de usuario: usuario administrador y usuario cliente.		
CRITERIOS DE ACEPTACION		
• Se debe re direccionar el ingreso según el tipo de usuario. • Se debe validar que ingrese con Google, Twitter o Facebook. • El diseño debe de adaptarse a cualquier dispositivo.		

Tabla 9: Definición del Sprint 1

3.1.2 Tareas del Sprint 1

Producto Backlog	Descripción de la tarea	Responsable	Estado	Duración (Días)
2	Validar la información ingresada y con las opciones de Google, Twitter o Facebook.	Brain Tribaldos, July Jimenez.	Terminado	1
10	Se creó un diseño de interfaz que permitiera al usuario acceder a la	Brain Tribaldos, July Jimenez.	Terminado	1

	plataforma.			
17	Consultar los datos de los usuarios que ingresan a la plataforma.	Brain Tribaldos, July Jimenez.	Terminado	1

Tabla 10: Tareas del Sprint 1.

3.1.3 Diseño del Sprint 1

Para este Sprint se diseñó una interfaz en la que el usuario pueda elegir la opción con la cual desea ingresar a la plataforma y valide el tipo de usuario que ingresa.

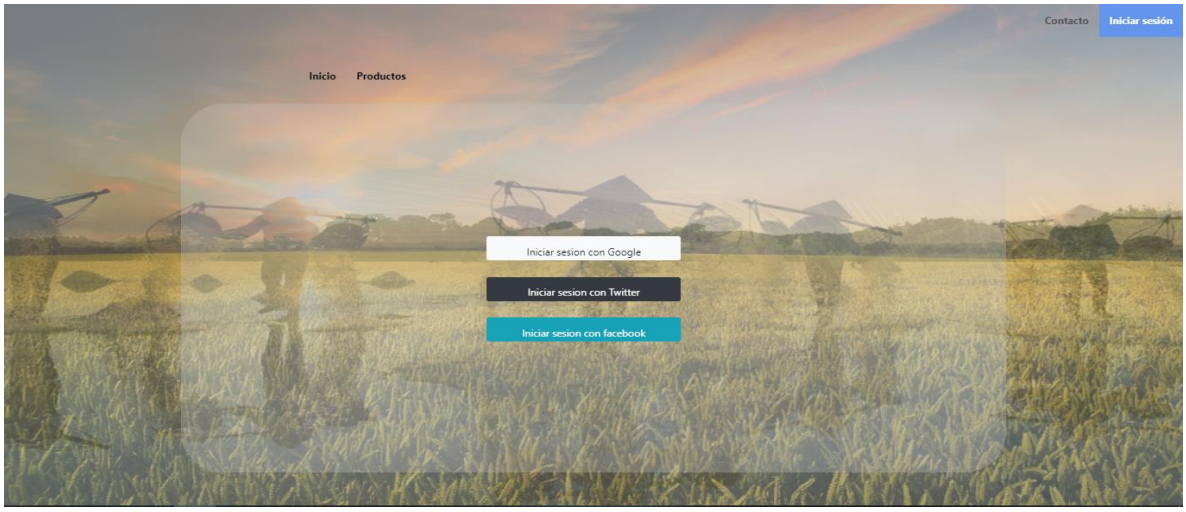


Ilustración 4: Diseño del Sprint 1.

3.1.4 Proceso de desarrollo de software del Sprint 1



Ilustración 5: Desarrollo del Sprint 1.



Ilustración 6: Desarrollo del Sprint 1.

3.1.5 Diagrama de contexto del Sprint 1

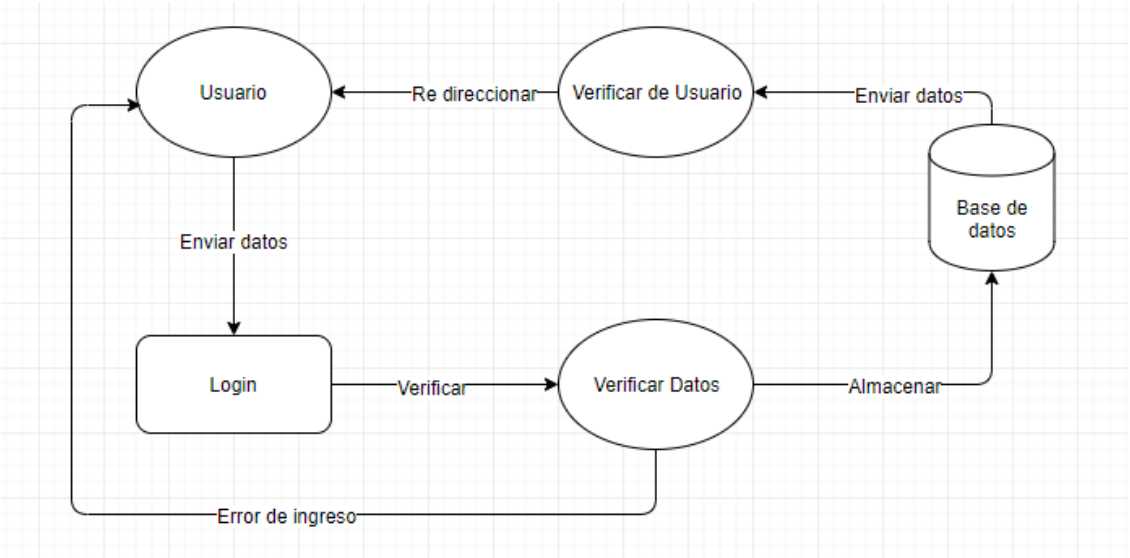


Ilustración 7: Diagrama del Sprint 1.

3.1.6 Prueba del Sprint 1

Sprint 1	Login	
Descripción de la prueba	Se realiza una autenticación por medio de un servicio de FireBase de autenticación de usuarios.	
Herramientas	Opera y FireBase	
Numero	Prueba	Estado

1	Se realiza la autenticación por medio de una ventana emergente de la red social utilizada.	OK
2	Los datos del usuario fueron usados de manera adecuada por medio del servicio de FireBase sin poner en riesgo los datos del usuario.	OK
3	Dependiendo de la red social que escoja se realiza con la respectiva BD de la red social.	OK

Tabla 11: Prueba del Sprint 1.

3.2 SPRINT 2 CARRITO DE COMPRAS

Esta fase de la metodología Scrum, el Sprint se basa en realizar el almacenamiento de los datos del producto en el carrito de compras.

3.2.1 Definición del Sprint 2

SPRINT 2	NOMBRE	Carrito de compras
OBJETIVO	Permitir al cliente almacenar los productos seleccionados, también se puede modificar los productos, eliminarlos y vaciar el carrito de compras.	
DETALLES		
Se implementara una interfaz en donde se guardaran los datos de los productos seleccionados por el cliente, en el cual también se podrá modificar la cantidad de cada producto y eliminar uno o todos los productos seleccionados.		
CRITERIOS DE ACEPTACION		
• Se debe validar el inicio de sesión del cliente para guardar el producto. • Establecer el tipo de transporte de envió (opcional). • Mostrar los datos de los productos de la compra. • El diseño debe de adaptarse a cualquier dispositivo.		

Tabla 12: Definición del Sprint 2.

3.2.2 Tareas del Sprint 2

Producto Backlog	Descripción de la tarea	Responsable	Estado	Duración (Días)
10	Se creó un diseño de interfaz que	Brain Tribaldos, July Jimenez.	Terminado	1

	permitiera al usuario acceder a la plataforma.			
11	Se pide que se confirme los datos de los productos y el usuario.	Brain Tribaldos, July Jimenez.	Terminado	1
12	Se da la opción de modificar la cantidad de los productos.	Brain Tribaldos, July Jimenez.	Terminado	1
13	Se da la opción de eliminar uno o varios productos.	Brain Tribaldos, July Jimenez.	Terminado	1
14	Se realiza la acción de vaciar el carrito.	Brain Tribaldos, July Jimenez.	Terminado	1
16	Permite al cliente seleccionar los productos para agregar al carrito.	Brain Tribaldos, July Jimenez.	Terminado	1
18	Enviar la plataforma un correo al administrador cuando se realice un compra.	Brain Tribaldos, July Jimenez.	Terminado	1

Tabla 13: Tareas del Sprint 2.

3.2.3 Diseño del Sprint 2

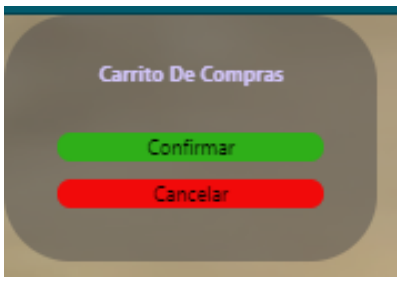


Ilustración 8: Diseño del Sprint 2.



Ilustración 9: Diseño del Sprint 2.

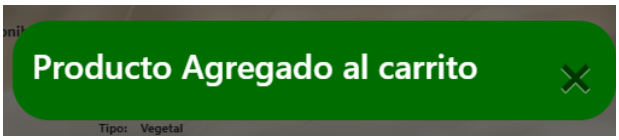


Ilustración 10: Diseño del Sprint 2.

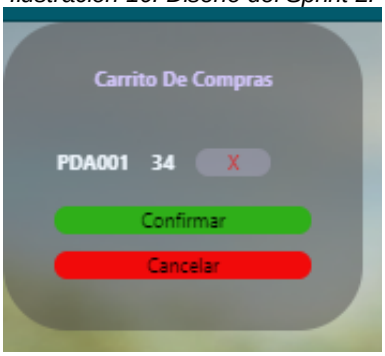


Ilustración 11: Diseño del Sprint 2.

3.2.4 Proceso de desarrollo de software del Sprint 2

```
/*aqui editamos la opcion de adicionar al carrito*/
addProdCar(formCarr: NgForm, producto: productoCarrito) {
if (formCarr.valid) {/*si el formulario es valido adiciona el objeto a la coleccion en firebase*/
producto.AddCar = true;/*deshabilitamos la vista para adicionar*/
this.productoEdit=producto;/*Aqui le asignamos un true para deshabilitar el boton*/
/*Le envia el producto a el objeto que contiene los productos del carrito*/
this._cc.addCarritoProd(this.productoEdit);
  this.openModal();
}else{
this.openModalError();}/*Si el formulario no es valido muestra este mensaje en la consola*/
}
openModal() {
  this.modalRef = this.ModalSer.show(ModalEveComponent, {
    initialState: {
      title: 'Producto Agregado al carrito',
      estMen: 'Add'
    }
  });
}
openModalError() {
  this.modalRef = this.ModalSer.show(ModalEveComponent, {
    initialState: {
      title: 'La cantidad no es valida',
      estMen: 'Er'
    }
  });
}
}
```

Ilustración 12: Desarrollo del Sprint 2.

```
addTransCar(event, transporte: transporteI) {
transporte.AddCar = true;/*deshabilitamos la vista para adicionar*/
this.transporteEdit=transporte;/*Aqui le asignamos un true para deshabilitar el boton*/
/*Le envia el producto a el objeto que contiene los transportes del carrito*/
this._cc.addCarritoTrans(this.transporteEdit);
this.openModal();
this._cc.editStateCarTrans=true;
}
openModal(){
this.modalRef = this.ModalSer.show(ModalEveComponent, {
  initialState: {
    title: 'Transporte adicionado al carrito',
    estMen: 'Add'}
});
}

}
```

Ilustración 13: Desarrollo del Sprint 2.

3.2.5 Diagrama de contexto del Sprint 2

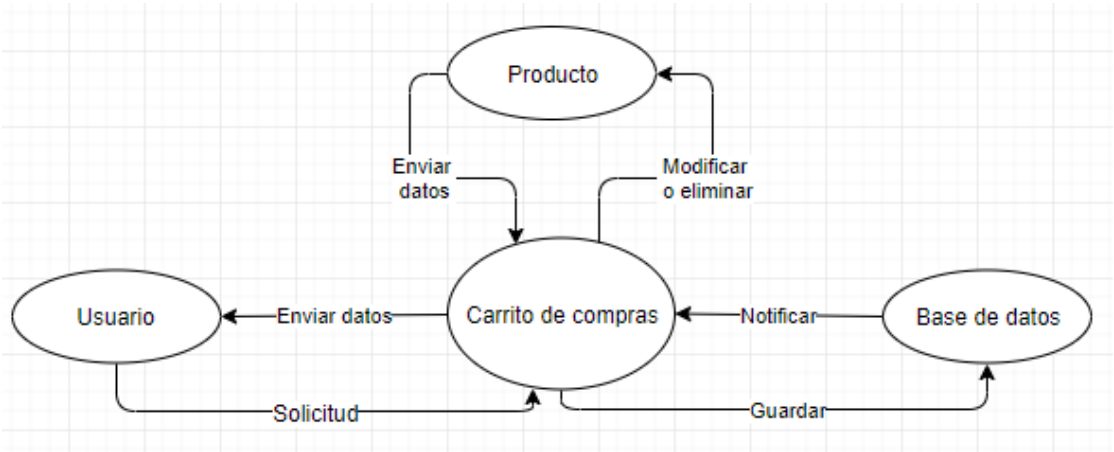


Ilustración 14: Diagrama del Sprint 2.

3.2.6 Prueba del Sprint 2

Sprint 2	Carrito de compras	
Descripción de la prueba	Se utiliza para los pedidos de los usuarios ya sea un listado de productos o un transporte (ya que solo se puede elegir 1 transporte por carrito).	
Herramientas	Opera y FireBase	
Numero	Prueba	Estado
1	Al realizar el proceso de adicionar un transporte se agrega al carrito y ese se muestra como adicionado y los demás se bloquean para no adicionar más y se muestra un mensaje donde se visualiza que ya se ha adicionado un transporte.	OK
2	Al confirmar el carrito de transportes se envía un correo al administrador con algunos datos del usuario y el transporte seleccionado, si no realiza la confirmación o cancelación no podrá ser re direccionado a otros componentes de la página.	OK
3	Al cancelar el carrito este se vacía y deja navegar a usuario con libertad de nuevo, al borrar cada transporte se	OK

	elimina completamente para no generar inconvenientes.	
4	Al realizar el proceso de adicionar varios productos al carrito se muestran como adicionados y se toma la cantidad digitada por el usuario la cual será la que desea de dicho producto.	OK
5	Al confirmar el carrito de productos se envía un correo al administrador con algunos datos del usuario y el catálogo de productos seleccionado, si no realiza la confirmación o cancelación no podrá ser re direccionado a otros componentes de la página.	OK
6	Al cancelar el carrito este se vacía y deja navegar a usuario con libertad de nuevo, al borrar cada producto del producto se elimina completamente para no generar inconvenientes.	OK

Tabla 14: Prueba del Sprint 2.

3.3 SPRINT 3 GESTION DE PRODUCTOS

Esta fase de la metodología Scrum, el Sprint se basa en el manejo de los datos de los productos, permite crear, modificar y eliminar los productos que se encuentran en la base de datos de la plataforma.

3.3.1 Definición del Sprint 3

SPRINT 3	NOMBRE	Gestión de productos
OBJETIVO	Permitir al administrador crear, modificar y eliminar los productos de la base de datos de la plataforma.	
DETALLES		
Se implementara una interfaz que permita al administrador crear, modificar y eliminar los productos de la base de datos de la plataforma.		
CRITERIOS DE ACEPTACION		
• Se debe validar el inicio de sesión que verifique el tipo de usuario. • El administrador debe poder crear, modificar y eliminar los productos. • El diseño debe de adaptarse a cualquier dispositivo.		

Tabla 15: Definición del Sprint 3.

3.3.2 Tareas del Sprint 3

Producto Backlog	Descripción de la tarea	Responsable	Estado	Duración (Días)
1	Se valida la información ingresada en los formularios de crear, modificar y eliminar producto.	Brain Tribaldos, July Jimenez.	Terminado	1
2	Validar la información ingresada y con las opciones de Google, Twitter o Facebook.	Brain Tribaldos, July Jimenez.	Terminado	1
4	Permite al administrador crear el producto.	Brain Tribaldos, July Jimenez.	Terminado	1
5	Permite al administrador modificar el producto.	Brain Tribaldos, July Jimenez.	Terminado	1
6	Permite al administrador eliminar el producto.	Brain Tribaldos, July Jimenez.	Terminado	1
10	Se creó un diseño de interfaz que permitiera al usuario acceder a la plataforma.	Brain Tribaldos, July Jimenez.	Terminado	1

Tabla 16: Tareas del Sprint 3.

3.3.3 Diseño del Sprint 3

Para este Sprint se diseñó una interfaz en la que el usuario pueda observar los datos de los productos que se encuentran registrados en la plataforma para su comercialización.

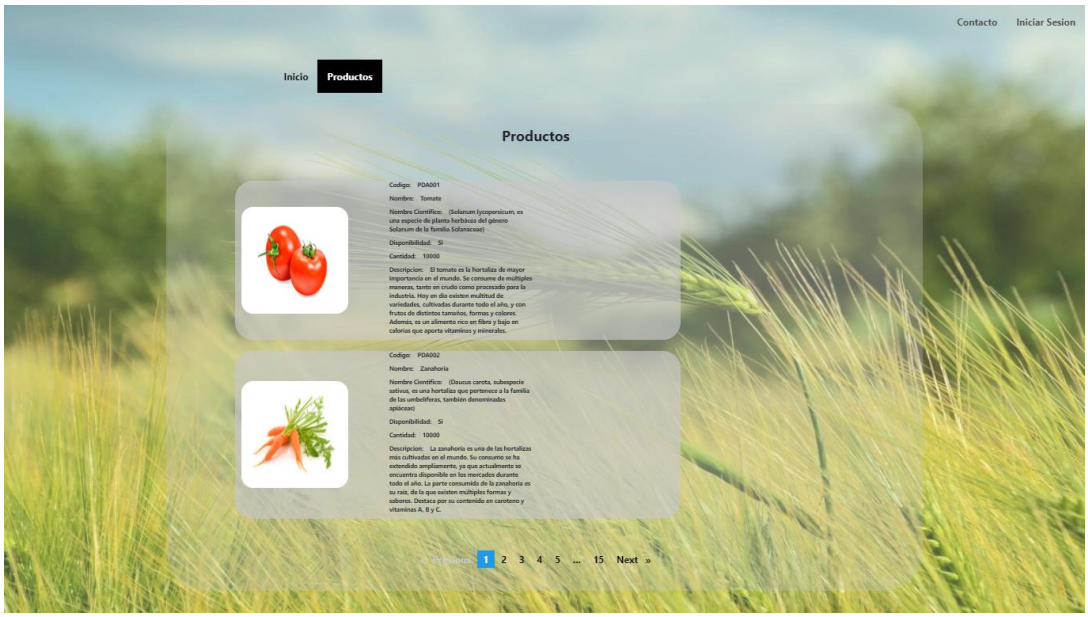


Ilustración 15: Diseño del Sprint 3.

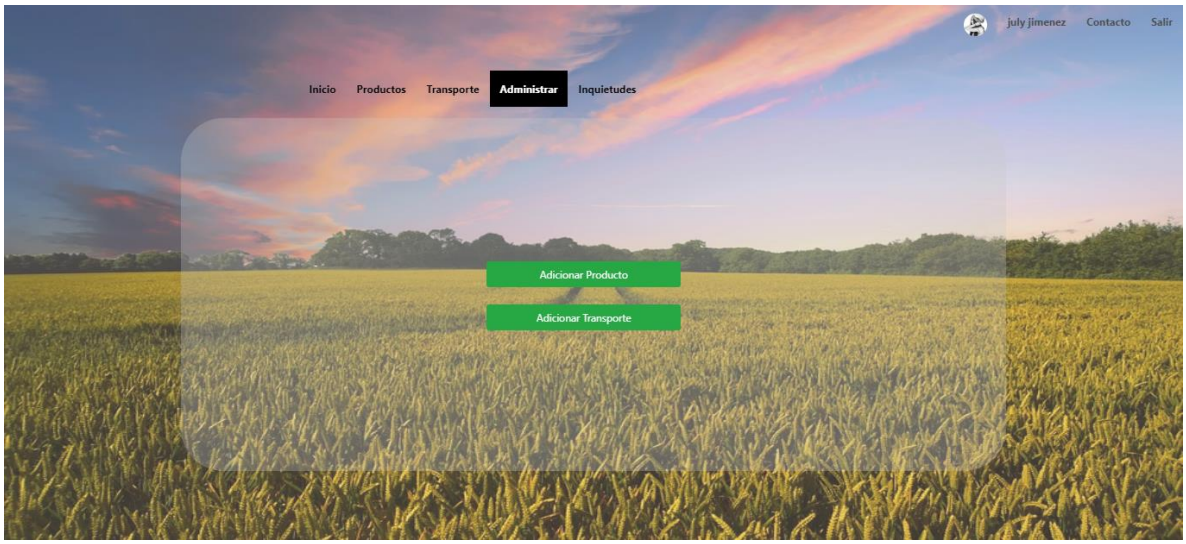


Ilustración 16: Diseño del Sprint 3.

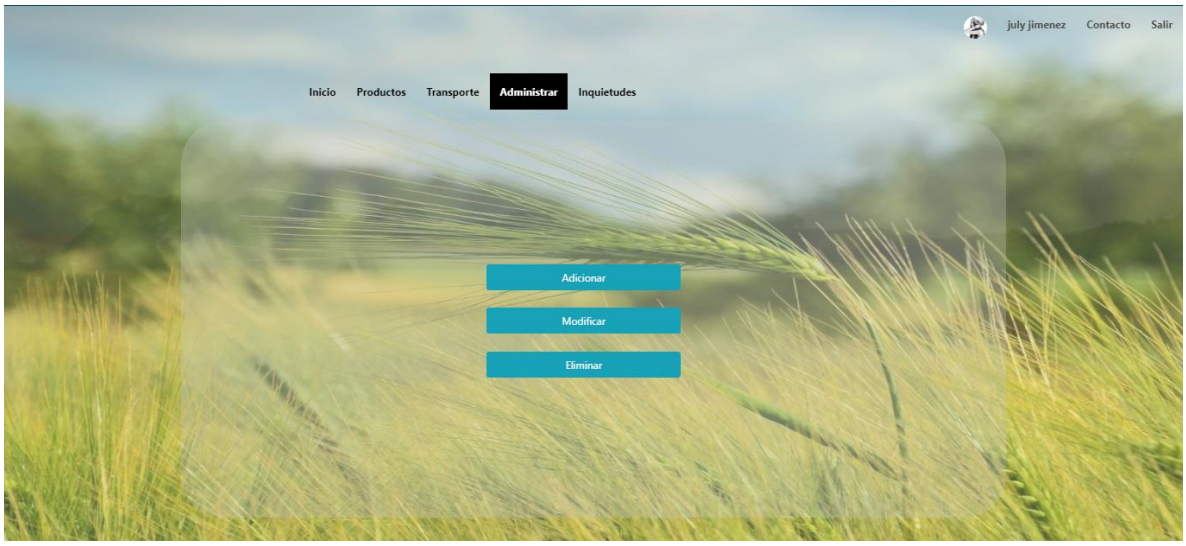


Ilustración 17: Diseño del Sprint 3.

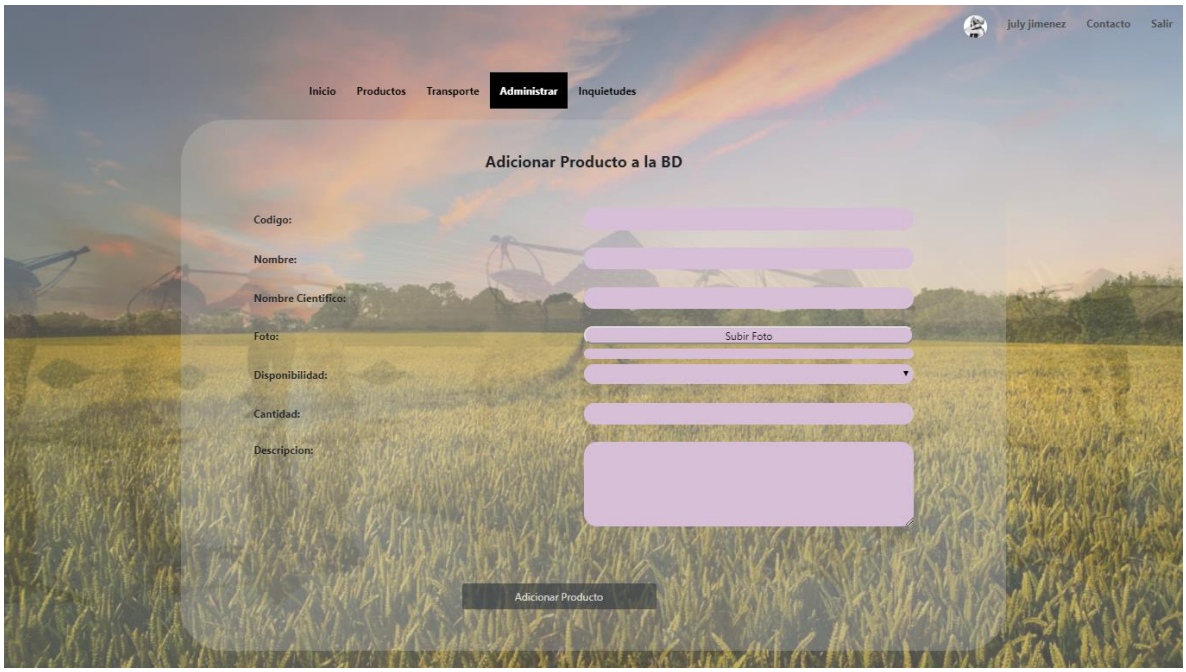


Ilustración 18: Diseño del Sprint 3.

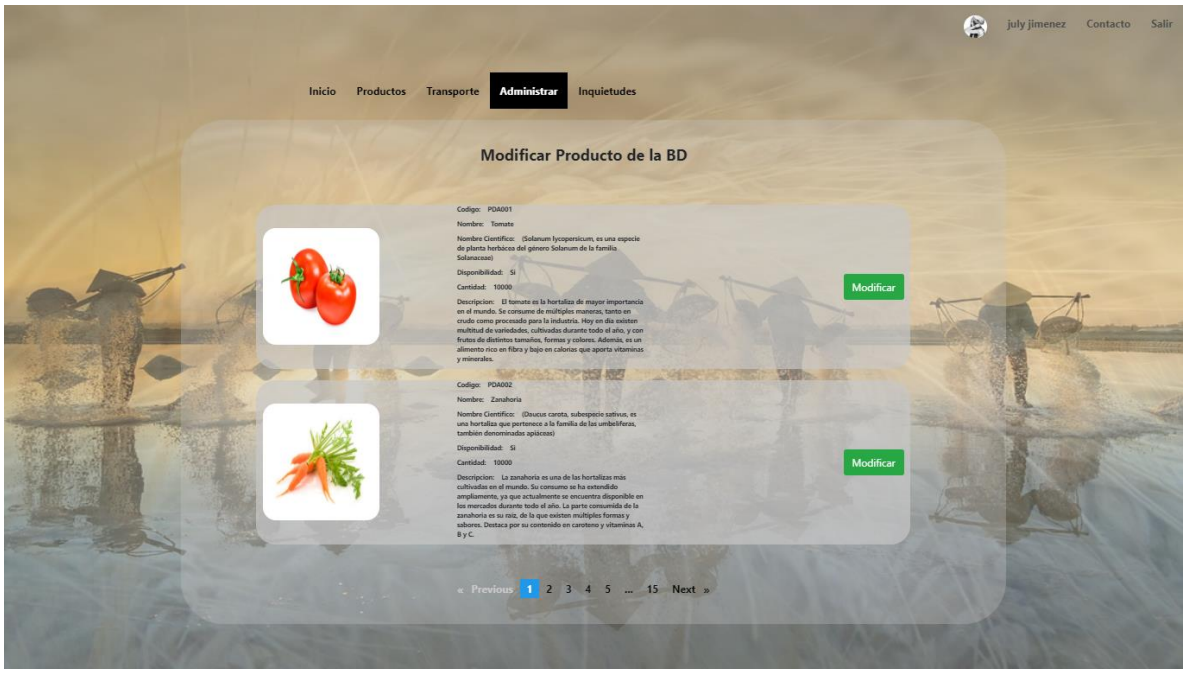


Ilustración 19: Diseño del Sprint 3.

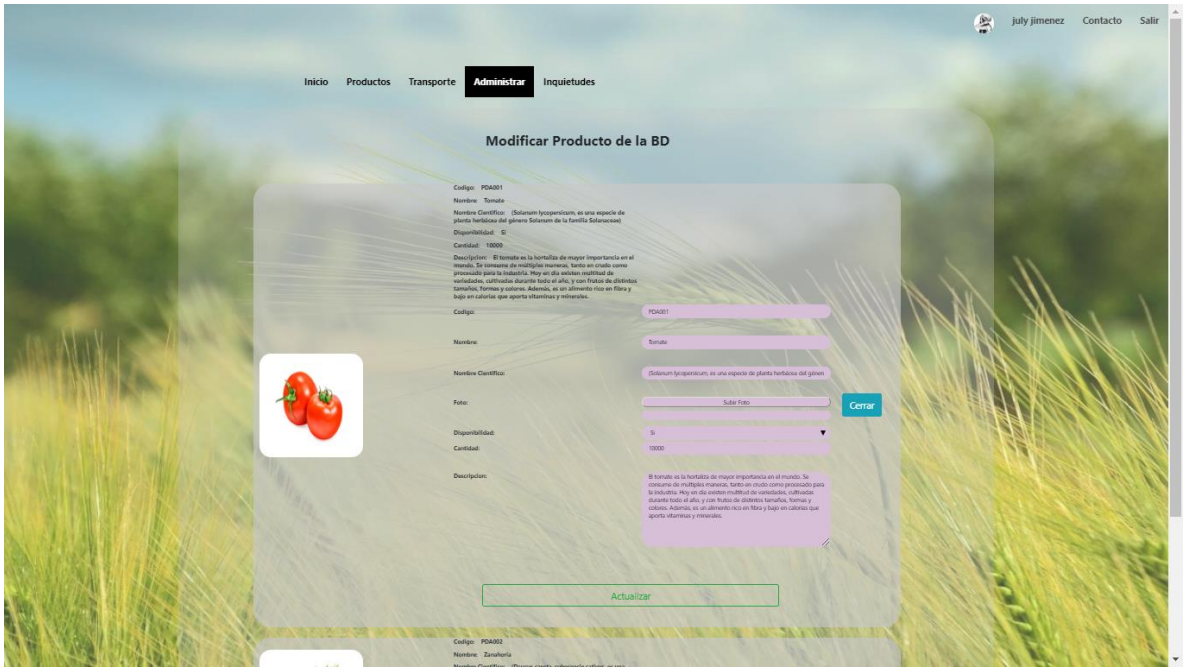


Ilustración 20: Diseño del Sprint 3.

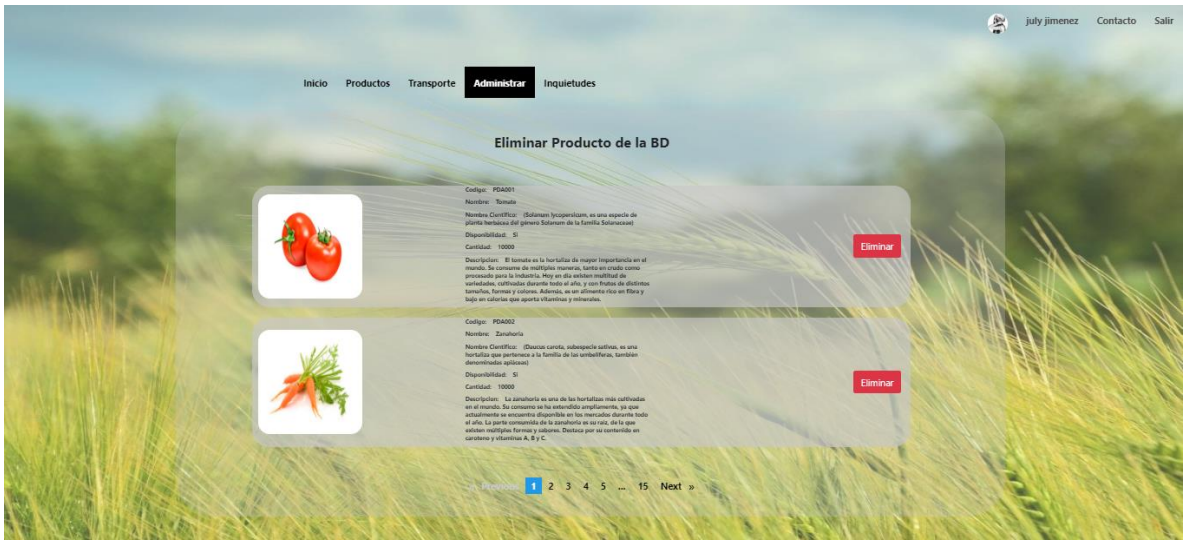


Ilustración 21: Diseño del Sprint 3.

3.3.4 Proceso de desarrollo de software del Sprint 3

```
TS producto.interface.ts ×
PJBA ▸ src ▸ app ▸ interface ▸ TS producto.interface.ts ▸ ...
1   /*Aqui definimos la estructura que tendran los productos con sus atributos*/
2   export interface productoI{
3       id?: string;
4       Codigo?: string;
5       NombreCientifico?: string;
6       Nombre?: string;
7       Foto?: string;
8       Disponibilidad?: string;
9       Descripcion?: string;
10      Cantidad?: number;
11  }
```

Ilustración 22: Desarrollo del Sprint 3.

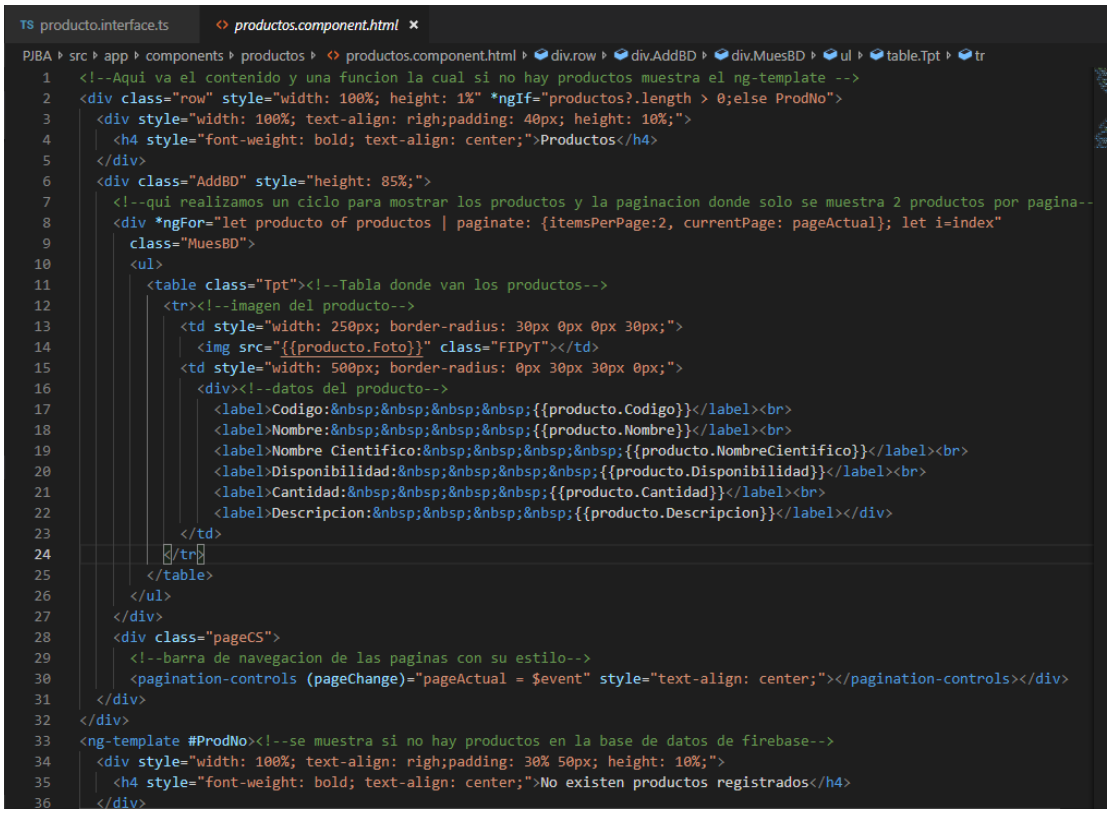


Ilustración 23: Desarrollo del Sprint 3.

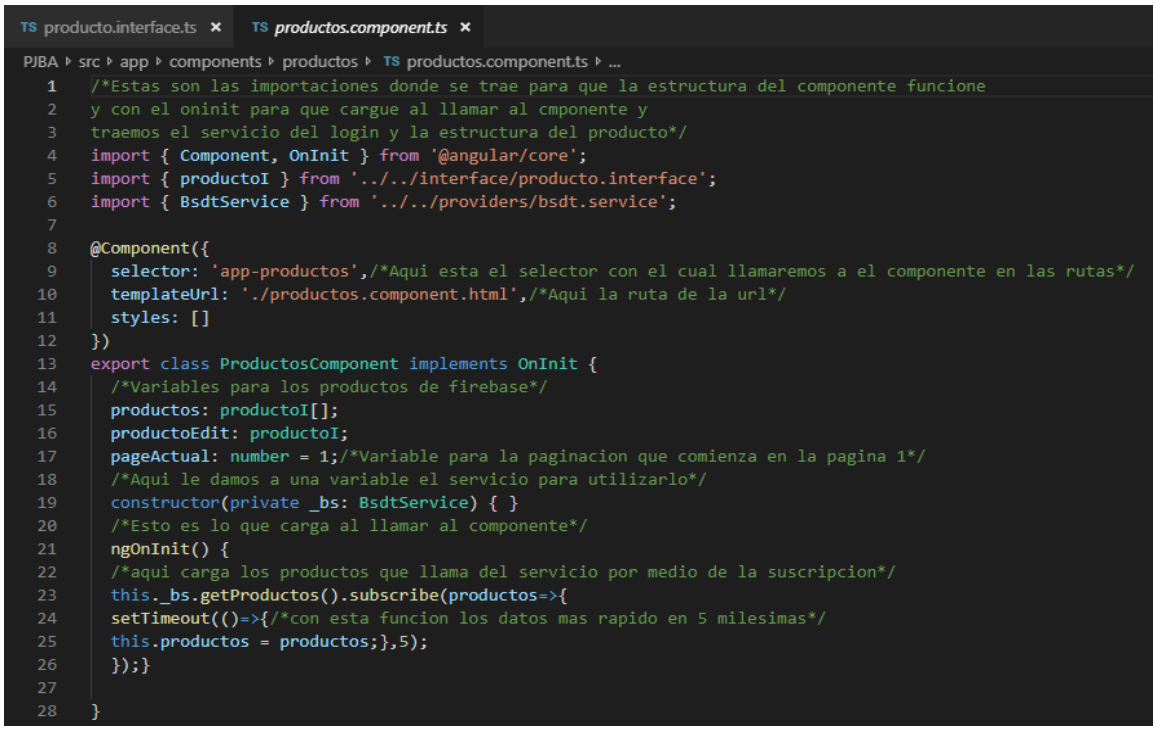
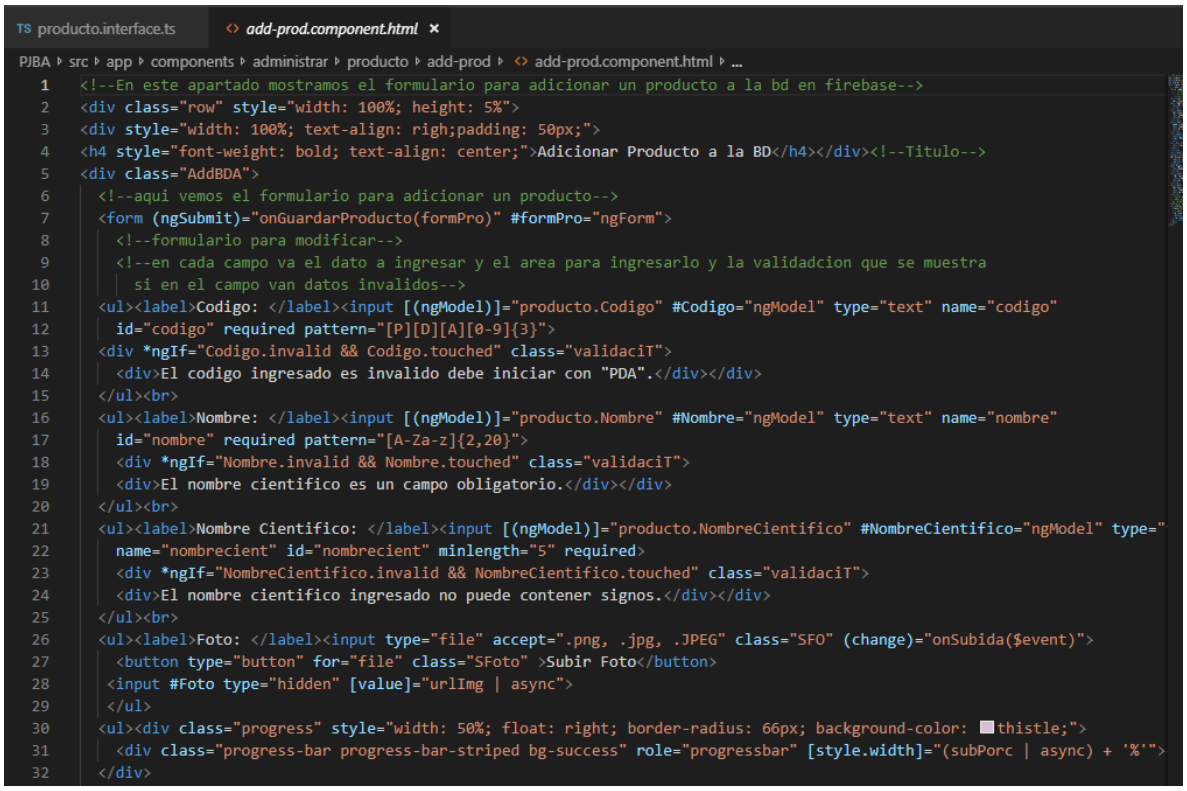
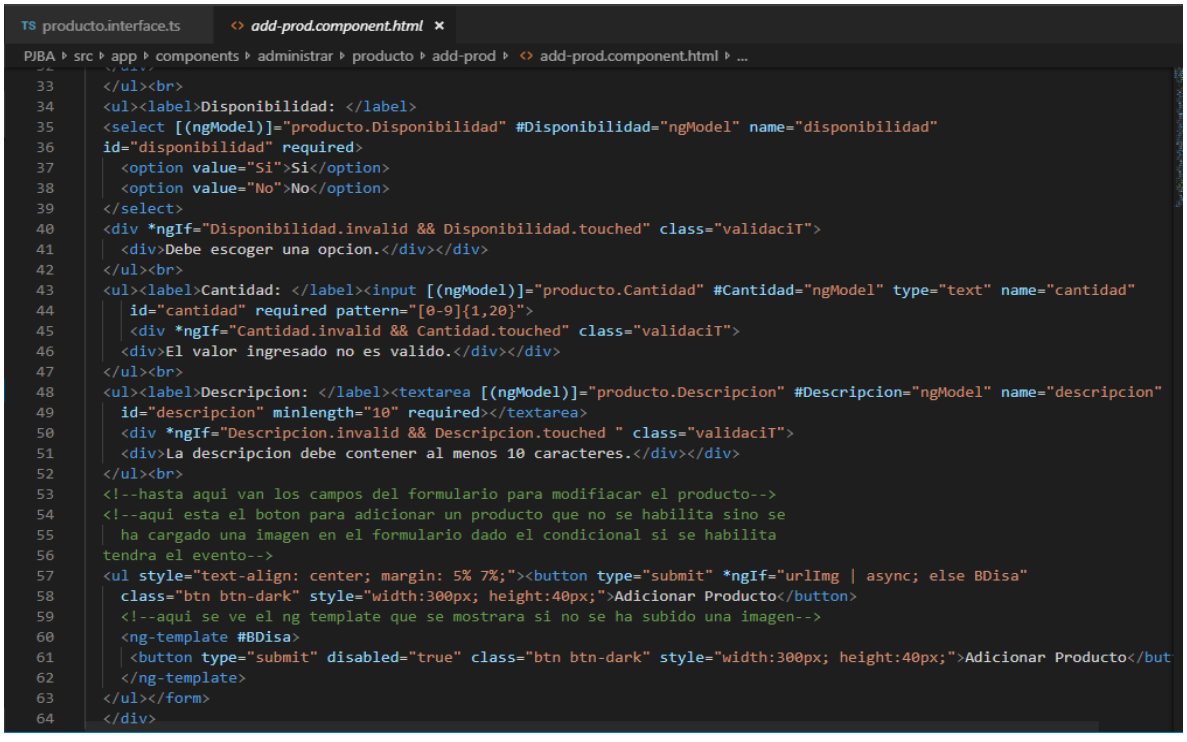


Ilustración 24: Desarrollo del Sprint 3.



```
<!--En este apartado mostramos el formulario para adicionar un producto a la bd en firebase-->
<div class="row" style="width: 100%; height: 5%">
<div style="width: 100%; text-align: righ;padding: 50px;">
<h4 style="font-weight: bold; text-align: center;">Adicionar Producto a la BD</h4></div><!--Titulo-->
<div class="AddBDA">
  <!--aqui vemos el formulario para adicionar un producto-->
  <form (ngSubmit)="onGuardarProducto(formPro)" #formPro="ngForm">
    <!--formulario para modificar-->
    <!--en cada campo va el dato a ingresar y el area para ingresarlo y la validadcion que se muestra
      si en el campo van datos invalidos-->
  <ul><label>Codigo: </label><input [(ngModel)]="producto.Codigo" #Codigo="ngModel" type="text" name="codigo"
    id="codigo" required pattern="[P][D][A][0-9]{3}">
  <div *ngIf="Codigo.invalid && Codigo.touched" class="validaciT">
    <div>El codigo ingresado es invalido debe iniciar con "PDA".</div></div>
  </ul><br>
  <ul><label>Nombre: </label><input [(ngModel)]="producto.Nombre" #Nombre="ngModel" type="text" name="nombre"
    id="nombre" required pattern="[A-Za-z]{2,20}">
    <div *ngIf="Nombre.invalid && Nombre.touched" class="validaciT">
    <div>El nombre cientifico es un campo obligatorio.</div></div>
  </ul><br>
  <ul><label>Nombre Cientifico: </label><input [(ngModel)]="producto.NombreCientifico" #NombreCientifico="ngModel" type="
    name="nombrecient" id="nombrecient" minlength="5" required>
    <div *ngIf="NombreCientifico.invalid && NombreCientifico.touched" class="validaciT">
    <div>El nombre cientifico ingresado no puede contener signos.</div></div>
  </ul><br>
  <ul><label>Foto: </label><input type="file" accept=".png, .jpg, .JPEG" class="SFO" (change)="onSubida($event)">
    <button type="button" for="file" class="SFoto" >Subir Foto</button>
   <input #Foto type="hidden" [value]="urlImg | async">
   </ul>
  <ul><div class="progress" style="width: 50%; float: right; border-radius: 66px; background-color: thistle;">
    <div class="progress-bar progress-bar-striped bg-success" role="progressbar" [style.width]="(subPorc | async) + '%'">
  </div>
```

Ilustración 25: Desarrollo del Sprint 3.



```
  </ul><br>
  <ul><label>Disponibilidad: </label>
  <select [(ngModel)]="producto.Disponibilidad" #Disponibilidad="ngModel" name="disponibilidad"
  id="disponibilidad" required>
    <option value="Si">Si</option>
    <option value="No">No</option>
  </select>
  <div *ngIf="Disponibilidad.invalid && Disponibilidad.touched" class="validaciT">
    <div>Debe escoger una opcion.</div></div>
  </ul><br>
  <ul><label>Cantidad: </label><input [(ngModel)]="producto.Cantidad" #Cantidad="ngModel" type="text" name="cantidad"
    id="cantidad" required pattern="[0-9]{1,20}">
    <div *ngIf="Cantidad.invalid && Cantidad.touched" class="validaciT">
    <div>El valor ingresado no es valido.</div></div>
  </ul><br>
  <ul><label>Descripcion: </label><textarea [(ngModel)]="producto.Descripcion" #Descripcion="ngModel" name="descripcion"
    id="descripcion" minlength="10" required></textarea>
    <div *ngIf="Descripcion.invalid && Descripcion.touched " class="validaciT">
    <div>La descripcion debe contener al menos 10 caracteres.</div></div>
  </ul><br>
  <!--hasta aqui van los campos del formulario para modifiacar el producto-->
  <!--aqui esta el boton para adicionar un producto que no se habilita sino se
    ha cargado una imagen en el formulario dado el condicional si se habilita
  tendra el evento-->
  <ul style="text-align: center; margin: 5% 7%;"><button type="submit" *ngIf="urlImg | async; else BDisa"
    class="btn btn-dark" style="width:300px; height:40px;">Adicionar Producto</button>
    <!--aqui se ve el ng template que se mostrara si no se ha subido una imagen-->
    <ng-template #BDisa>
      <button type="submit" disabled="true" class="btn btn-dark" style="width:300px; height:40px;">Adicionar Producto</but
    </ng-template>
  </ul></form>
  </div>
```

Ilustración 26: Desarrollo del Sprint 3.

producto.interface.ts | add-prod.component.ts

PJBA ▸ src ▸ app ▸ components ▸ administrar ▸ producto ▸ add-prod ▸ add-prod.component.ts ▸ ...

```
45      this.subPorc = task.percentageChanges();/*aqui vamos actualizando el valor de subporc para ir
46      mostrando la barra de carga*/
47      task.snapshotChanges().pipe(finalize(() => this.urlImg = ref.getDownloadURL())).subscribe();}/*aqui tomamos la url
48      y la asignamos a un observable*/
49    /*funcion para guardar un producto el cual trae el formulario del html*/
50    onGuardarProducto(formPro: NgForm){
51      if (formPro.valid) {/*si el formulario es valido adiciona el objeto a la coleccion en firebase*/
52        this.producto.Foto = this.InpImgProd.nativeElement.value;/*aqui le damos la url como string a el atributo foto del
53        producto para subirlo a la bd */
54        this._bs.addProducto(this.producto);/*aqui adicinamos el producto a la base de datos en firebase*/
55      }else{
56      console.log('El formulario no se lleno de manera correcta');}/*Si el formulario no es valido muestra este
57        mensaje en la consola*/
58    }
59
60  }
61
```

Ilustración 27: Desarrollo del Sprint 3.

producto.interface.ts | eli-prod.component.html

PJBA ▸ src ▸ app ▸ components ▸ administrar ▸ producto ▸ eli-prod ▸ eli-prod.component.html ▸ ...

```
1   <!--En este apartado mostramos la bd de los productos donde la podremos eliminar, si no hay nada en la
2   base de datos se mostrara el ng-template de abajo-->
3   <div class="row" style="width: 100%; height: 1%" *ngIf="productos?.length > 0;else ProdNo">
4     <div style="width: 100%; text-align: righ;padding: 40px; height: 10%;">
5       <h4 style="font-weight: bold; text-align: center;">Eliminar Producto de la BD</h4>
6     </div>
7     <div class="AddBD" style="height: 85%;">
8       <!--aqui usamos un for para mostrar todos los productos mostrando 2 por pagina-->
9       <div *ngFor="let producto of productos | paginate: {itemsPerPage:2, currentPage: pageActual}; let i=index"
10      class="MuesBD">
11        <ul><table class="Tpt"><!--en esta tabla va la foto, los datos del producto y el formulario para eliminarlo-->
12          <tr>
13            <!--Aqui va la foto del producto-->
14            <td style="width: 30%; border-radius: 30px 0px 0px 30px;">
15            <img src="{{producto.Foto}}" class="FIPyT"></td>
16            <!--aqui van los datos del producto de la bd-->
17            <td style="width: 55%;">
18              <div>
19              <label>Codigo:    {{producto.Codigo}}</label><br>
20              <label>Nombre:    {{producto.Nombre}}</label><br>
21              <label>Nombre Cientifico:    {{producto.NombreCientifico}}</label><br>
22              <label>Disponibilidad:    {{producto.Disponibilidad}}</label><br>
23              <label>Cantidad:    {{producto.Cantidad}}</label><br>
24              <label>Descripcion:    {{producto.Descripcion}}</label></div>
25            </td><!--hasta qui se ven los datos de un producto-->
26              <!--en este apartado se muestra el boton para eliminar un producto con su estilo-->
27            <td style="width: 10%; border-radius: 0px 30px 30px 0px;">
28              <ul style="text-align: center; margin: 5% 7%;"><button (click)="dltProducto($event, producto)"
29                    class="btn btn-danger" style="height:40px;">Eliminar</button></ul>
30            </td></tr></table></ul>
31      </div>
32      <!--aqui esta la paginacion de los productos donde la pagina actual es la primera-->
33      <div class="pageCS">
34        <pagination-controls (pageChange)="pageActual = $event" style="text-align: center;"></pagination-controls></div>
35    </div>
36  </div>
37  <!--Se muestra si no hay ningun producto en la bd donde se llama con la validacion de arriba-->
38  <ng-template #ProdNo>
39    <div style="width: 100%; text-align: righ;padding: 30% 50px; height: 10%;">
40    <h4 style="font-weight: bold; text-align: center;">No existen datos en la BD</h4>
41    </div>
42  </ng-template>
43
```

Ilustración 28: Desarrollo del Sprint 3.

producto.interface.ts | eli-prod.component.ts

PJBA ▸ src ▸ app ▸ components ▸ administrar ▸ producto ▸ eli-prod ▸ eli-prod.component.ts ▸ ...

```ts
/*Estas son las importaciones donde se trae para que la estructura del componente funcion y la
funcion oninit para que inicie auomaticamente al llamar el componente*/
import { Component, OnInit } from '@angular/core';
import { productoI } from '../../../../interface/producto.interface';/*importamos la estructura de el producto*/
/*importamos el servicio que utilizamos para modificar los transportes*/
import { BsdtService } from '../../../../providers/bsdt.service';
import { NgForm } from '@angular/forms';/*importamos para los formularios*/

@Component({
  selector: 'app-eli-prod',/*Aqui esta el selector con el cual llamaremos a el componente en las rutas*/
  templateUrl: './eli-prod.component.html',/*Aqui la ruta de la url*/
  styles: []
})
export class EliProdComponent implements OnInit {
  /*creamos variables para traer la bd de los productos de firebase*/
  productos: productoI[];
  productoEdit: productoI;
  pageActual: number = 1;/*variable que indica la pagina inicial de la paginacion*/
  /*Importamos el servicio para modificar la bd de los productos*/
  constructor(private _bs: BsdtService) {
  this.ngOnInit();
  }
  /*Esto es lo que carga al llamar al componente*/
  ngOnInit() {
  /*aqui traemos la coleccion de los productos para visualizarlos*/
  this._bs.getProductos().subscribe(productos=>{
  this.productos = productos;
  });
  }
  /*aqui esta la funcion para eliminar un producto de la bd*/
  dltProducto(event, producto: productoI){
  this._bs.elimProducto(producto);/*aqui llamamos la funcion de eliminar producto del servicio*/
  }

}
```

Ilustración 29: Desarrollo del Sprint 3.

producto.interface.ts | mod-prod.component.html

PJBA ▸ src ▸ app ▸ components ▸ administrar ▸ producto ▸ mod-prod ▸ mod-prod.component.html ▸ ...

```html
<!--En este apartado mostramos la bd de los productos donde la podremos modificar, si no hay nada en la
base de datos se mostrara el ng-template de abajo-->
<div class="row" style="width: 100%; height: 1%" *ngIf="productos?.length > 0;else ProdNo">
  <div style="width: 100%; text-align: righ;padding: 40px; height: 10%;">
    <h4 style="font-weight: bold; text-align: center;">Modificar Producto de la BD</h4><!--Titulo-->
  </div>
  <div class="AddBD" style="height: 85%;">
    <!--aqui usamos un for para mostrar todos los productos mostrando 2 por pagina-->
    <div *ngFor="let producto of productos | paginate: {itemsPerPage:2, currentPage: pageActual}; let i=index"
      class="MuesBD">
      <ul><table class="Tpt"><!--en esta tabla va la foto, los datos del producto y el formulario para modificarlo-->
        <tr>
          <!--Aqui va la foto del  Defines the radii of the outer border edge.
          <td style="width: 30%; border-radius: 30px 0px 0px 30px;">
            <img src="{{producto.Foto}}" class="FIPyT"></td>
          <!--aqui van los datos del producto de la bd-->
          <td style="width: 60%;">
            <div>
              <!--datos con su tipo-->
            <label>Codigo:    {{producto.Codigo}}</label><br>
            <label>Nombre:    {{producto.Nombre}}</label><br>
            <label>Nombre Cientifico:    {{producto.NombreCientifico}}</label><br>
            <label>Disponibilidad:    {{producto.Disponibilidad}}</label><br>
            <label>Cantidad:    {{producto.Cantidad}}</label><br>
            <label>Descripcion:    {{producto.Descripcion}}</label></div>
            <!--aqui vemos el formulario para modificar si el boton de modificar fue activado-->
            <div *ngIf="editState && productoEdit.id == producto.id">
            <form (ngSubmit)="actuProducto(producto)"><!--formulario para modificar-->
            <!--en cada campo va el dato a modificar el area para modificarlo y la validacion que se muestra
            si en el campo van datos invalidos-->
            <label>Codigo: </label><input [(ngModel)]="producto.Codigo" #Codigo="ngModel" type="text" name="codigo"
              id="codigo" required pattern="[P][D][A][0-9]{3}">
            <div *ngIf="Codigo.invalid && Codigo.touched" class="validaciT">
              <div>El codigo ingresado es invalido debe iniciar con "PDA".</div>
            </div><br><br><br>
            <label>Nombre: </label><input [(ngModel)]="producto.Nombre" #Nombre="ngModel" type="text" name="nombre"
```

Ilustración 30: Desarrollo del Sprint 3.

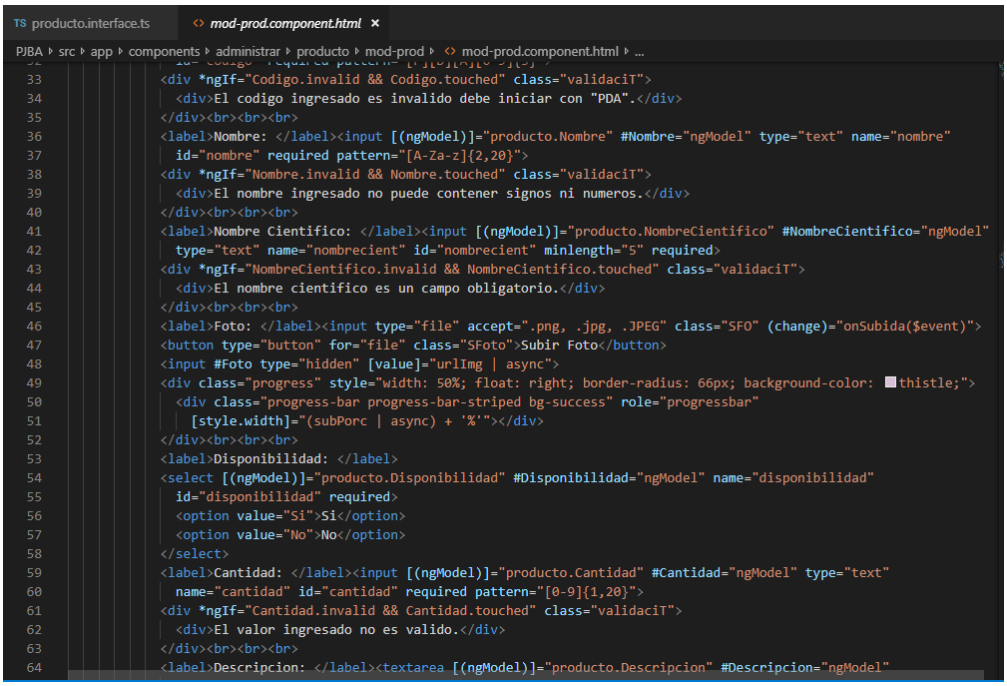


Ilustración 31: Desarrollo del Sprint 3.

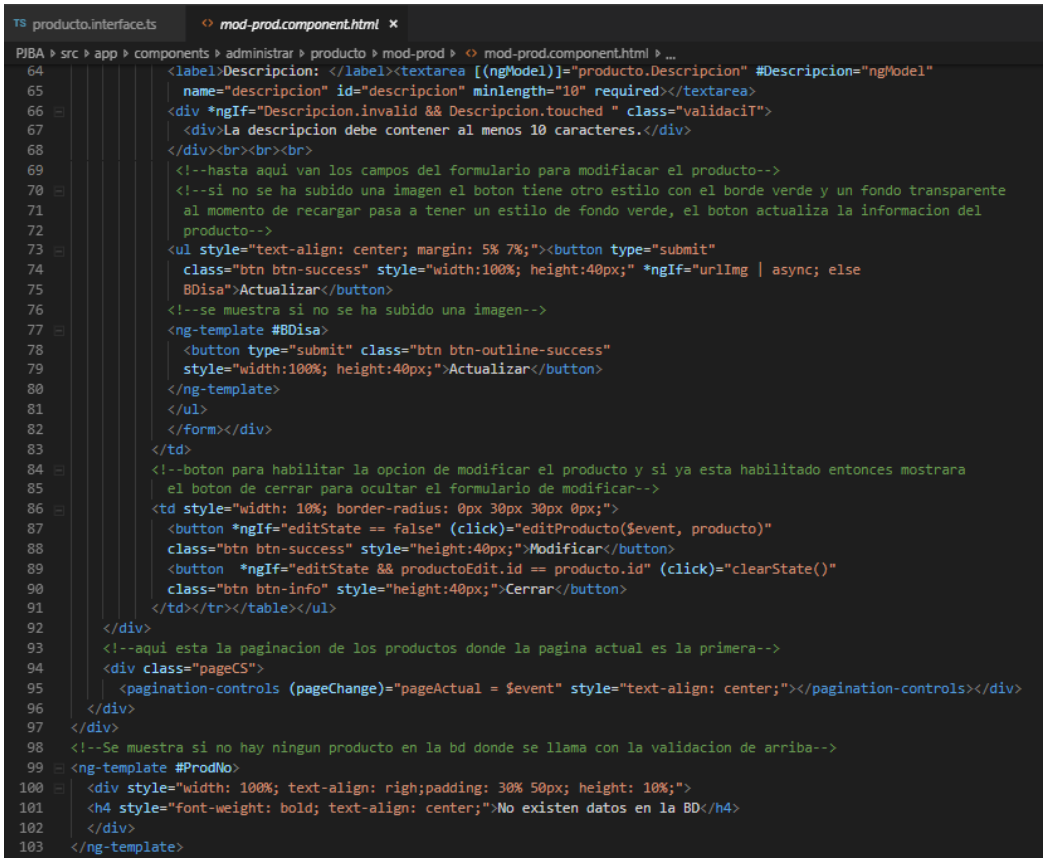


Ilustración 32: Desarrollo del Sprint 3.

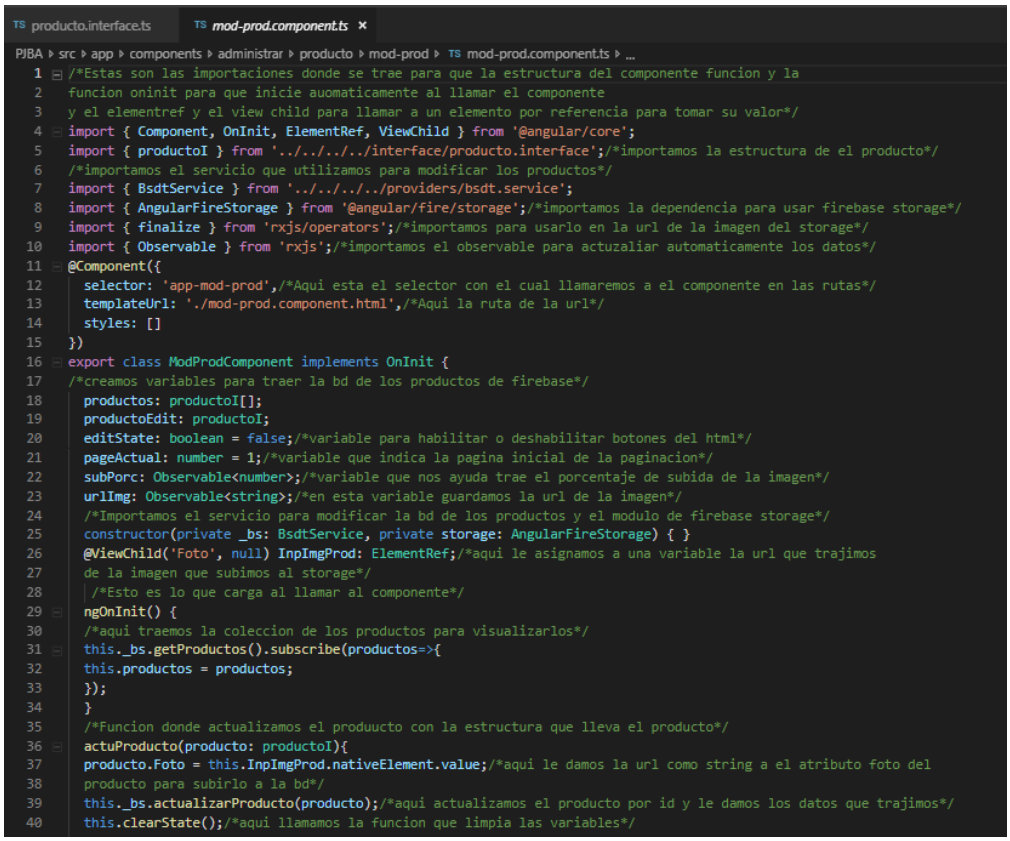


Ilustración 33: Desarrollo del Sprint 3.

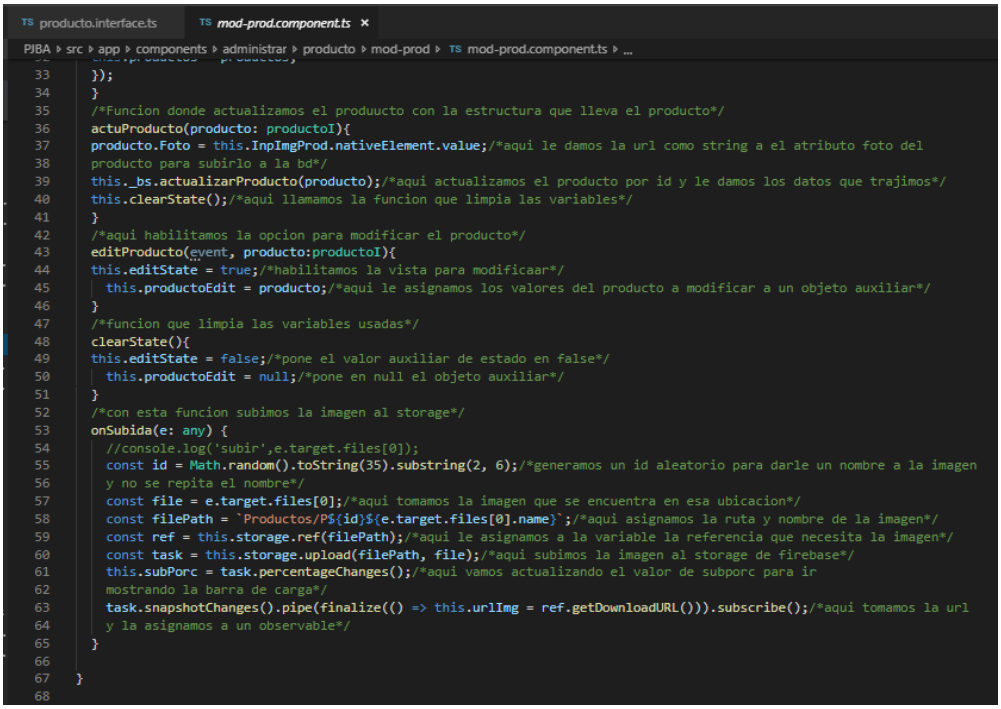


Ilustración 34: Desarrollo del Sprint 3.

producto.interface.ts | producto.component.html

PJBA ▸ src ▸ app ▸ components ▸ administrar ▸ producto ▸ producto.component.html ▸ ...

```
<!--Aqui tenemos los botones para adicionar, modificar y eliminar un producto y la validacion de
  si trae otro componente este se oculte-->
<div class="row" style="width: 100%; height: 100%" *ngIf="outlet.isActivated == false">
  <div class="col-md-12 text-center" style="padding: 200px; line-height: 50px">
    <ul [routerLink]="['Adicionar']"><button type="button" class="btn btn-info"
        style="width:300px; height:40px;">Adicionar</button></ul>
    <ul [routerLink]="['Modificar']"><button type="button" class="btn btn-info"
        style="width:300px; height:40px;">Modificar</button></ul>
    <ul [routerLink]="['Eliminar']"><button type="button" class="btn btn-info"
        style="width:300px; height:40px;">Eliminar</button></ul>
  </div>
</div>
<!--Aqui traemos el componente al cual llamamos sobreescribiendo el contenido que muestra la pagina-->
<router-outlet #outlet="outlet"></router-outlet>
```

Ilustración 35: Desarrollo del Sprint 3.

producto.interface.ts | producto.component.ts

PJBA ▸ src ▸ app ▸ components ▸ administrar ▸ producto ▸ producto.component.ts ▸ ...

```
/*Estas son las importaciones donde se trae para que la estructura del componente funcion*/
import { Component} from '@angular/core';

@Component({
  selector: 'app-producto',/*Aqui esta el selector con el cual llamaremos a el componente en las rutas*/
  templateUrl: './producto.component.html',/*Aqui la ruta de la url*/
  styles: []
})
export class ProductoComponent{

  constructor() { }

}
```

Ilustración 36: Desarrollo del Sprint 3.

3.3.5 Diagrama de contexto del Sprint 3

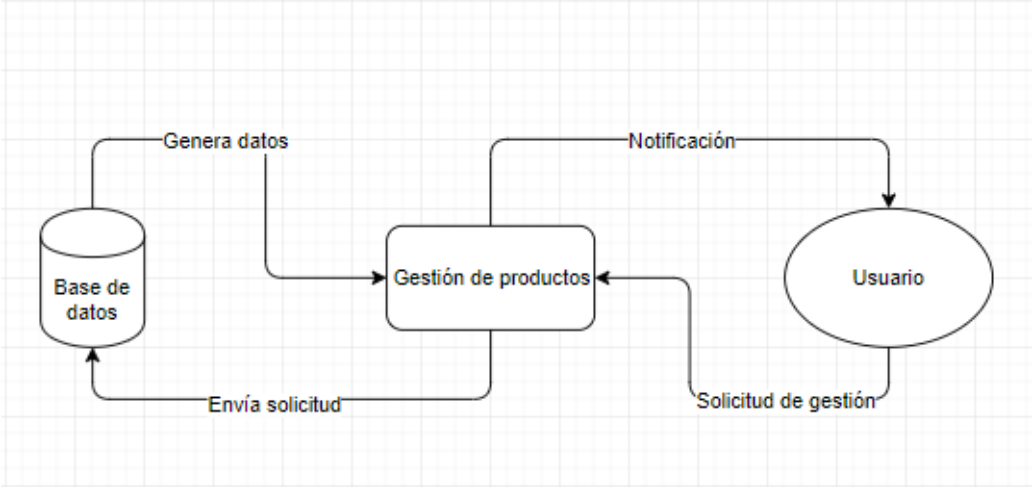


Ilustración 37: Diagrama del Sprint 3.

3.3.6 Prueba del Sprint 3

Sprint 3	Gestión de productos	
Descripción de la prueba	Se realiza un CRUD de la BD de Productos almacenada en la nube en FireBase.	
Herramientas	Opera y FireBase	
Numero	Prueba	Estado
1	El catálogo de los productos se muestra sin problemas con los productos que ya se han adicionado con antelación.	OK
2	La adición de cada producto esta validada y correctamente ejecutada para que no ocasione futuros problemas.	OK
3	La modificación se realiza con total confianza manteniendo las mismas validaciones de la adición para que no se presenten problemas al modificar.	OK
4	La eliminación de cada producto se realiza con seguridad sin dejar rastro del producto.	OK

Tabla 17: Prueba del Sprint 3.

3.4 SPRINT 4 GESTION DE TRANSPORTE

Esta fase de la metodología Scrum, el Sprint se basa en el manejo de los datos de los transportes, permite crear, modificar y eliminar los transportes que se encuentran en la base de datos de la plataforma.

3.4.1 Definición del Sprint 4

SPRINT 4	NOMBRE	Gestión de transporte
OBJETIVO	Permitir al administrador crear, modificar y eliminar los transportes de la base de datos de la plataforma.	
DETALLES		
Se implementara una interfaz que permita al administrador crear, modificar y eliminar los transportes de la base de datos de la plataforma.		

CRITERIOS DE ACEPTACION
• Se debe validar el inicio de sesión que verifique el tipo de usuario. • El administrador debe poder crear, modificar y eliminar los transportes. • El diseño debe de adaptarse a cualquier dispositivo.

Tabla 18: Definición del Sprint 4.

3.4.2 Tareas del Sprint 4

Producto Backlog	Descripción de la tarea	Responsable	Estado	Duración (Días)
1	Se valida la información ingresada en los formularios de crear, modificar y eliminar transporte.	Brain Tribaldos, July Jimenez.	Terminado	1
2	Validar la información ingresada y con las opciones de Google, Twitter o Facebook.	Brain Tribaldos, July Jimenez.	Terminado	1
7	Permite al administrador crear el transporte.	Brain Tribaldos, July Jimenez.	Terminado	1
8	Permite al administrador modificar el transporte.	Brain Tribaldos, July Jimenez.	Terminado	1
9	Permite al administrador eliminar el transporte.	Brain Tribaldos, July Jimenez.	Terminado	1
10	Se creó un diseño de interfaz que permitiera al usuario acceder a la plataforma.	Brain Tribaldos, July Jimenez.	Terminado	1

Tabla 19: Tareas del Sprint 4.

3.4.3 Diseño del Sprint 4

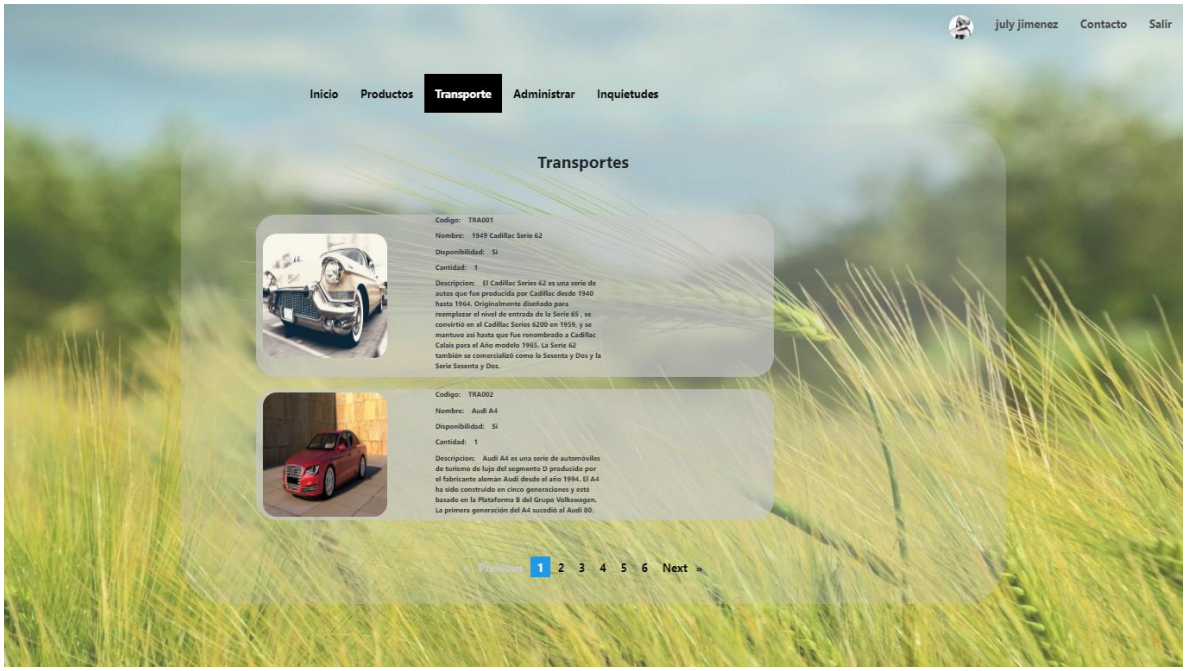


Ilustración 38: Diseño del Sprint 4.

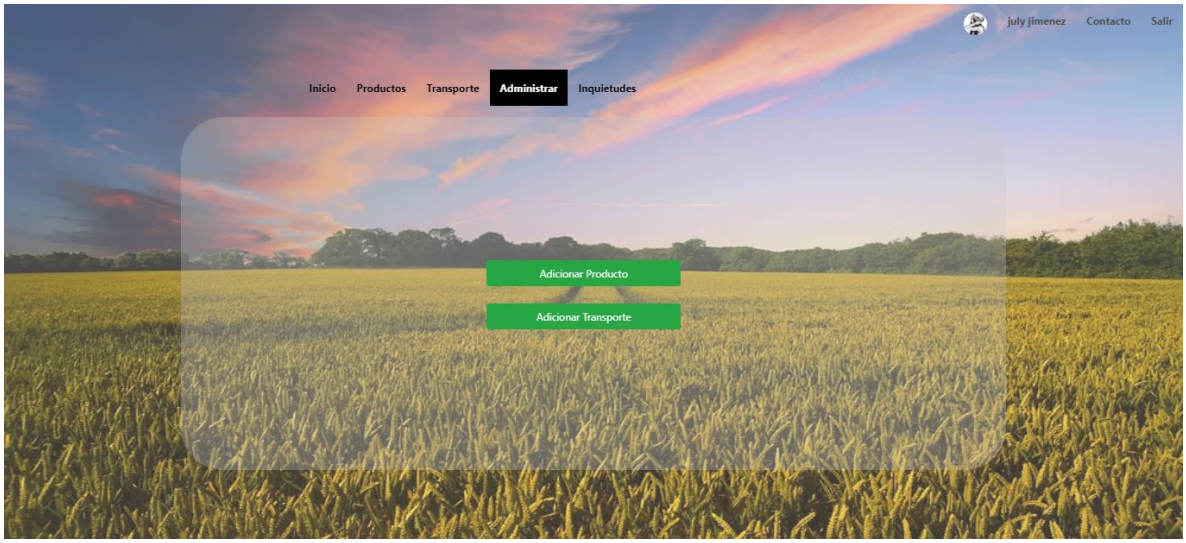


Ilustración 39: Diseño del Sprint 4.

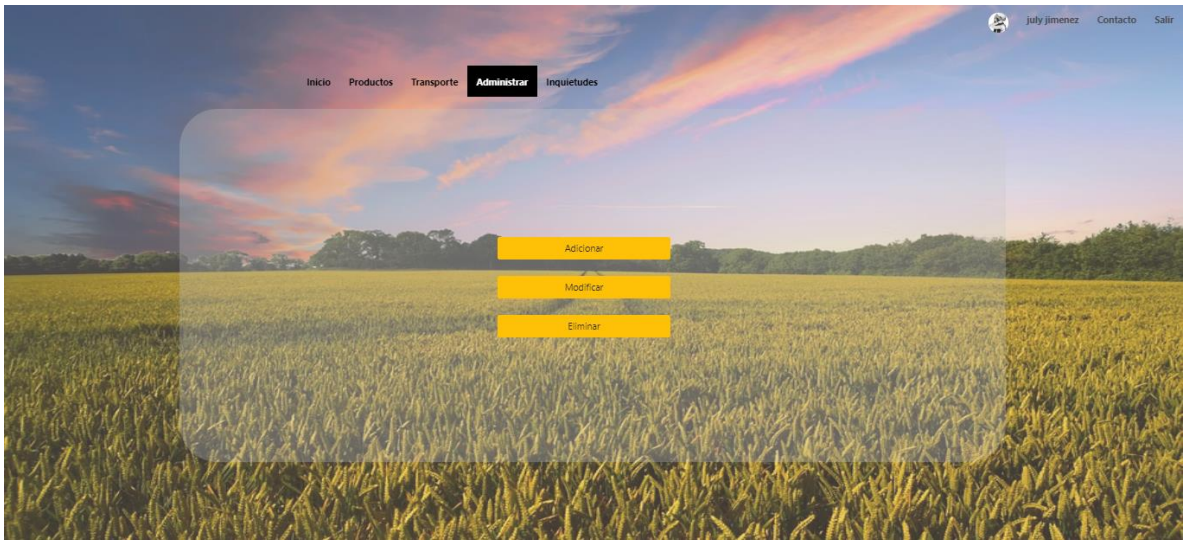


Ilustración 40: Diseño del Sprint 4.



Ilustración 41: Diseño del Sprint 4.

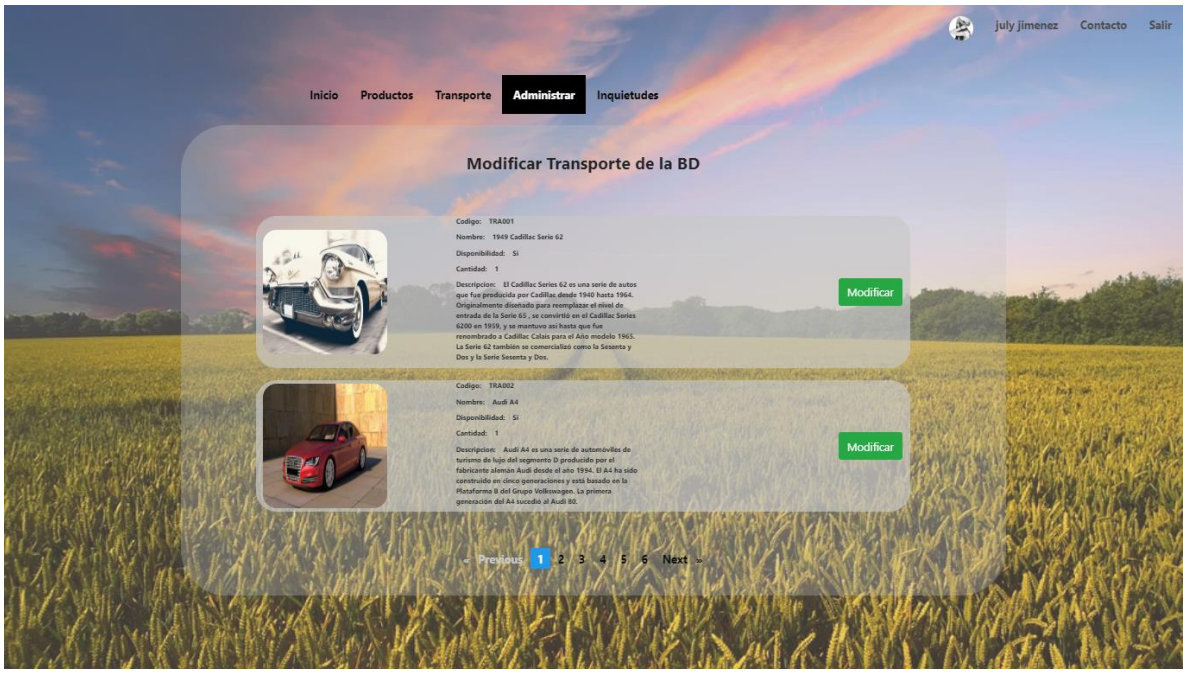


Ilustración 42: Diseño del Sprint 4.

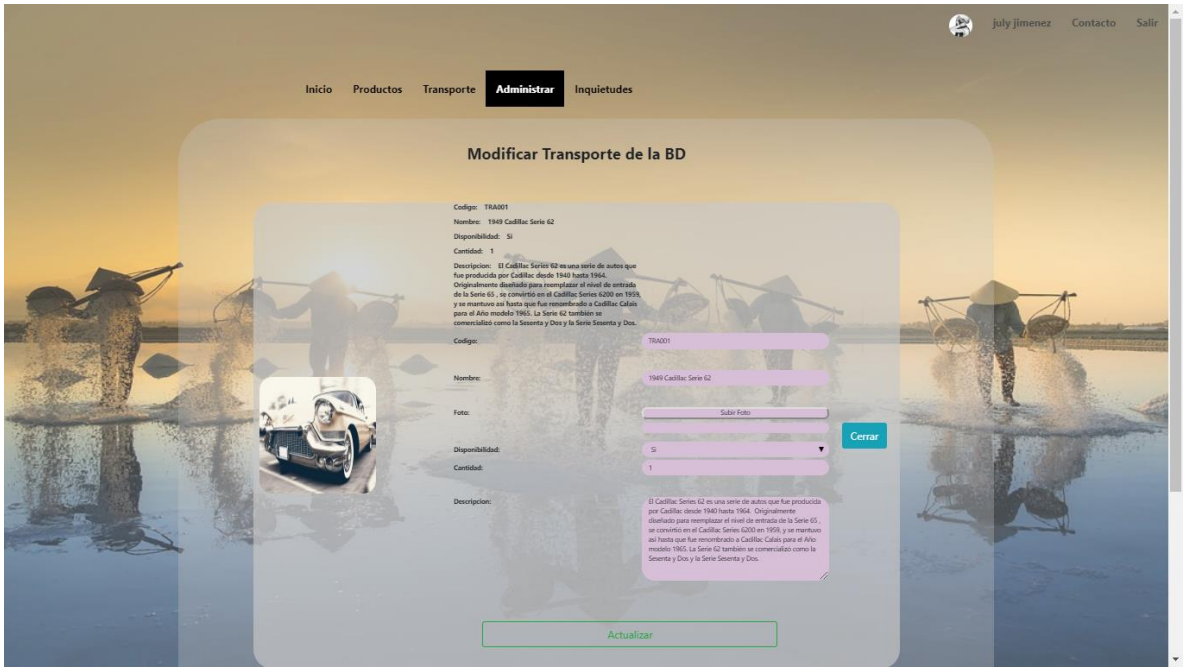


Ilustración 43: Diseño del Sprint 4.

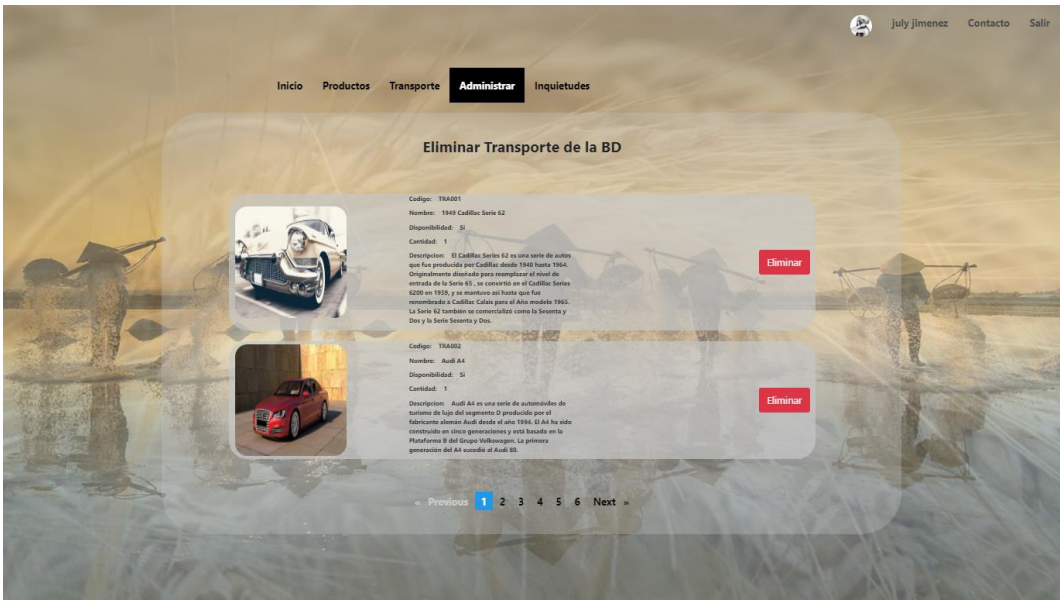


Ilustración 44: Diseño del Sprint 4.

3.4.4 Proceso de desarrollo de software del Sprint 4

producto.interface.ts | transportea.component.ts | transporte.component.html

PJBA › src › app › components › transporte › transporte.component.html › ...

```
<!--Aqui va el contenido y una funcion la cual si no hay transportes muestra el ng-template -->
<div class="row" style="width: 100%; height: 1%" *ngIf="transportes?.length > 0;else TransNo">
  <div style="width: 100%; text-align: righ;padding: 40px; height: 10%;">
    <h4 style="font-weight: bold; text-align: center;">Transportes</h4><!--Titulo-->
  </div>
  <div class="AddBD" style="height: 85%;">
    <!--qui realizamos un ciclo para mostrar los transportes y la paginacion donde solo se muestra 2 transportes por pagina-->
    <div *ngFor="let transporte of transportes | paginate: {itemsPerPage:2, currentPage: pageActual}; let i=index"
      class="MuesBD">
      <ul>
        <table class="Tpt"><!--Tabla donde van los transportes-->
          <tr><!--imagen del transporte-->
            <td style="width: 250px; border-radius: 30px 0px 0px 30px;">
              <img src="{{transporte.Foto}}" class="FIPyT"></td>
            <td style="width: 500px; border-radius: 0px 30px 30px 0px;">
              <div><!--datos del transporte-->
                <label>Codigo:    {{transporte.Codigo}}</label><br>
                <label>Nombre:    {{transporte.Nombre}}</label><br>
                <label>Disponibilidad:    {{transporte.Disponibilidad}}</label><br>
                <label>Cantidad:    {{transporte.Cantidad}}</label><br>
                <label>Descripcion:    {{transporte.Descripcion}}</label></div>
            </td>
          </tr>
        </table>
      </ul>
    </div>
    <div class="pageCS"><!--barra de navegacion de las paginas con su estilo-->
      <pagination-controls (pageChange)="pageActual = $event" style="text-align: center;"></pagination-controls></div>
  </div>
</div>
<ng-template #TransNo><!--se muestra si no hay transportes en la base de datos de firebase-->
  <div style="width: 100%; text-align: righ;padding: 30% 50px; height: 10%;">
    <h4 style="font-weight: bold; text-align: center;">No se encuentran transportes adicionados</h4>
  </div>
</ng-template>
```

Ilustración 45: Desarrollo del Sprint 4.

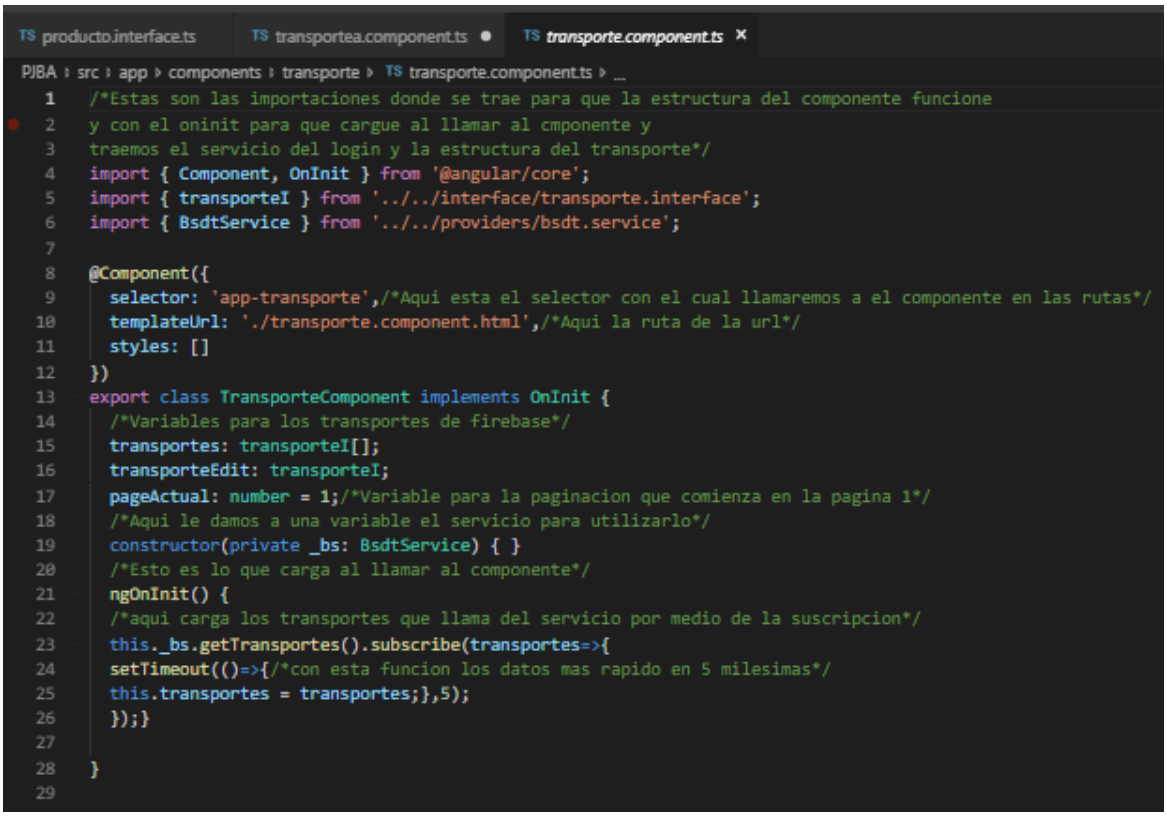


Ilustración 46: Desarrollo del Sprint 4.

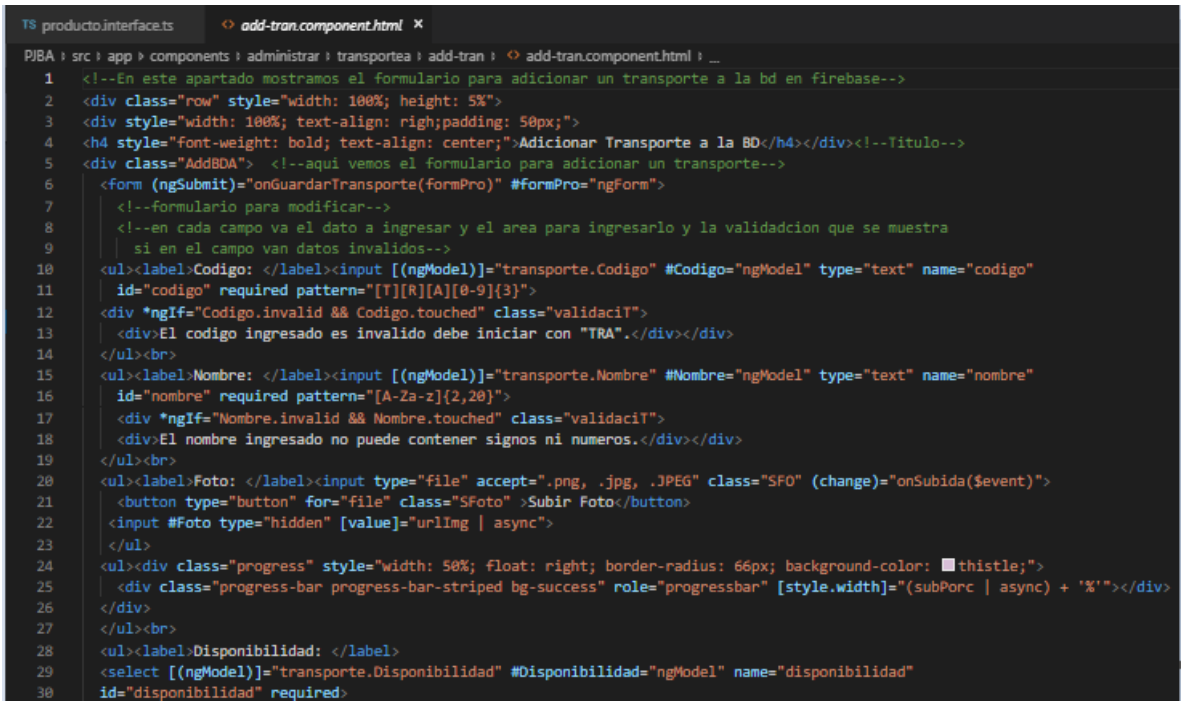


Ilustración 47: Desarrollo del Sprint 4.

```
  id="disponibilidad" required>
    <option value="Si">Si</option>
    <option value="No">No</option>
  </select>
  <div *ngIf="Disponibilidad.invalid && Disponibilidad.touched" class="validaciT">
    <div>Debe escoger una opcion.</div></div>
  </ul><br>
  <ul><label>Cantidad: </label><input [(ngModel)]="transporte.Cantidad" #Cantidad="ngModel" type="text" name="cantidad"
    id="cantidad" required pattern="[0-9]{1,20}">
    <div *ngIf="Cantidad.invalid && Cantidad.touched" class="validaciT">
    <div>El valor ingresado no es valido.</div></div>
  </ul><br>
  <ul><label>Descripcion: </label><textarea [(ngModel)]="transporte.Descripcion" #Descripcion="ngModel" name="descripcion"
    id="descripcion" minlength="10" required></textarea>
    <div *ngIf="Descripcion.invalid && Descripcion.touched " class="validaciT">
    <div>La descripcion debe contener al menos 10 caracteres.</div></div>
  </ul><br>
  <!--hasta aqui van los campos del formulario para modifiacar el transporte-->
  <!--aqui esta el boton para adicionar un transporte que no se habilita sino se
    ha cargado una imagen en el formulario dado el condicional si se habilita
  tendra el evento-->
  <ul style="text-align: center; margin: 5% 7%;"><button type="submit" *ngIf="urlImg | async; else BDisa"
    class="btn btn-dark" style="width:300px; height:40px;">Adicionar Transporte</button>
    <!--aqui se ve el ng template que se mostrara si no se ha subido una imagen-->
    <ng-template #BDisa>
      <button type="submit" disabled="true" class="btn btn-dark" style="width:300px; height:40px;">Adicionar Transporte</button>
    </ng-template>
  </ul></form>
  </div>
</div>
```

Ilustración 48: Desarrollo del Sprint 4.

producto.interface.ts | add-tran.component.ts

PJBA › src › app › components › administrar › transportea › add-tran › add-tran.component.ts › ...

```
/*Estas son las importaciones donde se trae para que la estructura del componente funcion y la
funcion oninit para que inicie auomaticamente al llamar el componente
y el elementref y el view child para llamar a un elemento por referencia para tomar su valor*/
import { Component, OnInit, ElementRef, ViewChild} from '@angular/core';
import { transporteI } from '../../../../interface/transporte.interface';/*importamos la estructura de el transporte*/
/*importamos el servicio que utilizamos para modificar los transportes*/
import { BsdtService } from '../../../../providers/bsdt.service';
import { NgForm } from '@angular/forms'; /*importamos para los formularios*/
import { AngularFireStorage } from '@angular/fire/storage';/*importamos la dependencia para usar firebase storage*/
import { finalize } from 'rxjs/operators';/*importamos para usarlo en la url de la imagen del storage*/
import { Observable } from 'rxjs';/*importamos el observable para actuzaliar automaticamente los datos*/
@Component({
  selector: 'app-add-tran',/*Aqui esta el selector con el cual llamaremos a el componente en las rutas*/
  templateUrl: './add-tran.component.html',/*Aqui la ruta de la url*/
  styles: []
})
export class AddTranComponent implements OnInit{
  /*inicializamos el objeto transporte con la estructura que trajimos de la interface*/
  transporte: transporteI = {
    Codigo: '',
    Nombre: '',
    Foto: '',
    Disponibilidad: '',
    Descripcion: '',
    Cantidad: null
  };
  subPorc: Observable<number>;/*variable que nos ayuda trae el porcentaje de subida de la imagen*/
  urlImg: Observable<string>;/*en esta variable guardamos la url de la imagen*/
  /*Importamos el servicio para adicionar a la bd de los transportes y el modulo de firebase storage*/
```

Ilustración 49: Desarrollo del Sprint 4.

producto.interface.ts | add-tran.component.ts

PJBA › src › app › components › administrar › transportea › add-tran › add-tran.component.ts › ...

```
30    constructor(private _bs: BsdtService, private storage: AngularFireStorage) { }
31    @ViewChild('Foto', null) InpImgProd: ElementRef;/*aqui le asignamos a una variable la url que trajimos
32    de la imagen que subimos al storage*/
33     /*Esto es lo que carga al llamar al componente*/
34    ngOnInit() { }
35    /*con esta funcion subimos la imagen al storage*/
36    onSubida(e: any) {
37      //console.log('subir',e.target.files[0]);
38      const id = Math.random().toString(35).substring(2, 6);/*generamos un id aleatorio para darle un nombre a la imagen
39      y no se repita el nombre*/
40      const file = e.target.files[0];/*aqui tomamos la imagen que se encuentra en esa ubicacion*/
41      const filePath = `Transportes/P${id}${e.target.files[0].name}`;/*aqui asignamos la ruta y nombre de la imagen*/
42      const ref = this.storage.ref(filePath);/*aqui le asignamos a la variable la referencia que necesita la imagen*/
43      const task = this.storage.upload(filePath, file);/*aqui subimos la imagen al storage de firebase*/
44      this.subPorc = task.percentageChanges();/*aqui vamos actualizando el valor de subporc para ir
45      mostrando la barra de carga*/
46      task.snapshotChanges().pipe(finalize(() => this.urlImg = ref.getDownloadURL())).subscribe();}/*aqui tomamos la url
47      y la asignamos a un observable*/
48    /*funcion para guardar un transporte el cual trae el formulario del html*/
49    onGuardarTransporte(formPro: NgForm) {
50      if (formPro.valid) {/*si el formulario es valido adiciona el objeto a la coleccion en firebase*/
51        this.transporte.Foto = this.InpImgProd.nativeElement.value;/*aqui le damos la url como string a el atributo foto del
52        transporte para subirlo a la bd */
53        this._bs.addTransporte(this.transporte);/*aqui adicinamos el transporte a la base de datos en firebase*/
54      } else {
55        console.log('El formulario no se lleno de manera correcta');/*Si el formulario no es valido muestra este
56        mensaje en la consola*/
57      }}
58  }
59
```

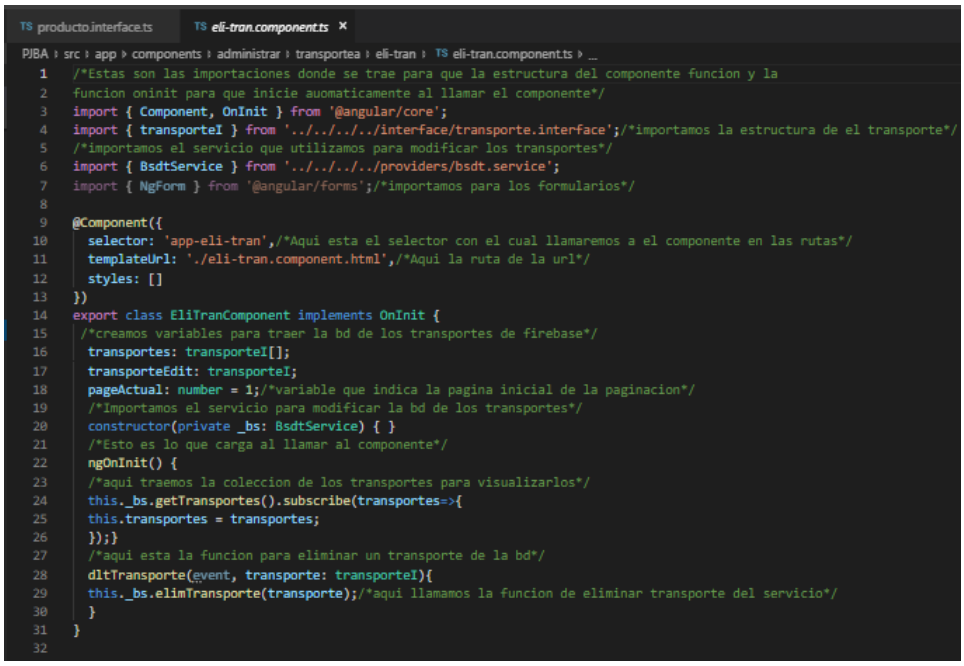
Ilustración 50: Desarrollo del Sprint 4.

producto.interface.ts | eli-tran.component.html

PJBA › src › app › components › administrar › transportea › eli-tran › eli-tran.component.html › ...

```
1   <!--En este apartado mostramos la bd de los transportes donde la podremos eliminar, si no hay nada en la
2   base de datos se mostrara el ng-template de abajo-->
3   <div class="row" style="width: 100%; height: 1%" *ngIf="transportes?.length > 0;else TransNo">
4     <div style="width: 100%; text-align: righ;padding: 40px; height: 10%;">
5       <h4 style="font-weight: bold; text-align: center;">Eliminar Transporte de la BD</h4><!--Titulo-->
6     </div>
7     <div class="AddBD" style="height: 85%;">
8       <!--aqui usamos un for para mostrar todos los transportes mostrando 2 por pagina-->
9       <div *ngFor="let transporte of transportes | paginate: {itemsPerPage:2, currentPage: pageActual}; let i=index"
10        class="MuesBD">
11        <ul>
12          <table class="Tpt"><!--en esta tabla va la foto, los datos del transporte y el formulario para eliminarlo-->
13            <tr>
14              <!--Aqui va la foto del transporte-->
15              <td style="width: 30%; border-radius: 30px 0px 0px 30px;">
16                <img src="{{transporte.Foto}}" class="FIPyT"></td>
17                <!--aqui van los datos del transporte de la bd-->
18              <td style="width: 60%;">
19                <div>
20                  <label>Codigo:    {{transporte.Codigo}}</label><br>
21                  <label>Nombre:    {{transporte.Nombre}}</label><br>
22                  <label>Disponibilidad:    {{transporte.Disponibilidad}}</label><br>
23                  <label>Cantidad:    {{transporte.Cantidad}}</label><br>
24                  <label>Descripcion:    {{transporte.Descripcion}}</label></div>
25              </td><!--hasta qui se ven los datos de un transporte-->
26              <!--en este apartado se muestra el boton para eliminar un transporte con su estilo-->
27              <td style="width: 10%; border-radius: 0px 30px 30px 0px;">
28                <ul style="text-align: center; margin: 5% 7%;"><button (click)="dltTransporte($event, transporte)"
29                    class="btn btn-danger" style="height:40px;">Eliminar</button></ul>
30              </td>
31            </tr>
32          </table>
33        </ul>
34      </div>
35      <!--aqui esta la paginacion de los transportes donde la pagina actual es la primera-->
36      <div class="pageCS">
37        <pagination-controls (pageChange)="pageActual = $event" style="text-align: center;"></pagination-controls></div>
38    </div>
39  </div>
40  <!--Se muestra si no hay ningun transporte en la bd donde se llama con la validacion de arriba-->
41  <ng-template #TransNo>
42    <div style="width: 100%; text-align: righ;padding: 30% 50px; height: 10%;">
43      <h4 style="font-weight: bold; text-align: center;">No existen datos en la BD</h4>
44    </div>
45  </ng-template>
46
```

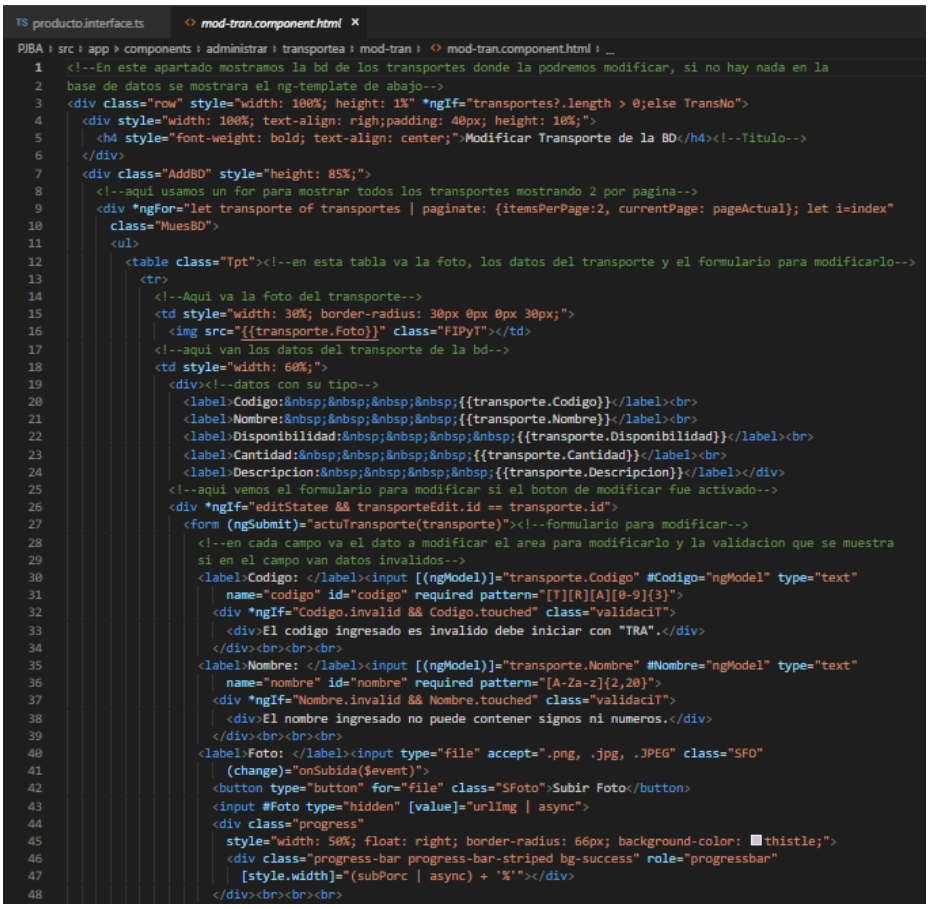
Ilustración 51: Desarrollo del Sprint 4.



```ts
/*Estas son las importaciones donde se trae para que la estructura del componente funcion y la
funcion oninit para que inicie auomaticamente al llamar el componente*/
import { Component, OnInit } from '@angular/core';
import { transporteI } from '../../../../interface/transporte.interface';/*importamos la estructura de el transporte*/
/*importamos el servicio que utilizamos para modificar los transportes*/
import { BsdtService } from '../../../../providers/bsdt.service';
import { NgForm } from '@angular/forms';/*importamos para los formularios*/

@Component({
  selector: 'app-eli-tran',/*Aqui esta el selector con el cual llamaremos a el componente en las rutas*/
  templateUrl: './eli-tran.component.html',/*Aqui la ruta de la url*/
  styles: []
})
export class EliTranComponent implements OnInit {
 /*creamos variables para traer la bd de los transportes de firebase*/
  transportes: transporteI[];
  transporteEdit: transporteI;
  pageActual: number = 1;/*variable que indica la pagina inicial de la paginacion*/
  /*Importamos el servicio para modificar la bd de los transportes*/
  constructor(private _bs: BsdtService) { }
  /*Esto es lo que carga al llamar al componente*/
  ngOnInit() {
  /*aqui traemos la coleccion de los transportes para visualizarlos*/
  this._bs.getTransportes().subscribe(transportes=>{
  this.transportes = transportes;
  });}
  /*aqui esta la funcion para eliminar un transporte de la bd*/
  dltTransporte(event, transporte: transporteI){
  this._bs.elimTransporte(transporte);/*aqui llamamos la funcion de eliminar transporte del servicio*/
  }
}
```

Ilustración 52: Desarrollo del Sprint 4.



```html
<!--En este apartado mostramos la bd de los transportes donde la podremos modificar, si no hay nada en la
base de datos se mostrara el ng-template de abajo-->
<div class="row" style="width: 100%; height: 1%" *ngIf="transportes?.length > 0;else TransNo">
  <div style="width: 100%; text-align: righ;padding: 40px; height: 10%;">
    <h4 style="font-weight: bold; text-align: center;">Modificar Transporte de la BD</h4><!--Titulo-->
  </div>
  <div class="AddBD" style="height: 85%;">
    <!--aqui usamos un for para mostrar todos los transportes mostrando 2 por pagina-->
    <div *ngFor="let transporte of transportes | paginate: {itemsPerPage:2, currentPage: pageActual}; let i=index"
      class="MuesBD">
      <ul>
        <table class="Tpt"><!--en esta tabla va la foto, los datos del transporte y el formulario para modificarlo-->
          <tr>
            <!--Aqui va la foto del transporte-->
            <td style="width: 30%; border-radius: 30px 0px 0px 30px;">
              <img src="{{transporte.Foto}}" class="FIPyT"></td>
            <!--aqui van los datos del transporte de la bd-->
            <td style="width: 60%;">
              <div><!--datos con su tipo-->
                <label>Codigo:    {{transporte.Codigo}}</label><br>
                <label>Nombre:    {{transporte.Nombre}}</label><br>
                <label>Disponibilidad:    {{transporte.Disponibilidad}}</label><br>
                <label>Cantidad:    {{transporte.Cantidad}}</label><br>
                <label>Descripcion:    {{transporte.Descripcion}}</label></div>
              <!--aqui vemos el formulario para modificar si el boton de modificar fue activado-->
              <div *ngIf="editStatee && transporteEdit.id == transporte.id">
                <form (ngSubmit)="actuTransporte(transporte)"><!--formulario para modificar-->
                  <!--en cada campo va el dato a modificar el area para modificarlo y la validacion que se muestra
                  si en el campo van datos invalidos-->
                  <label>Codigo: </label><input [(ngModel)]="transporte.Codigo" #Codigo="ngModel" type="text"
                    name="codigo" id="codigo" required pattern="[T][R][A][0-9]{3}">
                  <div *ngIf="Codigo.invalid && Codigo.touched" class="validaciT">
                    <div>El codigo ingresado es invalido debe iniciar con "TRA".</div>
                  </div><br><br><br>
                  <label>Nombre: </label><input [(ngModel)]="transporte.Nombre" #Nombre="ngModel" type="text"
                    name="nombre" id="nombre" required pattern="[A-Za-z]{2,20}">
                  <div *ngIf="Nombre.invalid && Nombre.touched" class="validaciT">
                    <div>El nombre ingresado no puede contener signos ni numeros.</div>
                  </div><br><br><br>
                  <label>Foto: </label><input type="file" accept=".png, .jpg, .JPEG" class="SFO"
                    (change)="onSubida($event)">
                  <button type="button" for="file" class="SFoto">Subir Foto</button>
                  <input #Foto type="hidden" [value]="urlImg | async">
                  <div class="progress"
                    style="width: 50%; float: right; border-radius: 66px; background-color: thistle;">
                    <div class="progress-bar progress-bar-striped bg-success" role="progressbar"
                      [style.width]="(subPorc | async) + '%'"></div>
                  </div><br><br><br>
```

Ilustración 53: Desarrollo del Sprint 4.

producto.interface.ts | mod-tran.component.html

PJBA › src › app › components › administrar › transportea › mod-tran › mod-tran.component.html › ...

```
                    <label>Disponibilidad: </label>
                      <select [(ngModel)]="transporte.Disponibilidad" #Disponibilidad="ngModel" name="disponibilidad"
                        id="disponibilidad" required>
                        <option value="Si">Si</option>
                        <option value="No">No</option>
                      </select>
                    <label>Cantidad: </label><input [(ngModel)]="transporte.Cantidad" #Cantidad="ngModel" type="text"
                        name="cantidad" id="cantidad" required pattern="[0-9]{1,20}">
                      <div *ngIf="Cantidad.invalid && Cantidad.touched" class="validaciT">
                        <div>El valor ingresado no es valido.</div>
                      </div><br><br><br>
                    <label>Descripcion: </label><textarea [(ngModel)]="transporte.Descripcion" #Descripcion="ngModel"
                        name="descripcion" id="descripcion" minlength="10" required></textarea>
                      <div *ngIf="Descripcion.invalid && Descripcion.touched " class="validaciT">
                        <div>La descripcion debe contener al menos 10 caracteres.</div>
                      </div><br><br><br>
                    <!--hasta aqui van los campos del formulario para modifiacar el transporte-->
                    <!--si no se ha subido una imagen el boton tiene otro estilo con el borde verde y un fondo transparente
                    al momento de recargar pasa a tener un estilo de fondo verde, el boton actualiza la informacion del
                    transporte-->
                    <ul style="text-align: center; margin: 5% 7%;"><button type="submit" class="btn btn-success"
                        style="width:100%; height:40px;" *ngIf="urlImg | async; else BDisa">Actualizar</button>
                      <!--se muestra si no se ha subido una imagen-->
                      <ng-template #BDisa>
                        <button type="submit" class="btn btn-outline-success"                                                    /div>
                        style="width:100%; height:40px;">Actualizar</button>
                      </ng-template>
                      </ul>
                  </form>
                </div>
              </td>
              <!--boton para habilitar la opcion de modificar el transporte y si ya esta habilitado entonces mostrara
              el boton de cerrar para ocultar el formulario de modificar-->
              <td style="width: 10%; border-radius: 0px 30px 30px 0px;">
                <button *ngIf="editStatee == false" (click)="editTransporte($event, transporte)" class="btn btn-success"
                  style="height:40px;">Modificar</button>
                <button *ngIf="editStatee && transporteEdit.id == transporte.id" (click)="clearState()" class="btn btn-info"
                  style="height:40px;">Cerrar</button>
              </td>
            </tr>
          </table>
        </ul>
      </div>
      <!--aqui esta la paginacion de los transportes donde la pagina actual es la primera-->
      <div class="pageCS">
        <pagination-controls (pageChange)="pageActual = $event" style="text-align: center;"></pagination-controls></div>
    </div>
  </div>
  <!--Se muestra si no hay ningun transporte en la bd donde se llama con la validacion de arriba-->
  <ng-template #TransNo>
    <div style="width: 100%; text-align: righ;padding: 30% 50px; height: 10%;">
      <h4 style="font-weight: bold; text-align: center;">No existen datos en la BD</h4>
    </div>
  </ng-template>
```

Ilustración 54: Desarrollo del Sprint 4.

producto.interface.ts | mod-tran.component.ts

PJBA › src › app › components › administrar › transportea › mod-tran › mod-tran.component.ts › ...

```
1  /*Estas son las importaciones donde se trae para que la estructura del componente funcion y la
2    funcion oninit para que inicie auomaticamente al llamar el componente
3    y el elementref y el view child para llamar a un elemento por referencia para tomar su valor*/
4    import { Component, OnInit, ElementRef, ViewChild } from '@angular/core';
5    import { transporteI } from '../../../../interface/transporte.interface';/*importamos la estructura de el transporte*/
6    /*importamos el servicio que utilizamos para modificar los transportes*/
7    import { BsdtService } from '../../../../providers/bsdt.service';
8    import { NgForm } from '@angular/forms'; /*importamos para los formularios*/
9    import { AngularFireStorage } from '@angular/fire/storage';/*importamos la dependencia para usar firebase storage*/
10   import { finalize } from 'rxjs/operators';/*importamos para usarlo en la url de la imagen del storage*/
11   import { Observable } from 'rxjs';/*importamos el observable para actuzaliar automaticamente los datos*/
```

Ilustración 55: Desarrollo del Sprint 4.



Ilustración 56: Desarrollo del Sprint 4.

```
TS producto.interface.ts    <> transportea.component.html ×
PJBA › src › app › components › administrar › transportea › <> transportea.component.html › ...
1   <!--Aqui tenemos los botones para adicionar, modificar y eliminar un transporte y la validacion de
2     si trae otro componente este se oculte-->
3   <div class="row" style="width: 100%; height: 100%" *ngIf="outlet.isActivated == false">
4     <div class="col-md-12 text-center" style="padding: 200px; line-height: 50px">
5       <ul [routerLink]="['Adicionar']"><button type="button" class="btn btn-warning"
6           style="width:300px; height:40px;">Adicionar</button></ul>
7       <ul [routerLink]="['Modificar']"><button type="button" class="btn btn-warning"
8           style="width:300px; height:40px;">Modificar</button></ul>
9       <ul [routerLink]="['Eliminar']"><button type="button" class="btn btn-warning"
10          style="width:300px; height:40px;">Eliminar</button></ul>
11    </div>
12  </div>
13  <!--Aqui traemos el componente al cual llamamos sobreescribiendo el contenido que muestra la pagina-->
14  <router-outlet #outlet="outlet"></router-outlet>
15
```

Ilustración 57: Desarrollo del Sprint 4.

```
TS producto.interface.ts    TS transportea.component.ts ●
PJBA › src › app › components › administrar › transportea › TS transportea.component.ts › TransporteaComponent
1   /*Estas son las importaciones donde se trae para que la estructura del componente funcion*/
2   import { Component} from '@angular/core';
3
4   @Component({
5     selector: 'app-transportea',/*Aqui esta el selector con el cual llamaremos a el componente en las rutas*/
6     templateUrl: './transportea.component.html',/*Aqui la ruta de la url*/
7     styles: []
8   })
9   export class TransporteaComponent{
10
11    constructor() { }
12
13  }
14
```

Ilustración 58: Desarrollo del Sprint 4.

3.4.5 Diagrama de contexto del Sprint 4

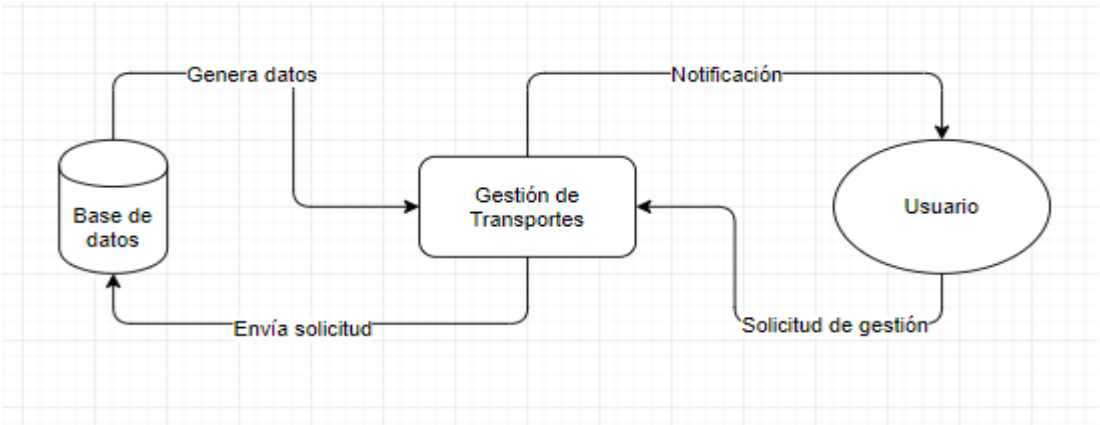


Ilustración 59: Diagrama del Sprint 4.

3.4.6 Prueba del Sprint 4

Sprint 4	Gestión de transporte	
Descripción de la prueba	Se realiza un CRUD de la BD de Transportes almacenada en la nube en FireBase.	
Herramientas	Opera y FireBase	
Numero	Prueba	Estado
1	El catálogo de los transportes se muestra sin problemas con los transportes que ya se han adicionado con antelación.	OK
2	La adición de cada transporte esta validada y correctamente ejecutada para que no ocasione futuros problemas.	OK
3	La modificación se realiza con total confianza manteniendo las mismas validaciones de la adición para que no se presenten problemas al modificar.	OK
4	La eliminación de cada transporte se realiza con seguridad sin dejar rastro del transporte.	OK

Tabla 20: Prueba del Sprint 4.

3.5 SPRINT 5 GESTION DE PRODUCTOS POSTULADOS

Esta fase de la metodología Scrum, el Sprint se basa en el manejo de los datos de los productos postulados, permite crear, modificar y eliminar los productos postulados que se encuentran en la base de datos de la plataforma.

3.5.1 Definición del Sprint 5

SPRINT 5	NOMBRE	Gestión de productos postulados
OBJETIVO	Permitir al usuario crear, modificar y eliminar los productos postulados de la base de datos de la plataforma.	
DETALLES		

Se implementara una interfaz que permita al cliente crear, modificar y al administrador modificar y eliminar los productos postulados de la base de datos de la plataforma.
CRITERIOS DE ACEPTACION
• Se debe validar el inicio de sesión que verifique el tipo de usuario. • El cliente debe poder crear y modificar los productos postulados. • El administrador debe poder modificar y eliminar los productos postulados. • El diseño debe de adaptarse a cualquier dispositivo.

Tabla 21: Descripción del Sprint 5.

3.5.2 Tareas del Sprint 5

Producto Backlog	Descripción de la tarea	Responsable	Estado	Duración (Días)
1	Se valida la información ingresada en los formularios de crear, modificar y eliminar productos postulados.	Brain Tribaldos, July Jimenez.	Terminado	1
2	Validar la información ingresada y con las opciones de Google, Twitter o Facebook.	Brain Tribaldos, July Jimenez.	Terminado	1
4	Permite al cliente crear el producto postulado.	Brain Tribaldos, July Jimenez.	Terminado	1
5	Permite al cliente modificar el producto postulado.	Brain Tribaldos, July Jimenez.	Terminado	1
5	Permite al administrador modificar el producto postulado.	Brain Tribaldos, July Jimenez.	Terminado	1
6	Permite al administrador	Brain Tribaldos, July Jimenez.	Terminado	1

	eliminar el producto postulado.			
10	Se creó un diseño de interfaz que permitiera al usuario acceder a la plataforma.	Brain Tribaldos, July Jimenez.	Terminado	1

Tabla 22: Tareas del Sprint 5.

3.5.3 Diseño del Sprint 5

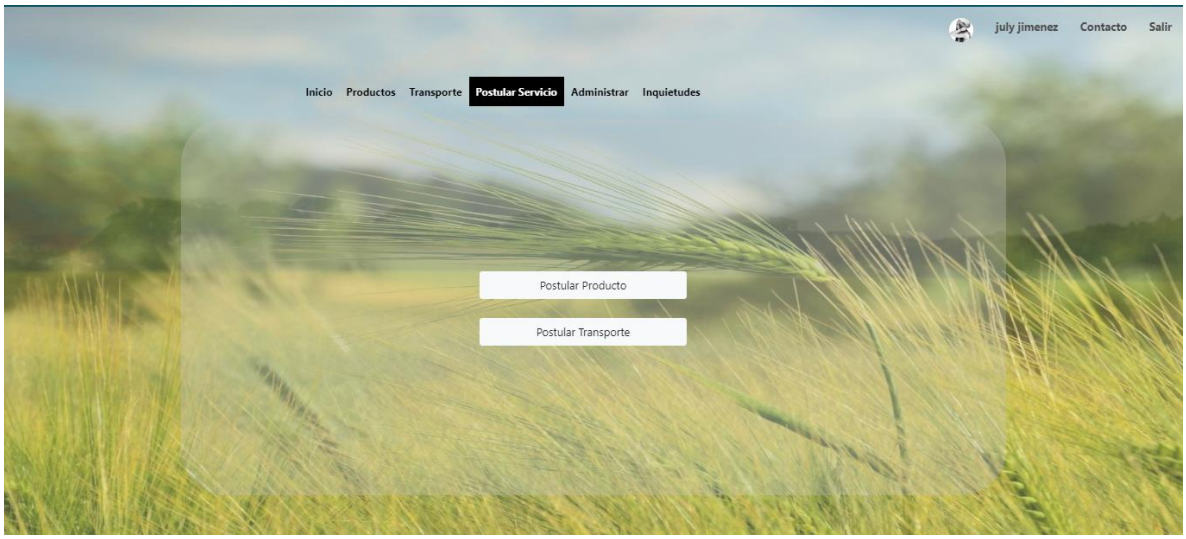


Ilustración 60: Diseño del Sprint 5.

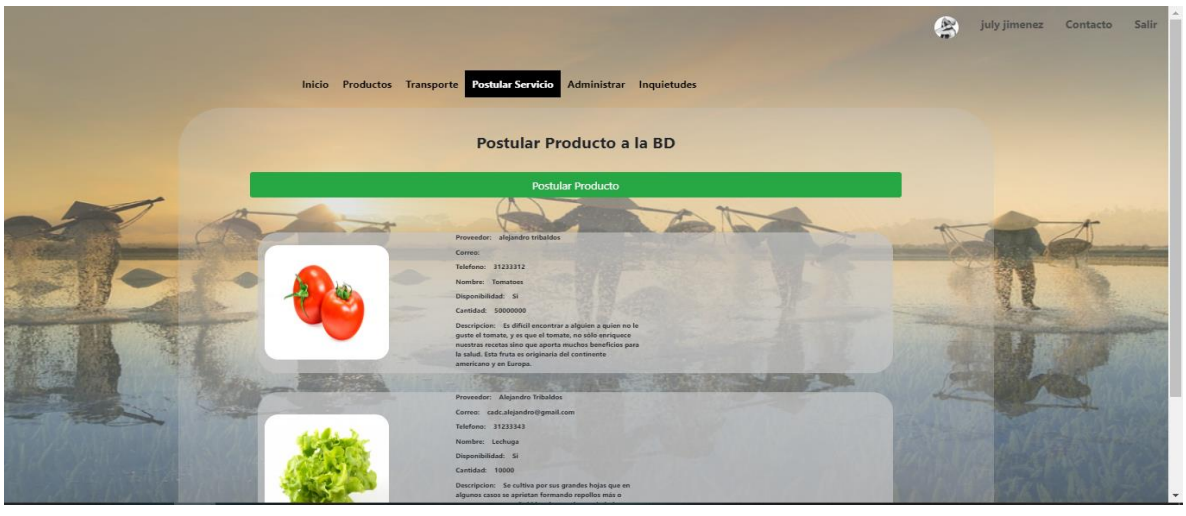


Ilustración 61: Diseño del Sprint 5.

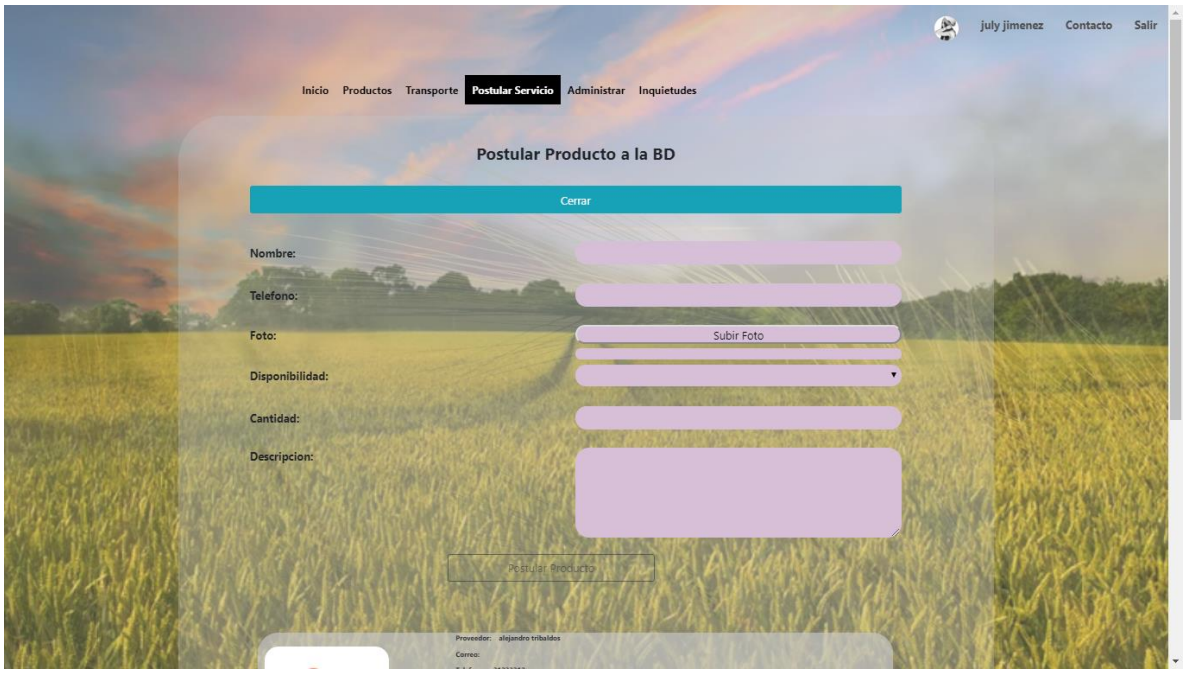


Ilustración 62: Diseño del Sprint 5.

3.5.4 Proceso de desarrollo de software del Sprint 5

postproducto.component.html

PJBA › src › app › components › postulaciones › postproducto › postproducto.component.html › ...

```
<!--En este apartado mostramos el formulario para postular un producto a la bd en firebase-->
<div class="row" style="width: 100%; height: 5%">
  <div style="width: 100%; text-align: righ;padding: 40px 0px 25px 0px;">
    <h4 style="font-weight: bold; text-align: center;">Postular Producto a la BD</h4>
    </div>
    <!--Titulo-->
    <div class="AddBDA">
    <table class="PosSer">
        <tr><td style="padding: 0px 0px 40px 0px;;"><ul style="text-align: center;">
        <!--Botones para habilitar el formulario de postular si no se visualiza y para ocultarlo si esta abierto-->
        <button *ngIf="editStatePProd == false" (click)="editProducto($event)" class="btn btn-success"
          style="width:100%; height:40px;">Postular Producto</button>
        <button *ngIf="editStatePProd" (click)="clearState()" class="btn btn-info"
          style="width:100%; height:40px;">Cerrar</button></ul>
        </td>
        </tr>
        <!--si se habilita el boton se muestra el formulario-->
        <tr *ngIf="editStatePProd"><td>
         <!--aqui vemos el formulario para adicionar un producto-->
        <form (ngSubmit)="onGuardarProductoPost(formProP)" #formProP="ngForm">
          <!--formulario para postular productos-->
          <!--en cada campo va el dato a ingresar y el area para ingresarlo y la validadcion que se muestra
        si en el campo van datos invalidos-->
          <ul><label>Nombre: </label><input [(ngModel)]="pproducto.Nombre" #Nombre="ngModel" type="text" name="nombre"
              id="nombre" minlength="5" required>
            <div *ngIf="Nombre.invalid && Nombre.touched" class="validaciT">
              <div>El nombre es un campo obligatorio mayor a 5 caracteres.</div>
            </div>
          </ul><br>
```

Ilustración 63: Desarrollo del Sprint 5.

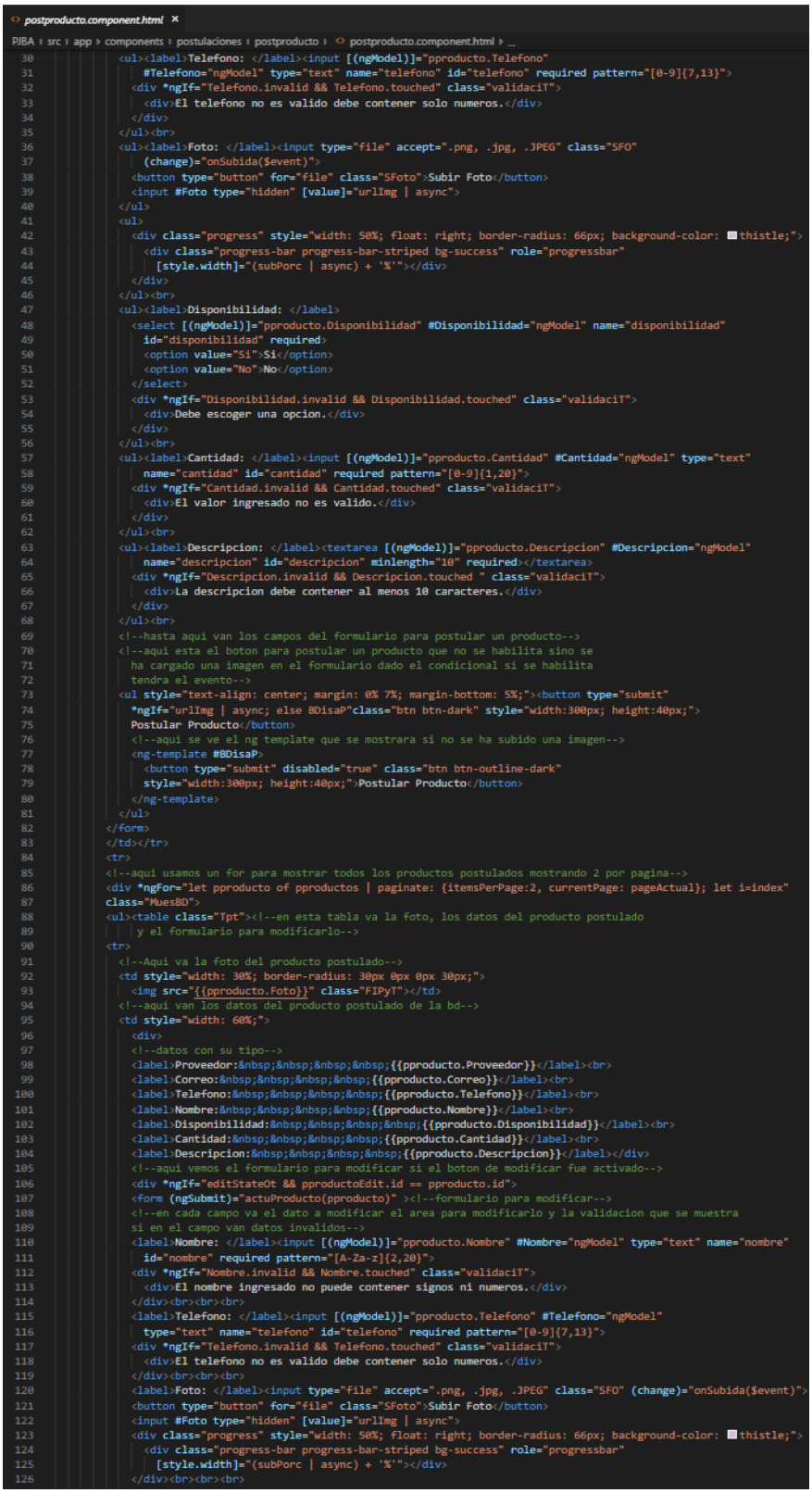


Ilustración 64: Desarrollo del Sprint 5.

postproducto.component.html

PJBA › src › app › components › postulaciones › postproducto › postproducto.component.html › ...

```
127                <label>Disponibilidad: </label>
128                <select [(ngModel)]="pproducto.Disponibilidad" #Disponibilidad="ngModel" name="disponibilidad"
129                  id="disponibilidad" required>
130                  <option value="Si">Si</option>
131                  <option value="No">No</option>
132                </select>
133                <label>Cantidad: </label><input [(ngModel)]="pproducto.Cantidad" #Cantidad="ngModel" type="text"
134                  name="cantidad" id="cantidad" required pattern="[0-9]{1,20}">
135                <div *ngIf="Cantidad.invalid && Cantidad.touched" class="validaciT">
136                  <div>El valor ingresado no es valido.</div>
137                </div><br><br><br>
138                <label>Descripcion: </label><textarea [(ngModel)]="pproducto.Descripcion" #Descripcion="ngModel"
139                  name="descripcion" id="descripcion" minlength="10" required></textarea>
140                <div *ngIf="Descripcion.invalid && Descripcion.touched " class="validaciT">
141                  <div>La descripcion debe contener al menos 10 caracteres.</div>
142                </div><br><br><br>
143                <!--hasta aqui van los campos del formulario para modificar el producto postulado-->
144                <!--si no se ha subido una imagen el boton tiene otro estilo con el borde verde y un fondo transparente
145                  al momento de recargar pasa a tener un estilo de fondo verde, el boton actualiza la informacion del
146                  producto postulado-->
147                <ul style="text-align: center; margin: 5% 7%;"><button type="submit"
148                  class="btn btn-success" style="width:100%; height:40px;" *ngIf="urlImg | async; else
149                  BDisaPM">Actualizar</button>
150                <!--se muestra si no se ha subido una imagen-->
151                <ng-template #BDisaPM>
152                  <button type="submit" class="btn btn-outline-success"
153                  style="width:100%; height:40px;">Actualizar</button>
154                </ng-template>
155                </ul>
156                <ul style="text-align: center; margin: 5% 7%;"><button (click)="dltPProducto($event, pproducto)"
157                    class="btn btn-danger" style="width:100%; height:40px;">Eliminar</button></ul>
158                </form></div>
159              </td>
160              <!--boton para habilitar la opcion de modificar el producto postulado y si ya esta habilitado entonces mostrara
161                el boton de cerrar para ocultar el formulario de modificar-->
162              <td style="width: 10%; border-radius: 0px 30px 30px 0px;" *ngIf="_is.Usuer.Uid == pproducto.Uid">
163                <button *ngIf="editStateOt == false " (click)="editPProductoOt($event, pproducto)"
164                class="btn btn-success" style="height:40px;">Modificar</button>
165                <button *ngIf="editStateOt && pproductoEdit.id == pproducto.id " (click)="clearStateOt()"
166                class="btn btn-info" style="height:40px;">Cerrar</button>
167              </td><td style="width: 10%; border-radius: 0px 30px 30px 0px;" *ngIf="_is.Usuer.Uid != pproducto.Uid">
168                 </td></tr></table></ul>
169            </div>
170            <!--aqui esta la paginacion de los productos postulados donde la pagina actual es la primera-->
171            <div class="pageCS">
172            <pagination-controls (pageChange)="pageActual = $event" style="text-align: center;"></pagination-controls>
173            </div>
174            </tr>
175        </table>
176        </div>
177    </div>
```

Ilustración 65: Desarrollo del Sprint 5.



Ilustración 66: Desarrollo del Sprint 5.

```
<!--Aqui tenemos los botones para postular un transporte o un producto y la validacion de
  si trae otro componente este se oculte-->
<div class="row" style="width: 100%; height: 100%" *ngIf="outlet.isActivated == false">
  <div class="col-md-12 text-center" style="padding: 200px; line-height: 50px">
    <ul [routerLink]="['PostularProducto']"><button type="button" class="btn btn-light"
        style="width:300px; height:40px;">Postular Producto</button></ul>
    <ul [routerLink]="['PostularTransporte']"><button type="button" class="btn btn-light"
        style="width:300px; height:40px;">Postular Transporte</button></ul>
  </div>
</div>
<!--Aqui traemos el componente al cual llamamos sobreescribiendo el contenido que muestra la pagina-->
<router-outlet #outlet="outlet"></router-outlet>
```

Ilustración 67: Desarrollo del Sprint 5.

```
/*Estas son las importaciones donde se trae para que la estructura del componente funcion*/
import { Component } from '@angular/core';

@Component({
  selector: 'app-postulaciones',/*Aqui esta el selector con el cual llamaremos a el componente en las rutas*/
  templateUrl: './postulaciones.component.html',/*Aqui la ruta de la url*/
  styles: []
})
export class PostulacionesComponent {

  constructor() { }


}
```

Ilustración 68: Desarrollo del Sprint 5.

3.5.5 Diagrama de contexto del Sprint 5

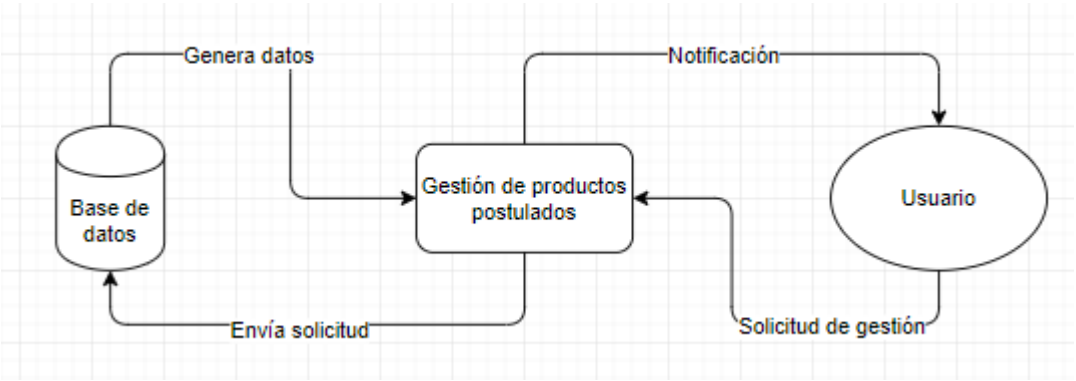


Ilustración 69: Diagrama del Sprint 5.

3.5.6 Prueba del Sprint 5

Sprint 5	Gestión de productos postulados	
Descripción de la prueba	Se realiza un CRUD de los productos postulados por parte del cliente y se modifica el estado por parte del administrador para ver si el producto se valida o no.	
Herramientas	Opera y FireBase	
Numero	Prueba	Estado
1	El catálogo de productos postulados se muestra con los datos que se encuentran en la BD de FireBase.	OK
2	El cliente postula su producto para su futura evaluación para conocer si es válido o no para agregarlo a los productos de la página.	OK
3	Se modifican los productos por parte del cliente si algún dato es incorrecto o desea cambiarlo y por parte del administrador el estado para conocer si es válido o no para ser ingresado a los productos.	OK
4	Los productos postulados son eliminados por arte del cliente si no desea ofrecerlos o por parte del administrador después de haber pasado un tiempo prudente de su evaluación para eliminar datos basura de la página.	OK

Tabla 23: Prueba del Sprint 5.

3.6 SPRINT 6 GESTION DE TRANSPORTE POSTULADOS

Esta fase de la metodología Scrum, el Sprint se basa en el manejo de los datos de los transportes postulados, permite crear, modificar y eliminar los transportes postulados que se encuentran en la base de datos de la plataforma.

3.6.1 Definición del Sprint 6

SPRINT 6	NOMBRE	Gestión de transportes postulados
OBJETIVO	Permitir al usuario crear, modificar y eliminar los transportes postulados de la base de datos de la plataforma.	
DETALLES		
Se implementara una interfaz que permita al cliente crear, modificar y al administrador modificar y eliminar los transportes postulados de la base de datos de la plataforma.		
CRITERIOS DE ACEPTACION		
• Se debe validar el inicio de sesión que verifique el tipo de usuario. • El cliente debe poder crear y modificar los transportes postulados. • El administrador debe poder modificar y eliminar los transportes postulados. • El diseño debe de adaptarse a cualquier dispositivo.		

Tabla 24: Definición del Sprint 6.

3.6.2 Tareas del Sprint 6

Producto Backlog	Descripción de la tarea	Responsable	Estado	Duración (Días)
1	Se valida la información ingresada en los formularios de crear, modificar y eliminar transporte postulado.	Brain Tribaldos, July Jimenez.	Terminado	1
2	Validar la información ingresada y con las opciones de Google, Twitter o Facebook.	Brain Tribaldos, July Jimenez.	Terminado	1
10	Se creó un diseño de interfaz que permitiera al usuario acceder a la plataforma.	Brain Tribaldos, July Jimenez.	Terminado	1
22	Permite al cliente crear el transporte postulado.	Brain Tribaldos, July Jimenez.	Terminado	1
23	Permite al cliente modificar el transporte postulado.	Brain Tribaldos, July Jimenez.	Terminado	1
24	Permite al cliente	Brain Tribaldos,	Terminado	1

	eliminar el transporte postulado.	July Jimenez.		

Tabla 25: Tareas del Sprint 6.

3.6.3 Diseño del Sprint 6

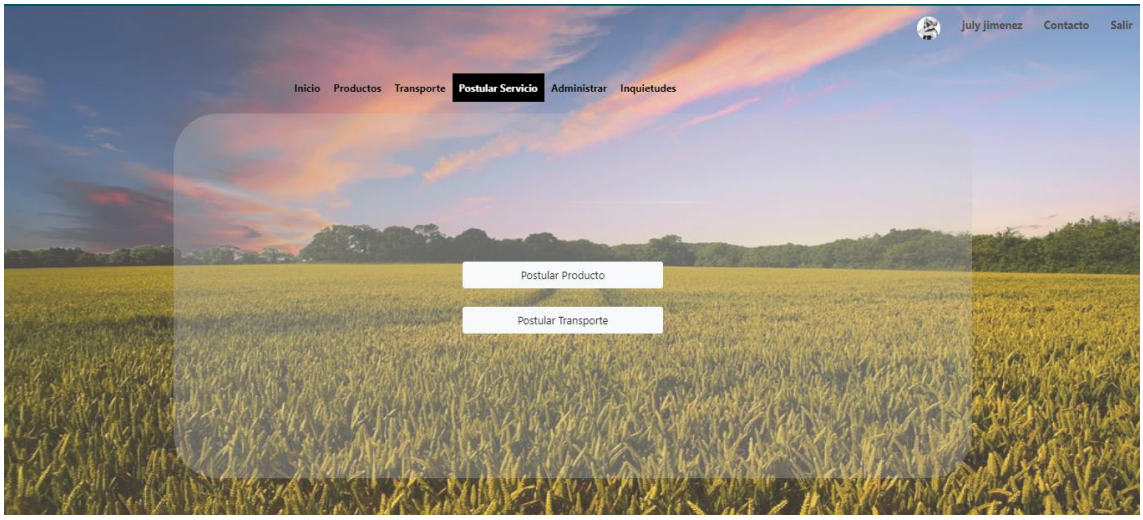


Ilustración 70: Diseño del Sprint 6.

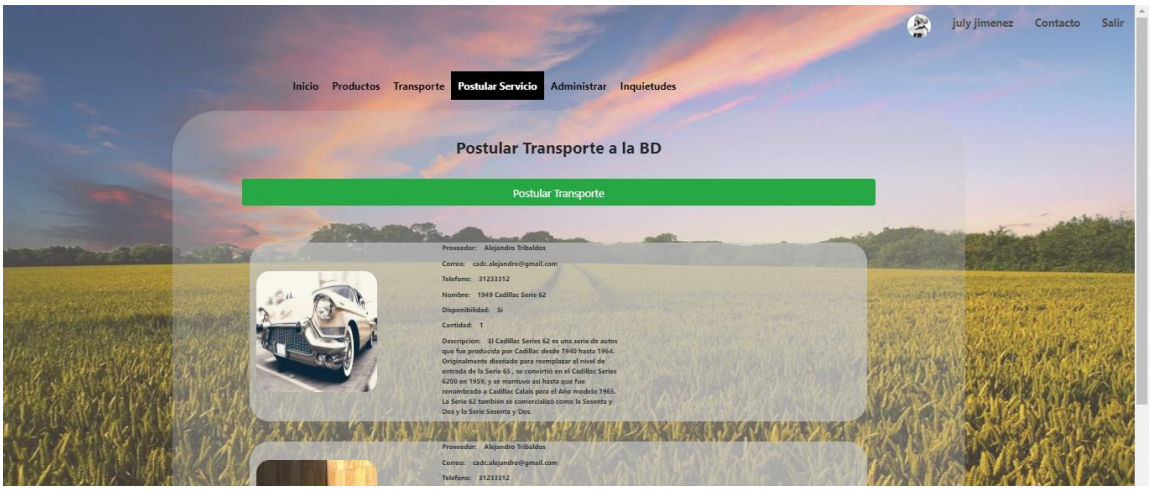


Ilustración 71: Diseño del Sprint 6.

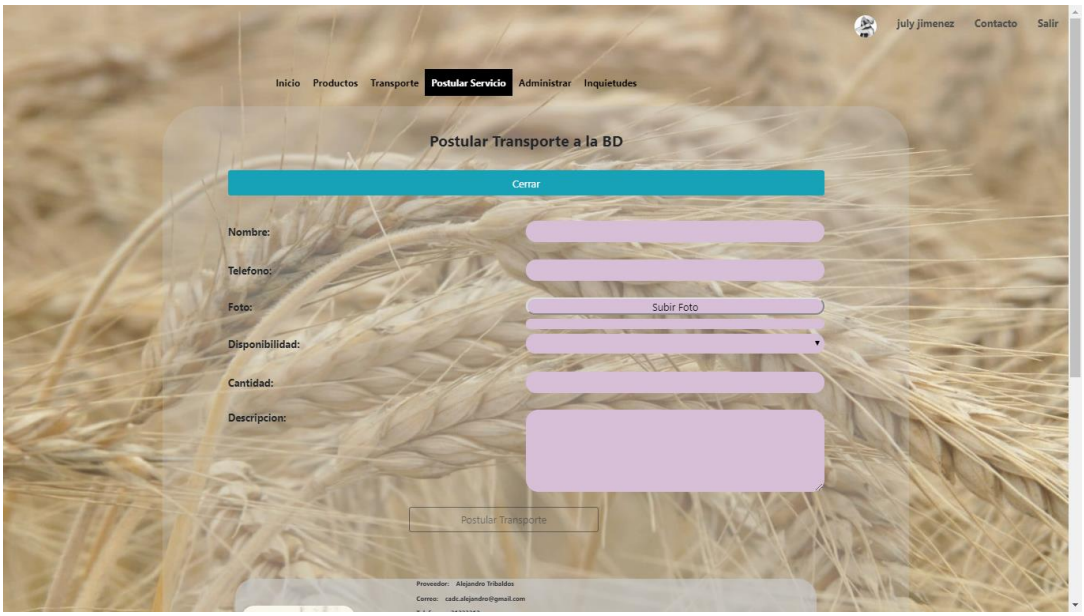


Ilustración 72: Diseño del Sprint 6.

3.6.4 Proceso de desarrollo de software del Sprint 6

posttransporte.component.html

PJBA › src › app › components › postulaciones › posttransporte › posttransporte.component.html › ...

```
<!--En este apartado mostramos el formulario para postular un transporte a la bd en firebase-->
<div class="row" style="width: 100%; height: 5%">
  <div style="width: 100%; text-align: righ;padding: 40px 0px 25px 0px;">
    <h4 style="font-weight: bold; text-align: center;">Postular Transporte a la BD</h4>
  </div>
  <!--Titulo-->
  <div class="AddBDA">
    <table class="PosSer">
      <tr>
        <td style="padding: 0px 0px 40px 0px;;">
          <ul style="text-align: center;">
            <!--Botones para habilitar el formulario de postular si no se visualiza y para ocultarlo si esta abierto-->
            <button *ngIf="editStatePTrans == false" (click)="editTransporte($event)" class="btn btn-success"
              style="width:100%; height:40px;">Postular Transporte</button>
            <button *ngIf="editStatePTrans" (click)="clearState()" class="btn btn-info"
              style="width:100%; height:40px;">Cerrar</button></ul>
        </td>
      </tr>
      <!--si se habilita el boton se muestra el formulario-->
      <tr *ngIf="editStatePTrans">
        <td>
          <!--aqui vemos el formulario para adicionar un transporte-->
          <form (ngSubmit)="onGuardarTransportePost(formTransP)" #formTransP="ngForm">
            <!--formulario para postular transportes-->
            <!--en cada campo va el dato a ingresar y el area para ingresarlo y la validadcion que se muestra
        si en el campo van datos invalidos-->
            <ul><label>Nombre: </label><input [(ngModel)]="ptransporte.Nombre" #Nombre="ngModel" type="text" name="nombre"
                id="nombre" minlength="5" required>
              <div *ngIf="Nombre.invalid && Nombre.touched" class="validaciT">
                <div>El nombre es un campo obligatorio mayor a 5 caracteres.</div>
              </div>
            </ul><br>
            <ul><label>Telefono: </label><input [(ngModel)]="ptransporte.Telefono"
              #Telefono="ngModel" type="text" name="telefono" id="telefono" required pattern="[0-9]{7,13}">
            <div *ngIf="Telefono.invalid && Telefono.touched" class="validaciT">
```

Ilustración 73: Desarrollo del Sprint 6.

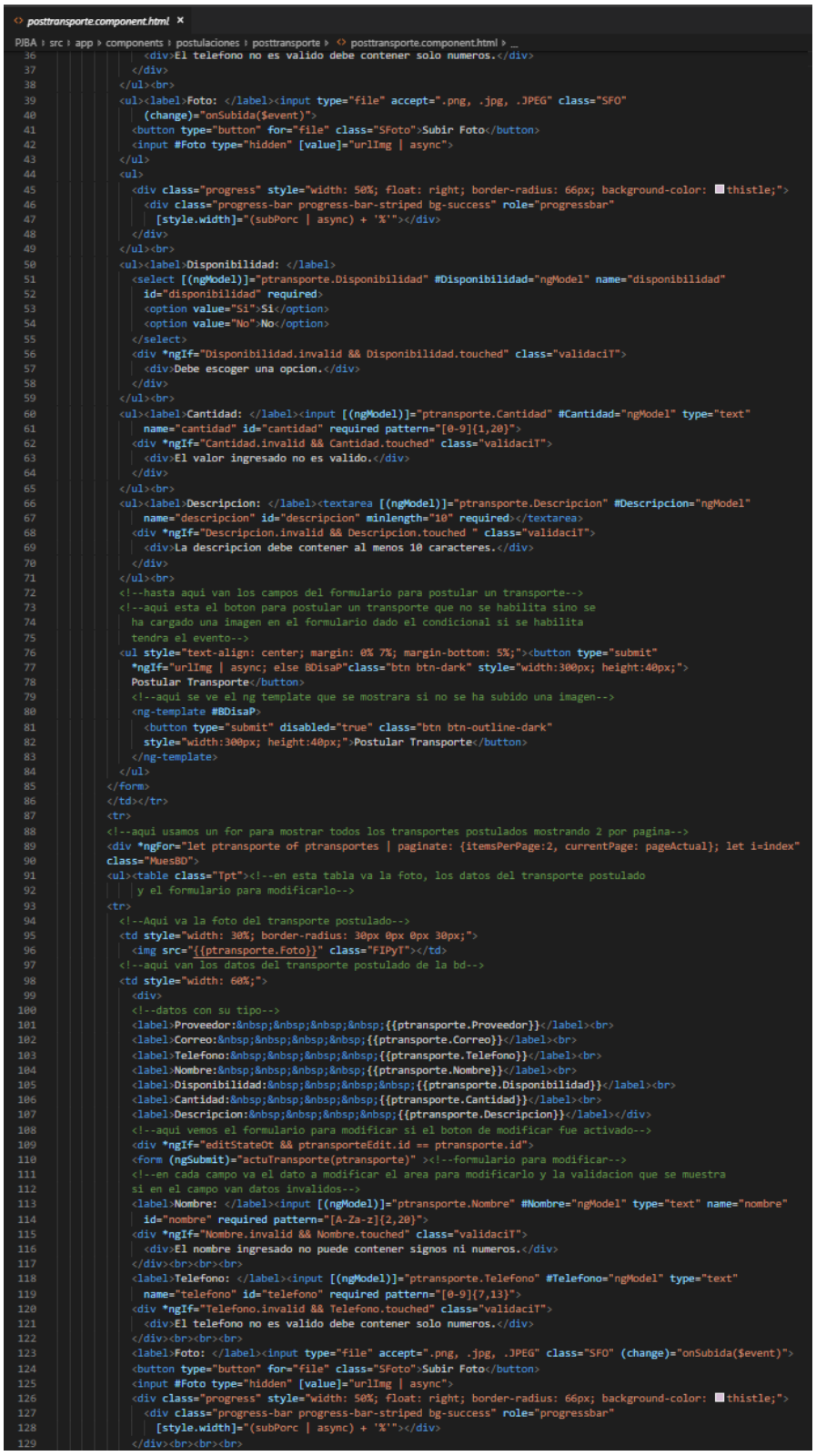


Ilustración 74: Desarrollo del Sprint 6.

posttransporte.component.html

PJBA › src › app › components › postulaciones › posttransporte › posttransporte.component.html › div.row › div.AddBDA › table.PosSer › tr › div.MuesBD

```
                <label>Disponibilidad: </label>
                <select [(ngModel)]="ptransporte.Disponibilidad" #Disponibilidad="ngModel" name="disponibilidad"
                  id="disponibilidad" required>
                  <option value="Si">Si</option>
                  <option value="No">No</option>
                </select>
                <label>Cantidad: </label><input [(ngModel)]="ptransporte.Cantidad" #Cantidad="ngModel" type="text"
                  name="cantidad" id="cantidad" required pattern="[0-9]{1,20}">
                <div *ngIf="Cantidad.invalid && Cantidad.touched" class="validaciT">
                  <div>El valor ingresado no es valido.</div>
                </div><br><br><br>
                <label>Descripcion: </label><textarea [(ngModel)]="ptransporte.Descripcion" #Descripcion="ngModel"
                  name="descripcion" id="descripcion" minlength="10" required></textarea>
                <div *ngIf="Descripcion.invalid && Descripcion.touched " class="validaciT">
                  <div>La descripcion debe contener al menos 10 caracteres.</div>
                </div><br><br><br>
                <!--hasta aqui van los campos del formulario para modificar el transporte postulado-->
                <!--si no se ha subido una imagen el boton tiene otro estilo con el borde verde y un fondo transparente
                  al momento de recargar pasa a tener un estilo de fondo verde, el boton actualiza la informacion del
                  transporte postulado-->
                <ul style="text-align: center; margin: 5% 7%;"><button type="submit" class="btn btn-success"
                    style="width:100%; height:40px;" *ngIf="urlImg | async; else
                  BDisaPM">Actualizar</button>
                  <!--se muestra si no se ha subido una imagen-->
                  <ng-template #BDisaPM>
                  <button type="submit" class="btn btn-outline-success"
                  style="width:100%; height:40px;">Actualizar</button>
                </ng-template>
                </ul>
                <ul style="text-align: center; margin: 5% 7%;"><button (click)="dltPTransporte($event, ptransporte)"
                    class="btn btn-danger" style="width:100%; height:40px;">Eliminar</button></ul>
                </form></div>
              </td>
              <!--boton para habilitar la opcion de modificar el transporte postulado y si ya esta habilitado entonces mostrara
                el boton de cerrar para ocultar el formulario de modificar-->
              <td style="width: 10%; border-radius: 0px 30px 30px 0px;" *ngIf="_is.Usuer.Uid == ptransporte.Uid">
                <button *ngIf="editStateOt == false " (click)="editPTransporteOt($event, ptransporte)"
                class="btn btn-success" style="height:40px;">Modificar</button>
                <button *ngIf="editStateOt && ptransporteEdit.id == ptransporte.id " (click)="clearStateOt()"
                class="btn btn-info" style="height:40px;">Cerrar</button>
              </td><td style="width: 10%; border-radius: 0px 30px 30px 0px;" *ngIf="_is.Usuer.Uid != ptransporte.Uid">
                 </td></tr></table></ul>
            </div>
            <!--aqui esta la paginacion de los transportes postulados donde la pagina actual es la primera-->
            <div class="pageCS">
            <pagination-controls (pageChange)="pageActual = $event" style="text-align: center;"></pagination-controls>
            </div>
            </tr>
        </table>
        </div>
    </div>
```

Ilustración 75: Desarrollo del Sprint 6.

posttransporte.component.ts

PJBA › src › app › components › postulaciones › posttransporte › posttransporte.component.ts › ...

```
/*Estas son las importaciones donde se trae para que la estructura del componente funcion y la
funcion oninit para que inicie auomaticamente al llamar el componente
y el elementref y el view child para llamar a un elemento por referencia para tomar su valor*/
import { Component, OnInit, ElementRef, ViewChild } from '@angular/core';
import { ptransporteI } from '../../../interface/ptransporte.interface';/*importamos la estructura de el transporte postulado*/
/*importamos el servicio que utilizamos para modificar los transportess postulados*/
import { PostularService } from '../../../providers/postular.service';
/*importamos el servicio que utilizamos para ver el usuario que postulara un transporte*/
import { InquietudesService } from '../../../providers/inquietudes.service';
import { NgForm } from '@angular/forms'; /*importamos para los formularios*/
import { AngularFireStorage } from '@angular/fire/storage';/*importamos la dependencia para usar firebase storage*/
import { finalize } from 'rxjs/operators';/*importamos para usarlo en la url de la imagen del storage*/
import { Observable } from 'rxjs';/*importamos el observable para actuzaliar automaticamente los datos*/
```

Ilustración 76: Desarrollo del Sprint 6.

```
posttransporte.component.ts
PJBA › src › app › components › postulaciones › posttransporte › TS posttransporte.component.ts › ...
@Component({
  selector: 'app-posttransporte',/*Aqui esta el selector con el cual llamaremos a el componente en las rutas*/
  templateUrl: './posttransporte.component.html',/*Aqui la ruta de la url*/
  styles: []
})
export class PosttransporteComponent implements OnInit {
  /*inicializamos el objeto transporte postulado con la estructura que trajimos de la interface*/
    ptransporte: ptransporteI = {
    Nombre: '',
    Foto: '',
    Disponibilidad: '',
    Descripcion: '',
    Cantidad: null,
    Telefono: null,
    Proveedor: '',
    Correo: '',
    Uid: '',
    };
  /*creamos variables para traer la bd de los transportes postulados de firebase*/
  ptransportes: ptransporteI[];
  ptransporteEdit: ptransporteI;
  editStateOt: boolean = false;/*variable para habilitar o deshabilitar botones del html*/
  pageActual: number = 1;/*variable que indica la pagina inicial de la paginacion*/
  editStatePTrans: boolean = false;/*variable para habilitar o deshabilitar botones del html*/
  subPorc: Observable<number>;/*variable que nos ayuda trae el porcentaje de subida de la imagen*/
  urlImg: Observable<string>;/*en esta variable guardamos la url de la imagen*/
  /*Importamos el servicio para adicionar a la bd de los transportes y el modulo de firebase storage*/
  constructor(private _ps: PostularService, private storage: AngularFireStorage, public _is: InquietudesService) { }
  @ViewChild('Foto', null) InpImgPProd: ElementRef;/*aqui le asignamos a una variable la url que trajimos
  de la imagen que subimos al storage*/
    /*Esto es lo que carga al llamar al componente*/
  ngOnInit() {
  /*aqui traemos la coleccion de los transportess para visualizarlos*/
  this._ps.getPTransportes().subscribe(ptransportes=>{
  this.ptransportes = ptransportes;
  });
  }
  /*con esta funcion subimos la imagen al storage*/
  onSubida(e: any) {
    //console.log('subir',e.target.files[0]);
    const id = Math.random().toString(35).substring(2, 6);/*generamos un id aleatorio para darle un nombre a la imagen
    y no se repita el nombre*/
    const file = e.target.files[0];/*aqui tomamos la imagen que se encuentra en esa ubicacion*/
    const filePath = `Postulaciones/Post${id}${e.target.files[0].name}`;/*aqui asignamos la ruta y nombre de la imagen*/
    const ref = this.storage.ref(filePath);/*aqui le asignamos a la variable la referencia que necesita la imagen*/
    const task = this.storage.upload(filePath, file);/*aqui subimos la imagen al storage de firebase*/
    this.subPorc = task.percentageChanges();/*aqui vamos actualizando el valor de subporc para ir
    mostrando la barra de carga*/
    task.snapshotChanges().pipe(finalize(() => this.urlImg = ref.getDownloadURL())).subscribe();
  }/*aqui tomamos la url
    y la asignamos a un observable*/
  /*funcion para guardar un transporte postulado el cual trae el formulario del html*/
  onGuardarTransportePost(formTransP: NgForm) {
    if (formTransP.valid) {/*si el formulario es valido adiciona el objeto a la coleccion en firebase*/
      /*Aignamos los datos del usuario*/
      this.ptransporte.Proveedor= this._is.Usuer.Usuario;
      this.ptransporte.Correo = this._is.Usuer.Correo;
      this.ptransporte.Uid = this._is.Usuer.Uid;
      this.ptransporte.Foto = this.InpImgPProd.nativeElement.value;/*aqui le damos la url como string a el atributo foto del
      transporte postulado para subirlo a la bd */
      this._ps.addTransporte(this.ptransporte);/*aqui adicinamos el transporte postulado a la base de datos en firebase*/
    } else {
      console.log('El formulario no se lleno de manera correcta');
    }/*Si el formulario no es valido muestra este
      mensaje en la consola*/
  }
  /*Aqui visualizamos el formulario con esta funcion*/
  editTransporte(event) {
    this.editStatePTrans = true;/*habilitamos la vista para modificaar*/
  }
  /*Aqui ocultamos el formulario con esta funcion*/
  clearState() {
    this.editStatePTrans = false;/*pone el valor auxiliar de estado en false*/
  }
  /*Funcion donde actualizamos el transporte postulado con la estructura que lleva el transporte*/
  actuTransporte(ptransporte: ptransporteI) {
    /*Asignamos los datos del usuario*/
    ptransporte.Foto = this.InpImgPProd.nativeElement.value;/*aqui le damos la url como string a el atributo foto del
  transporte postulado para subirlo a la bd*/
    this._ps.actualizarTransporte(ptransporte);/*aqui actualizamos el transporte postulado por id y le damos los datos que trajimos*/
    this.clearState();/*aqui llamamos la funcion que limpia las variables*/
  }
  /*aqui habilitamos la opcion para modificar el transporte postulado*/
  editPTransporteOt(event, ptransporte: ptransporteI) {
    this.editStateOt = true;/*habilitamos la vista para modificaar*/
    this.ptransporteEdit = ptransporte;/*aqui le asignamos los valores del transporte postulado a modificar a un objeto auxiliar*/
  }
  /*funcion que limpia las variables usadas*/
  clearStateOt() {
    this.editStateOt = false;/*pone el valor auxiliar de estado en false*/
    this.ptransporteEdit = null;/*pone en null el objeto auxiliar*/
  }
  /*aqui esta la funcion para eliminar un transporte postulado de la bd*/
  dltPTransporte(event, ptransporte: ptransporteI) {
    this._ps.elimTransporte(ptransporte);/*aqui llamamos la funcion de eliminar transporte postulado del servicio*/
  }
}
```

Ilustración 77: Desarrollo del Sprint 6.

3.6.5 Diagrama de contexto del Sprint 6

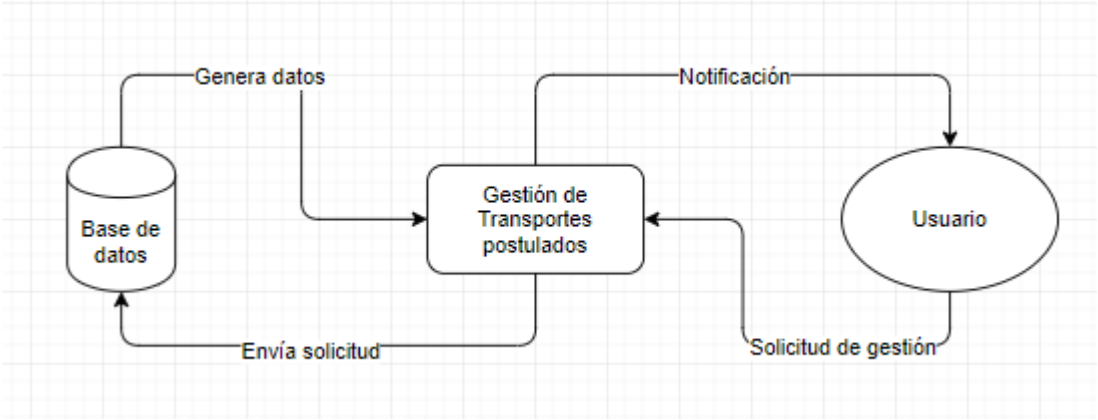


Ilustración 78: Diagrama del Sprint 6.

3.6.6 Prueba del Sprint 6

Sprint 6	Gestión de transportes postulados	
Descripción de la prueba	Se realiza un CRUD de los transportes postulados por parte del cliente y se modifica el estado por parte del administrador para ver si el transporte se valida o no.	
Herramientas	Opera y FireBase	
Numero	Prueba	Estado
1	El catálogo de transportes postulados se muestra con los datos que se encuentran en la BD de FireBase.	OK
2	El cliente postula su producto para su futura evaluación para conocer si es válido o no para agregarlo a los transportes de la página.	OK
3	Se modifican los transportes por parte del cliente si algún dato es incorrecto o desea cambiarlo y por parte del administrador el estado para	OK

	conocer si es válido o no para ser ingresado a los transportes.	
4	Los transportes postulados son eliminados por arte del cliente si no desea ofrecerlos o por parte del administrador después de haber pasado un tiempo prudente de su evaluación para eliminar datos basura de la página.	OK

Tabla 26: Prueba del Sprint 6.

3.7 SPRINT 7 GESTION DE INQUIETUDES

Esta fase de la metodología Scrum, el Sprint se hizo con el fin de que los usuarios tanto clientes como administradores pudieran comunicarse entre sí y resolver cualquier duda o problema que se le presente al cliente relacionado con los servicios prestados por la plataforma.

3.7.1 Definición del Sprint 7

SPRINT 7	NOMBRE	Gestión de inquietudes
OBJETIVO	Permitir la comunicación entre los usuarios de la plataforma.	
DETALLES		
Se creara una interfaz que permita a los usuarios tanto clientes como administradores comunicarse entre sí para poder brindar un mejor servicio.		
CRITERIOS DE ACEPTACION		
• Se debe validar el inicio de sesión que verifique el tipo de usuario. • Permitir a los usuarios generar sus comentarios o inquietudes. • El diseño debe de adaptarse a cualquier dispositivo.		

Tabla 27: Definición del Sprint 7.

3.7.2 Tareas del Sprint 7

Producto Backlog	Descripción de la tarea	Responsable	Estado	Duración (Días)
2	Validar la información ingresada y con las opciones de	Brain Tribaldos, July Jimenez.	Terminado	1

	Google, Twitter o Facebook.			
10	Se creó un diseño de interfaz que permitiera al usuario acceder a la plataforma.	Brain Tribaldos, July Jimenez.	Terminado	1
25	Permitir a los usuarios generar sus comentarios.	Brain Tribaldos, July Jimenez	Terminado	1

Tabla 28: Tareas del Sprint 7.

3.7.3 Diseño del Sprint 7

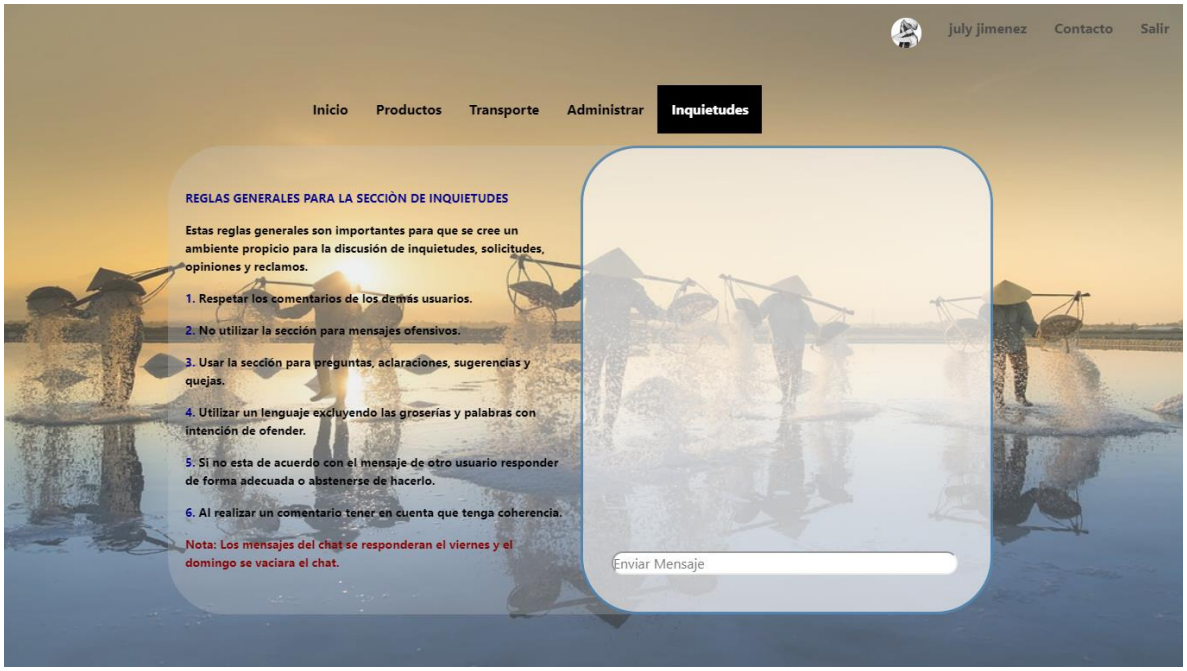
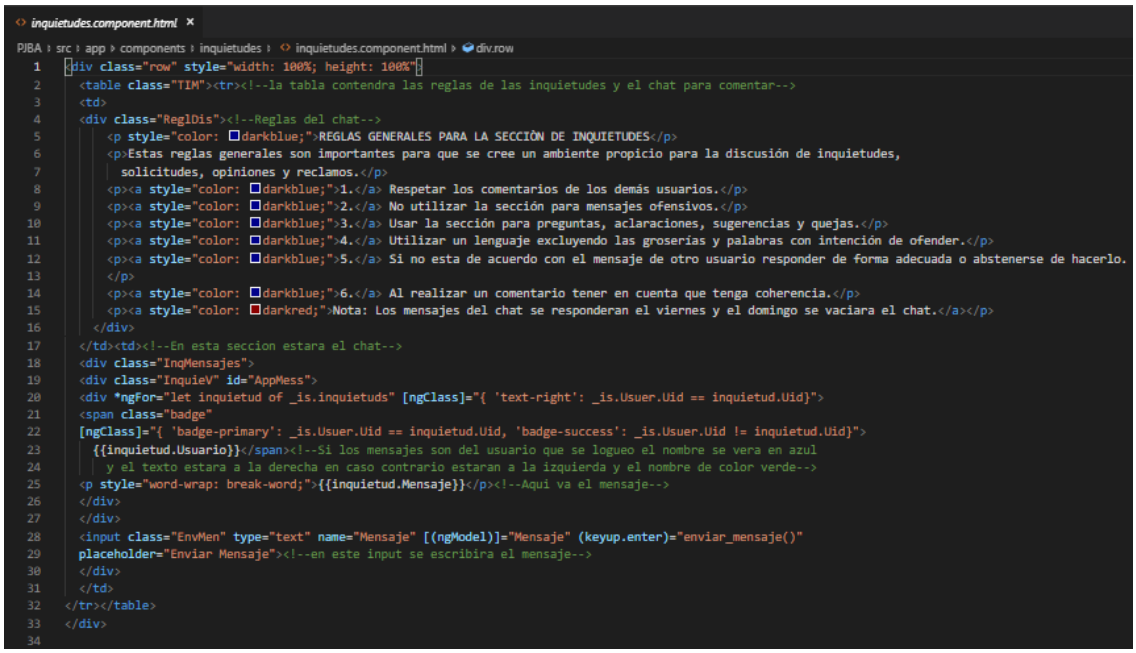


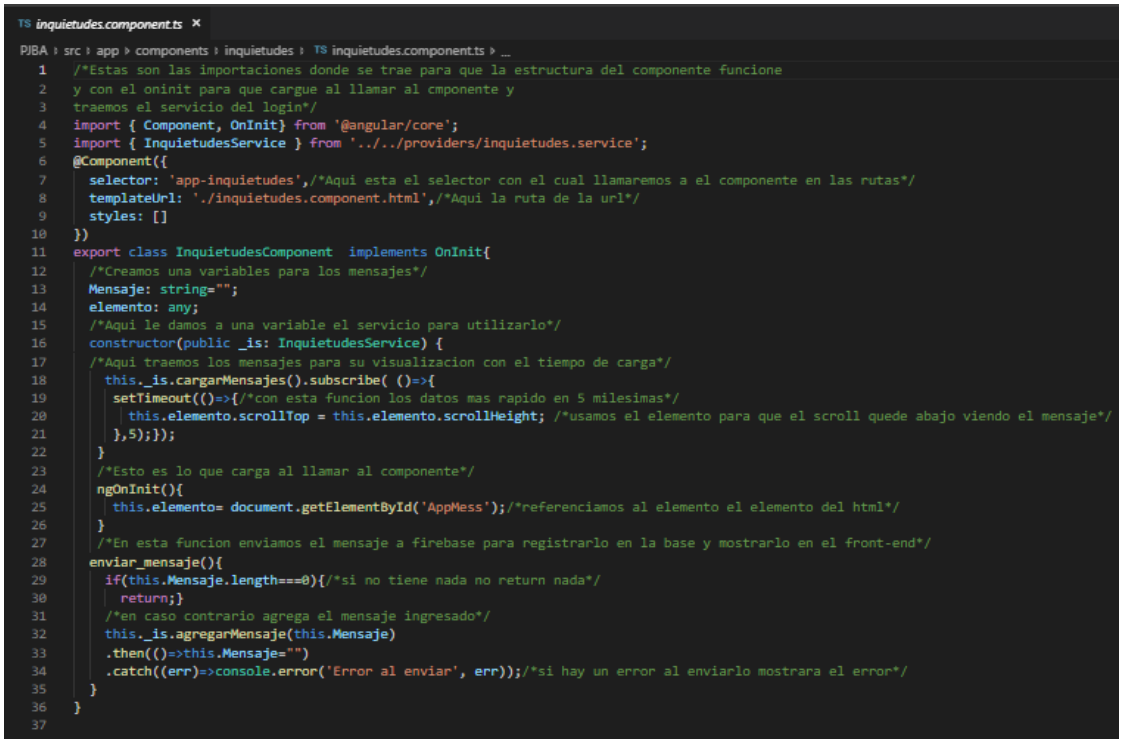
Ilustración 79: Diseño del Sprint 7.

3.7.4 Proceso de desarrollo de software del Sprint 7



```
<div class="row" style="width: 100%; height: 100%">
  <table class="TIM"><tr><!--la tabla contendra las reglas de las inquietudes y el chat para comentar-->
  <td>
  <div class="ReglDis"><!--Reglas del chat-->
      <p style="color: darkblue;">REGLAS GENERALES PARA LA SECCIÒN DE INQUIETUDES</p>
      <p>Estas reglas generales son importantes para que se cree un ambiente propicio para la discusión de inquietudes,
        solicitudes, opiniones y reclamos.</p>
      <p><a style="color: darkblue;">1.</a> Respetar los comentarios de los demás usuarios.</p>
      <p><a style="color: darkblue;">2.</a> No utilizar la sección para mensajes ofensivos.</p>
      <p><a style="color: darkblue;">3.</a> Usar la sección para preguntas, aclaraciones, sugerencias y quejas.</p>
      <p><a style="color: darkblue;">4.</a> Utilizar un lenguaje excluyendo las groserías y palabras con intención de ofender.</p>
      <p><a style="color: darkblue;">5.</a> Si no esta de acuerdo con el mensaje de otro usuario responder de forma adecuada o abstenerse de hacerlo.
      </p>
      <p><a style="color: darkblue;">6.</a> Al realizar un comentario tener en cuenta que tenga coherencia.</p>
      <p><a style="color: darkred;">Nota: Los mensajes del chat se responderan el viernes y el domingo se vaciara el chat.</a></p>
    </div>
  </td><td><!--En esta seccion estara el chat-->
  <div class="InqMensajes">
  <div class="InquieV" id="AppMess">
  <div *ngFor="let inquietud of _is.inquietuds" [ngClass]="{ 'text-right': _is.Usuer.Uid == inquietud.Uid}">
  <span class="badge"
  [ngClass]="{ 'badge-primary': _is.Usuer.Uid == inquietud.Uid, 'badge-success': _is.Usuer.Uid != inquietud.Uid}">
    {{inquietud.Usuario}}</span><!--Si los mensajes son del usuario que se logueo el nombre se vera en azul
      y el texto estara a la derecha en caso contrario estaran a la izquierda y el nombre de color verde-->
  <p style="word-wrap: break-word;">{{inquietud.Mensaje}}</p><!--Aqui va el mensaje-->
  </div>
  </div>
  <input class="EnvMen" type="text" name="Mensaje" [(ngModel)]="Mensaje" (keyup.enter)="enviar_mensaje()"
  placeholder="Enviar Mensaje"><!--en este input se escribira el mensaje-->
  </div>
  </td>
</tr></table>
</div>
```

Ilustración 80: Desarrollo del Sprint 7.



```
/*Estas son las importaciones donde se trae para que la estructura del componente funcione
y con el oninit para que cargue al llamar al cmponente y
traemos el servicio del login*/
import { Component, OnInit} from '@angular/core';
import { InquietudesService } from '../../providers/inquietudes.service';
@Component({
  selector: 'app-inquietudes',/*Aqui esta el selector con el cual llamaremos a el componente en las rutas*/
  templateUrl: './inquietudes.component.html',/*Aqui la ruta de la url*/
  styles: []
})
export class InquietudesComponent  implements OnInit{
  /*Creamos una variables para los mensajes*/
  Mensaje: string="";
  elemento: any;
  /*Aqui le damos a una variable el servicio para utilizarlo*/
  constructor(public _is: InquietudesService) {
  /*Aqui traemos los mensajes para su visualizacion con el tiempo de carga*/
    this._is.cargarMensajes().subscribe( ()=>{
     setTimeout(()=>{/*con esta funcion los datos mas rapido en 5 milesimas*/
       this.elemento.scrollTop = this.elemento.scrollHeight; /*usamos el elemento para que el scroll quede abajo viendo el mensaje*/
     },5);});
   }
   /*Esto es lo que carga al llamar al componente*/
   ngOnInit(){
     this.elemento= document.getElementById('AppMess');/*referenciamos al elemento el elemento del html*/
   }
   /*En esta funcion enviamos el mensaje a firebase para registrarlo en la base y mostrarlo en el front-end*/
  enviar_mensaje(){
    if(this.Mensaje.length===0){/*si no tiene nada no return nada*/
      return;}
    /*en caso contrario agrega el mensaje ingresado*/
    this._is.agregarMensaje(this.Mensaje)
    .then(()=>this.Mensaje="")
    .catch((err)=>console.error('Error al enviar', err));/*si hay un error al enviarlo mostrara el error*/
  }
}
```

Ilustración 81: Desarrollo del Sprint 7.

3.7.5 Diagrama de contexto del Sprint 7

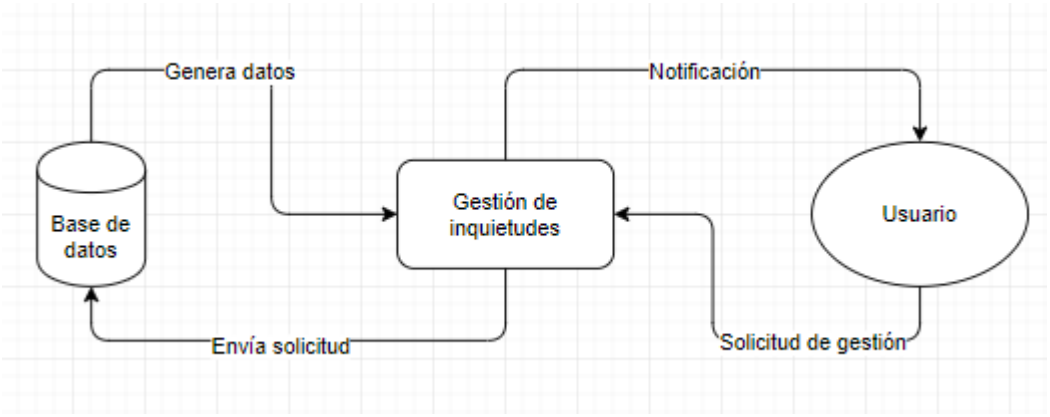


Ilustración 82: Diagrama del Sprint 7.

3.7.6 Prueba del Sprint 7

Sprint 7	Gestión de inquietudes	
Descripción de la prueba	En esta sección se visualiza un chat con las reglas de lo que se puede realizar para no generar inconvenientes con otro usuario donde se aclaran dudas, se dan sugerencias, entre otros.	
Herramientas	Opera y FireBase	
Numero	Prueba	Estado
1	Se puede ver y realizar comentarios y algunas otras observaciones de la plataforma para la retroalimentación de todos los usuarios implicados con la plataforma.	OK

Tabla 29: Prueba del Sprint 7.

3.8 SPRINT 8 INFORME

Esta fase de la metodología Scrum, el Sprint permitirá a los usuarios ver un informe por medio de consultas sobre los productos, los transportes y las compras realizadas.

3.8.1 Definición del Sprint 8

SPRINT 8	NOMBRE	Informe
OBJETIVO	Generar las consultas de productos, transportes y compras realizadas.	
DETALLES		
Se creara una interfaz que permita a los usuarios realizar las consultas de los productos, transportes y compras realizadas y se genere como un informe.		
CRITERIOS DE ACEPTACION		
• Se debe validar el inicio de sesión que verifique el tipo de usuario. • Permitir al usuario generar las consultas de productos, transportes y compras realizadas. • El diseño debe de adaptarse a cualquier dispositivo.		

Tabla 30: Definición del Sprint 8.

3.8.2 Tareas del Sprint 8

Producto Backlog	Descripción de la tarea	Responsable	Estado	Duración (Días)
2	Validar la información ingresada y con las opciones de Google, Twitter o Facebook.	Brain Tribaldos, July Jimenez.	Terminado	1
10	Se creó un diseño de interfaz que permitiera al usuario acceder a la plataforma.	Brain Tribaldos, July Jimenez.	Terminado	1
15	Permitir al usuario generar las consultas de productos, transportes y compras realizadas.	Brain Tribaldos, July Jimenez.	Terminado	1

Tabla 31: Tareas del Sprint 8.

3.8.3 Diseño del Sprint 8

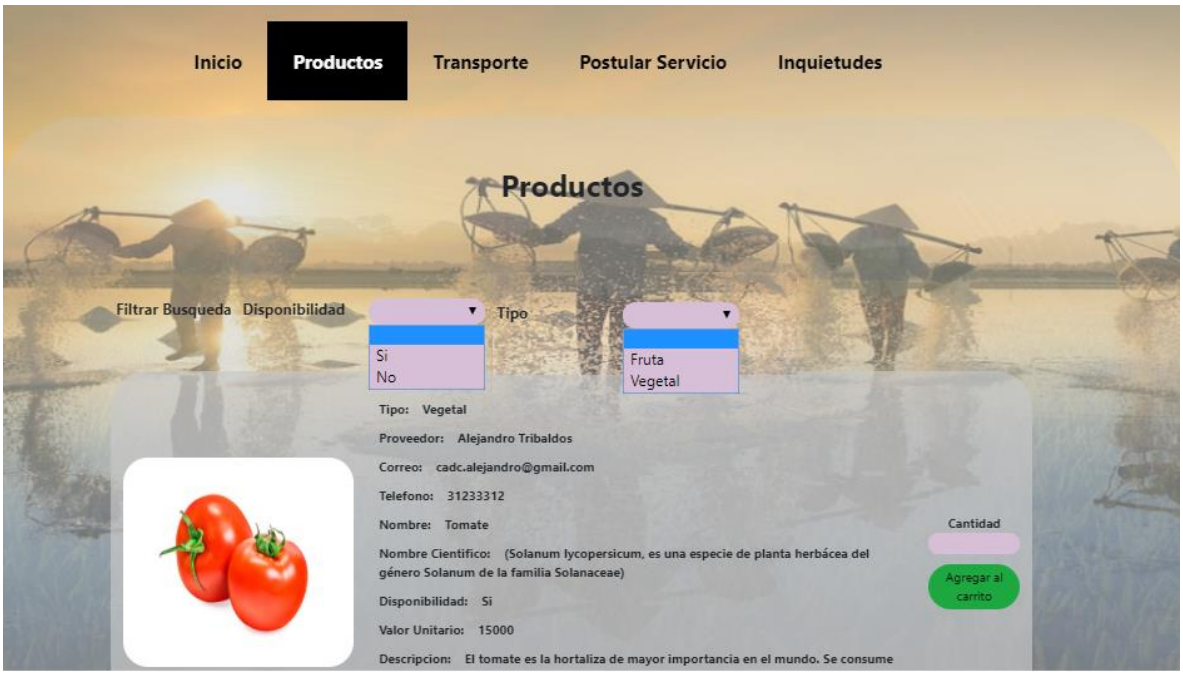


Ilustración 83: Diseño del Sprint 8.

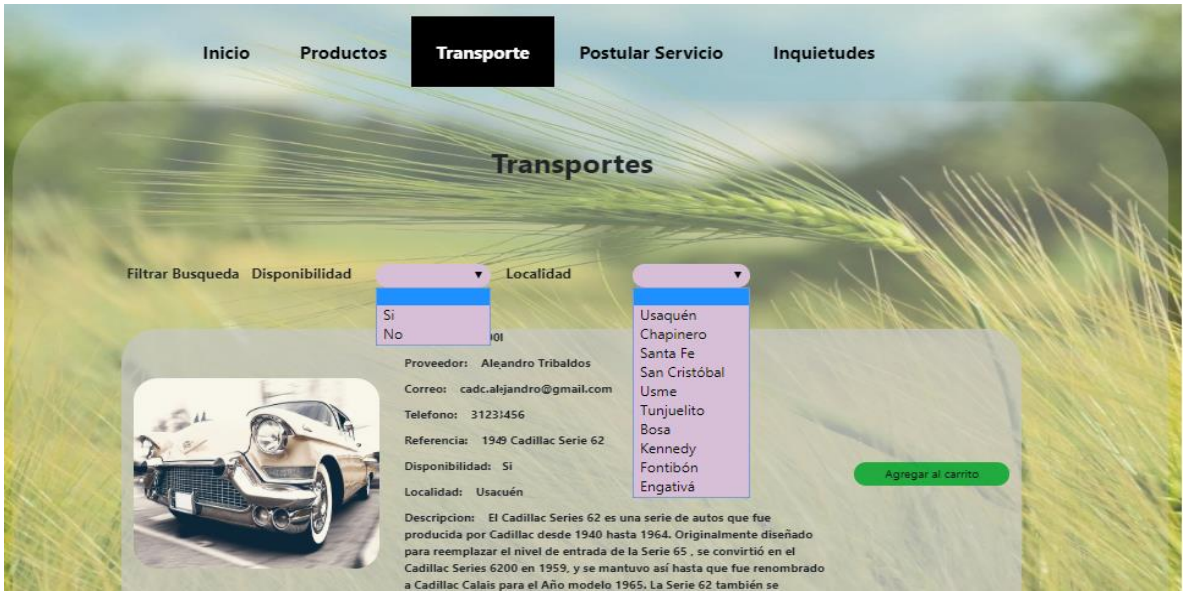


Ilustración 84: Diseño del Sprint 8.



Ilustración 85: Diseño del Sprint 8.



Ilustración 86: Diseño del Sprint 8.

3.8.4 Proceso de desarrollo de software del Sprint 8

```
<!--aqui van los filtros de los productos-->
<div class="filtros">
  <ul><table class="Tpt">
      <tr>
        <td><label>Filtrar Busqueda</label></td>
        <td><label>Disponibilidad</label></td>
        <td><select name="filtroDisp" [(ngModel)]="filtroDisp">
        <option value=""></option>
        <option value="Si">Si</option>
        <option value="No">No</option>
        </select></td>
      <td><label>Tipo</label></td>
      <td><select name="filtroTip" [(ngModel)]="filtroTip">
        <option value=""></option>
        <option value="Fruta">Fruta</option>
        <option value="Vegetal">Vegetal</option>
      </select></td>
  </tr></table></ul>
</div>
```

Ilustración 87: Desarrollo del Sprint 8.

```
<!--aqui van los filtros de los transportes-->
<div class="filtros">
  <ul><table class="Tpt">
      <tr>
        <td><label>Filtrar Busqueda</label></td>
        <td><label>Disponibilidad</label></td>
        <td><select name="filtroDisp" [(ngModel)]="filtroDisp">
        <option value=""></option>
        <option value="Si">Si</option>
        <option value="No">No</option>
        </select></td>
      <td><label>Localidad</label></td>
      <td><select name="filtroLocal" [(ngModel)]="filtroLocal">
        <option value=""></option>
        <option value="Usaquén">Usaquén</option>
        <option value="Chapinero">Chapinero</option>
        <option value="Santa Fe">Santa Fe</option>
        <option value="San Cristóbal">San Cristóbal</option>
        <option value="Usme">Usme</option>
        <option value="Tunjuelito">Tunjuelito</option>
        <option value="Bosa">Bosa</option>
        <option value="Kennedy">Kennedy</option>
        <option value="Fontibón">Fontibón</option>
        <option value="Engativá">Engativá</option>
      </select></td>
  </tr></table></ul>
</div>
```

Ilustración 88: Desarrollo del Sprint 8.

```
<!--aqui van los filtros de postular productos-->
<div class="filtros">
  <ul><table class="Tpt">
      <tr>
        <td><label>Filtrar Busqueda</label></td>
        <td><label>Estado</label></td>
        <td><select name="filtroEst" [(ngModel)]="filtroEst">
        <option value=""></option>
        <option value="Por Confirmar">Por Confirmar</option>
        <option value="Aprobado">Aprobado</option>
        <option value="Denegado">Denegado</option>
        </select></td>
        <td><label>Disponibilidad</label></td>
        <td><select name="filtroDisp" [(ngModel)]="filtroDisp">
        <option value=""></option>
        <option value="Si">Si</option>
        <option value="No">No</option>
        </select></td>
        <td><label>Tipo</label></td>
        <td><select name="filtroTip" [(ngModel)]="filtroTip">
        <option value=""></option>
        <option value="Fruta">Fruta</option>
        <option value="Vegetal">Vegetal</option>
        </select></td>
  </tr></table></ul>
</div>
```

Ilustración 89: Desarrollo del Sprint 8.

```
<!--aqui van los filtros de postular transportes-->
<div class="filtros">
  <ul><table class="Tpt">
      <tr>
        <td><label>Filtrar Busqueda</label></td>
        <td><label>Estado</label></td>
        <td><select name="filtroEst" [(ngModel)]="filtroEst">
        <option value=""></option>
        <option value="Por Confirmar">Por Confirmar</option>
        <option value="Aprobado">Aprobado</option>
        <option value="Denegado">Denegado</option>
        </select></td>
        <td><label>Disponibilidad</label></td>
        <td><select name="filtroDisp" [(ngModel)]="filtroDisp">
        <option value=""></option>
        <option value="Si">Si</option>
        <option value="No">No</option>
        </select></td>
        <td><label>Localidad</label></td>
        <td><select name="filtroLocal" [(ngModel)]="filtroLocal">
        <option value=""></option>
        <option value="Usaquén">Usaquén</option>
        <option value="Chapinero">Chapinero</option>
        <option value="Santa Fe">Santa Fe</option>
        <option value="San Cristóbal">San Cristóbal</option>
        <option value="Usme">Usme</option>
        <option value="Tunjuelito">Tunjuelito</option>
        <option value="Bosa">Bosa</option>
        <option value="Kennedy">Kennedy</option>
        <option value="Fontibón">Fontibón</option>
        <option value="Engativá">Engativá</option>
    </select></td>
    </tr></table></ul>
  </div>
```

Ilustración 90: Desarrollo del Sprint 8.

3.8.5 Diagrama de contexto del Sprint 8

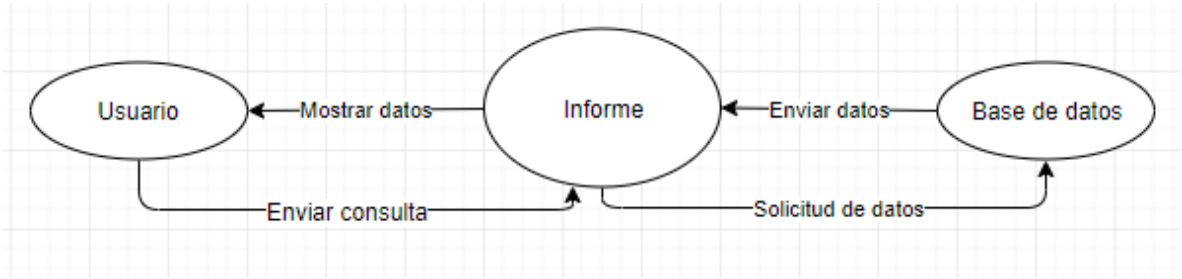


Ilustración 91: Diagrama del Sprint 8.

3.8.6 Prueba del Sprint 8

Sprint 8	Informe	
Descripción de la prueba	Se realizan informes al consultar las BD por medio de filtros visibles en la página para su futuro uso como consultas de alguna de ella.	
Herramientas	Opera y FireBase	
Numero	Prueba	Estado
	Las consultas de algunos productos, transportes, productos postulados y transportes postulados para verificar la información de estas BD o la acción para cual requirieron utilizar estos servicios.	OK

Tabla 32: Prueba del Sprint 8.

4. FASE 4: IMPLEMENTACION

4.1 DIAGRAMA DE PAQUETES

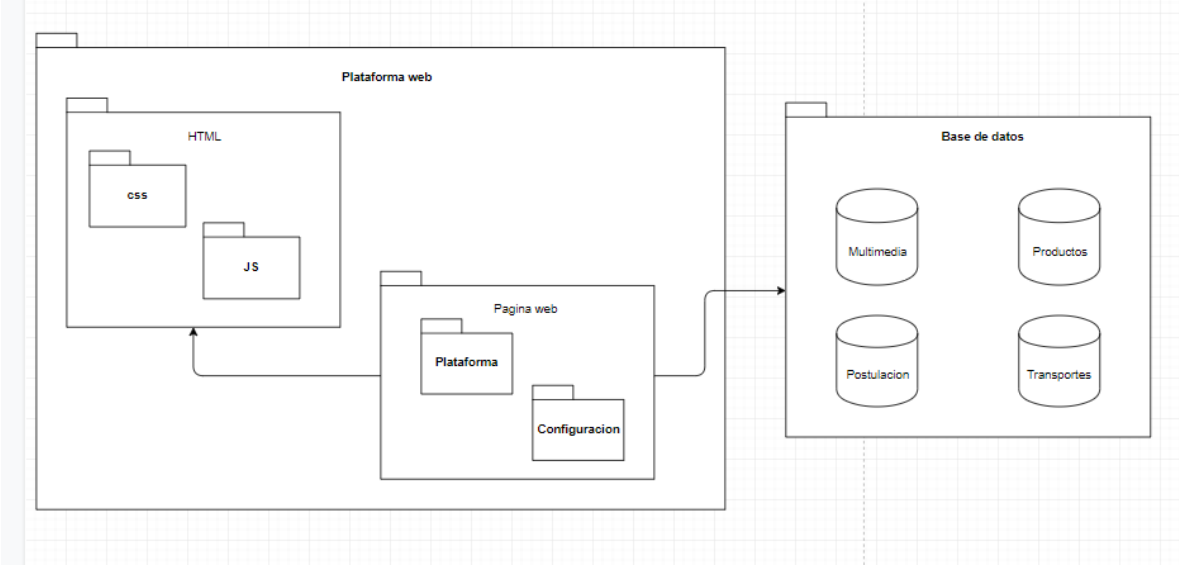


Ilustración 92: Diagrama de paquetes.

4.2 PRUEBAS EN DIFERENTES RESOLUCIONES Y NAVEGADORES

Sprint	Dispositivo	Características	Navegador web	Conexión	Tiempo (seg)	Visualización
1	Portátil (DELL)	Intel® Core™ i5, RAM 4 GB, Windows 10 home, resolución (1366 x 768)	Google Chrome	Wifi	Aprox. 5	Optima
1	Samsung Galaxy J6	Android 9, One UI 1.0 Knox 3.3, resolución (1.480 x 720)	Chrome	Datos Móviles Claro 4G	Aprox. 6	Optima
1	PC de escritorio (Hp)	Intel Core 2 Duo, RAM 4, Windows 7,	Opera	Cable utp-8	Aprox. 8	Optima

		resolución (1280 x 1024)				
2	Portátil (DELL)	Intel® Core™ i5, RAM 4 GB, Windows 10 home, resolución (1366 x 768)	Google Chrome	Wifi	Aprox. 7	Optima
2	Samsung Galaxy J6	Android 9, One UI 1.0 Knox 3.3, resolución (1.480 x 720)	Chrome	Datos Móviles Claro 4G	Aprox. 8	Optima
2	PC de escritorio (Hp)	Intel Core 2 Duo, RAM 4, Windows 7, resolución (1280 x 1024)	Opera	Cable utp-8	Aprox. 5	Optima
3	Portátil (DELL)	Intel® Core™ i5, RAM 4 GB, Windows 10 home, resolución (1366 x 768)	Google Chrome	Wifi	Aprox. 10	Optima
3	Samsung Galaxy J6	Android 9, One UI 1.0 Knox 3.3, resolución (1.480 x 720)	Chrome	Datos Móviles Claro 4G	Aprox. 9	Optima
3	PC de escritorio (Hp)	Intel Core 2 Duo, RAM 4, Windows 7, resolución (1280 x 1024)	Opera	Cable utp-8	Aprox. 14	Optima
4	Portátil (DELL)	Intel® Core™ i5, RAM 4 GB, Windows 10 home, resolución (1366 x 768)	Google Chrome	Wifi	Aprox. 10	Optima

4	Samsung Galaxy J6	Android 9, One UI 1.0 Knox 3.3, resolución (1.480 x 720)	Chrome	Datos Móviles Claro 4G	Aprox. 9	Optima
4	PC de escritorio (Hp)	Intel Core 2 Duo, RAM 4, Windows 7, resolución (1280 x 1024)	Opera	Cable utp-8	Aprox. 14	Optima
5	Portátil (DELL)	Intel® Core™ i5, RAM 4 GB, Windows 10 home, resolución (1366 x 768)	Google Chrome	Wifi	Aprox. 5	Optima
5	Samsung Galaxy J6	Android 9, One UI 1.0 Knox 3.3, resolución (1.480 x 720)	Chrome	Datos Móviles Claro 4G	Aprox. 3	Optima
5	PC de escritorio (Hp)	Intel Core 2 Duo, RAM 4, Windows 7, resolución (1280 x 1024)	Opera	Cable utp-8	Aprox. 4	Optima
6	Portátil (DELL)	Intel® Core™ i5, RAM 4 GB, Windows 10 home, resolución (1366 x 768)	Google Chrome	Wifi	Aprox. 5	Optima
6	Samsung Galaxy J6	Android 9, One UI 1.0 Knox 3.3, resolución (1.480 x 720)	Chrome	Datos Móviles Claro 4G	Aprox. 3	Optima
6	PC de escritorio (Hp)	Intel Core 2 Duo, RAM 4, Windows 7,	Opera	Cable utp-8	Aprox. 4	Optima

		resolución (1280 x 1024)				
7	Portátil (DELL)	Intel® Core™ i5, RAM 4 GB, Windows 10 home, resolución (1366 x 768)	Google Chrome	Wifi	Aprox. 5	Optima
7	Samsung Galaxy J6	Android 9, One UI 1.0 Knox 3.3, resolución (1.480 x 720)	Chrome	Datos Móviles Claro 4G	Aprox. 3	Optima
7	PC de escritorio (Hp)	Intel Core 2 Duo, RAM 4, Windows 7, resolución (1280 x 1024)	Opera	Cable utp-8	Aprox. 3	Optima
8	Portátil (DELL)	Intel® Core™ i5, RAM 4 GB, Windows 10 home, resolución (1366 x 768)	Google Chrome	Wifi	Aprox. 8	Optima
8	Samsung Galaxy J6	Android 9, One UI 1.0 Knox 3.3, resolución (1.480 x 720)	Chrome	Datos Móviles Claro 4G	Aprox. 8	Optima
8	PC de escritorio (Hp)	Intel Core 2 Duo, RAM 4, Windows 7, resolución (1280 x 1024)	Opera	Cable utp-8	Aprox. 10	Optima

Tabla 33: Pruebas en diferentes resoluciones y navegadores.

5. CONCLUSIONES

Este proyecto se realizó con el fin de brindarle la oportunidad a las personas de las zonas periféricas de Bogotá que trabajan en la rama de la agricultura de poder competir con las grandes empresas de productos agrícolas, por medio de una plataforma web que facilite la venta y compra de productos agrícolas directamente y no por terceros, generando más ganancias para las personas que cultivan el producto agrícola.

En cuanto la plataforma se diseñó de una manera amigable para los usuarios de fácil acceso, que permita validar los usuarios de manera más cómoda sin necesidad de registrarse, mostrar los servicios que se ofrecen y los productos disponibles para cada uno de los usuarios. También se generó un espacio que permita a los administradores resolver cualquier inconveniente o duda de los clientes y una mejor comunicación entre los mismos para poder ofrecer un mejor servicio.

La tecnología que se usó para su realización se escogió con el fin de que no consumiera muchos recursos tanto económicos como técnicos, para la edición del código se utilizó Visual Studio Code y las herramientas como angular, css, JavaScript, node.js y para el almacenamiento de los datos un motor no relacional como lo es FireBase.

6. RECOMENDACIONES

Dar a conocer el proceso de cultivo de los productos agrícolas que se encuentran en la plataforma para que tenga más información sobre el producto que está cotizando el cliente generando así más confianza a la hora de comercializarlos.

Ofrecer una gran variedad de datos sobre los productos para llegar a dividirlos por clima, época de mayor producción, entre otros, para ampliar la información que se ofrece de los productos y expandir su catálogo.

Ampliar los diferentes servicios que ofrece la pagina para volver los procesos agiles, seguros y confiables, por ejemplo el módulo de facturación.

7. BIBLIOGRAFIA

Perletti, J; Botero, J; Oviedo, S; Forero, F; Higuera, S; Correa, M; Garcia, J. (2017). Política comercial agrícola: nivel, costos y efectos de la protección en Colombia. Septiembre 21, 2018, de Fedesarrollo – Universidad EAFIT Sitio web: https://www.repository.fedesarrollo.org.co/bitstream/handle/11445/3443/Repor_Agosto_2017_Perfetti_et_al.pdf?sequence=3

Mendoza, G. (2002). Diagnóstico del mercadeo agrícola y agroindustrial en Colombia. Bogotá: U. Jorge Tadeo Lozano.

Secretaria Distrital de Ambiente. (2018). Sistema Agropecuario Distrital (SISADI). Septiembre 30, 2018, de Secretaria Distrital de Ambiente Sitio web: http://ambientebogota.gov.co/bg/sistema-agropecuario-distrital-sisadi

Secretaria de ciencia y tecnología. (2017). Proyectos de la Secretaría de Ciencia, Tecnología e Innovación. Septiembre 30, 2018, de Secretaria de ciencia y tecnología Sitio web: http://www.cundinamarca.gov.co/Home/SecretariasEntidades.gc/SecdeCienyTec/SecdeCienyTecDespliegue/aspropgramasyproyectos_contenidos/proyectos

Observatorio de desarrollo económico. (2011). Economía campesina, soberanía y seguridad alimentarias. La experiencia de Mercados Campesinos en Bogotá y la región central de Colombia. Septiembre 30, 2018, de Observatorio de desarrollo económico Sitio web: http://observatorio.desarrolloeconomico.gov.co/base/descargas/public/mercadoscampesinos.pdf

CIUDAD & TERRITORIO. (2016). Pasquilla, una vereda de Bogotá, presionada por crecimiento urbano. Mayo 23, 2016, de Agencia de Noticias UN Sitio web: http://agenciadenoticias.unal.edu.co/detalle/article/pasquilla-una-vereda-de-bogota-presionada-por-crecimiento-urbano.html

Centro Regional de Estudios Regionales Cafeteros y Empresariales (CRECE). MISIÓN PARA LA TRANSFORMACIÓN DEL CAMPO: Propuesta para Desarrollar un Modelo eficiente de Comercialización y Distribución de Productos. Bogotá: diciembre de 2014, 61p.

Garcia A. «Medium. ¿Que es Angular?,» 16 Febrero, 2019. [En línea]. Disponible en: https://medium.com/ngesyfirebase/qu%C3%A9-es-angular-877c99a607f4. [Último acceso: 15 Marzo 2019].

Bbvaopen4u, Herramientas básicas para desarrolladores de JavaScript. 11 septiembre, 2015 Disponible en:

https://bbvaopen4u.com/es/actualidad/herramientas-basicas-para-desarrolladores-en-javascript. [Último acceso: 12 Marzo 2019].

Firebase. (2019) Estructura tu base de datos | Firebase Realtime Database | Base de fuego. [en línea]. Disponible en: https://firebase.google.com/docs/database/web/structure-data?hl=es-419. [revisado 25 de marzo de 2018].

Universidad de Murcia, Introducción a HTML y CSS. Desarrollo de Aplicaciones Web. Febrero 11, 2019. [en línea]. Disponible en: https://www.um.es/docencia/barzana/DAWEB/2017-18/daweb-tema-1-introduccion-html-css.html. [Último acceso: 27 febrero 2019].

MDN web docs. Cómo funciona CSS. marzo 18, 2019. [en línea]. Disponible en: https://developer.mozilla.org/es/docs/Learn/CSS/Introduction_to_CSS/Como_funciona_CSS. [Último acceso: 27 marzo 2019].

php. ¿Qué es PHP? [en línea]. Disponible en: https://www.php.net/manual/es/intro-whatis.php. [Último acceso: 16 marzo 2019].

TRIGAS GALLEGO, Manuel. GESTIÓN DE PROYECTOS INFORMÁTICOS. Septiembre 21, 2018, de GESTIÓN DE PROYECTOS INFORMÁTICOS Sitio web: http://openaccess.uoc.edu/webapps/o2/bitstream/10609/17885/1/mtrigasTFC0612memoria.pdf

Yesica Lizeth. ¿Qué es Scrum y los roles en Scrum? Septiembre 21, 2018, de Platzi Sitio web: https://platzi.com/blog/que-es-scrum-y-los-roles-en-scrum

GARCIA DE LEON, Alicia. (2002). ETAPAS DE LA CREACIÓN DE UN SITIO WEB. Lima: Biblios.

EGUILUZ PEREZ, Javier. Introducción a XHTML. Creative Commons, 17 de diciembre de 2008, 168p.

Secretaria distrital de ambiente. (2018). Parque-minero-industrial. Septiembre 26, 2018, de Secretaria distrital de ambiente. Sitio web: http://ambientebogota.gov.co/nb/parque-minero-industrial

Alcaldía Local de Ciudad Bolívar. (2013-2016). PLAN AMBIENTAL LOCAL LOCALIDAD 19 CIUDAD BOLÍVAR. 2017, de Alcaldía Local de Ciudad Bolívar Sitio web: http://www.ambientebogota.gov.co/documents/10157/2883179/PAL+CBol%C3%ADvar+2013-2016.pdf

Cámara de comercio de Bogotá. (2016). La cadena de comercialización de alimentos en Colombia. Julio, 2016, de Cámara de comercio de Bogotá Sitio web: https://www.ccb.org.co/Sala-de-prensa/Noticias-sector-Agricola-y-Agroindustrial/Noticias-2016/La-cadena-de-comercializacion-de-alimentos-en-Colombia

Agronet. (2016). Agronet. Septiembre 21, 2018, de MINAGRICULTURA Sitio web: http://www.agronet.gov.co/Paginas/default.aspx

Comproagro. (2018). Comproagro. Septiembre 30, 2018, de Comproagro Sitio web: https://www.comproagro.com/

MINAGRICULTURA. (2018). Agrosavia. Septiembre 30, 2018, de MINAGRICULTURA Sitio web: http://www.corpoica.org.co/noticias/generales/somos-agrosavia