UNIVERSIDAD DE ANTIOQUIA

PLATAFORMA PARA SEGUIMIENTO DE SERVICIO O PROCESOS DE MANUFACTURA DE UN PRODUCTO

Autor(es)

David Alejandro Marin Alzate

Universidad de Antioquia

Facultad de ingeniería, Departamento de Sistemas

Medellín, Colombia

PLATAFORMA PARA SEGUIMIENTO DE SERVICIO O PROCESOS DE MANUFACTURA DE UN PRODUCTO

David Alejandro Marin Alzate

Trabajo de grado presentado como requisito para optar al título de:

Ingeniero de sistemas

Asesor

Fernando Eliécer Ávila Berrío

Universidad de Antioquia

Facultad de Ingeniería, Departamento de Sistemas

Medellín, Colombia

2021.

Contenido

Lista de Figuras

Lista de Tablas

PLATAFORMA PARA SEGUIMIENTO DE SERVICIOS O PROCESOS DE MANUFACTURA DE UN PRODUCTO.

1. Resumen

Mantenerse informado del estado en el que se encuentra el producto que un cliente a través de internet, a veces se vuelve engorroso, y depende de la comunicación con el proveedor al que se le realizó la compra para realizar el seguimiento u obtener el estado en el que se encuentra el pedido.

En este trabajo se presenta la construcción de una plataforma web para hacer que esta brecha sea reducida y el cliente pueda realizar el seguimiento de su producto desde cualquier parte del mundo en la cual posea internet y a cualquier hora. Para ello, se creó un back-end que es un microservicio reactivo con Java 8 y Spring boot, además, un front-end usando Ionic, esto con el fin de hacer un desarrollo para ser exportado hacia diferentes plataformas como Android, iOS y Windows con un solo desarrollo, conectado el back-end y el front-end a través de una API REST.

Se realizaron pruebas de rendimiento, tanto individuales como conjuntas, a diferentes servicios usando JMeter, en un servidor con bajas especificaciones expuesto a internet. Se encontró que la concurrencia soportada por la aplicación es de 38 peticiones cada segundo, cuando se realizaron las pruebas conjuntas en todos los servicios analizados. Teniendo en cuenta que este desarrollo está en su fase inicial y se tiene un mínimo producto viable, el valor de las peticiones por segundo da una luz de creación de una granja de microservicios para tener un mayor rendimiento y soportar mayor concurrencia.

2. Introducción

La creciente ola de las telecomunicaciones en Colombia, que en el primer trimestre del año 2020 ascendió a los 7.1 millones de accesos fijos a internet [1], y la pandemia por SARS-CoV-2 (Covid-19) incentivaron el fortalecimiento de la virtualidad con miras a cumplir las exigencias laborales y educativas. Como consecuencia, se incrementaron de forma exponencial las compras por internet y el comercio virtual [2], además, se ha fomentado la creación de una cultura cada vez más digital, ganándose la confianza de usuarios que en un principio se abstenían de hacer compras en línea. Sin embargo, una de las quejas que más expresan los usuarios es la carencia de un sistema eficiente de seguimiento de pedidos, que deja a algunos usuarios insatisfechos por no tener en sus manos el estado en tiempo real de su pedido y su tiempo de entrega [2]. El rastreo de diferentes productos se ha convertido en una necesidad para los clientes. Empresas de mensajería y paquetería, almacenes de grandes superficies y tiendas virtuales han implementado el seguimiento para mantener informados a sus usuarios del estado de la compra y entrega de su producto, es decir, un comprador puede conocer si su pedido se encuentra en despacho, en ruta, si ha llegado a la ciudad de destino e, incluso, si ya ha sido entregado [3].

Algunos software que existen en el mercado permiten hacer seguimiento, ejecución y optimización especialmente de procesos de manufactura a gran escala, tales como FactoryFour [4], cuyo principal objetivo es hacer el seguimiento interno del pedido y realizar la ejecución de la producción en tiempo real, sin embargo, no se revela a sus clientes dicha información y esto conlleva a dejar al cliente con la expectativa del estado del pedido. Otra opción que existe, SafetyChain [5], promete mejorar el rendimiento, el monitoreo en tiempo real y también maximizar la productividad, no obstante, este se enfoca principalmente en la industria de alimentos y bebidas. Las plataformas de software mencionadas anteriormente tienen la

particularidad de enfocarse en compañías robustas que requieren de elevado control y monitoreo por su alta demanda, y no en empresas pequeñas dado que estás no cuentan con el músculo financiero para adquirir una solución de este tipo y no se enfocan en lo que realmente les da valor como el cliente.

3. Objetivos

3.1. Objetivo general:

Construir una herramienta basada en la web de seguimiento de pedidos en tiempo real, de tal manera que se reduzca la brecha de comunicación entre el cliente y la empresa acerca del estado de su pedido.

3.2. Objetivos específicos:

- Levantar los requerimientos del software enfocados en los usuarios finales.
- Diseñar una estructura de datos que soporte la transacción y la concurrencia a la que se va a someter el sistema de información.
- Desarrollar un entorno de control que tenga la lógica de negocio y se encargue del CRUD en la estructura de datos.
- Implementar una interfaz de usuario final que permita la administración y seguimiento de los estados, por los cuales pasa la manufactura de un producto o servicio.
- Integración, despliegue y pruebas de las diferentes características del software con sus respectivas pruebas y desplegarlo.

4. Marco Teórico

Las aplicaciones web tienen, generalmente, dos componentes muy importantes, el back-end y el front-end, los cuales son integrados o comunicados a través de una API REST, como se aprecia en la Figura 1.

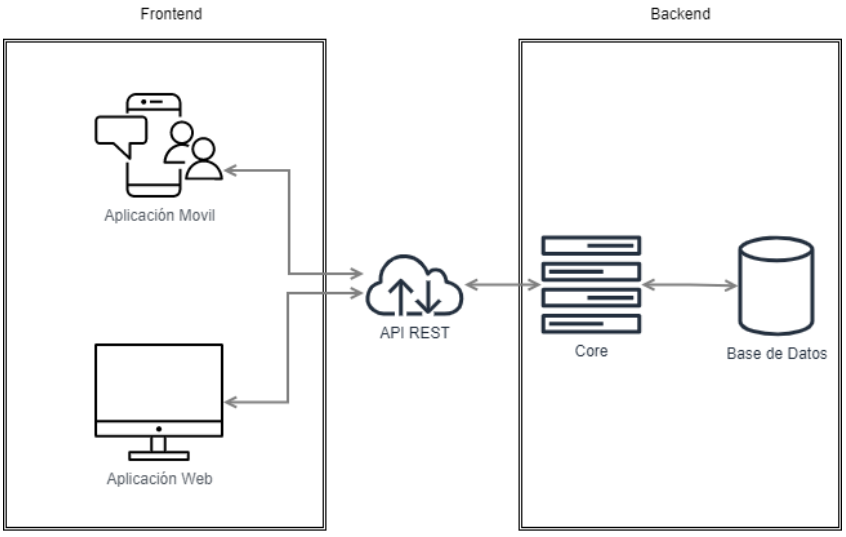


Figura 1. Diagrama de una aplicación web

El componente de front-end o cliente, es la interfaz o vista con la que los usuarios finales de una aplicación pueden interactuar, por ejemplo, una página web, una aplicación móvil, entre otras [6]; una de tantas herramientas que se usa para la construcción del front-end es Ionic, que es en un conjunto de herramientas (toolkit) para aplicaciones móviles híbridas que usa la vista web (Web View) de un teléfono inteligente o tableta para mostrar los componentes que conforman la funcionalidad de la aplicación, envolviéndolo en una aplicación instalable en un dispositivo móvil [7]. Este toolkit usa múltiples librerías o frameworks como Veu Js, React Js, Angular, entre otros; estos usan una tecnología conocida como HTML (Hyper Text Markup Language) siendo este el lenguaje que se interpreta en los navegadores para preparar la visual y mostrarla [8].

Solo con HTML no es suficiente para que una aplicación web funcione, para esto se acompaña de un lenguaje de programación llamado Javascript (Js) que maneja toda la parte lógica del comportamiento de los componentes o datos. Este lenguaje tiene la característica de que puede ser interpretado o compilado según la necesidad. Angular, particularmente, se codifica con Typescript, que en esencia es Javascript pero se le fue aplicado una capa para hacerlo más parecido a un lenguaje en el que sus variables tiene tipos de datos específicos y hacerlo más parecido a un lenguaje Orientado a objetos, pero sin dejar de lado las características de Javascript (funciones, métodos, clases, entre otros).

El front-end o cliente se comunica con el back-end por medio de servicios REST (Representational State Transfer). Básicamente, estos servicios proporcionan una interfaz o canal de comunicación entre el componente de front-end y back-end usando una estructura de datos conocida como JSON, el cual representa un objeto con sus atributos y valores separados por comas (vea la Figura 2).

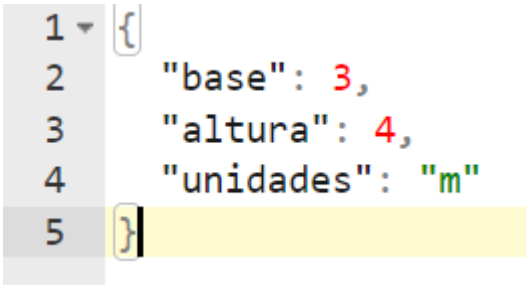


Figura 2. Ejemplo estructura de datos JSON, los atributos son: base, altura, unidades y los valores respectivamente son 3, 4 y m.

El back-end es el componente que maneja todas las operaciones en la base de datos (almacenar, consultar, modificar y eliminar) y es el encargado de velar por que las reglas de negocio se cumplan. Este se integra con dispositivos móviles o plataformas web a través de servicios REST [8]. Existen

muchos lenguajes con los cuales se puede hacer la construcción de un back-end, como NodeJs, PHP, Python, Java, etc. Hablaremos específicamente de Java, que es un lenguaje de programación orientado a objetos (OOP), comúnmente usado para la creación de aplicaciones empresariales de amplio alcance o de uso doméstico [9].

Pero ¿qué es un lenguaje orientado a objetos? Un lenguaje orientado a objetos es un paradigma de programación que intenta abstraer el mundo real en objetos ficticios. Por ejemplo, en la vida real todos somos personas con atributos, tales como nombre, edad, estatura, peso, etc [10]; en la programación orientada a objetos podemos modelar una persona creando una clase que se llama persona y esta internamente tiene unos atributos que guardan la información de cada persona, la cual puede ser accedida por los diferentes métodos de la aplicación (Figura 3).

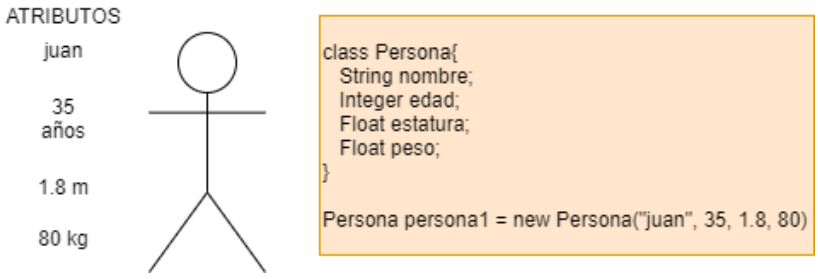


Figura 3. Ejemplo de una persona abstraída en una clase de POO, al lado izquierdo tenemos la realidad y al lado derecho como se defino un objeto de tipo persona con los atributos de Juan.

También existen algunos conceptos como el encapsulamiento, el polimorfismo, la herencia, las cuales no abordaremos debido a que es un tema extenso y no es nuestro enfoque en este trabajo.

Para Java existen múltiples marcos de trabajo, algunos de los más usados para aplicaciones empresariales son EJB (Enterprise Java Bean) y Spring Framework, según el reporte de tecnologías java 2020 [11]. Spring

Framework es un conjunto de librerías tales como spring security que abstrae la complejidad de implementarle seguridad a un sistema y spring data que abstrae la conexión con la base de datos entregando objetos con la información a partir de datos almacenado en un motor de base de datos. Entre otras cualidades que ofrece spring es la facilidad de adicionar componentes y hacer que la aplicación no tenga problemas de memoria usando un concept llamado inyección de dependencias, que básicamente es que se crea un objeto cuando la aplicación está subiendo y cada vez que se necesite se llama este objeto sin necesidad de crear uno nuevo [12].

Spring da la posibilidad de trabajar con programación reactiva, este es un paradigma que se centra en el trabajo de flujos de datos asíncronos, dando la posibilidad de aumentar el rendimiento de una aplicación por medio de hilos [13], para este propósito existen dos librerías muy usadas que son java rx y project reactor.

En el back-end, el almacenamiento de la información es una parte fundamental y para esto existen las base de datos, que son un conjunto de estructuras de datos organizadas almacenados en un servidor al que se le puede aplicar reglas para realizar consultas de esta información a través de un lenguaje que es interpretado por el motor de esta [14]. Existen dos tipos las bases de datos que son: relacionales y no relacionales. Para las relacionales tenemos tecnologías como MySql, MariaDb, OracleDb, SqlServer, entre otras y para las no relacionales tenemos MongoDb, DynamoDb, Casandra, entre otras.

Un back-end puede estar compuesto de múltiples microservicios o un solo microservicio, todo depende de la complejidad de la aplicación. Un microservicio es un estilo de arquitectura de software que se divide en pequeñas partes independientes. Un conjunto de estos se usa para llevar a cabo una tarea específica [15]. Regularmente los microservicios son

desplegados en contenedores de docker por su flexibilidad para crear más de una instancia del mismo. Docker es un software que permite empaquetar una aplicación junto con todo lo necesario para que funcione, este empaquetamiento se conoce como "imagen", cuando esta imagen se despliega, se conoce como contenedor de docker [16].

En este trabajo se va a llamar usuario a la persona que hace parte de la empresa y realiza configuraciones sobre la plataforma construida, y cliente es la persona que adquiere servicios a la empresa que adopta esta plataforma.

5. Metodología

A continuación, se describe la metodología que se llevó a cabo para el desarrollo de la solución propuesta:

5.1. Levantar los requerimientos del software enfocado en los usuarios finales.

Se diseñó una encuesta de nueve preguntas con el objetivo de conocer el interés de compradores y empresas de contar con una aplicación que entregue en tiempo real el estado de un pedido. Además, con esta encuesta se quería conocer la información más relevante de los clientes de una empresa que adopte esta solución.

Estas preguntas fueron diseñadas para dos tipos de posiciones, la primera es una persona que realiza compras por internet y la segunda es como empresario. En el

ANEXOS

Anexo 1 se muestran las preguntas y sus opciones de respuesta.

Esta encuesta se implementó en google forms y se compartió con 21 personas. La ventaja encontrada en la aplicación de una encuesta fue la

asincronía con la que las personas que participaban en ella podían responderla.

Se analizaron los datos para el levantamiento de los requisitos para la construcción del sistema con la información recolectada con la encuesta. Los resultados se muestran a continuación, El total de personas encuestadas fueron 21.

En la Figura 4 se observa la ocupación de las personas encuestadas, donde en un 81 % se encuentran empleados, que podrían eventualmente actuar como clientes, al igual que los estudiantes. En un 28.6 % se encontraron empresarios e independientes.

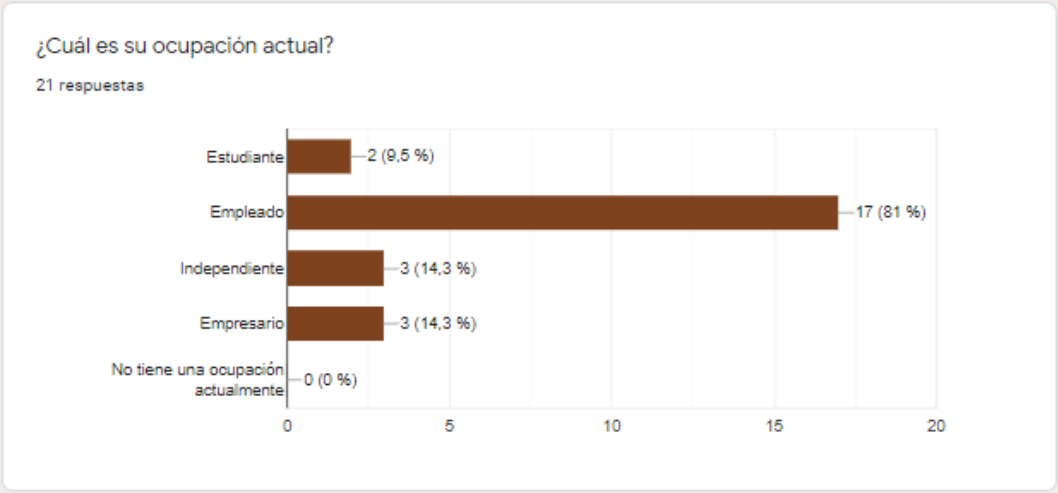


Figura 4. Pregunta 2

En la Figura 5 se listan las actividades económicas que realizan los encuestados, donde se observa una gran diversidad.

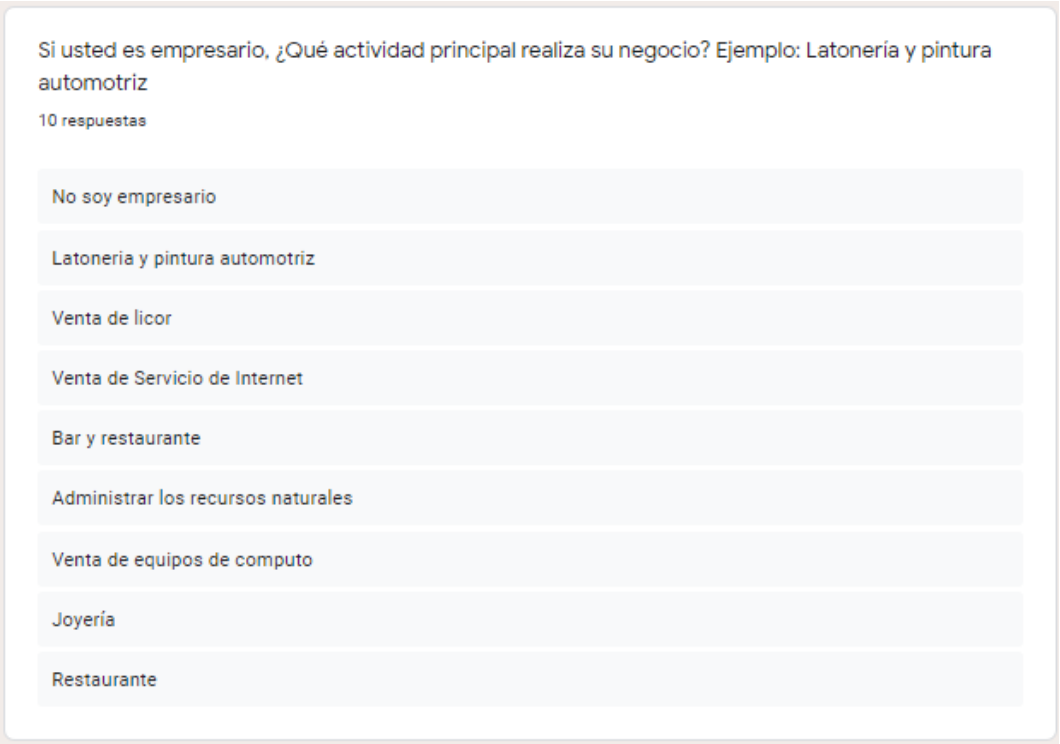


Figura 5. Pregunta 3

De los encuestados, el 81 % realiza compras o ventas frecuentes a través de internet (Figura 6). Este porcentaje podría verse beneficiado con la implementación de la aplicación. Todos los encuestados consideran que conocer el estado de su producto o pedido en etapa de manufactura tiene importancia, como se observa en la Figura 7, además, en la Figura 8 se muestra también que todos los encuestados consideran importante tener seguimiento de su producto o pedido en la etapa de entrega. Esto evidencia una tendencia entre los encuestados a querer conocer información del ciclo que sigue un producto, desde su orden hasta su entrega, como se muestra también en la Figura 11, donde aproximadamente el 95 % manifiesta su interés en adoptar la solución que ofrece esta aplicación ya que desean visualizar el estado de su producto o pedido desde una página web. Además, el 37.5 % de los empresarios reciben preguntas de sus clientes por el estado de su orden (Figura 9) y

manifiestan querer conocer, sobre todo, nombre, teléfono celular y dirección de sus clientes (Figura 10).

Figura 6. Pregunta 4

Figura 7. Pregunta 5

Figura 8. Pregunta 6

Figura 9. Pregunta 7

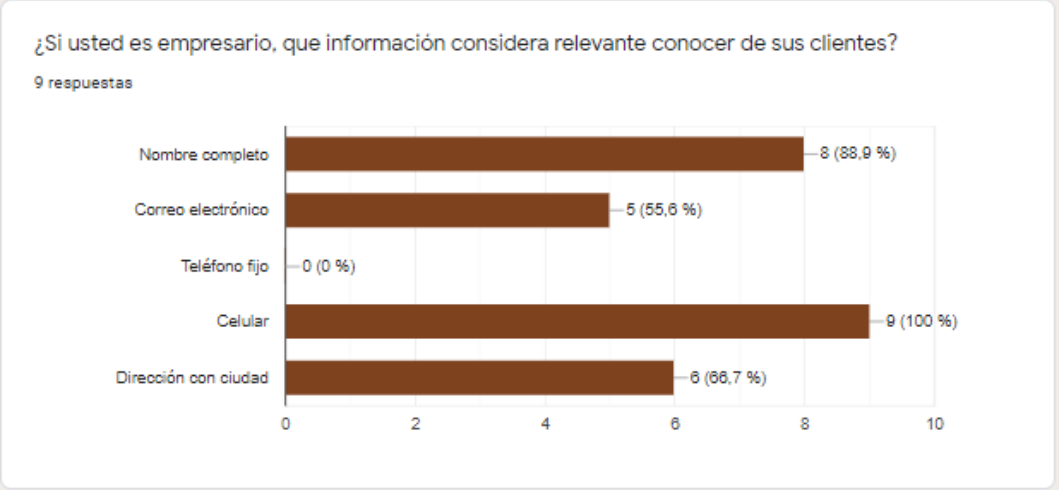


Figura 10. Pregunta 8

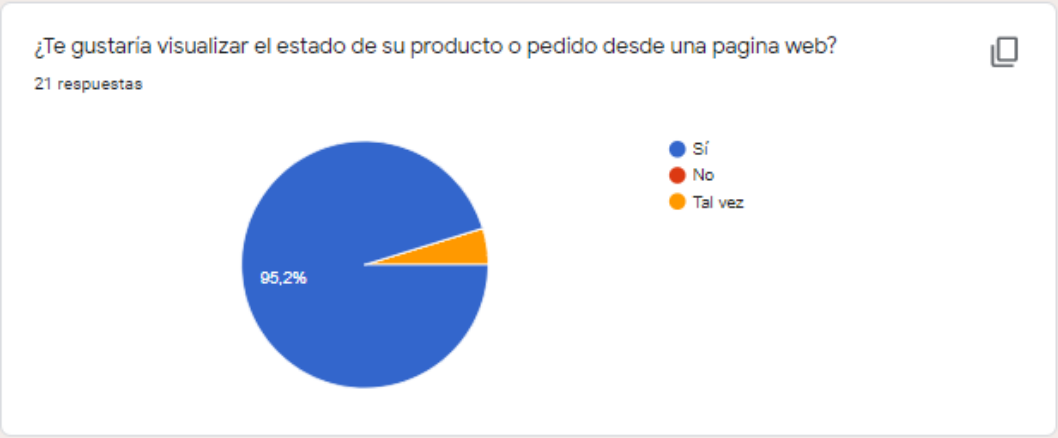


Figura 11. Pregunta 9

5.2. Estructura de datos.

Se construyó el modelo entidad relación (Figura 12), el cual soporta múltiples empresas, esto implica que en la misma base de datos se guarda información de diferentes empresas y una empresa no puede acceder a los datos de otra. Además, un usuario puede pertenecer a diferentes empresas, pero sólo se muestra la información o datos de la empresa con la que él inicia la sesión, en otras palabras, no se mezcla la información porque este usuario pertenezca a diferentes empresas. Todo el modelo de negocio de rastreo es totalmente configurable por cada empresa.

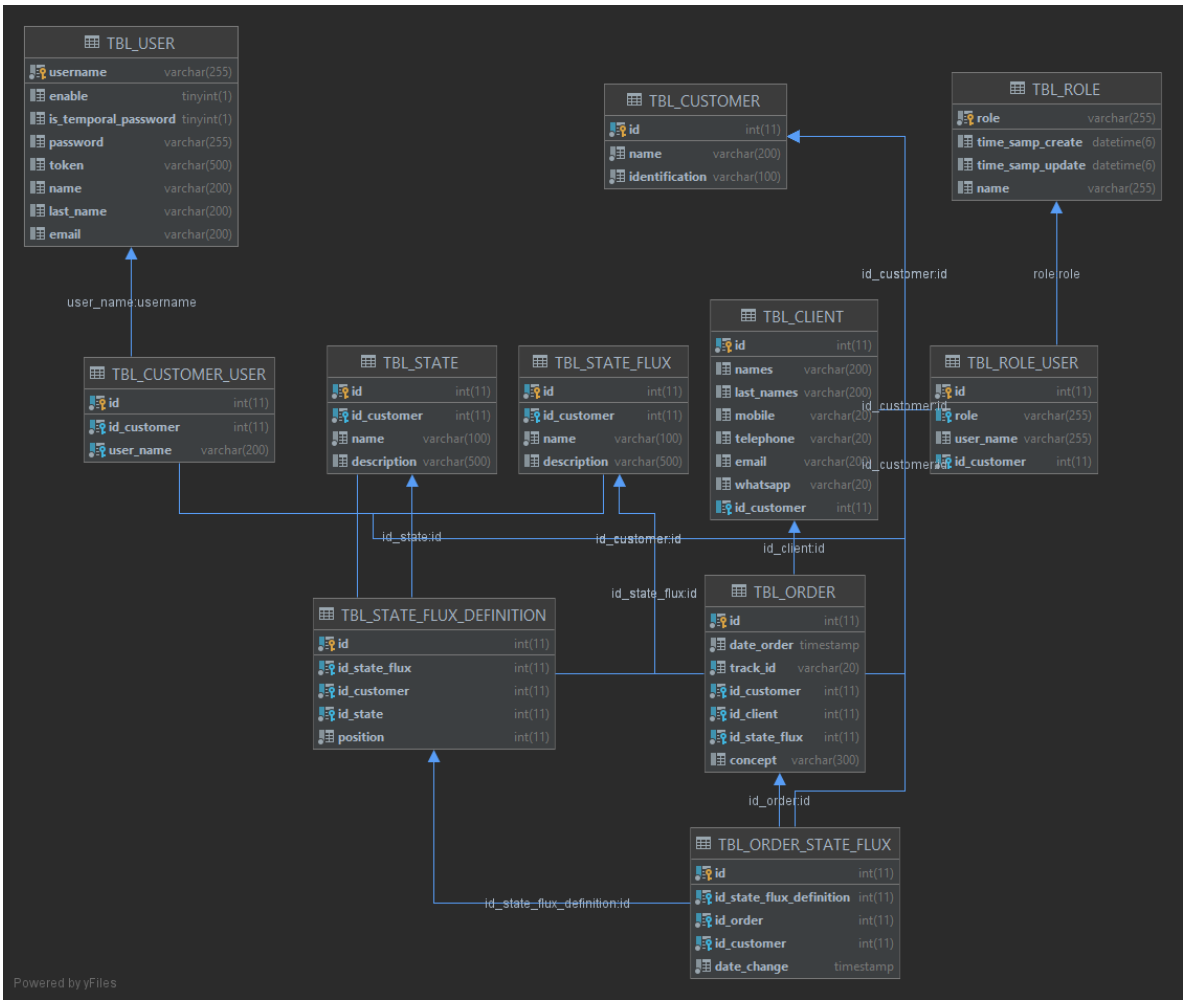


Figura 12. Modelo entidad relación (MER) que se implementó en la base de datos

5.3. Entorno de control de la aplicación.

Se construyó un microservicio reactivo en una arquitectura limpia modular, debido a que sus componentes están conectados a través de interfaces y no dependen de la librería que se utilizó para dicho fin. Estos componentes son módulos diferentes que se encuentran en la capa de infraestructura y son accedidos por el Core a través de interfaces, esto implica que, si en algún momento se desea cambiar la base de datos de MariaDb a un Postgres, solo se deben implementar los métodos de la interfaz, garantizando que se retornan las mismas estructuras que se tienen definidas

en el Core y esto hace que el proyecto sea muy modular y fácil de adaptar a nuevas implementaciones (ver Figura 13). Esto implica que el microservicio contenga separación por dependencias físicas, lo anterior se refiere a que, si un desarrollador quiere usar un componente que se encuentra en el módulo de infraestructura desde el componente core, al momento de realizar la compilación de la aplicación ella arroja un error porque no encuentra la clase, pero si usa la interfaz si es posible usar estas funcionalidades.

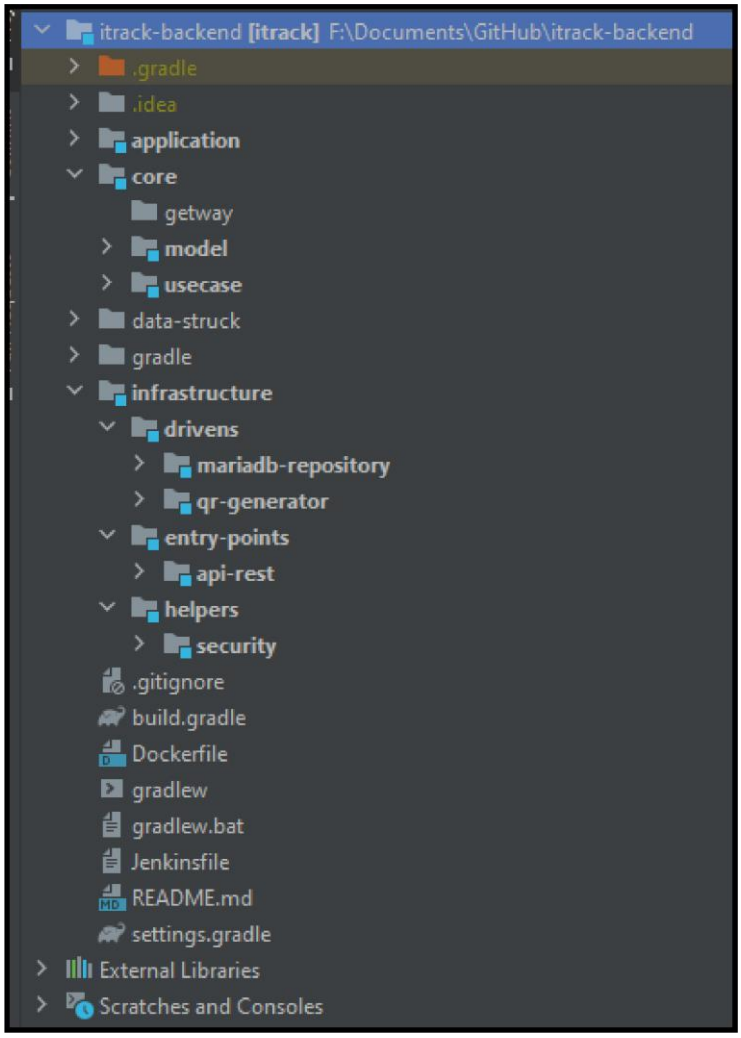


Figura 13. Estructura del microservicio

Otras tecnologías usadas para la construcción del microservicio fueron spring boot y java en su versión 8, que de acuerdo con 2020 Java

Technology Report [11] estas dos tecnologías fueron las más usadas durante el 2020, con una programación funcional y reactiva. La programación reactiva ofrece trabajar con flujos de datos paralelos, dando mayor rendimiento a la aplicación, y la programación funcional es una consecuencia de usar Project Reactor.

Se expusieron servicios REST de forma reactiva usando WebFlux. Se optó por implementar una base de datos relacional MariaDb debido a que es una base de datos de código libre evolucionada de MySql, además de soportar una conexión reactiva usando un driver reactivo llamado r2dbc-mariadb en su versión 1.0.1, Al realizar pruebas de carga se presentaron problemas con este driver reactivo, por lo cual fue necesario implementar un driver jdbc normal y envolvimos el repositorio en un flujo reactivo para no perder toda la reactividad del proceso. El modelo de base de datos resultante se ve en la Figura 12.

Se implementaron las entidades de base de datos con spring data reactivo.

Se construyeron 3 módulos importantes: módulo de seguridad, de ordenamientos y de flujos.

- Módulo de seguridad: el módulo de seguridad se implementa con la capa de seguridad de spring llamada spring security. Además, se adiciona una librería llamada jwt (Json web token) que provee una conexión segura entre dos partes.

  Parte del módulo de seguridad es la validación de la sesión:

  Si el usuario envía un token ya expirado o erróneo, el sistema entrega un error 401 que significa que no está autorizado. Si el usuario envía un token valido pero no tiene el roll para consumir este servicio, el sistema te entrega un error 403 que significa prohibido, y si el usuario envía un token valido y tiene el roll el módulo de seguridad llama el servicio y retorna la información solicitada. Ver Figura 14

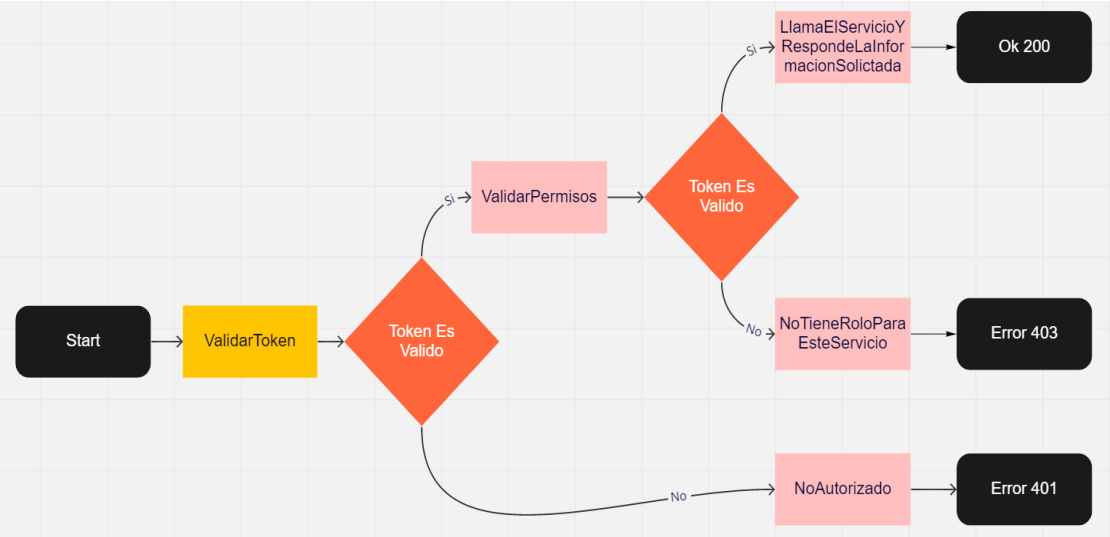


Figura 14. Diagrama de validar token, módulo de seguridad

- Módulo de órdenes: este módulo es el que se encarga del registro de órdenes, asignación de flujo, cambios de estados y consultas de estas por los usuarios teniendo el número de guía.

  En este módulo uno de los flujos más importantes es la creación de la orden. Esta consiste en validar si el flujo existe, y si este existe, procede a la creación del objeto, guardarlo en base de datos y enviar un email al cliente dueño de esta orden, este correo contiene la url para consultar el seguimiento de esta y adjunto viene un Qr el cual puede ser escaneado para obtener también esta url. Si el flujo no existe el servicio arroja un error en la petición. Ver Figura 15.

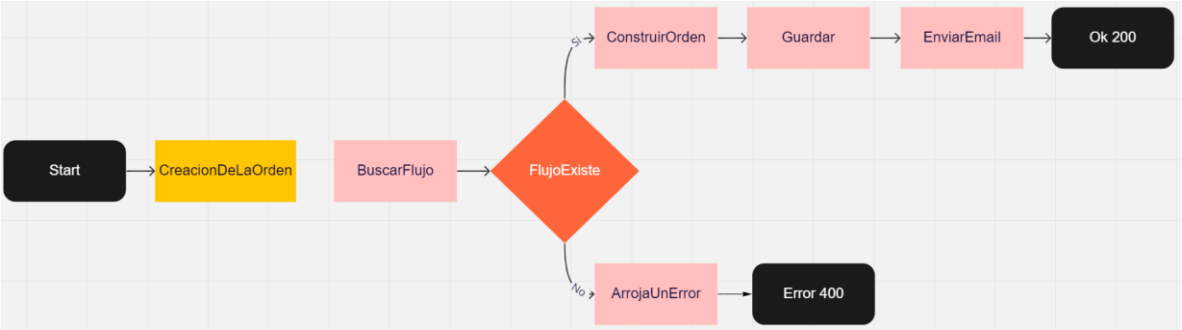


Figura 15. Diagrama de creación de una orden

- Módulo de Flujos: este módulo es el encargado de persistir la configuración de los diferentes flujos genéricos los cuales se le aplican a una orden y de persistir los estados que componen el flujo en un orden específico.

  Para este módulo la operación más importante es la creación de flujos, este consiste en la creación del objeto a guardar en base de datos. Si al momento de guardar se está insertando un estado que no existe la aplicación retorna un error 500 porque está violando la estructura de base de datos. Ver Figura 16.

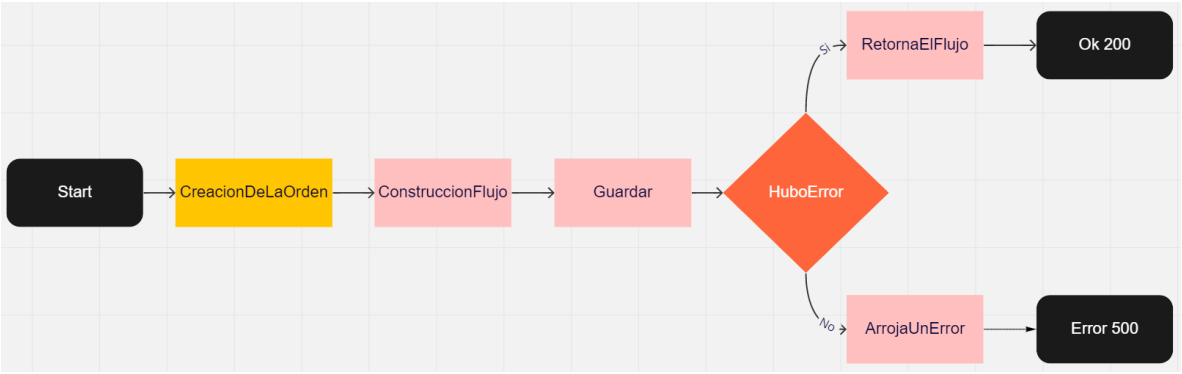


Figura 16. Diagrama de creación de flujos

Se creó también una imagen de docker para correr el microservicio y se desplegó en un servidor expuesto a internet usando integración continua con Jenkins. Un microservicio desplegado con docker da la posibilidad de

tener una granja de contenedores sirviendo al mismo propósito y poder así soportar la concurrencia de muchos usuarios.

5.4. Interfaz de Usuario final.

Se implementó el proyecto del front-end con un marco de trabajo de aplicaciones híbridas.

La interfaz gráfica se construyó haciendo uso de Ionic, con el objetivo de tener aplicación móvil y aplicación desplegada en la web con un solo desarrollo. Para la parte móvil se generó el instalador de la aplicación hacia dispositivos móviles Android. Para la parte web se creó una imagen de docker con Nginx compilando los binarios de Ionic e inyectándolos a la imagen, esta se despliega en un contenedor de docker alojada en el mismo servidor del back-end.

Se construyeron múltiples vistas integradas con el back-end por medio de servicios REST de forma segura, con token de autenticación y autorización.

- Módulo de inicio de sesión: se compone de cuatro vistas: la de inicio de sesión, la de selección de empresa, la de recuperación de contraseña y la de cambio de contraseña.
    - El inicio de sesión es el formulario por el cual se obtiene la sesión inicial con usuario y contraseña.
    - La selección de empresa es un formulario que ocurre después de que el usuario inicia su sesión, donde se selecciona la empresa a la que pertenece el usuario autenticado, debido a que puede pertenecer a varias empresas y la aplicación soporta múltiples empresas.

    - Formulario de cambio de contraseña ocurre cuando el usuario es creado e inicia por primera vez. Después de autenticado obliga el cambio de esta.

- Módulo de órdenes: este módulo comprende la creación de órdenes, la visualización de todas las órdenes y la visualización de una orden sin estar autenticado.
- Módulo de flujos: este módulo comprende la creación y visualización de estados que son necesarios para la definición de un nuevo flujo, el listado de flujos y la creación de los flujos con los estados configurados previamente.

A continuación, en las Figura 17 y Figura 18, se presentan las vistas que contiene la aplicación construida.

En la Figura 17-1se muestra el formulario de inicio de sesión que es la primera vista que tiene el usuario dentro de la aplicación. El nombre de usuario es el correo electrónico del usuario, y la contraseña es autogenerada cuando se crea el usuario, siendo enviada al correo electrónico y de cambio obligatorio al primer inicio de sesión. Además, el usuario puede restablecer su contraseña (Figura 17-2) en cualquier momento, donde se genera una contraseña aleatoria que llega a su correo y debe ser cambiada en el próximo inicio de sesión.

Después de iniciada la sesión, el usuario debe seleccionar la empresa a la cual está asociado el producto que desea consultar (Figura 17-3), ya que este puede pertenecer a varias empresas que cuentan con esta aplicación.

La vista que se muestra en la Figura 17-4 es el menú de la aplicación, donde tiene acceso a las demás funcionalidades.

En la Figura 17-5 se observa la lista de órdenes de la empresa, con su progreso en un círculo azul, el número de tracking (ACE719C para la orden

superior) y la fecha en la que se creó la orden, siguiendo el formulario mostrado en la Figura 17-6. Para crear la orden se debe contar con dos insumos: el flujo y el cliente. El flujo corresponde a la agrupación de estados por los que pasa la orden (Figura 17-7) y el cliente (Figura 18-9) es el dueño del producto, que también puede crearse si no existe (Figura 18-10). Si un flujo requerido no existe es posible crearlo, como se muestra en la Figura 17-8. Para ello se debe contar con la lista de estados (Figura 18-11) necesarios para definir el proceso. Si esta lista de estados no se ha definido al momento de la creación del flujo, esta puede crearse (Figura 18-12).

En la Figura 18-13 se muestra la orden perteneciente a un cliente dado. Este cliente puede consultarla únicamente con el número de tracking, sin necesidad de iniciar sesión. El link de consulta se envía al correo electrónico del cliente al momento de crear la orden.

Por último, los usuarios realizan el cambio de estado de un producto en una vista como la mostrada en la Figura 18-14 para que pueda ser consultado por el cliente. Cuando el usuario ingresa el número de tracking y da clic en el botón "Cambiar estado", por regla de negocio, este solo avanzará al siguiente estado definido en el flujo.

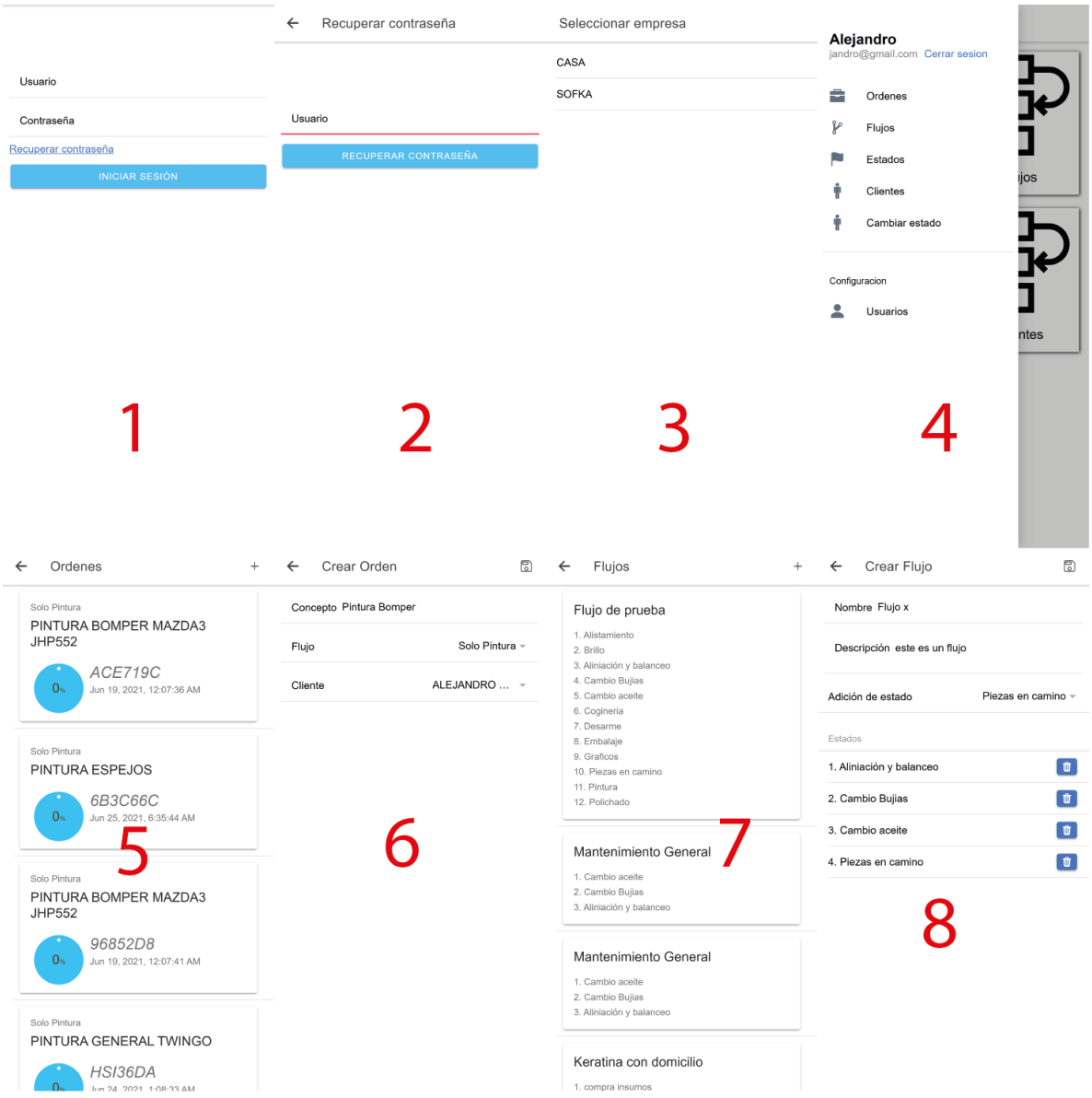


Figura 17. Capturas de pantalla de la aplicación

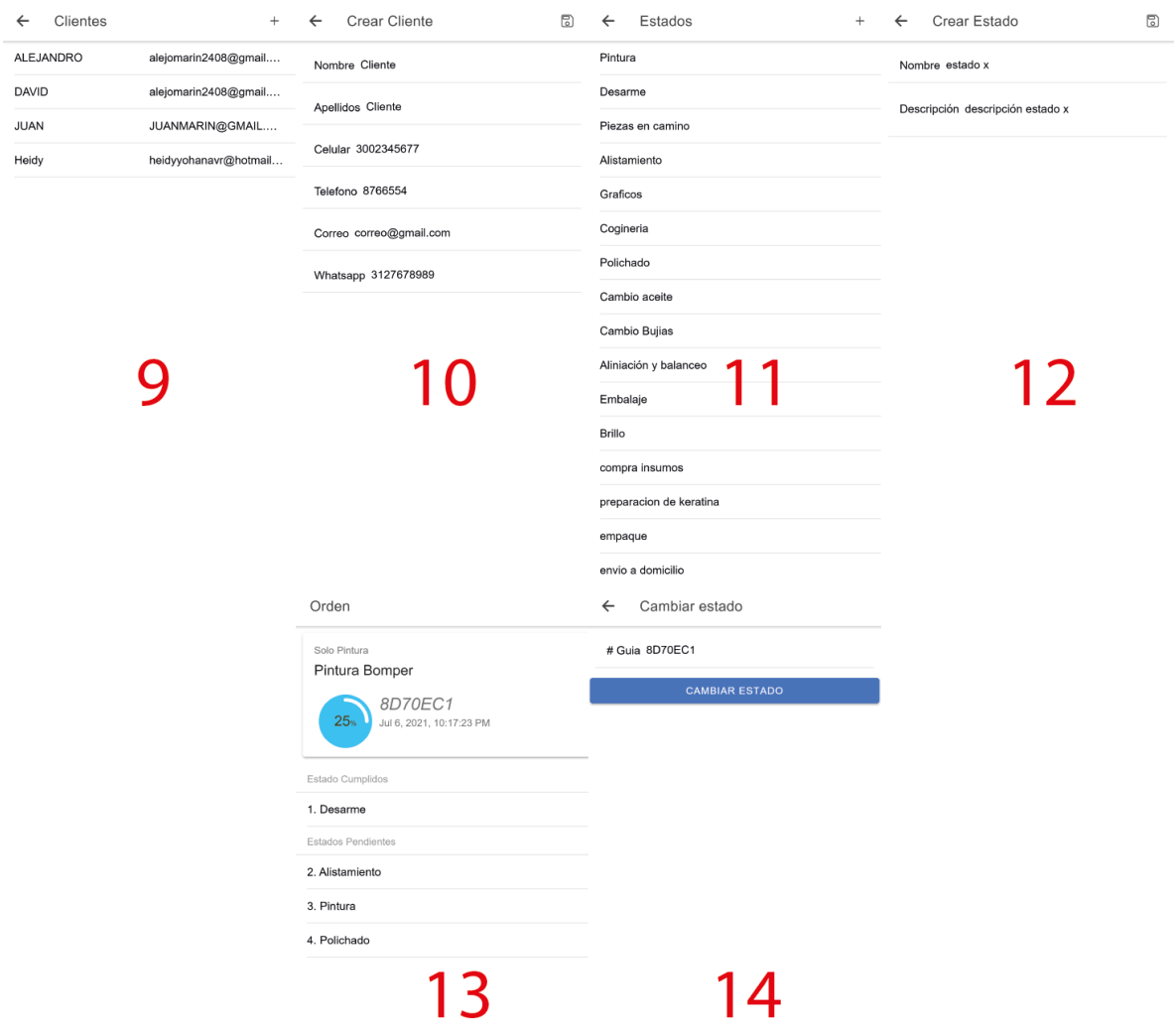


Figura 18. Capturas de pantalla de la aplicación

5.5. Integración y producto final.

Se hizo uso de la herramienta de integración continua Jenkins, aplicando el concepto de integración continua y automatizando todo el flujo de compilación, configuración de parámetros y despliegue de las dos aplicaciones (Figura 19).

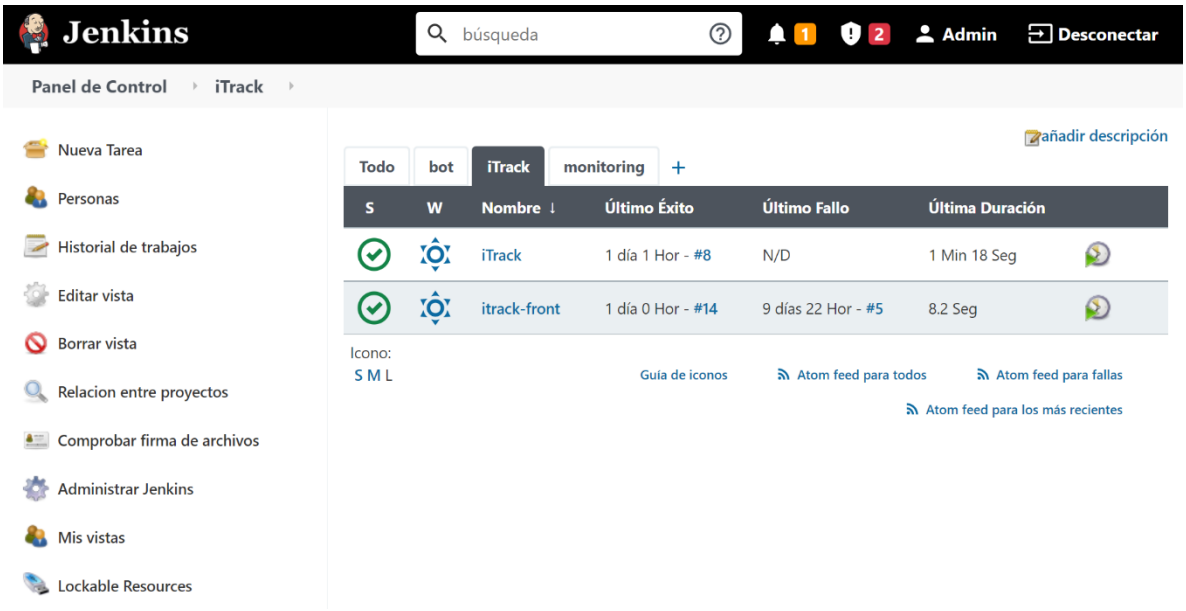


Figura 19. Herramienta Jenkins

La configuración de Jenkins consiste en automatizar todo el proceso de despliegue de una aplicación en un servidor productivo. Para desplegar el microservicio se configura el repositorio donde está alojado el código fuente junto con sus credenciales (Figura 20) para que la herramienta pueda acceder a este, debido a que es un repositorio de carácter privado.

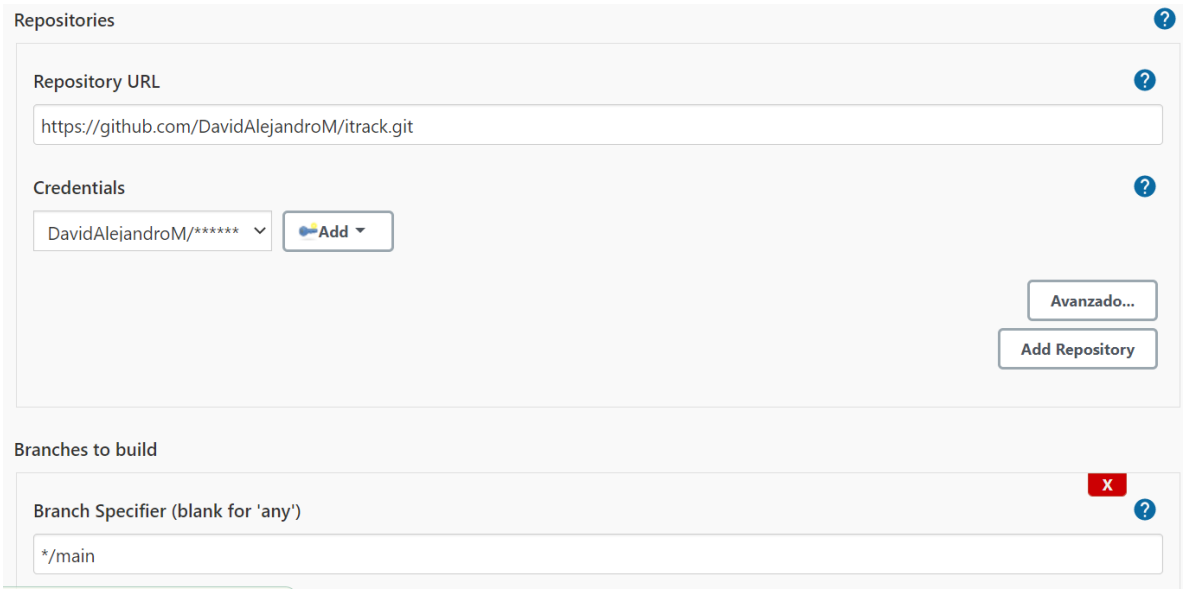


Figura 20. Configuración del repositorio de descarga de github del microservicio.

Antes de compilar la aplicación se deben reemplazar las propiedades de prueba del microservicio por las propiedades del ambiente productivo, tales como cadena de conexión a la base de datos, credenciales, etc (ver Figura 21).

Figura 21. Reemplazo de propiedades del microservicio

Luego de esto, se compila la aplicación usando gradle en su versión 6.3 para generar el archivo jar, que es el ejecutable del microservicio (Figura 22).

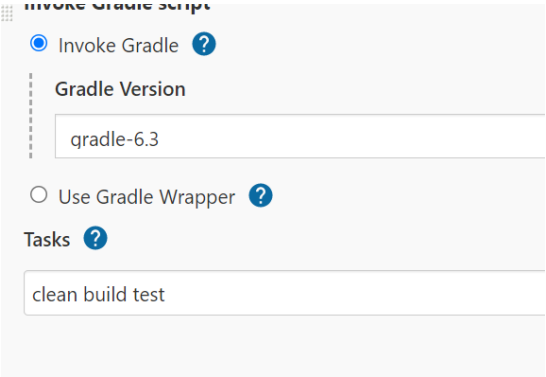


Figura 22. Compilación del microservicio

Después, se compila la imagen de Docker, donde toma el archivo jar y lo inyecta a la imagen construida. Luego de realizar esta construcción, se ejecuta la imagen en un contenedor de Docker dando como resultado el microservicio corriendo y expuesto por el puerto 8000. Adicionalmente, el

contenedor es agregado a una red de Docker donde se encuentra corriendo el contenedor de base de datos (ver Figura 23).

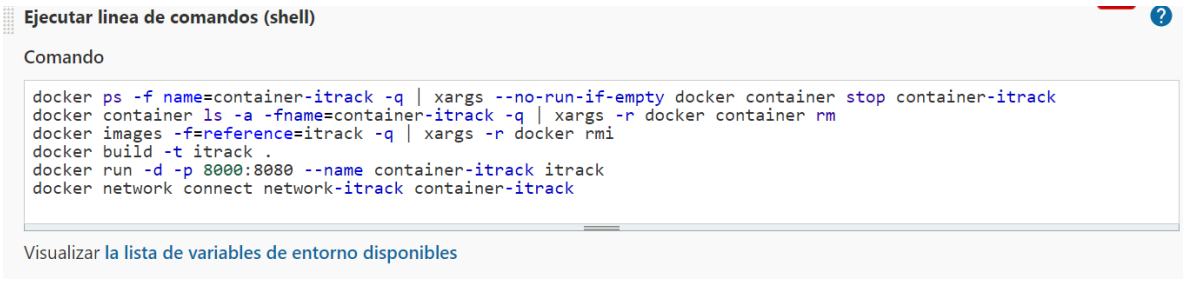


Figura 23. Dockerización y despliegue del microservicio

Para el despliegue del front-end se configura el pipeline descargando el código fuente desde github (ver Figura 24).

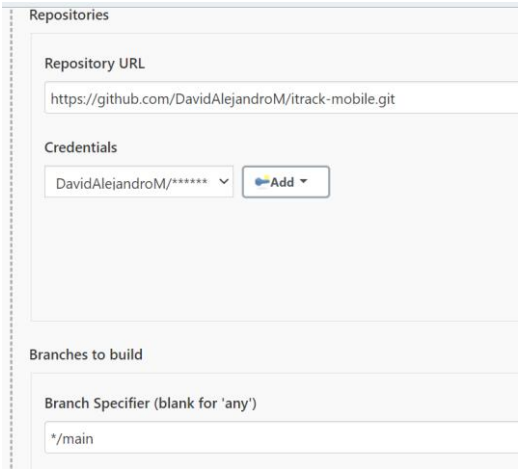


Figura 24. Configuración de repositorio front-end

Por último, se realiza la configuración de la creación de la imagen de Docker y el despliegue de esta en un contenedor, exponiéndola por el puerto 80 (ver Figura 25).

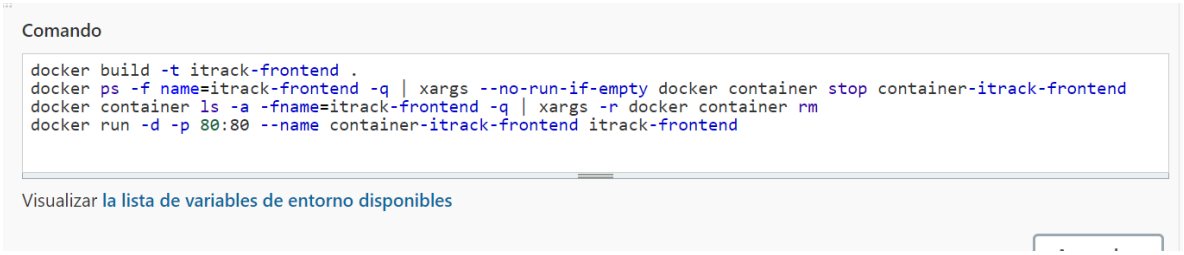


Figura 25. Dockerización y despliegue del front-end

Se realizaron pruebas funcionales de forma manual para validar el correcto funcionamiento de la aplicación, También se realizaron pruebas de carga para validar el correcto funcionamiento. Después de realizadas las pruebas y detectar errores se procedió a su corrección, estas pruebas funcionales y la corrección de errores se hacían a medida que el software iba creciendo en características.

Realizando las pruebas de carga se encontró que el driver de base de datos reactivo no estaba soportando la concurrencia y la conexión se perdía.

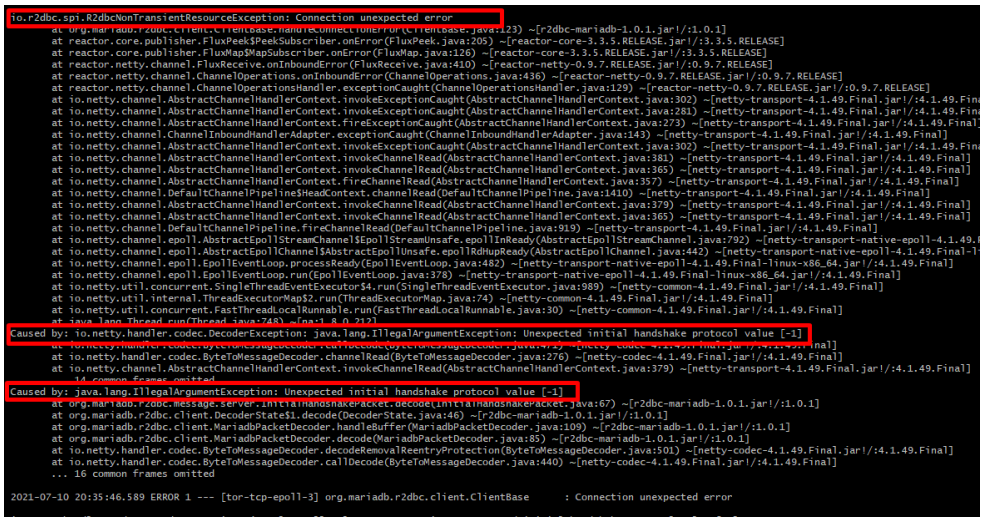


Figura 26. Error de conexión a la base de datos usando el driver reactivo cuando se realizaron pruebas de carga

Para resolver este error realizamos el cambio de la capa de datos dejando de usar un driver reactivo y usar el driver convencional jdbc. La lógica de negocio no cambio debido a la separación, porque no hay acoplamiento entre las reglas de negocio (Core) y el modelo de datos.

6. Resultados y análisis

Ya con la aplicación desplegada en el servidor se realizaron unas pruebas de carga individuales y conjuntas con algunos de los servicios. El servidor usado para las pruebas es un computador portátil corporativo con sistema operativo Ubuntu server en su versión 20.04 (Figura 27.). El servidor cuenta con un procesador Intel Core i5 M520 de 4 núcleos (Figura 29) a 2.4 y cuenta con una memoria RAM de 6GB.

Figura 27. Información del sistema operativo del servidor

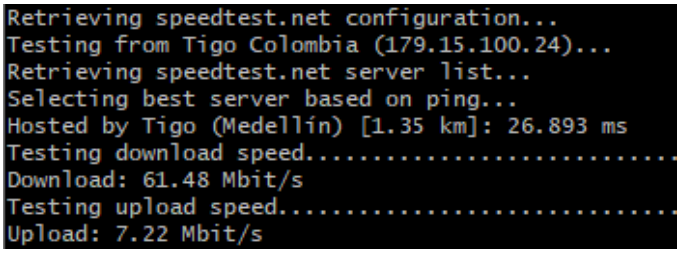


Figura 28. Prueba de velocidad de internet desde el servidor

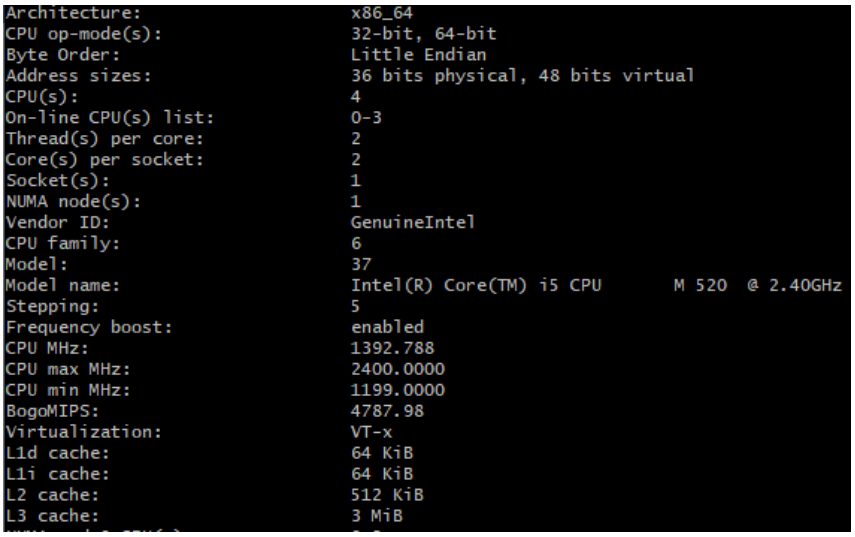


Figura 29. Especificaciones del procesador

Las pruebas de carga se realizaron usando la aplicación JMeter, es una aplicación de código abierto que corre bajo java que sirve para ejecutar pruebas de carga y medir el rendimiento de otras aplicaciones [17]. Para las pruebas de la plataforma que se construyó se realizaron de dos maneras, de forma individual a algunos de los servicios y de forma conjunta a los servicios objetivos, Para estas pruebas se simularon 1,10, 50 y 100 usuarios concurrentes cada segundo por 5 minutos cada prueba. Los usuarios concurrentes por segundo en JMeter se ingresa en el campo TargetRate y los 5 minutos se ingresan en el campo Hold Tarjet Rate Time. Convirtiendo los 5 minutos en segundos da un total de 300 segundos (ver imagen Figura 30). Y la configuración de cada petición se realiza como se muestra en la Figura 31.

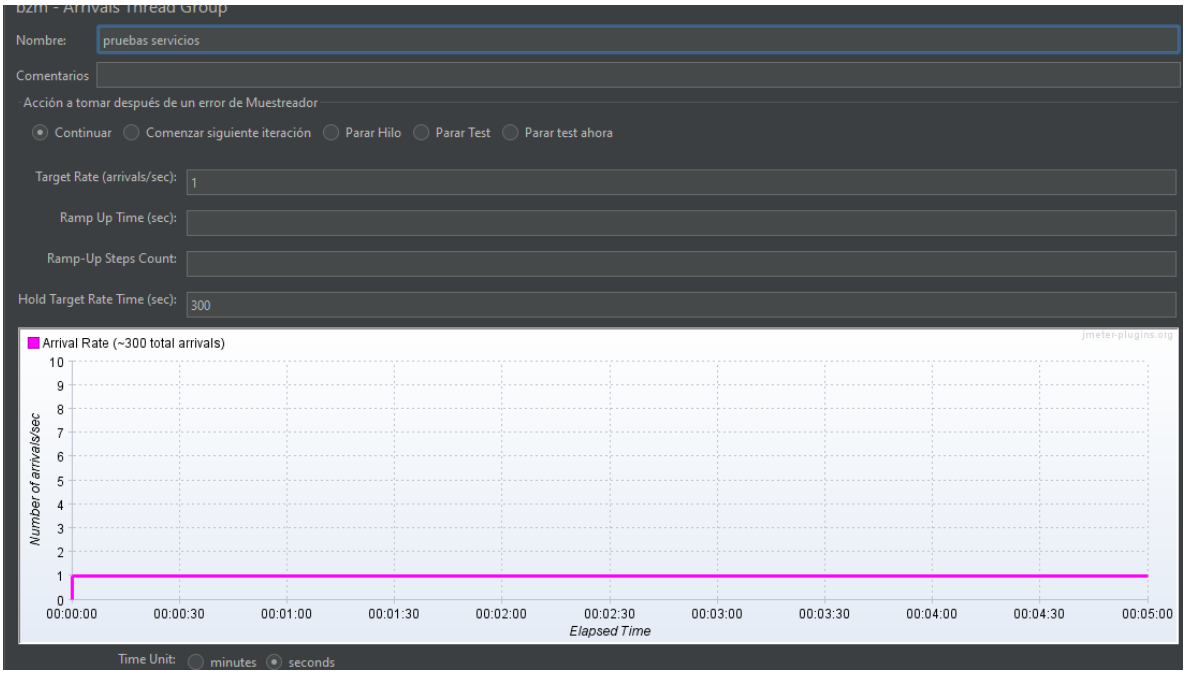


Figura 30. Configuración de usuarios concurrentes en JMeter

Figura 31. Configuración de un servicio REST en JMeter

6.1. Pruebas servicios individuales

Pruebas del servicio de obtener todos los estados.

Tabla 1. Pruebas del servicio de obtener todas las ordenes.

Usuarios concurrentes	N° Muestras	Media(ms)	Error(%)	Rendimiento/sec
1	300	252	0	1
10	3000	548	0	10
50	14999	273933	4,57	15
100				

Tabla 2. Pruebas del servicio de obtener una orden especifica

Usuarios concurrentes	N° Muestras	Media(ms)	Error(%)	Rendimiento/sec
1	300	241	0	1
10	3000	214	0	10
50	14999	90	0	50
100	30000	156	0,01	100

Tabla 3. Pruebas del servicio de creación de una orden

Usuarios concurrentes	N° Muestras	Media(ms)	Error(%)	Rendimiento/sec
1	300	143	0	1
10	3000	82	0	10
50	15000	365	0.01	50
100	30000	51841	26,44	77,9

6.2. Pruebas servicios conjuntas

Para estas pruebas solo se tuvieron en cuenta hasta 50 usuarios concurrentes debido al resultado de las pruebas con 10 y 50 usuarios concurrentes.

Tabla 4. Pruebas de servicio conjuntas

Serivicio	Usuarios concurrentes	N° Muestras	Media(ms)	Error(%)	Rendimiento/sec
getAllStates	1	300	43	0	**1**
getAllOrders		300	108	0	**1**
getOneOrder		300	12	0	**1**
createOrder		300	964	0	**0,99**
Total		1200	282	0	**4**
getAllStates	10	3000	94393	0	**5,3**
getAllOrders		1284	68802	0	**2,3**
getOneOrder		1514	45125	0	**2,7**
createOrder		1790	44987	0	**3,2**
Total		7588	68578	0	**13,4**
getAllStates	50	14999	235396	21,81	**10,4**
getAllOrders		11135	227703	62,04	**7,7**
getOneOrder		13986	250594	47.09	**9,7**
createOrder		24534	250594	21,96	**10,1**
Total		54654	251958	36,52	**38**

7. Trabajo a futuro

Debido al problema que tuvimos con el driver reactivo para la conexión a la base de datos, queda en consideración futura volver a cambiar el driver para que todo el microservicio tenga reactividad desde los puntos de entrada hasta la persistencia de los datos, esto para mejorar considerablemente los tiempos de respuesta de los servicios.

También se podría crear un proceso basado en eventos para el envío de emails a través de colas de Rabbit o un sistema de colas similar a este para que no dependa del flujo principal, y los servicios que dependen de este envío de emails puedan aumentar su rendimiento.

Se podría contratar una persona que tenga amplios conocimientos de experiencia de usuario para realizar un cambio radical a los formularios y vistas del front-end para que sea más amigable.

Implementar paginación en los servicios que retornan mucha información porque esto puede influir en los tiempos de respuesta de los servicios.

8. Conclusiones

- La implementación de la plataforma para seguimiento de servicios o procesos de manufactura de un producto no sólo es viable si no importante desde el punto de vista de clientes y empresarios.
- Para desarrollar soluciones en poco tiempo no es aconsejable usar librerías que no tengan mucho uso y sean relativamente nuevos porque pueden causar errores y atrasar el desarrollo.
- El microservicio se comporta muy bien cuando se llama servicios que no tienen múltiples consultas sobre la base de datos, como es el caso del servicio que retorna los estados.
- Es muy importante realizar las pruebas de carga a los servicios porque se pueden detectar errores que solo se podrían detectar cuando la aplicación ya se encuentre en ambiente productivo. Si se realizan el sistema sale con mucha mayor calidad.
- La empresa que adopte esta solución puede invertir el tiempo que ahora gasta en brindar información a sus clientes con respecto al estado de su producto, en mejoramiento de su empresa.
- Los clientes de empresas que cuenten con esta plataforma estarán informados en tiempo real sobre el estado y avance de su orden.
- El uso de microservicios envueltos con docker hace que las aplicaciones sean más escalables a nivel físico.

- El tiempo de respuesta de las aplicaciones reactivas puede ser menor si se usan las bondades de la librería Project Reactor, ya que esta usa paralelismo por hilos.
- Construir aplicaciones híbridas es una opción de bajo coste para empresas que aún están empezando con su modelo de negocio y este depende de una app móvil y un portal web porque se puede desarrollar una única aplicación y exportarla para diferentes tecnologías como Android, iOS.

9. Referencias

[1] Oficina asesora de planeación y estudios sectoriales, "Colombia Siguió Mejorando Las Cifras de Conectividad En El Primer Trimestre Del Año," 2020. https://www.mintic.gov.co/portal/inicio/Sala-de-Prensa/Noticias/151386:Colombia-siguio-mejorando-las-cifras-de-conectividad-en-el-primer-trimestre-del-ano.

[2] P. A. Vargas Rubio, "Comercio electrónico ha crecido más de 300% en Latinoamérica en la pandemia," 2020. https://www.larepublica.co/globoeconomia/e-commerce-ha-crecido-mas-de-300-en-latinoamerica-en-medio-de-la-pandemia-3000424 (accessed Feb. 04, 2021).

[3] "Rastreo de envíos de la mercancía con softwares en tiempo real," *movistar*. https://destinonegocio.com/co/gestion-co/rastreo-de-envios-mercancia-software/ (accessed Feb. 11, 2021).

[4] "Configure, Capture, Analyze | FactoryFour." https://factoryfour.com/ (accessed Feb. 04, 2021).

[5] "Enterprise Plant Management Platform | SafetyChain Software." https://safetychain.com/ (accessed Feb. 04, 2021).

[6] N. Chapaval, "Qué es Frontend y Backend," 2018. https://platzi.com/blog/que-es-frontend-y-backend/ (accessed Feb. 04, 2021).

[7] J. M. Agüero A and R. Malenda V, "Qué es Ionic: ventajas y desventajas de usarlo para desarrollar apps móviles híbridas," 2021. https://profile.es/blog/que-es-ionic/.

[8] P. Raquel, "Lenguaje HTML," 2017. https://economipedia.com/definiciones/lenguaje-html.html.

[9] S. J Perry, "Conceptos básicos del lenguaje Java," 2012.

https://developer.ibm.com/es/languages/java/tutorials/j-introtojava1/.

[10] "¿Qué es un lenguaje de programación orientado a objetos?," [Online]. Available: https://lenguajesdeprogramacion.net/diccionario/que-es-programacion-orientado-a-objetos/.

[11] "2020 java technology report," 2020. https://www.jrebel.com/blog/2020-java-technology-report.

[12] "Spring Framework," [Online]. Available: https://spring.io/projects/spring-framework.

[13] M. Pulido, "Todo Lo Que Deberías Saber Sobre Programación Reactiva," 2019, [Online]. Available: https://slashmobility.com/blog/2019/09/programacion-reactiva/.

[14] "¿Qué es una base de datos? | Oracle Colombia." https://www.oracle.com/co/database/what-is-database/ (accessed Feb. 04, 2021).

[15] "¿Qué son y para qué sirven los microservicios?" https://www.redhat.com/es/topics/microservices (accessed Feb. 04, 2021).

[16] "¿Qué es Docker?" https://aws.amazon.com/es/docker/.

[17] "Apache JMeter." https://jmeter.apache.org/.

10. ANEXOS

Anexo 1

Encuesta previa

Encuesta iTrack

Para responder esta encuesta, tome la posición del cliente (usuario de la aplicación) o como empresario (administrador de la aplicación). Esto con el objetivo de recolectar información necesario para la construcción de un sistema de información de seguimiento en tiempo real de manufactura de productos u servicio como trabajo de grado de la Universidad de Antioquia para optar por el titulo de Ingeniero de Sistemas.

Toda la información subministrada va ser tratada confidencialmente bajo la Ley 1581 de 2012. Será usada única y exclusivamente para uso académico.

*Obligatorio

Correo Electrónico

El correo será usado para verificación de la información

Tu respuesta

¿Cuál es su ocupación actual?

- [ ] Estudiante
- [ ] Empleado
- [ ] Independiente
- [ ] Empresario
- [ ] No tiene una ocupación actualmente
- [ ] Otro:

Si usted es empresario, ¿Qué actividad principal realiza su negocio? Ejemplo: Latonería y pintura automotriz

Tu respuesta

¿Realiza compras o ventas por internet frecuentemente? *

- ◯ Sí
- ◯ No
- ◯ Omitir pregunta

¿Qué tan importante es tener información del estado de su producto o pedido en etapa de manufactura? Siendo 5 muy importante y 1 poco importante *

- ◯ 5
- ◯ 4
- ◯ 3
- ◯ 2
- ◯ 1

¿Qué tan importante es tener seguimiento de su producto o pedido en la etapa de entrega? Siendo 5 muy importante y 1 poco importante *

- ◯ 5
- ◯ 4
- ◯ 3
- ◯ 2
- ◯ 1

¿Si usted es empresario, que tantas veces al día preguntan sus clientes por el estado de su pedido o producto al día?

- ◯ 0 a 4
- ◯ 5 a 19
- ◯ 20 a 99
- ◯ mas de 100

¿Si usted es empresario, que información considera relevante conocer de sus clientes?

- ☐ Nombre completo
- ☐ Correo electrónico
- ☐ Teléfono fijo
- ☐ Celular
- ☐ Dirección con ciudad

¿Te gustaría visualizar el estado de su producto o pedido desde una pagina web? *

- ◯ Sí
- ◯ No
- ◯ Tal vez
- ◯ Otro: _____

Enviar

Página 1 de 1

Anexo 2

Manual de usuario

PLATAFORMA PARA SEGUIMIENTO DE SERVICIO O PROCESOS DE MANUFACTURA DE UN PRODUCTO

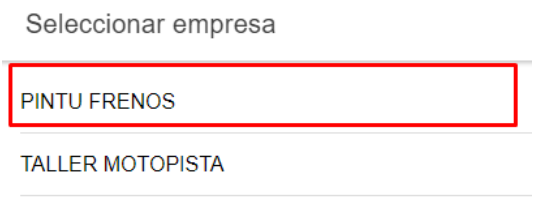
Inicio de sesión

Para iniciar sesión se debe contar con una cuenta previamente creada. El nombre de usuario es el correo electrónico asociado a la cuenta y la contraseña se autogenera al crear el usuario y se cambia en cualquier momento recuperando la contraseña.

Se ingresa el nombre de usuario, la contraseña y se da click en iniciar sesión.

Selección de empresa

Luego de iniciar la sesión se tiene que seleccionar la empresa que se desea entrar.

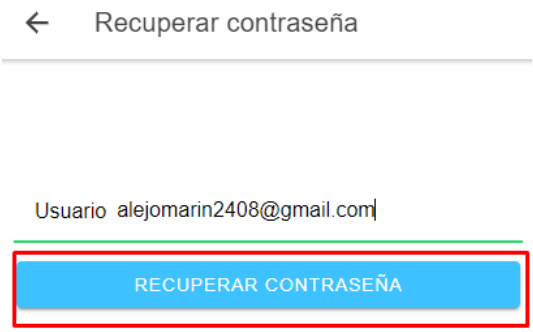


Para entrar en la empresa PINTU FRENOS solo es seleccionar y la aplicación llega a la vista principal cargando la información correspondiente a la empresa seleccionada.

Recuperar contraseña

Para recuperar la contraseña se debe dar click en "Recuperar contraseña" en la ventana del login.

Se ingresa el correo electrónico asociado a la cuenta y se da clic en el botón recuperar contraseña.



Cuando se recupera contraseña el sistema genera una contraseña temporal la cual debe ser usada para iniciar sesión.

Al iniciar la sesión con esta contraseña al ser temporal la aplicación detecta que es temporal y redirecciona a la vista de cambiar contraseña, se ingresa el correo asociado a la cuenta en el campo de usuario, la contraseña temporal en el campo de contraseña antigua y repetimos la contraseña nueva en los dos campos de contraseña antigua, esta contraseña debe coincidir y damos clic en cambiar contraseña.

Pantalla principal

En la pantalla principal se encuentran todas las operaciones que se pueden realizar en la aplicación.

Menú

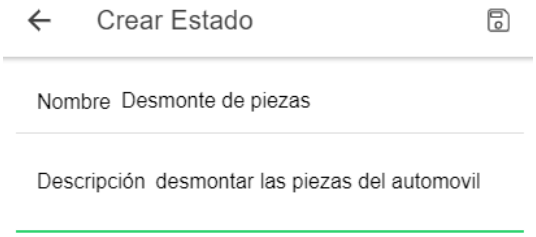
En el menú se encuentran las mismas operaciones que tiene la pantalla principal.

Crear estados

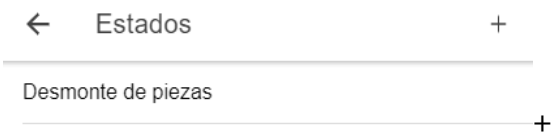
Para la creación de estados podemos ingresamos desde la pantalla principal o desde el menú en la opción "Estados".

Al ingresar en estados, nos muestra la lista de estados y el botón de adicionar un estado nuevo, damos clic en el botón "+" para ingresar el formulario de creación de estados.

Una vez en el formulario ingresamos el nombre del estado junto con su descripción y damos en el clic en el icono de guardar (disquete).



Cuando la creación es satisfactoria nos redirecciona a la vista donde se encuentran todos los estados creados.



Lista de estados

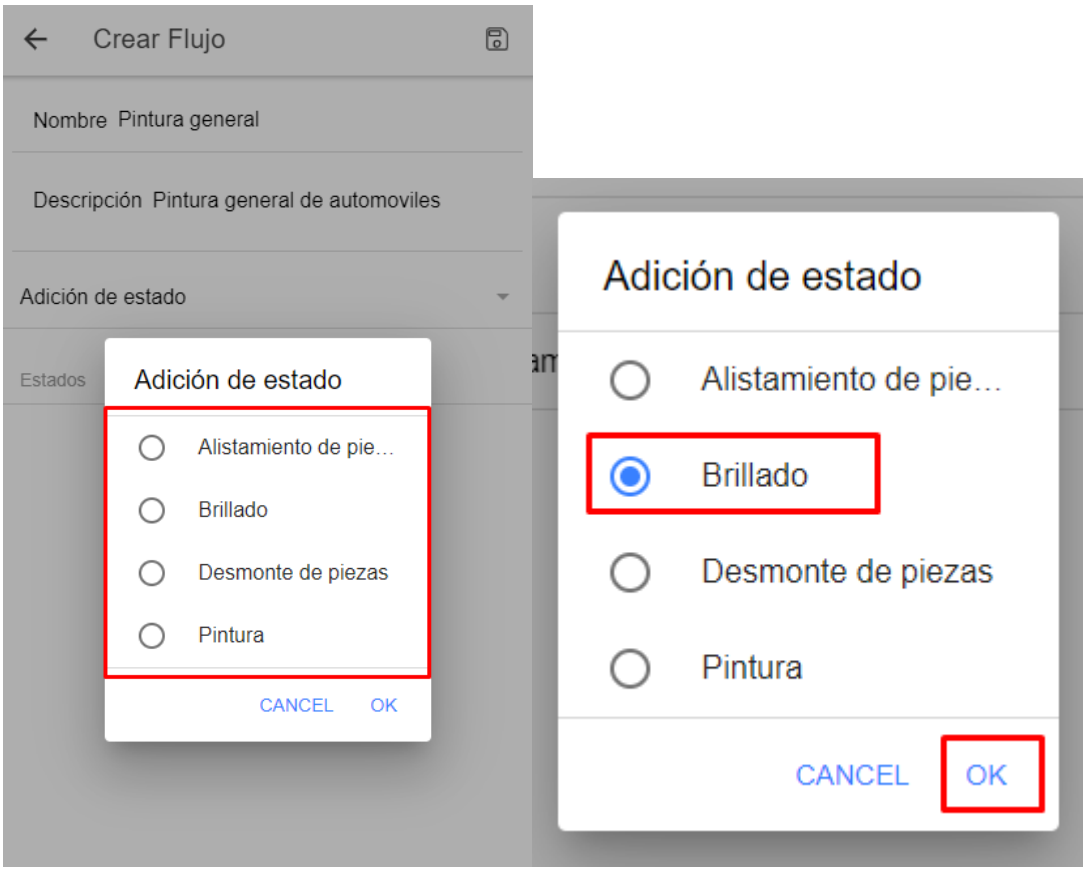
Para ingresar a ver la lista de estados, damos clic en el menú o en la pantalla principal al botón "Estados" y nos muestra la lista de estados creados previamente por cualquier usuario perteneciente a ese negocio.

Crear Flujos

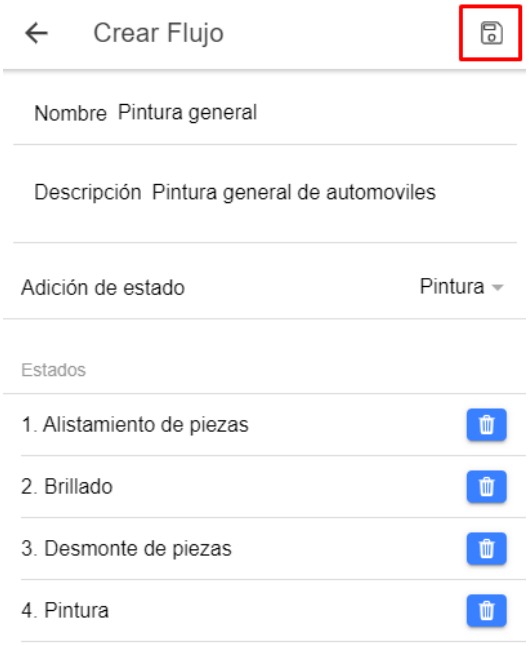
Debemos considerar que ya tenemos los estados creados para poder hacer la definición del flujo que vamos a crear, para ello damos clic en el botón" Flujos" en el menú o en la pantalla principal.

Damos clic en el botón de icono "+" para ir al formulario de creación de flujo.

Entrado al formulario ingresamos el nombre que va tener el flujo junto con su descripción y empezamos a seleccionar los estados que componen este flujo, cabe destacar que el flujo queda en el orden que se vayan seleccionando los estados. Para adicionar estados al flujo damos clic en "Adición de estado" nos muestra un modal en el cual seleccionamos el estado y damos clic en "OK".

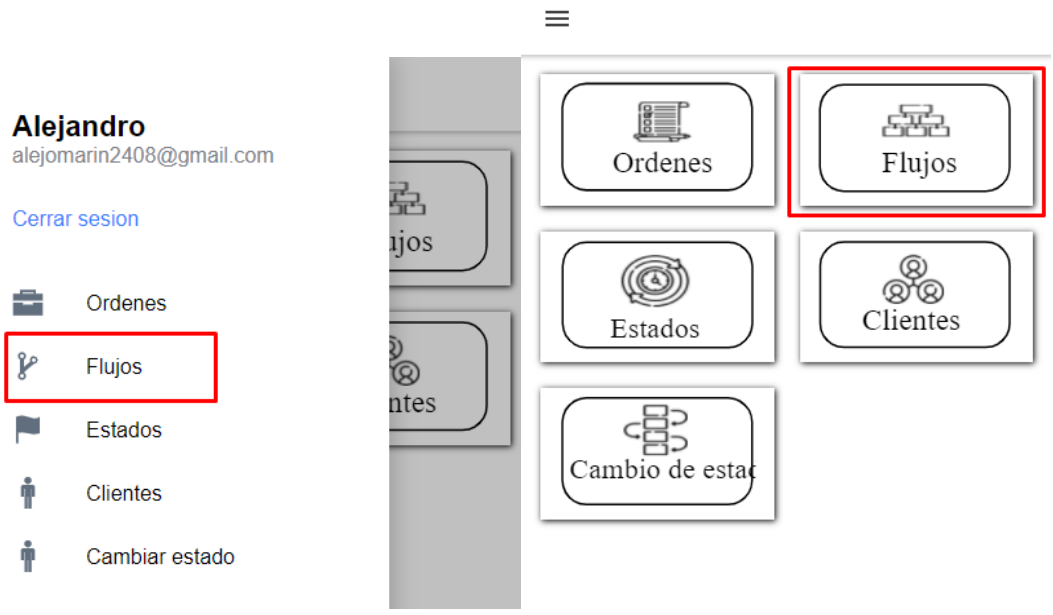


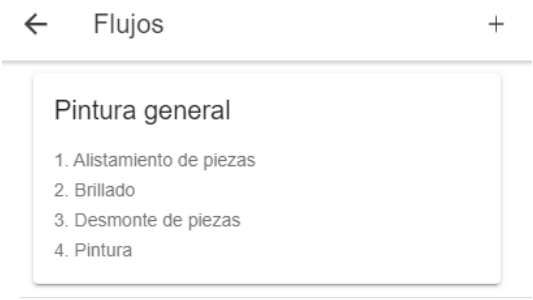
Luego de tener adicionados todos nuestros estados procedemos a guardar el flujo dando clic en el botón de guardar (icono disquete) y nuestro flujo quedará creado.



Lista de flujos

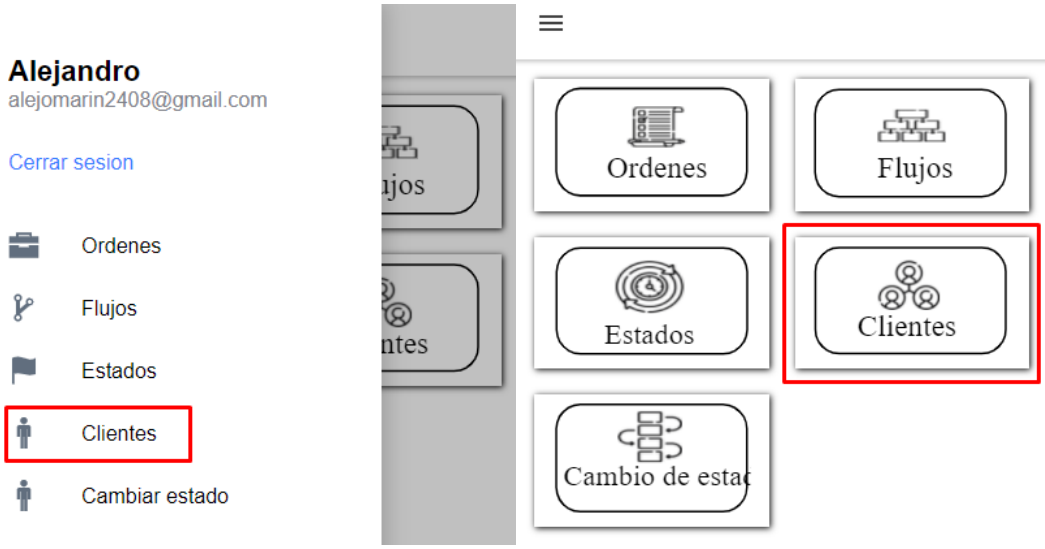
Para listar los flujos desde la pantalla principal o desde el menú damos clic en el botón "Flujos" el cual nos lleva a la vista donde se listan todos los flujos creados previamente.



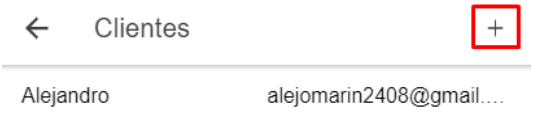


Crear cliente

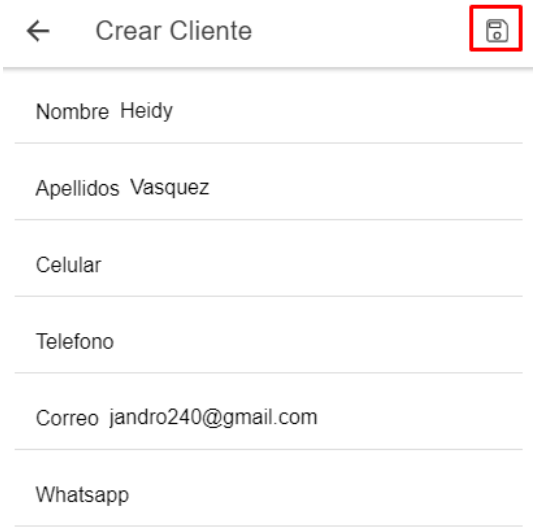
Para crear un cliente desde la pantalla principal o desde el menú damos clic en el botón "Clientes" el cual nos lleva a la vista de lista de clientes.



Damos clic en el botón "+" para ir al formulario de creación de clientes

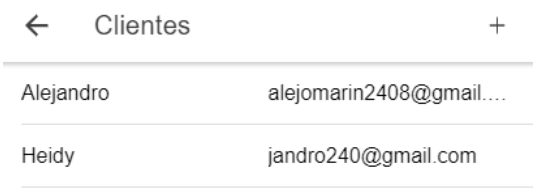


Estando en el formulario ingresamos la información mínima como: Nombre, Apellidos, Correo y damos en el botón de guardar.



Lista de clientes

Para listar los clientes desde la pantalla principal o desde el menú damos clic en el botón "Clientes" el cual nos lleva a la vista donde se listan todos los clientes creados previamente.



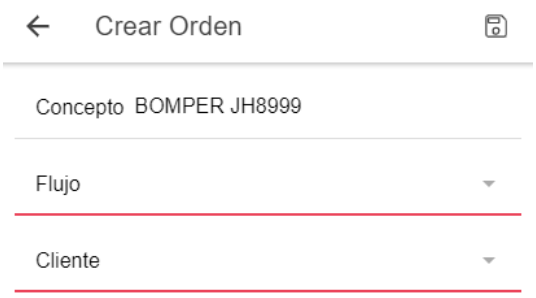
Crear orden

Para crear una orden, desde la pantalla principal o desde el menú damos clic en el botón "Ordenes" el cual nos lleva a la vista de lista de órdenes. Hay que tener en cuenta que para crear la orden el flujo ya debe estar creado y el cliente al que se le asocia la orden también.

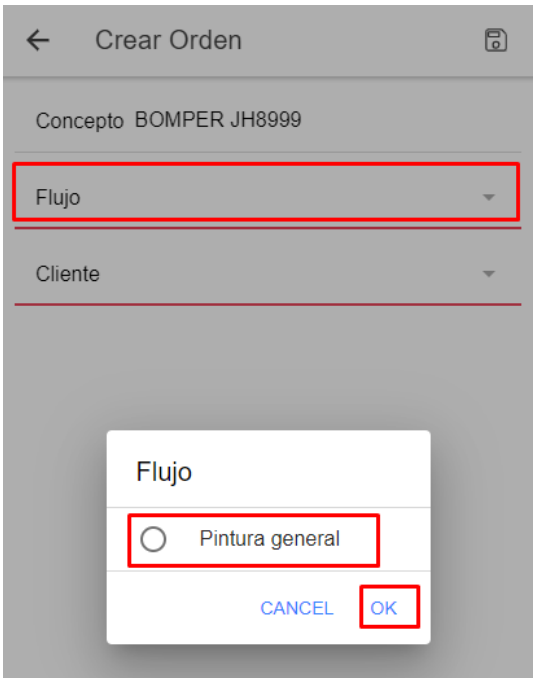
Damos clic en el botón "+" para ir al formulario de creación de la orden.

En el formulario de crear la orden es bastante sencillo en el concepto va el nombre del producto o el servicio que se presta, el flujo son los estados por

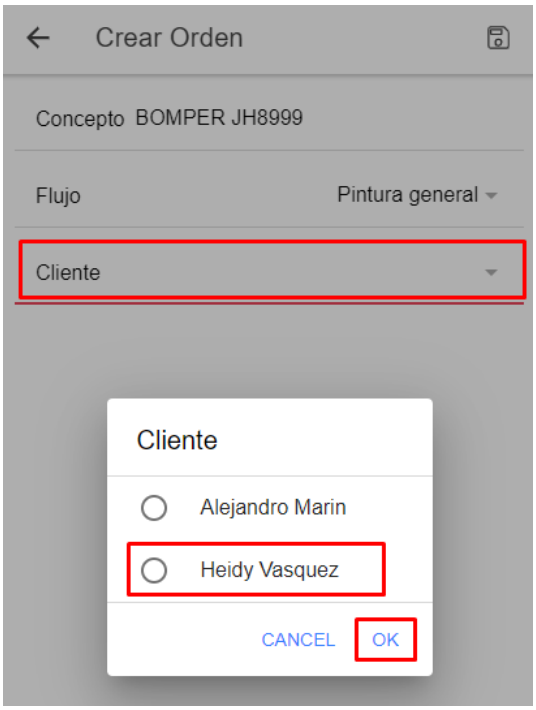
los que pasa el estado y el cliente es el cliente que contrata el servicio o compra el producto.



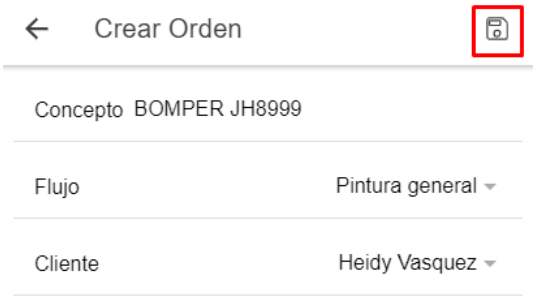
Para seleccionar el flujo se da clic en "Flujo" mostrando un modal en el cual uno selecciona el flujo que quiere asociar a la orden y se da clic en "OK".



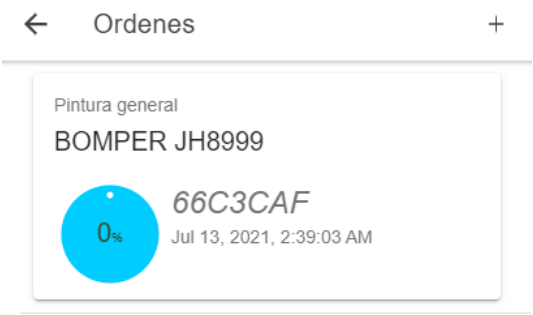
Para seleccionar el cliente se da clic en "Cliente" mostrando un modal en el cual uno selecciona el cliente que quiere asociar a la orden y se da clic en "OK".



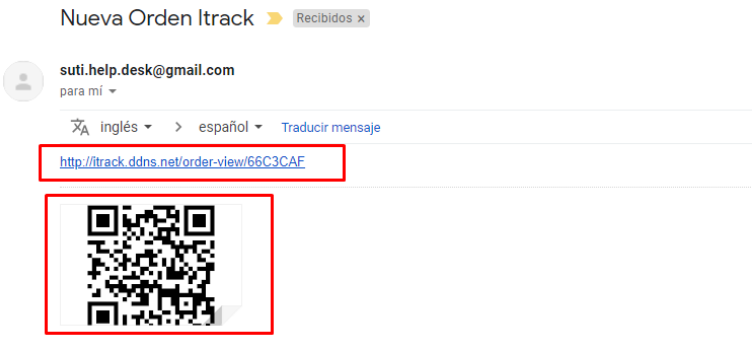
Luego de tener el formulario lleno se debe dar clic en el botón de guardar (icono disquete)



Al guardar la orden se redirecciona a la vista donde está el listado de las ordenes.

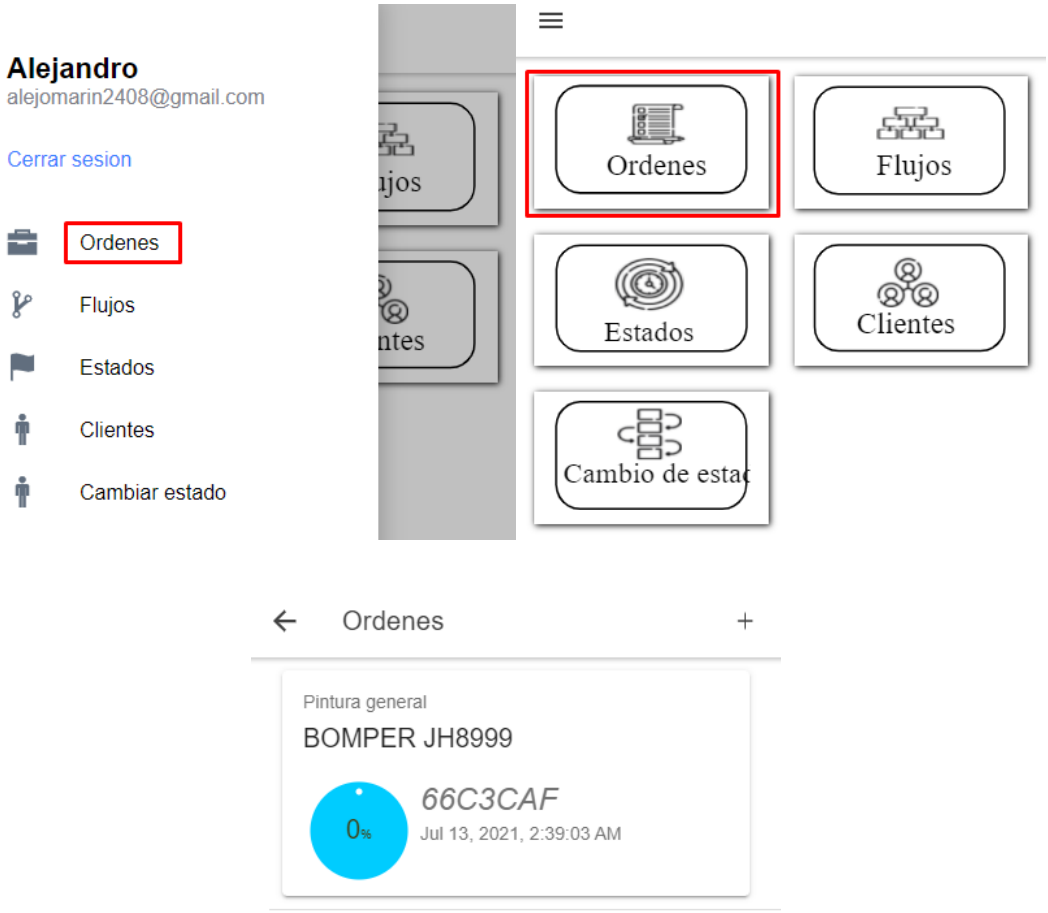


Cuando se crea una orden al cliente le llega un correo electrónico informándole que se cree una orden, enviando el link de seguimiento del pedido y adjunto un código qr para escanear desde el celular para abrir la vista de seguimiento.



Listar ordenes

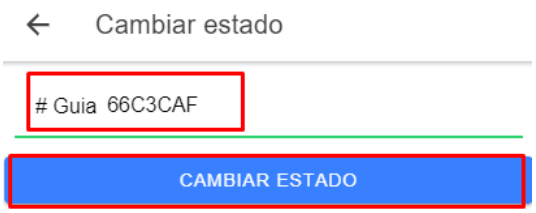
Para listar las ordenes desde la pantalla principal o desde el menú damos clic en el botón "Ordenes" el cual nos lleva a la vista donde se listan todos las ordenes creadas previamente.



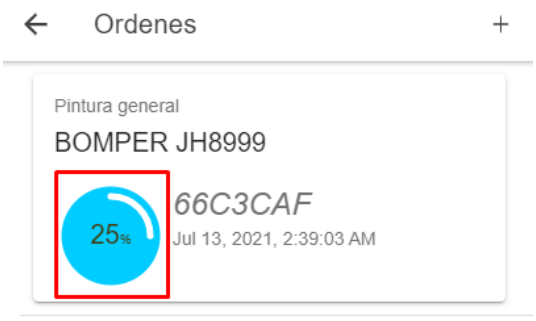
Cambiar estado

Para cambiar una orden al estado siguiente desde la pantalla principal o desde el menú damos clic en el botón "Cambiar estado" el cual nos lleva a la vista de cambiar estado. Esta funcionalidad está diseñada para ir al estado siguiente definido en el flujo, no se permite cambiar a un estado especifico.

En el formulario de cambio de estado tan solo hay que ingresar el número de guía de la orden y dar clic en el botón "cambiar estado".



Luego de cambiar el estado en la lista de ordenes la orden se actualiza con su progreso de completitud.



Adicional a esto cuando se realiza un cambio de estado el cliente recibe un correo electrónico informando el estado que ya fue completado junto con

la url para hacerle seguimiento el código qr para escanearlo desde el celular.

Seguimiento

Los clientes constantemente están recibiendo correos informando el progreso de su orden, para realizar seguimiento tan solo es dando clic en el link que aparece en todos los correos llevándolo a la página de la plataforma.

Luego de abrir la página, se muestra la información de progreso de la orden, junto con un resumen de los estados que ya se cumplieron y los estados que faltan por cumplirsen.


Orden
Pintura general
BOMPER JH8999
25%
66C3CAF
Jul 13, 2021, 2:39:03 AM
Estado Cumplidos
1
1. Alistamiento de piezas
Estados Pendientes
2
2. Brillado
3. Desmonte de piezas
4. Pintura