IMPLEMENTACIÓN DE PLATAFORMA WEB PARA LA DIVULGACIÓN DE MENSAJES DE TEXTO Y AUTOMATIZACIÓN DE HERRAMIENTA EXCEL PARA LA ACTUALIZACIÓN DE DATOS EN BASE DE DATOS PARA EL ÁREA DE MARKETING DE LA EMPRESA PARTNERS TELECOM COLOMBIA S.A.S

JOHN SEBASTIÁN ARÉVALO RODRÍGUEZ

UNIVERSIDAD SANTO TOMÁS
FACULTAD DE INGENIERÍA
INGENIERÍA DE TELECOMUNICACIONES
BOGOTÁ D.C
2024

IMPLEMENTACIÓN DE PLATAFORMA WEB PARA LA DIVULGACIÓN DE MENSAJES DE TEXTO Y AUTOMATIZACIÓN DE HERRAMIENTA EXCEL PARA LA ACTUALIZACIÓN DE DATOS EN BASE DE DATOS PARA EL ÁREA DE MARKETING DE LA EMPRESA PARTNERS TELECOM COLOMBIA S.A.S

Presentado por:
JOHN SEBASTIÁN ARÉVALO RODRÍGUEZ

CÓDIGO: 2182660

TRABAJO OPCIÓN DE GRADO PASANTÍAS EN LA EMPRESA PARTNERS TELECOM COLOMBIA S.A.S

DIRECTOR:
VÍCTOR MANUEL CASTRO RAMIREZ
INGENIERO ELECTRÓNICO

UNIVERSIDAD SANTO TOMÁS
FACULTAD DE INGENIERÍA
INGENIERÍA DE TELECOMUNICACIONES
BOGOTÁ D.C
2024

Tabla de contenido

Índice de figuras

Índice de tablas

Introducción

La empresa de telecomunicaciones Partners Telecom Colombia S.A.S inicio en el mercado colombiano de servicios de comunicaciones móviles en el año 2019, ofreciendo planes móviles a precios bajos con gran capacidad de ancho de banda, gigas de navegación, llamadas y siendo más permisibles en los contratos de permanencia para los planes pospagos. Este tipo de estrategia cambio el panorama del sector del servicio móvil en Colombia, permitiendo que más ciudadanos del país tuvieran acceso a las redes móviles. Esta estrategia de negocio permitió que la proveedora de servicios móviles WOM lograra entrar al mercado colombiano con nuevas estructuras de redes móviles.

La empresa WOM debe generar estrategias de negocio para el mercado cambiante del sector de las comunicaciones móviles, por lo que el área de marketing desarrollo realiza descuentos considerando el flujo de ventas en cada sucursal en todo el país, que consiste en que los descuentos son indirectamente proporcionales al flujo de ventas, variados semanalmente. La empresa requiere desplegar o divulgar nuevos servicios o promociones aprovechando los canales de comunicaciones como mensajes de texto de forma masiva a un número determinado de teléfonos.

La empresa se caracteriza por implementar y desarrollar aplicaciones web para ejecutar, monitorear y dar soporte a los procesos internos de cada una de las áreas. También desarrollan aplicaciones para uso comercial y de prestación de servicio, uno de esos aplicativos se llama RISKKEYFLOW, que les permite a los vendedores ofrecer los descuentos de los planes según la campaña, promoción o estrategia de marketing semanal. La empresa actualmente la arquitectura de las aplicaciones se manejan en lenguajes de TypeScript, java, microservicios en Docker y Oracle para la base datos.

El área de desarrollo de aplicaciones tiene especialistas, analistas y soporte técnico que reciben los requerimientos internos de las diferentes áreas de la organización e implementan mejoras de procesos, soporte técnico de nivel dos de cada aplicación y el análisis de optimización de procesos. Actualmente el área se encuentra desarrollado un aplicativo web que permita la divulgación masiva de mensajes de texto para promocionar las estrategias de marketing y mejorando la herramienta de plantilla para la aplicar los descuentos.

En este documento se pretende plasmar a detalle el proceso, el paso a paso y las cosas a tener en cuenta para solicitar una mejora o la implementación de una herramienta para agilizar los procesos internos. Para este caso la empresa necesita agilizar los tiempos de entrega para la aplicación de descuentos y la disponibilidad de programar los mensajes de divulgación de las promociones, una vez se tiene el requerimiento, se debe analizar, diseñar, desarrollar, implementar y probar la solución. Cada uno de estos pasos se desarrolla y explica durante este documento.

Planteamiento del problema

A partir del 1 de febrero del 2024 el operador móvil WOM tiene seis millones de suscriptores y una calificación de excelente calidad del servicio según las estadísticas de OOKLA, pero la empresa necesita atraer más suscripciones y usa aplicativos web y móviles para llevar publicidad a más ciudadanos. Por lo siguiente la empresa cuenta con grupo de profesionales en el área de las aplicaciones web permitiendo la implementación de las herramientas necesarias para lograr el alcance requerido para obtener suscriptores.

En el área de IT se divide en varias áreas, siendo alguna de ella el área de desarrollo de aplicaciones y el área de soporte. El área de desarrollo tiene la función de recibir las peticiones o necesidades internas de las diferentes áreas de la compañía y el área de soporte de ejecutar cambios en la base de datos y otras funciones. Actualmente, el área de desarrollo implemento una plataforma web donde se muestran ofertas de descuento según la cedula de ciudadanía del cliente, y se encuentra desarrollando un aplicativo web para desplegar estrategias de marketing con mensajes de texto.

La estrategia de marketing para realizar descuentos en los planes consiste en considerar el score, tipo de documento y lugar del país para realizar un descuento especifico al cliente, cada descuento está dividido por grupos. El aplicativo web muestra los descuentos registrados en la base de datos, esto consiste en que cada vez que marketing cambia los descuentos se debe actualizar las tablas en la base datos, esto implica que marketing envía la solicitud de actualizar los descuentos al área de desarrollo genera los comandos SQL para realizar la actualización y finalmente los queries son enviados al área de soporte en donde se ejecutan los cambios. Este proceso de actualización de descuentos tiene un tiempo de ejecución de 3 a 4 días debido a la carga laboral de cada una de las áreas sin embargo el área de marketing siempre solicita realizar los cambios de forma rápida, lo que implica una mayor carga de trabajo para el área de desarrollo y a su vez el personal de soporte que tiene disponibilidad 24 por 7 no cuenta con los conocimientos de comandos SQL.

Para el caso de divulgación de ofertas de descuentos o promociones por medio de mensajería de texto móvil, actualmente este servicio se encuentra tercerizado, la cual el proceso consiste en que el área de marketing envía la solicitud del despliegue de las ofertas a la empresa tercera que realiza la configuración de los mensajes kannel y actualización de los mensajes en la base de datos, finalmente el requerimiento es enviado al área de desarrollo en donde se genera una URL para desplegar los mensajes a una cierta cantidad de usuarios en un determinado tiempo y en ciertas horas del día. Este proceso tiene un tiempo de respuesta demasiado tardío y limitado que no permite divulgar las estrategias de mercadeo oportunamente, quitando la oportunidad de aprovechar el mercado de manera libre y rápida entre usuarios móviles.

De acuerdo con lo mencionado anteriormente se plantea la siguiente pregunta ¿Es posible desarrollar aplicaciones tecnológicas que mejoren algunas funciones básicas del área soporte técnico de la empresa Partners Telecom Colombia S.A.S?

Justificación

Las empresas de telecomunicaciones móviles deben de recurrir a diferentes estrategias de mercadeo para lograr tener más suscriptores a los operadores, algunas de esas estrategias es aplicar descuentos a los planes pospagos u ofrecer descuentos en celulares por la compra de una línea nueva. Estas estrategias de mercadeo deben ser divulgadas o transmitidas a los ciudadanos para hacer el consumo de esa publicidad, por lo tanto, los operadores móviles provechan el uso de la mensajería corta o comúnmente llamado mensajes de texto a lo que el operador WOM la usa, pero no de una manera óptima.

Una de las principales estrategias desarrolladas por el área de marketing para obtener más clientes pospagos es realizar descuentos a los diferentes planes, de entre 10 % y 100 %, con esto buscan incentivar a los clientes y usar voz a voz para atraer más clientes. Los descuentos se registran en una base de datos mediante códigos, donde los consulta un aplicativo web y que a su vez pueden consultarlo los vendedores y así aplicar los descuentos según los requerimientos.

En el proceso de actualización de descuentos tiene una duración de 3 día y una de las principales razones de tanto tiempo es que el área de soporte técnico no tiene persona con los conocimientos básicos para generar comandos SQL de actualización implica que en el área de desarrollo deben generar esos comandos, lo que implica una sobrecarga de tareas para el área.

Por otro lado, para la divulgación de mensajes de texto para ofertar las promociones en el momento este proceso lo lleva a cabo un software de una empresa tercera, la cual toma los requerimientos del área de marketing y generar los mensajes kannel para posterior mente ser programados para la divulgación. Este proceso tiene cuenta con un tiempo de 2 días y limita el número de mensajes.

Concluyendo las dos situaciones de tiempo y limitación que presenta el área de marketing se encuentra necesario la implementación de herramientas tecnologías que reduzca el tiempo de ejecución de actualizaciones y divulgación de mensajes de texto sin limitaciones.

Objetivos

Objetivo general

Desarrollar herramientas tecnológicas para mejorar los procesos, los tiempos de actualización de datos y despliegue de mensajes para el área de marketing en la empresa Partners Telecom Colombia S.A.S.

Objetivos específicos

1. Recopilar información y datos de los procesos de actuales para la actualización de datos para los descuentos y en la divulgación de mensajes de texto.
2. Diseñar la estructura de la implementación de la herramienta web para la divulgación de mensajes.
3. Implementar una herramienta de tecnología web en donde los usuarios puedan registrar los mensajes Kannel y programar los tiempos de divulgación.
4. Desarrollar automatización en la herramienta Excel en función de mejorar los tiempos para aplicación de descuentos del servicio móvil.
5. Generar pruebas en ambientes productivos que demuestren el comportamiento y la mejora de los tiempos en los procesos de aplicación de descuentos y divulgación de mensajes de texto.

Metodología

Dado el caso de estudio que involucra diferentes áreas de conocimiento o aplicación de profesiones de la empresa en división de desarrollo de aplicación, se debe implementar metodologías cuantitativas de investigación, ya que se requiere un paradigma empírico y analítico esto porque se deben tomar los requerimientos de personas que desconocen los ambientes de la programación de las nuevas tecnologías. [1]

Primero se adquiere un requisito para suprimir una necesidad se deben hacer pasos, procesos o seguimientos para adaptar la necesidad de la persona a la tecnología, cubrir de forma y diversa lo necesario, indagar sobre los procesos actuales para investigar cuál tecnología se adapta rápidamente a la solución. Por lo tanto, el método de investigación que se diseñó para implementar la solución a la problemática descrita en este documento se divide en 5 etapas.

1. Etapa de recolección de información

En el momento en que se presenta una necesidad de adaptar o mejorar un proceso es necesario la recolección de información de cómo se encuentra estructurado el proceso, cuáles son las funciones, que partes están involucradas y los tiempos de respuesta de cada función.

2. Etapa de análisis de la información recolectada

Para esta etapa se analiza la información recolectada y se estudia que herramienta tecnológica se puede adaptar a la necesidad de los involucrados, dando enfoque en que debe ser robusta por la alta contenido de datos, debe ser escalable para implementar nuevas funciones y debe ser optima en tiempos.

3. Etapa Diseño de la solución

En base a los análisis se diseña la estructura de la herramienta tecnológica dando énfasis a los principales requerimientos.

4. Etapa de implementación

Siguiendo las buenas prácticas del desarrollo de aplicaciones web y teniendo en cuenta la etapa de diseño se desarrolla la solución.

5. Etapa de pruebas

Para evidenciar que la herramienta implementada da solución a las necesidades se diseñan pruebas que contempla cada escenario.

Marco teórico

Uno de los objetivos principales de este desarrollo es implementar una aplicación web que permita a los trabajadores del área de marketing, diseñar y programar los mensajes de texto de divulgación de promociones por medio de una interfaz gráfica en forma de formulario. Principalmente se debe indagar y conocer los conceptos y funcionamiento que conllevar la transmisión de mensajes cortos SMS. Por tanto, en esta sección se detallan los componentes y funciones que permiten comunicar redes móviles mediante el protocolo WAP.

- Redes móviles GSM (Clobal System for Mobile Communication)

Desde que en el mundo se implementaron las redes de telefonía móvil se genero un gran reto de comunicación entre los países debido a que las redes no estaban estandarizadas y la frecuencia de uso era diferentes y causaba un Roaming indeficiente. Por tal motivo en en el año 2000 la ETSI le otorgo a la Asociación de proyectos de Tercera Generación (3GPP) el mantenimiento y la estandarización para la globalización de las redes GSM, estandarizando como las bandas de transmisión entre la 1800 y 1900 MHZ.

Así que las redes GSM se caracterizan por el transporte de voz digital, datos y envío y recepción de mensajería de texto. Y a su vez genero una infraestructura de red que se componen por: celdas, base de antenas satelitales, estaciones base y subsistemas de red. En la Figura 1 se ilustra una infraestructura básica de una red GSM.

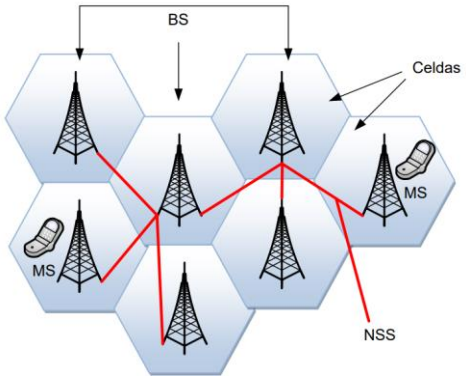


Figura 1, Infraestructura básica de una red GSM. Fuente: [3]

- SMS (servicio de mensajes cortos)

El servicio de mensajería corta consiste en transmitir y recibir mensajes con tasa no superiores a 160 caracteres, estos mensajes se transmiten especialmente a los equipos móviles, permitiendo una comunicación instantánea sin aplicación robusta. [2]

Siendo uno de los servicios principales de la telefonía móvil, el servicio de mensajes cortos tienes su propia infraestructura, en donde se compone por servidores SMTP, SMSC (Centro de servicio de mensajes cortos) este componente tiene la función de almacenar los

mensajes cortos a transmitir, Billing es un servidor que almacena el tráfico de mensaje para el cobro. En la Figura 2 se muestra un diagrama de la infraestructura SMS.

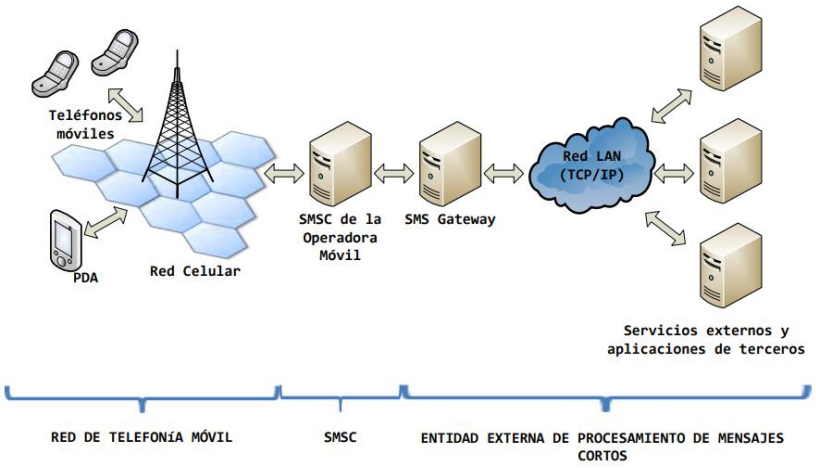


Figura 2, Diagrama de la infraestructura SMS. Fuente: [3]

Para el área de marketing de las empresas de prestación de servicio de telefonía móvil es una herramienta bastante útil para la divulgación de ofertas, promoción o mensajes relacionados con los estados de cuenta de los usuarios.

- Kannel

Es un código abierto para mensajería móvil y pasarela del protocolo WAP en redes móviles que permite que los celulares tengan conexión y navegación a internet, este tipo de conexión se realiza en velocidades bajas lo que es más útil para la mensajería de mensajes cortos (SMS). [3]

- Microservicios

Es un estilo de arquitectura enfocado para desarrollar aplicaciones mediante componentes independientes llamados servicios, donde cada servicio tiene la función de relacionar las actividades entre los servidores. Estas estructuras surgen con la necesidad de mantener los servicios con una unidad lógica con el uso de recursos computacionales compartidas. Este tipo de estructura garantiza que las aplicaciones sean más fáciles de escalar, desarrollar y reutilizar. [9]

- API REST

Cuando la estructura de la aplicación se basa en microservicios, ya que las funciones se dividen en componentes independientes. Esta estructura permite tener una comunicación y transferencia de datos entre el cliente, servidor y la base de datos.

En la figura 3 se muestra un breve esquema de comunicación entre cliente, servidor y base de datos mediante la API REST.

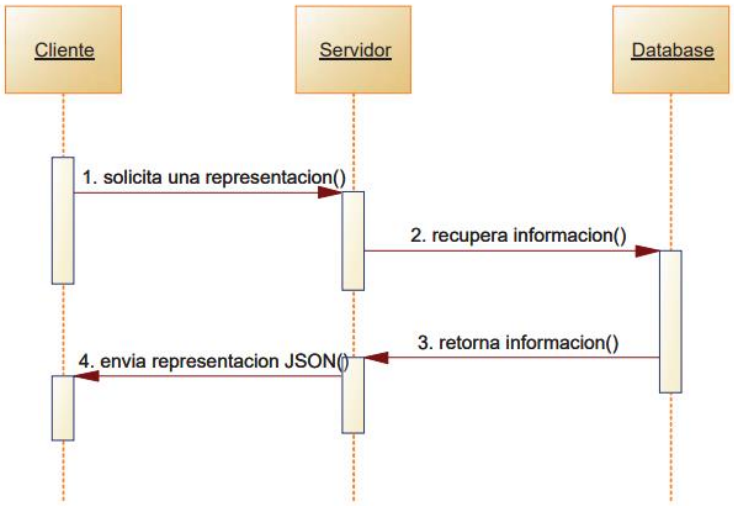


Figura 3. Funcionamiento API REST. Fuente: [9]

Las peticiones o representaciones del cliente a los servidores se hacen con una estructura JSON, la estructura JSON es un formato estándar con clave y valor. donde la clave es una propiedad del funcionamiento del servicio y el valor es el dato que el servidor espera para procesar y almacenarlo en la base de datos. [9]

Mediante las API se pueden solicitar diferentes actividades con los métodos HTTP, como el GET, POST, PUT, DELETE.

1. GET: Permite al cliente obtener información de la base de datos.
2. POST: Permite enviar información por medio de un JSON al servidor y registrar la información de la base datos.
3. PUT: Permite editar una propiedad especifica de la información obtenida.
4. DELETE: Permite eliminar registros de la base datos.

- Modelo MVC (Modelo Vista Control)

Es un patrón de diseño comúnmente usado para implementar interfaces de usuario, desarrollo frontend. La cual consiste en separar el negocio de la vista, o sea, que la lógica de las funciones de la vista está apartada de la vista, pero conectada. Esto permite obtener una mejor interpretación y división del código. [10]

El patrón MVC se compone de tres partes:

- Modelo: contiene los datos del negocio.

- Vista: Implementa el diseño visual y la representación de los datos.
- Control: Enruta los datos desde el modelo a la vista y contiene la lógica del negocio.

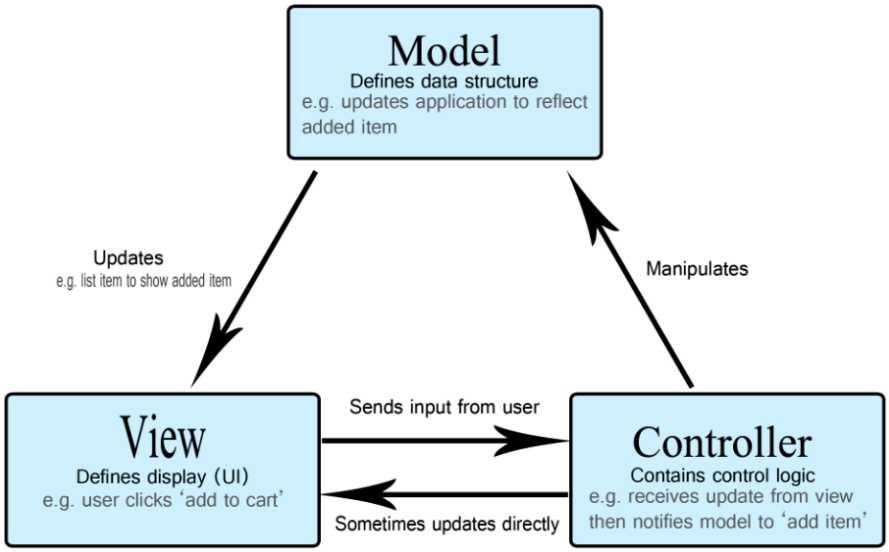


Figura 4. Esquema MVC. Fuente: [10].

- Visual Basic for Applications (VBA)

VBA es un entorno de desarrollo basado en el lenguaje de programación Visual Basic, implementado en los productos de Microsoft Office que permite crear nuevas funciones o acciones que van desde un mensaje emergente de recordatorio hasta capturar información procesar y generar una salida automática. Este entorno de desarrollo es usado de forma generar para automatizar u optimizar actividades incorporadas en las herramientas de Microsoft Office. [11]

- Base 64

Es una herramienta en línea que permite codificar y decodificar de formar rápida y sencilla archivos para transmitirse con JSON o XML. Este tipo de codificación garantiza que los datos permanezcan intactos protegiendo la integridad durante el transporte.

El proceso de codificación para un archivo a Base64 funciona de la siguiente forma:

1. La herramienta coge cada letra y la codifica en ASCII y la almacena.
2. Luego codifica los numero ASCII y los codifica en base 2 es decir a binario.
3. Una vez codificado en base 2 cada letra de una frase, estas se unen en un buffer de 64 bits.
4. Luego en el paquete de 64 bits se dividen en 6 bits, luego esos 6 bits son codificados en base 10 y multiplicado por el número 6.

5. El resultado de la multiplicación se codifica al número correspondiente de Base 64.

Contenido del texto	METRO				a			norte			
ASCII	77				97			110			
Patrón de bits	0 1 0 0 1 1 0 1				0 1 1 0 0 0 0 1			0 1 1 0 1 1 1 0			
Índice	19				22		5	46			
Codificado en Base64	t				W.		F	tu			

Figura 5. Ejemplo de codificación Base64. Fuente: [11]

Desarrollo

Etapa de recolección de información

Como parte del desarrollo de una mejora y la automatización de un proceso, hay que usar métodos de recolección de información como los procesos que se realizan para tratar datos para una determinada función. En este caso se hace uso de método de entrevista, ya que, en estos procesos se involucran diferentes áreas de la empresa que determinan los valores de entrada, los parámetros del proceso y los resultados de la salida de un determinado proceso.

Para el caso de la divulgación de mensajes de texto el proceso para programar los mensajes es el siguiente:

1. El inicio de la programación de la divulgación de mensajes de texto se da en el área de marketing, donde los especialistas de productos y servicios formulan la estructura de los mensajes según la campaña o la estrategia implementando. Una vez tenga los mensajes a divulgar, el área de marketing genera un requerimiento a una empresa tercera.

2. La empresa tercera recibe el requerimiento y por medio de un programa robusto de Kannel, registra los mensajes en la base de datos y haciendo uso de los servidores de WOM para despliegue de mensajes de texto. Siendo que la inserción de los nuevos mensajes se den hacer una base de datos, los cambios se ejecutan después de un horario no pico (10 p.m.).

3. Finalmente, en el área de desarrollo de aplicaciones se encarga de generar los URL, en donde se integra los mensajes de las tablas de datos y los números de teléfonos a donde serán dirigido los mensajes.

La empresa tercera como el área de aplicaciones mantiene una carga de trabajo alta, por lo que los requerimientos son atendidos por nivel de prioridad y por orden de creación.

En la Figura 3 se ilustra el flujo del proceso de la divulgación de ofertas por medio de mensajes de texto.

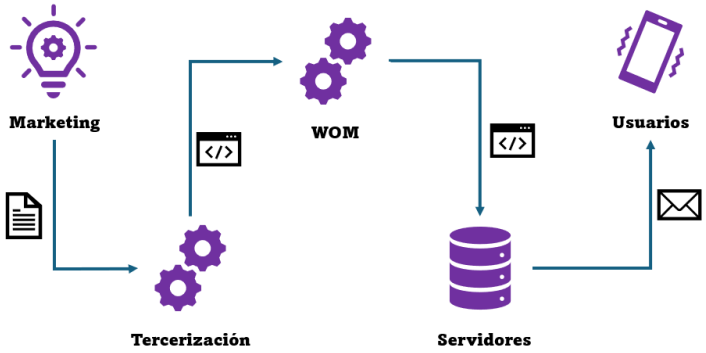


Figura 6. Flujo de proceso de programación de divulgación de mensaje de texto. Fuente: Elaboración del autor.

Para el caso de la aplicación de descuentos en los planes de servicios móviles, se hace por medio de actualización de datos en la base datos, por lo que el flujo de trabajo es:

1. La estrategia del área de marketing es aplicar descuentos a partir del flujo de ventas obtenidas por semana. Por lo tanto, el área de mercado haciendo uso de una matriz diseña en la herramienta de Excel planea los descuentos a realizar dependiendo los siguientes parámetros:
    a. Score (puntuación del usuario en DataCredito).
    b. Canal (Por que medio realizo la compra ej: tienda virtual o presencial).
    c. Tipo de cedula (NIT o cedula de ciudadanía).
    d. Departamento (Costa o resto del país).
    e. Grupo (Configuración de la base de datos).
    f. Tipo plan de servicio de telefonía móvil.

2. Luego de que el área de marketing diseña la estrategia de descuentos, pasa a manos de un especialista de productos pospagos, la cual tiene la función de validar y estudiar las posibilidades de aplicar los descuentos según la matriz.

3. Una vez la matriz de descuentos se encuentra completa es enviada al área de configuraciones y aprovisionamiento, donde un especialista en base a los descuentos de la matriz genera un código de descuento que son los datos actualizar en la base de datos y especifica el grupo de planes, una vez genera los códigos los ingresa en la matriz del documento Excel.

4. El área de configuración y aprovisionamiento envía la matriz de descuentos al área de desarrollo de aplicación con un analista de software y en base a los parámetros establecidos y el código de descuento genera comandos SQL de actualización.

5. Luego de generar los comandos SQL, el área de desarrollo solicita al área de soporte de aplicación FRONTEND levantar un caso de cambio en la base datos con

el objetivo de que el área de soporte de la empresa INDRA tome el caso y lo programe para la ejecución.

6. Finalmente, el área de soporte de INDRA en las horas no pico ejecutan los comandos SQL en la base de datos y reportan los cambios al área de soporte de WOM, para posteriormente documentar los cambios y verificar que los cambios se realizaron de forma correcta.

En la Figura 4 se ilustra el flujo del proceso realizado para generar los descuentos en los planes del servicio de telefonía y se ofrecen en cada sucursal o canal de ventas.

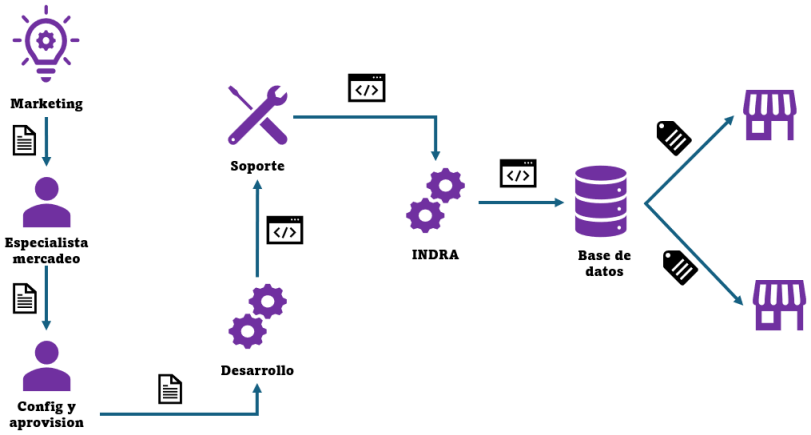


Figura 7. Flujo de proceso para aplicar descuentos en los planes del servicio móvil. Fuente: Elaboración del autor.

Etapa de análisis de requerimientos

En base a la información recolectada frente a los dos procesos de actualización de datos y la divulgación de mensajes de texto, se comprende una solución tecnológica para cada proceso.

Para el caso de la divulgación de mensajes de texto se plantea la integración de una plataforma web con funciones de acceso a usuario con un determinado rol, formulario en donde se capture la información requerida, programación de los mensajes, panel de control que permita administrar el estado de las programaciones, reprogramación de fecha y envío de información por medio de peticiones JSON al BackEnd para el proceso de datos y actualización en base de datos.

Mientras para el proceso de actualización de datos para aplicar descuentos, se adapta una automatización con la herramienta Excel, ya que en el proceso hay un documento único que debe pasar por las diferentes áreas para ser diligenciado con determinada información y no es necesario el desarrollo de una herramienta para un resultado de poco proceso.

En el caso de la programación de divulgación de promociones por medio de mensajes de texto, se implementa una plataforma web con las siguientes características en la infraestructura:

5. Para el FrontEnd se desarrolla con TypeScript con el framework Angular, se hace uso de esta tecnología porque contiene la estructura MVC (Modelo, Vista, Control) esto permite que código sea escalable, la estructura se construye en base a componentes permitiendo la comunicación de la vista HTML con el control TS sea más rápida, permitiendo ser reutilizables o globales y permite ser una compilación más ágil.
6. Dividir las funciones de la aplicación con microservicios alojados en contenedores Docker, donde el consumo de esos servicios se realiza con petición JSON. Se estructura de esta manera aprovechando que la empresa cuenta con servicios generales como el login, captación de información y procesamiento de datos sensibles.
7. Para el caso de BackEnd se utiliza lenguaje JAVA, debido que en la compañía cuenta con otra área que es la encarga de desarrollar todos los recursos que se encuentran en comunicación con los servidores.
8. Para el caso del almacenamiento de los datos, la empresa cuenta con el motor Oracle, la cual ya se cuenta con la estructura de las tablas para la aplicación por lo que el sistema tercerizado hace uso de los servidores de la empresa, pero el software de implementación Kannel es ajeno de la empresa Partners Telecom Colombia S.A.S.

Para el caso de actualización de descuentos para aplicar descuentos a los planes de servicio se plantea automatizar la plantilla de la herramienta Excel, por medio del desarrollo de funciones y macros bajo el lenguaje de programación Visual Basic para aplicaciones (VBA). La estructura de este código solo se compone de métodos y funciones que obtenga la información de las celdas, procese los datos y exporte archivos con la extensión .sql con el fin de que simplemente el área de desarrolle con un botón exporte el conjunto de comandos SQL de actualización y programe la ejecución de los cambios.

Etapa de diseño de solución

De acuerdo con los requerimientos y necesidades analizadas con el desarrollo de una plataforma web para la divulgación de promociones por mensajes de texto se diseña la plataforma en donde se demuestre los datos almacenados en la base de datos que se encuentran alojados los id Kannel, programación, plantilla y permite la posibilidad de adjuntar una nueva plantilla de mensaje por medio de un archivo xlsx.

Teniendo en cuenta que actualmente solo el software Kannel es el tercerizado y la base de datos es de la compañía, el diseño de la base de datos, como algunos servicios, ya se encuentran se encuentra implementados, por lo que el diseño de plataforma se enfoca en la parte de frontend y por políticas de seguridad de la empresa, no se puede exponer el diagrama de relaciones de la base de datos y las clases de los servicios del backend.

Historias de usuario

Para iniciar con el diseño de la solución se debe crear historias de usuario en donde se establecen los Actor/roles, funciones, criterios y eventos por la cual van a hacer la interacción entre el usuario y la maquina con diferentes funciones que permita cubrir con las necesidades del proyecto.

Tabla 1. Historia de usuario de iniciar sesión. Fuente: Elaboración del autor.

Evento	Iniciar Sesión.
Actor/Rol	Visitante anónimo.
Descripción	Como visitante deseo iniciar sesión a la plataforma web.
Criterios	El visitante debe contar con usuario y contraseña registrado.
	El visitante debe estar autorizado para iniciar sesión en la plataforma.
	Las credenciales del usuario deben ser correctas.

Tabla 2.Historia de usuario de cerrar sesión. Fuente: Elaboración del autor.

Evento	Cerrar Sesión.
Actor/Rol	Administrador y usuarios.
Descripción	Como usuario que inicio sesión desea cerrar sesión en la plataforma.
Criterios	El usuario debió iniciar sesión previamente en la plataforma.
	El usuario debe confirmar la programación de divulgación de mensajes de texto.
	El usuario debe confirmar la cancelación de la programación de la programación de la divulgación de mensajes de texto.

Tabla 3. Historia de usuario Ver lista de programación de divulgación de mensajes. Fuente: Elaboración del autor.

Evento	Ver lista de programación de mensajes de texto y el estado de cada uno.
Actor/Rol	Administrador.
Descripción	Como usuario administrador puede visualizar en una lista las programaciones de divulgación de mensajes realizado por otros usuarios.
Criterios	El administrador debe contar con los privilegios para ver la lista de programaciones.
	Por lo menos debe haber registro de una programación de divulgación.

Tabla 4. Historia de usuario de Cambiar el estado de las programaciones de divulgación de mensajes de texto. Fuente: Elaboración del autor.

Evento	Cambiar el estado de programación de divulgación.
Actor/Rol	Administrador.

Descripción	Como usuario administrador puede cambiar el estado de las programaciones como: • Programado. • Pendiente. • Enviado. • Cancelado.
Criterios	El administrador debe contar con los privilegios para cambiar el estado de las programaciones.
	Las programaciones deben contar con la fecha de vencimiento vigente.

Tabla 5. Historia de usuario de Reprogramar fecha de la programación de divulgación de mensajes de texto. Fuente: Elaboración del autor.

Evento	Reprogramar fecha de la programación de divulgación.
Actor/Rol	Administrador.
Descripción	Como usuario administrador puede reprogramar la fecha de divulgación de mensajes de texto.
Criterios	El administrador debe contar con los privilegios para cambiar el estado de las programaciones.
	Las programaciones deben contar con la fecha de vencimiento vigente.

Tabla 6. Historia de usuario de modificar mensaje de la programación de divulgación. Fuente: Elaboración del autor.

Evento	Modificar mensajes de la programación.
Actor/Rol	Administrador
Descripción	Como usuario administrador puede modificar los mensajes de las programaciones de divulgación.
Criterios	El administrador debe contar con los privilegios para modificar los mensajes de las programaciones.
	La programación debe contar por lo menos con un mensaje.

Tabla 7. Historia de usuario de eliminar programación de divulgación de mensajes. Fuente: Elaboración del autor.

Evento	Eliminar programación de divulgación de mensajes.
Actor/Rol	Administrador.
Descripción	Como usuario administrador puede eliminar las programaciones de divulgación insertadas por otros usuarios.
Criterios	El administrador debe contar con los privilegios para eliminar las programaciones insertadas.
	Por lo menos debe haber un registro de una programación de divulgación.

Tabla 8. Historia de usuario de crear programación de divulgación de mensajes. Fuente: Elaboración del autor.

Evento	Crear programación de divulgación de mensajes.
Actor/Rol	Usuario de marketing.
Descripción	El usuario de mercadeo puede programar la divulgación de mensajes.
Criterios	El usuario debe ingresar todos los campos del formulario como: • Kannel. • Programación. • Plantilla. • Adjuntar archivo Excel. • Fecha de programación.
	El usuario debe accionar el botón de programar para registrar la programación de mensajes.

Tabla 9. Historia de usuario de cancelar valores del formulario. Fuente: Elaboración del autor.

Evento	Cancelar los valores del formulario.
Actor/Rol	Usuario de marketing.
Descripción	El usuario de mercado puede cancelar el formulario que se encuentra diligenciando.
Criterios	El usuario por lo menos debe diligenciar alguno de los campos del formulario.
	El usuario debe accionar el botón cancelar.
	El usuario debe aceptar el mensaje de confirmación de cancelación.

Para el caso de la automatización de la actualización de datos para aplicar descuentos a los planes del servicio móvil, la solución implementada solo se presenta para un actor con única función.

Tabla 10. Historia de usuario de exportar archivos SQL. Fuente: Elaboración del autor.

Evento	Exportar archivo SQL
Actor/Rol	Miembro de soporte
Descripción	El miembro de equipo quiere generar archivos SQL para ejecutar cambios en la base de datos.
Criterios	Las tablas o matrices deben contar con el número de grupo a aplicar los descuentos.
	Los campos de descuentos deben estar diligenciados por el área de marketing.
	Los campos de CODE IPP deben estar diligencias por el área de configuración y aprovisionamiento.
	El miembro de soporte debe accionar el botón de exportar para generar los archivos.

Diagramas de Casos de uso

Según las historias de usuario, se plantean dos roles o tipos de usuarios para la implementación de herramienta para la programación de divulgación de promociones, como el usuario administrativo y el usuario de marketing, que tienen diferentes funciones y privilegios para realizar acciones programando divulgaciones.

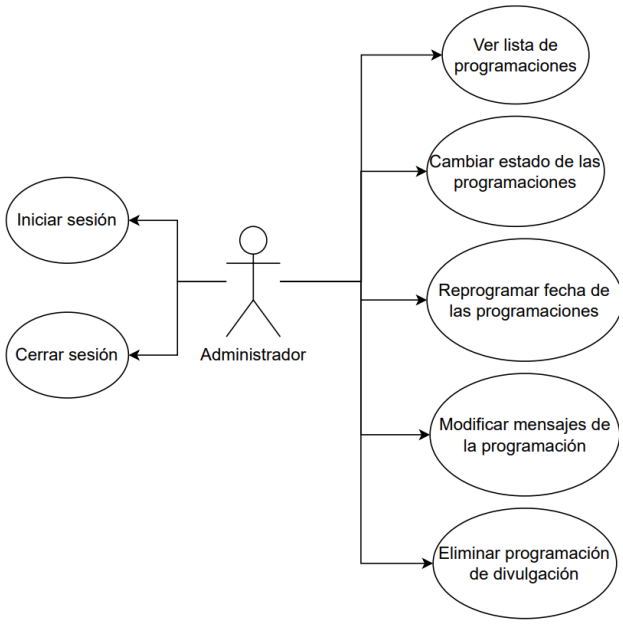


Figura 8. Diagrama de uso de administrador. Desarrollo web para programación de divulgación de mensajes de texto. Fuente: Elaboración del autor.

El usuario administrativo para el aplicativo web de programación de divulgación de mensajes de texto tiene acciones y funciones relacionadas con la administración de las programaciones de las divulgaciones de mensajes, las acciones van desde modificar hasta eliminar las programaciones.

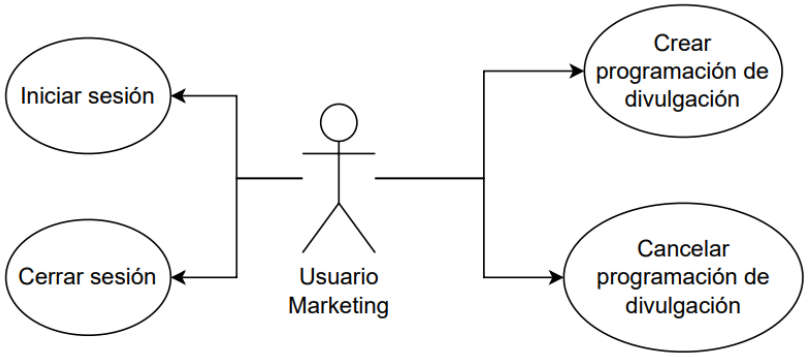


Figura 9.Diagrama de uso de usuario marketing. Desarrollo web para programación de divulgación de mensajes de texto. Fuente: Elaboración del autor.

El usuario de marketing para el aplicativo web de programación de divulgación de mensajes de texto tiene dos clases de funciones como lo es la creación y la creación de la programación de la divulgación de los mensajes.

Para el caso de la automatización del proceso de actualización de datos para aplicar descuentos, se considera solo los miembros de soporte dado que es por parte de esa área que se ejecutan los cambios en la base de datos.

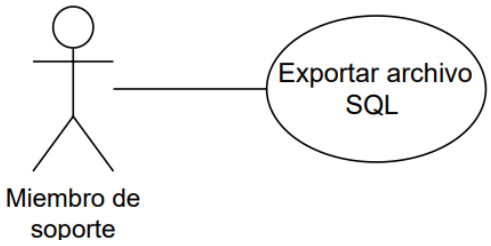


Figura 10. Diagrama de uso de miembro de soporte. Automatización para actualización de datos para aplicar descuentos. Fuente: Elaboración del autor.

Los miembros de soporte tienen la acción de ejecutar el botón de exportar lo que permite exportar los archivos .sql para realizar los cambios en la base de datos.

Etapa de implementación de la solución

De acuerdo con el análisis previo a los requerimientos de mejora para el caso de la implementación de la plataforma web se desarrolla la aplicación con la herramienta tecnológica o framerwork Angular 9 que es basado en el lenguaje de programación de programación de Typescript. Este framework se escogió por la escalabilidad y robustes que tiene al momento de crear aplicaciones web porque cuenta con características como el módulo enrutamiento que permite la navegación entre las páginas y en donde cada una de las páginas se denomina componentes estructurados por el modelo MVC.

El enrutamiento de las páginas/componentes lo realiza concatenación de la URL y el nombre de cada componente. En la Figura 8 se ilustra el esquema principal de la estructura del framework Angular 9.

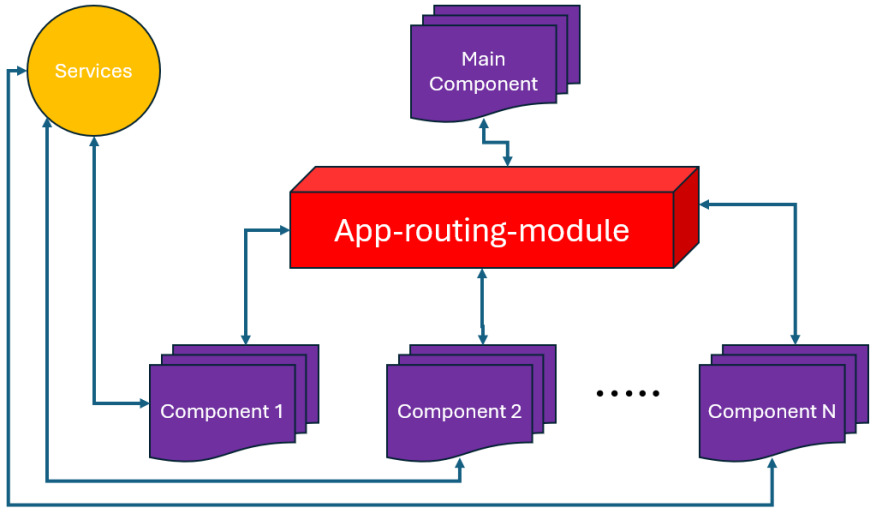


Figura 11. Esquema de estructura del framework Angular. Fuente: Elaboración del autor

Como punto inicial de la creación del proyecto y como es un ambiente de pruebas, se debe instalar las dependencias y los componentes necesarios de angular. Como angular es un framework basado en typescript es necesario la instalación del Node JS de seguido de la instalación de Angular.

Desde la página oficial de Node JS se descarga la última versión y se instala en la computadora, una vez se tenga instalado el Node JS, desde el CMD (Consola de línea de comandos) se instala los paquetes npm y el angular CLI.

```
npm install -g @angular/cli
```

Figura 12. Instalación de angular CLI desde línea de comandos. Fuente: [7]

Luego de la instalación del angular CLI se puede verificar si se instaló correctamente usando el comando ng -version en donde el resultado debe ser la version instalada. El proyecto angular se crea después mediante líneas de comando usando el comando ng new de seguido del nombre del proyecto.

```
ng new my-app
```

Figura 13. Comando para crear un nuevo proyecto angular desde la línea de comando. Fuente: [7]

Una vez se tenga creado el proyecto, se continua con la creación de los componentes que van a conformar cada una de las páginas del aplicativo, para el caso de la implementación se van a crear 7 componentes. En la Tabla 11 se relaciona los nombres de los componentes con la función que va a realizar en la aplicación.

Tabla 11. Componentes del proyecto con sus respectivas funciones. Fuente: Elaboración por el autor.

Componente	**Función**
Login	Contiene la vista iniciar o el formulario en donde los actores se podrán iniciar sesión. Consume el servicio de login y captura el token para autenticación de los actores.
Sms-form	Contiene el formulario con los campos requeridos para programar la divulgación de mensajes. Consume el servicio de VALIDATE USER, la cual trae información desde la base de datos. Consume el servicio de ENTER FILE, la cual captura los datos del formulario y los envía al backend.
Page-admin	Contiene la vista de la lista de las programaciones. Tiene las acciones de modificar mensajes, reprogramar y eliminar programaciones.
dialogError	Contiene cajas de dialogo que muestra mensajes del sistema.
Toast	Contiene cajas de dialogo para mostrar mensajes de error producidos desde backend.
HandleHttpResponseError	Captura los logs de error producciones desde backend.

En el momento que se generar un componente, el framework genera 4 archivos: HTML, CSS, TS Y SPEC.TS. El diseño de esta estructura está pensado para que cada vista, tenga su propio diseño de HTML enlazado con sus propios estilos CSS y su propio controlador TS.

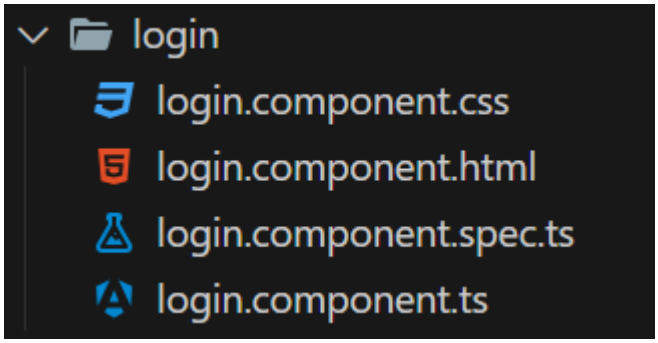


Figura 14. Estructura de componentes. Fuente: Elaboración del autor.

Una vez se crean los componentes para las vistas, se procede a generar los servicios, como se mencionó en la fase de análisis, se reutiliza algunos servicios por medio de peticiones http y JSON para tener interconexión con funciones del backend. Por lo tanto, se crean microservicios en el proyecto por la cual se implementan funciones para que el proyecto de angular pueda realizar peticiones http a los microservicios.

Luego de generar los servicios en el proyecto se deben generar interfaces estas permiten la interconexión o transporte de datos entres cada uno de los componentes y cada servicio. Finalmente, de generar los elementos necesarios para el desarrollo de la aplicación, la estructura del proyecto es como se muestra en la Figura 12.

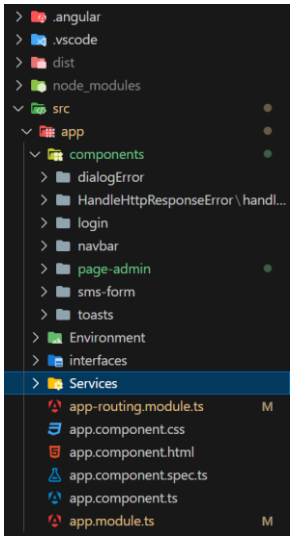


Figura 15. Estructura del proyecto angular. Fuente: Elaboración del autor.

Implementación de la vista de inicio de sesión

Para la implementación de esta vista se deben tener varios aspectos importantes, como lo son:

- Se debe visualizar el logo de la empresa con el eslogan.
- El diseño de fondo debe mantener los colores corporativos, como lo es el morado oscuro y pequeñas escalas para sombras.
- Como es de uso interno se debe visualizar el nombre de la empresa.
- Para el control de versiones se debe mostrar la version del aplicativo.
- Nombre del aplicativo.
- Título de la vista.
- Usar fuente de letra de acuerdo con la corporación.

Según los requerimientos para considerar, se diseña la pantalla de inicio de sesión como se muestra la Figura 13.

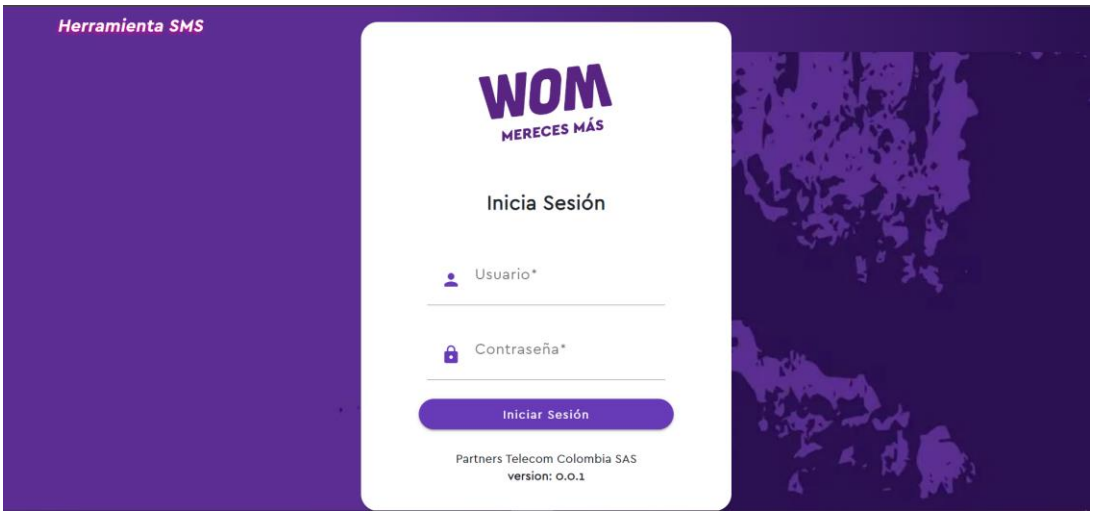


Figura 16. Vista Inicio de sesión. Fuente: Elaboración del autor.

Para el inicio de sesión, la empresa tiene un repositorio donde alojan imágenes o figuras que pueden usarse para fines internos de la corporación y comerciales, lo que selecciona una de las imágenes para el fondo de la vista.

En la parte superior se usan un DIV para visualizar el nombre de la aplicación (Herramienta SMS) y con comandos CSS se implementa un estilo de sombra de color fucsia.

Para el formulario se hace uso de angular materia y la importación de las bibliotecas *FormBuilder*, *FormControl* y *FormGroup*. Con ayuda del framework Bootstrap se usa la división de grillas para ubicar en la parte superior el logo de WOM con su logotipo, de seguido del nombre de la vista (Iniciar Sesión), más abajo se encuentra el formulario en donde el usuario podrá ingresar el usuario y la contraseña y posteriormente con un botón de color morado y un estilo CSS de *hover* que indica que el usuario se encuentra sobre el botón con el puntero accionar la función de iniciar sesión.

En la parte de abajo del DIV del formulario se visualiza el nombre de la empresa y la versión de desarrollo.

Después de desarrollar la vista se implementa las funciones, para ellos se usa el archivo del componente que termina en TS, en este archivo es donde se va a controlar el comportamiento de la vista con comandos typescript. Para iniciar se debe tener encuentra la estructura del archivo.

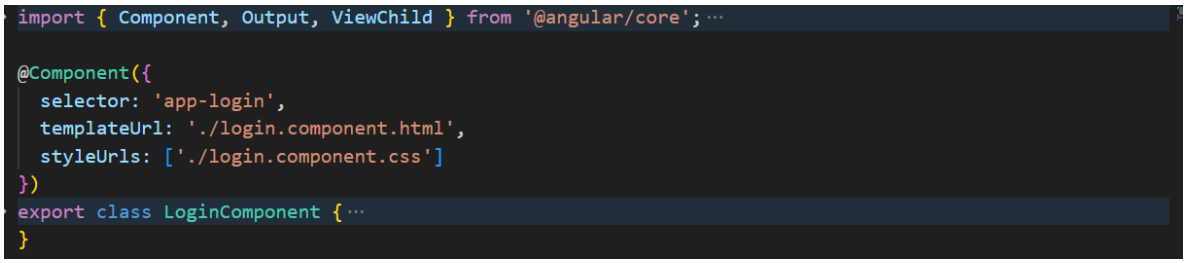


Figura 17. Estructura del archivo TS. Fuente: Elaboración del autor.

En la parte superior se importan todas las bibliotecas o dependencias haciendo el uso de la palabra reservada *import*.

Luego se tiene la sección de los componentes, que es donde se relaciona el *selector* que es el nombre por la cual se va a hacer el enrutamiento, el *templateUrl* que es el archivo HTML y el *styleUrls* que es el archivo CSS.

Por último, se encuentra la función *export class* que es donde se van a implementar las funciones de comportamiento de la vista.

En dado caso que se necesite hacer uso de algún servicio o funcionalidad que se encuentre en otra parte del proyecto, se debe crear una variable en la función *constructor*, ya que typescript usa inyección de dependencias.

Implementación de la vista de formulario para programación de mensajes

Para el caso de la vista en donde los usuarios diferentes al administrador podrán ingresar los datos requeridos para programar la divulgación de mensajes. Se diseña los campos haciendo uso de librerías de angular material como *FormBuilder*, *FormControl* y *FormGroup.*

Con ayuda del framework Bootstrap se diseña el campo del formulario con el uso de grillas para posicionar de forma proporcional cada uno de los campos. Para se hace su de la propiedad *col-md-12,* esta propiedad es usada en un DIV con la palabra reservada *class,* esta propiedad permite que el DIV se divida en 12 columnas.

Una vez se tiene un contenedor o DIV divido en 12 columnas se posiciona los *inputs* para capturar los datos para la programación de los mensajes. De a acuerdo a los requerimientos de ingresa un total de 4 *inputs,* antes de implementar los campos de ingreso de datos (input) se debe hacer uso de DIV con las propiedades *row mb-2* cada uno, esto permite que ese campo tenga un ancho de 2 filas de las 12 del contenedor principal. En la figura 15 y 16 se ilustra la división y posición de cada campo de ingreso de datos. Mirar Anexo A.

```
<div class="col-md-12">
    <img src="../../../assets/images/WOM-blanco(400x200).png" class="img-fluid mx-auto text-ce
    <form [formGroup]="formSubmit" (ngSubmit)="submitForm()" #campaignForm="ngForm">
        <!-- Select Kannel -->
        <div class="row mb-2"> ···
            </div>

        <!-- Select Programacion -->
        <div class="row mb-2">···
        </div>

        <!-- Select Plantilla -->
        <div class="row mb-2">···
        </div>

        <div class="row mb-2">···
        </div>
    </form>
</div>
</div>
```

Figura 18. Código de distribución de los campos input. Fuente: Elaboración del autor

Figura 19. Vista de la distribución de los inputs. Fuente: Elaboración del autor

Los inputs tienen la capacidad de capturar diferentes tipos de datos como lo es desde texto a un archivo, para este caso se hace uso de *DownDropList* que en las propiedades de angular se hace uso de *mat-select* junto con la palabra reservada **ngFor* permite mostrar los datos de una lista por medio de una lista desplegable, ya que en cada uno de los usuarios contiene conjuntos de lista de información Kannel, programación y plantilla. Para el caso de adjuntar un archivo el input debe ser de tipo *File* y limitar que solo permita archivos xlsx.

Como parte del funcionamiento de la vista del formulario es enviar la información a las funciones del back, por lo que se implementan botón para realizar acciones, como lo es el:

- Guardar y Enviar: Captura los datos ingresados en el formulario para posteriormente enviarlos al backend por medio de una petición JSON.
- Cancelar: Borra los datos ingresados en el formulario y posteriormente retorna a la vista de inicio de sesión.
- Programar: Visualiza un modal con un calendario en donde el usuario puede ingresar la fecha que desea programar la divulgación de mensajes.
- Guardar: Captura los datos del formulario y guarda los datos en la base de datos, pero no realiza una programación.
- Salir: Permite retornar a la vista de inicio de sesión y limpia los campos del formulario.

Para el diseño de los botones se hace uso de comandos CSS con el fin de lograr los colores bordes redondos, y *hover* para el cambio de color cuando el puntero del ratón este haciendo foco al botón.

Figura 20. Botones de la vista de formulario. Fuente: Elaboración del autor

Finalmente, el diseño de la vista de formulario de divulgación de mensajes mantiene los criterios iniciales de la vista de iniciar sesión con el fin de mantener una consistencia en el fondo de las vistas y el nombre de la aplicación.

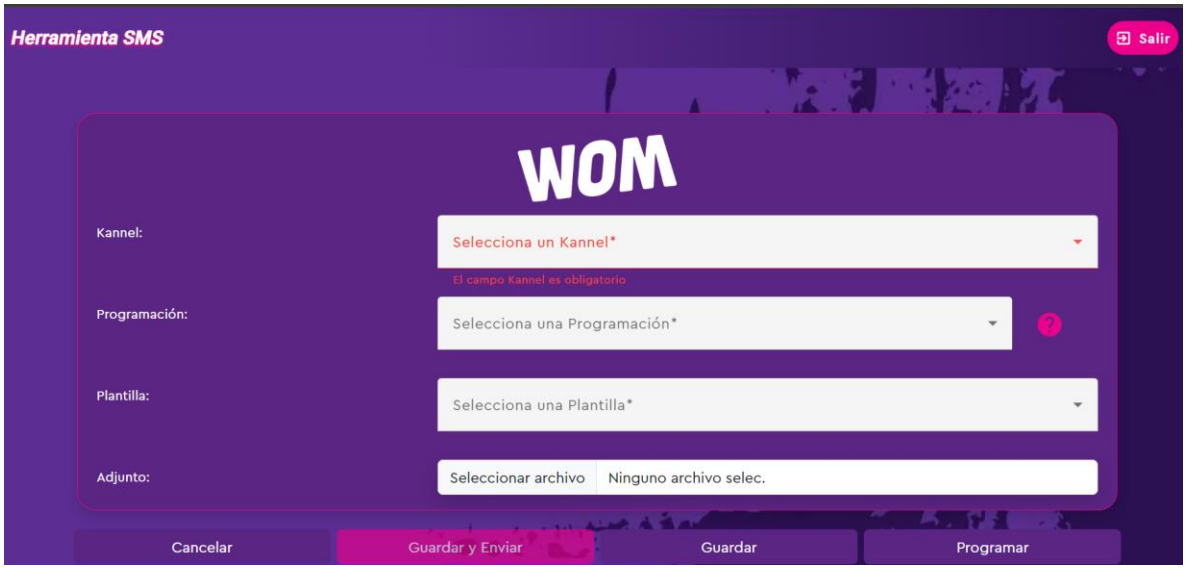


Figura 21. Vista de formulario de programación de divulgación de mensajes. Fuente: Elaboración del autor.

Implementación de la vista de la lista de programaciones

Como parte de los funcionamientos del proyecto es visualizar la lista de las programaciones registradas y su administración se desarrolla la vista del administrador donde podrá visualizar, modificar, reprogramar y eliminar las programaciones.

Para esta vista el principal componente va a ser una tabla, filtro de búsqueda y la paginación de la tabla ya que solo se muestra 10 registros por vista. Con el fin de cumplir el propósito de mostrar los registros en una tabla, se hace uso de la propiedad *table* y *mat-table* de angular.

Para los encabezados de las columnas se usan los títulos de: Kannel, Programación, Plantilla, Adjunto y opciones. Para las opciones se usan iconos para determinar las posibles acciones a realizar sobre cada registro.

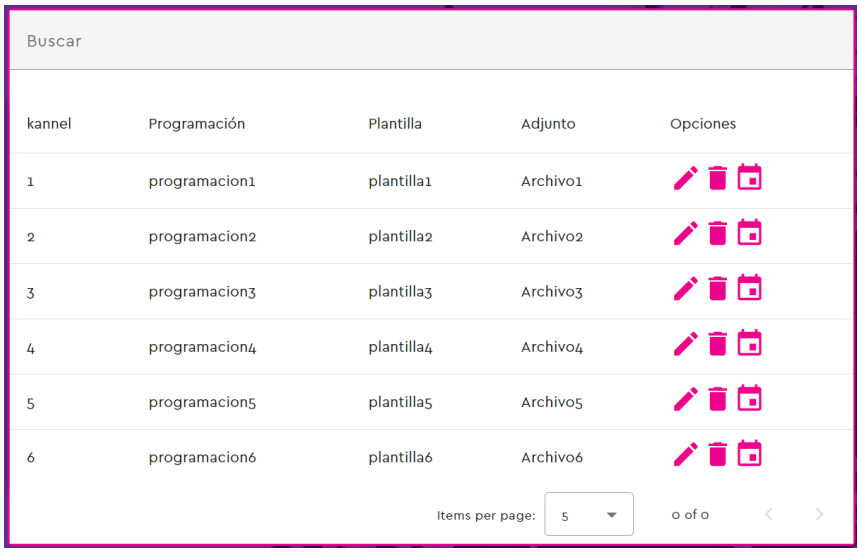


Figura 22. Tabla de registro de programaciones de divulgación de mensajes. Fuente: Elaboración del autor.

Microservicios para funcionamientos de la solución

Como se mencionó en secciones anteriores la empresa cuenta con microservicios y la estructura del proyecto determina que la comunicación entre el frontend y backend se realiza por medio de microservicios con peticiones JSON.

Antes de desarrollar el funcionamiento de cada una de las vistas del frontend se debe conocer la estructura de los JSON para identificar cuáles son los datos obligatorios u opcionales.

Como se sabe para el consumo de microservicios desde una aplicación se realiza con API REST y la API contesta o responde con una estructura JSON. Por policías de la empresa solo se hace peticiones POST para el consumo de APIs.

- Microservicio de autenticación

Como parte de las políticas de seguridad y buenas prácticas para el desarrollo de aplicaciones web, el microservicio de autenticación se desarrolló con la finalidad de que en el momento que un usuario y contraseña sean correctos se genere un token único para cada inicio de sesión.

La estructura JSON para la autenticación se divide en dos secciones, la sección *init* y *body* en donde la sección *init* contiene las propiedades *isRedirect* es la que identifica que tipo de petición y la propiedad *query* que identifica cual es el tipo de servicio que se está haciendo la petición.

Luego se tiene la sección *body,* esta sección contiene las propiedades que serán enviadas al backend para realizar procesos y almacenamiento en la base de datos.

Figura 23. Estructura JSON para autenticación de inicio de sesión. Fuente: Elaboración del autor.

La anterior figura muestra la estructura JSON que la API del microservicio de autenticación de sesión espera recibir. Para el caso de este microservicio las propiedades del JSON definen:

- isRedirect: Tipo de aplicación.
- Query: Nombre del microservicio.
- origin: nombre del aplicativo backend a donde va dirigido el servicio.
- ipClient: Dirección IP del servidor donde se encuentra alojado el servicio backend.
- thirdPart: Contiene las propiedades que requiere la base de datos para hacer consultas o actualizaciones.

Haciendo usa de la herramienta POSTMAN, ingresa la estructura del JSON y la dirección IP de la API, se puede visualizar la respuesta de la petición enviada y de esa manera conocer el token (jwt) de autenticación, código de respuesta, mensaje y mensaje técnico de la respuesta. Estos últimos son usados para soporte en dado caso de falla.

Figura 24. Herramienta POSTMAN con la respuesta de la petición. Fuente: Elaboración del autor.

- Microservicio de validación de usuario

Cuando se genera el token de autenticación, el servicio para validar el usuario y obtener la lista con los datos necesarios para ver el formulario en la petición JSON en la cabecera, se debe enviar el token generado y obtener las listas necesarias.

Figura 25. Petición JSON y respuesta JSON del microservicio para validar el usuario. Fuente: Elaboración del autor.

- Microservicio ingresar datos

Este microservicio va a permitir encapsular todos los datos capturados en el formulario de la vista en donde se programan la divulgación de mensajes y posteriormente hacer un tratamiento de datos en el backend y actualizar la base datos.

Figura 26. Petición JSON y respuesta JSON del microservicio de ingresar datos. Fuente: Elaboración del autor.

Implementación de los servicios en el proyecto angular

Antes de desarrollar código para implementar los servicios en cada una de las vistas, en el comento que creamos el proyecto angular se generó un archivo llamado *enviroment* es el lugar donde se define las URL y los endpoint. Para la solución una de las URL es la dirección IP y el puerto de la API que vamos a consumir y dos endpoint que hacer referencia al inicio de sesión ya que solo necesitamos ingresar en la cabecera el tipo de aplicación JSON y el otro endpoint hace referencia al token ya que en la cabecera debe ir Access-Control-Security que es el token generado.

Luego de implementar el ambiente con la definición de la URL de la API y los endpoint para los servicios continuamos con el desarrollo de los servicios, esto se implementa en los archivos generados al inicio del proyecto.

En el servicio global se hace uso de la biblioteca *HttpClient* por medio de inyección de dependencia permite realizar peticiones http por medio de los métodos get, post, put o delete. Cada uno de estos métodos tiene parámetros de entra como el paso del POST se debe ingresar la URL, los parámetros y la cabecera.

En la figura 24 encapsula el encabezado dependiendo el endpoint que se utilice en el momento de hacer uso del servicio. Si el endpoint es login solo encapsula la propiedad *Access-Control* con la respectiva IP y si el endpoint es token encapsula las propiedades de acceso al control y acceso al control de la seguridad con el token generado por el microservicio *VALIDATE_CODE.* Mirar Anexo B.

```
export class GlobalServicesRequestService {

  constructor(private httpClient: HttpClient) {

  }

  postRequest(petition:any, parameters:any): Observable<any>{
    var headers = {};

    if(petition == environment.endpointLogin){
      headers = {'Access-Control'               : '127.0.0.1'};
    }else if(petition == environment.endpointToken){
      headers = {
        'Access-Control'               : '127.0.0.1',
        'Access-Control-Security'      : localStorage.getItem('token')
      };
    }

    console.log('headers',headers)
    console.log(parameters)

    return this.httpClient.post(environment.url, parameters, {observe:'response', headers:headers});
  }
```

Figura 27. Uso del método POST de la dependencia httpClient. Fuente: Elaboración del autor.

- Servicio de autenticación en la vista de inicio de sesión

Lo primero para hacer uso del servicio de autenticación es generar un nuevo servicio en el proyecto que permita encapsular y canalizar las peticiones y respuestas de la API, para ello hacemos uso las bibliotecas *rxjs* con las funciones *Observable* y *map* con esta función se crea una clase de tipo *Observable* con parámetros de entrada del endpoint y parámetros que retorna el servicio global en forma de canalización. Mirar Anexo C.

```
export class LoginUserService {

  constructor(private globalServiceRequest: GlobalServicesRequestService) { }

  loginQuery(parameters: IloginRequest): Observable<any>{

    return this.globalServiceRequest.postRequest(environment.endpointLogin, parameters).pipe(
      map((response: any)=> {
        console.log(response.body);
        console.log(response.headers);

          return response.body;
      }),
      catchError(HandleHttpResponseError)
    );
  }

}
```

Figura 28. Consumo del servicio de autenticación. Fuente: Elaboración del autor.

Luego de la creación del servicio en el proyecto se debe capturar los datos ingresados en el HTML de los inputs haciendo uso de los formularios reactivos de angular para ellos en los inputs hacemos uso de la palabra clave *formControlName* de seguido de un nombre que identifique el campo.

Una vez que se tiene capturado los datos desde la vista en el controlador, se crea una clase asíncrona que tendrá la función de encapsular la información captura y retornar una variable de tipo *subscribe* ya que la clase creada es de tipo *Observable* permite des-encapsular y mostrar los elementos que responde la API. Mirar Anexo D.

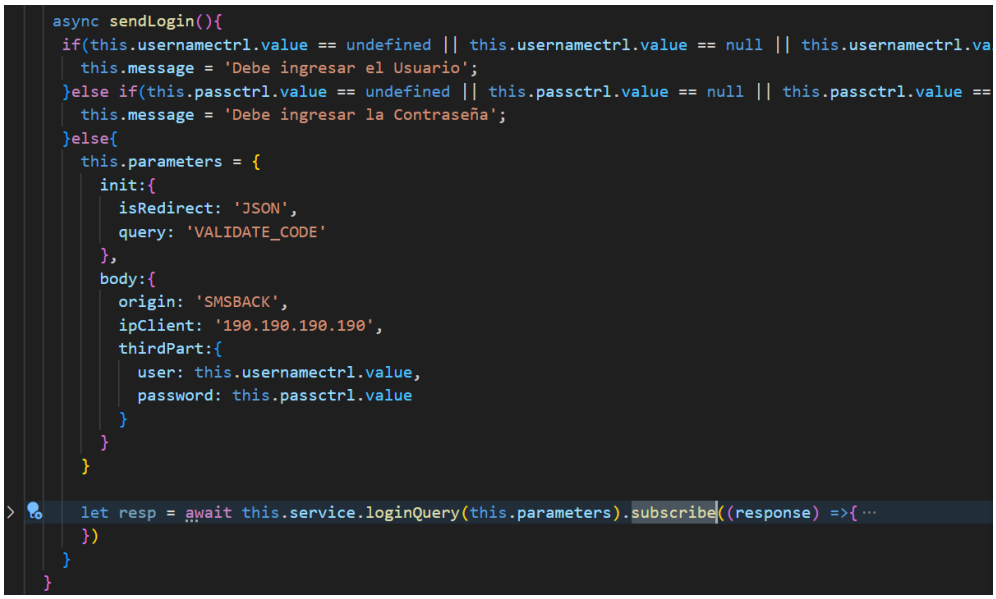


Figura 29. Función asíncrona para consumir servicio de autenticación. Fuente: Elaboración del autor.

- Servicio de validación de usuario en la vista de formulario

De la misma manera que el servicio de autenticación, se crea un servicio para el consumo del servicio de validación de usuario y se implementa una función con las mismas características que el anterior servicio implementado, la única diferencia es que se encapsula el token. Mirar Anexo E.

Figura 30.Consumo del servicio de validación de usuario. Fuente: Elaboración del autor.

Para el caso del controlador se implementa una clase con las mismas características que el controlador de la vista de inicio de sesión.

En este caso se implementa funciones para convertir el archivo Excel en Base64 ya que para poder capturar archivos y enviarlos por la API se debe hacer por medio de la serialización Base64. Para poder realizar la conversión se hace una conversión del archivo Excel en string con el método *ReplaySubject* y posteriormente el convertir la cadena de string es binario con el método *readAsBinaryString* y finalmente pasar de binario a nuevamente a string. Mirar Anexo F y G.

```
//Guarda el archivo Excel
onFileInput(file: File): void{
  this.convertFileTo64(file).subscribe(base64 => {
    this.xlsToBase64 = base64;
  });
}

//Conviernte el archivo XLSX a BASE64
convertFileTo64(file: File): Observable<string>{
  const result = new ReplaySubject<string>(1);
  const reader = new FileReader();

  reader.readAsBinaryString(file);
  reader.onload = (e) => result.next(btoa(e.target!.result!.toString()));

  return result;
}
```

Figura 31. Implementación de conversión archivo Excel a cadena de string. Fuente: Elaboración del autor.

```
//Consumo del servicio Validación Usuario
validateUserRequest(): any{
  this.parametersValidateUser= {
    init:{
      isRedirect: 'JSON',
      query: 'VALIDATION_USER'
    },
    body:{
      origin: 'SMSBACK',
      username: 'JSCARDENAS',
      ip: '10.10.10',
      xlat:'12.3232',
      ylon:'1.22222'
    }
  }

  let resp = this.validateUserService.validationUserQuery(this.parametersValidateUser).subscribe((re
  })

  return resp;
}
```

Figura 32.Función asíncrona para consumir servicio de validación de usuario. Fuente: Elaboración del autor.

- Servicio de ingresar datos en la vista de formulario

El mismo caso y procedimiento que las anteriores implementaciones para consumo de servicios aplica para encapsular y canalizar los datos para ser enviados.

Para el caso del controlador en la captura de los datos del formulario se crea una función asíncrona que encapsula la información y retorna la respuesta del servicio. Mirar Anexo H e I.

```
submitFormQuery(parameters: IformSumbitRequest): Observable<any>{

  return this.globalServiceRequest.postRequest(environment.endpointToken, parameters).pipe(
    map((response: any) => {

      console.log(response.header);
      console.log(response.body);

      return response.body
    }),
    catchError(HandleHttpResponseError)
  );
}
```

Figura 33. Consumo del servicio de ingreso de datos. Fuente: Elaboración del autor.

```
//Consumo del servicio ENTER FILE
async submitForm() {
  this.kannel = this.formSubmit.get('kannel')?.value;
  this.programacion = this.formSubmit.get('programacion')?.value;
  this.plantilla = this.formSubmit.get('plantilla')?.value

  this.parametersForm = {
    init:{
      isRedirect: 'JSON',
      query: 'ENTER_FILE'
    },
    body:{
      origin: 'SMSBACK',
      nombre_archivo: this.nameFile,
      asunto: 'ejemplo',
      content_type: this.fileExtension,
      id_kannel: this.kannel,
      id_plantilla: this.plantilla,
      id_programacion: this.programacion,
      fileB64: this.xlsToBase64
    }
  }

  let resp = await this.enterFileService.submitFormQuery(this.parametersForm).subscribe((response) =
  })
}
```

Figura 34.Función asíncrona para consumir servicio de ingresar datos. Fuente: Elaboración del autor.

Luego de que se desarrolló las vistas, la implementación de los servicios y las funciones se procede a implementar funciones, servicios y métodos con sentencias IF para blindar el código para evitar errores de datos en la base de datos, causar excepciones en el backend e introducción de datos herrones en el momento de encapsular y realizar las peticiones JSON.

Blindaje de errores del proyecto en angular

Esta solución está enfocada para que personal de marketing que no tiene conocimiento en el desarrollo de aplicaciones web, por lo tanto, el usuario puede ingresar valores erróneos o realizar acciones que puedan interrumpir el proceso de la aplicación. Por eso es necesario implementar funciones o métodos que eviten estos incidentes en la aplicación.

Para evitar incidentes en la aplicación se implementa funciones que muestre mensajes en las vistas mostrando la mala operación o el ingreso erróneo de algún dato ingresado.

En el caso que el usuario no ingrese datos en alguno de los campos requeridos se hace uso de la propiedad *mat-error* la cual permite mostrar un mensaje de error debajo de los inputs en el momento en que se genere un desenfoque y el campo no tenga un valor ingresado.

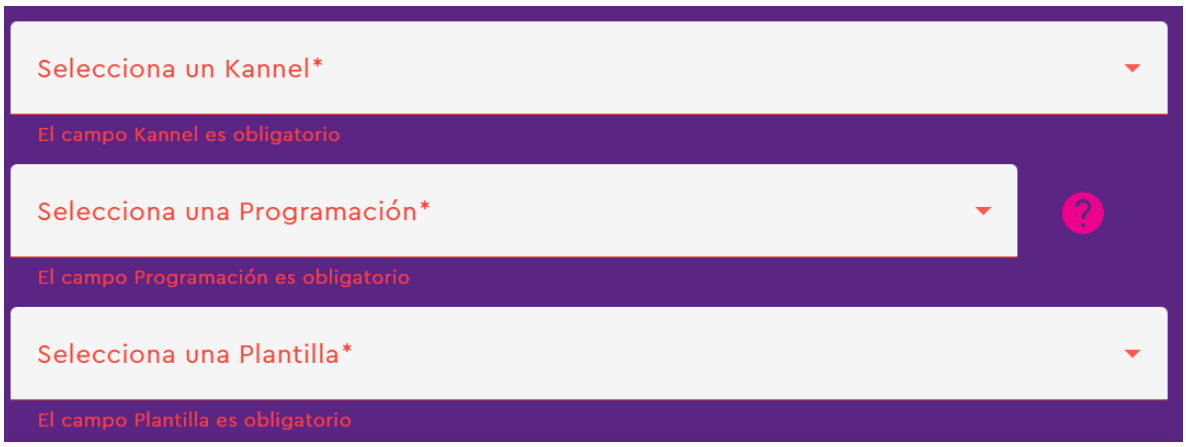


Figura 35. Blindaje de campos requeridos sin ingreso de valores. Fuente: Elaboración del autor.

En el caso del formulario se debe garantizar que en el momento que el usuario accione el botón *guardar y Enviar* todos los campos se contenga un valor, por lo que se hace uso del método de los formularios reactivos de angular que es *valid* este método retorna un True si el campo contiene algún valor o false si está vacío. Aprovechando este método se puede habilitar el botón solo cuando el formulario contenga todos los campos con algún valor. Mirar Anexo J.

```
//Valida que el form este completo
validateForm(): boolean{
  let validate: boolean = false;
  if (this.formSubmit.get('kannel')?.valid &&
      this.formSubmit.get('programacion')?.valid &&
      this.formSubmit.get('plantilla')?.valid &&
      this.formSubmit.get('addFile')?.valid &&
      this.xlsToBase64 != undefined)
  {
    validate = true;
  }

  return validate;
}
```

Figura 36. Función que valida que el formulario esta completo con el método valid. Fuente: Elaboración del autor.

Del lado de las API también se implementaron mensajes del sistema, estos mensajes del sistema permiten realizar soporte o mantenimiento a la aplicación, por ese motivo se debe generar un mensaje emergente en donde muestre el código y mensaje de error en la pantalla.

Para implementar el mensaje emergente se hace uso de un componente propio de angular material y es *MatSnackBar* este componente permite mostrar un mensaje a partir de una vista html que para este proyecto la vista se llama *dialogError* la cual contiene una estructura HTML con sus propios estilos CSS.
El SnackBar permite mostrar un mensaje emergente en cualquier posición de la pantalla y un determinado tiempo. Este mensaje se muestra cuando JSON genera una excepción desde el back, comunicado desde el API. Mirar Anexo K.

```
showDialog(data: ImessageError, typeMsg: string){
  this.__snackBar.openFromComponent(DialogErrorComponent,{
    data:{
      idTransaction: data.idTransaction,
      tecnicalMsg: data.tecnicalMessage
    },
    duration: 7000,
    horizontalPosition:'right',
    verticalPosition:'top',
    panelClass: typeMsg
  });
}
```

Figura 37. Implementación de mensaje de dialogo con SnackBar. Fuente: Elaboración del autor.

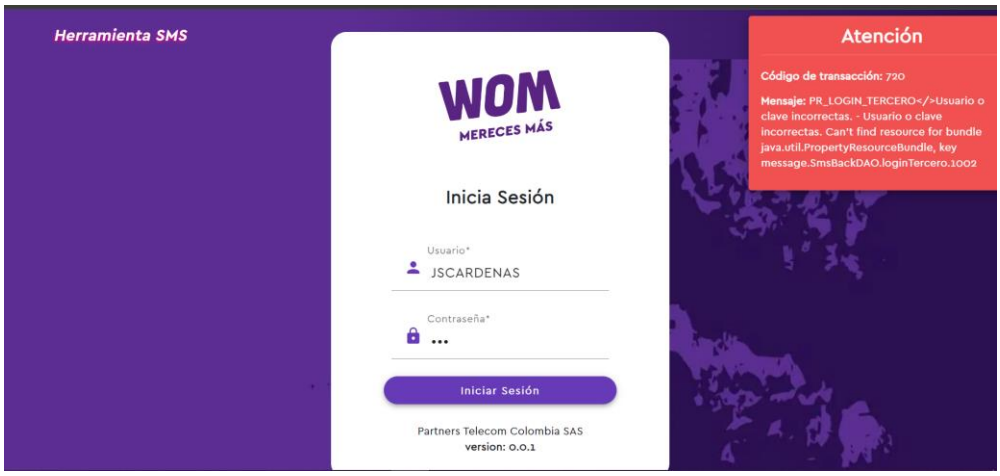


Figura 38. Mensaje de dialogo del sistema generado con SnackBar. Fuente: Elaboración del autor.

Automatización de la solución de actualización de datos para aplicar descuentos

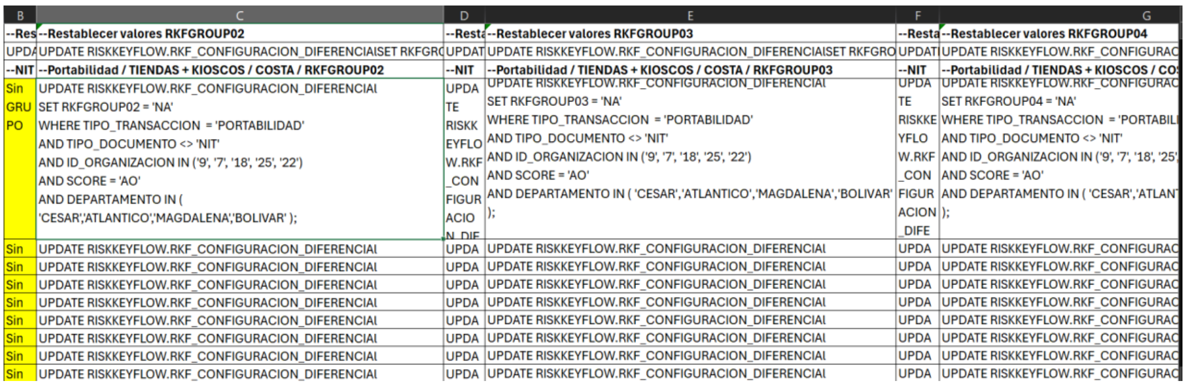
En el caso de la automatización de aplicación de descuentos en los servicios de telefonía, se hace uso de lenguaje de programación Visual Basic Aplication (VBA), ya que la herramienta Excel contiene una función de programación que permite integrar y crear nuevas funciones o macros para un archivo Excel.

Para generar los comandos SQL de actualización con solo el ingreso de datos en un Excel, primero se crea una nueva función Excel que permita concatenar cadenas de string y con sentencias IF para determinar los filtros del query SQL. Mirar Anexo L.

```
Function generarSQL(grupo As String, codeIpp As String, transaccion As String, documento As String, canal As String,
    Dim scriptSql As String

    Dim updateSql, setGrupo, condTrans, condDocum, condCanal, condScore, condDept, condDoc As String

    updateSql = "UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAl" & vbCrLf

    '---------Set grupo con CodeIPP---------'
    setGrupo = "SET " & grupo & " = '" & codeIpp & "'" & vbCrLf

    If codeIpp = "No venta" Or codeIpp = "N/A" Or codeIpp = "0" Then
        setGrupo = "SET " & grupo & " = 'NA'" & vbCrLf
    End If

    If grupo = "GRUPO_MANUAL" And codeIpp = "NA" Then
        setGrupo = "SET " & grupo & " = NULL" & vbCrLf
    End If

    If grupo = "GRUPO_MANUAL" And codeIpp = "N/A" Then
        setGrupo = "SET " & grupo & " = NULL" & vbCrLf
    End If

    If grupo = "GRUPO_MANUAL" And codeIpp = "No venta" Then
        setGrupo = "SET " & grupo & " = NULL" & vbCrLf
    End If

    '---------Condición tipo transacción---------'
    If transaccion = "PortIn POS" Or transaccion = "PortIn PRE" Then
        condTrans = "WHERE TIPO_TRANSACCION  = 'PORTABILIDAD'" & vbCrLf
    End If

    If transaccion = "LN POS" Or transaccion = "PLN PRE" Then
        condTrans = "WHERE TIPO_TRANSACCION IN('VENTA', 'VENTA_POSPAGO')" & vbCrLf
    End If

    If transaccion = "Migración POS" Or transaccion = "Migración PRE" Then
        condTrans = "WHERE TIPO_TRANSACCION  = 'UPGRADE'" & vbCrLf
    End If
```

Figura 39. Función que genera Queries SQL. Fuente: Elaboración del autor.

Una vez que se tiene la función que generar los query en una nueva hoja en cada una de las celdas se hace uso de la función en donde los parámetros de estrada son: grupo, código IPP, tipo de transacción, tipo de documento, tipo de canal, Score y departamento del país. La función retorna una cadena de string que conforman el query teniendo en cuenta los valores ingresados.



B	C	D	E	F	G
--Res	--Restablecer valores RKFGROUP02	--Resta	--Restablecer valores RKFGROUP03	--Resta	--Restablecer valores RKFGROUP04
UPDA	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAISET RKFGRO	UPDAT	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAISET RKFGRO	UPDAT	UPDATE RISKKEYFLOW.RKF_CONFIGURAC
--NIT	--Portabilidad / TIENDAS + KIOSCOS / COSTA / RKFGROUP02	--NIT	--Portabilidad / TIENDAS + KIOSCOS / COSTA / RKFGROUP03	--NIT	--Portabilidad / TIENDAS + KIOSCOS / CO
Sin GRU PO	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAL SET RKFGROUP02 = 'NA' WHERE TIPO_TRANSACCION = 'PORTABILIDAD' AND TIPO_DOCUMENTO <> 'NIT' AND ID_ORGANIZACION IN ('9', '7', '18', '25', '22') AND SCORE = 'AO' AND DEPARTAMENTO IN ('CESAR','ATLANTICO','MAGDALENA','BOLIVAR');	UPDA TE RISKK EYFLO W.RKF _CON FIGUR ACIO N_DIF	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAL SET RKFGROUP03 = 'NA' WHERE TIPO_TRANSACCION = 'PORTABILIDAD' AND TIPO_DOCUMENTO <> 'NIT' AND ID_ORGANIZACION IN ('9', '7', '18', '25', '22') AND SCORE = 'AO' AND DEPARTAMENTO IN ('CESAR','ATLANTICO','MAGDALENA','BOLIVAR');	UPDA TE RISKKE YFLO W.RKF _CON FIGUR ACION _DIFE	UPDATE RISKKEYFLOW.RKF_CONFIGURAC SET RKFGROUP04 = 'NA' WHERE TIPO_TRANSACCION = 'PORTABILI AND TIPO_DOCUMENTO <> 'NIT' AND ID_ORGANIZACION IN ('9', '7', '18', '25' AND SCORE = 'AO' AND DEPARTAMENTO IN ('CESAR','ATLANT);
Sin	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAL	UPDA	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAL	UPDA	UPDATE RISKKEYFLOW.RKF_CONFIGURAC
Sin	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAL	UPDA	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAL	UPDA	UPDATE RISKKEYFLOW.RKF_CONFIGURAC
Sin	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAL	UPDA	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAL	UPDA	UPDATE RISKKEYFLOW.RKF_CONFIGURAC
Sin	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAL	UPDA	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAL	UPDA	UPDATE RISKKEYFLOW.RKF_CONFIGURAC
Sin	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAL	UPDA	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAL	UPDA	UPDATE RISKKEYFLOW.RKF_CONFIGURAC
Sin	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAL	UPDA	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAL	UPDA	UPDATE RISKKEYFLOW.RKF_CONFIGURAC
Sin	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAL	UPDA	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAL	UPDA	UPDATE RISKKEYFLOW.RKF_CONFIGURAC
Sin	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAL	UPDA	UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAL	UPDA	UPDATE RISKKEYFLOW.RKF_CONFIGURAC

Figura 40.Queries generados por la función GenerarSQL. Fuente: Elaboración del autor.

Ya que se tiene los diferentes queries generados, estos deben ser exportados en un archivo sql, para ello hacemos uso de librería propia de Microsoft para poder generar y exportar archivos a partir de la lectura y escritura de un conjunto de celdas.

Para poder hacer uso de la librería se crea una función cuyos parámetros de entrada es la ruta del directorio en donde se va a exportar el archivo y columna de la hoja en donde se generaron los queries. Mirar Anexo M.

```
Sub crearArchivo(ruta As String, columna As Integer)

'Macro que recibe la ruta y la columna para crear el archivo. sql'

Dim fso As FileSystemObject
Dim tsTArchivo As Scripting.TextStream
Dim wsScript As Worksheet
Dim i As Long
Dim strCelda As String
Dim ultimaFila As Integer

Set wsScript = ThisWorkbook.Worksheets("ScriptsSQL")

ultimaFila = Hoja5.Cells(Rows.Count, 1).End(xlUp).Row 'Identifica la ultima fila usada en la Hoja5'

Set fso = New FileSystemObject

Set tsTArchivo = fso.CreateTextFile(ruta, False) 'Crea el archivo'

For i = 1 To ultimaFila
    strCelda = wsScript.Cells(i, columna).Text 'Lee las filas de las columna'
    If strCelda <> "Sin GRUPO" Then
        If strCelda <> "Sin CODE IPP" Then
            tsTArchivo.WriteLine strCelda 'Escribe el archivo'
        End If
    End If
Next i

tsTArchivo.Close 'Cierra el archivo'

End Sub
```

Figura 41. Función que permite exportar archivos. Fuente: Elaboración del autor

Comando	**Función**
FileSystemObject	Permite la creación del archivo.
Scripting.TextStream	Permite que el archivo creado pueda ser abierto, escrito y cerrado.

Worksheet	Identifica cual hoja se va a leer para escribir el archivo.

1. Se identifica cuál será la hoja que se extraerá o leerá os datos de la celda.
2. Se crea una variable de tipo integer que identifica cual es la última celda o fila que es diferente de vacío es decir que no tiene un valor.
3. Se crea el archivo que se va a escribir.
4. En un ciclo FOR se recorre la columna del parámetro de entrada hasta la última celda que no estén vacía.
5. Luego en una varia se lee las celdas no vacías.
6. Como forma de blindar el código, el archivo solo se va a escribir si en el campo de grupo y código no está vacío.
7. El archivo se escribe para luego cerrarse.

Luego de que se implementó la generación de queries se procede a crear una simple tabla en donde el usuario tendrá la opción de seleccionar alguno de los grupos e ingresar la ruta en donde se va a exportar los archivos.

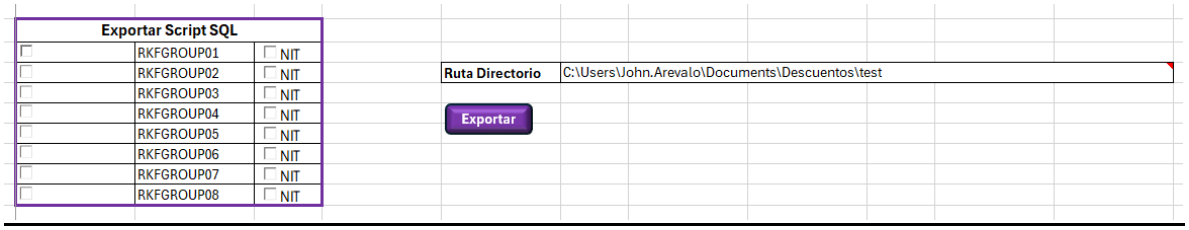


Figura 42. Interfaz de usuario para seleccionar y exportar archivos. Fuente: Elaboración del autor.

Como medida de blindaje para evitar errores de ingreso de datos en la base de datos se desarrolla dos funciones: una que no permite que el usuario ingrese el nombre de uno de los grupos de forma errónea y la repetición de grupos.

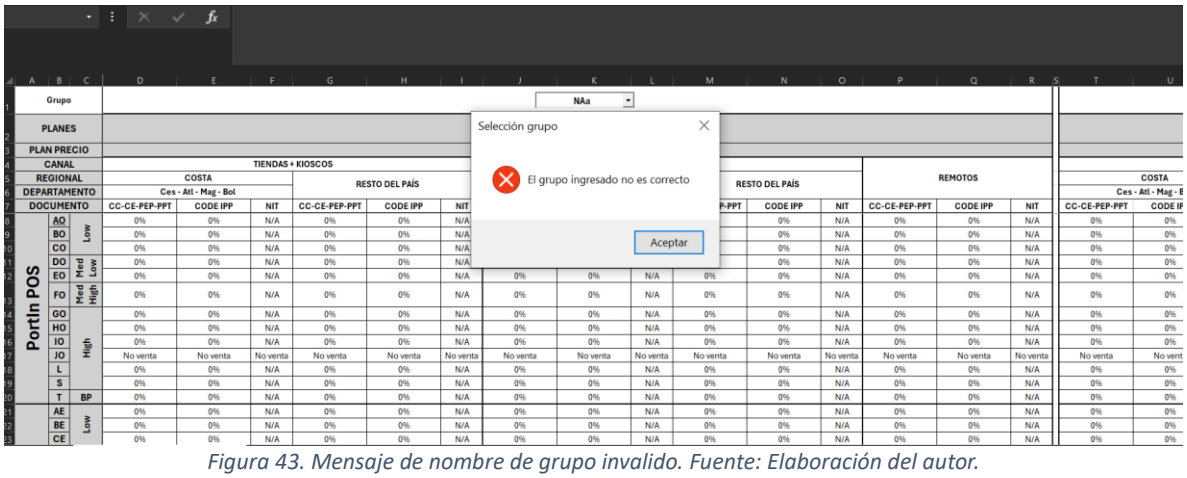


Figura 43. Mensaje de nombre de grupo invalido. Fuente: Elaboración del autor.

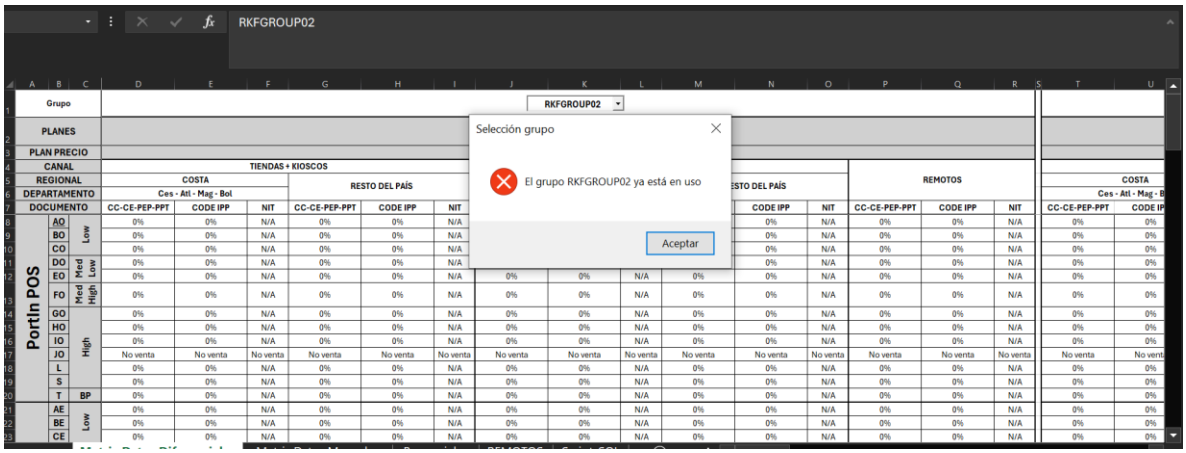


Figura 44. Mensaje de duplicación de grupo. Fuente: Eaboración del autor.

También se desarrolló macros para evitar la duplicidad de exportación de archivos y el ingreso no valido de la dirección de un directorio.

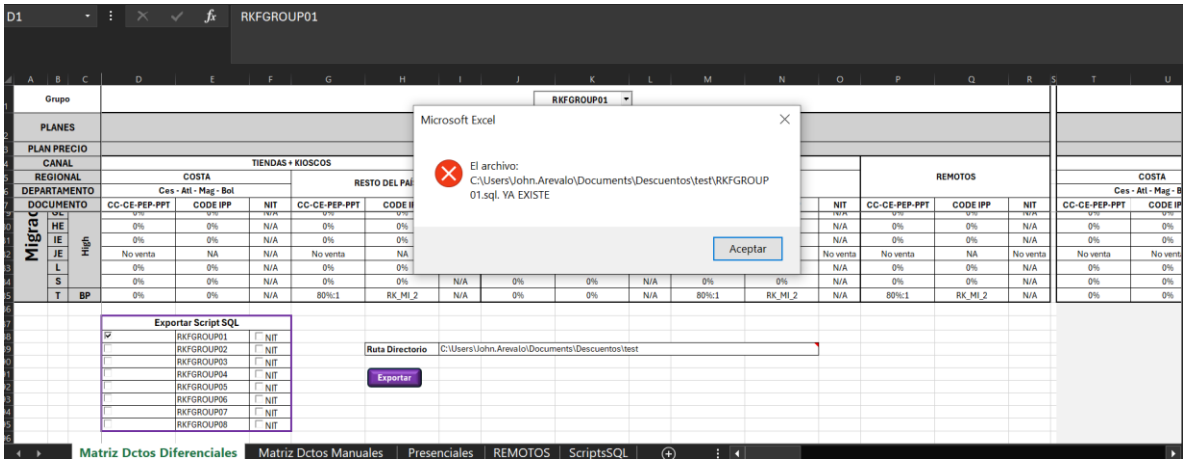


Figura 45. Mensaje de un archivo a exportar ya existe. Fuente: Elaboración del autor.

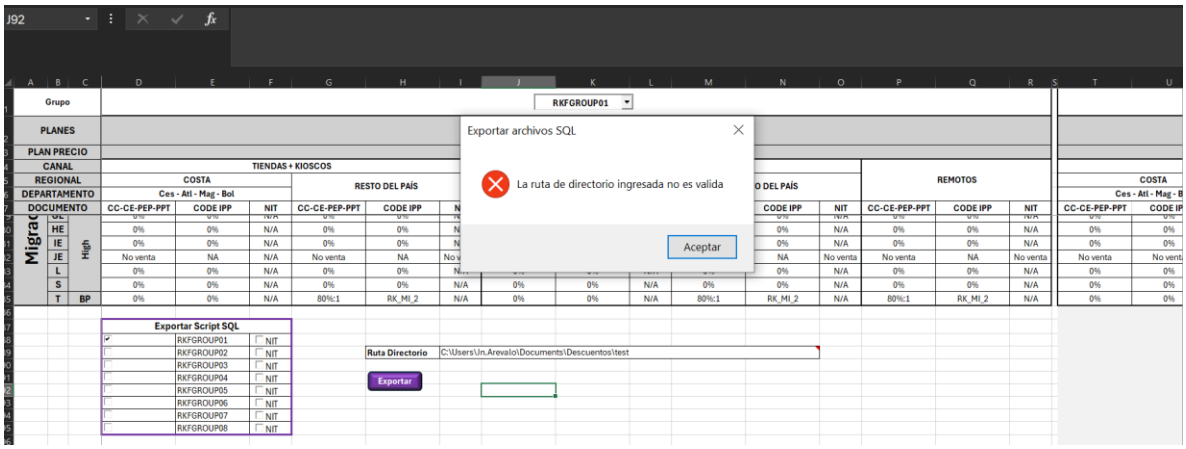


Figura 46. Mensaje de ruta de directorio ingresada no valida. Fuente: Elaboración del autor.

Etapa de pruebas

Para esta fase no se implementa el desarrollo de pruebas unitarias porque no está en el alcance de las funciones del área de aplicaciones, las desarrolla el área de QA. Pero por efectos de calidad y calidad del funcionamiento del desarrollo de las aplicaciones de diseñan pruebas de escritorio, las pruebas de escritorio consisten en generar datos de pruebas o escenarios de pruebas que pongan a prueba el funcionamiento esperado de la aplicación.

Pruebas de funcionamiento de la herramienta SMS Masivos

Para realizar las pruebas de escritorio los desarrolladores de la parte del backend registraron datos de prueba en la tabla donde se va a registrar la información de la programación de la divulgación de mensajes con el fin de que el aplicativo del frontend muestre correctamente la información.

La primera prueba por realizar es el servicio de autenticación, que consiste en ingresar el usuario y contraseña registrado en la base de datos y con los permisos necesarios para generar el token de autenticación.

Para evidenciar que el token es capturado por el aplicativo frontend para posteriormente tener acceso a la vista del formulario, hacemos uso de la consola de inspección del browser en la cual se va a imprimir el token generado desde la API.

Como se puede evidenciar en la figura 44 en la parte derecha se encuentra la consola del navegador que nos permite visualizar los mensajes producidos por el comando *console.log.* Para efectos de la prueba se usa el comando para mostrar el token generado por la API y capturada por el aplicativo del mismo modo en la parte inicial de la consola se puede observar el header usado para encapsular en la cabecera del JSON del servicio de VALIDATION_USER que para poder hacer uso de la API se debe encapsular en la cabecera el token como requisito de seguridad.

Con esta prueba se evidencia que el aplicativo captura la información tanto desde el frontend y el backend para permitir el acceso a la vista del formulario cumpliendo los parámetros de seguridad de la corporación.

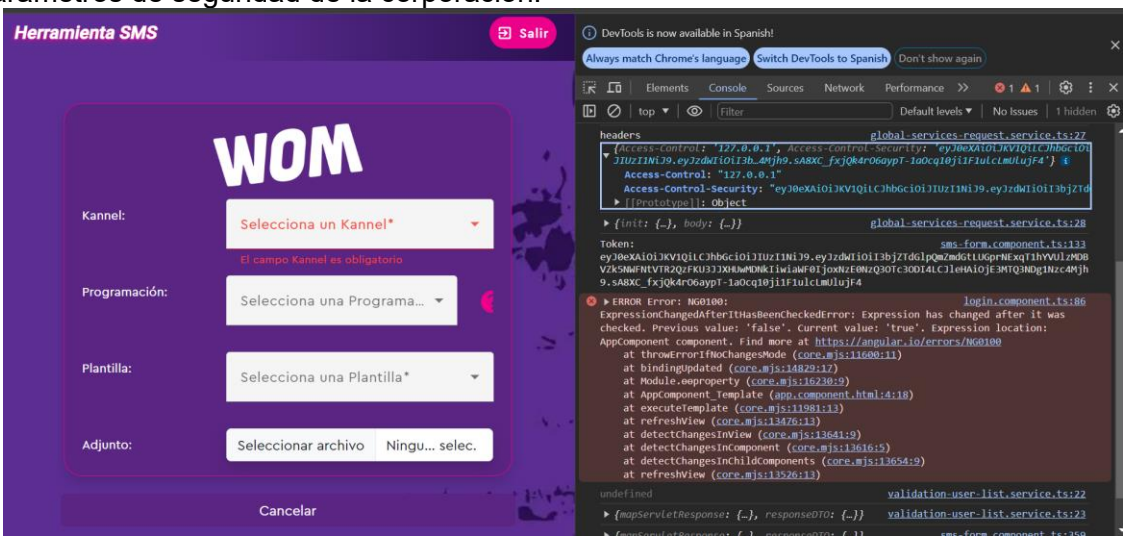


Figura 47. Prueba de captura de Token para crear acceso. Fuente: Elaboración del autor.

La segunda prueba es para la vista del formulario es donde se visualiza las diferentes listas que están relacionadas con el usuario, esta información es suministrada por el servicio VALIDATION_USER que encapsula las listas por medio del JSON y el aplicativo frontend consume la API y muestra la información correspondiente.

Para realizar esta prueba solo es necesario evidencia que los *DownDropList* enliste la información correspondiente a cada uno de los campos como los es el Kannel, Programación y Plantilla.

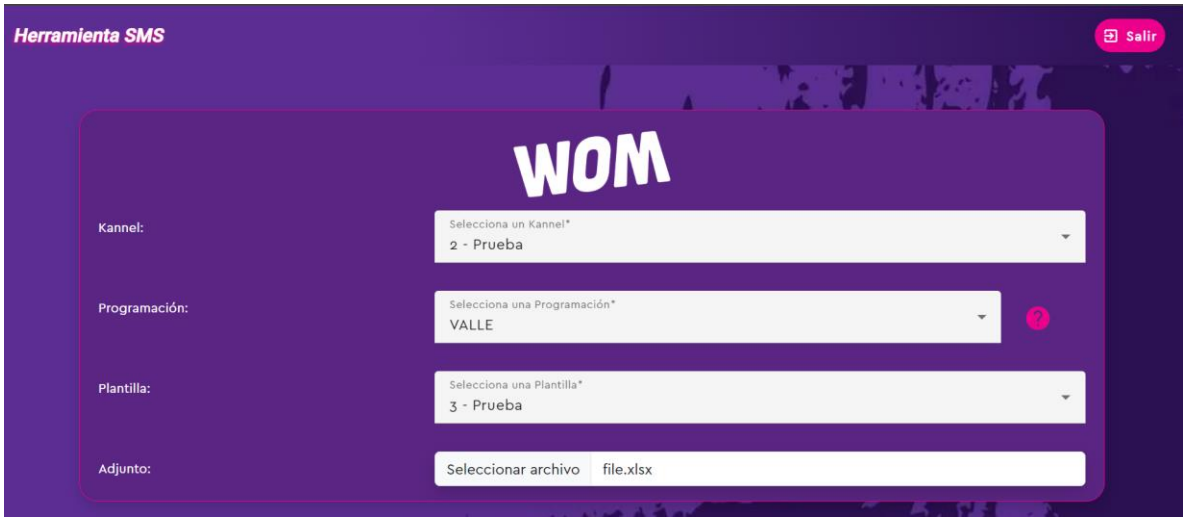


Figura 48. Prueba de consumo de servicio VALIDATION_USER. Fuente: Elaboración del autor.

Como se puede evidenciar en la figura 45 los campos *DownDropList* muestra correctamente la información en cada uno de los campos del formulario y se puede seleccionar para posteriormente ser seleccionado.

La última prueba evidencia que el aplicativo frontend captura la información, la encapsula y la envía por medio del JSON para posteriormente ser procesada y almacenada en la base datos.
Para realizar esta prueba solo es necesario realizar la acción de clic en el botón *Guardar y Enviar*. Este botón tiene la acción de llamar a la función que captura la información del HTML, encapsula la información y consume el servicio implementado para comunicarse con la API del microservicio ENTER_FILE.

	ID_ARCHIVO	NOMBRE	ASUNTO	CONTENTYPE	ESTADO	ID_KANNEL	ID_PLANTILLA	PROGRAM
1	50	file	ejemplo	xlsx	INICIAL	2	3	VALLE
2	46	SMSBACK_13	ejemplo	xlxs	INICIAL	2	3	VALLE
3	45	SMSBACK_12	ejemplo	xlxs	INICIAL	2	3	VALLE
4	44	SMSBACK_11	ejemplo	xlxs	INICIAL	2	3	VALLE
5	43	SMSBACK_10	ejemplo	xlxs	INICIAL	2	3	VALLE
6	42	SMSBACK_9	ejemplo	xlxs	CANCELADO	2	3	VALLE
7	41	SMSBACK_8	ejemplo	xlxs	CANCELADO	2	3	VALLE
8	39	SMSBACK_7	ejemplo	xlxs	INICIAL	2	3	VALLE
9	37	SMSBACK_6	ejemplo	xlxs	INICIAL	2	3	VALLE
10	35	SMSBACK_5	ejemplo	xlxs	INICIAL	2	3	VALLE
11	34	SMSBACK_4	ejemplo	xlxs	INICIAL	2	3	VALLE
12	32	SMSBACK_3	ejemplo	xlxs	INICIAL	2	3	VALLE

Figura 49. Prueba de registro de la programación en la base de datos. Fuente: Elaboración del autor.

Teniendo en cuenta la figura 46 se puede evidenciar en el registro identificado como ID_ARCHIVO 50 es la información enviada desde el aplicativo frontend en donde se evidencia que el ID_KANNEL corresponde al 2 que fue seleccionado desde la vista del formulario, ID_PLANTILLA correspondiente al valor seleccionado en el formulario, PROGRAMACION VALLE que fue seleccionado y el nombre del archivo Excel adjunto desde el aplicativo frontend.

Pruebas de funcionamiento de automatización de herramienta Excel

En el caso de la automatización de la generación de queries SQL de actualización las pruebas se realizan en un ambiente de producción, pero con asistencia de analistas y especialista dando soporte esto porque en dado caso se puede que se presente un incidente en la base de datos.

Como el objetivo de esta optimización es eliminar el paso del área de aplicaciones y reducir el tiempo que lleva generar los queries de forma manual. Se agenda una reunión con todas las áreas involucradas con el fin de explicar el funcionamiento y el correcto ingreso de los datos en la nueva plantilla.

Una vez las áreas involucradas conocen el proceso y resultado de la automatización de la plantilla, se guía al personal en cómo se debe generar los archivos y posteriormente enviar esos archivos generados al área de soporte para ejecutar los cambios. Durante la ejecución

de los quieres generados por la plantilla se evidencia que no hubo errores de sintaxis ni de ingresos erróneos en la base de datos.

Durante la fase de prueba de la automatización de la plantilla Excel y la ejecución de los cambios en la base de datos, se puede evidencia que hay una reducción de tiempo de un día en la elaboración de los queries, pero un aumento en de tiempo en la ejecución de cambios en la base de datos, esto se debe a que al no generar los comandos manuales hay un incremento de queries. Sin embargo, este aumento de tiempo no tiene un impacto leve ya que estos cambios solo se realizan en tiempo nocturnos y tiene un valle de tiempo de 8 horas.

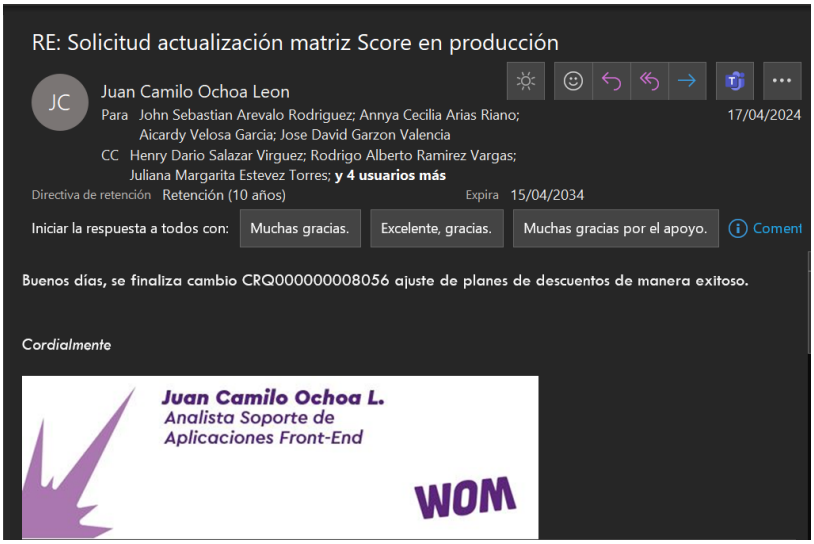


Figura 50. Correo del área de soporte informado que el cambio fue un éxito. Fuente: Elaboración por personal de WOM.

Conclusiones

Luego de recolectar información y analizar los requerimientos necesarios para cada uno de los procesos que involucra información sensible para la corporación, se cumple con la reducción de los tiempos de ejecución de los procesos, desarrollando y automatizando herramientas tecnológicas que permiten cambiar actividades manuelas realizadas por personal a funciones y procesos automáticos.

Al momento de automatizar la herramienta Excel por medio de la programación de macros, se logra evidenciar una reducción en los tiempos de entrega de queries al área de soporte para posteriormente ejecutar los cambios en la base de datos. Así mismo se evidencia que no es necesario tener personal con conocimientos en base de datos para generar los queries, es decir que se cuenta con una comunicación directa desde el área de marketing y el área de soporte por medio de la plantilla Excel.

Durante las pruebas de escritorios realizadas para la implementación de la plataforma web para programar la divulgación de mensaje se puede evidenciar la reducción de tiempo y personal para ejecutar cambios en la base de datos, ya que la aplicación permite que el personal de marketing de forma individual y autónoma puedan programar la divulgación de mensajes sin la necesidad de crear peticiones al área de soporte de la empresa tercera.

La capacidad y conocimiento del área de desarrollo de aplicaciones tiene el objetivo de lograr reducir tiempos de procesos, la capacidad de recolectar requerimiento de funciones manuales realizadas por personal humano e implementar tecnologías que permiten realizar las mismas funciones, pero de una forma autónoma e independiente de la labor humana.

Con base en las estadísticas internas del área de marketing para la aplicación de descuentos en los servicios de telefonía se logra obtener un aumento en la puntuación positiva de los usuarios lo que permite que semanalmente se pueda obtener un incremente de nuevos suscriptores. Este incremento se debe a la reducción de un día para la ejecución de la actualización en la base datos para aplicar los descuentos.

Se logra desarrollar una herramienta tecnológica y automatizar una herramienta que permite reemplazar o cubrir la necesidad de personal humano con conocimientos específicos especialmente en los comandos SQL. También se cumple con la implementación de aplicativos propios de la empresa para suprimir algunos servicios que actualmente se encuentran tercerizados.

Referencias

[1] Mousalli-Kayat, G. (2015). Métodos y Diseños de Investigación Cuantitativa. Mérida.

[2] Haydée Rico H. (2007). El SMS y la mensajería instantánea. Aguascalientes.

[3] Yam Cumi, Ernesto Alonso (2011) Estudio e implementación de un Gateway SMS. Quintana Roo.

[4] Naranjo Granja, Rodrigo Abelardo (2011) PROTOCOLO DE APLICACIONES INALAMBRICAS(WAP).

[5] Muñoz Tapia, José Luis (2017) Tecnología de Contenedores Docker.

[6] Ramos Yánez, Marcos Andrés (2023) Análisis comparativo sobre Frameworks de Javascript Angular Js y Vue Js para el desarrollo para aplicaciones web.

[7] https://angular.io/docs

[8] Suarez Giraldo, Olmer Stiven (2020) Diseño e implementación de un modelo de sistema de información web para la socialización de proyectos de la empresa ARUS.

[9] Gonzales Suarez, Juan Carlos (2020) Arquitectura basada en Microservicios y DevOps para una ingeniería de software continua.

[10] https://developer.mozilla.org/es/docs/Glossary/MVC.

[11] https://learn.microsoft.com/es-es/office/vba/library-reference/concepts/getting-started-with-vba-in-office.

[12] https://www.base64decode.org/.

Anexos

Anexo A. Código de distribución de los campos input

```
<div class="container mt-2">
    <div class="row shadow p-2 mb-2 bg-body mx-1  " id="row-main">
        <div class="col-md-12">
            <img src="../../../assets/images/WOM-blanco(400x200).png"
class="img-fluid mx-auto text-center d-block  logowom" alt="">
            <form [formGroup]="formSubmit" (ngSubmit)="submitForm()"
#campaignForm="ngForm">
                <!-- Select Kannel -->
                <div class="row mb-2">
                    <label for="campaignName" class="col-sm-4 col-form-
label" >Kannel:</label>
                    <mat-form-field class="col-sm-8 selectOption">
                            <mat-label>Selecciona un Kannel</mat-label>
                            <mat-select formControlName="kannel">
                                <mat-option *ngFor="let lk of listKannel"
value="{{lk.idKannel}}">{{lk.codigoCorto}} - {{lk.nombre}}</mat-
option>
                            </mat-select>
                            <mat-error>El campo Kannel es obligatorio</mat-
error>
                        </mat-form-field>
                    </div>

                <!-- Select Programacion -->
                <div class="row mb-2">
                    <label for="description" class="col-sm-4 col-form-label"
id="Formulario">Programación:</label>
                        <mat-form-field class="col-sm-7 selectOption">
                            <mat-label>Selecciona una Programación</mat-
label>
                            <mat-select formControlName="programacion"
(selectionChange)="captDescription($event)">
                                <mat-option *ngFor="let lp of
listProgramacion" value="{{lp.programacion}}">{{lp.programacion}}</mat-
option>
                            </mat-select>
                            <mat-error>El campo Programación es
obligatorio</mat-error>
```

```
                        </mat-form-field>
                        <div class="col-sm-1 helpDiv">
                                <mat-icon
(click)="CaptureDescrProgram()">help</mat-icon>
                        </div>
                </div>

                <!-- Select Plantilla -->
                <div class="row mb-2">
                    <label for="description" class="col-sm-4 col-form-label"
id="Formulario">Plantilla:</label>
                        <mat-form-field class="col-sm-8 selectOption">
                            <mat-label>Selecciona una Plantilla</mat-label>
                            <!-- <mat-select
(selectionChange)="selectPlantilla($event)"> -->
                            <mat-select formControlName="plantilla">
                                <mat-option *ngFor="let lp of listPlantilla"
value="{{lp.idPlantilla}}">{{lp.codigoPlantilla}} -
{{lp.nombrePlantilla}}</mat-option>
                            </mat-select>
                            <mat-error>El campo Plantilla es
obligatorio</mat-error>
                        </mat-form-field>
                </div>

                <!-- Adjuntar Archivo -->
                <div class="row mb-2">
                    <label for="fileInput" class="col-sm-4 col-form-
label  ">Adjunto:</label>
                    <div class="col-sm-8">
                        <div class="input-group shadow bg-body rounded">
                            <input type="file" formControlName="addFile"
class="form-control" id="fileInput" (change)="handleFileInput($event)"
(blur)="validateFile()" accept=".xlsx" />
                        </div>
                    </div>
                </div>

                <!-- label de fecha de programación-->
                <div class="row mb-2" *ngIf="dateSelectedAval">
```

```
                    <label for="fileInput" class="col-sm-4 col-form-
label  ">Fecha de Programación:</label>
                    <div class="col-sm-8">
                        <label class="col-sm-8 col-form-label
">{{dateString}}</label>
                    </div>
                </div>
            </form>
        </div>
    </div>
```

Anexo B. Uso del método POST de la dependencia httpClient

```
@Injectable({
  providedIn: 'root'
})
export class GlobalServicesRequestService {

  constructor(private httpClient: HttpClient) {

  }

  postRequest(petition:any, parameters:any): Observable<any>{
    var headers = {};

    if(petition == environment.endpointLogin){
      headers = {'Access-Control'                : '127.0.0.1'};
    }else if(petition == environment.endpointToken){
      headers = {
        'Access-Control'                : '127.0.0.1',
        'Access-Control-Security'       : localStorage.getItem('token')
      };
    }

    console.log('headers',headers)
    console.log(parameters)

    return this.httpClient.post(environment.url, parameters,
{observe:'response', headers:headers});
  }
}
```

Anexo C. Consumo del servicio de autenticación

```
@Injectable({
  providedIn: 'root'
})
export class LoginUserService {

  constructor(private globalServiceRequest: GlobalServicesRequestService) {
}

  loginQuery(parameters: IloginRequest): Observable<any>{

    return this.globalServiceRequest.postRequest(environment.endpointLogin,
parameters).pipe(
      map((response: any)=> {
        console.log(response.body);
        console.log(response.headers);

          return response.body;
      }),
      catchError(HandleHttpResponseError)
    );
  }

}
```

Anexo D. Función asíncrona para consumir servicio de autenticación

```
async sendLogin(){
    if(this.usernamectrl.value == undefined || this.usernamectrl.value ==
null || this.usernamectrl.value == ''){
      this.message = 'Debe ingresar el Usuario';
    }else if(this.passctrl.value == undefined || this.passctrl.value == null
|| this.passctrl.value == ''){
      this.message = 'Debe ingresar la Contraseña';
    }else{
      this.parameters = {
        init:{
          isRedirect: 'JSON',
          query: 'VALIDATE_CODE'
        },
        body:{
```

```
          origin: 'SMSBACK',
          ipClient: '190.190.190.190',
          thirdPart:{
            user: this.usernamectrl.value,
            password: this.passctrl.value
          }
        }
      }

      let resp = await
this.service.loginQuery(this.parameters).subscribe((response) =>{
        this.responseLogin = response;

        if(this.responseLogin.responseDTO?.code === '0000'){
          localStorage.setItem('token',
this.responseLogin?.mapServletResponse.jwt);
          this.router.navigate(['smsForm']);
        }else if(this.responseLogin?.responseDTO?.code === '1005'){
          this.message = 'El usuario ingresado no existe';
        }else{
          this.msgError = {
            idTransaction:this.responseLogin?.responseDTO?.idTransaction,
            tecnicalMessage:
this.responseLogin?.responseDTO?.tecnicalMessage
          }
          this.dialogMsg.showDialog(this.msgError, 'error-snackbar');
        }
        console.log(this.responseLogin);
      })
    }
  }
```

Anexo E. Consumo del servicio de validación de usuario

```
Injectable({
  providedIn: 'root'
})
export class ValidationUserListService {

  responseValidateUser: any;
```

```
  constructor(private globalServiceRequest: GlobalServicesRequestService) {
}

  validationUserQuery(parameters:IvalidatacionUser):Observable<any>{

    return this.globalServiceRequest.postRequest(environment.endpointToken,
parameters).pipe(
      map((response : any) => {

        console.log(response.header);
        console.log(response.body);

        return response.body
      }),
      catchError(HandleHttpResponseError)
    );
  }

}
```

Anexo F. Implementación de conversión archivo Excel a cadena de string

```
//Guarda el archivo Excel
  onFileInput(file: File): void{
    this.convertFileTo64(file).subscribe(base64 => {
      this.xlsToBase64 = base64;
    });
  }

  //Conviernte el archivo XLSX a BASE64
  convertFileTo64(file: File): Observable<string>{
    const result = new ReplaySubject<string>(1);
    const reader = new FileReader();

    reader.readAsBinaryString(file);
    reader.onload = (e) => result.next(btoa(e.target!.result!.toString()));

    return result;
  }
```

Anexo G. Función asíncrona para consumir servicio de validación de usuario

```
validateUserRequest(): any{
    this.parametersValidateUser= {
      init:{
        isRedirect: 'JSON',
        query: 'VALIDATION_USER'
      },
      body:{
        origin: 'SMSBACK',
        username: 'JSCARDENAS',
        ip: '10.10.10',
        xlat:'12.3232',
        ylon:'1.22222'
      }
    }

    let resp =
this.validateUserService.validationUserQuery(this.parametersValidateUser).su
bscribe((response) =>{
      this.responseValiteUser = response;

      if(response?.responseDTO.code === '0000'){
        this.listPlantilla =
this.responseValiteUser?.mapServletResponse?.list?.plantilla;
        this.listKannel =
this.responseValiteUser?.mapServletResponse?.list?.kannel;
        this.listProgramacion =
this.responseValiteUser?.mapServletResponse?.list?.programacion;
        this.listcalendario =
this.responseValiteUser?.mapServletResponse?.list?.calendario?.calendar;
      }else if(response?.responseDTO.code === '0002'){
        // this.showMesgError(response.responseDTO?.code, 'La sesión ha
expirado, por favor iniciar sesión nuevamente');

        // this.router.navigate(['login']);
        // localStorage.removeItem('token');
        // this.env.authenticated = false;
      }else{
        this.showMesgError(response.responseDTO?.code,
response?.responseDTO?.tecnicalMessage);
```

```
      }

      console.log(this.responseValiteUser);

      return this.responseValiteUser;
    })

    return resp;
  }
```

Anexo H. Consumo del servicio de ingreso de datos

```
@Injectable({
  providedIn: 'root'
})
export class FormSubmitRequestService {

  constructor(private globalServiceRequest: GlobalServicesRequestService) {
}

  submitFormQuery(parameters: IformSumbitRequest): Observable<any>{

    return this.globalServiceRequest.postRequest(environment.endpointToken,
parameters).pipe(
      map((response: any) => {

        console.log(response.header);
        console.log(response.body);

        return response.body
      }),
      catchError(HandleHttpResponseError)
    );
  }
}
```

Anexo I. Función asíncrona para consumir servicio de ingresar datos

```
async submitForm() {
    this.kannel = this.formSubmit.get('kannel')?.value;
    this.programacion = this.formSubmit.get('programacion')?.value;
    this.plantilla = this.formSubmit.get('plantilla')?.value
```

```
    this.parametersForm = {
      init:{
        isRedirect: 'JSON',
        query: 'ENTER_FILE'
      },
      body:{
        origin: 'SMSBACK',
        nombre_archivo: this.nameFile,
        asunto: 'ejemplo',
        content_type: this.fileExtension,
        id_kannel: this.kannel,
        id_plantilla: this.plantilla,
        id_programacion: this.programacion,
        fileB64: this.xlsToBase64
      }
    }

    let resp = await
this.enterFileService.submitFormQuery(this.parametersForm).subscribe((respon
se) => {
      this.responseEnterFile = response;
      console.log(this.responseEnterFile);
      if(this.responseEnterFile.responseDTO?.code === '0000'){

      }else if(this.responseEnterFile.responseDTO?.code === '0002'){
        this.showMesgError(response.responseDTO?.code, 'La sesión ha
expirado, por favor iniciar sesión nuevamente');

        this.router.navigate(['login']);
        localStorage.removeItem('token');
        this.env.authenticated = false;
      }else{
        this.showMesgError(response.responseDTO?.code,
response?.responseDTO?.tecnicalMessage);
      }
      console.log(this.responseEnterFile);
    })
  }
```

Anexo J. Función que valida que el formulario esta completo con el método valid

```
validateForm(): boolean{
    let validate: boolean = false;
    if (this.formSubmit.get('kannel')?.valid &&
        this.formSubmit.get('programacion')?.valid &&
        this.formSubmit.get('plantilla')?.valid &&
        this.formSubmit.get('addFile')?.valid &&
        this.xlsToBase64 != undefined)
    {
      validate = true;
    }

    return validate;
  }
```

Anexo K. Implementación de mensaje de dialogo con SnackBar

```
showDialog(data: ImessageError, typeMsg: string){
    this.__snackBar.openFromComponent(DialogErrorComponent,{
      data:{
        idTransaction: data.idTransaction,
        tecnicalMsg: data.tecnicalMessage
      },
      duration: 7000,
      horizontalPosition:'right',
      verticalPosition:'top',
      panelClass: typeMsg
    });
  }
```

Anexo L. Función que genera Queries SQL

```
Function generarSQL(grupo As String, codeIpp As String, transaccion As String,
documento As String, canal As String, score As String, dept As String) As String
    Dim scriptSql As String

    Dim updateSql, setGrupo, condTrans, condDocum, condCanal, condScore,
condDept, condDoc As String

    updateSql = "UPDATE RISKKEYFLOW.RKF_CONFIGURACION_DIFERENCIAl" & vbCrLf

    '---------Set grupo con CodeIPP---------'
    setGrupo = "SET " & grupo & " = '" & codeIpp & "'" & vbCrLf
```

```
    If codeIpp = "No venta" Or codeIpp = "N/A" Or codeIpp = "0" Then
        setGrupo = "SET " & grupo & " = 'NA'" & vbCrLf
    End If

    If grupo = "GRUPO_MANUAL" And codeIpp = "NA" Then
        setGrupo = "SET " & grupo & " = NULL" & vbCrLf
    End If

    If grupo = "GRUPO_MANUAL" And codeIpp = "N/A" Then
        setGrupo = "SET " & grupo & " = NULL" & vbCrLf
    End If

    If grupo = "GRUPO_MANUAL" And codeIpp = "No venta" Then
        setGrupo = "SET " & grupo & " = NULL" & vbCrLf
    End If

    '---------Condición tipo transacción---------'
    If transaccion = "PortIn POS" Or transaccion = "PortIn PRE" Then
        condTrans = "WHERE TIPO_TRANSACCION  = 'PORTABILIDAD'" & vbCrLf
    End If

    If transaccion = "LN POS" Or transaccion = "PLN PRE" Then
        condTrans = "WHERE TIPO_TRANSACCION IN('VENTA', 'VENTA_POSPAGO')" &
vbCrLf
    End If

    If transaccion = "Migración POS" Or transaccion = "Migración PRE" Then
        condTrans = "WHERE TIPO_TRANSACCION  = 'UPGRADE'" & vbCrLf
    End If

    '---------Condición tipo documento---------'
    If documento = "NIT" Then
        condDoc = "AND TIPO_DOCUMENTO = 'NIT'" & vbCrLf
    End If

    If documento = "CC-CE-PEP-PPT" Then
        condDoc = "AND TIPO_DOCUMENTO <> 'NIT'" & vbCrLf
    End If

    '---------Condición canal---------'
    If canal = "TIENDAS + KIOSCOS" Then
        condCanal = "AND ID_ORGANIZACION IN ('9', '7', '18', '25', '22')" &
vbCrLf
    End If

    If canal = "RETAIL + DEALERS" Then
        condCanal = "AND ID_ORGANIZACION IN ('4', '24', '23', '1')" & vbCrLf
    End If
```

```
    If canal = "REMOTOS" Then
        condCanal = "AND ID_ORGANIZACION IN ('26')" & vbCrLf
    End If

    If canal = "PRESENCIALES" Then
        condCanal = "AND ID_ORGANIZACION IN ('9', '7', '18', '25', '22', '4',
'24', '23', '1')" & vbCrLf
    End If

    '---------Condición score---------'
    condScore = "AND SCORE = '" & score & "'" & vbCrLf

    '---------Condición departamento---------'
    If dept = "COSTA" Then
        condDept = "AND DEPARTAMENTO IN (
'CESAR','ATLANTICO','MAGDALENA','BOLIVAR' );" & vbCrLf
    End If

    If dept = "RESTO DEL PAÍS" Then
        condDept = "AND DEPARTAMENTO NOT IN (
'CESAR','ATLANTICO','MAGDALENA','BOLIVAR' );" & vbCrLf
    End If

    If dept = "REMOTOS" Or dept = "GRUPO_MANUAL" Then
        condDept = ";"
    End If

    scriptSql = updateSql & setGrupo & condTrans & condDoc & condCanal &
condScore & condDept

    generarSQL = scriptSql

    '---------Blindaje---------'
    If grupo = "" Or grupo = "NA" Then
        generarSQL = "Sin GRUPO"
    End If

    If codeIpp = "" Then
        generarSQL = "Sin CODE IPP"
    End If

End Function
```

Anexo M. Función que permite exportar archivos

```
Sub crearArchivo(ruta As String, columna As Integer)

'Macro que recibe la ruta y la columna para crear el archivo. sql'

Dim fso As FileSystemObject
```

```
Dim tsTArchivo As Scripting.TextStream
Dim wsScript As Worksheet
Dim i As Long
Dim strCelda As String
Dim ultimaFila As Integer

Set wsScript = ThisWorkbook.Worksheets("ScriptsSQL")

ultimaFila = Hoja5.Cells(Rows.Count, 1).End(xlUp).Row 'Identifica la ultima fila
usada en la Hoja5'

Set fso = New FileSystemObject

Set tsTArchivo = fso.CreateTextFile(ruta, False) 'Crea el archivo'

For i = 1 To ultimaFila
    strCelda = wsScript.Cells(i, columna).Text 'Lee las filas de las columna'
    If strCelda <> "Sin GRUPO" Then
        If strCelda <> "Sin CODE IPP" Then
            tsTArchivo.WriteLine strCelda 'Escribe el archivo'
        End If
    End If
Next i

tsTArchivo.Close 'Cierra el archivo'

End Sub
```