UCAB Universidad Católica ANDRÉS BELLO

Universidad Católica Andrés Bello
Facultad de Ingeniería
Escuela de Ingeniería Informática

SERVICIO WEB REST API PARA LA APLICACIÓN WEB SOCIALGEST CRM DE LA EMPRESA GEST LABS, C.A.

Este Jurado; una vez realizado el examen del presente trabajo ha evaluado su contenido con el resultado: diecinueve puntos (19)

JURADO EXAMINADOR

Firma:	Firma:	Firma:
Nombre: FANNY HERNÁNDEZ	Nombre: RAFAEL A. LARA CAMPOS	Nombre: Gloria Tarrío V.

REALIZADO POR	José Daniel Andara Giannotti
TUTOR EMPRESARIAL	Lic. Leon Lacourt
TUTOR ACADÉMICO	Ing. Gloria Tarrío Villaverde
FECHA	Marzo 2018

TESIS
II 2018
A5

Universidad Católica Andrés Bello
Facultad de Ingeniería
Escuela de Ingeniería Informática

SERVICIO WEB REST API PARA LA APLICACIÓN WEB SOCIALGEST CRM DE LA EMPRESA GEST LABS, C.A.

TRABAJO INSTRUMENTAL DE GRADO
Presentado ante la
UNIVERSIDAD CATÓLICA ANDRÉS BELLO
Como parte de los requisitos para optar al título de
INGENIERO EN INFORMÁTICA

REALIZADO POR	José Daniel Andara Giannotti
TUTOR EMPRESARIAL	Lic. Leon Lacourt
TUTOR ACADÉMICO	Ing. Gloria Tarrío Villaverde
FECHA	Marzo 2018

DEDICATORIA

A mis abuelos Ramón Andara y Alicia de Andara que siempre me inspiraron a crecer como persona, contándome sus anécdotas de lo importante que es el conocimiento para obtener el éxito y me consintieron lo más que pudieron. Los amo abuelos, espero que donde sea que estén, sepan que este trabajo de grado va dedicado a ustedes, gracias por todo.

A mi madre Miriam Andara que con su apoyo y cariño incondicional, estuvo conmigo en los momentos más difíciles, días que tuve que trasnocharme, siempre preocupada por mi desempeño en la universidad, nunca falto la atención de parte de ella, a pesar de que no supiera nada de lo que hago, siempre hizo el mejor esfuerzo para ayudarme, además de comprarme todo lo necesario para poder cumplir mis asignaciones laborales como académicas y principalmente su amor, que me ha impulsado a continuar. Te amo demasiado mamá, muchas gracias por todo.

A mi esposa Mayra Luna, que durante el transcurso de mi vida como estudiante, tuve la honra de poder conocerla, una persona tan especial, el amor de mi vida, que siempre me ha apoyado en todo y nunca ha dejado de preocuparse por mi bienestar, ayudándome en todo lo que puede. Te amo hasta el infinito mi vida, muchas gracias por todo.

A mis tíos Alicia Andara y Ricardo Marruffo que siempre me brindaron su apoyo, cuando más lo necesite, además de siempre pensar en mí en todo momento, preguntando por todas mis necesidades y como me sentía en todo momento, con el fin de ayudarme. Los amo mucho y muchas gracias por todo.

A mis suegros y mi cuñada, Mariana Perez, Raúl Luna y Xiomara Luna que siempre estuvieron pendiente de mí, preocupados de mi bienestar y apoyándome en todo momento durante estos años de crecimiento profesional, pude sentir el cariño, con sus gestos de amor y consejos de vida, me sentí parte de la familia desde el primer momento en lo que los conocí. Los quiero mucho, gracias por todo.

UCAB Universidad Católica ANDRÉS BELLO

AGRADECIMIENTOS

Agradezco a mi madre, mis tíos, mi esposa, mis suegros y mi cuñada quienes me apoyaron en todo el transcurso de la carrera, en los momentos más duros y difíciles como en los más gratos e importantes, siempre guiándome para ser el mejor en lo que me he dedicado en el transcurso de todos estos años como estudiante, para llegar a ser el profesional que tanto he deseado.

A mis amigos y compañeros de trabajo que compartimos grandes experiencias: Miguel Coppola, Ángel Guacache, Ricardo Pérez, Robert Galeano, Ángel Rivillo, Juan Zozaya y Enrique Nichols, que dedicamos nuestro esfuerzo para especializarnos en nuestras áreas de trabajos, con el fin de fortalecer nuestras habilidades y siempre procurar ayudarnos en momentos de dificultad para poder alcanzar las metas establecidas.

A los profesores: Gloria Tarrio, Ana K Fernandes, Lucia Cardoso y Nelson Rodríguez que gracias a su experiencia obtuve los mejores conocimientos y lecciones, mejorando mi crecimiento profesional y personal a lo largo de estos años de estudio académico.

Pero muchas gracias por sobre todo a mi tutora académica Gloria Tarrío, que me apoyó en la culminación de este trabajo instrumental de grado.

A Gest Labs C.A, por brindarme la oportunidad de trabajar con ellos y formar parte de este gran equipo.

Muchas Gracias.

UCAB Universidad Católica ANDRÉS BELLO

ÍNDICE GENERAL

DEDICATORIA 1
AGRADECIMIENTOS 2
ÍNDICE GENERAL 3
ÍNDICE DE FIGURAS 7
ÍNDICE DE TABLAS 10
SINOPSIS 11
CAPÍTULO I – NECESIDADES DE LA EMPRESA 12
Necesidades de la empresa 12
Solución propuesta 13
Objetivo general 14
Objetivos especificos 14
Aporte funcional 15
Aporte tecnológico 15
Alcance 15
Limitaciones 24
CAPÍTULO II - MARCO TEÓRICO 25
API 25
API de servicio web 26
SOURCE-CODE API 26
LEGACY API 26
Componente de software 26
Modelo de componente 27
REST 27
Aplicación web 28
Cliente 29
Servidor 29
Back-End 29
Front-End 29
PHP 30
Guzzle 30
Framework 30
Laravel 31
Redes sociales 31

Twitter 32
Facebook 32
Access Token 33
Webhook 33
Web chat 33
WebSocket 33
Pusher 34
CAPÍTULO III – MARCO METODOLÓGICO 35
Características scrum 35
Proceso scrum 36
Roles 38
Justificación 40
Product Backlog – Planificación del Sprint 42
CAPÍTULO IV - DESARROLLO 44
Sprint 0 44
Sprint 1 51
Sprint 2 61
Sprint 3 65
Sprint 4 67
Sprint 5 74
Sprint 6 78
Sprint 7 (No Contemplado) 80
Sprint 8 (No Contemplado) 81
CAPÍTULO V - RESULTADOS 84
Resultados obtenidos por objetivos 84
Aporte funcional 87
Aporte tecnológico 88
CAPÍTULO VI – CONCLUSIONES Y RECOMENDACIONES 89
Conclusiones 89
Recomendaciones 90
REFERENCIAS BIBLIOGRÁFICAS 92
ANEXOS 95
Apendice A: Autenticación 95
A.1 Login 95

A.2 Token de refrescamiento 97
A.3 Logout 98
Apéndice B: Permisología 99
B.1 Tipología de permisos 99
B.2 Verificar permisos 101
Apéndice C: Equipos 102
C.1 Consultar equipos 102
C.2 Agregar equipo 103
C.3 Editar equipo 104
C.4 Eliminar equipo 105
C.5 Cambiar el rol de un usuario 106
C.6 Eliminar usuario de un equipo 107
C.7 Invitar usuario 108
C.8 Registrar usuario con equipo 110
Apéndice D: Redes sociales 111
D.1 Consultar redes sociales 111
D.2 Agregar redes sociales 113
D.3 Eliminar redes sociales 114
D.4 Agregar chat 115
Apéndice E: Casos 117
E.1 Responder caso 117
E.2 Responder chat 119
Apéndice F: Mensajes 121
F.1 Consultar casos 121
F.2 Consultar mensajes 123
Apéndice G: Carrito de compra 125
G.1 Consultar ordenes 125
G.2 Agregar orden 127
G.3 Eliminar orden 128
G.4 Consultar planes 129
Apéndice H: Pasarelas de pago 130
H.1 Notificar transferencia 130
Apéndice I: Prueba gratuita 131
I.1 Activar prueba gratuita 131

Apéndice J: Plantillas 132
J.1 Consultar plantillas 132
J.2 Agregar plantilla 133
J.3 Editar plantilla 134
J.4 Eliminar plantilla 135
J.5 Agregar mensaje en plantilla 136
J.6 Modificar mensaje en plantilla 137
J.7 Eliminar mensaje en plantilla 138
Apéndice K: Mensaje automático 139
K.1 Consultar mensajes automáticos 139
K.2 Agregar mensajes automáticos 140
K.3 Modificar mensajes automáticos 141
K.4 Eliminar mensajes automáticos 142
K.5 Consultar horario laboral 143
K.6 Modificar horario laboral 144
K.7 Agregar día de excepción 145
K.8 Modificar día de excepción 146
K.9 Eliminar día de excepción 147
Apéndice L: Caso interno 148
L.1 Agregar caso interno 148
L.2 Agregar caso interno de correo 149
Apéndice M: Escalamiento 150
M.1 Escalamiento de caso 150
Apéndice N: Bitácora 151
N.1 Consultar bitácora 151
Apéndice M: Cliente 153
M1. Consultar información de cliente 153
M2. Agregar información de cliente 154
Apéndice P: Perfil 155
P1. Consultar perfil 155
P2. Editar perfil 157

UCAB Universidad Católica ANDRÉS BELLO

ÍNDICE DE FIGURAS

Figura 1. Proceso actual para la venta de los servicios de la aplicación SocialGest CRM 19
Figura 2. Propuesta del nuevo proceso para la venta de los servicios de la aplicación SocialGest CRM. 20
Figura 3. Proceso actual para escalamiento de casos de la aplicación SocialGest CRM 21
Figura 4. Propuesta del nuevo proceso para el sistema escalamiento de casos de la aplicación SocialGest CRM 22
Figura 5. Pasos en el proceso de Scrum 37
Figura 6. Gráfica comparativa entre Laravel, CakePHP, y Codeigniter 50
Figura 7. Arquitectura del sistema Fuente: Elaboración propia 51
Figura 8. Estructura del JSON de respuesta exitosa del API de SocialGest CRM 52
Figura 9 Estructura del JSON de respuesta errónea del API de SocialGest CRM 53
Figura 10 Modelo entidad relación de la base de datos actual 54
Figura 11 Modelo entidad relación de la base de datos actual 55
Figura 12 Proceso actual para la activación de un plan 59
Figura 13 Nuevo proceso para la activación de un plan Fuente: Elaboración propia 60
Figura 14. Modelo ER de autenticación de usuario 62
Figura 15. Solicitud de servicio de Webhook 68
Figura 16 Conversación entrelazada (*Threaded Conversation*) 70
Figura 17. Ejemplo de notificación de pago. 77
Figura 18 Ejemplo de mensaje automático en Twitter 79
Figura 19. API Login método POST 95
Figura 20. Respuesta API Login método POST 96
Figura 21. API Refresh token método POST 97
Figura 22. Respuesta API Refresh token método POST 98
Figura 23. API Logout método POST 98
Figura 24. Respuesta API Logout método POST 99
Figura 25. API Tipos de permisos método GET 99
Figura 26. Respuesta API tipos de permisos método GET 100
Figura 27. API Verificar permisos método GET 101
Figura 28. Respuesta verificar permisos método GET 101
Figura 29. API Consultar equipos método GET 102
Figura 30. API Respuesta consultar equipos método GET 102
Figura 31. API Agregar equipo método POST 103
Figura 32. Respuesta API agregar equipo método POST 103
Figura 33. API Modificar equipos método PUT 104
Figura 34. Respuesta API modificar equipos método PUT 104
Figura 35. API Eliminar equipos método DELETE 105
Figura 36. Respuesta API eliminar equipos método DELETE 105
Figura 37. API Modificar rol de un usuario PUT 106
Figura 38. Respuesta API modificar rol de un usuario método PUT 106
Figura 39. API Eliminar usuario de un equipo método DELETE 107
Figura 40. Respuesta API eliminar usuario de un equipo método DELETE 107
Figura 41. API Invitar usuario método POST 108

Figura 42. Respuesta API invitar usuario de un equipo POST 109
Figura 43. API Registrar usuario con equipo método POST 110
Figura 44. Respuesta API registrar usuario con equipos método POST 111
Figura 45. API Consultar redes sociales método GET 111
Figura 46. Respuesta API consultar redes sociales método GET 112
Figura 47. API Agregar redes sociales método POST 113
Figura 48. Respuesta API agregar redes sociales método POST 114
Figura 49. API Eliminar redes sociales método DELETE 114
Figura 50.Respuesta API eliminar redes sociales método DELETE 115
Figura 51. API Agregar chat método POST 115
Figura 52. Respuesta API agregar chat método POST 116
Figura 53. API Responder caso método POST 117
Figura 54. Respuesta API responder caso método POST 118
Figura 55. API Responder chat método POST 119
Figura 56. Respuesta API responder chat método POST 120
Figura 57. API Consultar casos método GET 121
Figura 58. Respuesta API consultar casos método GET 122
Figura 59. API Consultar mensajes método GET 123
Figura 60. Respuesta API consultar mensajes método GET 124
Figura 61. API Consultar ordenes método GET 125
Figura 62. Respuesta API consultar ordenes método GET 126
Figura 63. API Agregar ordenes método POST 127
Figura 64. Respuesta API agregar ordenes método POST 127
Figura 65. API Eliminar ordenes método DELETE 128
Figura 66. Respuesta API eliminar ordenes método DELETE 128
Figura 67. API Consultar planes método GET 129
Figura 68. Respuesta API consultar planes método POST 129
Figura 69. API Notificar transferencia método POST 130
Figura 70. Respuesta API notificar transferencia método POST 130
Figura 71. API Activar prueba gratuita método POST 131
Figura 72. Respuesta API activar prueba gratuita método POST 131
Figura 73. API Consultar plantillas método GET 132
Figura 74. Respuesta API consultar plantillas método GET 132
Figura 75. API Agregar plantilla método POST 133
Figura 76. API Agregar plantilla método POST 133
Figura 77. API Modificar plantilla método PUT 134
Figura 78. Respuesta API modificar plantilla método PUT 134
Figura 79. API Eliminar plantilla método DELETE 135
Figura 80. Respuesta API eliminar plantilla método DELETE 135
Figura 81. API Agregar mensaje en plantilla método POST 136
Figura 82. Respuesta API agregar mensaje en plantilla método POST 136
Figura 83. API Modificar mensaje en plantilla método PUT 137
Figura 84. Respuesta API modificar mensaje en plantilla método PUT 137
Figura 85. API Eliminar mensaje en plantilla método DELETE 138
Figura 86. Respuesta API eliminar mensaje en plantilla DELETE 138
Figura 87. API Consultar mensaje automático método GET 139

Figura 88. Respuesta API consultar mensaje automático método GET 139
Figura 89. API Agregar mensaje automático método POST 140
Figura 90. Respuesta API agregar mensaje automático método POST 140
Figura 91. API Modificar mensaje automático método PUT 141
Figura 92. Respuesta API modificar mensaje automático método PUT 141
Figura 93. API Eliminar mensaje automático método DELETE 142
Figura 94. Respuesta API eliminar mensaje automático método DELETE 142
Figura 95. API Consultar horario laboral laboral método GET 143
Figura 96. Respuesta API agregar Consultar horario laboral método GET 143
Figura 97. API Modificar horario laboral método PUT 144
Figura 98. Respuesta API modificar horario laboral método PUT 144
Figura 99. API Agregar día de excepción método POST 145
Figura 100. Respuesta API agregar día de excepción método POST 145
Figura 101. API Modificar día de excepción método PUT 146
Figura 102. Respuesta API modificar día de excepción método PUT 146
Figura 103. API Eliminar día de excepción método DELETE 147
Figura 104. Respuesta API eliminar día de excepción método DELETE 147
Figura 105. API Agregar caso interno método POST 148
Figura 106. Respuesta API agregar caso interno método POST 148
Figura 107. API Agregar caso interno de correo método POST 149
Figura 108. Respuesta API agregar caso interno de correo método POST 149
Figura 109. API Escalamiento de caso método POST 150
Figura 110. Respuesta API escalamiento de caso método POST 150
Figura 111. API Consultar bitácora método GET 151
Figura 112. Respuesta API consultar bitácora método GET 152
Figura 113. API Consultar información de cliente método GET 153
Figura 114. Respuesta API consultar información de cliente método GET 153
Figura 115. API Agregar información de cliente método POST 154
Figura 116. Respuesta API agregar información de cliente POST 154
Figura 117. API Consultar perfil método GET 155
Figura 118. Respuesta API consultar bitácora método GET 156
Figura 119. API Editar perfil método PUT 157
Figura 120. Respuesta API editar perfil método PUT 158

ÍNDICE DE TABLAS

Tabla 1. Métodos de petición (CRUD) 27
Tabla 2. Product backlog 42
Tabla 3. Sprint backlog 43
Tabla 4. Requerimientos para la selección del *framework* 45
Tabla 5. Comparación entre los diferentes *frameworks* en base a los requerimientos. 46
Tabla 6. Comparativo en base al requerimiento de arquitectura RESTful 50
Tabla 7. Tabla de permisología 63
Tabla 8. Valores correspondiente al atributo "source" 72
Tabla 9. Status de tickets 73
Tabla 10. Acciones de bitácora 83

SINOPSIS

El presente Trabajo Instrumental de Grado titulado titulado "Servicio Web Rest API para la aplicación Web SocialGest CRM de la Empresa Gest Labs, C.A" se basó en los lineamientos impuestos por la empresa con el fin de satisfacer todas las necesidades requeridas y eliminar la problemática principal que es que no ha podido compartir las funcionalidades de la aplicación con otras empresas clientes.

Actualmente Gest Labs C.A es una compañía que posee actualmente una suite de herramientas digitales con 2 productos reconocidos: SocialGest y SocialGest CRM. SocialGest CRM es una aplicación con interfaz simple que coloca todas tus conversaciones en una misma bandeja de entrada para un mejor control. Además permite obtener detalles de atención e información accionable para los objetivos del negocio. A comienzos del año 2018 la empresa cambió el nombre de la aplicación SocialGest CRM a Tikket para tener un nombre más comercial.

La metodología empleada para el desarrollo de este proyecto fue SCRUM, que es un marco de trabajo, para el desarrollo y el mantenimiento de productos complejos; que emplea varias técnicas y procesos, con el propósito de alcanzar el éxito del proyecto y destaca por su desarrollo iterativo incremental compuesto por bloques de tiempo (time-box) denominados "Sprints" el cual tiene como objetivo definir qué se va a construir, el diseño y el plan que guiará la construcción de los trabajos y el desarrollo del producto.

Finalmente, el entorno utilizado en el desarrollo del proyecto fue PhpStorm, el lenguaje PHP y el *framework* Laravel (5.* última versión) con una base de datos relacional en MYSQL.

CAPÍTULO I – NECESIDADES DE LA EMPRESA

Necesidades de la empresa

Gest Labs C.A. es una empresa que partió como un emprendimiento no hace mucho, la misma cuenta con poco tiempo en el mercado y se dedica actualmente al desarrollo de herramientas (aplicaciones) para la gestión de redes sociales a nivel nacional e internacional, una de ellas es SocialGest CRM, la cual es una solución que brinda estrategia integral de atención al cliente mediante las mismas redes

La aplicación actualmente se encuentra desarrollada en lenguaje PHP y no cuenta con la implementación de un *framework* de desarrollo (a nivel de *Back-End*). A su vez, está diseñada en lenguaje HTML, CSS, JavaScript, AngularJS y Bootstrap las cuales forman parte del *framework* de *Front-End,* sin embargo carece del uso de patrones de arquitectura de software para separar los datos y la lógica de negocio.

Además la empresa no ha podido compartir las funcionalidades de la aplicación con otras empresas clientes para garantizar un servicio completo, tanto en la solicitud de información, como en adaptar a sus sistemas la utilización de módulos y funcionalidades específicas de la aplicación. A su vez, su sistema de ventas actual representa un proceso largo y tedioso, además de no garantizar la venta de los servicios de la aplicación.

Otro punto es que al momento de generar casos (mensajes de clientes a una red social de la empresa que contrato los servicios) en la aplicación, los operadores no poseen una herramienta para poder escalar los casos a departamentos superiores, causando fuertes retrasos en las respuestas, ocasionando que los mismos abandonen.

Por otro lado, la poca escalabilidad y el mantenimiento que presenta la aplicación SocialGest CRM (debido a la falta de buenas prácticas de desarrollo), impiden que nuevas funcionalidades sean agregadas a la misma, así como la realización de múltiples llamadas para la obtención de datos y verificación del manejo de sesiones o autentificación de los usuarios.

Solución propuesta

La propuesta se basa en el desarrollo de un servicio web REST API, que sustituya todos los módulos *Back-end* de la plataforma, utilizando un *framework* de desarrollo que permita solventar el problema actual. El servicio web REST API será la primera versión de la aplicación y tendrá todos los módulos actuales de la empresa, simplificando el trabajo de los desarrolladores de *Front-end*.

Para empezar se realizará una investigación de los frameworks más usados en la actualidad para el lenguaje PHP, con el fin de realizar una comparación entre los mismos, para llegar a solventar con éxito sus problemas, cubriendo sus expectativas de escalabilidad y mantenimiento de la aplicación, optando por el óptimo funcionamiento de la aplicación y utilizando buenas prácticas para el desarrollo.

Se documentará todo el trabajo realizado para tener un manual (API DOC) con todas las llamadas utilizadas en cada módulo para que la empresa pueda compartir más adelante sus servicios web y mantener informado a los próximos desarrolladores de lo que poseen actualmente.

Objetivo general

Desarrollar el servicio web REST API para la aplicación web SocialGest CRM de la empresa Gest Labs, C.A.

Objetivos específicos

1. Desarrollar la interfaz de programación de aplicaciones de *Login*.
2. Desarrollar la interfaz de programación de aplicaciones de Agregar Usuarios.
3. Desarrollar la interfaz de programación de aplicaciones de Casos.
4. Desarrollar la interfaz de programación de aplicaciones de Mensajes.
5. Desarrollar la interfaz de programación de aplicaciones de Caso Interno.
6. Desarrollar la interfaz de programación de aplicaciones de Clientes.
7. Desarrollar la interfaz de programación de aplicaciones de Bitácora.
8. Desarrollar la interfaz de programación de aplicaciones de Perfil.
9. Desarrollar la interfaz de programación de aplicaciones de Plantillas.
10. Desarrollar la interfaz de programación de aplicaciones de Permisología.
11. Desarrollar la interfaz de programación de aplicaciones de Mensajes Automáticos.
12. Desarrollar la interfaz de programación de aplicaciones de Equipos.
13. Desarrollar la interfaz de programación de aplicaciones de Carrito de Compra.
14. Desarrollar la interfaz de programación de aplicaciones de Prueba Gratuita.

Aporte funcional

15. Rediseñar el proceso de venta de los servicios de la aplicación.
16. Sistema de escalamiento de casos.

Aporte tecnológico

17. Desarrollar un servicio web REST API para la venta de los servicios de la aplicación.
18. Evaluar los distintos *framework* de desarrollo (a nivel de Back-End) disponibles para la creación del servicio web REST API de la aplicación y a su vez reestructurar base de datos existentes para que pueda ser utilizada por el *framework* a implementar.

Alcance

1. Desarrollar la interfaz de programación de aplicaciones de *Login*.

   Este módulo permitirá el acceso de un usuario a la aplicación, almacenando un *token* de acceso (*refresh token*) necesario para utilizar las funcionalidades de la misma, garantizando seguridad en los módulos y la opción de cerrar sesión para finalizar su uso. Todo exclusivamente orientado al *framework* a utilizar.

2. Desarrollar la interfaz de programación de aplicaciones de Agregar Usuarios.

   Construcción de las funcionalidades necesarias para los complemento de seguridad que permitirán autentificar a los usuarios de Facebook y Twitter de la cuenta principal que se conectarán a la aplicación mediante un *token* que proporcionará acceso perdurable y seguro a sus redes sociales. Con la opción de añadir un usuario exclusivo para chat, que proporcionará un URL predefinido generando otro medio de contacto al cliente. Todo exclusivamente orientado al *framework* a utilizar.

3. Desarrollar la interfaz de programación de aplicaciones de Casos.

   Construcción de las funcionalidades necesarias para mostrar la información de los casos provenientes de la red social (Twitter y Facebook), que el usuario (cliente) haya escogido como preferencia para contactar al sistema de gestión de relaciones con los clientes. Además de la elaboración de los complementos necesarios para la recepción y respuesta de las redes sociales y el chat. Adicionalmente para las conversaciones de tipo chat se podrán subir y recibir imágenes o documentos.

   Cabe destacar que en la recepción por parte de Facebook se utilizaran *Webhooks* para obtención de la información en tiempo real. Todo exclusivamente orientado al *framework* a utilizar.

4. Desarrollar la interfaz de programación de aplicaciones de Mensajes.

   Este módulo tendrá como propósito permitir a la empresa visualizar la conversación que realiza la cuenta con el cliente, a través de la red social o mediante el chat dependiendo de donde haya sido contactado (usuario de la empresa o URL del chat) y de qué forma fue contactado (mensaje directo, comentario o chat). Todo exclusivamente orientado al *framework* a utilizar.

5. Desarrollar la interfaz de programación de aplicaciones de Caso Interno.

   Construcción de las funcionalidades necesarias para enviar por correo el histórico de una conversación que hayan sido a través el chat o utilizando el usuario de red social y se dispondrá de otra funcionalidad para iniciar conversaciones mediante la red social (Twitter y Facebook). Todo exclusivamente orientado al *framework* a utilizar.

6. Desarrollar la interfaz de programación de aplicaciones de Clientes.

   Construcción de las funcionalidades para almacenar y mostrar toda la información referente al cliente que contacto a la empresa. Adicionalmente para los casos del chat se obtendrá el IP y país de procedencia. En caso de proxy se utilizará *X-Forwarded-For (XFF)* ya que el encabezado HTTP es un método común para identificar la dirección IP de origen de un cliente ya sea a través de un proxy HTTP o equilibrador de cargas. Todo exclusivamente orientado al *framework* a utilizar.

7. Desarrollar la interfaz de programación de aplicaciones de Bitácora.

   Construcción de las funcionalidades que llevarán a cabo registro de las acciones realizada tanto por la cuenta como por los sub-usuarios (cuentas invitadas a un equipo), dichas acciones serán almacenadas en la base de datos. Todo exclusivamente orientado al *framework* a utilizar.

8. Desarrollar la interfaz de programación de aplicaciones de Perfil.

   Construcción de las funcionalidades necesarias para mostrar la información del perfil de la cuenta o cuenta principal y conjuntamente la cantidad de casos atendidos, en bandeja, cerrados y transferidos, bitácora y permisos. Todo exclusivamente orientado al *framework* a utilizar.

9. Desarrollar la interfaz de programación de aplicaciones de Plantillas.

   Construcción de las funcionalidades necesarias para almacenar y mostrar la información correspondiente a las plantillas del usuario, con el fin de que puedan ser utilizadas a través de un alias para acelerar las conversaciones entre la cuenta y el cliente (*shortcuts*). Todo exclusivamente orientado al *framework* a utilizar.

10. Desarrollar la interfaz de programación de aplicaciones de Permisología.

    Permitirá a la cuenta administrar las secciones visibles para los sub-usuarios, dado que no todos los sub-usuarios tendrán acceso a todos los módulos de la aplicación, asegurando la protección a áreas con información valiosa para la misma. Todo exclusivamente orientado al *framework* a utilizar.

11. Desarrollar la interfaz de programación de aplicaciones de mensajes automáticos.

    Con el propósito de permitir a la aplicación a través de una bandeja de mensajes precargados por el usuario, responderles a los clientes automáticamente, para cuando los sub-usuarios se encuentren fuera del horario laboral. Todo exclusivamente orientado al *framework* a utilizar.

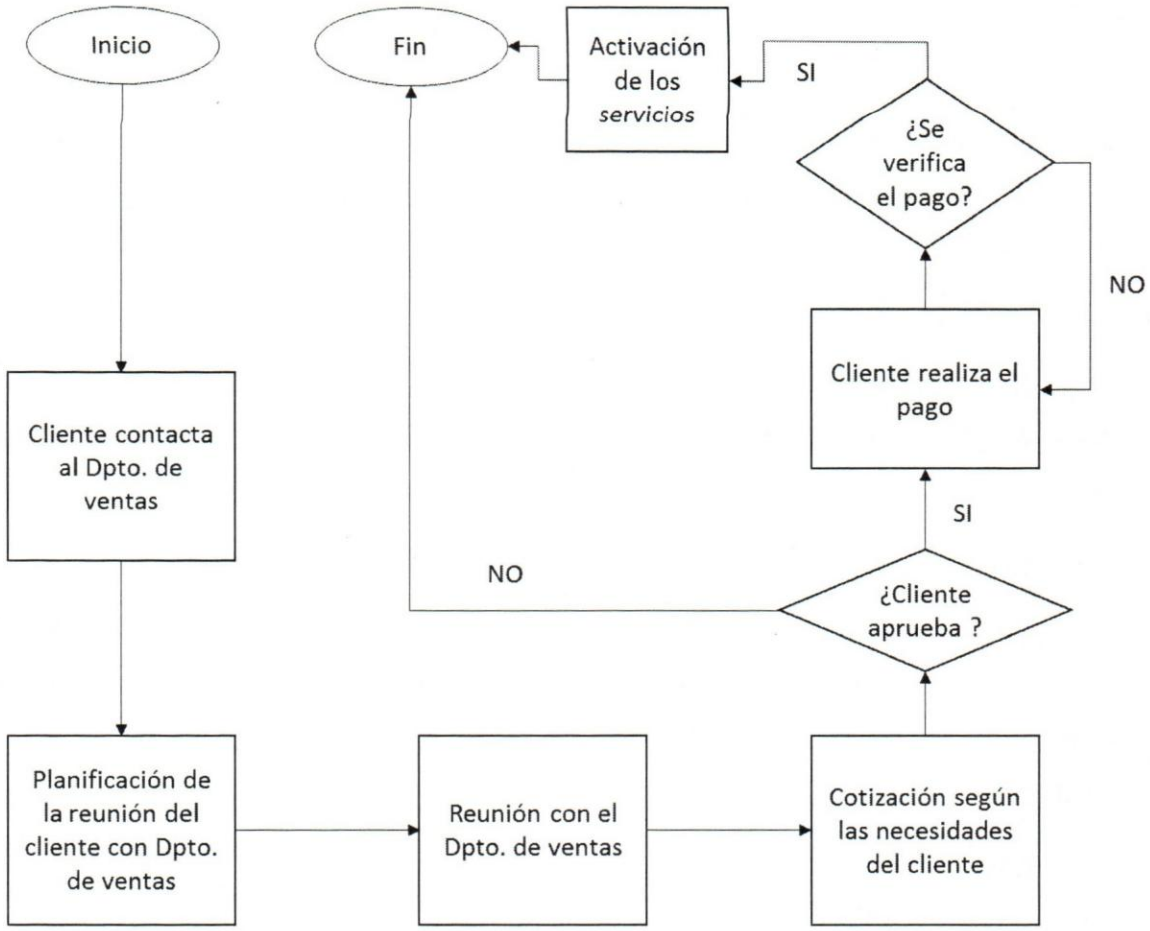


Figura 1. Proceso actual para la venta de los servicios de la aplicación SocialGest CRM

Fuente: Elaboración propia

La propuesta planteada para el nuevo proceso para la venta de los servicios de la aplicación SocialGest CRM es: el cliente ingresa al carrito de compras por la página web, selecciona el plan de su preferencia, se carga la información fiscal, se genera la orden, se realiza el pago, se comprueba el pago y se activa su plan. Cabe destacar que la aplicación necesita de la información fiscal del cliente para poder seguir con el proceso de compra.

En la Figura 2, se puede observar de manera general, una propuesta de cómo podría ser el nuevo proceso de venta de los servicios:

12. *Desarrollar la interfaz de programación* de aplicaciones de Equipos.

    Se encargará de gestionar los usuarios de la cuenta principal, con la finalidad de definir un equipo encargado de gestionar los usuarios de redes sociales y asignarle un rol con permisología a cada uno de ellos a sub-usuarios que pertenecerán. Todo exclusivamente orientado al *framework* a utilizar.

13. Desarrollar la interfaz de programación de aplicaciones de Carrito de Compra.

    Gestionará todo la parte de órdenes que el cliente va a tener que generar para el debido pago de la aplicación web, además de su comprobación a través de Instapago, MercadoPago o PayPal. Todo exclusivamente orientado al *framework* a utilizar.

14. Desarrollar la interfaz de programación de aplicaciones de Prueba Gratuita.

    Construcción de las funcionalidades necesarias para permitir a los nuevos clientes tener la oportunidad de usar la aplicación en un periodo de tiempo, con el fin de que puedan realizar pruebas de las funcionalidades y tener una mejor perspectiva de la misma.

15. Rediseñar el proceso de venta de los servicios de la aplicación.

    La falta de una interfaz de programación de aplicaciones orientada en funcionalidades que faciliten la venta de los servicios de la aplicación, es la mayor desventaja de la empresa, ya que por esto no puede expandirse más rápido en el mercado actual de aplicaciones que ofrecen esos servicios, quedándose atrás ante la competencia.

    Actualmente el proceso que abarca la venta de los servicios de la aplicación SocialGest CRM es: el cliente contacta al Dpto. de ventas, planifica la reunión con el mismo, luego se reúnen y se cotizan las necesidades (servicios) que desea el cliente, el cual tiene la potestad de aprobar o rechazar, finalmente, al ser aprobado realiza el pago y seguidamente se activan los servicios.

    Para visualizar mejor el proceso que se lleva a cabo para el acceso a los servicios, en la Figura 1 se detalla paso a paso el mismo:

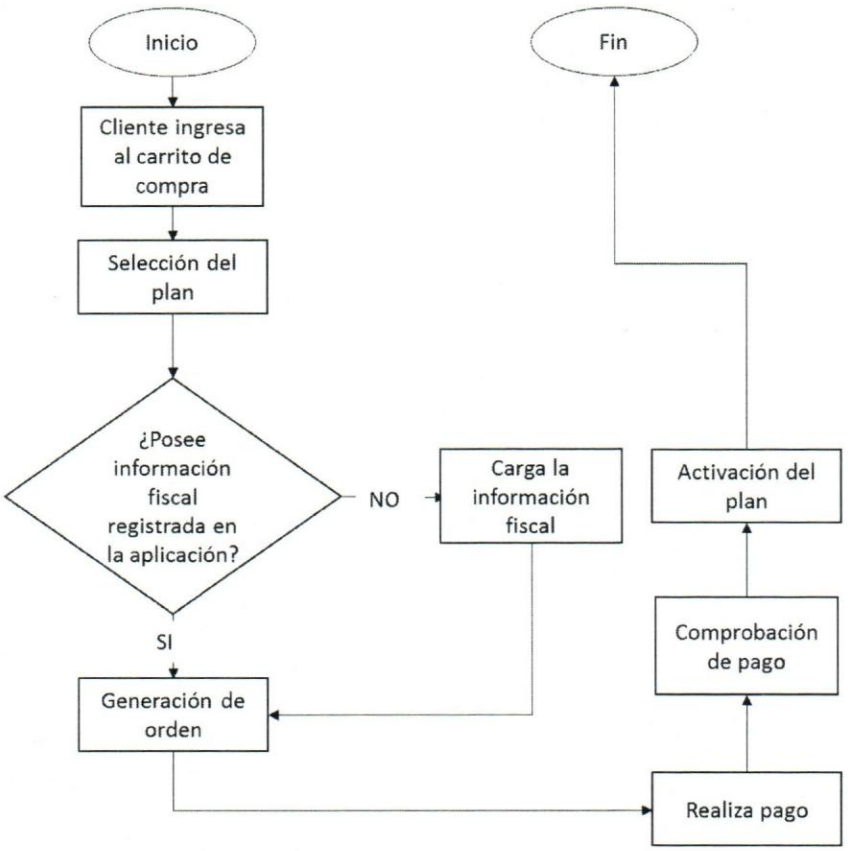


Figura 2. Propuesta del nuevo proceso para la venta de los servicios de la aplicación SocialGest CRM.

Fuente: Elaboración propia

16. Sistema de escalamiento de casos.

Actualmente la aplicación no posee un sistema de escalamiento de casos para recibir respuesta de departamentos superiores, generando fuertes retrasos en los operadores al momento de responder a clientes con inquietudes que exceden a los temas que fueron entrenados.

El proceso actual para "escalar" los casos es: un operador genera una solicitud con el caso dirigido al Dpto. Superior, se aprueba la solicitud, operador vuelve para contactar con el Dpto. Superior, un encargado procesa la solicitud, soluciona el problema y por último almacena el reporte.

Con el fin de visualizar mejor el proceso en la Figura 3 se detallado este proceso:

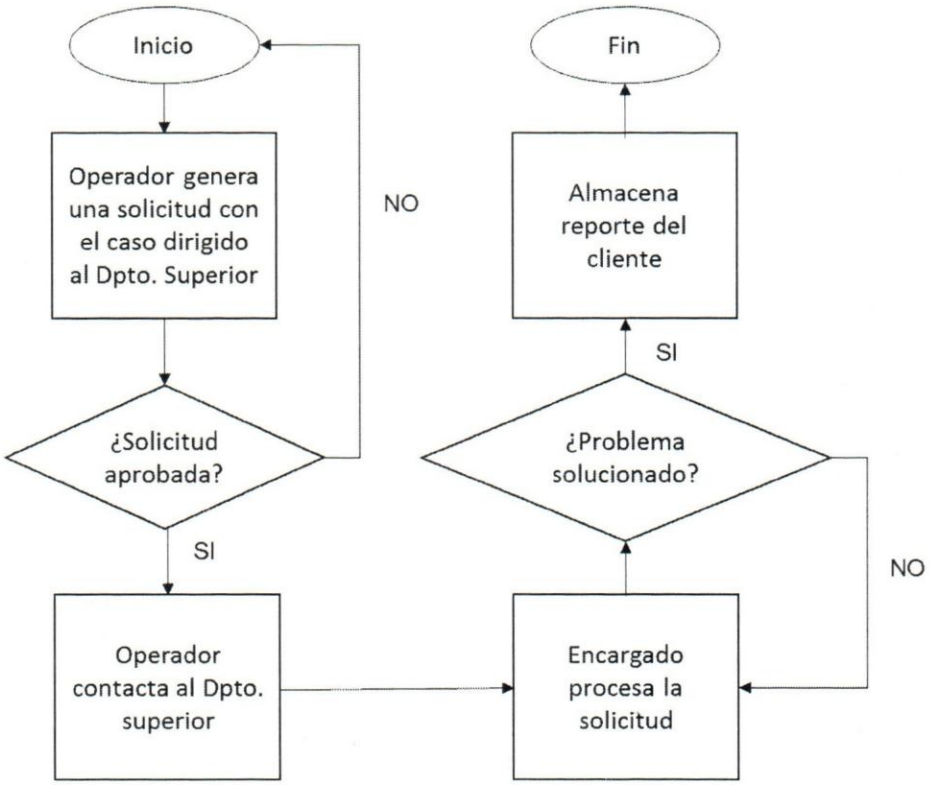


Figura 3. Proceso actual para escalamiento de casos de la aplicación SocialGest CRM

Fuente: Elaboración propia

El proceso actual para "escalar" los casos es: un operador genera una solicitud con el caso dirigido al Dpto. Superior, se aprueba la solicitud, operador vuelve para contactar con el Dpto. Superior, un encargado procesa la solicitud, soluciona el problema y por último almacena el reporte.

Con la inclusión del sistema de escalamiento de casos se espera acabar con el proceso de solicitudes para la resolución de inconvenientes, precisamente aprovechando los servicios API para la generación de tickets provenientes de casos (con toda la información del mismo), para contactar por correo a un encargado del Dpto. Superior con el fin de agilizar este proceso, que intervendrá por el mismo medio donde fue contactado o con su usuario en SocialGest CRM.

Para el sistema de escalamiento el nuevo proceso sería: un operador envía al Dpto. Superior un ticket con toda la información relevante al caso, el encargado procesa la solicitud, se soluciona el problema y se almacena el reporte.

Se puede visualizar en la Figura 4 la propuesta de cómo cambiaría al ser utilizando el sistema de escalamiento:

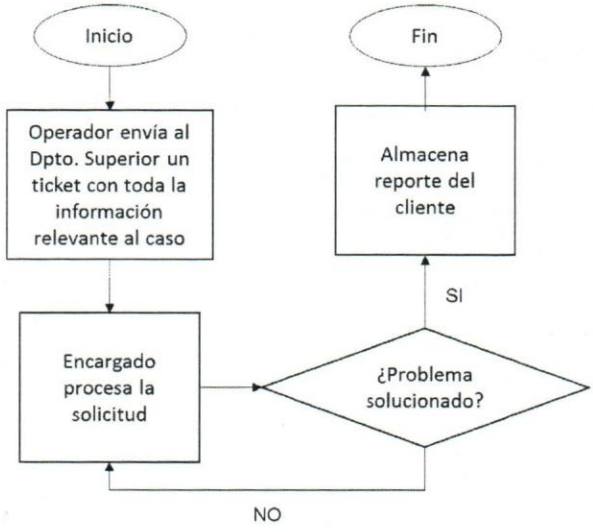


Figura 4. Propuesta del nuevo proceso para el sistema escalamiento de casos de la aplicación SocialGest CRM

Fuente: Elaboración propia

APORTE TECNOLÓGICO

17. Con el desarrollo de una interfaz de programación de aplicaciones orientada a funcionalidades que faciliten la venta de los servicios de la aplicación, se espera acabar con el problema principal que presenta la empresa, ya que sus clientes actuales podrían adquirir sus servicios a través de la página web, donde seleccionará el plan y procederá a realizar el pago, luego de ser verificado el mismo, se le activarán los servicios correspondientes a dicho plan.

18. Evaluar los distintos *framework* de desarrollo (a nivel de Back-End) disponibles para la creación del servicio web REST API de la aplicación y a su vez reestructurar base de datos existentes para que pueda ser utilizada por el *framework* a implementar.

    Evaluar distintos *framework* de *Back-End*, para estudiar sus beneficios y la facilidad que brindan para lograr la escalabilidad de futuras actualizaciones y el mantenimiento de a largo plazo al cumplir con los estándares de desarrollo, también se evaluará la inclusión de herramientas (librerías) para facilitar el desarrollo y orientar la carga de trabajo en la lógica del módulo.

Limitaciones

- El servicio web REST API debe integrarse al ambiente de desarrollo PHP (*frameworks Back-end*) y al manejador de base de datos MySQL.

CAPÍTULO II - MARCO TEÓRICO

Este capítulo está constituido por todos los conceptos teóricos relevantes para el desarrollo de la solución.

API

API es el acrónimo utilizado para *Application Programming Interface*, se utiliza como interfaz para que los componentes de software puedan comunicarse entre sí. Puede incluir especificaciones para rutinas, estructuras de datos, clases de objetos y variables. Un API se caracteriza por ser:

- Dependiente del lenguaje, lo que significa que solo está disponible al usar la sintaxis y los elementos de un lenguaje en particular, lo que hace que la API sea más cómoda de usar.
- Independiente del lenguaje, escrito para que puede ser usado desde varios lenguajes de programación. Esta es una característica deseable para una API orientada a servicios que no está vinculada a un proceso o sistema específico, dando la posibilidad de poder proporcionar llamadas a procedimientos remotos o servicios web.

Las API varían dependiendo del mecanismo de transporte que esté basado (hipertexto, código fuente o funciones binarias); su documentación lista los métodos y requerimientos para interconectar procesos e intercambiar data. Con el crecimiento de las aplicaciones web y móviles, las API se han vuelto omnipresentes. [1]

API de servicio web

Cuanto se utiliza el contexto de desarrollo web, la API se define como el conjunto de solicitudes (*request*) utilizadas en el protocolo de transferencia de hipertexto HTTP, que retornan una respuesta con una estructura especifica de datos *Extensible Markup Language* (XML) o *JavaScript Object Notation* (JSON); la práctica de publicar API ha permitido a las comunidades web crear una arquitectura abierta para compartir contenido y datos entre comunidades y aplicaciones. De esta forma, el contenido que se crea en un lugar puede ser dinámicamente publicado y actualizado en múltiples ubicaciones en la web. [1]

SOURCE-CODE API

Las API de código fuente ofrecen bibliotecas de objetos, clases, etc. Las API de código fuente a menudo se usan en proyectos de desarrollo para crear una aplicación compuesta. Las llamadas se realizan de acuerdo con los estándares del entorno de la aplicación, como J2EE o .NET. [1]

LEGACY API

Contiene una variedad de enfoques, para interfaces de aplicaciones que emplean archivos planos, protocolos de objetos remotos, interfaces de sistema operativo, API de hardware, protocolos de comunicación, colas de mensajes y otros medios. Esto no quiere decir que sea obsoleto, sino es una forma de agrupar una variedad de protocolos antiguos y establecidos, aunque algo más antiguos. Un buen ejemplo sería CORBA (COMMON Object Request Broker Architecture). [1]

Componente de software

Es un elemento de software que se ajusta a un modelo de componentes y se puede implementar y componer de forma independiente sin modificación de acuerdo con un estándar de composición (paquete de software, servicio web). [2]

Modelo de componente

Define los estándares específicos de interacción y composición. La implementación de un modelo de componente es establecer un conjunto dedicado de elementos ejecutables de software requeridos para mantener la ejecución de los componentes que conforman el modelo. [2]

REST

Es un estilo de arquitectura basado en un conjunto de principios que describen cómo se definen y abordan los recursos en red (desarrollar servicios web). REST es popular debido a su simplicidad y al hecho de que se basa en los sistemas y funciones existentes de HTTP para lograr sus objetivos, en lugar de crear nuevos estándares, frameworks y tecnologías. [6]

Al momento de crear interfaces, se debe limitar las interacciones de los métodos CRUD, además de hacerlos simples y fáciles de entender, ya que requiere que el cliente realice una solicitud al servidor para consultar, crear, modificar o eliminar un recurso del servidor. Los métodos de petición (CRUD) son los siguientes:

GET	Obtener un recurso específico (por ID) o una colección de recursos
PUT	Crear o actualizar un recurso conocido (por ID)
POST	Crear o añadir un nuevo recurso
DELETE	Eliminar un recurso especifico por ID

Tabla 1. Métodos de petición (CRUD)

Fuente: https://www.codecademy.com/articles/what-is-rest

Aplicación web

En la ingeniería del software aquellas aplicaciones que los usuarios pueden utilizar accediendo a un servidor web a través de Internet o de una intranet mediante un navegador. Se codifica en un lenguaje soportado por los navegadores web y en la que se confía la ejecución de la aplicación al navegador.

Características Generales:

Son populares por lo práctica al permitir una comunicación más fluida y dinámicamente en la computación Cliente-Servidor. Una página Web puede contener elementos que permiten una comunicación activa entre el usuario y la información (vía Servidor); logrando el usuario acceso a los datos de modo interactivo, gracias a que la página responderá a cada una de sus acciones.

Las aplicaciones web para Internet e Intranet presentan una serie de ventajas y beneficios con respecto al software de escritorio, con lo cual logrará aprovechar y acoplar los recursos de su empresa de una forma mucho más práctica que el software tradicional.

Ventajas de las aplicaciones web:

- Compatibilidad multiplataforma.
- Siempre se mantienen actualizadas.
- Inmediatez de acceso.
- Menos requerimiento de memoria.
- Menos propensas a fallas.
- Múltiples usuarios concurrentes. [5]

Cliente

En la arquitectura REST, los clientes envían solicitudes para obtener o modificar recursos y los servidores se encargan de enviar respuestas a estas solicitudes.

- Un método de petición (*request*) HTTP, que define el tipo de operación que se va a realizar.
- Un encabezado (*header*), que permite al cliente pasar información sobre la solicitud.
- Una ruta (*path*) que va ligada a la operación a realizar de un recurso.
- Un cuerpo (*body*) de mensaje opcional que contiene datos. [6]

Servidor

El servidor es el encargado de recibir las conexiones y responde a las solicitudes para proporcionar acceso a sus servicios. Capaz de proporcionar recursos y gestionar los mismos utilizando las especificaciones HTTP como protocolo de comunicación. [6]

Back-End

Se refiere a todo el desarrollo donde se centran las funcionalidades de una aplicación (lado del servidor). Donde surgirán actualizaciones y cambios y asimismo de encargarse del monitoreo de las mismas. Generalmente consta de tres partes: un servidor, una aplicación y una base de datos. Es lo que comunica la información de la base de datos a la aplicación. Todo lo que el cliente no puede visualizar (bases de datos, servidores, funciones). [7]

Front-End

Se refiere a todo el desarrollo que se centra principalmente en el "lado del cliente", corresponde a toda la apariencia y diseño que va a tener la aplicación, al mismo tiempo se analizará el código, se diseñará la interfaz, se depurará la aplicación y debe garantizar una excelente experiencia de usuario (UX). [7]

PHP

PHP (acrónimo recursivo de PHP: *Hypertext Preprocessor*) es un lenguaje de código abierto (lenguaje de scripting) del lado del servidor, utilizado para el desarrollo web y puede integrarse con HTML. Se tiene la libertad de elegir un sistema operativo y un servidor web. Igualmente, también tiene la opción de utilizar la programación de procedimientos o la programación orientada a objetos (POO), o una combinación de ambos. Adicionalmente una de las características más importantes y sólidas es su compatibilidad con una amplia gama de bases de datos. [8]

Guzzle

Guzzle es un cliente HTTP de PHP que permite enviar solicitudes HTTP de manera sencilla y trivial para integrarse con servicios web.

Algunas de sus ventajas son:

- Interfaz simple para crear consultas (*querys*), solicitudes POST, transmisión de grandes cargas (*streaming large uploads*), transmisión de grandes descargas (*streaming large downloads*), uso de cookies HTTP, carga de datos JSON, etc.
- Puede enviar solicitudes síncronas y asíncronas utilizando la misma interfaz.
- Utiliza interfaces PSR-7 para solicitudes (*request*), respuestas (*response*) y transmisiones (*streams*). Esto le permite utilizar otras bibliotecas compatibles con PSR-7 con Guzzle.
- No existe una dependencia fuerte en cURL, secuencias PHP, *sockets*, entre otros.
- El sistema de middleware le permite aumentar y componer el comportamiento del cliente (*client behavior*). [33]

Framework

Los frameworks ofrecen un enfoque más rígido para el desarrollo, otros permiten una mayor fluidez en el proceso: facilita a los desarrolladores según las necesidades del proyecto o sus propios estilos de trabajo. Se define como un entorno reutilizable que forma parte de una plataforma de software más grande. Están específicamente diseñados para facilitar el desarrollo de aplicaciones de software e incluyen

componentes, como bibliotecas de códigos, programas de soporte, compiladores, conjuntos de herramientas y API específicas que facilitan el flujo de datos.

Los frameworks de aplicaciones web son frameworks de software utilizados para optimizar la aplicación web y el desarrollo de sitios web, servicios web y recursos web. Ejemplo: la arquitectura Model-View-Controller (MVC). [9]

Laravel

Laravel es un framework de aplicaciones web con una sintaxis expresiva y elegante. Laravel facilita las tareas comunes que se utilizan en la mayoría de los proyectos web, como la autenticación, el enrutamiento, las sesiones y el almacenamiento en caché. Combina lo mejor de los frameworks web, incluidos los marcos implementados en otros lenguajes, como Ruby on Rails, ASP.NET MVC y Sinatra. Proporciona herramientas potentes, necesarias para aplicaciones grandes y robustas. Una magnífica inversión de contenedores (encargados de gestionar las instancias) de control, un sistema de migración expresivo y un soporte de pruebas unitarias estrechamente integrado, brindan las herramientas que se necesitan para construir cualquier aplicación. [10]

Redes sociales

Las redes sociales son aplicaciones web, para hacer conexiones con amigos, familiares, compañeros de clase, visitantes y clientes; pueden ocurrir con fines sociales (para hablar, compartir ideas e intereses), comerciales o ambos a través de las distintas redes sociales como: Facebook, Twitter, LinkedIn, Classmates.com y Yelp. Las redes sociales también son un área objetivo importante para los profesionales del marketing que buscan atraer a los usuarios, por su parte contienen contenido creado por cientos o incluso millones de personas diferentes. [11]

Twitter

Twitter es una red social que permite a los miembros registrados publicar publicaciones cortas llamadas tweets. Los miembros de Twitter pueden transmitir tweets y seguir tweets de otros usuarios mediante el uso de múltiples plataformas y dispositivos. Los tweets y las respuestas a los tweets se pueden enviar por mensaje de texto del teléfono celular, cliente de escritorio o publicando en el sitio web de Twitter.com. La configuración predeterminada para Twitter es pública. Los tweets, que pueden incluir hipervínculos, están limitados a 140 caracteres, debido a las limitaciones del sistema de entrega del Servicio de mensajes cortos (SMS) de Twitter. [12]

Facebook

Facebook es una red social gratuita que permite a los usuarios registrados crear perfiles, subir fotos y videos, enviar mensajes y mantenerse en contacto con amigos, familiares y colegas. Esta red social cuenta con una disponibilidad de hasta 37 idiomas diferentes e incluye funciones públicas tales como:

- Marketplace: permite a los miembros publicar, leer y responder avisos clasificados.
- Grupos: permite que los miembros que tienen intereses comunes se encuentren e interactúen.
- Eventos: permite a los miembros publicar un evento, invitar a invitados y hacer un seguimiento de quién planea asistir.
- Páginas: permite a los miembros crear y promocionar una página pública creada en torno a un tema específico.
- Tecnología de presencia: permite a los miembros ver qué contactos están en línea y chatear. [13]

Access Token

El token de acceso (*access token*) es una credencial que puede ser utilizada por un cliente para acceder a una API. Puede ser cualquier tipo de token (como una cadena opaca o un JWT) y está destinado a una API. Su propósito es autorizar para acceder a los APIs y realizar acciones específicas (según lo que permita el alcance del token que se ha otorgado).Asimismo debe usarse como una credencial del portador y transmitirse en un encabezado de autorización HTTP a la API. Actualmente es generado en dos formatos: como cadenas opacas (no pueden ser interpretadas) o como tokens web JSON (JWT), dependiendo de los requisitos del usuario. [16]

Webhook

Los Webhooks son básicamente devoluciones de llamada HTTP definidas por el usuario (o pequeños fragmentos de código vinculados a una aplicación web) que se desencadenan por eventos específicos. Siempre que ese evento desencadenante ocurra en el sitio fuente, el webhook ve el evento, recopila los datos y los envía al URL especificado en forma de solicitud HTTP. Incluso se puede configurar un evento en un sitio para desencadenar una acción en otro sitio. [14]

Web chat

Es una aplicación o módulo que permite a los usuarios comunicarse en tiempo real utilizando interfaces web de fácil acceso (chat en línea), sin necesidad de instalación, permitiendo a los usuarios un acceso instantáneo y siempre estarán actualizados con la última versión del mismo. Igualmente permite a los visitantes (clientes) tener una conversación virtual con su equipo de servicio o ventas a través del sistema de mensajería instantánea dentro del mismo sitio web. [32]

WebSocket

WebSockets representa la evolución en la tecnología web cliente / servidor. Permite establecer una conexión de socket TCP entre el cliente y el servidor (por periodo un tiempo indefinido), que permite que los mensajes sean bidireccionales, permitiendo

una distribución instantánea con poca sobrecarga, lo que da como resultado una conexión con latencia muy baja. [17]

Pusher

Pusher es un servicio alojado que hace que sea fácil agregar datos y funcionalidades en tiempo real a aplicaciones web y móviles. Ya que mantiene conexiones persistentes con los clientes, a través de un WebSocket si es posible, recurriendo a conectividad basada en HTTP, y tan pronto como sus servidores tengan nuevos datos, puedan hacerlo instantáneamente a través de Pusher. [18]

CAPÍTULO III – MARCO METODOLÓGICO

Este capítulo consiste en describir el marco de trabajo establecido por la empresa para llevar a cabo la realización del presente TIG (Servicio web REST API para la aplicación web SocialGest CRM de la empresa Gest Labs, C.A.), se utilizó el marco de trabajo SCRUM y será descrito a continuación.

Características Scrum

- Más transparencia y visibilidad del proyecto.
- Motivación.
- Responsabilidad.
- Equipos auto-organizados.
- Cambios fáciles de adaptar.
- Ahorro de costos.
- Liviano, iterativo e incremental.
- Resultados anticipados.
- Productividad y calidad.
- Entregas

Proceso Scrum

El marco de trabajo Scrum se basa en una serie de iteraciones de longitud fija llamadas *sprints* que brinda al equipo un marco de trabajo para el desarrollo de un proyecto de software en una cadencia regular. Al final de un *sprint* se declaran los hitos que traen consigo una sensación de progreso tangible, logrando enfoque y motivación en el mismo. Las iteraciones cortas también refuerzan la importancia de una buena estimación y un rápido feedback de las pruebas.

Durante un *sprint*, los artefactos visuales como los tableros de tareas y los diagramas de burndown, ayudan al equipo a visualizar, cuales son las tareas que deben realizarse, además de observar y evaluar su desempeño. La oportunidad de mostrar un nuevo trabajo en la demostración de *sprint* es igualmente motivador, y la retroalimentación consistente y gradual que el equipo obtiene de los interesados en cada demostración crea una forma poderosa de desarrollar productos.

Scrum sigue un conjunto de roles, responsabilidades y reuniones que nunca cambian. Se convoca a cuatro reuniones que brindan estructura a cada *sprint*:

- ***Sprint planning* (Planificación de Sprint):** es una reunión de planificación de equipo que determina los objetivos que se van a completar en el próximo *sprint*.
- ***Daily stand-up* (Reuniones diarias):** también conocido como *daily scrum* (scrum diario), es una reunión de 15 minutos para que el equipo de software se sincronice.
- ***Sprint demo* (Demostración de Sprint):** es una reunión para compartir lo que ha realizado el equipo en ese *sprint*.
- ***Sprint retrospective* (Retrospectiva de Sprint):** es la revisión de lo que estuvo bien y lo que no salió bien con las acciones para mejorar el próximo *sprint*.

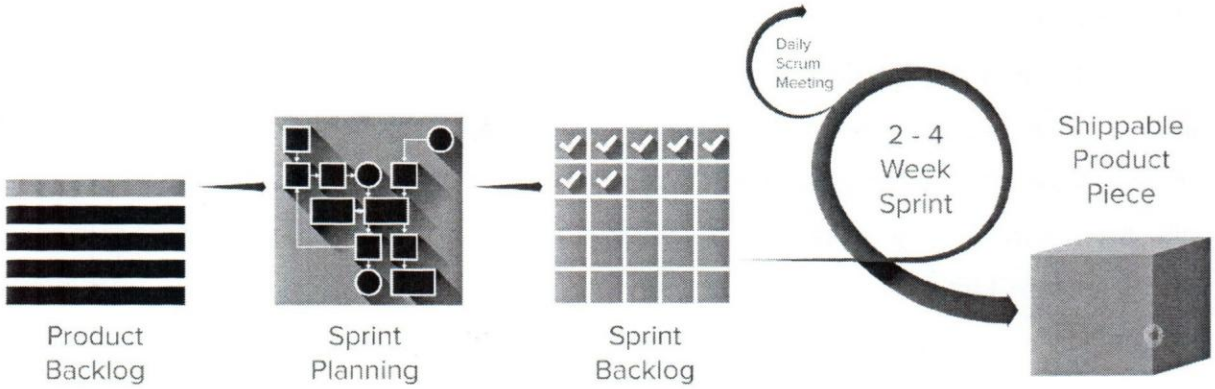


Figura 5. Pasos en el proceso de Scrum

Fuente: https://www.smartsheet.com/agile-vs-scrum-vs-waterfall-vs-kanban

Ya que su desarrollo se efectúa de forma iterativa e incremental. Cada *sprint* posee una duración promedio de 2 a 4 semanas, duración que es establecida al inicio del proceso y al final de cada iteración se debe arrojar como resultado una versión funcional del producto ya sea con nuevos requerimientos Hay un conjunto de pasos específicos e inmutables en el flujo de Scrum especificados en la figura 5:

- **Product backlog:** el propietario del producto (*Product Owner*) y el equipo (*Scrum Team*) se reúnen para priorizar las actividades (requisitos) del *product backlog* (el trabajo proviene de las historias de usuario y de los requerimientos). El *product backlog* no es una lista que se va a cumplir por completo sino una lista de características deseadas para el producto. El equipo de desarrollo se encarga de realizar el trabajo descrito en el *product backlog* durante cada *sprint*.
- **Sprint planning:** antes de cada *sprint* el *Product Owner* presenta las actividades principales del product backlog al equipo en una reunión de planificación de *sprints* (*sprint planning meeting*). Luego, el equipo elige qué trabajo pueden completar durante el *sprint* y mueve el trabajo desde el *product backlog* hasta el *sprint backlog* (que es una lista de tareas para completar en el *sprint*).
- **Backlog refinement/grooming:** al final de un *sprint*, el equipo y el *Product Owner* se reúnen para asegurarse de que el *backlog* esté listo para el próximo *sprint*. El equipo puede eliminar historias de usuarios que no son relevantes, crear nuevas historias, volver a evaluar la prioridad de las historias o dividir historias de usuarios

en tareas más pequeñas. El objetivo de esta reunión de "preparación" es garantizar que el *backlog* solo contenga elementos que sean relevantes y detallados y que cumplan con los objetivos del proyecto.

- **Daily Scrum meetings:** el *Daily Scrum* es una reunión de 15 minutos en la que cada miembro del equipo habla sobre sus objetivos y sobre cualquier problema que haya surgido durante el desarrollo. El *Daily Scrum* ocurre todos los días durante el *sprint* y ayuda a mantener la trayectoria del equipo.
- **Sprint review meeting:** al final de cada *sprint*, el equipo presenta el trabajo que han completado en una reunión de revisión de *sprint* (*sprint review meeting*). Esta reunión cuenta con una demostración en vivo de lo que se realizó en el *sprint*.
- **Sprint retrospective meeting:** también al final de cada *sprint*, el equipo reflexiona sobre cuán bien se está trabajando y se conversa sobre cualquier cambio que deba hacerse en el próximo *sprint*. Se puede hablar sobre qué fue bien, qué salió mal y qué podrían hacer de manera diferente, durante el *sprint*.

Roles

Hay tres roles específicos en Scrum que son:

- **Product Owner:** Es el propietario del producto (*Product Owner*) tiene la visión de lo que se quiere construir y transmite esa visión al equipo. Se enfoca en los requisitos comerciales y de mercado, priorizando todo el trabajo que se necesita hacer. Crea y administra el *backlog* del proyecto, proporciona orientación en los requisitos que se van a realizar próximamente e interactúa con el equipo y otras partes interesadas para asegurarse de que todos entiendan las actividades del product backlog. Administrar el estado y el progreso del proyecto y su objetivo es motivar al equipo con una meta y visión.
- **Scrum Master:** Es el encargado de ayudar al equipo a hacer su mejor trabajo posible. Esto significa organizar reuniones, lidiar con obstáculos y desafíos, y trabajar con el *Product Owner* para garantizar el cumplimiento del *product backlog*. También se asegura de que el equipo siga el proceso de Scrum ya que tiene la autoridad sobre el mismo y podría proponer una nueva cadencia (número de días o semanas en un *sprint* o lanzamiento) de *sprint*.

- **Scrum Team:** El equipo de Scrum está compuesto por cinco o siete miembros. Todos en el proyecto trabajan, se ayudan y comparten el objetivo de cumplir el proyecto. No existen roles definidos en el mismo y completan el conjunto de las actividades propuestas en el *product backlog*. Al mismo tiempo anticipan cuánto trabajo pueden completar en cada iteración.

Además de los roles y las reuniones, Scrum también incluye ciertas herramientas y artefactos, que ayudan a facilitar el desarrollo del proyecto

- **Scrum Board**: Permite visualizar las actividades del *sprint* backlog. Tradicionalmente implica crear tarjetas que vienen siendo las actividades, con notas de observaciones. Generalmente se divide en tres categorías: por hacer, en trabajo en progreso y terminado. El Scrum Team se encarga de actualizar el tablero durante todo el *sprint*.
- **User stories:** Una historia de usuario (*user storie*) describe una función de software desde la perspectiva del cliente. Incluye el tipo de usuario, lo que desea y por qué lo desea así; comparten la siguiente estructura: <tipo de usuario>, quiero realizar <alguna tarea> para poder <alcanzar el siguiente objetivo.> El equipo utiliza estas historias para crear código que cumpla con los requisitos de las historias.
- **Burndown chart:** Representa todo el trabajo pendiente. El backlog es usualmente el eje vertical, con el tiempo a lo largo del eje horizontal. El trabajo restante se puede representar mediante puntos de historia, días ideales, días de equipo u otras métricas. Advierte al equipo si las cosas no van según lo planeado y ayuda a mostrar el impacto de las decisiones.
- **Timeboxing:** Es un período de tiempo establecido durante el cual el equipo trabaja para completar un objetivo. En lugar de dejar que un equipo trabaje hasta alcanzar un objetivo, se define y limita la cantidad de tiempo dedicado a una actividad, se detiene el trabajo cuando se alcanzó el límite de tiempo para evaluar lo que se logró.

- **Icebox:** todas las historias de usuario que se registran pero no se mueven al desarrollo se almacenan en la nevera.

En la figura 6 se puede observar un esquema que denota el proceso y los elementos que forman parte de Scrum.

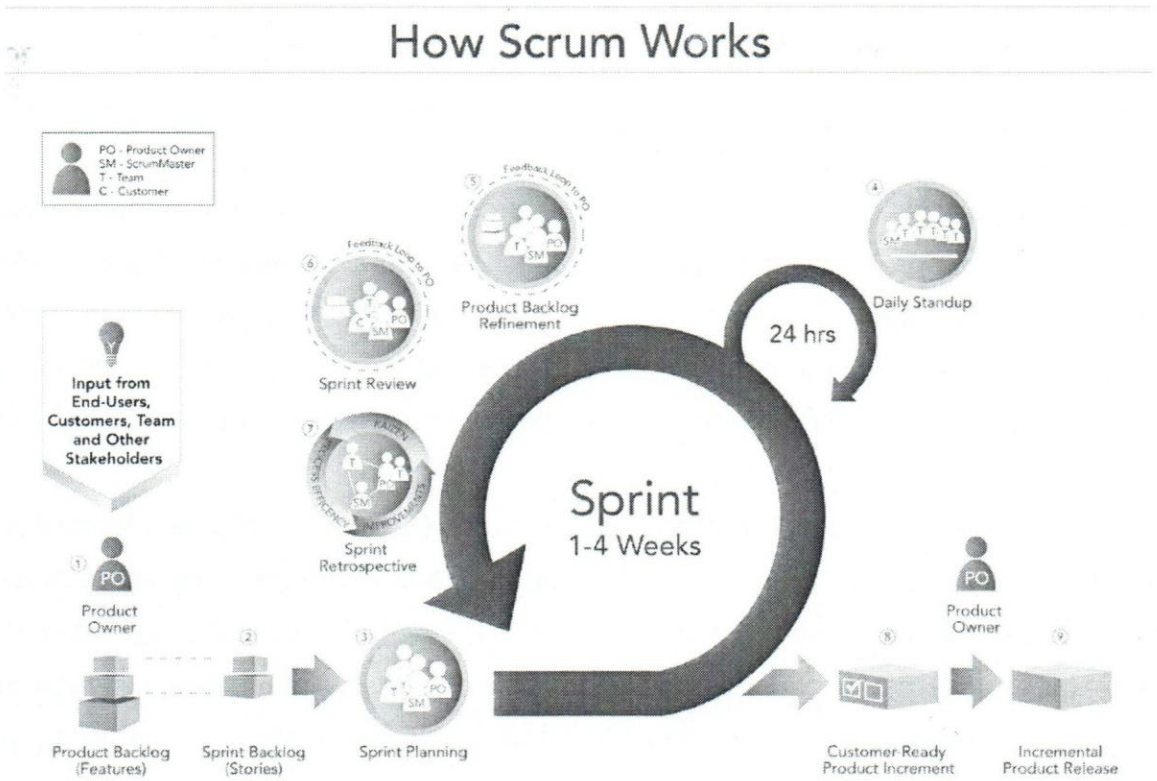


Figura 6. Pasos en el proceso de Scrum

Fuente: https://cdn.thinglink.me/api/image/535528547733733377/1240/10/scaletowidth

Justificación

Se utilizó el marco de trabajo SCRUM (preestablecido en la empresa) para el desarrollo de proyectos, ya que es un marco de trabajo adaptable que cuenta con *sprints* de corta duración es más fácil lidiar con los cambios y adaptarse a ellos, tomando en

cuenta la importancia de una buena estimación y un rápido *feedback*, esto permite que los requisitos más importantes, se desarrollen por orden de prioridad de acuerdo a las necesidades de la empresa. Cabe destacar que la empresa cuenta con 2 clientes sumamente importantes (Movistar y Seguros Caracas) por lo que amerita una ejecución en el menor tiempo y con la mayor efectividad posible.

La comunicación entre el equipo de desarrollo también pasa a ser un punto clave en el desarrollo, por lo que poder llevar a cabo: *Product backlog*, *Sprint Plannings*, *Backlog refinement/grooming*, *Daily Scrums*, *Sprint review meeting* y *Sprint retrospective meeting*, todas estas "ceremonias", mejoran abruptamente el desarrollo de un proyecto ya que ataca todos los puntos claves (dificultades, cambios, nuevos requerimientos, entre otros) que pueden ocasionar problemas más adelante y perjudicar la trayectoria del proyecto. Por otra parte el equipo está al tanto del rendimiento de los participantes de su equipo y como comparten un mismo objetivo, cooperando entre sí para lograrlo.

Por último sus artefactos: el tableros de tareas (*Scrum Board*) y los diagramas de burndown, son poderosos motivadores, puesto a que dan una mejor perspectiva de lo que se está llevando a cabo y lo que se ha hecho. Dando la posibilidad de dar un análisis de rendimiento y desempeño del producto; con una retroalimentación consistente e incremental proveniente de los *stakeholders* a través de las demostraciones del equipo.

Product Backlog – Planificación del Sprint

Product Backlog y Sprint Backlog (Ver Tabla 2 y 3).

PRODUCT BACKLOG	
ID	**FUNCIONALIDAD**
1	El ingreso a la aplicación por los usuarios registrados en la plataforma SocialGest CRM, mediante usuario y contraseña.
2	Consultar, agregar, modificar, eliminar canales digitales en la aplicación (Twitter y Facebook)
3	Consultar, agregar, modificar y eliminar casos provenientes de las redes sociales
4	Consultar y agregar (funcionalidad de envío) mensajes a clientes a través de las redes sociales y el chat.
5	Poder generar casos a través de la plataforma
6	Escalamiento de casos a través de correo
7	Consultar, agregar, modificar y eliminar información del cliente
8	Consultar y agregar información relevante de la bitácora
9	Consultar y modificar información relevante del perfil de los usuarios
10	Consultar, agregar, modificar, y eliminar plantillas
11	Consultar la permisología
12	Consultar, agregar, modificar y eliminar equipos, con su debido proceso de invitación
13	Consultar, agregar, modificar y eliminar órdenes de compra
14	Realizar las pasarelas de pago
15	Canjear periodo de prueba gratuita.

Tabla 2. Product backlog

Fuente: Elaboración propia

SPRINT BACKLOG		
Sprint	**Actividades**	**Objetivos**
0	• Entrevista con el Product Owner. • Levantamiento de información y definición de los requerimientos. • Definición del Product Backlog. • Investigación de *frameworks* y tecnologías de desarrollo. • Selección y definición de criterios de evaluación. • Elección y aprobación del *framework* y tecnología de desarrollo.	**18**
1	• Diseño e implementación de la estructura REST API de la aplicación. • Reestructuración de la base de datos. • Rediseñar el proceso de venta de los servicios de la aplicación.	**18,15**
2	• Implementación del módulo de *login*. • Implementación del módulo de permisología. • Implementación del módulo de equipo.	**1,10,12**
3	• Implementación del módulo de agregar usuarios. • Implementación del módulo de perfil	**2,8**
4	• Implementación del módulo de casos • Implementación del módulo de mensajes • Implementación del módulo de clientes	**3,4,6**
5	• Implementación del módulo de órdenes en la aplicación y pagos • Implementación del módulo de prueba gratuita	**13,14,17**
6	• Implementación del módulo de plantillas • Implementación del módulo de mensaje automático	**9,11**
7	• Implementación del módulo de caso interno • Elaboración de funcionalidad para el sistema de escalamiento	**5,16**
8	• Implementación del módulo de bitácora	**7**

Tabla 3. Sprint backlog

Fuente: Elaboración propia

CAPÍTULO IV - DESARROLLO

Este capítulo consiste en describir las actividades realizadas empleando el marco de trabajo Scrum, se establecieron nueve *sprints* con una duración de tres semanas cada uno.

Sprint 0

Para comenzar en el *sprint* se estableció la primera reunión con el *Product Owner*, para conocer al equipo y ver el papel que cumple cada uno, cómo se aplicaría el marco de trabajo Scrum, duración de los *sprints*, días para las "ceremonias" y toda la información necesaria para poder comenzar el desarrollo. Los roles y responsables que se presentaron fueron los siguientes:

- Product Owner: José Enrique Martorano (CEO de GestLabs.C.A).
- Scrum Master: Ángel Guacache (Jefe del Equipo de Desarrollo).
- Scrum Team: José Andara (Desarrollador Back-End).
  Miguel Coppola (Desarrollador Back-End).
  Robert Galeano (Desarrollador Front-End).
  Angel Rivillo (Desarrollador Front-End).

Luego se definieron los requerimientos y funcionalidades que abarcará la aplicación en actividades que se encuentran en el Product Backlog (adicionalmente los departamentos de soporte y mercadeo participaron en este evento) del proyecto. Posteriormente se procedió a hacer el listado de requisitos necesarios para la inclusión del framework de desarrollo Back-end.

¿Por qué la utilización de un *framework* es importante al momento de desarrollar?

Algunos beneficios de los *frameworks* son:

- Permite que el desarrollo sea más rápido debido a que proporcionan operaciones CRUD y gran cantidad librerías que permiten ahorrar una gran cantidad de tiempo.
- Permite a los desarrolladores escalar sistemas fácilmente.
- El mantenimiento del código es mucho más fácil dado que el código de la aplicación es conciso y fácil de usar.
- El modelo MVC asegura un desarrollo rápido.
- Son mejores para proteger la aplicación web de amenazas de seguridad comunes.

En vista de que el lenguaje de programación a usar en el desarrollo del proyecto en el área de Back-End es PHP, se seleccionó por la lista de requisitos necesarios establecidos para la inclusión del framework, expuesta en la tabla 4.

Requerimientos para la selección del *framework*	
ID	**Requerimientos**
1	Debe permitir manejo de conexiones con bases de datos MySQL
2	Debe seguir el patrón modelo vista controlador (MVC)
3	Debe poseer un manejador de paquetes para manejar las dependencias o librerías del proyecto
4	Debe permitir realizar servicios web que sigan la arquitectura RESTful
5	Debe soportar Amazon SQS

Tabla 4. Requerimientos para la selección del *framework*

Fuente: Elaboración propia

Se investigó en base a los requerimientos establecidos en la tabla 4 y fueron seleccionados los siguientes 3 *frameworks*: Laravel, CakePHP y CodeIgniter, ya que estos están basados en la arquitectura MVC (modelo-vista-controlador) y resaltan por ser los más sólidos en los artículos [21] [22].

Para establecer una selección, se elaboraron dos tablas una de acuerdo a la comparación establecida en el artículo [21] empezando por: tendencia y popularidad, estructura y actualizaciones, bibliotecas y ayuda en línea, integración de bases de datos, facilidad de desarrollo y creación de plantillas, costo del desarrollador, luego a partir de los requerimientos faltantes se realizó una tabla adicional para evaluar si cumplen con los mismos.

A continuación se presenta la tabla 5 que contiene la comparación entre los *frameworks*, calificados del 1 al 5, tomando en cuenta que 1 es la calificación más baja y 5 la más alta:

Tabla Comparativa *Frameworks*					
FWK	**Tendencia y popularidad**	**Estructura y actualizaciones**	**Bibliotecas y ayuda en línea**	**Integración de bases de datos**	**Facilidad de desarrollo e interfaz**
Laravel	5	5	5	4	4
CakePHP	5	4	4	4	4
CodeIgniter	3	3	4	3.5	4

Tabla 5. Comparación entre los diferentes *frameworks* en base a los requerimientos.

Fuente: https://therightsw.com/codeigniter-vs-laravel-vs-yii-vs-cakephp/

En tendencia y popularidad Laravel actualmente se encuentra en su pico máximo de popularidad, gracias a su fuerte que es su codificación "expresiva" (fácil de entender), con una calificación de 5, mientras que CakePHP sigue siendo uno de los más utilizado pero lentamente ha venido decayendo su uso desde 2009 y cada vez más sigue siendo opacado por los demás *frameworks,* este cuenta, con una calificación de 5, CodeIgniter siempre ha sido popular y proporcionó una gran mejora desde su actualización 2.x, sin embargo actualmente ha tenido un giro ligeramente descendente gracias a la popularidad de Laravel, con una calificación de 3.

De acuerdo a estructura y actualizaciones, Laravel trae consigo una herramienta conocida como Artisan, que te permite crear modelos, vistas y controladores, además de configurar tareas programadas, colas (SQS) y comandos personalizados, que incluye Eloquent que ayuda a interacción con la base de datos, permite crear métodos y obtener información de la misma, sus actualizaciones son comunes y recurrentes, asimismo como se mencionó anteriormente cumple con la arquitectura MVC, obtuvo una calificación de 5, CakePHP implementa la misma arquitectura pero su enfoque es muy metódico y muy estricto, su desarrollo y todavía continúa con mucha actividad en Github, además posee una librería que soporta SQS (https://github.com/lorenzo/cakephp-sqs), fue asignado con calificación de 4, CodeIgniter con respecto a su estructura se basa libremente en la programación orientada a objetos pero se puede utilizar de acuerdo a las necesidades (MVC), gracias a su actualización 3.x se ha vuelto mucho más estricto y ya el inconveniente que presentaba antes que era insensibilidad con las mayúsculas y minúsculas fue solucionado, actualmente la última versión es la 3.1.7 y por último CakePHP cuenta con una librería para el sdk de Amazon que soporta SQS (https://github.com/linuxjuggler/codeigniter-amazon-sdk), consiguió una calificación de 3. Destacando que todos cumplen con los requerimientos 2 y 5 de la tabla 5.

Laravel posee su propia documentación oficial que es muy elaborada y útil pero permite obtener ayuda de un sitio específicamente para este *framework* que es: Laracast.com, además de admitir un gran número de librerías en un proyecto de Laravel (Composer), fue calificado con 5. Se pueden encontrar cientos de bibliotecas en Packagist. CakePHP tiene una larga historia, por lo que encontrar ayuda y tutoriales en línea de CakePHP es muy fácil, ya que hay muchos recursos, se calificó con 4. Adicionalmente viene con Composer y debería ser bastante fácil agregar bibliotecas de proveedores, CodeIgniter proporciona una gran cantidad de funcionalidad incorporada y su sitio web tiene una guía práctica para su uso. Sin embargo, existe un consenso general de que CodeIgniter no proporciona el mismo nivel de bibliotecas de terceros que otros *frameworks*, obtuvo 4 de calificación. Cumpliendo así con el requerimiento 3 de la tabla 5.

En cuanto a base de datos Laravel viene con su propia herramienta ORM. Laravel viene con migraciones (migrations) que permite con facilidad realizar cambios de base de datos y control de versiones. Del mismo modo, Laravel posee una herramienta conocida como Seeder que se utiliza para introducir datos de prueba. También tiene paquetes para cada base de datos disponible e incluye Redis en sus paquetes y fue calificado con 4. El acceso a la base de datos es muy fácil en CakePHP, éste viene con ORM y herramientas de creación de consultas, viene con soporte para bases de datos relacionales, incluyendo Oracle. Aunque, para las bases de datos NoSQL debe obtener bibliotecas de terceros, con una calificación de 4. CodeIgniter proporciona soporte a bases de datos relacionales tales como MySQL o PostgreSQL (bastante limitado) de forma predeterminada. A pesar de ello, CodeIgniter no viene con Composer de manera predeterminada, por lo que obtener bibliotecas de terceros para bases de datos como Mongo sería una tarea difícil en la que tendría que probar copiando los archivos manualmente, obtuvo una calificación de 3.5. Viene con su propia herramienta ORM. Cumple. Conteniendo así el requerimiento 1 de la tabla 5.

Al momento de desarrollar interfaces Laravel viene con el motor de *templates* (vistas) Blade, que es bueno para el desarrollador *Front-end*, con el beneficio de que se pueden heredar *templates* y definir secciones Al mismo tiempo no es necesario saber PHP para utilizarlo. Ante todo igualmente facilita el desarrollo al proporcionar características como enrutamiento (*routes*), migraciones de bases de datos (*migrations*), Eloquent que ayuda con la interacción con la base de datos, comandos, páginas integradas, entre otros. Acertó con una calificación de 4. CakePHP posee un ciclo de desarrollo más rápido dado su apego a sus estándares. Su estructura al momento de desarrollar *templates* sigue siendo PHP crudo, pero permite el uso de *templates de* Twig que debería ser genial para los desarrolladores de UI, se calificó con 4. CodeIgniter es fácil de usar y por lo tanto, la curva de aprendizaje no es un problema. Además de que no hay un motor de templates, fue asignado con una calificación de 4.

Adicionalmente se realizó un estudio con la finalidad de evaluar el nivel de uso (popularidad) de cada *frameworks*, a través de los años (2015-2018) para ello se usó la herramienta "Google Trends ", que nos arrojó como resultado a Laravel como el marco de trabajo con mayor usabilidad a nivel mundial entre los años 2015, 2016, 2017 y principios de 2018. (Ver Figura 6)

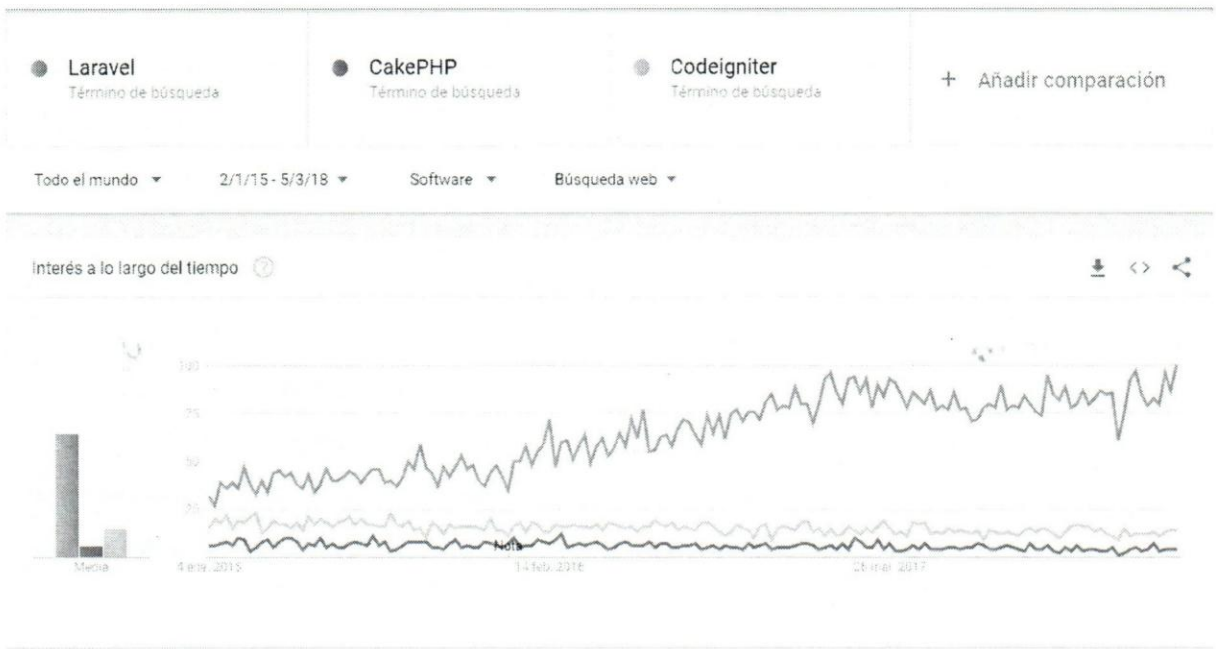


Figura 6. Gráfica comparativa entre Laravel, CakePHP, y Codeigniter

Fuente: https://trends.google.es/trends/explore?cat=32&date=2015-01-02%202018-03-05&q=Laravel,Codeigniter,Yii

Habiendo expuesto el análisis de la tabla 5, quedan pendiente dos requerimientos solicitados para culminar la evaluación, el primero es que debe permitir realizar servicios web que sigan la arquitectura RESTful por lo que se elaboró una nueva tabla para especificar cuáles frameworks cumplen con este requerimiento.

Tabla Comparativa *Frameworks*	
FWK	**Arquitectura RESTful**
Laravel	✓
CakePHP	✓
CodeIgniter	✗

Tabla 6. Comparativo en base al requerimiento de arquitectura RESTful.

Fuente: Propia

Laravel y CakePHP son *frameworks* que poseen una sección en su documentación exclusivamente para elaborar arquitecturas RESTful (RESTful Web APIs), puesto que ya viene incluido los mismos, mientras que CodeIgniter depende de librerías para poder realizar esto. Y como último requerimiento, el framework debe soportar Amazon SQS, los 3 poseen librerías para integración de servicio de colas de mensajes de Amazon.

Para concluir se procedió a la totalización de los puntos, obteniendo los siguientes resultados: Laravel 23 puntos, CakePHP 21 puntos, y CodeIgniter 17.5 puntos, posteriormente se estableció una reunión para conversar y seleccionar el *framework* a utilizar en el desarrollo del producto; por la obtención del mayor puntaje en base a sus características fue seleccionado Laravel.

Sprint 1

En primer lugar se estableció la nueva arquitectura del sistema la cual se muestra a continuación en la figura 7.

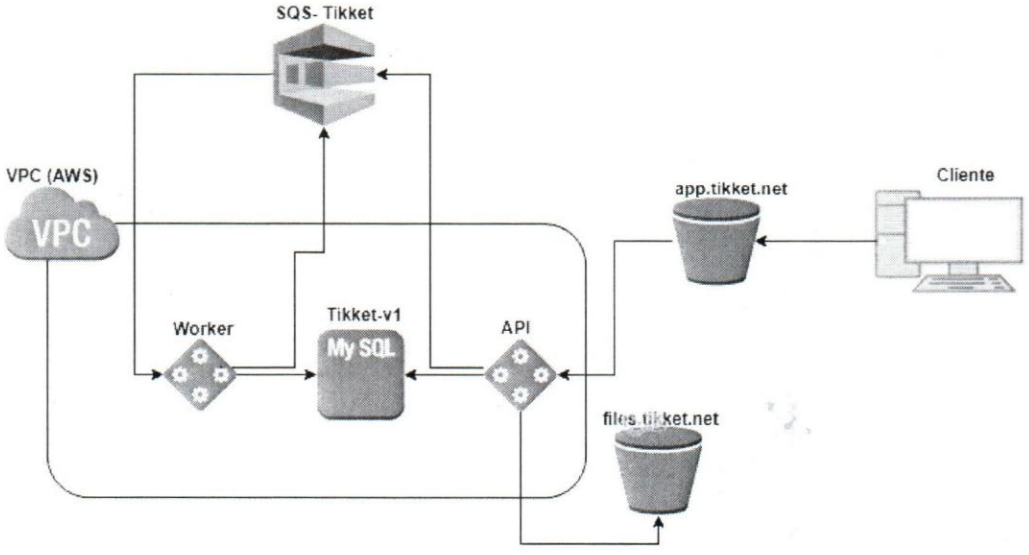


Figura 7. Arquitectura del sistema
Fuente: Elaboración propia

SocialGest CRM está conformada por una VPC (*Virtual Private Cloud*) que proporciona a los usuarios acceso a los servidores EC2 a través de una red privada virtual basada en IPsec donde se reside la API, en este caso como se puede apreciar en la figura 7 existen 2 servidores EC2 "Worker" y "API", el "Worker" es el encargado de ejecutar los comandos (*commands*) y despachar los trabajos (*jobs*) al SQS (*Amazon Simple Queue Service*) donde serán procesados. El "API" se encargará de recibir las peticiones provenientes de los clientes a través del portal app.tikket.net y dar respuesta a las mismas (éxito o error). Igualmente se dispone de una base de datos relacional (RDS) de amazon "Tikket-v1" para todas las tareas de almacenamiento de información, además por otro lado se dispondrá de un S3 Bucket (*Amazon Simple Storage Service*) files.tikket.net responsable de almacenar y obtener archivos que se hayan subido en la aplicación .

Por otra parte, se definió la estructura del API siguiendo los lineamientos de Instagram (siempre la respuesta del servidor será exitosa pero se clasificará por *meta code*). En la figura 8 se puede apreciar cómo estará compuesto el JSON de respuesta exitosa: meta contendrá un código específico de éxito, data contendrá la información relevante solicitada.

```
"meta": {
    "code": Código de éxito,
    "message": "Mensaje"
},
"data": {
    "Información":  "Información"
}
```

Figura 8. Estructura del JSON de respuesta exitosa del API de SocialGest CRM

Fuente: Elaboración propia

Cuando se realiza una petición y su respuesta es errónea, el JSON contendrá en meta un código de error, acompañado de un mensaje de error visualizado en la figura 9.

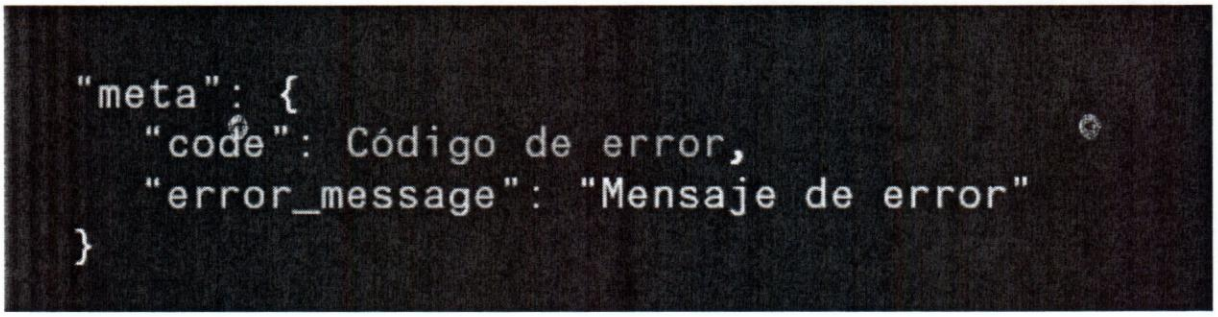


Figura 9 Estructura del JSON de respuesta errónea del API de SocialGest CRM
Fuente: Elaboración propia

La definición de la estructura de los servicios web, se basó en arquitectura REST usando los métodos GET, POST, UPDATE y DELETE.

En segundo lugar se procedió a analizar cada uno de los ER, añadiendo en la nueva propuesta las tablas generadas anteriormente, con el fin de observar las tablas que faltan por definir para cumplir con los requisitos o actividades especificadas en el *product backlog* y poder alcanzar el producto aspirado por la empresa.

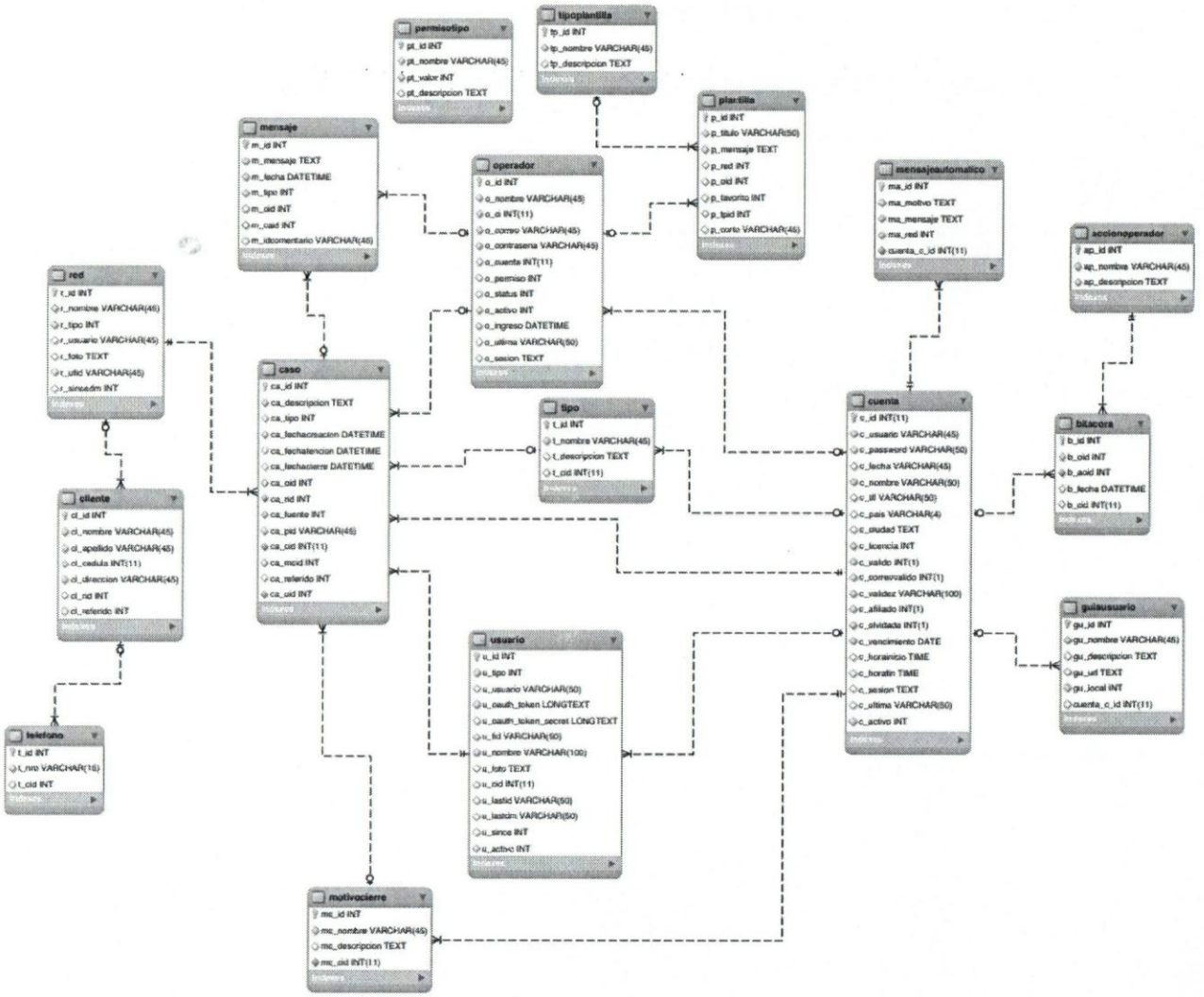


Figura 10 Modelo entidad relación de la base de datos actual

Fuente: Elaboración propia

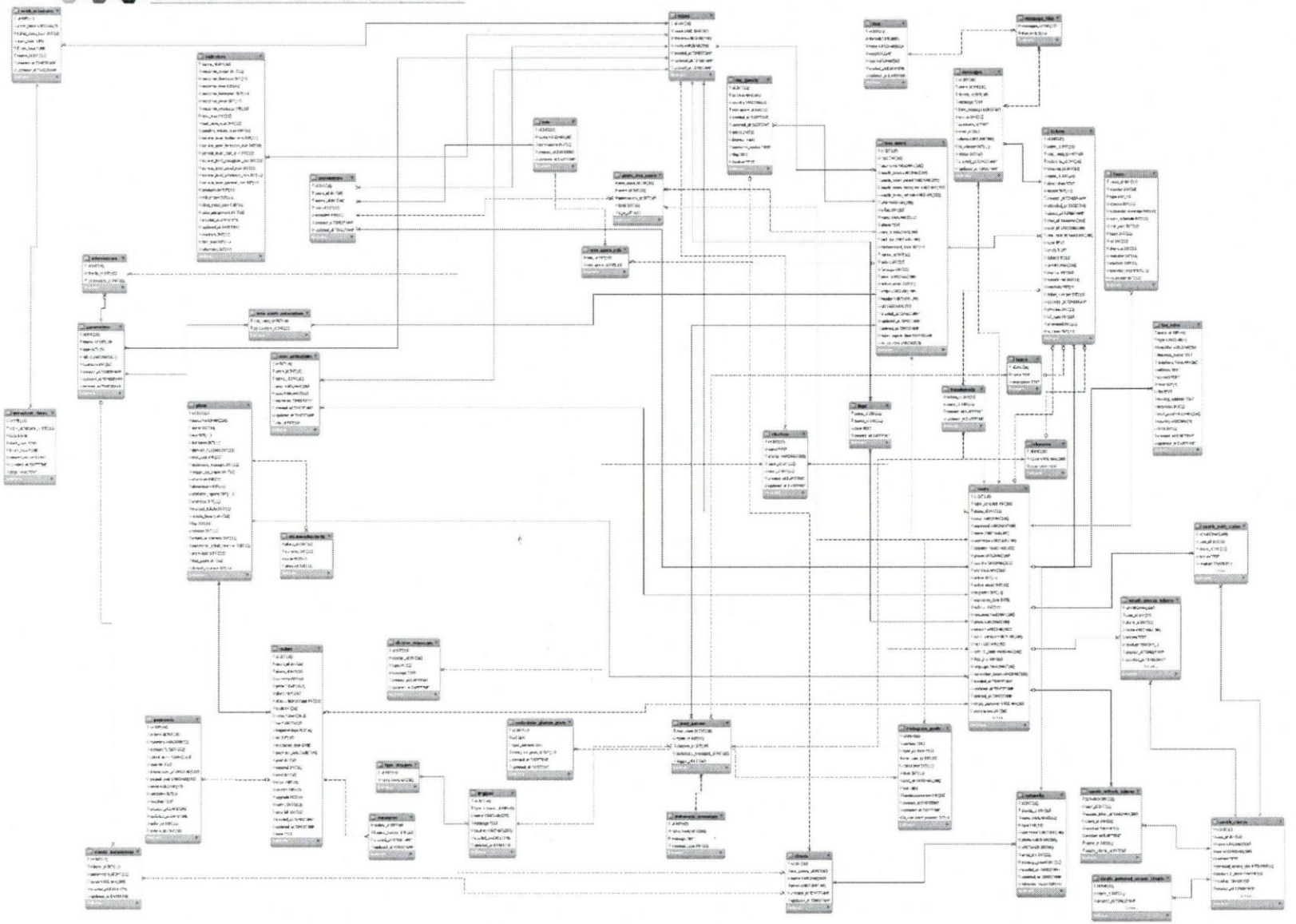


Figura 11 Modelo entidad relación de la base de datos actual

Fuente: Elaboración propia

En la figura 10, se puede apreciar el ER actual de la empresa, muchos de los nombres ya están en español y se estandarizaron por buenas prácticas en inglés, asimismo a todos los nombres de las tablas de las bases de datos se les agregó una 's' al final para que al momento de utilizar el *framework* fuera de utilidad la herramienta que facilita los métodos e interacciones con la base de datos (Eloquent), las fechas y horas, trabajadas en formato UTC (*Coordinated Universal Time*) por las diferentes zonas horarias *timezones* de los usuarios.

De igual forma en la figura 11 se puede visualizar las tablas nuevas necesarias para alcanzar los requerimientos establecidos en el *product backlog*, son las siguientes:

- **orders:** se encarga de la creación de órdenes a partir de que el cliente selecciona su plan de preferencia en el carrito de compra, generando así un registro que le permitirá ver su pedido y poder seleccionar su pago de preferencia (Instapago, MercadoPago, PayPal, Stripe, Transferencia).
- **plans:** contiene toda la información respecto a los planes (costo por tipo de moneda y los beneficios de cada uno)
- **payments:** almacena toda la información relevante a los pagos realizados por las pasarelas de pago con su moneda, tipo de pago, monto a cancelar, IVA y total. Además de sus atributos.
- **measures:** almacena el número de licencias que va a solicitar el cliente durante su compra, a partir de que se genera una orden el cliente deberá colocar el número de licencias que desea.
- **teams:** guarda la información del equipo (nombre) que haya establecido el cliente para invitar a sus sub-usuarios.
- **clients:** almacena la información relevante de los clientes que contacten en la aplicación a través los canales digitales que se hayan asociado (Facebook, Twitter y Chat).
- **my_guests:** contiene toda la información relevante al IP, país de procedencia de donde contacta una persona en el chat. Otros atributos fueron agregados en la figura 11 debido a otros requisitos solicitados por la empresa.

- **shorters_messages:** se encarga de guardar todos los mensajes asociados a una plantilla, para contener una bandeja de mensajes.
- **logs:** se almacena toda la información relevante de acuerdo a las acciones de un usuario dentro de un equipo. Parametrizado por equipo y usuario.
- **permissions:** se guarda las relaciones entre el equipo y los sub-usuarios, además de su rol y conocer si fue aceptada la invitación para pertenecer al equipo o se encuentra en espera por respuesta.
- **rols:** almacenará la información de roles que genere el usuario asociados.
- **rols_rrss_users:** contiene todos los canales digitales asociados a un rol en específico, sin permitir la repetición de canales digitales entre roles de un mismo equipo.
- **type_permissions:** tabla que almacena todos los permisos establecidos por la empresa.
- **tax_infos:** guarda la información fiscal del usuario (teléfono, dirección, contacto, rif, ciudad, entre otros). Otros atributos fueron agregados en la figura 11 debido a otros requisitos solicitados por la empresa.
- **user_activations:** se almacena la información relevante a invitaciones, cambio de contraseña y validaciones pendientes de correo al momento de registrarse.
- **parameters:** contendrá toda la información adicional del cliente.
- **files:** se encarga de guardar todos los archivos (imágenes y documentos) que se hayan adjuntado o recibido en los tickets tipo chat.
- **exception_dates:** almacena toda la información referente a los días de excepción de los mensajes automáticos (automatic_messages).

Se pueden observar las tablas oauth_access_token, oauth_clients, oauth_personal_access_clients, oauth_auth_codes y oauth_refresh_tokens, utilizadas para el desarrollo del módulo de *login*.

Las tablas que se modificaron fueron las siguientes: cuenta pasó a ser *users*, usuario pasó a ser *rrss_users*, bitácora pasó a ser log, cliente pasó a ser *clients*, red pasó a ser *networks*, caso pasó a ser tickets, plantilla pasó a ser shorters y mensaje pasó a ser

messages. Igualmente se agregaron atributos extras para cumplir con los requerimientos establecidos:

- **users:** se le agregó el atributo "permissions", en el cual se almacena los permisos correspondientes al usuario.
- **tickets:** se le agregó el atributo "internal", booleano con valores 0(negativo) y 1(afirmativo), con el fin de verificar si un ticket fue realizado internamente (corresponde al módulo de caso interno establecido para el sprint 7).
- **shorters:** se colocó el atributo "teams_id" para establecer plantillas generales para el equipo y es un booleano con valores 0(negativo) y 1(afirmativo).
- **users, tickets, networks, rrss_users, type, closure, triggers, transferreds:** se agregó "created_at" y "updated_at" que son atributos necesarios para que Laravel mantener registro de creaciones y actualizaciones.

Por último se estableció el rediseño del proceso de compra de SocialGest CRM, como se indica en el capítulo 1 objetivo 15 el proceso de venta de los servicios de la aplicación SocialGest CRM es: el cliente contacta al departamento de ventas, planifica la reunión con el mismo, luego se reúnen y se cotizan las necesidades (servicios) que desea el cliente, el cual tiene la potestad de aprobar o rechazar, finalmente, al ser aprobado realiza el pago y seguidamente se activan los servicios.

Cuando se efectuaba un pago la única forma posible era a través de transferencia por lo que el cliente siempre debía ser el encargado de notificar al departamento de ventas cuando se había realizado un pago ya sea a través de llamadas o correos, además de esperar por la respuesta del mismo. Como se observa en la figura 12.

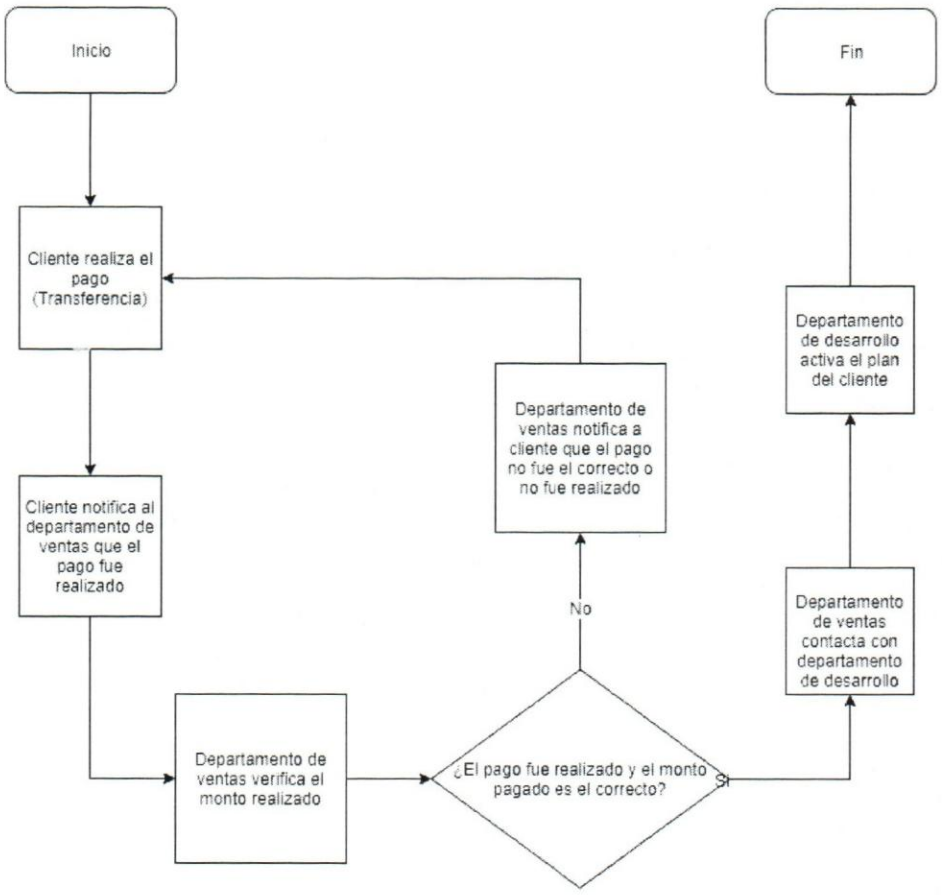


Figura 12 Proceso actual para la activación de un plan

Fuente: Elaboración propia

Ya que se llevó a cabo el rediseño del proceso de compra de SocialGest CRM, al momento de generar una orden y realizar los pagos, se podrán realizar transferencias y si es realizado a través de tarjeta de crédito o mediante alguna de las siguientes pasarelas de pago: Instapago, MercadoPago, Paypal y como adicional Stripe para personas exclusivamente ubicadas en cualquier parte de Venezuela dispondrán de las 2 primeras (Instapago, MercadoPago), por otra parte las últimas 2 (Stripe y Paypal) será para personas extranjeras que deseen pagar en la plataforma, el plan se activa de forma automática, sin embargo en el caso que este sea realizado por depósito o por transferencia dicha persona puede declarar el pago a través de la aplicación. Evitando

todo el inconveniente generado en el proceso anterior, se puede apreciar en la figura 13 el nuevo proceso.

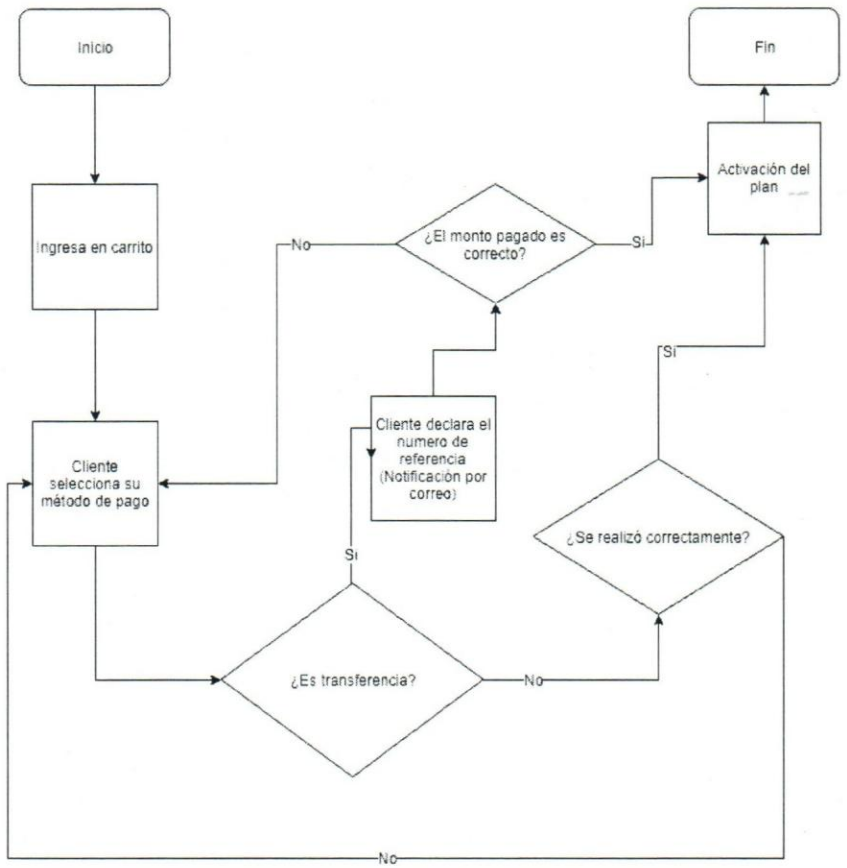


Figura 13 Nuevo proceso para la activación de un plan

Fuente: Elaboración propia

Al culminar, se procedió a realizar una retrospectiva, tomando en cuenta los avances, criterios y apreciaciones, además de presentar lo que se obtuvo durante el *sprint* (*Sprint review meeting* y *Sprint retrospective meeting*).

Sprint 2

En este sprint se procedió a desarrollar el módulo de *Login* para comenzar con el desarrollo del API lo primero que se realizó fue investigar a fondo qué librerías existían para el manejo de sesiones, gracias al uso de Laravel, su documentación cuenta con una sección exclusivamente para API de autenticación (Passport), después de haber definido el uso de PHP Passport [10] para el desarrollo de la API, se procedió a instalar el paquete en el proyecto dado que contiene un comando de instalación a través del uso de la línea de comando Artisan (*php artisan passport:install*) gracias a esto se autogeneran las tablas necesarias para realizar la autenticación de usuarios (oauth_access_token, oauth_clients, oauth_personal_access_clients, oauth_auth_codes y oauth_refresh_tokens), suministrando los atributos necesarios para el manejo de sesiones ver figura 14.

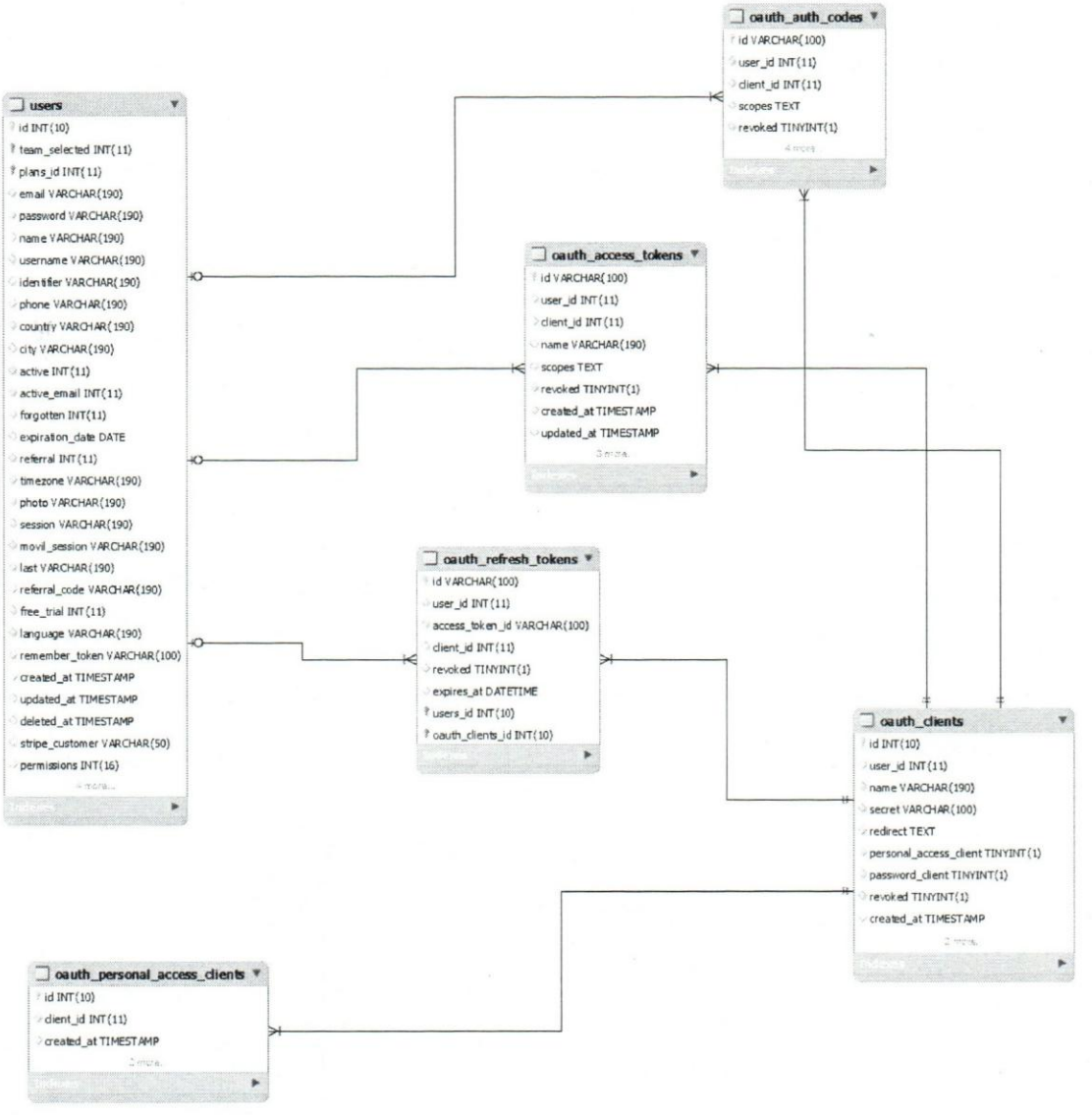


Figura 14. Modelo ER de autenticación de usuario

Fuente: Elaboración propia

Se realizó una función que mediante el protocolo HTTP (POST) se encarga de recibir los parámetros requeridos de email, contraseña, para que la librería PHP Passport, que cuenta con un servidor interno de Oauth2 con los datos obtenidos se encargue de generar las credenciales únicas y correspondientes al usuario que desea ingresar. La respuesta contendrá un token de acceso (*access token*) de tipo Bearer para

garantizar al usuario autorización en la aplicación, con una duración es de 15 días, un token de refrescamiento del token (*refresh token*) para garantizar la renovación del token de acceso a usuarios que todavía sigan dentro de la misma y finalmente un código de validación de refrescamiento (*refresh code*) que resulta ser el encriptamiento del email de usuario para corroborar quien es el usuario que desea renovar las credenciales. Además de toda la información correspondiente al usuario que será utilizada por *Front-end* (correo, nombre, username, foto, equipo seleccionado, tours, entre otros) .Se crearon una serie rutas (*path*) de acceso dispuestas en el servidor para el consumo de este servicio. Con sus respectivos códigos de éxito (*success*) y error.

Para el módulo de permisología, se definió una tabla con todos los permisos posibles de la aplicación, estos van atados a los planes de SocialGest CRM, cada plan lleva consigo una lista de posibles permisos, el API fue realizado para uso exclusivamente interno y la lista de permisos se puede apreciar en la tabla acompañado de su valor.

Tabla de Permisología	
Módulo o Acción	**Valor**
Bitácora	1
Canales de atención	2
Mensaje automático	4
Estadísticas	8
Descarga de estadísticas	16
Susurro	32
Horario de operadores	64
Generación casos internos	128
Plantillas	256
Historial	512
Invitar usuarios	1024
Generar informe de cierre	2048

Tabla 7. Tabla de permisología

Fuente: Elaboración propia.

Como se mencionó anteriormente cada plan posee un permiso base, para verificar cuales son las posibles acciones de un usuario al momento de utilizar la plataforma, garantizando confianza y seguridad para el producto; los usuarios superiores a plan gratuito tienen la posibilidad de crear roles acompañados con los permisos del plan actual, por lo que al momento de almacenar los permisos, los mismos se suman y se guarda el total, al momento de efectuar la verificación se transforman en números binarios tanto el total como el valor del permiso, completando con 0 a la izquierda del número menor hasta llegar a la longitud del mayor, se procede a realizar la operación de lógica (AND) entre los dos números binarios y obteniendo el resultado se devuelve a número decimal, si el valor es igual al permiso puede acceder, en el caso contrario el acceso es negado.

Para el desarrollo de módulo de equipos, se realizaron las funcionalidades de crear, consultar, modificar y eliminar equipos provenientes de un usuario, sin embargo al momento de registrarse un usuario a la aplicación de SocialGest CRM, este deberá ingresar el nombre de un equipo porque es obligatorio para la creación de una cuenta. Esto permite establecer una jerarquía en la aplicación donde un usuario dueño de un equipo con un plan superior al gratuito tiene la posibilidad de invitar a más usuarios a pertenecer parte de un equipo y este podrá definir

Se adaptó la funcionalidad de registro, puesto que el mismo proceso necesita obligatoriamente de un equipo para poder culminarse, donde se solicitará el nombre (único) del mismo para poder realizar su creación y será indispensable al momento de ingresar, ya que si un usuario fue expulsado de un equipo (único equipo que posee) al momento de ingresar deberá realizar un paso adicional que será la creación del mismo. Además se incluyó el desarrollo de la funcionalidad para invitar usuarios y por ende se estableció desarrollar funcionalidades para la creación, modificación, listado y eliminación de roles que son fundamentales para establecer una jerarquía de trabajo en el equipo, además de poder especificar los permisos con respecto a canales digitales dentro de la aplicación.

Al momento de invitar a un usuario se especifica el correo y se puede establecer un rol específico o se puede invitar como con un rol general que garantiza todas las permisologías posibles, de acuerdo al plan del usuario dueño del equipo y con todos sus canales digitales asociados. A su vez se desarrolló las funcionalidades para eliminar un miembro del equipo y cambiar el rol de un operador en un equipo. Por último se realizó nuevamente una reunión (*Sprint review meeting* y *Sprint* retrospective) conocer los módulos obtenidos, los cuales fueron verificados y aceptados,

Al culminar, se procedió a realizar una retrospectiva, tomando en cuenta los avances, criterios y apreciaciones, además de presentar lo que se obtuvo durante el *sprint* (*Sprint review meeting* y *Sprint retrospective meeting*).

Sprint 3

Para desarrollar el módulo de agregar usuarios se procedieron a realizar las funcionalidades de crear, consultar, modificar y eliminar, canales digitales, a su vez la empresa tuvo que crear las aplicaciones necesarias para poder autenticar usuarios de las siguientes redes sociales: Twitter y Facebook [34] [35]. Luego se procedió a investigar qué librerías existen para poder establecer un token de acceso y poder utilizar los APIs necesarios de estas 2 redes sociales. Se clasificaron con valores numéricos en el atributo "type": 1 para Twitter, 2 para Facebook, 3 Chat.

Para Twitter se utilizó la librería de Guzzle (https://github.com/guzzle/guzzle) que contiene la herramienta de Guzzle Subscriber que permite trabajar con Oauth 1.0 que es utilizado para tener acceso a la información de un usuario de Twitter sin la necesidad de utilizar su contraseña por lo que es indispensable para realizar la autenticación en Twitter porque trabaja con ese tipo de versión. Para ello se hizo la funcionalidad que contiene el *callback* que proporcionará el token de acceso perdurable. Antes que nada se procede

a generar un *Handler Stack* ya que es importante comprender que varias opciones de solicitud utilizadas por Guzzle requiere que su middleware contenga la misma estructura, luego se procede a crear el objeto tipo Oauth1 que contiene el "consumer key", "consumer secret" ambos identificadores de la aplicación y el "redirect_uri" (*callback*) suministrado por el *Front-end* se crea un objeto tipo Client de Guzzle para poder realizar la solicitud POST a la siguiente ruta y poder obtener el token de solicitud para la ventana emergente de autorización usuario (https://api.twitter.com/oauth/request_token). luego de que el usuario autoriza la aplicación se vuelve a generar un objeto tipo Oauth1 que contiene el "consumer key", "consumer secret", "oauth_token" (token) y "oauth_verifier" (verificador de token) y se vuelve a realizar una petición POST a la siguiente ruta (https://api.twitter.com/oauth/access_token) para obtener el token de acceso, por último se genera otro objeto Oauth1 con consumer key", "consumer secret", "oauth_token" realizando ahora una petición GET a la siguiente ruta incluyendo el nombre de la red social (canal digital) (https://api.twitter.com/1.1/users/show.json?screen_name='nombre_red_social') después se procede a almacenar en base de datos la información de ese canal digital más su token de acceso. Por otra parte para refrescar la información del canal digital se utiliza la librería de Laravel "Thujohn" [25].

Para Facebook también se utilizó la librería de Guzzle solo que Facebook trabaja con Oauth 2.0 por lo que es necesario solo definir un objeto tipo Client de Guzzle y se establece un arreglo con los parámetros necesarios para generar la autorización "code" código de autorización generado desde el *Front-end*, el "client_id" y "client_secret" de la aplicación (identificadores) y el "redirect_uri" (*callback*) con esto se realiza una petición GET a la ruta especificada por Facebook para obtener el token de acceso de corta duración a la aplicación (https://graph.facebook.com/v2.8/oauth/access_token), inmediatamente se realiza una segunda petición esta vez con los parámetros de "client_id", "client_secret", "grant_type" (credenciales del usuario en este caso el "fb_exchange_token") y "fb_exchange_token" (token de acceso) con esto obtendremos el token de duración (ilimitada) y finalmente se realiza una petición GET con el parámetro "query" que es un arreglo que contiene el "access_token", "fields" (id,nombre,permisos,foto) y "limit" (para no utilizar la limitación por defecto de Facebook) consiguiendo

así la información importante de los canal digitales que posee el usuario y que será almacenada en la base de datos, tomando en cuenta de que sólo se pueden almacenar *fan pages* y por lo tanto tuvo que filtrarse el contenido (eliminando la cuenta personal). Se debe realizar una petición POST adicional para subscribirse al servicio de Webhook (https://graph.facebook.com/v2.8/me/subscribed_apps) con el parámetro query conteniendo el "access_token" únicamente. Se hace uso de la librería de Laravel "SammyK" [16] para refrescar la información del canal digital del usuario.

Para el chat se definió una ruta HTTP POST para la creación del mismo, suministrando los siguientes parámetros: nombre del chat, correo (opcional) para tickets fuera de horario, es necesario indicar que viene incluido la ruta del Widget (URL predefinido) autogenerada para poder hacer uso de él.

.

En el módulo de perfil se desarrolló la funcionalidad necesaria para consultar y editar la información básica de la cuenta, ya sea su nombre completo, correo, username, país, zona horaria, idioma aparte podrá cambiar su contraseña con opción a ser modificado excluyendo el correo, además podrá observar sus datos dentro de la aplicación referente a los tickets atendidos, en bandeja, cerrados, transferidos y sus permisos. También tendrá la posibilidad de modificar la contraseña del usuario en la aplicación. Al culminar nuevamente se procedió a realizar una retrospectiva, tomando en cuenta los avances, criterios y apreciaciones, además de presentar lo que se obtuvo durante el *sprint* (*Sprint review meeting* y *Sprint retrospective meeting*).

Sprint 4

Se elaboró el complemento con las funcionalidades necesarias para procesar y listar la información proveniente de los canales digitales (Twitter, Facebook y Chat), para Twitter existen 2 maneras para comunicarse: por mención y por mensaje directo. Por lo tanto se tuvieron que elaborar 2 funcionalidades encargadas de procesar la información que llega a un canal digital de Twitter y almacenar la misma ya sea como mensaje o ticket expuesto anteriormente, seguidamente se estableció un comando (*command*) que

se ejecutará cada 1 minuto (para no superar las peticiones límites del API de Twitter) con su debido trabajo (*job*) correspondiente a las funcionalidades. Se trabajó nuevamente con la librería "Thujohn" que permite facilitar el trabajo con las llamadas al API de Twitter, la primera funcionalidad se encarga de procesar y filtrar las menciones (tweets) del canal digital con una petición GET al API de Twitter (statuses/mentions_timeline), a partir de que un canal de Twitter es agregado se empiezan a recibir las menciones correspondientes al mismo y se almacena el último ID de la mención que fue listado al momento de finalizar el trabajo (*job*) para que al momento de realizar la petición nuevamente no liste las menciones creadas antes que esa mención, con respecto a la segunda funcionalidad que es para los mensajes directos se realiza una petición con una petición GET al API de Twitter (direct_messages) u al igual que la anterior se empiezan a recibir los mensajes directos correspondientes al mismo y se almacena el último ID del mensaje directo que fue listado al momento de finalizar el trabajo.

Figura 15. Solicitud de servicio de Webhook

Fuente: Elaboración propia

Para Facebook se realizaron 2 funcionalidades para el procesamiento de la información al igual que Twitter, una para todo el manejo de comentarios, publicaciones de visitantes (*feeds*) utilizando el servicio de Webhook, que debe ser solicitado en la sección de productos de la página de desarrolladores de Facebook en la aplicación como se puede apreciar en la figura 15 se debe establecer las rutas específicas (*path*) para poder recibir solicitudes GET y POST (todas las notificaciones de cambios se enviarán

como solicitudes POST e incluirán una carga JSON) y un token de verificación. Asimismo se enviará una solicitud GET para verificar conexión durante algún proceso de configuración o cuando se actualice la suscripción (Webhook) a un tema existente. Además cuando la ruta establecida de GET reciba una petición se debe realizar lo siguiente:

- Verificar que el hub_verify_token coincida con el valor que suministraste al activar el webhook. Esta es una comprobación de seguridad para autenticar la solicitud e identificar el webhook.
- Responde con el valor hub_challenge de esta manera, podemos saber que el Webhook está configurado correctamente y que la respuesta es auténtica.

Por último se enviará una solicitud HTTPS POST a la URL de devolución de llamada (rutas) cada vez que haya actualizaciones para los campos de cualquier tema al que se haya suscrito.

Para el chat se estableció una funcionalidad HTTP POST, para el almacenamiento de un ticket en la aplicación solicitando los siguientes parámetros: identificador del chat (para conocer a qué chat se dirige el cliente), correo, nombre y pregunta, a su vez estará acompañado de un token encriptado para mayor seguridad, ya que será necesario para poder establecer tanto la conversación como obtener la información; esta funcionalidad será requerida para el desarrollo del Web Chat.

En cuanto al complemento de envío de mensajes se desarrollaron 3 funcionalidades bases que gestionarán toda la parte de envío, en Twitter se utilizó nuevamente la librería "Thujohn" [24] esta vez dependiendo del tipo de mensaje que el usuario desea enviar en un ticket de Twitter (mención o mensaje directo), se realiza una solicitud POST al API de Twitter para mención (statuses/update) o mensaje directo (direct_messages/new) suministrando los tokens de la aplicación y del canal digital (verificación).

Figura 16 Conversación entrelazada (*Threaded Conversation*)

Fuente: Elaboración Propia

Para Twitter se especifica el mensaje que se quiere enviar, en caso de que sea mención se añade el identificador de la misma para mantener la conversación "entrelazada" como se puede ver en la figura 16 con respecto a Facebook se hizo uso de la librería "SammyK" [25] nuevamente dependiendo del tipo de mensaje que el usuario desea enviar en un ticket de Facebook (comentario o mensaje directo) se solicita el token de acceso del canal digital, para poder responder una publicación de visitante o un comentario se utilizó el servicio POST del API de Facebook para comentarios (https://graph.facebook.com/v2.8/"identificador"/comments).

En cambio para responder un comentario con un mensaje directo, se ha utilizado el servicio POST del API de Facebook, para respuestas privadas. (https://graph.facebook.com/v2.8/"identificador"/private_replies) y se ha procededido a almacenar la conversación con el servicio GET del API de Facebook (https://graph.facebook.com/v2.8/'identificador_fanpage'/conversations?fields=participants&limit=100) que contiene el identificador de la fanpage y permite obtener el identificador de una conversación, debido a que el servicio de API de respuestas privadas no permite más de 1 respuesta por comentario, para responder un mensaje directo se utilizó específicamente el servicio,(https://graph.facebook.com/v2.8/"identificador"/messages) que necesita del identificador de la conversación para especificar a cual se dirige. Cabe destacar que existen 2 excepciones fundamentales al momento de realizar enviar el dicho mensaje:

- No se pueden enviar mensajes directos a publicaciones de visitantes.
- No se pueden enviar más de un mensaje directo a un comentario.

Parar culminar en el chat se introdujo una ruta (*path*) de servicio POST para las conversaciones relacionadas a tickets tipo chat, que mediante el token de verificación generado en el ticket, solicita el parámetro de mensaje para realizar el envío de parte del cliente y en el caso del operador solicita su parámetro de autenticación en la aplicación (Bearer) y así logran establecer una conversación, adicionalmente se tuvo que crear una funcionalidad aparte para la subida de archivos (imágenes o documentos) al servidor S3 (files.tikket.net) que estarán enlazados a los mensajes con las siguientes especificaciones:

- Solo podrán ser archivos en formato pdf, jpg, png, jpeg, doc, docx, xlsx, xls, csv.
- Pesar un máximo de 30Mb.

Para el módulo de mensajes se crearon los métodos necesarios para crear, consultar, modificar y eliminar tickets y mensajes. Se clasificaron con valores numéricos

en el atributo "source" de la tabla "tickets" y "message" como se puede apreciar en la tabla 8:

SOURCE	DESCRIPCIÓN
0	Mención de Twitter
1	Mensaje directo de Twitter
2	Publicaciones de visitantes (*feeds*)
3	Mensaje directo de Facebook
4	Comentario de Facebook
5	Comentario de comentario de Facebook
6	Conversaciones del chat

Tabla 8. Valores correspondiente al atributo "source"

Fuente: Elaboración propia

A continuación se presenta el reglamento que se estableció para trabajar el módulo de mensajes:

- Si un cliente contacta a un canal digital por primera vez, se deberá almacenar como un ticket nuevo ("created_at" y "updated_at" actualizados con la hora y fecha de contacto) y su texto será tomado como descripción del ticket y guardará como primer mensaje.
- Si un mismo cliente contacta a un canal digital y no posee tickets abiertos (atributo en la tabla ticket "closed_at" con valor nulo) se deberá almacenar como un ticket nuevo ("created_at" y "updated_at" actualizados con la hora y fecha de contacto) y su texto será tomado como descripción del ticket y guardará como primer mensaje.
- Si un mismo contacta a un canal digital y ya cuenta con un ticket abierto se deberá almacenar actualizar el atributo "updated_at" del ticket a la hora y fecha de contacto y su texto será almacenado como mensaje.

También es importante aclarar los siguientes atributos que serán claves al momento de guardar los datos.

- **Network:** Atributo referente al cliente que contactó al canal digital, se debe tomar en que si es primera vez que contacta, debe crear un nuevo registro en la tabla referente a la misma, si ya contacto antes solo se trabaja con el ID con el que fue almacenado.
- **Rrss_user:** Atributo referente al canal digital que fue contactado por el cliente para realizar su solicitud.
- **Chat_id:** Contiene los IDs de conversaciones en Facebook y hash encriptado de seguridad para el chat.
- **Post_id:** Comprende de una variedad de IDs provenientes a conversaciones y menciones de Twitter, comentarios, feeds y mensajes de Facebook.

A su vez la funcionalidad que se encargará de consultar que listará los tickets que posee un equipo, se clasificarán con un status como se puede apreciar en la tabla 9, seguidamente se retornan 3 arreglos para facilitar la clasificación en *Front-end* (nuevos y atendidos, atendidos por otros y cerrados).

SOURCE	DESCRIPCIÓN
0	Nuevo
1	Respondido
2	Cerrado
3	Atendido por otro
4	Transferido
5	Pendiente por respuesta

Tabla 9. Status de tickets

Fuente: Elaboración propia

Se utilizó Pusher para el manejo de WebSocket en la aplicación y poder trabajar con peticiones en tiempo real para el WebChat donde se tuvo que hacer una funcionalidad aparte para procesar los eventos (Webhook) de Pusher tipo Client event generados desde *Front-end* para que pudieran ser procesados y ejecutados debidamente, además se tuvieron que elaborar eventos en Laravel para el manejo de Broadcasting en toda la aplicación.

Luego se construyeron las funcionalidades necesarias para almacenar y mostrar toda la información referente al cliente que contacto a la empresa (nombre, correo) y los parámetros adicionales que son generados al momento de crear el equipo y pueden ser modificados (Teléfonos). Para los casos del chat se obtendrá el IP y país de procedencias se realizan a través de una llamada GET en *Front-end* a un API especializado en obtener información relacionada con el IP donde se obtiene el mismo y país de procedencia. Al culminar nuevamente se procedió a realizar una retrospectiva, tomando en cuenta los avances, criterios y apreciaciones, además de presentar lo que se obtuvo durante el *sprint* (*Sprint review meeting* y *Sprint retrospective meeting*).

Sprint 5

En este Sprint se culminó el módulo de carrito de compra que corresponde a las (órdenes y pasarelas de pago), con respecto a las ordenes se realizaron 3 funcionalidades bases para gestión de dicho modulo la generación de órdenes, activación de planes, renovación.

Para las órdenes se crearon los métodos de agregar, consultar y eliminar ya que pueden ser generadas por el usuario de acuerdo a las necesidades del plan de su preferencia, indicando a su vez el número de licencias (sub-usuarios) que necesita para su equipo, dependiendo de la ubicación se mostraran los precios en moneda extranjera o local como se expone en el sprint 1, solo puede existir una orden vigente al momento

que fue creada, si ya tiene una orden y desea modificarla debe eliminar la anterior lo que será un borrado suave (*soft delete*) en la base de datos con el fin de dejar registro. Las órdenes poseen fecha de vencimiento y no al ser pagadas en el plazo disponible son eliminadas con un comando encargado de buscar las órdenes que hayan expirado.

Existen órdenes de renovación que son aquellas órdenes que se generan automáticamente por el sistema, al estar cerca de su plazo de vencimiento (10 días antes), comparando la fecha de expiración del usuario y se genera una copia de la última orden pagada, recalculando nuevamente con los precios vigente en la base de datos. Se creó un comando (*command*), encargado de ejecutarse una vez al día y se delega de buscar todos los usuarios que poseen su plan próximo a vencerse que no posean nuevas órdenes creadas.

Para concluir la activación de órdenes se creó un comando (*command*), para establecer la lógica de los pagos, que se encargará de buscar en la tabla de payments los pagos ingresados que no hayan sido verificados y tomando en cuenta que:

- El pago realizado sea a una orden existente.
- El pago realizado sea a una orden sin pago registrado.
- El pago se haya realizado con la misma moneda que la orden.
- El pago haya sido de monto correcto.

Por último, si el pago cumple con todas las validaciones mencionadas anteriormente, se procede a realizar la activación del plan, si la orden a la que se le realizó el pago es de renovación, se le suma a su fecha de vencimiento la cantidad de meses de servicios que le corresponda según la orden, de lo contrario si es una orden inicial esa cantidad de meses se suma al día actual. De igual forma se calcula y se asignan los permisos de acceso de la cuenta, según el plan y los adicionales que el cliente compró.

Luego para las pasarelas de pago se realizó una investigación exhaustiva para la inclusión de los 4 métodos de pagos primero se trabajó con los métodos nacionales para la inclusión de MercadoPago se utilizó la librerías de "santigraviano" [27] de Laravel y se procedió a elaborar una funcionalidad HTTP GET para mostrar la orden en mercado pago especificando el ID de la orden, el título de la orden y el valor total por último se elaboró la funcionalidad HTTP POST para procesar el pago con los siguientes parámetros devueltos por MercadoPago: identificador del pago, comisión, monto, correo e identificador de la orden, siguiendo todos los lineamientos del API de MercadoPago [26]. Para Instapago se utilizó la librería de "socialgest" [28] y se realizó la funcionalidad HTTP POST que recibe los siguientes parámetros para generar el pago en esta pasarela (monto, descripción, nombre del portador de la tarjeta, cedula, cvc, fecha de expiración), siguiendo los lineamientos de su API [27] si la respuesta es exitosa es almacenado el pago.

Para los extranjeros se utilizó Paypal con la librería "sh0umik" [29] y cuenta con un servicio de Webhooks, especificado en su API [30], se utilizó para agilizar el proceso de verificación de los pagos, la ventana de pago es procesada en *Front-end* y se elaboró únicamente la función HTTP POST para registrar el pago con la información proveniente del evento del Webhook almacenando el monto, comisión, correo, identificador de transacción y el tipo de moneda. Para finalizar se utilizó la librería "cartalyst" [36] de Laravel para Stripe con el fin de procesar los pagos y se desarrollaron 2 funcionalidades HTTP POST adicionales para dicho funcionamiento la primera para procesar la orden en Stripe con los siguientes parámetros requeridos: token de Stripe, tipo de moneda, monto y descripción luego de que el pago sea procesado exitosamente, Stripe también cuenta con servicios de Webhooks [31], con la respuesta de dicho servicio, se almacena el tipo de moneda y el id de transacción.

Por su parte para los pagos realizados por transferencia o depósito se desarrolló una función HTTP POST que recibe los parámetros de número de referencia, banco de origen y banco de destino que se encargará de notificar por correo electrónico, los datos del cliente más los datos de la orden (plan, monto, entre otros) al personal de

administración quienes son encargados de registrar el pago. Todo lo elaborado anteriormente queda solventado el problema principal que presenta la empresa expuesto, ya que sus clientes actuales podrían adquirir sus servicios a través de la página web, donde seleccionará el plan y procederá a realizar el pago, luego de ser verificado el mismo, se le activarán los servicios correspondientes a dicho plan(ver figura 17).

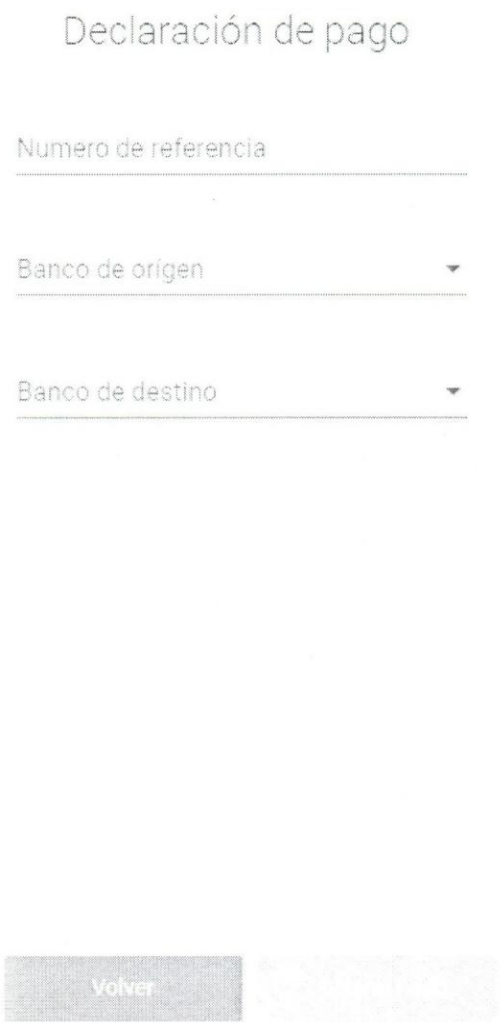


Figura 17. Ejemplo de notificación de pago.

Fuente: Elaboración propia.

Para concluir se elaboraron las funcionalidades necesarias para las activaciones de prueba gratuita siguiendo el reglamento proporcionado por la empresa:

- El proceso de activación tiene que ser por 15 días.
- Solo puede activarse una vez por usuario.
- Luego una vez que se activó el periodo de prueba, no se puede cambiar o volver a activar.
- No se puede activar un periodo de prueba gratuita una vez que el cliente ya haya realizado un pago.

Se creó un servicio HTTP POST para la activación de prueba gratuita, que permite al usuario gozar de todos los beneficios del plan que haya escogido, se le otorgan los permisos y se modifica el atributo "free_trial" a 1, para corroborar que se haya realizado esta activación. Al culminar nuevamente se procedió a realizar una retrospectiva, tomando en cuenta los avances, criterios y apreciaciones, además de presentar lo que se obtuvo durante el *sprint* (*Sprint review meeting* y *Sprint retrospective meeting*).

Sprint 6

Para el desarrollo del módulo de plantillas se realizaron los métodos de agregar, consultar, modificar y eliminar plantillas además se incluyó agregar, consultar, modificar y eliminar los mensajes contenidos en una plantilla. Se estableció que para mayor utilidad y poca repetitividad se pudiera almacenar una lista de mensajes correspondientes a una plantilla, para hacer de la atención a los clientes sea un servicio más amigable.

Se incluyó la opcionalidad de que las plantillas puedan ser generales para el equipo o puede ser única por usuario y se realizó la funcionalidad encargada de elegir un mensaje aleatorio, al momento de especificar el *shortcut*, se obtiene la lista mensajes, que luego a través de un aleatorio se filtra un mensaje, con el que va a ser contacto el cliente.

En el caso del módulo de mensaje automático, se realizaron todos los métodos necesarios para agregar, consultar, modificar y eliminar los mensajes automáticos que estarán asociado a un canal digital, asimismo se tuvo que desarrollarse todas las funcionalidades necesarias para la creación, modificación y consulta de un horario laboral (*work schedule*) ya que el mismo es necesario para determinar el envío de un mensaje automático tanto en Twitter como en Facebook como se puede ver en la figura 18.

Figura 18 Ejemplo de mensaje automático en Twitter

Fuente: Elaboración propia

El horario laboral (*work schedule*) es asignado para el equipo en general y se genera a partir de la creación del equipo con un estándar (desde las 00:00:00am a 23:59:59pm, con todos los días de semana como laborales) debido a que es fundamental para la recepción de los tickets, el usuario seguidamente se debe dirigir a su sección para modificarlo ya que corresponderá a los horarios que establece el usuario dueño del equipo para la gestión de los tickets. El usuario coloca 2 horas: una de inicio y otra de fin, igualmente añade los días específicos de trabajo (lunes, martes, miércoles, entre

otros) que corresponderá a un cadena de caracteres (*string*) de longitud 7 formado de 1 y 0 donde las posiciones representan los días laborales, a su vez tiene la opción de colocar fechas de excepciones al horario establecido (días festivos, eventos, entre otros) especificando el horario de atención (inicial y final) de la misma. Los tickets que correspondan a horas fueras de este horario, pasaran a tomar parte de la gestión de mensaje automático.

Para esto se realizó un evento (*event*) tipo disparador (*trigger*) en Laravel encargado verificar que al momento de que se almacena un ticket nuevo esté comprendido entre las horas del horario laboral, en el caso de lo contrario dependiendo del tipo de canal digital que se contactó, se enviara un mensaje de la bandeja de mensajes automáticos cargada por el usuario al correspondiente canal digital, para esto se reutilizaron los métodos de envío de mensaje definidos en el sprint 4. Al culminar nuevamente se procedió a realizar una retrospectiva, tomando en cuenta los avances, criterios y apreciaciones, además de presentar lo que se obtuvo durante el *sprint* (*Sprint review meeting* y *Sprint retrospective meeting*).

Sprint 7 (No Contemplado)

Para el desarrollo del módulo de caso interno, se realizó una funcionalidad utilizando los métodos de envío de mensaje establecidos en el sprint 4, solo que en los parámetros requeridos se incluye uno adicional que es "internal", para saber que es un caso interno, se especifica nuevamente el tipo de mensaje (mención o comentario en caso de Twitter, comentario o mensaje directo en caso de Facebook) y se procede a la creación del ticket con el mensaje especificado por el usuario hacia al cliente que desea contactar, siempre y cuando no exista un ticket abierto con dicho cliente, ya que si es la excepción se notificara al usuario que ya existe un ticket creado con este cliente.

Con respecto a la funcionalidad correspondiente al envío del histórico por correo, lo que se realizó fue integrar las funcionalidades, realizadas aparte correspondiente al

envío de correo de Laravel con Mailgun, permitiendo a un usuario poder generar un ticket tipo correo, para poder hacer seguimiento de lo que ha ocurrido, comunicándose a través de correo electrónico.

Sus 2 clientes más importantes, no contaban con un funcionalidad que le permitiera escalar los casos y cuando algún operador del personal deseaba realizar el escalamiento de un ticket debía seguir el ciclo de escalamiento mostrado en la en la figura 3. Para evitar este ciclo se elaboró una nueva funcionalidad para transferir los tickets de acuerdo a los roles por equipo, con el fin de evitar este proceso.

Se solicitará el usuario al que será transferido (escalado) del ticket y una vez realizado se podrá gestionar a partir del último mensaje recibido en el ticket, teniendo como historial todo la gestión del operador que lo atendió, permitiendo así una mejor atención a los problemas y una clasificación certera de lo que ha ocurrido y si se pudo solventar el inconveniente como se muestra en la figura 4.

Sprint 8 (No Contemplado)

Para finalizar con el desarrollo, se procedió a elaborar el último modulo bitácora encargado de almacenar las acciones significativas tanto para el equipo como para el usuario. Se definió un middleware de registro con el fin de procesar todas las solicitudes entrantes a la aplicación, filtrando por los códigos de respuesta respectivos a acciones dentro de la misma, para esto se realizó la tabla con todos códigos de respuesta exitosa para almacenar que una acción fue realizada correctamente.

En vista de que es de suma importante para la seguridad de los clientes conocer lo que están realizando sus miembros del equipo, además para manejar respaldo de esta información que puede considerarse valiosa en la aplicación para posibles estudios y análisis de acciones que realizan los usuarios.

Código	Acción
513	*Login*
66	Motivo de cierre creado
72	Motivo de cierre modificado
77	Motivo de cierre eliminado
307	Motivo de contacto creado
9	Motivo de contacto modificado
308	Motivo de contacto eliminado
534	RRSS eliminada
527	RRSS actualizada
539	RRSS agregada (Todo menos chat)
502	RRSS agregada (Chat)
84	Ticket atendido
87	Ticket transferido
245	Rol creado
249	Rol modificado
252	Rol eliminado
93	Horario laboral creado
95	Horario laboral modificado
97	Horario laboral eliminado
604	Nuevo indicador creado
19	Mensaje automático creado
25	Mensaje automático actualizado
30	Mensaje automático eliminado
3	Nueva orden creada
160	Activar free trial
806	Pago paypal
2003	Mercado pago
201	Pago instapago

548	Equipo creado
555	Equipo actualizado
561	Equipo eliminado
507-599	Invitar al equipo
559	Eliminar miembro el equipo
580	Modificar el rol del miembro de un equipo
183	Acortador creado
184	Acortador modificado
186	Mensaje del acortador creado
188	Mensaje del acortador modificado
189	Acortador eliminado
191	Mensaje de acortador eliminado

Tabla 10. Acciones de bitácora

Fuente: Elaboración propia

No obstante se elaboraron las funcionalidades para consultar la información de las acciones importantes tanto para el usuario como en general para el equipo, así se tiene una perspectiva de lo que se ha hecho y para el almacenamiento de la información que será utilizada en el *middleware*. Al concluir la última retrospectiva, tomando en cuenta los avances, criterios y apreciaciones del módulo de bitácora, además de presentar lo que se obtuvo durante el *sprint* (*Sprint review meeting* y *Sprint retrospective meeting*).

Al este Sprint se mostraron los módulos obtenidos de la aprobación de los mismos por parte del Product Owner y el Scrum Master.

CAPÍTULO V - RESULTADOS

Finalizado todo el desarrollo de los objetivos planteados anteriormente, se describirán los resultados obtenidos en este Trabajo Instrumental de Grado

Resultados obtenidos por objetivos

1. Desarrollar la interfaz de programación de aplicaciones de Login.

Se desarrolló y se obtuvo como resultado la interfaz de programación (API) correspondiente al manejo de sesiones de usuario para permitir su consumo por parte de los módulos de la plataforma, especialmente por el módulo de ingreso encargado de gestionar las credenciales obtenidas en el cliente mediante la comunicación con esta API. El resultado obtenido se detalla en el cuarto capítulo detallado en el Sprint 2, anexo A.

2. Desarrollar la interfaz de programación de aplicaciones de Agregar Usuarios.

Se obtuvo la interfaz de programación de aplicaciones (API) que permite crear, consultar, editar y eliminar usuarios (canales digitales) especificados (Twitter, Facebook, Chat), con todas las funcionalidades necesarias para llevar a cabo el proceso de verificación con la aplicación en Twitter, Facebook, exceptuando el Chat y añadiendo suscripción al Webhook en el caso de Facebook. Todo esto especificado en el cuarto capítulo en el Sprint 3, anexo D.

3. Desarrollar la interfaz de programación de aplicaciones de Casos.

Se obtuvo la interfaz de programación de aplicaciones (API) que permite crear, consultar, editar y eliminar casos (tickets), con todas las funcionalidades necesarias para procesar los mensajes de los canales digitales especificados (Twitter, Facebook, Chat). Todo esto especificado en el cuarto capítulo detallado en el Sprint 4, anexo E.

4. Desarrollar la interfaz de programación de aplicaciones de Mensajes.

Se obtuvo la interfaz de programación de aplicaciones (API) que permite crear, consultar, editar y eliminar mensajes, con todas las funcionalidades necesarias para llevar a cabo el envío mensajes en los canales digitales especificados (Twitter, Facebook, Chat). Todo esto especificado en el cuarto capítulo en el Sprint 4, anexo F.

5. Desarrollar la interfaz de programación de aplicaciones de Caso Interno.

Se obtuvo la interfaz de programación de aplicaciones (API) que permite crear, consultar, editar y eliminar los casos (tickets) internos, con todas las funcionalidades necesarias para crear un ticket dirigido al canal digital de un cliente (Twitter, Facebook). Todo esto especificado en el cuarto capítulo detallado en el Sprint 7, anexo L.

6. Desarrollar la interfaz de programación de aplicaciones de Clientes.

Se obtuvo la interfaz de programación de aplicaciones (API) que permite crear, consultar, editar y eliminar clientes y parámetros adicionales, con todas las funcionalidades necesarias para almacenar la información básica de los canales digitales especificados (Twitter, Facebook, Chat) con su información adicional (Teléfonos). Todo esto especificado en el cuarto capítulo detallado en el Sprint 4, anexo M.

7. Desarrollar la interfaz de programación de aplicaciones de Bitácora.

Como resultado de crear una bitácora se pueden consultar las informaciones de las acciones importantes y así se tiene una perspectiva más clara de lo que se ha estado realizando, ayudando a mantener un respaldo de la información que puede considerarse un gran valor en la aplicación para posibles estudios y análisis de acciones que realizan los usuarios especificado en el cuarto capítulo detallado en el Sprint 8, anexo N.

8. Desarrollar la interfaz de programación de aplicaciones de Perfil.

Se alcanzó el desarrollo de la interfaz de programación de aplicaciones (API) encargada de consultar, editar la información del usuario con la opción de ser modificado

o personalizado por el mismo, excluyendo el correo. Además de permitir al usuario modificar su contraseña en la aplicación. Todo esto especificado en el cuarto capítulo en el Sprint 3, anexo P.

9. Desarrollar la interfaz de programación de aplicaciones de Plantillas.

Se desarrolló la interfaz de programación (API) con el fin de crear, consultar, editar y eliminar las plantillas de los usuarios, con la opcionalidad de especificar si será para el equipo o única y exclusivamente para el usuario todo esto descrito en el cuarto capítulo en el Sprint 6, anexo J.

10. Desarrollar la interfaz de programación de aplicaciones de Permisología.

Al desarrollar este módulo se obtuvo una interfaz de programación de aplicaciones (API) con el fin de crear, consultar, editar y eliminar las permisologías y roles que corresponden al núcleo principal de este módulo. Cada plan comprende de una permisología y los equipos comprenden de roles que contienen una permisología especificando los canales digitales asociados al mismo, dándole confianza y seguridad a la aplicación, todo esto especificado en el cuarto capítulo en el Sprint 2, anexo B.

11. Desarrollar la interfaz de programación de aplicaciones de Mensajes Automáticos.

Se obtuvo la interfaz de programación de aplicaciones (API) que permite crear, consultar, editar y eliminar mensajes automáticos y horario laboral (exceptuando el eliminar para el caso de horario laboral), con todas las funcionalidades necesarias para llevar a cabo el envío mensajes automáticos a los tickets generados fuera de horario laboral de los canales digitales especificados (Twitter, Facebook). Todo esto descrito en el cuarto capítulo en el Sprint 6, anexo K.

12. Desarrollar la interfaz de programación de aplicaciones de Equipos.

Se desarrolló la interfaz de programación de aplicaciones (API) que permite crear, consultar, editar y eliminar equipos en la aplicación, además de incorporar la creación al

momento de registrarse, con un nombre del equipo único y está especificado en el cuarto capítulo en el Sprint 2, anexo C.

13. Desarrollar la interfaz de programación de aplicaciones de Carrito de Compra.

Se alcanzó el desarrollo de la interfaz de programación de aplicaciones (API) que permite crear, consultar, editar y eliminar los distintos tipos de órdenes y pagos, además de la realización de un comando para activación de órdenes pagadas, englobando así todas las funcionalidades necesarias para completar el ciclo de compra de la aplicación, todo especificado en el cuarto capítulo en el Sprint 5, anexo G.

14. Desarrollar la interfaz de programación de aplicaciones de Prueba Gratuita.

Se desarrolló la interfaz de programación de aplicaciones (API), para permitir a los clientes con planes gratuitos, gozar de la experiencia (15 días) y servicios que ofrecen los planes pagos en la aplicación, para tener una mejor perspectiva de lo que se le ofrece y si cumple con los intereses del mismo, todo especificado en el cuarto capítulo en el Sprint 5, anexo I.

Aporte funcional

15. Rediseñar el proceso de venta de los servicios de la aplicación.

Con el nuevo proceso de venta, se espera que los clientes seguidamente después de generar una orden con el plan de su preferencia y al realizar el pago tengan la potestad de elegir como lo desean (transferencia, pasarela de pago) y no tengan la necesidad de pagar únicamente a través de transferencia y a su vez tener que contactar al departamento de ventas (llamadas o correos) cuándo se efectuó el pago. Todo especificado en el cuarto capítulo de en el Sprint 1.

16. Sistema de escalamiento de casos.

Al implementar este sistema, genera un impacto en el proceso de escalamiento ya que evita, caer en la complejidad del ciclo de escalamiento que ejecutaban dichas empresas, asegurando la solvencia de los inconvenientes presentados por los clientes

de la mejor manera posible, todo esto se encuentra especificado en el cuarto capítulo en el Sprint 7, anexo M.

Aporte tecnológico

17. Desarrollar un servicio web REST API para la venta de los servicios de la aplicación.

Se desarrolló un servicio web REST API para las ventas de los servicios de la aplicación que consiste en la generación de órdenes y realización de pagos a través de las pasarelas especificadas en el cuarto capítulo Sprint 5. Además de los procesos necesarios para la activación, todo esto especificado en el cuarto capítulo en el Sprint 5.

18. Evaluar los distintos *framework* de desarrollo (a nivel de Back-End) disponibles para la creación del servicio web REST API de la aplicación y a su vez reestructurar base de datos existentes para que pueda ser utilizada por el *framework* a implementar.

Laravel se seleccionó como *framework* de desarrollo en PHP con el fin de desarrollar el API evaluando los requerimientos y requerimientos establecidos en las tablas 4 y 5 especificada en el cuarto capítulo en el Sprint 0. A su vez se realizó la reestructuración necesaria en la base de datos para así poder implementar el *framework* y que éste pudiera utilizarla sin ningún inconveniente.

CAPÍTULO VI – CONCLUSIONES Y RECOMENDACIONES

Conclusiones

Finalizado el desarrollo del Trabajo Instrumental de Grado, a través de los resultados obtenidos se pudo concluir lo siguiente:

- Con la creación de este nuevo conjunto de funciones y procedimientos *(API)* se facilitará el acceso a sus servicios y proporcionará más alcance a sus clientes, ya que se puede compartir y distribuir con mucha más facilidad, garantizando la entrega de información y una excelente experiencia de usuario.

- Gracias al *Framework* Laravel le permite a la aplicación mantener el uso de una sintaxis refinada, expresiva y ordenada, facilitando el rendimiento de sus operaciones y teniendo el servicio actualizado en cuanto a la aplicación de nuevas tecnologías y el desarrollo de sistemas robustos.

- Los servicios web de Amazon garantizan (*Amazon Web Services*) suministra un mejor manejo, control y desarrollo de la aplicación ya que proporciona: estabilidad, seguridad y escalabilidad que son elementos importantes para la efectividad de una aplicación.

- Se logró la comunicación, agilidad y motivación necesaria para el equipo para sus labores de desarrollo, todo gracias a la utilización del marco de trabajo *Scrum*, debido a que se pudieron efectuar importantes modificaciones puestas en consideración en el transcurso del desarrollo de la aplicación.

- Con el desarrollo del módulo de carrito de compras, los clientes pueden efectuar órdenes y pagos dentro de la aplicación e incluye gran variedad de pasarelas de pago *(InstaPago, Mercado Pago, Paypal y* Stripe) y tiene la posibilidad de declarar pagos efectuados por transferencia.

- Con la implementación de la librería Pusher (WebSocket), se puede trabajar la información en tiempo real, permitiendo al usuario obtener un monitoreo real de lo que está ocurriendo en sus canales digitales (Twitter, Facebook, Chat) de la manera más rápida posible.

- La aplicación goza de servicios Webhook en el canal digital Facebook lo que permite recibir notificaciones en tiempo real (comentarios, publicaciones de visitantes, mensaje directo).

- Se completó el desarrollo de la nueva versión de la plataforma "TIKKET V1", al finalizar el trabajo instrumental de grado y por consiguiente estará en proceso de mercadeo.

Recomendaciones

- Se recomienda la evaluación la inclusión de personal más experimentado en el área de desarrollo de proyectos, ya que no se encuentra ningún experto en el área.

- Se recomienda la utilización de procedimientos (*procedures*) en base de datos para agilizar la obtención de información correspondiente a las funcionalidades.

- Se recomienda que al momento de almacenar los tickets de la aplicación específicamente en Facebook, de tipo comentario y comentario de comentario, se especifique qué publicación en el canal digital los contiene.

- Se recomienda incluir Youtube como canal digital adicional, puesto que es un canal digital muy demandado en ésta época y lograrán atraer más clientes a la aplicación.

Se recomienda que la gestión de equipo sea escribiendo el equipo directamente en el URL (equipo.tikket.net), para que se pueda cargar automáticamente, sin tener la necesidad de cambiar el equipo y evitar pasos extras al usuario.

- Se recomienda filtrar el contenido al momento de realizar peticiones, ya que se consulta información que no es necesaria.

- Se recomienda establecer 2 funcionalidades más para el envío de mensaje directo de Facebook *(*Webhook*)* con el motivo de manejar el estatus de un mensaje que ha sido entregado o leído.

- Se recomienda establecer la inclusión de robots (*bots*) de mensajería para solicitudes básicas y que dependiendo de los mensajes que lleguen, analizar la información y que pueda responder automáticamente.

- Se recomienda incorporar el Webhook de mensaje directo *(Messenger)* de Facebook, para que tanto el cliente como el operador sepa cuando un mensaje es enviado y recibido por los mismos.

REFERENCIAS BIBLIOGRÁFICAS

[1] Immagic (2017). Application programming interface. Disponible en: https://www.immagic.com/eLibrary/ARCHIVES/GENERAL/WIKIPEDI/W120623A.pdf [2017, Julio]

[2] Councill Bill, George T. Heineman (2010). Definition of a Software Component and Its Elements. Disponible en: http://heim.ifi.uio.no/~frank/inf5040/CBSE/Component-Based_Software_Engineering_-_ch1.pdf [2017, Agosto]

[3] Johnson Glenn (2015). Application Programming Interface (API) Definition and Classification by Types. Disponible en: https://it.toolbox.com/blogs/glennjohnson/application-programming-interface-api-definition-and-classification-by-types-040214 [Junio, 2017]

[4] Rouse Margaret (2014). REST (REpresentational State Transfer). Disponible en: http://searchmicroservices.techtarget.com/definition/REST-representational-state-transfer [Junio, 2017]

[5] Peñafiel Mayra. Aplicaciones Web. Disponible en: https://es.scribd.com/doc/136052164/APLICACIONES-WEB-pdf [2017, Agosto]

[6] CodeAcademy (2016). What is Rest? Disponible en: https://www.codecademy.com/articles/what-is-rest [2017, Agosto]

[7] Steward Lauren (2018). Front End vs Back End Development. Disponible en: https://www.coursereport.com/blog/front-end-development-vs-back-end-development-where-to-start [2018, Marzo]

[8] PHP (2015). What can PhP do? Disponible en: http://php.net/manual/en/intro-whatcando.php [2017, Junio]

[9] UpWork (2016). Understanding the frameworks software. Disponible en: https://www.upwork.com/hiring/development/understanding-software-frameworks/ [2017, Julio]

[10] Laravel. The PHP Framework for Web Artisans. Disponible en: https://laravel.com [2017, Julio]

[11] Investopedia (2018). Social Networking. Disponible en: https://www.investopedia.com/terms/s/social-networking.asp [2018, Enero]

[12] Rouse Margaret (2014). Twitter. Disponible en: http://whatis.techtarget.com/definition/Twitter [2017, Agosto]

[13] Rouse Margaret (2014). Facebook. Disponible en: http://whatis.techtarget.com/definition/Facebook [2017, Agosto]

[14] Balaji Sadhana (2017). What are Webhooks and why you can't afford to ignore them? Disponible en: https://www.chargebee.com/blog/what-are-webhooks-explained/ [2017, Junio]

[15] Mercadeo para emprendedores (2015). Chat en línea: Una poderosa herramienta de Marketing. Disponible en: http://mercadeoparaemprendedores.com/blog/chat-en-linea-una-poderosa-herramienta-de-marketing/ [2017, Marzo]

[16] Auth0 (2013). Access Token. Disponible en: https://auth0.com/docs/tokens/access-token [2017, Junio]

[17] Pusher (2018). What are the web sockets? Disponible en: https://pusher.com/websockets [2018, Enero]

[18] What is pusher? Disponible en: https://pusher-community.github.io/real-time-laravel/introduction/what-is-pusher.html [2017, Agosto]

[19] Radigan Dan. Scrum. Disponible en: https://www.atlassian.com/agile/scrum [2017, Agosto]

[20] Tushar (2017). What's the difference? Agile vs Scrum vs waterfall vs Kanban. Disponible en: https://www.smartsheet.com/agile-vs-scrum-vs-waterfall-vs-kanban [2017, Julio]

[21] Garbade Michael J. (2016). How to choose a PHP framework. Disponible en: https://opensource.com/business/16/6/which-php-framework-right-you [2017, Agosto]

[22] The right software (2017). PHP framework comparison 2018: CodeIgniter vs laravel vs yii vs cakePHP. Disponible en: https://therightsw.com/codeigniter-vs-laravel-vs-yii-vs-cakephp/ [2017, Agosto]

[23] Malthora Mantra (2018). Codekniter vs CakePHP vs Yii vs Laravel with one to use. Disponible en: https://www.valuecoders.com/blog/technology-and-apps/codeigniter-vs-cakephp-vs-yii-vs-laravel/# [2018, Enero]

[24] Thuau Jonathan (2017). Twitter. Disponible en: https://github.com/thujohn/twitter [2017, Agosto]

[25] Sammy Kaye (2017). Laravel Facebook SDK. Disponible en: https://github.com/SammyK/LaravelFacebookSdk [2017, Agosto]

[26] Graviano Santiago (2017). Laravel Facade para MercadoPago v0.5.2. Disponible en: https://github.com/santigraviano/laravel-mercadopago [2017, Agosto]

[27] Developers (2018). MercadoPago API. Disponible en: https://www.mercadopago.com.ve/developers/en/api-docs/ [2018, Enero]

[28] Instapago. Integration Guide. Disponible en: https://instapago.com/wp-content/uploads/Integration-Guide-Instapago-API-1.6.pdf [2017, Agosto]

[29] Shoumik Shariar Fahim (2017). Laravel5 - PaypalIPN. Disponible en: https://github.com/sh0umik/Laravel5-PaypalIPN [2017, Septiembre]

[30] Paypal Developer (2017). Webhooks. Disponible en: https://developer.paypal.com/docs/integration/direct/webhooks/ [2017, Agosto]

[31] Stripe Docs (2017). Using Webhooks. Disponible en: https://stripe.com/docs/webhooks [2017, Agosto]

[32] Fairbanks Lory (2018). Chat software. Disponible en: https://www.business.com/categories/live-chat-software/ [2018, March]

[33] Guzzle. Guzzle Documentation. Disponible en: http://docs.guzzlephp.org/en/stable/ [2017, Agosto]

[34] Application Management. Twitter Apps. Disponible en: https://apps.twitter.com [2017, Agosto]

[35] Facebook for Developers. Developers. Disponible en: https://developers.facebook.com [2017, Agosto]

[36] Cartalyst. Stripe Laravel. Disponible en: https://github.com/cartalyst/stripe-laravel [2017, Agosto]

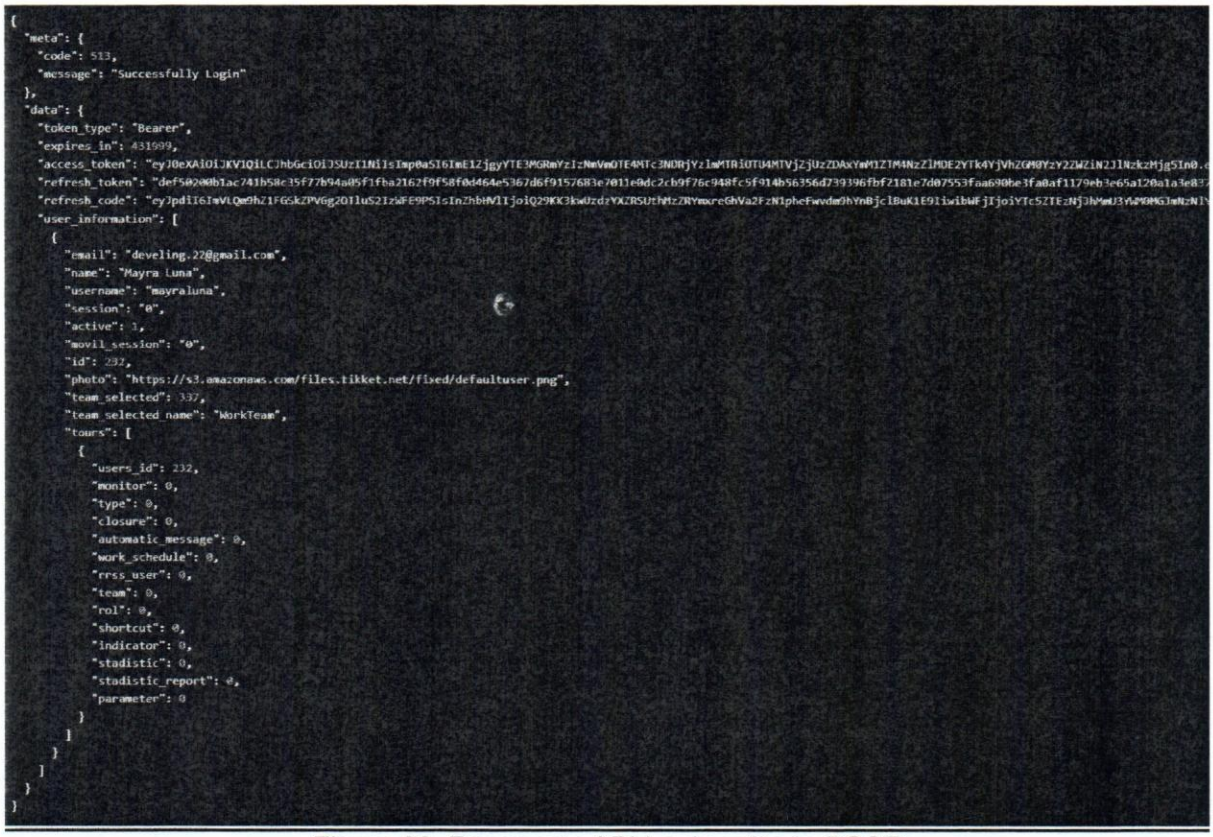
ANEXOS

Apendice A: Autenticación

A.1 Login

Figura 19. API Login método POST

Fuente: Elaboración Propia



```
{
  "meta": {
    "code": 513,
    "message": "Successfully Login"
  },
  "data": {
    "token_type": "Bearer",
    "expires_in": 431999,
    "access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6ImE1ZjgyYTE3MGRmYzIzNmVmOTE4MTc3NDRjYzlmMTRiOTU4MTVjZjUzZDAxYmM1ZTM4NzZlMDE2YTk4YjVhZGM0Yzy22WZiN2JlNzkzMjg5In0.
    "refresh_token": "def50200b1ac741b58c35f77b94a05f1fba2162f9f58f0d464e5367d6f915768e7011e0dc2cb9f76c948fc5f914b56356d739396fbf2181e7d07553faa690be3fa0af1179eb3e65a120a1a3e83
    "refresh_code": "eyJpdlI6ImVLQm9hZ1FGSkZPVGg2OIluS21zWFE9PSIsInZhbHVlIjoiQ29KK3kwUzdzYXZRSUthMzZRYmxreGhVa2FzN1pheFwvdm9hYnBjclBuK1E9IiwibWFjIjoiYTc5ZTEzNjJhMmU3Yw49MGlmNzN1
    "user_information": [
      {
        "email": "develing.22@gmail.com",
        "name": "Mayra Luna",
        "username": "mayraluna",
        "session": "0",
        "active": 1,
        "movil_session": "0",
        "id": 232,
        "photo": "https://s3.amazonaws.com/files.tikket.net/fixed/defaultuser.png",
        "team_selected": 337,
        "team_selected_name": "WorkTeam",
        "tours": [
          {
            "users_id": 232,
            "monitor": 0,
            "type": 0,
            "closure": 0,
            "automatic_message": 0,
            "work_schedule": 0,
            "rrss_user": 0,
            "team": 0,
            "rol": 0,
            "shortcut": 0,
            "indicator": 0,
            "stadistic": 0,
            "stadistic_report": 0,
            "parameter": 0
          }
        ]
      }
    ]
  }
}
```


Figura 20. Respuesta API Login método POST

Fuente: Elaboración Propia

A.2 Token de refrescamiento

POST Refresh Token

http://localhost/api-tikket/public/api/v1/users/refresh_token

Refresh token para web

HEADERS

Accept	application/json

BODY

access_token	eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6ImE1ZjgyYTE3MGRmYzIzNmVmOTE4MTc3NDRjYzImMTRlOTU4MTVjZjU zZDAxYmM1ZTM4NzZlMDE2YTk4YjVhZGM0YzY2ZWZlN2JlNzkzMjg5In0.eyJhdWQiOiIxIiwianRpIjoiYTVmODJhMTcwZGZjM jM2ZWY5MTgxNzc0NGNjOWYxNGI5NTgxNWNmNTNkMDFiYzVlMzg3NmUwMTZhOThiNWFkYzRjNjZlZmI3YmU3OTMyO DkiLCJpYXQiOjE1MTk2NjkwNTAsIm5iZiI6MTUxOTY2OTA1MCwiZXhwIjoxNTIwMTAxMDQ5LCJzdWIiOiIyMzIiLCJzY29wZXM iOIsiKiJdfQ.K2D0AmhYB08HhxSItwxsNUOW4eHLEtyKL1PWgiXg9nqo9QM08n497Eg1Ir1UShDe9cNZQU_oWUe72Iu2FxGo oyuOgH7VGTZCNMFX9ALPGBWrTotTBUggduqci94STcg_m6yUZ8IdC-bHAziYE9tkM-VNme2wQPOZr3zqAltq3s_C1RW- gm2wW3JT6LX0EMUIoEPuBXiRtrneq2EQoL_Trpa4NujIEnR3OcZ4H65OQDykNBjXtzxuspz8SX2dMb1O9H7If6QYovUHQY- t_0neoqAZMkB7w3MBbv4nHZJZqEC9Irl6MILwOAFx06FNZCq6RIuFt4dxrTTuLtPC6VJgpce4jRm7DZsjgE1eWpoRgi5YhptyG EkN10s9WhnvyuCm2V8- SgE7F247ktEkpi3i6Sux7RE90oPmZq3zCkLzT54NKN48AdmQf1OmhA463EKUw5XkR5GhIkEfrYwFIPUfNR0oOBMvcZUILIXz te1GyBhcxL9e__mEpZf2n-_Rq6xLOW2KWbEd-wjc1Umf2pj0pca9w10QUI63cFVgH4WRmNqO_Y-FI3- rE24kugsmbXRfXnXkII2IGgGxoj6Iw9RuwDcj81W365xxIf9KSDA5euxatr7NystAoSOQ3WgkZPMNodoUgZkHkTIWdV2r6DIV 5euWoqNmVWmMMswR2bGp_5o Access token del usuario
refresh_token	def50200b1ac741b58c35f77b94a05f1fba2162f9f58f0d464e5367d6f9157683e7011e0dc2cb9f76c948fc5f914b56356d73 9396fbf2181e7d07553faa690be3fa0af1179eb3e65a120a1a3e837af1b5edfd08bcc57e63f0934cf9c9727f75a6723f72f94f 696c3d6e64ba65e50861b9496669a4ee955ea5b342071a5a35b9159155cc1038d7247e8abe65bd074523a64f46b3e3812 c9ad52314746415f69939ec32909e497a61425aea6fbf6f07ce6cba40ac63f9a552845b16512027d7dfe6d41432b948459444 47eb3e7a7f597b7499f334407b90230edc1ac898c9a75f41c5702ec2539d04d52df4201f197cda5528978c077333100d6ab 10f68f09dd29c5105879256130d5c3fbf76b7b008d7605ab079ae941440cd61842e49a9c0752f1e193be55e9c29ebfa8f099 229f9b652d0f2582d9c815f037c24ac31fd6cbde6a1bc13a2319f9af10328d696d7464546c95f95b2907e147555e6646edff4 a5e8eaae60f1aaf6 Refresh token del usuario

Figura 21. API Refresh token método POST

Fuente: Elaboración Propia

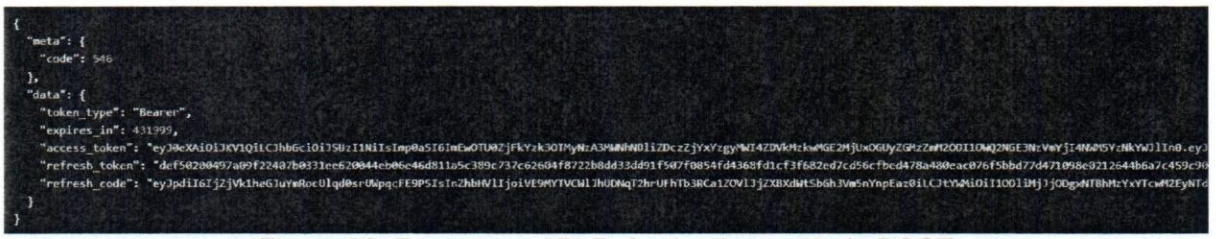


Figura 22. Respuesta API Refresh token método POST

Fuente: Elaboración Propia

A.3 Logout

POST Logout

http://localhost/api-tikket/public/api/v1/users/logout

Logout web

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4N jE3ZmVhMmNiMTUwNTg2YzE5MmFlMmY1NzIwZjQ0ZmQ0YjdiZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhd WQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTlyYjg2MTdmZWEyY2IxNTA1ODZjMTkyYWUyZjU 3MjBmNDRmZDRiN2JmOTcyZGQ3MTVmYjQwOTFlYTYiLCJpYXQiOjE1MjEyMzYzODAsIm5iZiI6MTUyMTIzNj M4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3dSjGh46SBjZInZ8klQY429S FN25HrYRohMJD3WxU6-1WMZP-DUChdGjjzGLF3hgyTL4BpW8ULdWgeAJK6qXiLzDSj3rffyjYmudw6UIelcuv s1KQNU5-6HGguPrCyt7y5luhP1jnGxKFfc_R4SbYh1wfBoXPP5wjMkfKeb9gRBRJoYscoaQIhkVQ5VG7ndckne 5NkGxogyz1ElVavjZb2dvGuAFPUHt_WMRLTSwVI6oQHflorkNktJJuBKFGLcS4QVWhEeuhaa-Gkdobv0gBPOe 82JaX9IZ8nclAkWMn7daQ1TZV45Te1k2WSRrDXIpg4XjoQ1DMolf655z015wpHwYYZG69lHln_HtWuUkF6aV Kb6gLPRg_Y2-AJCVRexjoJ0vE43Q0LY1M0hjVL8RuCZwPViaADlbA-5cfdLj9AOVjmDqpNzO7ZMxJEMM_8VR09 NOl8G2QukjXDe_xZeVlwfoLLa8JowITtgni0chsW4_DBnNpwT4WMy3QG2FjGaSGTvEif-dT63fb3aAfuvO-7Itm ORw_zH69Zwusxfsx79OZOCycjF5LJjgGpAvy3_91ghl-y5aGtRPr4CeRPjk7DmcFln2gV_x5HWi3O6LNtvlhc4tAt XmBa_2svHtTcWI2wTqhBbDCfMeXodaHl_OeqPV4Oy1Vl7qIEq7tQ
Accept	application/json

BODY

logout_all_devices	0 Cerrar sesión de todos los dispositivos

Figura 23. API Logout método POST

Fuente: Elaboración Propia

```
{
  "meta": {
    "code": 541
  },
  "data": {
    "message": "You have successfully logged out!"
  }
}
```


Figura 24. Respuesta API Logout método POST

Fuente: Elaboración Propia

Apéndice B: Permisología

B.1 Tipología de permisos

GET Index TypePermissions

http://localhost/api-tikket/public/api/v1/typepermissions

Consultar tipo de permisos

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6ImU3NDc3NGI5NjNmYTgxYWE2ZGI4ZjkwYWI3Y2E yMWRmODhiNmRlYWRmZGE5NmU0ZDYxYTI5MWY1NWY4YmRlMjNmZGQ1NGUzNjMxODFjNGRmIn0.eyJ hdWQiOiIxIiwianRpIjoiZTc0Nzc0Yjk2M2ZhODFhYTZkYjhmOTBhYjdjYTIxZGY4OGI2ZGJhZGZkYTk2ZTRkNjFh MjkxZjU1ZjhiZGUyM2ZkZDU0ZTM2MzE4MWM0ZGYiLCJpYXQiOjE1MjE0MTMzOTAsIm5iZiI6MTUyMTQxMz M5MCwiZXhwIjoxNTIxODQ1MzkwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.IFOA2qh9PjX6losPuyhKXiGh7os sijkyK85x2oUDnAzdwXBLBDiihVO9GCDdcSbgraArHhRumav6Nh2RiistOldV1V8PSuWKE9KmydY5ATl3mvE N_zLakICkUcC_TOqv85QdR3ZHenB5fJvLpAcol_Pbtg130CIYiS1QcKOU-QEFMNhJWPMWvZZzi0sg6ZCJ4MF60 oYH4ELFoH4hwpYJuYj1i_slB925UOCtIGVQnCrUPvsVR0IhDZiFau8fBAAlydvMkwTCpTCSMt74_LBU2ixV7Qp 6Ph8pUHHc-JNYolQXPP34mR42vnnGh3fvMWZrscYUqdHvbZ9vk1Jk4CIL3LaZ8c9fLVp2pwqFQNBt5BJ808ag RPkJoPNLAQHNwByT2Au-Mulgu14wcPE2MFyOcbOzEMKu7PGug-m-tvIn3kcoGPD30RxqIHpcFamqCbVc1P SEhLrY-JeT6jZLf3FfcSD0LxDZBCDl8xR-8dpUeE2ezUjZC0pbXin7MU-pQjIXw7rI76ENUrdkMq2RSFQCje-nnyP s8OgLGFwPWhPfQ3okpdwxc9cr4ZajQD1tnyCl43LHRmwDDTFfDdco47vS6zzmhemp8rukxqeJeSMN_-C0_O OlW7XQIIqhsBLQIUEsINs05uBQOT9CXneqBOSnWVGjyo8SPJkjnJA_WtkmqHQ
Accept	application/json

Figura 25. API Tipos de permisos método GET

Fuente: Elaboración Propia

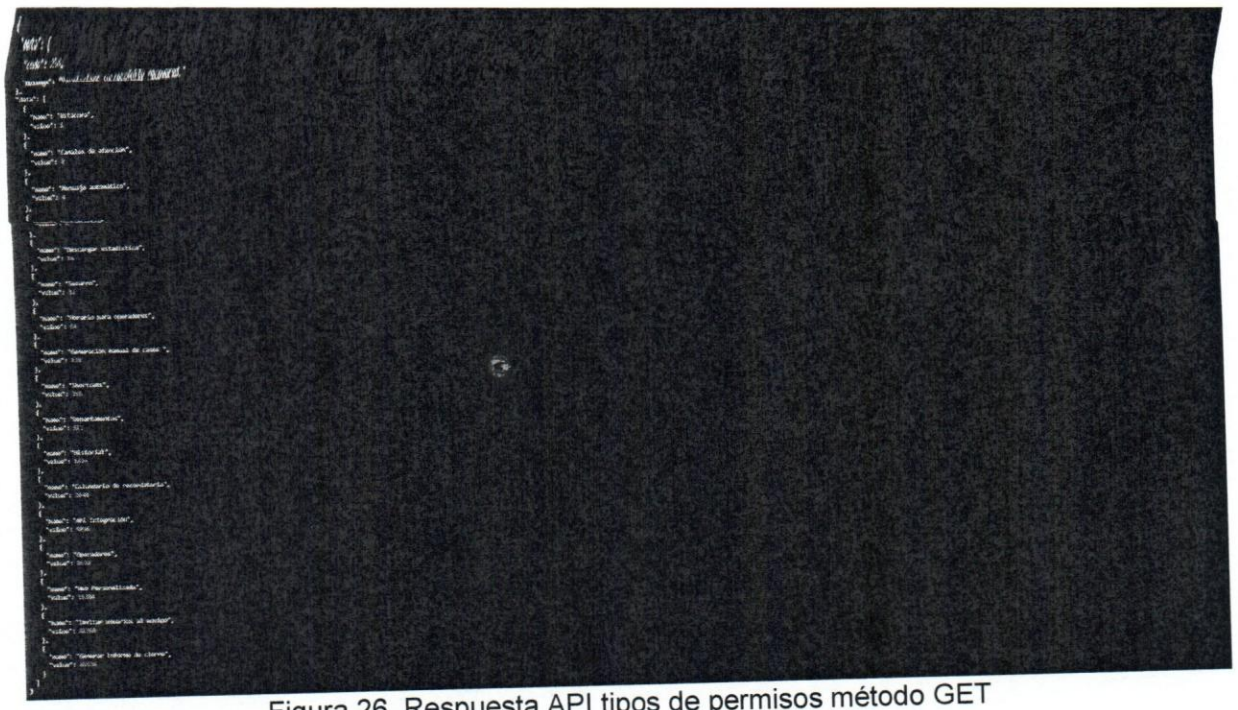


Figura 26. Respuesta API tipos de permisos método GET

Fuente: Elaboración Propia

B.2 Verificar permisos

GET CheckPermission

http://localhost/api-tikket/public/api/v1/checkpermission?team=262&permission=1&user=78

Verificar permiso

HEADERS

Accept	application/json
Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNiMTUwNTg2YzE5MmFlMmY1NzlwZjQ0ZmQ0YjdlZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhdWQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTIyYjg2MTdmZWEyY2IxNTA1ODZjMTkyYWUyZjU3MjBmNDRmZDRiN2JmOTcyZGQ3MTVmYjQwOTFlYTYiLCJpYXQiOjE1MjEyMzYzODAsIm5iZiI6MTUyMTIzNjM4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3dSjGh46SBjZInZ8klQY429SFN25HrYRohMjD3WxU6-1WMZP-DUChdGjjzGLF3hgyTL4BpW8ULdWgeAjK6qXiLzDSj3rffyjYmudw6UIeIcuvs1KQNU5-6HGguPrCyt7y5IuhP1JnGxKFfc_R4SbYh1wfBoXPP5wjMkfKeb9gRBRJoYscoaQIhkVQ5VG7ndckne5NkGxogyz1ElVavjZb2dvGuAFPUHt_WMRLTSwVI6oQHflorkNktJjuBKFGLcS4QVWhEeuhaa-Gkdobv0gBPOe82JaX9IZ8nclAkWMn7daQ1TZV45Te1k2WSRrDXIpg4XjoQ1DMolf655z015wpHwYYZG69iHIn_HtWuUkF6aVKb6gLPRg_Y2-AjCVRexjoJ0vE43Q0LY1M0hjVL8RuCZwPViaADlbA-5cfdLj9AOVjmDqpNzO7ZMxjEMM_8VR09NOI8G2QukjXDe_xZeVIwfoLLa8JowITtgni0chsW4_DBnNpwT4WMy3QG2FjGaSGTvEif-dT63fb3aAfuvO-7ItmORw_zH69Zwusxfsx79OZOCycjF5LjjgGpAvy3_91ghl-y5aGtRPr4CeRPjk7DmcFln2gV_x5HWi3O6LNtvIhc4tAtXmBa_2svHtTcWl2wTqhBbDCfMeXodaHI_OeqPV4Oy1Vl7qIEq7tQ

PARAMS

team	262
permission	1
user	78

Figura 27. API Verificar permisos método GET

Fuente: Elaboración Propia


```
{
  "data": {
    "isFather": 1,
    "info": "Access granted"
  },
  "meta": {
    "code": 200
  }
}
```


Figura 28. Respuesta verificar permisos método GET

Fuente: Elaboración Propia

Apéndice C: Equipos

C.1 Consultar equipos

GET List of teams

http://localhost/api-tikket/public/api/v1/teams

Consultar equipos

HEADERS

Accept	application/json
Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNiMTUwNTg2YzE5MmFlMmY1NzIwZjQ0ZmQ0YjdlZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhdWQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTIyYjg2MTdmZWEyY2IxNTA1ODZjMTkyYWUyZjU3MjBmNDRmZDRlN2JmOTcyZGQ3MTVmYjQwOTFlYTYiLCJpYXQiOjE1MjEyMzYzODAsImN5ZiI6MTUyMTIzNjM4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3d5jGh46SBjZInZ8kIQY429SFN25HrYRohMjD3WxU6-1WMZP-DUChdGjJzGLF3hgyTL4BpW8ULdWgeAjK6qXILzDSj3rffyjYmudw6UIeIcuvs1KQNU5-8HGguPrCyt7y5IuhP1JnGxKFfc_R45bYn1wfBoXPP5wjMkfKeb9gRBRJoYscoaQInkVQ5VG7ndckne5NkGxogyc1ElVavjZbZdvGuAFPUHt_WMRLTSwVI6oQHflorkNktjJuBKFGLcS4QVWrEeuhaa-Gkdobv0gBPOe82JaX9IZ8nclAkWMn7daQ1TZV45Te1k2WSRrDXlog4XjoQ1DMolf655z015wpHwYYZG69lHIn_HtWuUkF6aVKb6gLPRg_Y2-AJCVRexjoJ0vE43Q0LY1M0njVL8RuCZwPVIaADibA-5cfdLj9AOVjmDqpNzO7ZMxJEMM_8VR09NOi8G2QukjXDe_xZeVIwfoLLa8JcwITtgnI0chsW4_DBnNpwT4WMy3QG2FjGaSGTvEif-dT63fb3aAfuvO-7ItmORw_zH69Zwusxfsx79OZOCycjF5LjjgGpAvy3_91ghI-y3aGtRPr4CeRPjk7DmcPln2gV_x5HWi3O6LNtvihc4tAtXmBa_2svHtTcWl2wTqhBoDCfMeXodaHI_OeqPV4Oy1VI7qIEq7tQ

Figura 29. API Consultar equipos método GET

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 546,
    "message": "Retrieving information of teams was successfully"
  },
  "data": [
    {
      "id": 262,
      "name": "Test",
      "domain": "test",
      "ownership": 1,
      "accepted_count": 0,
      "pending_count": 1
    },
    {
      "id": 267,
      "name": "Paloma",
      "domain": "paloma",
      "ownership": 1,
      "accepted_count": 0,
      "pending_count": 0
    }
  ]
}
```


Figura 30. API Respuesta consultar equipos método GET

Fuente: Elaboración Propia

C.2 Agregar equipo

POST Store team

http://localhost/api-tikket/public/api/v1/teams

Agregar equipo

HEADERS

Authorization Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNlMTUwNTg2YzE5MmFlMmY1NzIwZjQ0ZmQ0YjdlZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhdWQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTlyYjg2MTdmZWEyY2IxNTA1ODZjMTkyYWUyZjU3MjBmNDRmZDRlN2JmOTcyZGQ3MTVmYjQwOTFlYTYiLCJpYXQiOjE1MjEyMzYzODAsIm5iZiI6MTUyMTIzNjM4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3dSjGh46SBjZInZ8kIQY429SFN25HrYRohMjD3WxU6-1WMZP-DUChdGjjzGLF3hgyTL4BpW8ULdWgeAJK6qXiLzDSj3rffyJYmudw6UIeIcuvs1KQNU5-6HGguPrCyt7y5IuhP1JnGxKFfc_R45bYh1wfBoXPP5wjMkfKeb9gRBRJoYscoaQIhKVQ5VG7ndckne5NkGxogyz1EIVavjZb2dvGuAFPUHt_WMRLTSwVI6oQHfIorkNktJJuBKFGLcS4QVWhEeuhaa-Gkdobv0gBPOe82JaX9IZ8nclAkWMn7daQ1TZV45Te1k2WSRrDXIpg4XjoQ1DMoif655z015wpHwYYZG69IHIn_HtWuUkF6aVKb6gLPRg_Y2-AJCVRexjoJ0vE43Q0LY1M0hjVL8RuCZwPVIaADIbA-5cfdLj9AOVjmDqpNzO7ZMxJEMM_8VR09NOI8G2QukjXDe_xZeVIwfoLLa8JowITtgnI0chsW4_DBnNpwT4WMy3QG2FjGaSGTvEif-dT63fb3aAfuvO-7ItmORw_zH69Zwusxfsx79OZOCycjF5LJjgGpAvy3_91ghI-y5aGtRPr4CeRPjk7DmcFIn2gV_x5HWI3O6LNtvIhc4tAtXmBa_2svHtTcWI2wTqhBbDCfMeXodaHI_OeqPV4Oy1VI7qIEq7tQ

Accept application/json

BODY

name Equipo de prueba
Nombre del equipo

Figura 31. API Agregar equipo método POST

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 548,
    "message": "Team created successfully"
  },
  "data": {
    "name": "Equipo de prueba",
    "domain": "equipodeprueba",
    "id": 365,
    "ownership": 1
  }
}
```


Figura 32. Respuesta API agregar equipo método POST

Fuente: Elaboración Propia

3 Editar equipo

PUT Update team

http://localhost/api-tikket/public/api/v1/teams/365

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNlMTUwNTg2YzE5MmFlMmY1NzIwZjQ0ZmQ0YjdiZjk3MmRkNzE1Zml0MDkxZWE2In0.eyJhdWQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTIyYjg2MTdmZWEyY2IxNTA1ODZjMTkyYWUyZjU3MjBmNDRmZDRlN2JmOTcyZGQ3MTVmYjQwOTFlYTYiLCJpYXQiOjE1MjEyMzYzODAsImN5ZiI6MTUyMTIzNjM4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3dSjGh46SBjZlnZ8klQY429SFN25HrYRohMJD3WxU6-1WMZP-DUChdGjJzGLF3hgyTL4BpW8ULdWgeAjK6qXiLzDSj3rffyjYmudw6UIeIcuvs1KQNU5-6HGguPrCyt7y5IuhP1JnGxKPfc_R4SbYh1wfBdXPP5wjMkfKeb9gRBRJoYscoaQInkVQ5VG7ndckne5NkGxdgyz1EiVavjZo2dvGuAFPUHt_WMRLTSwVI6oQHflorkNktjJuBKFGLcS4QWWhEeuhaa-Gkdobv0gBPOe82JaX9IZ8ndIAkWMn7daQ1TZV45Te1kZW5RrDXlpg4XjoQ1DMoIf655z015wpHwYYZG69iHln_HtWuUkF6aVKb6gLPRg_Y2-AJCVRexjoJ0vE43Q0LY1M0njVL8RuCZwPViaADIbA-5cfdLj9AOVjmDqpNzO7ZMxjEMM_8VR09NOl8G2QukjXDe_xZeVIwfoLLa8JowITtgnI0chsW4_DBnNpwT4WMy3QG2FjGa5GTvEif-dT63fb3eAfuvO-7IrmORw_zH69Zwusxfsx79OZOCycjF5LjgGpAvy3_31gnI-y5aGtRPr4CeRPJk7DmcFln2gV_x5HWi3O6LNtvIhc4tAtXmBa_2svHtTcWI2wTqhBbDCfMeXodaHI_OeqPV4Oy1VI7qIEq7tQ
Accept	application/json
Content-Type	application/x-www-form-urlencoded

BODY

name	Equipo de prueba 2 Nombre del equipo

Figura 33. API Modificar equipos método PUT

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 555,
    "message": "Team updated"
  },
  "data": {
    "id": 365,
    "name": "Equipo de prueba 2",
    "domain": "equipodeprueba2",
    "photo": "https://s3.amazonaws.com/files.tikket.net/fixed/defaultuser.png",
    "ownership": 1
  }
}
```


Figura 34. Respuesta API modificar equipos método PUT

Fuente: Elaboración Propia

C.4 Eliminar equipo

DELETE Delete team

http://localhost/api-tikket/public/api/v1/teams/365

HEADERS

Authorization

Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmN
lMTUwNTg2YzE5MmFlMmY1NzIwZjQ0ZmQ0YjdiZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJndWQiOiIxIiwianRpIjoiOTBlZGUy
NzNjZTQxMzVlODAxYTRhYTlyYjg2MTdmZWEyY2lxNTA1ODZjMTkyYWUyZjU3MjBmNDRmZDRlN2JmOTcyZGQ3MTVmYjQ
wOTFlYTYiLCJpYXQiOjE1MjEyMzYzODAsIm5iZiI6MTUyMTIzNjM4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3Bl
cyI6WyIqIl19.tGtnX3dSjGh46SBjZInZ8kIQY429SFN25HrYRohMjD3WxU6-1WMZP-DUChdGjjzGLF3hgyTL4BpW8ULdWgeAJ
K6qXiLzDSj3rffyjYmudw6UleIcuvs1KQNU5-6HGguPrCyt7y5IuhP1JnGxKFfc_R45bYh1wfBoXPP5wjMkfKeb9gRBRJoYscoaQI
hkVQ5VG7ndckne5NkGxogyz1EIVavjZb2dvGuAFPUHt_WMRLTSwVI6oQHflorkNktjJuBKFGLcS4QVWhEeuhaa-Gkdobv0gB
PDe82JaX9IZ8ncIAkWMn7daQ1TZV45Te1k2WSRrDXlpg4XjoQ1DMoIf655z015wpHwYYZG69IHIn_HtWuUkF6aVKb6gLPRg
_Y2-AJCVRexjoJ0vE43Q0LY1M0hjVL8RuCZwPVIaADIbA-5cfdLj9AOVjmDqpNzO7ZMxJEMM_8VR09NOI8G2QukjXDe_xZeVI
wfoLLa8jowITtgni0chsW4_DBnNpwT4WMy3QG2FjGaSGTvEif-dT63fb3aAfuvO-7itmORw_zH69Zwusxfsx79OZOCycjF5LJjg
GpAvy3_91ghI-y5aGtRPr4CeRPjk7DmcFIn2gV_x5HWI3O6LNtvIhc4tAtXmBa_2svHtTcWI2wTqhBbDCfMeXodaHI_OeqPV4O
y1VI7qIEq7tQ

Accept

application/json

Figura 35. API Eliminar equipos método DELETE

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 561,
    "message": "Team successfully deleted"
  },
  "data": {
    "id": 365,
    "users_id": 78,
    "name": "Equipo de prueba 2",
    "domain": "equipodeprueba2",
    "photo": "https://s3.amazonaws.com/files.tikket.net/fixed/defaultuser.png",
    "created_at": "2018-03-16 23:21:25",
    "updated_at": "2018-03-16 23:28:46",
    "deleted_at": null
  }
}
```


Figura 36. Respuesta API eliminar equipos método DELETE

Fuente: Elaboración Propia

C.5 Cambiar el rol de un usuario

PUT Change user's rol

http://localhost/api-tikket/public/api/v1/permissions/134

Cambiar el rol de un operador

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNiMTUwNTg2YzE5MmFlMmY1NzIwZjQ0ZmQ0YjdlZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhdWQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTlyYjg2MTdmZWEyY2IxNTA1ODZjMTkyYWUyZjU3MjBmNDRmZDRiN2JmOTcyZGQ3MTVmYjQwOTFlYTYiLCJpYXQiOjE1MjEyMzYzODAsIm5iZiI6MTUyMTIzNjM4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3dSjGh46SBjZInZ8klQY429SFN25HrYRohMjD3WxU6-1WMZP-DUChdGjjzGLF3hgyTL4BpW8ULdWgeAJK6qXiLzDSj3rffyjYmudw6UIeIcuvs1KQNU5-6HGguPrCyt7y5IuhP1JnGxKFfc_R4SbYh1wfBoXPP5wjMkfKeb9gRBRJaYscoaQIhkVQ5VG7ndckne5NkGxogyz1ElVavjZb2dvGuAFPUHt_WMRLTSwVI6oQHflorkNktJJuBKFGLcS4QVWhEeuhaa-Gkdobv0gBPOe82JaX9IZ8nclAkWMn7daQ1TZV45Te1k2WSRrDXIpg4XjoQ1DMoIf655z015wpHwYYZG69iHIn_HtWuUkF6aVKb6gLPRg_Y2-AJCVRexjoJ0vE43Q0LY1M0hjVL8RuCZwPViaADIbA-5cfdLj9AOVjmDqpNzO7ZMxJEMM_8VR09NOl8G2QukjXDe_xZeVIwfoLLa8JowITtgni0chsW4_DBnNpwT4WMy3QG2FjGa5GTvEif-dT63fb3aAfuvO-7itmORw_zH69Zwusxfsx79OZOCycJF5LJjgGpAvy3_91ghI-y5aGtRPr4CeRPjk7DmcFin2gV_x5HWi3O6LNtvIhc4tAtXmBa_2svHtTcWI2wTqhBbDCfMeXodaHI_OeqPV4Oy1VI7qIEq7tQ
Accept	application/json
Content-Type	application/x-www-form-urlencoded

BODY

rol	88 Identificador del nuevo rol

Figura 37. API Modificar rol de un usuario PUT

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 580,
    "message": "The rol was successfully updated"
  }
}
```


Figura 38. Respuesta API modificar rol de un usuario método PUT

Fuente: Elaboración Propia

C.6 Eliminar usuario de un equipo

DELETE Delete user from team

http://localhost/api-tikket/public/api/v1/permissions/134?user=true

Eliminar operador

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNiMTUwNTg2YzE5MmFlMmY1NzIwZjQ0ZmQ0YjdiZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhdWQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTlyYjg2MTdmZWEyY2IxNTA1ODZjMTkyYWUyZjU3MjBmNDRmZDRiN2JmOTcyZGQ3MTVmYjQwOTFlYTYiLCJpYXQiOjE1MjEyMzYzODAsIm5iZiI6MTUyMTIzNjM4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3dSjGh46SBjZInZ8kIQY429SFN25HrYRohMjD3WxU6-1WMZP-DUChdGjjzGLF3hgyTL4BoW8ULdWgeAjK6qXiLzDSj3rffyjYmudw6UIelcuvs1KQNU5-6HGguPrCyt7y5IuhP1JnGxKFfc_R4SbYh1wfBoXPP5wjMkfKeb9gRBRJoYscoaQIhkVQ5VG7ndckne5NkGxogyz1ElVavjZb2dvGuAFPUHt_WMRLTSwVI6oQHflorkNktJuBKFGLcS4QVWhEeuhaa-Gkdobv0gBPOe82JaX9IZ8nclAkWMn7daQ1TZV45Te1k2WSRrDXlpg4XjoQ1DMolf655z015wpHwYYZG69iHIn_HtWuUkF6aVKb6gLPRg_Y2-AjCVRexjoJ0vE43Q0LY1M0hjVL8RuCZwPVIaADIbA-5cfdLj9AOVjmDqpNzO7ZMxjEMM_8VR09NOl8G2QukjXDe_xZeVIwfoLLa8JowITtgni0chsW4_DBnNpwT4WMy3QG2FjGa5GTvEif-dT63fb3a4fuvO-7itmORw_zH69Zwusxfsx79OZOCycjF5LjjgGpAvy3_91ghI-y5aGtRPr4CeRPjk7DmcFln2gV_x5HWi3O6LNtvIhc4tAtXmBa_2svHtTcWI2wTqhBbDCfMeXodaHI_QeqPV4Oy1VI7qIEq7tQ
Accept	application/json

PARAMS

user	true

Figura 39. API Eliminar usuario de un equipo método DELETE

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 559,
    "message": "The user was successfully deleted"
  }
}
```


Figura 40. Respuesta API eliminar usuario de un equipo método DELETE

Fuente: Elaboración Propia

C.7 Invitar usuario

POST Invitation Basic (First Time)

http://localhost/api-tikket/public/api/v1/teams/262/invitation

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNiMTUwNTg2YzE5MmFlMmY1NzIwZjQ0ZmQ0YjdiZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhdWQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTIyYjg2MTdmZWEyY2IxNTA1ODZjMTkyYWUyZjU3MjBmNDRmZDRiN2JmOTcyZGQ3MTVmYjQwOTFlYTYiLCJpYXQiOjE1MjEyMzYzODAsIm5iZiI6MTUyMTIzNjM4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3dSjGh46SBjZInZ8kIQY429SFN25HrYRohMjD3WxU6-1WMZP-DUChdGjjzGLF3hgyTL4BpW8ULdWgeAJK6qXiLzDSj3rffyjYmudw6UIelcuvs1KQNU5-6HGguPrCyt7y5IuhP1JnGxKFfc_R4SbYh1wfBoXPP5wjMkfKeb9gRBRJoYscoaQIhkVQ5VG7ndckne5NkGxogyz1ElVavjZb2dvGuAFPUHt_WMRLT5wVl6oQHflorkNktJJuBKFGLc54QVWhEeuhaa-Gkdobv0gBPOe82JaX9IZ8nclAkWMn7daQ1TZV45Te1k2WSRrDXIpg4XjoQ1DMoIf655z015wpHwYYZG69iHIn_HtWuUkF6aVKb6gLPRg_Y2-AJCVRexjoJ0vE43Q0LY1M0hjVL8RuCZwPViaADIbA-5cfdLj9AOVjmDqpNzO7ZMxJEMM_8VR09NOI8G2QukjXDe_xZeVIwfoLLa8JowITtgni0chsW4_DBnNpwT4WMy3QG2FjGaSGTvEif-dT63fb3aAfuvO-7ItmORw_zH69Zwusxfsx79OZOCycjF5LJjgGpAvy3_91ghI-y5aGtRPr4CeRPjk7DmcFln2gV_x3HWi3O6LNtvIhc4tAtXmBa_2svHtTcWI2wTqhBbDCfMeXodaHI_OeqPV4Oy1VI7qIEq7tQ
Accept	application/json

BODY

emails[0][email]	apibackendsocialgestcrm@gmail.com Correo del operador
emails[0][rol]	1 Rol del operador

Figura 41. API Invitar usuario método POST

Fuente: Elaboración Propia

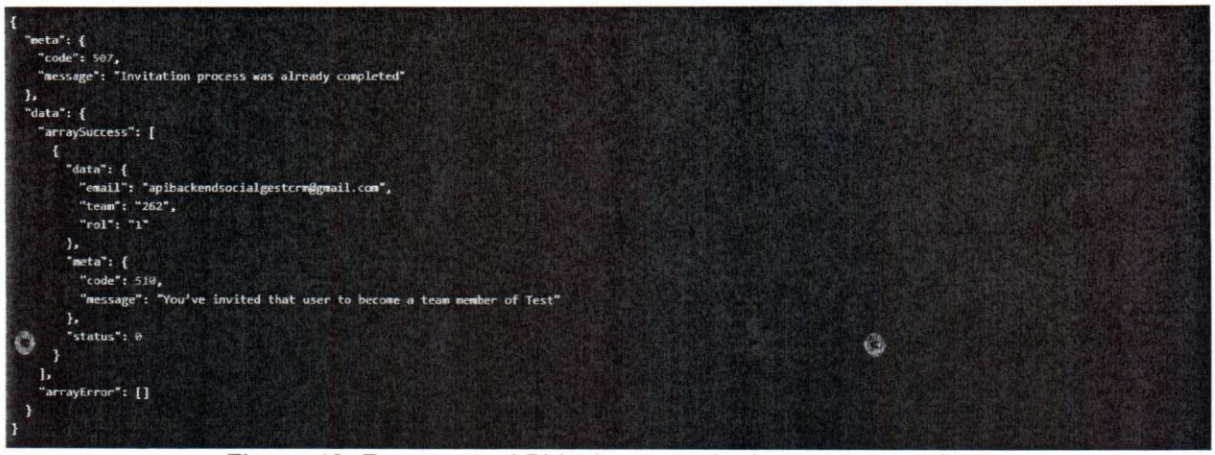


Figura 42. Respuesta API invitar usuario de un equipo POST

Fuente: Elaboración Propia

C.8 Registrar usuario con equipo

POST Register

http://localhost/api-tikket/public/api/v1/users/register

Registro para web

HEADERS

Accept	application/json

BODY

email	develing.22@gmail.com Email de registro para el usuario
password	socialgest Contraseña del usuario
name	Mayra Luna Nombre del usuario
username	mayraluna Alias del usuario
country	VE Ubicación del usuario
timezone	America/Caracas Timezone del usuario
language	es Lenguaje del usuario
team	WorkTeam Equipo que va a ser creado por el usuario

Figura 43. API Registrar usuario con equipo método POST

Fuente: Elaboración Propia

```
{
  "meta": {
    "code": 503,
    "message": "User and team created"
  }
}
```


Figura 44. Respuesta API registrar usuario con equipos método POST

Fuente: Elaboración Propia

Apéndice D: Redes sociales

D.1 Consultar redes sociales

GET List Rrss Users

http://localhost/api-tikket/public/api/v1/teams/262/rrss_users

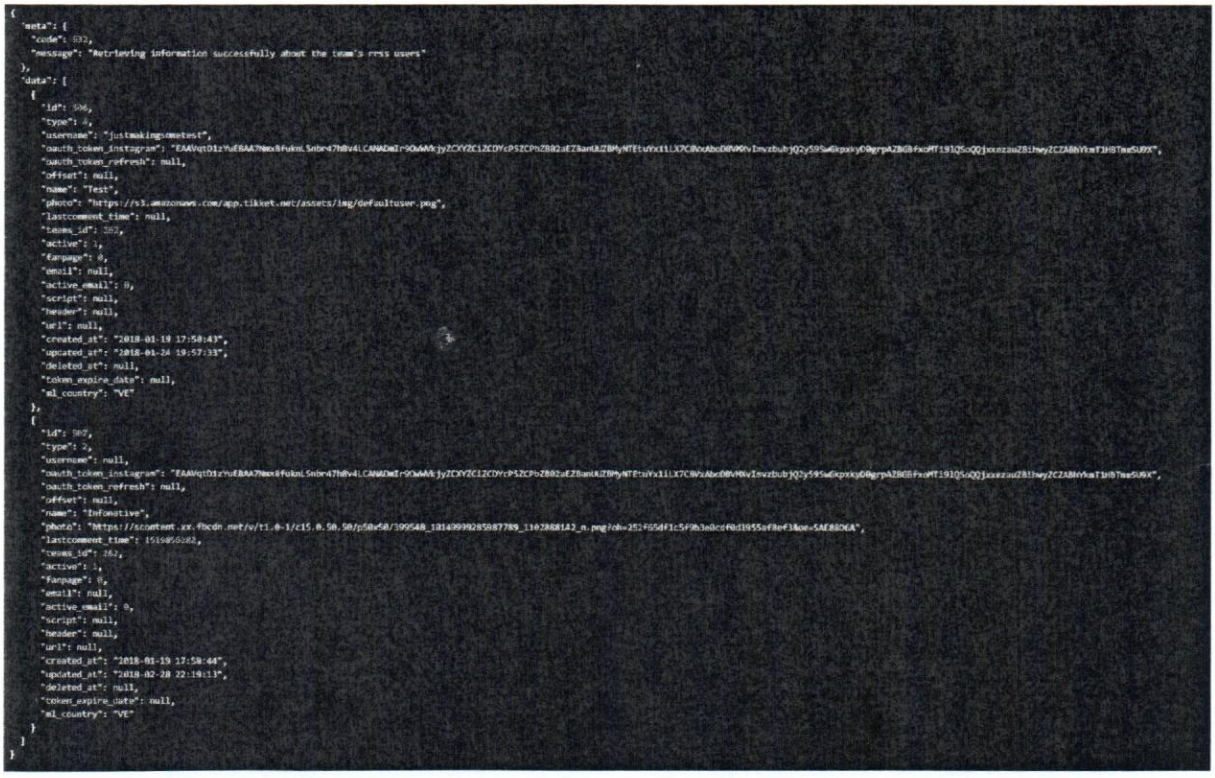
Consultar canal digital

HEADERS

Accept	application/json
Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNiMTUwNTg2YzE5MmFlMmY1NzIwZjQ0ZmQ0YjdlZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJndWQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTlyYjg2MTdmZWEyY2IxNTA1ODZjMTkyYWUyZjU3MjBmNDRmZDRlN2JmOTcyZGQ3MTVmYjQwOTFlYTYiLCJpYXQiOjE1MjEyMzYzODAsIm5iZiI6MTUyMTIzNjM4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3dSjGh46SBjZInZ8klQY429SFN25HrYRohMjD3WxU6-1WMZP-DUChdGjjzGLF3hgyTL4BpW8ULdWgeAjK6qXiLzDSj3rffyjYmudw6UIeIcuvs1KQNU5-6HGguPrCyt7y5IuhP1JnGxKFfc_R45bYh1wfBoXPP5wjMkfKeb9gRBRJoYsccaQIhkVQ5VG7ndckne5NkGxogyz1ElVavjZo2dvGuAFPUHt_WMRLTSwVI6oQHflorkNktjJuBkFGLcS4QVWnEeuhea-Gkdoov0gBPDe82JaX9IZ8nclAkWMn7daQ1TZV45Te1k2WSRrDXlpg4XjoQ1DMolf655z015wqHwYYZG69lHIn_HtWuUkF6aVKo6gLPRg_Y2-AJCVRexjoJ0vE43Q0LY1M0hjVL8RuCZwPVIaADibA-5cfdLj9AlOVjmDqpNzO7ZMxjEMM_8VR09NOI8G2QukjXDe_xZeVIwfoLLa8JowITtgnI0chsW4_DBnNpwT4WMy3QG2FjGaSGTvEif-dT63fo3aAfuvO-7ItmORw_zH69Zwusxfsx79OZOCycjF5LjgGcAvy3_91ghI-y5aGtRPr4CeRPjk7DmcFln2gV_xSHWI3O6LNtvInc4tAtXmBa_2svHtTcWl2wTqnBbDCfMeXodsHI_OeqPV4Oy1VI7qiEq7tQ

Figura 45. API Consultar redes sociales método GET

Fuente: Elaboración Propia




```
{
  "meta": {
    "code": 832,
    "message": "Retrieving information successfully about the team's rrss users"
  },
  "data": [
    {
      "id": 506,
      "type": 4,
      "username": "justmakingsometest",
      "oauth_token_instagram": "EAAVqtD1zYuEBAA7Nmx8fuknLSnbr47h8v4LCANADmIr9OwWVkjyZCXYZC1ZCDYcP5ZCPbZB02aEZBanUAZBMyNTEtuYx1iLX7CBVxAbcD0VMNvImvzbubjQ2y59SwGkpxkyD0grpAZBGBfxoMT191QSoQQjxxezauZBihwyZCZABhYkmT1HBTmeSU9X",
      "oauth_token_refresh": null,
      "offset": null,
      "name": "Test",
      "photo": "https://s3.amazonaws.com/app.tikket.net/assets/img/defaultuser.png",
      "lastcomment_time": null,
      "teams_id": 262,
      "active": 1,
      "fanpage": 0,
      "email": null,
      "active_email": 0,
      "script": null,
      "header": null,
      "url": null,
      "created_at": "2018-01-19 17:50:43",
      "updated_at": "2018-01-24 19:57:33",
      "deleted_at": null,
      "token_expire_date": null,
      "ml_country": "VE"
    },
    {
      "id": 507,
      "type": 2,
      "username": null,
      "oauth_token_instagram": "EAAVqtD1zYuEBAA7Nmx8fuknLSnbr47h8v4LCANADmIr9OwWVkjyZCXYZC1ZCDYcP5ZCPbZB02aEZBanUAZBMyNTEtuYx1iLX7CBVxAbcD0VMNvImvzbubjQ2y59SwGkpxkyD0grpAZBGBfxoMT191QSoQQjxxezauZBihwyZCZABhYkmT1HBTmeSU9X",
      "oauth_token_refresh": null,
      "offset": null,
      "name": "Infomative",
      "photo": "https://scontent.xx.fbcdn.net/v/t1.0-1/c15.0.50.50/p50x50/399540_10149999285987789_1102888142_n.png?oh=252f65df1c5f9b3e0cdf0d1955af8ef3&oe=5AE88D6A",
      "lastcomment_time": 1519056282,
      "teams_id": 262,
      "active": 1,
      "fanpage": 0,
      "email": null,
      "active_email": 0,
      "script": null,
      "header": null,
      "url": null,
      "created_at": "2018-01-19 17:50:44",
      "updated_at": "2018-02-28 22:19:13",
      "deleted_at": null,
      "token_expire_date": null,
      "ml_country": "VE"
    }
  ]
}
```


Figura 46. Respuesta API consultar redes sociales método GET

Fuente: Elaboración Propia

D.2 Agregar redes sociales

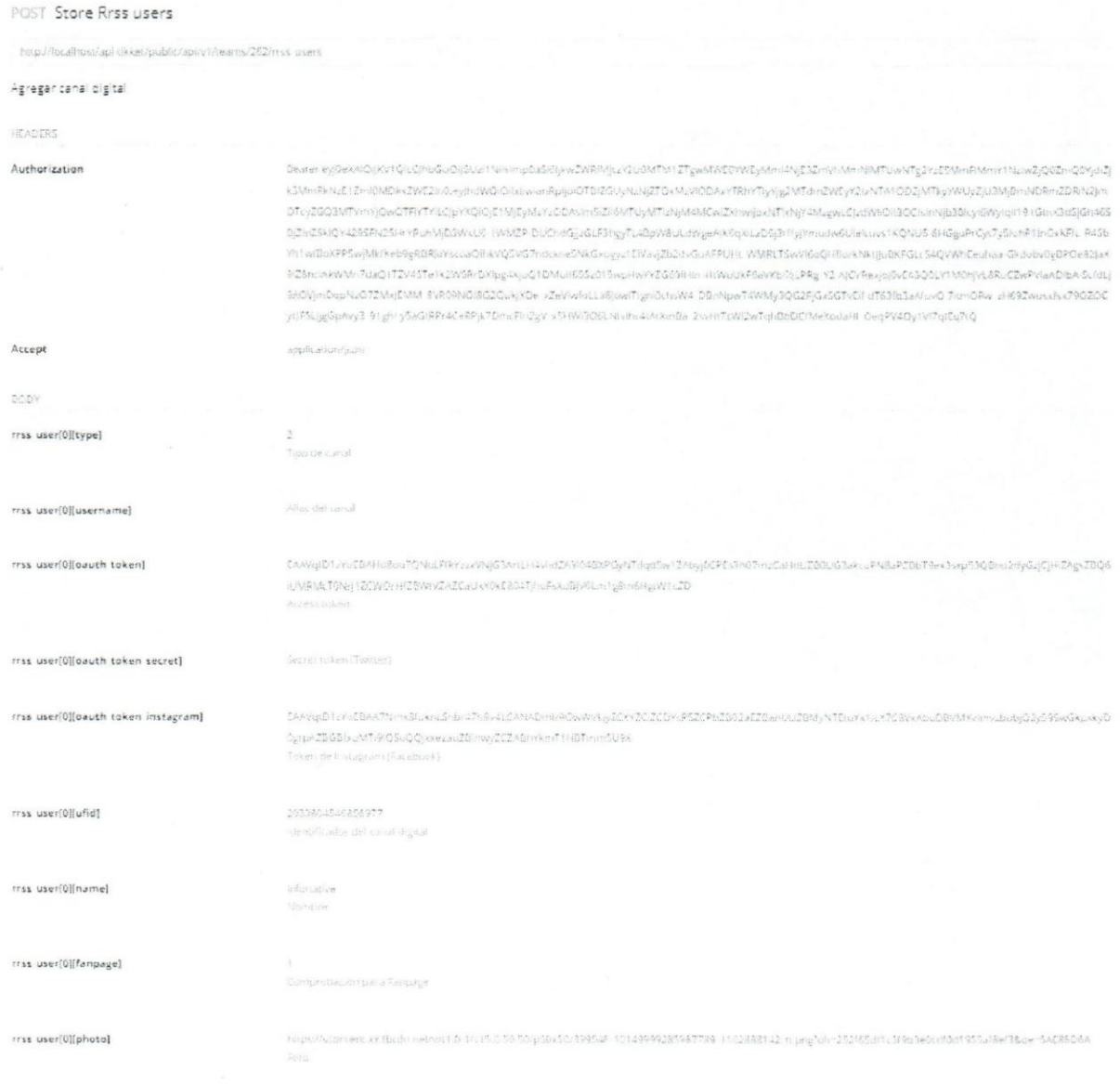


Figura 47. API Agregar redes sociales método POST

Fuente: Elaboración Propia

Figura 48. Respuesta API agregar redes sociales método POST

Fuente: Elaboración Propia

D.3 Eliminar redes sociales

DELETE Delete Rrsss Users

http://localhost/api-tikket/public/api/v1/rrss_users/506

Eliminar canal digital

HEADERS

Accept	application/json
Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNlMTUwNTg2YzE5MmFlMmY1NzIwZjQ0ZmQ0YjdlZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhdWQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTIyYjg2MTdmZWEyY2IxNTA1ODZjMTkyYWUyZjU3MjBmNDRmZDRiN2JmOTcyZGQ3MTVmYjQwOTFlYTYiLCJpYXQiOjE1MjEyMzYzODAsImSlZlI6MTUyMTIzNjM4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3dSjGh465BjZInZ8klQY429SFN25HrYRonMjD3WkU6-1WMZP-DUChdlGjjzGLF3hgyTL4BpW8ULdWgeAjK6qXlLzD5j3/ffyjYmudw6UIeIcuvs1KQNU5-6HGguPrCyt7y5IuhP1JnGxKFfc_R4SbYn1wfBoXPP5wjMkfKeb9gRBRJoYscoaQInkVQ5VG7ndckne5NkGxogyz1ElVavjZbZdvGuAFPUHt_WMRLTSwVI6oQHflorkNktjJuBKFGLcS4QVWnEeuhaa-GkdobvOgBPOe82JaX9IZ8nclAkWMn7daQ1TZV45Te1k2WSRrDXIpg4XjoQ1DMolf655z015wpHwYYZG69IHIn_HtWUUkF6aVKb6gLPRg_Y2-AJCVRexjoj0vE43Q0LY1M0hjVL8RuCZwPVIaADIbA-5cfdLj9AOVjmDqpNzO7ZMxJEMM_3VR09NOI8G2QukjXDe_xZeVIwfoLLa8JowITcgni0chsW4_DBnNpwT4WMy3QG2FjGaSGTvEif-qT63fo3aAfuv0-7itmORw_zH69Zwusxfsx79OZOCycjF5LjjgGpAvy3_91ghI-y3aGtRPr4CeRPjk7DmcFln2gV_x5HWI3O6LNtvihc4tAtXmBa_2svHtTcWl2wTqhBbDCfMeXcdaHI_OeqPV4Oy1VI7qIEq7tQ

Figura 49. API Eliminar redes sociales método DELETE

Fuente: Elaboración Propia

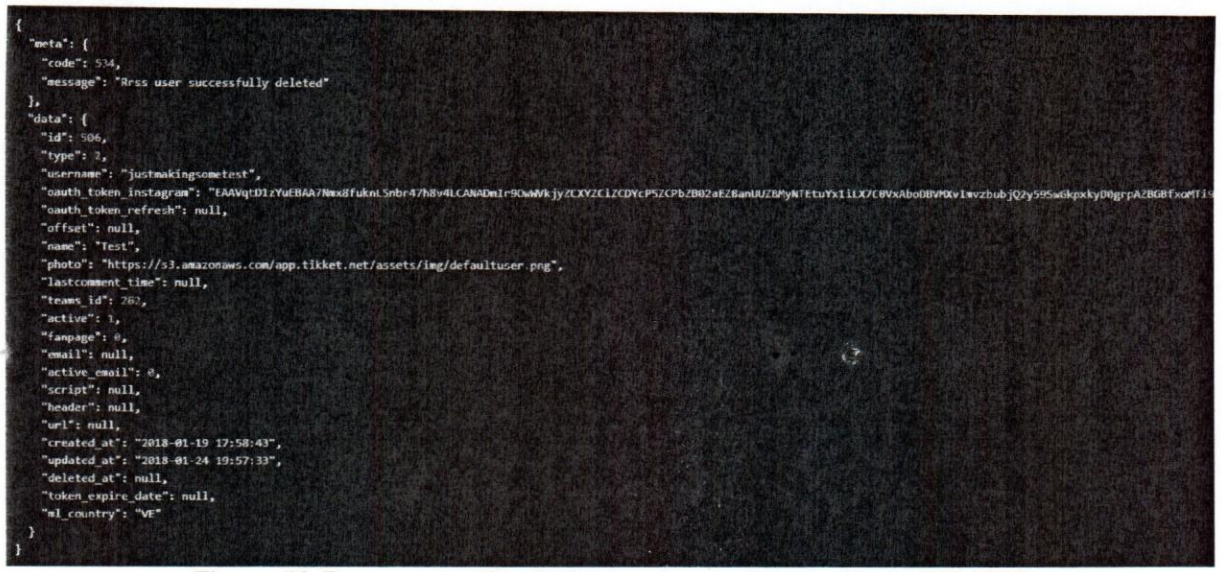


Figura 50.Respuesta API eliminar redes sociales método DELETE

Fuente: Elaboración Propia

D.4 Agregar chat

Figura 51. API Agregar chat método POST

Fuente: Elaboración Propia

```
{
  "meta": {
    "code": 502,
    "message": "Chat was inserted successfully"
  },
  "data": {
    "name": "Prueba",
    "script": "<script src=\"https://files.tikket.net/fixed/widget.js\" data-id=d8e66e3142c4b26d577affa4409f5d7a></script>",
    "url": "https://app.tikket.net/chat?id=d8e66e3142c4b26d577affa4409f5d7a",
    "photo": "https://files.tikket.net/fixed/tikket-icon.png",
    "email": "arguacache.4@gmail.com",
    "active_email": 1,
    "type": 3,
    "teams_id": "1",
    "updated_at": "2018-03-17 08:32:02",
    "created_at": "2018-03-17 08:32:02",
    "id": 629
  }
}
```


Figura 52. Respuesta API agregar chat método POST

Fuente: Elaboración Propia

Apéndice E: Casos

E.1 Responder caso

POST Answer Ticket

http://localhost/api-tikket/public/api/v1/tickets/181460/messages

Responder ticket

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNiMTUwNTg2YzE5MmFlMmY1NzIwZjQ0ZmQ0YjdiZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhdWQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTlyYjg2MTdmZWEyY2IxNTA1ODZjMTkyYWUyZjU3MjBmNDRmZDRiN2JmOTcyZGQ3MTVmYjQwOTFlYTYiLCJpYXQiOjE1MjEyMzYzODAsIm5iZiI6MTUyMTIzNjM4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3dSjGh46SBjZInZ8kIQY429SFN25HrYRohMjD3WxU6-1WMZP-DUChdGjjzGLF3hgyTL4BpW8ULdWgeAjK6qXiLzDSj3rffyjYmudw6UIeIcuvs1KQNU5-6HGguPrCyt7y5IuhP1JnGxKFfc_R4SbYh1wfBoXPP5wjMkfKeb9gRBRJoYscoaQIhkVQ5VG7ndckne5NkGxogyz1ElVavjZb2dvGuAFPUHt_WMRLTSwVI6oQHflorkNktJjuBKFGLcS4QVWhEeuhaa-Gkdobv0gBPOe82JaX9IZ8nclAkWMn7daQ1TZV45Te1k2WSRrDXIpg4XjoQ1DMoIf655z015wpHwYYZG69iHIn_HtWuUkF6aVKb6gLPRg_Y2-AJCVRexjoJ0vE43Q0LY1M0hjVL8RuCZwPVIaADlbA-5cfdLj9AOVjmDqpNzO7ZMxjEMM_8VR09NOI8G2QukjXDe_xZeVIwfoLLa8JowITtgni0chsW4_DBnNpwT4WMy3QG2FjGaSGTvEif-dT63fb3aAfuvO-7itmORw_zH69Zwusxfsx79OZOCycjF5LjjgGpAvy3_91ghI-y5aGtRPr4CeRPjk7DmcFln2gV_x5HWi3O6LNtvIhc4tAtXmBa_2svHtTcWl2wTqhBbDCfMeXodaHI_OeqPV4Oy1VI7qIEq7tQ
Accept	application/json

BODY

type	0 Tipo de mensaje
text	Respondiendo mensaje de prueba Texto del mensaje

Figura 53. API Responder caso método POST

Fuente: Elaboración Propia

```
{
  "meta": {
    "code": 207,
    "message": "Message successfully sent"
  },
  "data": {
    "tickets_id": 181460,
    "users_id": 78,
    "message": "Respondiendo mensaje de prueba",
    "source": 3,
    "comment_id": "2043734609197304_2062037607367004",
    "chat_id": "t_100024380589985",
    "updated_at": "2018-03-18 17:39:04",
    "created_at": "2018-03-18 17:39:04",
    "id": 548498
  }
}
```


Figura 54. Respuesta API responder caso método POST

Fuente: Elaboración Propia

E.2 Responder chat

POST StoreMessagesClient

http://localhost/api-tikket/public/api/v1/tickets/182009/messageclient

Enviar mensaje desde el chat

HEADERS

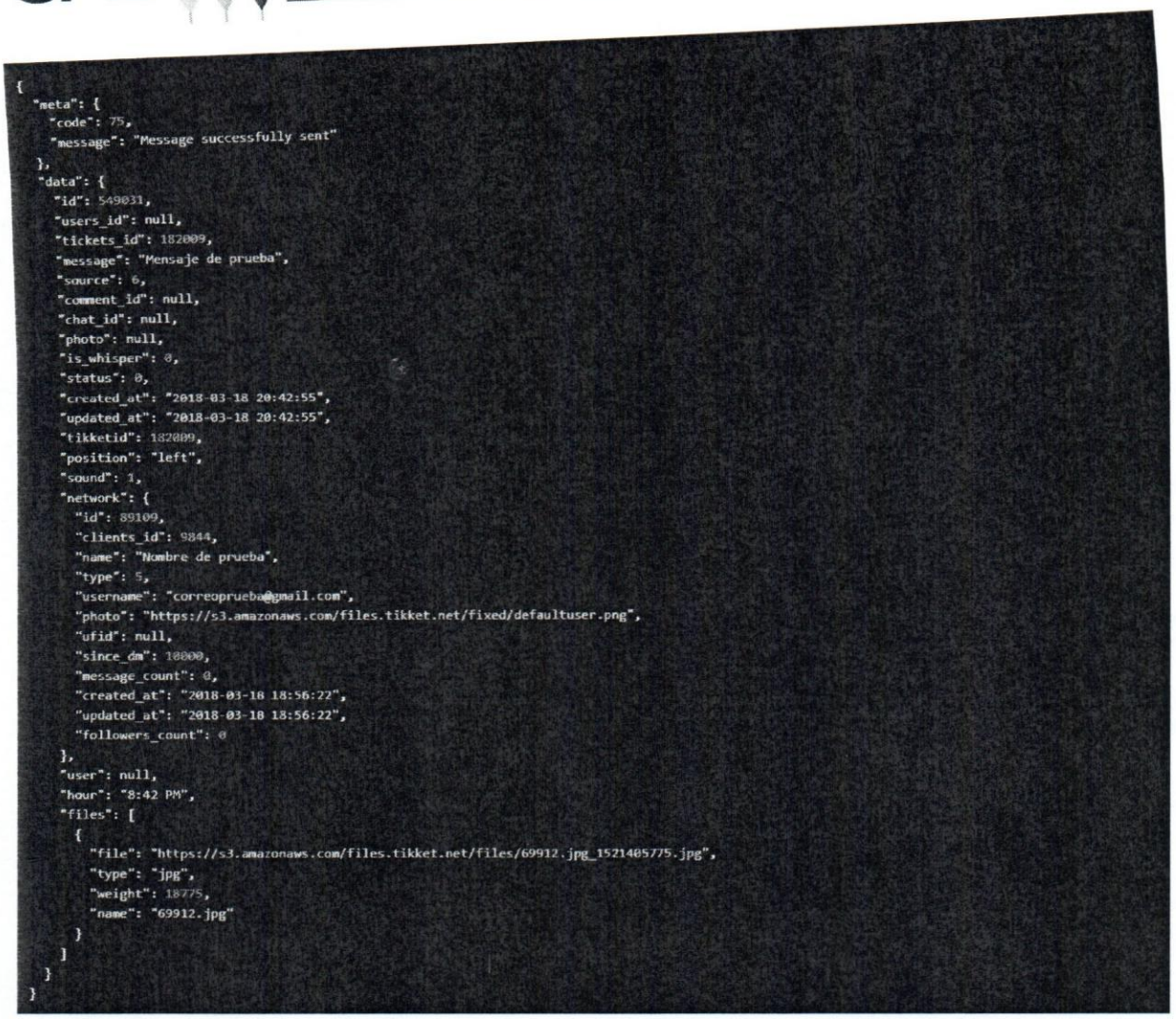
Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjE3NWZjMjdlZDU2ZTg0MGJjNDMzZjU2NDFlZTE3YWM1MjU3MzEzNjQwMjVhZmIwZWNhNTkyMmRhNjljM2U3NzBiYWQ0YjYxYTAyOWZhN2IwIn0.eyJhdWQiOiIxIiwianRpIjoiMTc1ZmMyN2VkNTZlODQwYmM0MzNmNTY0MWJlMTdhYzUyNTczMTM2NDAyNWFmYjBlY2E1OTIyZGE2OWMzZTc3MGJhZDRiNjFhMDI5ZmE3YjAiLCJpYXQiOjE1MjA1Mzc1MTIsImNiZiI6MTUyMDUzNzUxMiwiZXhwIjoxNTIwOTY5NTEyLCJzdWIiOiI4Iiwic2NvcGVzIjpbIioiXX0.Zrmz7YAQRHtmtjE-w2knw2UgRZeEOX_w7Vffdmx86eOIrk-Rbps112eU1bPIC76L08Tdq9_OZ58cF3fTnnb1cKfi1HLd5jpci7fJPIMIJAFTwNj5mrCOHHg0kSNH1IpBV9kzcj_IG1IzuhrWjBcBnZhm-wpgOTLOtfwPydMJU3Ij-5n8FEkWfbcO1JDBAS71DLa7fCxfoz_2akajgANGKfYXVZ73uVBHsifs9paNvRVwHQE45OrNK3N0E4X60wLRIFqsWoNeu70eYWv0UxaGsB4J9sDdOJdyfYCD4m2b24SQqi_UWNyCZDDf1jo2UhuPyXWupOfAPGIV9fZMsiwtVMcDUuj6wohqaInJVttblp8Tityi6bd1sFIbXs9stLjt_zTOzTfgQwPdh7I6Uv7J0fMokdMdMDiCaJXd3VhIjqO0_1SG7Gmww3q85qi8tI53U72z-cGB3Ir6REAy3KBH1WqqSG5IiyyaYhfhNd5OLu7IIvt3Uj-xImvYLargEpVXII-v5b0BY1TATI03odBylIYxQ-msC4OhtPBULoVp2NMDHD3_qXiycHRhKh1VAtMepklpf75NpTYQTc5mymIG2-IMmKM6OUKleEc1GgGdKW3kxV_PAaMjtZe77bxCub41uajctPQhBdzvSzBWDgZ8tecL0TIJXxab3svCxvxKdyE
Accept	application/json

BODY

text	ok pues Mensaje a enviar
token	eyJpdiI6IldQUFhNSGVoYjFqd0hRMjVjMHF4TEE9PSIsInZhbHVlIjoiXC9oTXpwdmwzbVp1b0dJQlJlb2Ewemc9PSIsIm1hYyI6Ijk4OTliYjg2ZTc4NzJlYjA1NGMwMTJlYzQ2NDViNDE0OTlkMGNiZDVmMmNjNTczYmRhYTJjMDM5NTYyMmU0NTlifQ== Token
files[]	Archivo

Figura 55. API Responder chat método POST

Fuente: Elaboración Propia




```
{
  "meta": {
    "code": 75,
    "message": "Message successfully sent"
  },
  "data": {
    "id": 549031,
    "users_id": null,
    "tickets_id": 182009,
    "message": "Mensaje de prueba",
    "source": 6,
    "comment_id": null,
    "chat_id": null,
    "photo": null,
    "is_whisper": 0,
    "status": 0,
    "created_at": "2018-03-18 20:42:55",
    "updated_at": "2018-03-18 20:42:55",
    "tikketid": 182009,
    "position": "left",
    "sound": 1,
    "network": {
      "id": 89109,
      "clients_id": 9844,
      "name": "Nombre de prueba",
      "type": 5,
      "username": "correoprueba@gmail.com",
      "photo": "https://s3.amazonaws.com/files.tikket.net/fixed/defaultuser.png",
      "ufid": null,
      "since_dm": 18000,
      "message_count": 0,
      "created_at": "2018-03-18 18:56:22",
      "updated_at": "2018-03-18 18:56:22",
      "followers_count": 0
    },
    "user": null,
    "hour": "8:42 PM",
    "files": [
      {
        "file": "https://s3.amazonaws.com/files.tikket.net/files/69912.jpg_1521405775.jpg",
        "type": "jpg",
        "weight": 18775,
        "name": "69912.jpg"
      }
    ]
  }
}
```


Figura 56. Respuesta API responder chat método POST

Fuente: Elaboración Propia

Apéndice F: Mensajes

F.1 Consultar casos

Tickets

GET Index Tickets

http://localhost/api-tikket/public/api/v1/tickets?date=&rangedate=&network=&limit=15&offset=0&team=252&mode=null

Consultar información de tickets

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNiMTUwNTg2YzE5MmFlMmY1NzIwZjQ0ZmQ0YjdiZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhdWQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTlyYjg2MTdmZWEyY2IxNTA1ODZjMTkyYWUyZjU3MjBmNDRmZDRlN2JmOTcyZGQ3MTVmYjQwOTFlYTYiLCJpYXQiOjE1MjEyMzYzODAsImF1ZiI6MTUyMTIzNjM4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3d5jGh465BjZInZ8klQY4295FN25HrYRohMJD3WxU6-1WMZP-DUChdGjjzGLF3hgyTL4BpW8ULdWgeAjK6qXiLzD5j3rffyjYmudw6UIeIcuvs1KQNU5-6HGguPrCyt7y5IuhP1JnGxKFfc_R4SoYh1wfBoXPP5wjMkfKeb9gRBRJoYscoaQInKVQ5VG7ndckne5NkGxogyz1ElVavjZo2dvGuAFPUHt_WMRLTSwVl6oQHflorkNktjJuBKFGLcS4QVWhEeuhaa-Gkdobv0gBPOe82JaX9IZ8nclAkWMn7daQ1TZV45Te1k2WSRrDXlog4XjoQ1DMolf655z015wpHwYYZG69lHln_HtWuUkF6aVKb6gLPRg_Y2-AJCVRexjoJ0vE43Q0LY1M0hjVL8RuCIwPVlaADIbA-5cfdLj9AOVjmDqpNzO7ZMxjEMM_8VR09NOl8G2QukjXDe_xZeVIwfoLLa8JowiTtgnI0chsW4_DBnNpwT4WMy3QG2FjGaSGTvElf-dT63fb3aAfuvO-7ItmORw_zH69Zwusxfsx790ZOCycjF5LjjgGpAvy3_91ghI-y5aGtRPr4CeRPJk7DmcFln2gV_x5HWI3O6LNtvIhc4tAtXmBa_2svHtTcWl2wTqhBbDCfMeXodaHI_OeqPV4Oy1VI7qIEq7tQ
Accept	application/json

PARAMS

date	
rangedate	
network	
limit	15
offset	0
team	252
mode	null

Figura 57. API Consultar casos método GET

Fuente: Elaboración Propia

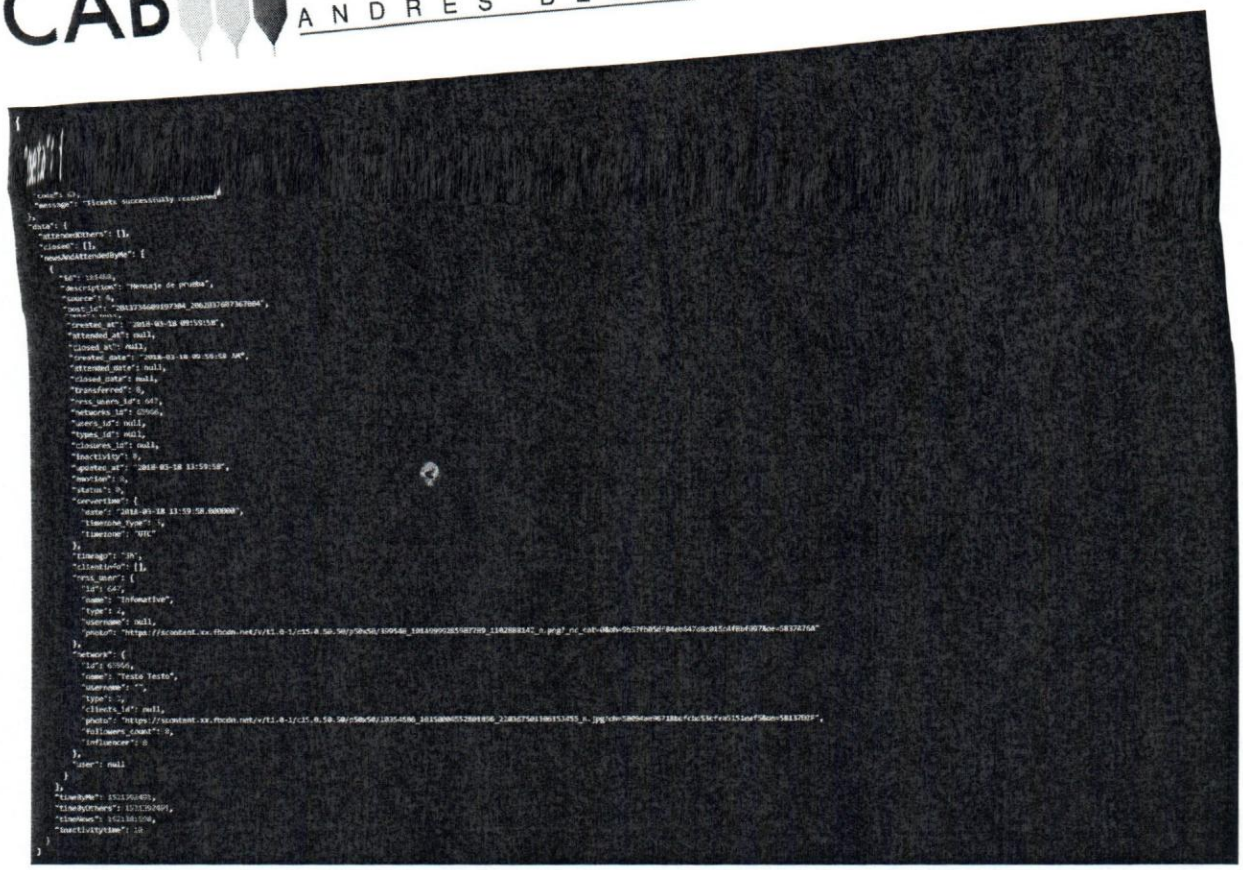


Figura 58. Respuesta API consultar casos método GET

Fuente: Elaboración Propia

F.2 Consultar mensajes

GET Messages Ticket

http://localhost/api-tikket/public/api/v1/tickets/181460/messages?limit=15&offset=0

Consultar mensajes

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNiMTUwNTg2YzE5MmFlMmY1NzIwZjQ0ZmQ0YjdlZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhdWQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTlyYjg2MTdmZWEyY2IxNTA1ODZjMTkyYWUyZjU3MjBmNDRmZDRlN2JmOTcyZGQ3MTVmYjQwOTFlYTYiLCJpYXQiOjE1MjEyMzYzODAsIm5iZiI6MTUyMTIzNjM4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3dSjGh46SBjZInZ8klQY429SFN25HrYRohMjD3WxU6-1WMZP-DUChdGjjzGLF3hgyTL4BpW8ULdWgeAjK6qXiLzDSj3rffyjYmudw6UlelcuvvS1KQNU5-6HGguPrCyt7y5IuhP1JnGxKFfc_R4SbYh1wfBoXPP5wjMkfKeb9gRBRJoYscoaQIhkVQ5VG7ndckne5NkGxogyz1ElVavjZb2dvGuAFPUHt_WMRLT5wVI6oQHflorkNktJjuBKFGLcS4QVWhEeuhaa-Gkdobv0gBPOe82JaX9IZ8nclAkWMn7daQ1TZV45Te1k2WSRrDXIpg4XjoQ1DMolf655z015wpHwYYZG69iHIn_HtWuUkF6aVKb6gLPRg_Y2-AJCVRexjaJ0vE43Q0LY1M0hjVL8RuCZwPVIaADIbA-5cfdLj9AOVjmDqpNzO7ZMxjEMM_8VR09NOI8G2QukjXDe_xZeVIwfoLLa8JowITtgni0chsW4_DBnNpwT4WMy3QG2FjGaSGTvEif-dT63fb3aAfuvO-7itmORw_zH69ZwusxfsX79OZOCycJF5LJjgGpAvy3_91ghI-y5aGtRPr4CeRPjk7DmcFln2gV_x5HWi3O6LNtvIhc4tAt[illegible]mBa_2svHtTcWl2wTqhBbDCfMeXodaHI_OeqPV4Oy1Vl7qIEq7tQ
Accept	application/json
PARAMS	
limit	15
offset	0

Figura 59. API Consultar mensajes método GET

Fuente: Elaboración Propia

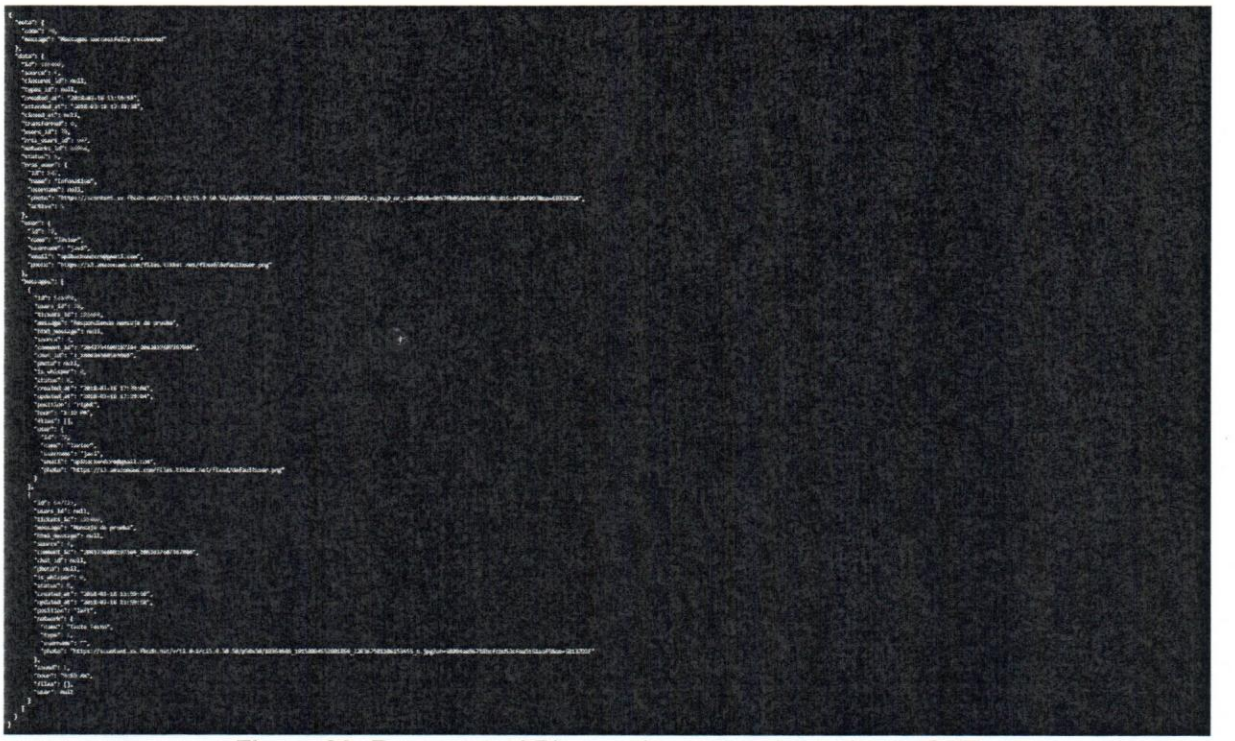


Figura 60. Respuesta API consultar mensajes método GET

Fuente: Elaboración Propia

UCAB Universidad Católica ANDRÉS BELLO

Apéndice G: Carrito de compra

G.1 Consultar ordenes

GET Index Orders

http://localhost/api-tikket/public/api/v1/shoppingcart

Consultar las orden

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4N jE3ZmVhMmNiMTUwNTg2YzE5MmFlMmY1NzIwZjQ0ZmQ0YjdlZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhd WQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTIyYjg2MTdmZWEyY2IxNTA1ODZjMTkyYWUyZjU 3MjBmNDRmZDRiN2JmOTcyZGQ3MTVmYjQwOTFlYTYiLCJpYXQiOjE1MjEyMzYzODAsIm5iZiI6MTUyMTIzNj M4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3d5jGh46SBjZInZ8klQY429S FN25HrYRohMjD3WxU6-1WMZP-DUChdGjjzGLF3hgyTL4BpW8ULdWgeAJK6qXiLzDSj3rffyjYmudw6UIelcuv s1KQNU5-6HGguPrCyt7y5IuhP1JnGxKFfc_R45bYh1wfBoXPP5wjMkfKeb9gRBRJoYscoaQIhkVQ5VG7ndckne 5NkGxogyz1ElVevjZb2dvGuAFPUHt_WMRLTSwVI6oQHflorkNktJJuBKFGLcS4QVWhEeuIhaa-Gkdobv0gBPOe 82JaX9IZ8nclAkWMn7daQ1TZV45Te1k2WSRrDXIpg4XjoQ1DMoIf655z015wpHwYYZG69iHln_HtWuUkF6aV Kb6gLPRg_Y2-AjCVRexjoJ0vE43Q0LY1M0hjVL8RuCZwPVIaADIbA-5cfdLj9AOVjmDqpNzO7ZMxjEMM_8VR09 NOI8G2QukjXDe_xZeVIwfoLLa8JowITtgni0chsW4_DBnNpwT4WMy3QG2FjGaSGTvEif-dT63fb3aAfuvO-7itm ORw_zH69Zwusxfsx79OZOCycjF5LJjgGpAvy3_91ghI-y5aGtRPr4CeRPjk7DmcFln2gV_x5HWi3O6LNtvIhc4tAt XmBa_2svHtTcWI2wTqhBbDCfMeXodaHl_OeqPV4Oy1VI7qIEq7tQ
Accept	application/json

Figura 61. API Consultar ordenes método GET

Fuente: Elaboración Propia

```
{
  "meta": {
    "code": 01,
    "message": "Orders recovered"
  },
  "data": [
    {
      "id": 72,
      "currency": 1,
      "cycle": 1,
      "users_id": 78,
      "idplan": 3,
      "expiration_date": "2018-04-02",
      "paid": 0,
      "value": 50,
      "iva": 0,
      "irpf": 0,
      "bill": 0,
      "plan_name": "Pro",
      "coupon": 0,
      "amount": 50
    }
  ]
}
```


Figura 62. Respuesta API consultar ordenes método GET

Fuente: Elaboración Propia

G.2 Agregar orden

POST Store Orders

http://localhost/api-tikket/public/api/v1/shoppingcart

Agregar orden

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6 [illegible]
Accept	application/json

BODY

type	0 Tipo de orden
currency	1 Tipo de moneda
bill	1 Envío de factura
cycle	1 Ciclo de plan (Mensual, Trimestral, Semestral, Anual)
plan	3 Tipo de plan
licence_number	2 Número de lincencias

Figura 63. API Agregar ordenes método POST

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 3,
    "message": "Order Created"
  },
  "data": {
    "users_id": 78,
    "plans_id": "3",
    "currency": "1",
    "price": 50,
    "cycle": "1",
    "value": 50,
    "discount": 0,
    "discountpercentage": 0,
    "iva": 0,
    "ivapercentage": 0,
    "bill": 0,
    "irpf": 0,
    "expiration_date": "2018-04-02",
    "updated_at": "2018-03-18 00:16:13",
    "created_at": "2018-03-18 00:16:13",
    "id": 72
  }
}
```


Figura 64. Respuesta API agregar ordenes método POST

Fuente: Elaboración Propia

G.3 Eliminar orden

DELETE Delete Orders

http://localhost/api-tikket/public/api/v1/shoppingcart/80

Eliminar orden

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6 [illegible]
Accept	application/json

BODY

type	0 0 - new / 1 - renewal / 2 - upgrade
currency	1 1 - $ / 2 - Bs. / 3 - eur
bill	1 1 - con envio / 2 - sin envio
cycle	3 1 mes 3 meses 6 meses y 12 meses
plan	2 2 3 y 4
licence_number	10 infinito MINIMO 1 siempre

Figura 65. API Eliminar ordenes método DELETE

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 10,
    "message": "Orden deleted"
  }
}
```


Figura 66. Respuesta API eliminar ordenes método DELETE

Fuente: Elaboración Propia

G.4 Consultar planes

GET Index plans

http://localhost/api-tikket/public/api/v1/plans

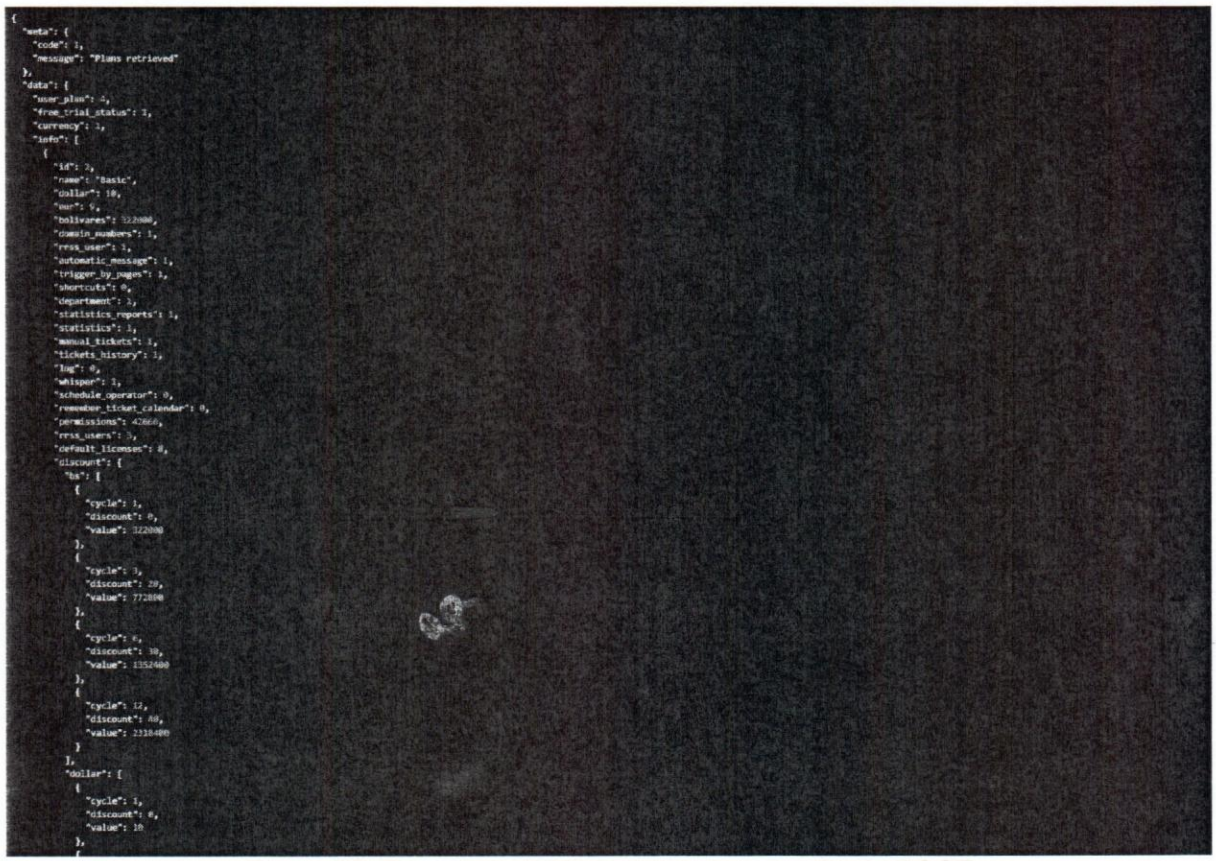
Lista todas las planes

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6ImUyNTNjMzViMTQ0ZGVlNGFmNDZkMTZmNTI2Njg2OWZmNWU5MDZkN2YxMTAxNWNhODg2NWNhODg1NDE3ZDYwNGEyZWM5NWRmNDM1ZmY1YTA2InO.eyJhdWQiOiIxIiwianRpIjoiZTI1M2MzNWIxNDRkZWU0YWY0NmQxNmY1MjY2ODY5ZmY1ZTkwNmQ3ZjExMDE1Y2E4ODY1Y2E4ODU0MTdkNjA0YTjlYzk1ZGY0MzVmZjVhMDYiLCJpYXQiOjE1MDczMTQ1ODQsIm5iZiI6MTUwNzMxNDU4NCwiZXhwIjoxNTA3NzQ2NTg0LCjzdWIiOiI0IiwiC2NvcGVzIjpbIlcXX0.mr2M5yBNjG0D4_ixCNnRwz3xv0Cu5KUY4ihjwVo_aS573gwnSxPSEII22IKwVmxQFIsIW-MNW92a7wCnHs5BU6xm_xWCfURxtG6m3fMXqaPjEjwiuN3ytwHBL3eZGzjN6s3rI4p0jmZuCt3H9GjuuRgaOE5vnIo5HmDgOu0AK-C7UrI5-nSkjP14Br7wPr9fBsDC35gWeu8QBTN8DrFsZm6KMRj4E88Gzx3Efzug8kwOFvP45e60e9I4vQoP0z9kwbgZRrMVmrGph10x2ri1gtSBTMvkDTfojCjoejU7cCoqMqe5E3pZLoIO_CnWggY40uGKvI4pLIQiQyaFAIh73mQ1nJUgJqDLkmY0o73zi_PGwyv7IsYHbLCF0BceuEq80MZq4ojhitLo3wbri9NPk84A5Kx7e6wv4SAePZvwIOCkpsr7-tzUQRt5yC3C3bmaJcvs6Bu8Nzg42x8p5nmHV2ddcM_UfP1dAR5DOGfr4e9y5o2H5FiwMZSR7Yk6H4eTfkRMepR1ovjIHfyNzB3sP6TnjfjvCngKa4gIUP7YAhNWYqtckkp5Zh_Zd50nFcifnoxx4dUVrLQsoe1jvxwodNjNINvnWmeBNKd1t9EFBKOgEu9gLkZi4ep8-QGHLxCpx9iqG0B91myhHZ7ikvs28Q3UOIIjXP27QNKj5xI
Accept	application/json

Figura 67. API Consultar planes método GET

Fuente: Elaboración Propia



```
{
  "meta": {
    "code": 1,
    "message": "Plans retrieved"
  },
  "data": {
    "user_plan": 4,
    "free_trial_status": 1,
    "currency": 1,
    "info": [
      {
        "id": 2,
        "name": "Basic",
        "dollar": 10,
        "eur": 9,
        "bolivares": 322000,
        "domain_numbers": 1,
        "rrss_user": 1,
        "automatic_message": 1,
        "trigger_by_pages": 1,
        "shortcuts": 0,
        "department": 1,
        "statistics_reports": 1,
        "statistics": 1,
        "manual_tickets": 1,
        "tickets_history": 1,
        "log": 0,
        "whisper": 1,
        "schedule_operator": 0,
        "remember_ticket_calendar": 0,
        "permissions": 42660,
        "rrss_users": 3,
        "default_licenses": 0,
        "discount": {
          "bs": [
            {
              "cycle": 1,
              "discount": 0,
              "value": 322000
            },
            {
              "cycle": 3,
              "discount": 20,
              "value": 772800
            },
            {
              "cycle": 6,
              "discount": 30,
              "value": 1352400
            },
            {
              "cycle": 12,
              "discount": 40,
              "value": 2318400
            }
          ],
          "dollar": [
            {
              "cycle": 1,
              "discount": 0,
              "value": 10
            },
            {
```


Figura 68. Respuesta API consultar planes método POST

Fuente: Elaboración Propia

Apéndice H: Pasarelas de pago

H.1 Notificar transferencia

POST Deposit Transfer

http://localhost/api-tikket/public/api/v1/orders/72/payments/deposit_transfer

Reportar depósito

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6ImU3NDc3NGI5NjNmYTgxYWE2ZGI4ZjkwYWI3Y2EyMWRmODhlNmRlYWRmZGE5NmU0ZDYx YTI5MWY1NWY4YmRlMjNmZGQ1NGUzNjMxODFjNGRmIn0.eyJhdWQiOiIxIiwianRpIjoiZTc0Nzc0YjK2M2ZhODFhYTZkYjhmOTBhYjdjYTIxZGY4OGI2ZGJ hZGZkYTk2ZTRkNjFhMjkxZjU1ZjhiZGUyM2ZkZDU0ZTM2MzE4MWM0ZGYiLCJpYXQiOjE1MjE0MTMzOTAsIm5iZiI6MTUyMTQxMzM5MCwiZXhwIjoxNTIx ODQ1MzkwLCJzdWIiOiI3OCIsInNjb3BIcyI6WyIqIl19.IFOA2qh9PjX6IosPuyhKXiGh7ossjkyK85x2oUDhAzdwXBLBDihVO9GCDdc5bgraArHhRumav6Nh2 RlstOIdV1V8PSuWKE9KmydY5ATI3mvEN_zLakICkUcC_TOqv85QdR3ZHenB5fjvLpAcoi_Pbtg130CIYS1QcKOU-QEFMNhjWPMWvZZzi0sg6ZCj4MF6OoY H4ELFoH4hwpYjuYj1i_zIB925UOCtIGVQhCrUPvsVR0ihDZIFau8fBAAIydvMkwTCpTC5Mt74_LBU2iXV7Qp6Ph8pUHHc-jNYoiQXPP34mR42vnnGh3fvMW ZrscYUqdHvbZ9vk1jk4CIL3LaZ8c9fLVb2pwqFQNBt5BJ8O8agRPkjoPNLAQHNwByT2Au-Mu1gu14wcPE2MFyOcbOzEMKu7PGug-m-tvIn3kcoGPD30RxqI HpcFamqCbVc1P5EhLrY-JeT6_2Lf3Pfz3D0LxDZBCDI8xR-8cbUeE2ezUjZC0pbXIn7MU-bQjIXw7rI76ENUrdkMq2R5FQCje-nnyPs8OgLGFwPWnPfQ3okpd wxc9cz4ZajQD1tnyCI43LHRmwDDTFfDdco47v56zzmhemp8rukxqejeSMN_-C0_OOIW7XQIlqnsBLQiUEsINs05uBQOT9CXneqBOSnWVGjyo8SPjkjnJA_W tkmdHQ
Accept	application/json

BODY

number	65984465 Numero de referencia
origin_bank	Banco de origen Banco de origen
destiny_bank	Banco destino Banco destino

Figura 69. API Notificar transferencia método POST

Fuente: Elaboración Propia

```
{
  "meta": {
    "code": 305,
    "message": "Mail send"
  }
}
```

Figura 70. Respuesta API notificar transferencia método POST

Fuente: Elaboración Propia

Apéndice I: Prueba gratuita

I.1 Activar prueba gratuita

POST Activate Free Trial

http://localhost/api-tikket/public/api/v1/plans/activate_free_trial?activate=1

Activar una prueba gratuita

HEADERS

Authorization	Bearer [illegible]
Accept	application/json

PARAMS

activate	1

BODY

plan	3

Figura 71. API Activar prueba gratuita método POST

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 160
  },
  "data": {
    "message": "The user was activated for a free trial"
  }
}
```


Figura 72. Respuesta API activar prueba gratuita método POST

Fuente: Elaboración Propia

Apéndice J: Plantillas

J.1 Consultar plantillas

GET Index Shorter

http://localhost/api-tikket/public/api/v1/teams/262/shorters?all=true

Consultar plantilla

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6ImU3NDc3NGI5NjNmYTgxYWE2ZGI4ZjkwYWI3Y2EyMWRmODhlNmRiYWRmZGE5NmU0ZDYx YTI5MVYiNWY4YmRiMjNmZGQ1NGUzNjMxODFjNGRmIn0.eyJhdWQiOiIxIiwianRpIjoiZTc0Nzc0Yjk2M2ZhODFhYTZkYjhmOTBhYjdjYTlxZGY4OGI2ZGJ hZGZkYTk2ZTRkNjFhMjkxZjU1ZjhiZGUyM2ZkZDU0ZTM2MzE4MWM0ZGYiLCJpYXQiOjE1MjE0MTMzOTAsIm5iZiI6MTUyMTQxMzM5MCwiZXhwIjoxNTIx ODQ1MzkwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIiI9.IFOA2qh9PjX6IosPuyhKXIGh7ossjkyK85x2oUDnAzdwXBLBDiihV09GCDdcSograArHhRumav6Nh2 RistOIdV1V8PSuWkE9KmydY5ATi3mvEN_zLakICkUcC_TOdv85QdR3ZHenB5fjvLpAcoi_Pbtg130CiYiS1QcKOU-QEFMNhjWPMWvZZz0sg6ZCj4MF60oY H4ELFoH4hwpYjuYj1i_siB925UOCtIGVQnCrUPvsVR0InDZiFau8fBAAIydvMkwTCpTC5Mt74_LBU2lxV7Qp6PhBoUHHc-jNYoiQXPP34mR42vnnGh3fvMW ZrscYUqdHvbZ9vk1jk4CIL3LaZ8c9fLVp2pwdFQNBtSBj808agRPkjoPNLAQHNwByT2Au-MuIgu14wcPE2MFyOcbOzEMKu7PGug-m-tvln3kcoGPD30Rxqi HpcFamqCbVc1P5EhLrY-jeT6jZLf3FfcSD0LxDZBCDi8xR-8dpUeE2ezUjZC0pbXin7MU-pQjXw7rI76ENUrdkMq2R5FQCje-nnyPs8OgLGFwPWhPfQ3okpd wxc9cz4ZajQD1tnyCI43LHRmwDDTFfDdcp47v56zzmhemp8rukxqejeSMN_-C0_OOIW7XQIiqhsBLQiUEsiNs05uBQOT9CKneqBOSnWVGjyo8SPjkjnJA_W tkmqHQ
Accept	application/json

PARAMS

all	true

Figura 73. API Consultar plantillas método GET

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 100,
    "message": "Retrieving shortcuts successfully"
  },
  "data": [
    {
      "id": 209,
      "name": "Plantilla de prueba2",
      "shorter": "pdp",
      "team_id": 262,
      "user_id": null,
      "public": 1,
      "shortermessages": [
        {
          "id": 243,
          "shorter_id": 209,
          "type": 0,
          "message": "Mensaje de prueba2",
          "created_at": "2018-03-18 23:01:38",
          "updated_at": "2018-03-18 23:06:48",
          "message_plain_text": "Mensaje de prueba2"
        }
      ]
    }
  ]
}
```


Figura 74. Respuesta API consultar plantillas método GET

Fuente: Elaboración Propia

J.2 Agregar plantilla

POST Post shorter

http://localhost:8000/api/public/v1/teams/262/shorters

Agregar plantilla

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6ImU3NDc3NGI5NjNmYTgxYWE2ZGI4ZjkwYWI3Y2EyMWRmODhiNmRlYWRmZGE5NmU0ZDYxYTI5MWY1NWVkYmRlMjNmZGQ1NGUzNjMxODFjNGRmMn0.eyJhdWQiOiIxIiwianRpIjoiZTc0NzE0Yjk2M2ZhODFhYTZkYjcmOTBhYjgyYTk2ZGY4OGI2ZGJhZGIxYTk1ZTRkNjFhMjkxZjU1ZjhlZGUyM2ZkZDU0ZTM2MzE4MWVlOZGYiLCJpYXQiOjE1MjE0MTMzOTAsImS1ZI6MTUyMTQxMzM5MCwiZXhwIjoxNTIxODQ1MzkwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.IFOAZqn9PJKi6IosPuyhKkGh7ozbjxyK8Sx2oUDhAzdxXBLBDJlrVO9GCDdzSbgraArHrRumaJ6Nn1RjstOlqV1VBPSuWKE9kmyzX5ATBmvENj_zLaxiCkUcC_TOqV85QdR3ZHenB5fjVLpAco_Pbsg130CIKS1QcKOJ-QEPMNnjWPiMWvZZs0zg6ZQJ4MF6OYH4ELFoH4hvoYjUYj1_gIB92EUOCbGVQnCrUPvsVR0 nDIJFau8fBAAlydvMkwTCoTCSMc74_LBUZkY7Qp6PnBpUHHc-jNYoiQXPP34mR4ZvnrGh3FMWZ1scVUcoHvbZ9vk1jx4CIL3LaZ8c9fLVo2bswqFQNBt3BBj8C8agRPxjoPNL4QHNvByT2AJ-Mugu14wcPE2MFyOcbOzEMku7PGug-m-svln3kzoGPD30Rkq1HacFamoCbVc1PSEhLrY-jeT6jZLY3F4s5D0LxDIBCD8xR-8coUeBZezUjZC0pbXin7MU-oQjlXw7nT66NUrckMaZR5FQCje-nnyPz8OgLGPwPWnPfQ3oXpdwxc9ozz4ZajQD1tnyCl43LHRmwDDTPfDdco47vS6zzmhemp8ruKxoejeBMN_-C0_OOW7XQIlchtzBLQ-UEzNs05uBQOT9CXreqBOEnWVGjyo8SPjKyrjA_WtkmqHQ
Accept	application/json

BODY

name	Plantilla de prueba Nombre de la plantilla
shorter	pdp Shortcut
public	1 Para el equipo lo privado

Figura 75. API Agregar plantilla método POST

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 101,
    "message": "The shorter was successfully inserted"
  },
  "data": {
    "name": "Plantilla de prueba",
    "team_id": "262",
    "shorter": "pdp",
    "id": 209,
    "public": 1,
    "shortermessages": []
  }
}
```


Figura 76. API Agregar plantilla método POST

Fuente: Elaboración Propia

J.3 Editar plantilla

PUT Update shortcut

http://localhost/api-tikket/public/api/v1/teams/1/shorters/168

Modificar plantilla

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6ImMyMzIwZWNlY2NmYWU4OWY5NzUyMzZhMDI1ZDQ4OWM5NmNkOTE2N2RiZmM4ODM3MTQ3ZjE3OGZlNTRkZmI1OGJmYmQ1YjBhZTU2NjgxN2EyIn0.eyJhdWQiOiIxIiwianRpIjoiMjBiY2VjY2ZhZTg5ZjA3NTIzNmEwMjVkNDg5YzkzZjQ3MTY3ZGJmYzg4MzcxNDdmMTc4ZmI1NGRmYjU4YmZlZDM3MTY2ODE3YTliLCJpYXQiOjE1MTU1OTQ4MDYsImE3ICI6MTUxNTU5NDgwNiwiZXhwIjoxNTE1MDI2ODA2LCJzdWIiOiIxIiwic2NvcGVzIjpbIioiXX0.efB2UdiP6hupM2EL_vmtbbkm5YzAbFM-3kt6dWo9vPXOPsaAFIB4782vk1fX9pvGqmZppa1N2Nlbo9PSEaDpMNWPLeAH5MKLTmhxqPoyuM64zZa_YauGE7Ovary7xw05N3ytjsZEBJUy-5BV-wIjzsPlckM6GvaVB_qcE03Tt75D6fBpWT-vR3ccK_Wr9rjbcUJ2JOqA9EcPAMYlz3PeoTrjUW3cH3d5BnpIuS6lTo8kwynPsPnvfRuvcwGPTHP6dlWDpcBPcknaE-BVnpEYxRwrfCGrNyAcO4EKb-L4IAD6euc5NzxIZfc2XxoV2fFRbvj3RTGPHLZMVHbMYyQ_cGlgneeofgPHWo45jXoyo-AnxHlWvTVnGN5XjaTOpc0L5IuBjYrj99n3crTvjtxKlkTojT0urBsPILcs2KufNDtqGAS89Y4ftaégXrEua6lTaKlCVg-2cM1P6lM6OrH-WRYXOG76hTlTAgyWXjEXdjUogT_uMXi4yo31BjZZpOj2MczzctSMx6oP64luwj66m2TcN9oUmjnKPeoEF4Z51jZI_9YLGuRWl4W0t-FowVjU7goy0Ct__NeOtW4-vlEKyvhoWLHVuHnrBsjTakhU_OMLW3ljwPf0lMcqxfoIup0uZuh1vfaGAajLoy11n_Kg9mufVaKFk
Accept	application/json
Content-Type	application/x-www-form-urlencoded

BODY

name	Saludo Inicial
shorter	Saludo4
public	1

Figura 77. API Modificar plantilla método PUT

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 184,
    "message": "Shortcut successfully updated"
  },
  "data": {
    "id": 269,
    "name": "Plantilla de prueba2",
    "shorter": "pdp",
    "team_id": "262",
    "user_id": null,
    "public": 1,
    "shortermessages": []
  }
}
```


Figura 78. Respuesta API modificar plantilla método PUT

Fuente: Elaboración Propia

J.4 Eliminar plantilla

DELETE Delete shortcut

http://localhost/api-tikket/public/api/v1/shorters/209

Eliminar plantilla

HEADERS

Authorization	Bearer [illegible]
Accept	application/json

Figura 79. API Eliminar plantilla método DELETE

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 199,
    "message": "Shortcut successfully deleted"
  },
  "data": {
    "id": 209,
    "name": "Plantilla de prueba2",
    "shorter": "pdp",
    "team_id": 262,
    "user_id": null
  }
}
```


Figura 80. Respuesta API eliminar plantilla método DELETE

Fuente: Elaboración Propia

J.5 Agregar mensaje en plantilla

POST Post shorter message

http://localhost/api-crm/public/api/v1/shortermessage

Agregar mensaje a una plantilla

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6ImY5ZjgzYjExOWUxNzVlODZlODM5ZTZiZTVkZDlwMGY0ZmM1NGIxNzIzMTBmNDEwYTFkYzli ZThiYzQzN2YyN2MwZDIlMTQ1MDVlMTJhZDBlIn0.eyJhdWQiOiIxIiwianRpIjoiZjlmODhlMTE5ZTE3NWU4NmU4MzlINmVlNWRkMjAwZjRmYzU0YjE3MjM xMGY0MTBhMWRjOWJlOGVjNDM3ZjI3YzBkOWUxNDUwNWIxMmFkMGIiLCJpYXQiOjE1MDIxMzk4NTksIm5iZiI6MTUwMjEzOTg1OSwiZXhwIjoxNTAyN TcxODU5LCJzdWIiOiI0Iiwic2NvcGVzIjpbIlo3XX0.rDD4pTo5y83guuTdwKCuIbmyO_srAQ0_BFLkBoy91yXcz2WbnB1vw4uIhwAeRrG0wBPtUy_eB315H2oB 7O4_Ei9fnH9l0s3DCvs-8ck1ueagk6iGLjZ-0hDwtX2F0GoE6ADM-wbkcRtCDCiyVR-HoQo85jg14k7Pt031GjcggFmTZaHCrDLNL5xOZfn7fDnXidi95mHtn mwrHGb7GZiBQZF1U50ozxL2wctZwFCXnYOc3xW1fCG7Y1zPuMMEV-BjH-M9yE0NzGeZAXHQp28ZgUDWjW6PPFctw-jcyMKtuHYzQWaxiX6aO4vyvzi3e kW1p2QhgqdWIhN4sKmVNL1E0NcwvGjuHj7cGx3cy1vYNh2AntLBkRzuecg9C7VWY5dBaYgvRDE52IPZZyw--azNIktpKjMhEDqcaDA0Lcnz6GWjn0V5owi xhdCS-H2BIb2Uda34WbvP36yGLhHcxOjXMaHRrcC8NmdCf6D4l66Wh2AeMIVuf5fHKRBzH7evg0OMIGX0oScWj-vapcrVi4ssLj3uTIB524_0ILg5-tCkVvsK gNpWw1gNnGg6x_oMf1IFlntopG7GMev0fctk5xmz5cc6Ea1nHGNRxCBV3zrspa6B1jWA4vyczk8MIkeATPtoGuSaa5jYwCUt9e8oWuvknHy55pl_eT4a3L_ ED4uQ8
Accept	application/json

BODY

shorter_id	2
message	Hola pedazo de trolazo dios mio

Figura 81. API Agregar mensaje en plantilla método POST

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 186,
    "message": "Shorter message successfully stored"
  },
  "data": {
    "id": 209,
    "name": "Plantilla de prueba2",
    "shorter": "pdp",
    "team_id": 202,
    "user_id": null,
    "shorter_message": {
      "shorter_id": 209,
      "type": "0",
      "message": "Mensaje de prueba",
      "updated_at": "2018-03-18 23:01:38",
      "created_at": "2018-03-18 23:01:38",
      "id": 243
    }
  }
}
```


Figura 82. Respuesta API agregar mensaje en plantilla método POST

Fuente: Elaboración Propia

J.6 Modificar mensaje en plantilla

PUT Update shortcut message

http://localhost/api-dvkes/public/api/v1/shorters/209/shortermessages/243

Modificar un mensaje de una plantilla

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6ImU3NDc3NGI5NjNmMTgxYWE2ZGI4ZjkwYWI3Y2EyMWRmODhiNmRlYWRmZGE5NmU0ZDYx YTI5MWY1NWY4YmRlMjNmZGQ1NGUzNjMbODFjNGRmIn0.eyJhdWQiOiIxIiwianRpIjoiZTc0NzdlYjk2M2ZhODFkYTZkYjhmOTBhYjdjYTkzZGY4OGI2ZGj hZGZkYTk2ZTRkNjFhMjkxZjU1IiwiaWF0IjoxNjU2MzE4MWM0ZGYiLCJqYXQiOjE1MjE3MTMzOTAsImSiZiI6MTUyMTQxMzM5MCwiZXhwIjoxNTIx ODQ1MzkwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyJvaWlI19.PQA2qm9PjNI6losPuyhKXiGh7ossjxyK8Sx2ouDhAzowXBLBDlhVC9GCDcc3ograArHnRumav6NH2 RlstOlcV1VBP5uWKE9KmydY$AT3mvEN_zLaKiCkUcC_TOqv85QcR3IHenBEfjVLpAcoi_Pbtgf3OCIVI51QckOU-QEFMNnjWPMWVZZz0zg6ZCJ4MF6OoY H4ELPoH4hvoYjuYj1i_zIB9ISUOCtIGVQhCrUPvzVR0hDZiRaU9fBAAIydVMkwTCpTCSMt74_LBU2iKV7Qp6Fh3oUHHo-jNYoQXPP34mR4ZvnnGn3MMW ZrscYUcoHvoI9vK1jx4CIL3LaZ0c9fLYa2owqFQNBt5BJ03agRPKjoPNLAQHNwByT2Au-Muigu14wcPE2MFyOctrOzEMKuTPGug-m-tMn3kcoGPD30Rxqi HocFamoCbVc1P5EhLrY-jeT6jZLf3FfcSD0LxDZBCDBxR-8dpUaEIezUjZC0bbXin7MU-oQjNw7ri76ENUrokMo2R5FQCje-nnyPs8OgLGFwPWhPfQ3oKod wxc9tz4ZajQD1tnyCi43LHRmwDDTFfDdcs47i86zzmHemoBruKiqejeSMN_-C0_DOiWTkQIlcHsBLQ_uEsiNs05LBQOT3CYneoB03hWVGjyo89PjkjrJA_W tkmqHQ
Accept	application/json
Content-Type	application/x-www-form-urlencoded

BODY

message	Mensaje de prueba2 Mensaje

Figura 83. API Modificar mensaje en plantilla método PUT

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 188,
    "message": "Shortcut message successfully updated"
  },
  "data": {
    "id": 209,
    "name": "Plantilla de prueba2",
    "shorter": "pdp",
    "team_id": 202,
    "user_id": null,
    "shorter_message": {
      "id": 243,
      "shorter_id": 209,
      "type": 0,
      "message": "Mensaje de prueba2",
      "created_at": "2018-03-18 23:01:38",
      "updated_at": "2018-03-18 23:06:48"
    }
  }
}
```


Figura 84. Respuesta API modificar mensaje en plantilla método PUT

Fuente: Elaboración Propia

J.7 Eliminar mensaje en plantilla

DELETE Delete shortcut message

http://localhost/api-tikket/public/api/v1/shortermessages/243

Eliminar mensaje de una plantilla

HEADERS

Authorization	Bearer [illegible]
Accept	application/json

Figura 85. API Eliminar mensaje en plantilla método DELETE

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 191,
    "message": "Shortcut message successfully deleted"
  },
  "data": {
    "id": 243,
    "shorter_id": 209,
    "type": 0,
    "message": "Mensaje de prueba2",
    "created_at": "2018-03-18 23:01:38",
    "updated_at": "2018-03-18 23:06:48"
  }
}
```


Figura 86. Respuesta API eliminar mensaje en plantilla DELETE

Fuente: Elaboración Propia

Apéndice K: Mensaje automático

K.1 Consultar mensajes automáticos

GET Index AutomaticMessages

http://localhost/api-dixkel/public/api/v1/teams/262/automaticmessage

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNiMTUwNTgZYzE5MmRlMmY1NzlWZjQ0ZmQ0YjdlZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhdWQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTlyYgZlMTdmZWEyY2lkNTA1ODZjMTkyYWUyZjU3MjBmNDRmZDRlNZjmOTcyZGQ3MTVmYjQwOTFhYTYlLCJpYXQiOjE1MjEyMzYzODAsImNiZiI6MTUyMTIzNjM4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyJyIl19.tGtrX3dBjGH48BBjZInZ8xiQY4285FN2SHrYRohMjD3WxU6-1WMZP-OUChdGjjzGLF3hgyTL48oW8ULdWge4jK6clXLzDSj3ffjjYmucw6UIelcuvs1kQNuJ6-6HGguPrCyt7y6luoP1jnGrXPfc_R4EbYn1wfBoXPP5wjMxfhecbgRBRjoYsccaQIHxVQ5VG7ndckne5NxGxbgyd1ElVayjZoDs/GuAFPUHt_MMRLT5xWI6bQHflonXtzjjuEKFGLc54QWVnBeuHaa-Gkdobx0gBPCe82JaK9IZ8ndAkWMnTdaQ1TDV45Tef1dW5RfDYlcg4XjoQYDMolf6S5z015vpHxYrIG69IHn_HtWuUkF8aVKb6gLPRg_Y2AJCfRexcbjDvE43Q0LY1MOhjVL3RuCIWPiVaADlbAr5cfdLj9AOVjmDqpNzO7IMxJEMM_8VR09NOl8G2QukjYDe_xZeWwfoLLa8JcwITtgm0chsW4_DBrNzw74WMy3QG2FjGa5GTvEf4q763fb3aAfuvD-TtmORw_zH69ZWusxfaX79QZOCyqf5LjJgGpAiy3_81ghIy5aGtRPr4CeRPyK7DmcFin2gV_y8HWI3O6LNtvIht4tAtXmBa_2svHtTcWI2WTchBsDCfMeXodaHJ_OecPW4Oy1VI7olEq7tQ
Accept	application/json

Figura 87. API Consultar mensaje automático método GET

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 15,
    "message": "AutomaticMessages for user successfully recovered."
  },
  "data": [
    {
      "id": 970,
      "name": "Mensaje de prueba",
      "message": "Prueba",
      "message_type": 1,
      "username": "Infonative",
      "rrss_user_id": 597,
      "rrss_user_type": 2
    }
  ]
}
```


Figura 88. Respuesta API consultar mensaje automático método GET

Fuente: Elaboración Propia

K.2 Agregar mensajes automáticos

POST Store AutomaticMessages

http://localhost/api-tikket/public/api/v1/rrss_users/507/automaticmessage

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNkMTUwNTg2YzE5MmFlMmY1NzNlZjQ0ZmQ0YjdlZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhdWQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRkYTIyYjg2MTdmZWEyY2NkNTA1ODZjMTkyYWUyZjU3MjBmNDRmZDRlN2JmOTcyZGQ3MTVmYjQwOTRlYTliZjpXNQIOjE1MjEyMzYzODAsImBiZI6MTUyMTIzNjM4MCwiZXhwIjoxNTkxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3dSjGh46SBjZInZ8klQY4295FN25HrYRohMjD3WxU6-1WMZP-DuChdGjzGLF3hgyTL4BpWBULdWgeAJk6qXLtzDSj3-ffyjYznudw6UIeicuva1kQNU5-6HGguPrCyt7y5IunP1jnGrKPfc_R45bYh1wfBoXPP5wjMkfKeb9gRBRjoYscoaQIhkVQ5VG7ndzkne3NkGxogyc1ElVavjZb2dvGuAPPUHt_WMRLT5wVl6oQHflorkNxtjJuBKPBLcI4QVWnEeuhaa-GxdobvBgBROe8ZjaX9IZ8ndAkWMn7caQ1TDV4ETe1k2W5RrDXlpg4YjoQ1DMolf6SSz015wpHwYYZG69lHln_HsWuUkF6aVKb6gLPRg_Y2-AjCVRexjojOvE43Q0LY1MOnjVL8RuCIwPVaADlbA-8cfdLj9AOVjmDqpNzOTIMxjEMM_8VR0BNCI3G2QukjXDe_xZeVlwfoLLa8JowiTtgn0cfsW4_DBhNzwT4WMy3QlG2FjGa5GTvEf4oT63fs3aAfuvO-7tmORw_zH69Zwuzxfsv79OIOCyojF5LjgGzAwy3_9Tgm1y5aGtRPr4CePPjk7DmcRn1gV_y8HWi3O6LNbvlhc4tAtXmBa_2svHt7cWl2WTqnBbDCfMeXocaHl_DeqPV4Oy1Vl7qlEb7bQ
Accept	application/json

BODY

name	Mensaje de prueba 2 Nombre del mensaje automático
message	Prueba 2 Mensaje para el mensaje automático

Figura 89. API Agregar mensaje automático método POST

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 19,
    "message": "AutomaticMessages created successfully."
  },
  "data": {
    "name": "Mensaje de prueba 2",
    "message": "Prueba 2",
    "id": 871,
    "rrssuse": [
      {
        "id": 507,
        "type": 2,
        "username": null,
        "name": "Informative"
      }
    ]
  }
}
```


Figura 90. Respuesta API agregar mensaje automático método POST

Fuente: Elaboración Propia

K.3 Modificar mensajes automáticos

PUT Update AutomaticMessages

http://localhost/api-tikket/public/api/v1/rrss_users/507/automaticmessage/971

Actualiza un mensaje automatico en especifico

HEADERS

Authorization	Bearer [illegible]
Accept	application/json
Content-Type	application/x-www-form-urlencoded

BODY

name	Mensaje de prueba 3 Nombre del mensaje automático
message	Prueba 3 Mensaje para el mensaje automatico
team	262

Figura 91. API Modificar mensaje automático método PUT

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 25,
    "message": "Updated successfully."
  },
  "data": {
    "id": 971,
    "name": "Mensaje de prueba 3",
    "message": "Prueba 3",
    "message_type": 1,
    "rrssuser": [
      {
        "id": 507,
        "type": 2,
        "username": null,
        "name": "Infonative"
      }
    ]
  }
}
```


Figura 92. Respuesta API modificar mensaje automático método PUT

Fuente: Elaboración Propia

K.4 Eliminar mensajes automáticos

POST Destroy AutomaticMessages

http://localhost/api-tikket/public/api/v1/automaticmessage/destroy

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJ5UzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNiMTUwNTg2YzE5MmFlMmY1 NzIwZjQ0ZmQ0YjdiZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhdWQiOiIxIiwianRpIjoiOTBiZGUyNzNjZTQxMzVlODAxYTRhYTIyYjg2MTdmZWEyY2IxNTA1O DZjMTkyYWUyZjU3MjBmNDRmZDRiN2JmOTcyZGQ3MTVmYjQwOTFiYTYiLCJpYXQiOjE1MjEyMzYzODAsIm5iZiI6MTUyMTIzNjM4MCwiZXhwIjoxNTIkNj Y4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3oSjGh46SBjZInZ8kIQY4295FN25HrYRohMjD3WxU6-1WMZP-DUChoGjjzGLF3hgyTL4BpW3ULd WgeAjK6qXiLzDSj3rffyjYmudw6UIeIcuvs1KQNU5-6HGgvPrCyt7y5IuhP1jnGxKPfc_R4SbYh1wfBoXPP5wjMkfKeb9gRBRjoYscoaQlhkVQ5VG7ndckne5Nk Gxogyz1EIVavjZb2dvGuAFPUHt_WMRLT5wVI6oQHflorkNktjjuBKFGLcS4QVWhEeuhaa-Gkdobv0gBPOe82jaX9IZ8ndAkWMn7daQ1TZV45Te1k2W5RrD XIpg4XjoQ1DMoIf655z015wpHwYYZG69iHIn_HtWuUkF6aVKb6gLPRg_Y2-AjCVRexjoj0vE43Q0LY1M0hjVL8RuCZwPVaADIbA-5cfdLj9AOVjmDqpNzO7Z MxjEMM_8VR09NOi8G2QukjXDe_xZeVIwfoLLa8jowlTtgnl0chsW4_DBnNpwT4WMy3QG2FjGaSGTvEif-dT63fb3aAfuvO-7ltmORw_zH69ZwusxfsX79OZO CycjF5LjgGpAvy3_91ghl-y5aGtRPr4CeRPjk7DmcFln2gV_xSHWi3O6LNtyIhc4tAtXmBa_2svHtTcWi2wTchBbDCfMeXodaHI_OeqPV4Oy1VI7qIEq7tQ
Accept	application/json
Content-Type	application/json

BODY


```
{"data":
  [
    {
      "rrss_users_id": 507,
      "automatic_message_ids": [
        971
      ]
    }
  ]
}
```


Figura 93. API Eliminar mensaje automático método DELETE

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 30,
    "message": "Successfully deleted."
  },
  "data": {
    "data": {
      "name": "Mensaje de prueba 3",
      "message": "Prueba 3"
    },
    "rrssuser": [
      {
        "id": 507,
        "type": 2,
        "username": null,
        "name": "Infonative"
      }
    ]
  }
}
```


Figura 94. Respuesta API eliminar mensaje automático método DELETE

Fuente: Elaboración Propia

K.5 Consultar horario laboral

GET Index WorkSchedules

[illegible]

Consultar horario laboral

HEADERS

Authorization	Bearer [illegible]
Accept	application/json

Figura 95. API Consultar horario laboral método GET

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 213,
    "message": "Work Schedules retrieved successfully."
  },
  "data": {
    "id": 262,
    "users_id": 78,
    "name": "Test",
    "domain": "test",
    "photo": "https://s3.amazonaws.com/files.tikket.net/fixed/defaultuser.png",
    "created_at": "2018-01-19 17:49:06",
    "updated_at": "2018-01-19 17:49:06",
    "deleted_at": null,
    "workschedules": {
      "id": 363,
      "week_labor": "1111111",
      "ticket_close_hour": null,
      "start_hour": "08:00:00",
      "finish_hour": "16:00:00",
      "teams_id": 262,
      "created_at": "2018-03-17 06:52:56",
      "updated_at": "2018-03-17 07:06:10",
      "exceptionsdates": [
        {
          "id": 50,
          "work_schedules_id": 363,
          "date": "2018-03-24 00:00:00",
          "start_hour": "07:30:00",
          "finish_hour": "09:00:00",
          "created_at": "2018-03-17 07:21:25",
          "updated_at": "2018-03-17 07:21:25",
          "description": "why"
        }
      ]
    }
  }
}
```


Figura 96. Respuesta API agregar Consultar horario laboral método GET

Fuente: Elaboración Propia

K.6 Modificar horario laboral

Figura 97. API Modificar horario laboral método PUT

Fuente: Elaboración Propia

Figura 98. Respuesta API modificar horario laboral método PUT

Fuente: Elaboración Propia

K.7 Agregar día de excepción

Figura 99. API Agregar día de excepción método POST

Fuente: Elaboración Propia

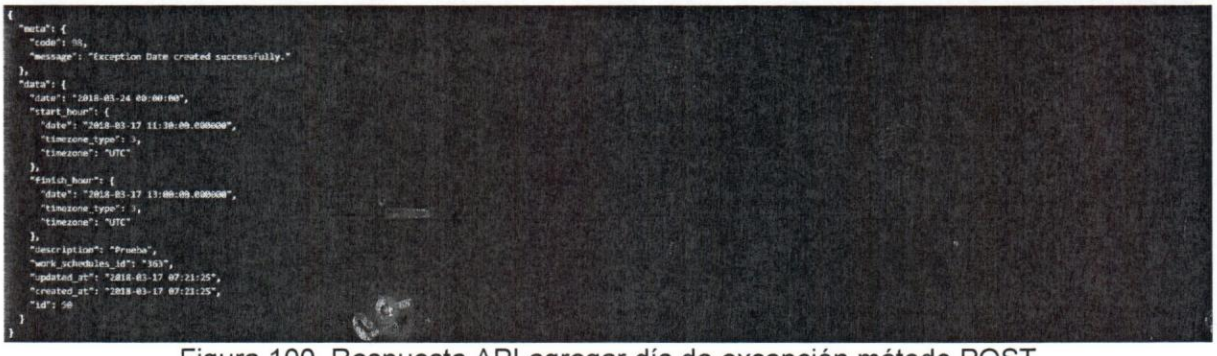


Figura 100. Respuesta API agregar día de excepción método POST

Fuente: Elaboración Propia

K.8 Modificar día de excepción

PUT Update ExceptionDate

http://localhost/api-ckkan/public/api/v1/exceptiondates/50

Modificar fecha de excepción

HEADERS

Authorization	Bearer [illegible]
Accept	application/json
Content-Type	application/x-www-form-urlencoded

BODY

date	2018-04-17 Fecha de excepción
start hour	11:13 Hora inicial
finish hour	15:15 Hora final
description	Ok Descripción

Figura 101. API Modificar día de excepción método PUT

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 100,
    "message": "Exception Date updated successfully."
  },
  "data": {
    "id": 50,
    "work_schedules_id": 363,
    "date": "2018-04-17 00:00:00",
    "start_hour": "11:13:00",
    "finish_hour": "15:15:00",
    "created_at": "2018-03-17 07:21:25",
    "updated_at": "2018-03-17 07:27:02",
    "description": "Ok"
  }
}
```


Figura 102. Respuesta API modificar día de excepción método PUT

Fuente: Elaboración Propia

K.9 Eliminar día de excepción

Figura 103. API Eliminar día de excepción método DELETE

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 102,
    "message": "Exception Date destory successfully."
  }
}
```


Figura 104. Respuesta API eliminar día de excepción método DELETE

Fuente: Elaboración Propia

Apéndice L: Caso interno

L.1 Agregar caso interno

POST Create Internal Ticket

http://localhost/api-tikket/public/api/v1/tickets/createInternalTicket

Crear ticket interno

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNiMTUwNTg2YzE5MmFlMmY1 NzIwZjQ0ZmQ0YjdlZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhdWQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTIyYjg2MTdmZWEyY2lxNTA1O DZjMTkyYWUyZjU3MjBmNDRmZDRlNZjmOTcyZGQ3MTVmYjQwOTFlYTYiLCJpYXQiOjE1MjEyMzYzODAsIm5iZiI6MTUyMTIzNjM4MCwiZXhwIjoxNTIxNj Y4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3dSjGh46SBjZInZ8kIQY4295FN25HrYRohMjD3WxU6-1WMZP-DUChdGjzGLF3hgyTL4BpW8ULd WgeAjK6qXiLzDSj3rffjjYmudw6UIeIcuvs1KQNU6-6HGguPrCyt7y5IunP1jnGxKFfc_R4SbYh1wfBoXPP5wjMkfKeb9gRBRjoYscoaQihkVQ5VG7ndckne5Nk Gxogyz1EIVavjZb2dvGuAFPUHt_WMRLT5wVI6oQHflorkNktjjuBKFGLc54QVWhEeuhaa-Gkdobv0gBPOe82JaX9lZ8ncIAkWMn7daQ1TZV45Te1k2W3RrD XIpg4XjoQ1DMoif655z01EwoHwYYZG69lHIn_HtWuUkF6aVKb6gLPRg_Y2-AjCVRexjoj0vE43Q0LY1M0hjVL8RuCZwPVaADIbA-5cfdLj9AOVjmDqpNzO7Z MxjEMM_8VR09NOi8G2QukjXDe_xZeVIwfoLLa8jowITtgnl0chsW4_DBnNpwT4WMy3QG2FjGaSGTvEif-dT63fb3aAfuvO-7ltmORw_zH69Zwuxfsx79OZO CycjF5LjjgGpAvy3_91ghIy5aGtRPr4CeRPjk7DmcFln2gV_x5HWI3O6LNtvihc4tAtXmBa_2svHtTcWI2wTqhBoDCfMeXodaHI_OeqPV4Oy1VI7qIEq7tQ
Accept	application/json
Content-Type	application/x-www-form-urlencoded

BODY

network	65966 Cliente va ha ser contactado
rrssUser	647 Canal digital con que se va a contactar
text	Mensaje de prueba Mensaje
type	0 Tipo de mensaje

Figura 105. API Agregar caso interno método POST

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 207,
    "message": "Internal case successfully created"
  },
  "data": {
    "tickets_id": 182152,
    "users_id": 78,
    "message": "Mensaje de prueba",
    "source": 3,
    "chat_id": "t_1000243805899905",
    "comment_id": "m_mid.$cAAdjRLgiX78obGtGOIiOyX93l3dz",
    "updated_at": "2018-03-18 22:06:20",
    "created_at": "2018-03-18 22:06:20",
    "id": 549382
  }
}
```

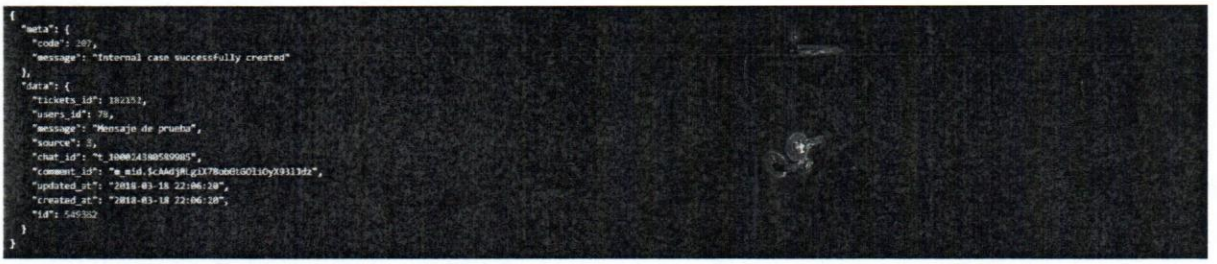



Figura 106. Respuesta API agregar caso interno método POST

Fuente: Elaboración Propia

L.2 Agregar caso interno de correo

POST Answer Ticket

http://localhost/api-tikket/public/api/v1/tickets/181460/messages?tikket=true

Responder ticket

HEADERS

Authorization	Bearer [illegible]
Accept	application/json

PARAMS

tikket	true

BODY

type	1 Tipo de mensaje
text	Generando caso interno con datos de correo Texto del mensaje
rrss_user	648 Correo específico

Figura 107. API Agregar caso interno de correo método POST

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 582,
    "message": "Ticket created successfully"
  }
}
```


Figura 108. Respuesta API agregar caso interno de correo método POST

Fuente: Elaboración Propia

Apéndice M: Escalamiento

M.1 Escalamiento de caso

PUT Transfer Ticket

http://localhost/api-tikket/public/api/v1/tickets/181460/transfer

Transferir ticket

HEADERS

Authorization	Bearer [illegible]
Accept	application/json

BODY

user_id	179 Operador a transferir el ticket

Figura 109. API Escalamiento de caso método POST

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 87,
    "message": "Ticket successfully transferred"
  },
  "data": {
    "id": 181460,
    "source": 4,
    "description": "Mensaje de prueba",
    "created_at": "2018-03-18 13:59:58",
    "attended_at": "2018-03-18 17:38:38",
    "transferred": "1",
    "users_id": 179,
    "rrss_users_id": 647,
    "updated_at": "2018-03-18 18:53:12",
    "user": {
      "id": 179,
      "team_selected": null,
      "email": "apibackendsocialgestcrm@gmail.com",
      "name": "Test",
      "username": "testy",
      "country": "VE",
      "city": null,
      "referral": 0,
      "timezone": "America/Caracas",
      "photo": "https://s3.amazonaws.com/files.tikket.net/fixed/defaultuser.png"
    },
    "network": null
  }
}
```


Figura 110. Respuesta API escalamiento de caso método POST

Fuente: Elaboración Propia

Apéndice N: Bitácora

N.1 Consultar bitácora

GET Index log

http://localhost/api-tikket/public/api/v1/teams/262/logs?offset=0

HEADERS

Accept	application/json
Authorization	Bearer [illegible]

PARAMS

offset	0

Figura 111. API Consultar bitácora método GET

Fuente: Elaboración Propia

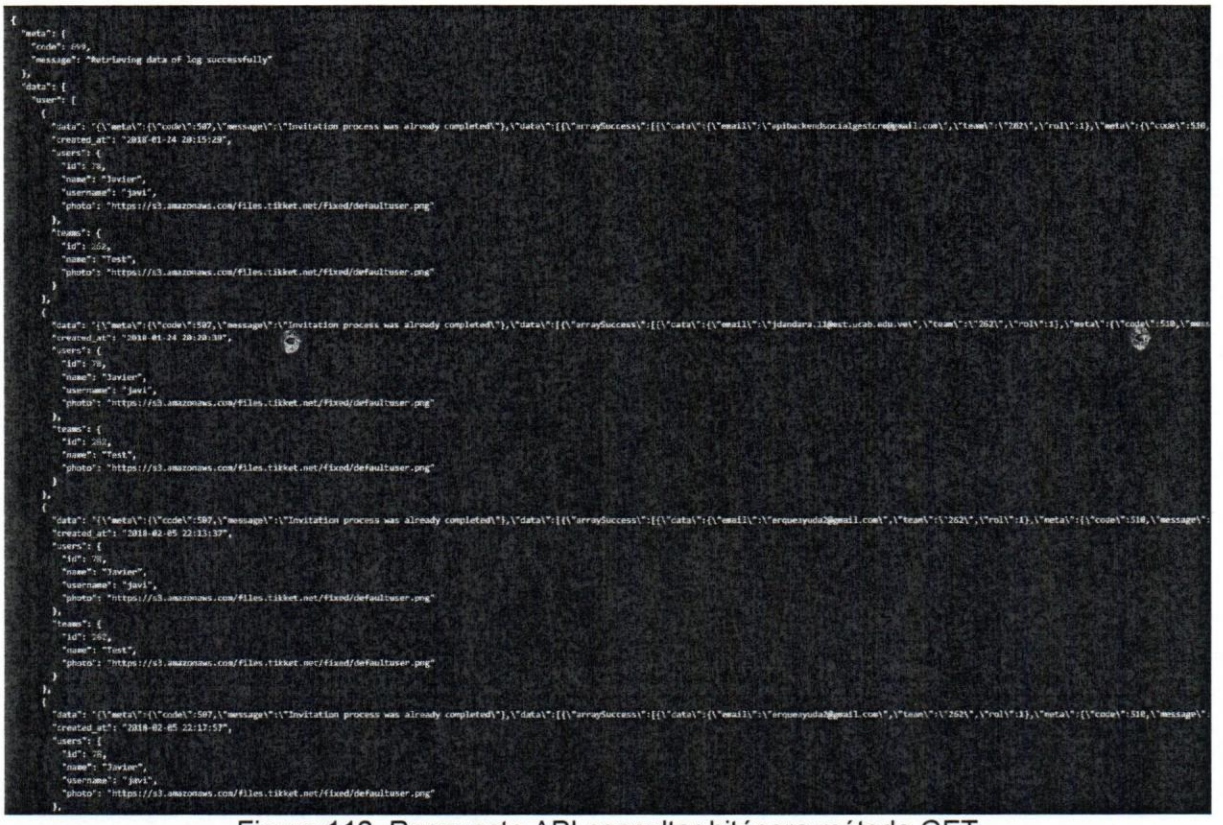


Figura 112. Respuesta API consultar bitácora método GET

Fuente: Elaboración Propia

Apéndice M: Cliente

M1. Consultar información de cliente

GET Get Client

http://localhost/api-crm/public/api/v1/tickets/8335/client

Consultar información del cliente

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjUzMWU1NDVkYTRlMDE3NGEzYTk2NDVhZWE4MGVhNDIzNWRmNmM5YjI2ZTdjOTE4MWVlZTM5OTEyN2EyZDVlYmNmMDdiZWY2YWIwZWRhNjkzIn0.eyJhdWQiOiIxIiwianRpIjoiNTMxZTU0NWRhNGUwMTc0YTNhOTY0NWFlYTgwZWE0MjM1ZGY2YzliMjZlN2M5MTgxZWVlMzk5MTI3YTjkNWViY2YwN2JlZjZhYjBlZGE2OTMiLCJpYXQiOjE1MDE4MjQwNzgsIm5iZiI6MTUwMTgyNDA3OCwiZXhwIjoxNTAyMjU2MDc4LCJzdWIiOiI0OSIsInNjb3BlcyI6WyIqIl19.k74LyyYKjsuFZL_2ysYWIe9ktPqr0um168fPIfV-73JORw2ES3rwHH13Mk3y3otYPqyAlGLqnxhKjO2P3A2UjWwUph0rjKbdlDi7XJ1TFaxNltNI_on8yEuCghPnGLjxjGQQC76AEg5v5YsLe9qyS0zN8xOtqI8iUR5-bDYk13WLJCZcqc4DibfIRXgjn-Ey2YbDQQ-jVpWpjeqfny0HFNmB0CjDHqP1llz3oIx9Dchnh3DgHw3-MYGGzOeliQdkKNuihbZYvVDN4ZHMFQblbvnlhVjCUoTL4LU8oKL9cZqnmBZre4et8N9a0c8DDyIF06BwAWlc8CxTTGS4kNrU4oWZMrCzvQQrOGZVX7bAQVB2k-Zl9qS0DpE-Rlk5l92KB44zXPihVFvHMhghiMCZ4E_tLiNM9FaGV-zM5RhqUZC7VqZfixY2bJyyuglq7NPbj2bacyltrgLtEwv6tpZzTNXvMM2dJjFJq9cM4jktewpoD-IRAzKkgN3AEHe__lyCqgcnCEJ_xJ2JD3a7UF1RiL3DUYnGdJcoAUgxsfNVlcomfEsrBcz9PXZEeVw9x40WVKOTZZU0l9X-xPMRdWkLeDtD332kLFpiWn_uZVq5BDjUovetg1ZfyvsMDRrWgS6__PtkUlv7sU7eXjdk74-M-wFakijFnoa6fplUWYcLjKY
Accept	application/json
Content-Type	application/x-www-form-urlencoded

Figura 113. API Consultar información de cliente método GET

Fuente: Elaboración Propia

Figura 114. Respuesta API consultar información de cliente método GET

Fuente: Elaboración Propia

M2. Agregar información de cliente

POST Store Client

http://localhost/api-tikket/public/api/v1/tickets/181460/client

Agregar información de cliente

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNiMTUwNTg2YzE5MmFlMmY1NzIwZjQ0ZmQ0YjdiZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhdWQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTIyYjg2MTdmZWEyY2IxNTA1ODZjMTkyYWUyZjU3MjBmNDRmZDRiN2JmOTcyZGQ3MTVmYjQwOTFlYTYiLCJpYXQiOjE1MjEyMzYzODAsIm5iZiI6MTUyMTIzNjM4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3dSjGn46SBjZInZ8kIQY429SFN25HrYRohMjD3WxU6-1WMZP-DUChdGjjzGLF3hgyTL4BpW8ULdWgeAjK6qXiLzDSj3rffyjYmudw6UIelcuvs1KQNU5-6HGguPrCyt7y5IuhP1JnGxKFfc_R45bYh1wfBoXPP5wjMkfKeb9gRBRJoYscoaQIhkVQ5VG7ndckne5NkGxogyz1ElVavjZb2dvGuAFPUHt_WMRLTSwVI6oQHflorkNktJjuBKFGLcS4QVWhEeuhaa-Gkdobv0gBPOe82JaX9lZ8nclAkWMn7daQ1TZV45Te1k2WSRrDXlpg4XjoQ1DMoIf655z015wpHwYYZG69IHIn_HtWuUkF6aVKb6gLPRg_Y2-AJCVRexjoJ0vE43Q0LY1M0hjVL8RuCZwPVIaADIbA-5cfdLj9AOVjmDqpNzO7ZMxJEMM_8VR09NOI8G2QukjXDe_xZeVIwfoLLa8JowITtgni0chsW4_DBnNpwT4WMy3QG2FjGaSGTvEif-dT63fb3aAfuvO-7itmORw_zH69Zwusxfsx79OZOCycjF5LjjgGpAvy3_91ghI-y5aGtRPr4CeRPjk7DmcFin2gV_x5HWi3O6LNtvIhc4tAtXmBa_2svHtTcWI2wTqhBbDCfMeXodaHI_OeqPV4Oy1VI7qIEq7tQ
Accept	application/json

BODY

name	Felipe Arturo Nombre del cliente
email	felipe.42i@gmail.com Correo del cliente

Figura 115. API Agregar información de cliente método POST

Fuente: Elaboración Propia


```
{
  "meta": {
    "code": 550,
    "message": "Client successful created"
  }
}
```


Figura 116. Respuesta API agregar información de cliente POST

Fuente: Elaboración Propia

Apéndice P: Perfil

P1. Consultar perfil

GET User Info

http://localhost/api-tikket/public/api/v1/teams/262/userinfo

Consultar perfil

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRiMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNiMTUwNTg2YzE5MmFlMmY1NzIwZjQ0ZmQ0YjdiZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhdWQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTlyYjg2MTdmZWEyY2IxNTA1ODZjMTkyYWUyZjU3MjBmNDRmZDRiN2JmOTcyZGQ3MTVmYjQwOTFlYTYiLCJpYXQiOjE1MjEyMzYzODAsIm5iZiI6MTUyMTIzNjM4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3dSjGh46SBjZInZ8kIQY429SFN25HrYRohMjD3WxU6-1WMZP-DUChdGjjzGLF3hgyTL4BpW8ULdWgeAjK6qXiLzDSj3rffyjYmudw6UIelcuvs1KQNU5-6HGguPrCyt7y5IuhP1JnGxKFfc_R4SbYh1wfBoXPP5wjMkfKeb9gRBRJoYscoaQIhkVQ5VG7ndckne5NkGxogyz1ElVavjZb2dvGuAFPUHt_WMRLTSwVI6oQHflorkNktJjuBKFGLcS4QVWhEeuhaa-Gkdobv0gBPOe82JaX9IZ8nclAkWMn7daQ1TZV45Te1k2WSRrDXlpg4XjoQ1DMoif655z015wpHwYYZG69IHIn_HtWuUkF6aVKb6gLPRg_Y2-AJCVRexjoJ0vE43Q0LY1M0hjVL8RuCZwPVIaADIbA-5cfdLj9AOVjmDqpNzO7ZMxJEMM_8VR09NOI8G2QukjXDe_xZeVIwfoLLa8JowITtgni0chsW4_DBnNpwT4WMy3QG2FjGaSGTvEif-dT63fb3aAfuvO-7itmORw_zH69Zwusxfsx79OZOCycjF5LJjgGpAvy3_91ghI-y5aGtRPr4CeRPjk7DmcFln2gV_x5HWi3O6LNtvIhc4tAtXmBa_2svHtTcWl2wTqhBbDCfMeXodaHI_QeqPV4Oy1Vl7qIEq7tQ
Accept	application/json

Figura 117. API Consultar perfil método GET

Fuente: Elaboración Propia

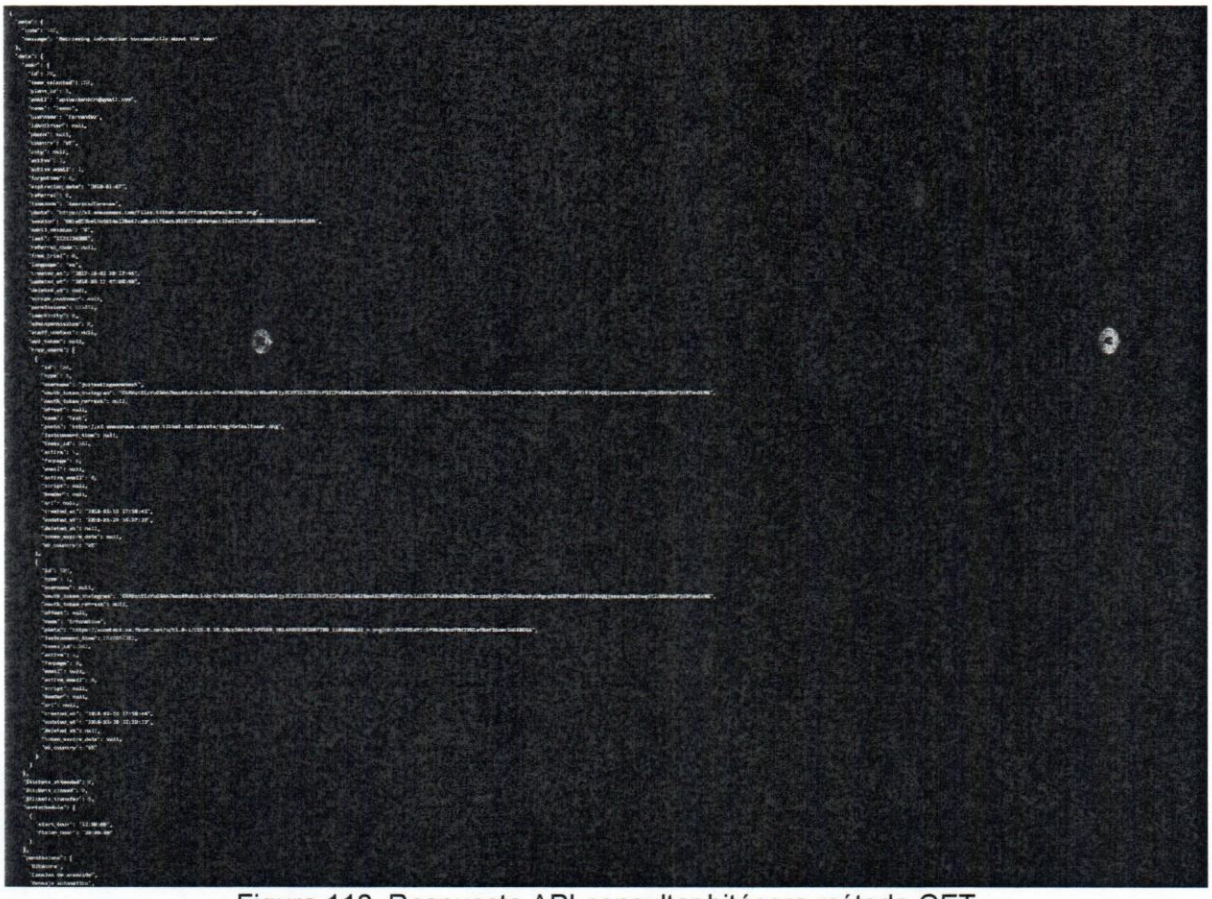


Figura 118. Respuesta API consultar bitácora método GET

Fuente: Elaboración Propia

P2. Editar perfil

PUT Update user info

http://localhost/api-tikket/public/api/v1/userinfo

Modificar perfil

HEADERS

Authorization	Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImp0aSI6IjkwZWRlMjczY2U0MTM1ZTgwMWE0YWEyMmI4NjE3ZmVhMmNiMTUwNTg2YzE5MmFiMmY1NzIwZjQ0ZmQ0YjdiZjk3MmRkNzE1ZmI0MDkxZWE2In0.eyJhdWQiOiIxIiwianRpIjoiOTBlZGUyNzNjZTQxMzVlODAxYTRhYTIyYjg2MTdmZWEyY2IxNTA1ODZjMTkyYWJiZjU3MjBmNDRmZDRiN2ZjmOTcyZGQ3MTVmYjQw[illegible]YTYiLCJpYXQiOjE1MjEyMzYzODAsIm5iZiI6MTUyMTIzNjM4MCwiZXhwIjoxNTIxNjY4MzgwLCJzdWIiOiI3OCIsInNjb3BlcyI6WyIqIl19.tGtnX3d5jGh[illegible]8kIQY4295FN25HrYRohMjD3WxU6-1WMZP-DUChdGjzGLF3hgyTL4BpW8ULdWgeAjK6qXiLzDSj3rffyJYmucw6UIeicuvs1KQNU5-6HGguPrCyt7y[illegible]jnGxKPfc_R45pYh1wfBoXPP5wjMkfKeb9gRBRjoYzcoaQIhkVQ5VG7ndckne5NkGxogyz1BiVavjZb2dvGuAFPUHt_WMRLTSwVI6oQHflorxNktjJuBKFGLc34QVWhEeuhaa-Gxdobv0gBPOe82jaX9IZ3nclAkWMn7daQ1TZV45Te1k2WSRrDXIbg4XjoQ1DMolf655z015wpHwYYZG69IHln_HtWuUkF6aVKb6gLPRg_Y2-AjCVRexjoj0vE43Q0LY1M0hjVL3RuCZwPViaADibA-5cfdLj9AOVjmDqpNzD7ZMxjEMM_3VR09NOI8G2QukjXDe_xZeVlwfoLLa8jowITtgnI0chsW4_DBnNpwT4WMy3QG2FjGaSGTvElf-cT63fb3aAfuvO-7itmORw_zH69ZwusxfSx79OZOCytjF5LjjgGdAvy3_91ghl-y5aGtRPr4CeRPjk7DmcFln2gV_xSHWI3O6LNtvlhc4tAtXmBa_2avHtTcWI2wTqnBbDCfMeXodaHI_OeqPV4Oy1VI7qIEq7tQ
Accept	application/json
Content-Type	application/x-www-form-urlencoded

BODY

name	javier Nombre
password_old	Contraseña antigua
password_new	Contraseña nueva
photo	Foto de perfil
username	javi Alias
country	VE País
timezone	America/Caracas Zona horaria
language	es Lenguaje

Figura 119. API Editar perfil método PUT

Fuente: Elaboración Propia

Figura 120. Respuesta API editar perfil método PUT

Fuente: Elaboración Propia