	UNIVERSIDAD FRANCISCO DE PAULA SANTANDER OCAÑA			
	Documento	Código	Fecha	Revisión
	FORMATO HOJA DE RESUMEN PARA TRABAJO DE GRADO	F-AC-DBL-007	08-07-2021	B
Dependencia	Aprobado	Pág.		
DIVISIÓN DE BIBLIOTECA	SUBDIRECTOR ACADEMICO	1(132)		

RESUMEN – TRABAJO DE GRADO

AUTORES	Juan Sebastián Duran Peña		
FACULTAD	Ingenierías		
PLAN DE ESTUDIOS	Ingeniería de Sistemas		
DIRECTOR	Fabian Ranulfo Cuesta Quintero		
TÍTULO DE LA TESIS	Definición de lineamientos de documentación y pruebas para los desarrollos de software de la empresa Smart Data y Automation		
TITULO EN INGLES	Definition of documentation and testing guidelines for the software developments of Smart Data and Automation.		
RESUMEN			
<p>El objetivo principal de este trabajo es poder realizar el correcto proceso de documentación y pruebas, generando una guía para los desarrolladores, lineamientos a seguir y un plan de pruebas para todos los desarrollos de software. Se realiza una investigación muy amplia en cuanto a normas y estándares internacionales, donde se evalúa cual se adapta mejor a la empresa para comprender y generar unos lineamientos más ajustados a la organización.</p>			
RESUMEN EN INGLES			
<p>The main objective of this work is to be able to perform the correct documentation and testing process, generating a guide for developers, guidelines to follow and a test plan for all software developments. A very extensive research is carried out in terms of international norms and standards, where it is evaluated which is best suited to the company to understand and generate guidelines more tailored to the organization.</p>			
PALABRAS CLAVES	Documentación, Diseño de plan de pruebas, lineamientos de documentación y pruebas.		
PALABRAS CLAVES EN INGLES	Documentation, test plan design, documentation and testing guidelines.		
CARACTERÍSTICAS			
PÁGINAS: 132	PLANOS:	ILUSTRACIONES:	CD-ROM:



**Definición de lineamientos de documentación y pruebas para los desarrollos de software de
la empresa Smart Data y Automation**

Juan Sebastián Duran Peña

Facultad de Ingenierías, Universidad Francisco de Paula Santander Ocaña

Ingeniería de Sistemas

Mgtr. Fabian Ranulfo Cuesta Quintero

26 Abril del 2023

Índice

	pag.
Resumen.....	7
Introducción	8
1. Definición de lineamientos de documentación y pruebas para los desarrollos de software de la empresa Smart Data y Automation	10
1.1 Descripción breve de la empresa.	10
1.1.1 Misión	11
1.1.2 Visión.....	11
1.1.3 Objetivos de la empresa	12
1.1.4 Descripción de la estructura organizacional	12
1.1.5 Descripción de la dependencia y/o proyecto al que fue asignado	13
1.2 Diagnóstico inicial de la dependencia asignada.....	13
1.2.1 Planteamiento del problema.....	15
1.3 Objetivos de la pasantía	16
1.3.1 General.....	16
1.3.2 Específicos	16
1.4 Descripción de las actividades a desarrollar en la misma.....	18
2. Enfoques referenciales	19
2.1 Enfoque conceptual.....	19
2.1.1 Documentación	19
2.1.2 Pruebas de software	31
2.1.3 Herramientas utilizadas para el cumplimiento de los objetivos.....	40
2.1.4 Antecedentes	44
2.2 Enfoque legal	45
2.2.1 Protección de datos personales o ley 1581 de 2012.....	45
2.2.2 Ley 23 de 1982 Sobre derechos de autor.	46
2.2.3 Normas para el desarrollo de software y pruebas	47
3. Informe de cumplimiento de trabajo.....	50
3.1 Descripción del trabajo	50
3.1.1 Recolección de información	51
3.1.2 Etapas de trabajo	52
3.1.3 Entendimiento empresarial	53
3.2 Definición de lineamientos y estándares de documentación	59

3.2.1 Documentación de API específicas	62
3.2.2 Documentación API genérica	66
3.2.3 Documentación de nueva funcionalidad	67
3.2.4 Documentación nuevo servicio	68
3.2.5 Documentación de nuevo módulo	76
3.3 Guía para realizar el proceso de pruebas	81
3.3.1 Desarrollos de tipo implementación de BOT.....	83
3.3.2 Desarrollos de APIs específicas.....	84
3.3.3 Desarrollos de APIs genérica.....	88
3.3.4 Desarrollos de nuevas funcionalidades	89
3.3.5 Corrección de error	90
3.3.6 Nuevo módulo.....	91
3.3.7 Nuevo paso agente	92
3.4 Casos de uso.....	95
3.4.1 Proceso creación de campañas.....	95
3.5 Diseño de plan de pruebas para validar los lineamientos	97
4. Diagnostico final.....	98
5. Conclusiones	99
6. Recomendaciones	100
Referencias.....	101
Apéndice A. Formatos	104
Apéndice B. Documentos lineamientos	118
Apéndice C. Casos de uso.....	123

Lista de figuras

	pag.
Figura 1. Descripción estructural de la empresa.....	12
Figura 2. Diagrama documentación externa.....	21
Figura 3. Diagrama documentación interna.....	22
Figura 4. Comentarios relativos.....	27
Figura 5. Código sin necesidad de comentarios.....	27
Figura 6. Etapas de trabajo.....	52
Figura 7. Diagrama contexto BOTAI.....	54
Figura 8. Diagrama proceso de negocio.....	56
Figura 9. Diagrama identificación de productos.....	57
Figura 10. Diagrama proceso de negocio SMD&A - aprobaciones por parte de cliente.....	58
Figura 11. Excel con información que se debe llenar para cada desarrollo – parte 1.....	60
Figura 12. Excel con información que se debe llenar para cada desarrollo – parte 2.....	61
Figura 13. Excel con información que se debe llenar para cada desarrollo – parte 3.....	62
Figura 14. Excel plantilla directorio de APIs donde se especifica toda la información.	63
Figura 15. Imagen documento estructura.....	64
Figura 16. Documento manual de configuración.....	66
Figura 17. Documento especificación de nuevo requerimiento.....	67
Figura 18. Plantilla especificación general.....	69
Figura 19. Documento plantilla especificación general.....	70
Figura 20. Plantilla especificación de requerimientos.....	71
Figura 21. Documento de diseño.....	72
Figura 22. Documento plan de pruebas.....	74
Figura 23. Plantilla documento de arquitectura.....	76
Figura 24. Documento manual de usuario.....	79
Figura 25. Proceso de pruebas dependiendo del tipo de desarrollo.....	81
Figura 26. Diagrama proceso de pruebas.....	82
Figura 27. Especificación Excel.....	83
Figura 28. Plantilla pruebas unitarias.....	84

Figura 29. Plantilla prueba de integración	85
Figura 30. Plantilla prueba extremo a extremo	86
Figura 31. Plantilla prueba interfaz de usuario	87
Figura 32. Plantilla prueba extremo a extremo	93
Figura 33. Plantilla prueba interfaz de usuario	94
Figura 34. Diagrama proceso creación de campañas.....	95

Lista de tablas

	pag.
Tabla 1. Matriz DOFA.....	13
Tabla 2. Estrategias DOFA.....	14
Tabla 3. Actividades a desarrollar	18

Resumen

El objetivo principal de este trabajo es poder realizar el correcto proceso de documentación y pruebas, dejar documentos que sirvan como guía para los desarrolladores, lineamientos a seguir y un plan de pruebas para todos los desarrollos que se realicen. Se realiza una investigación muy amplia en cuanto a normas y estándares internacionales, donde se evalúa cual se adapta mejor a la empresa para comprender y generar unos lineamientos más ajustados a la mecánica empresarial que permita generar unos lineamientos más ajustados a la mecánica empresarial.

Se realizó una investigación profunda sobre las normas que rigen la documentación del código, estructura, diseño de plantillas y demás componentes que nos permiten definir el proceso de las pruebas. Para esta etapa fue crucial el primer mes de la pasantía, ya que sobre estos elementos se trabajaría todos los demás objetivos.

Se implementa el plan de pruebas teniendo en cuenta la norma IEEE 829 que nos da una base para construir el mismo, donde se especifican las pruebas que se realizan al software donde este mismo sirve de ejemplo para próximos desarrollos.

Introducción

Desde hace muchos años, las empresas tecnológicas vienen creciendo de una forma exponencial y a gran escala, esto se da gracias a las nuevas alternativas que la tecnología nos brinda para poder facilitar nuestro día a día, nos ayuda con los procesos dentro de las empresas, almacenar la información, tenerla disponible en cualquier momento y en cualquier hora.

Gracias a esto también se debe tener claridad en los procesos que se realizan dentro de la empresa, esto con el fin de poder entenderlos, y para esto se debe documentar toda la información que se pueda, con el fin de tener material sustentable de todas las funcionalidades que se tienen, los servicios que se prestan y cómo se operan, donde esto también sirve como un manual operativo dentro de la empresa. Por esto se debe tener en cuenta la documentación dentro de la empresa ya que esta nos permite el intercambio de información, lo que capacita a nuestros empleados a saber cómo funcionan los procesos y como pueden ser finalizados sin cometer errores.

El proceso de documentación es una ventaja porque hace posible el manejo adecuado y cuidadoso de los documentos necesarios para el normal desarrollo de las actividades dentro de la organización, también es fundamental en la explicación de los procesos o procedimientos que se realizan en la empresa de forma estandarizada, esto con el fin de minimizar errores y pérdida en los recursos.

Los beneficios de documentar son evitar la ambigüedad operativa, nos sirve como material de capacitación a nuevos empleados de la empresa, nos sirve para analizar y comparar versiones de los procesos, esto con el fin de mejorarlos, minimizar errores, fallas y costos.

1. Definición de lineamientos de documentación y pruebas para los desarrollos de software de la empresa Smart Data y Automation

1.1 Descripción breve de la empresa.

Smart Data & Automation es una empresa con más de 9 años de experiencia en el sector tecnológico que busca hacer más fácil los procesos de comunicación entre las empresas y sus clientes a través del uso de chatbots, Inteligencia Artificial, Machine Learning, Mediciones, Analytics, SMS masivo y Llamadas automáticas.

La empresa cuenta con más de 150 clientes distribuidos en 8 países diferentes. Dentro de los casos de uso implementados cuenta con:

- Gestión de cartera digital.
- Envío y pago de facturas.
- Atención al cliente.
- Soporte técnico.
- Preguntas frecuentes.
- Geolocalización - Ubicación sedes.
- Biometría.
- Encuestas de servicio.
- Gestión de recursos humanos.
- Actualización de datos.

- Autogestión de servicios.
- Campañas de comunicación masiva.
- Publicidad.
- Ventas y colocación de productos.
- Almacenamiento blockchain.
- Entre otros.

1.1.1 Misión

Smart Data & Automation es una empresa de tecnología especialista en Inteligencia Artificial enfocada en hacer más rápidos y eficientes los procesos de comunicación de las empresas con sus clientes y usuarios.

1.1.2 Visión

Contar con 20 clientes importantes en LATAM que nos generen al menos el 80% de nuestra meta de \$20M de dólares de ingresos con un EBITDA superior al 35% en el año 2.025 y con esto, convertirnos en la empresa exportadora de tecnología más relevante en Santander y top 5 en Colombia.

1.1.3 Objetivos de la empresa

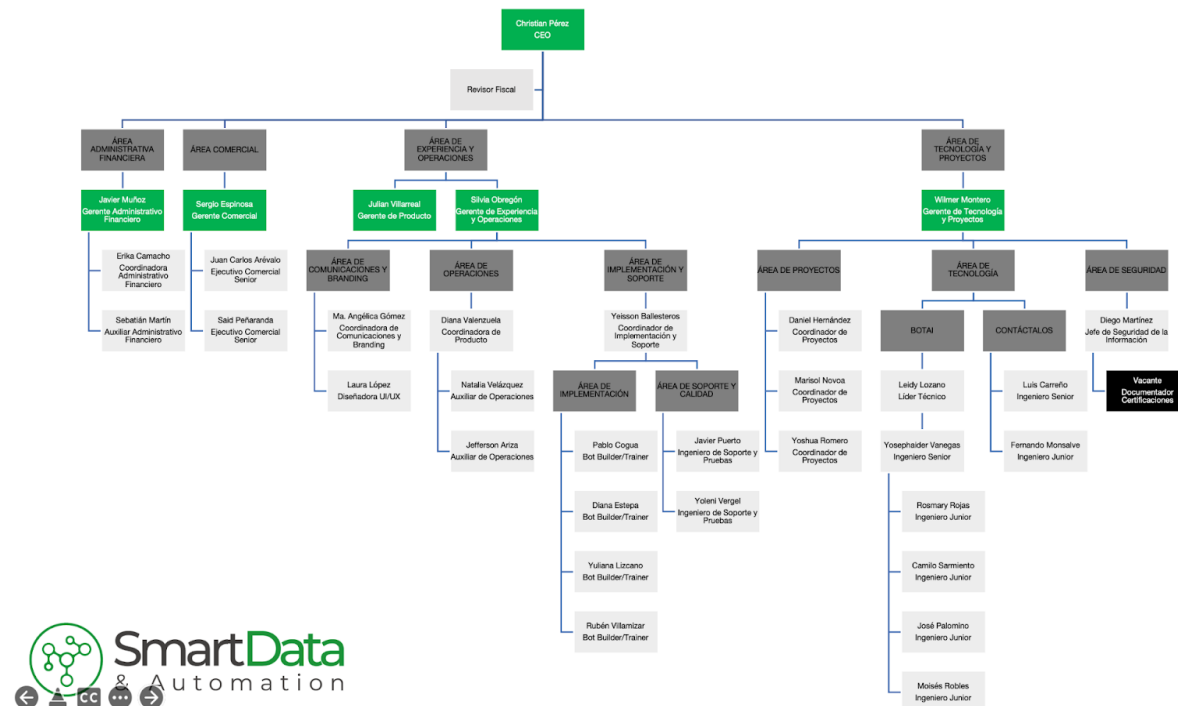
Hacer más inteligente, eficiente, cercana y segura la comunicación entre las empresas y sus clientes, por medio de chatbots y plataformas multicanal.

1.1.4 Descripción de la estructura organizacional

La empresa Smart Data & Automation se encuentra estructurada de la siguiente manera:

Figura 1

Descripción estructural de la empresa



Nota. Obtenido de (SmarData & Automation, 2022)



1.1.5 Descripción de la dependencia y/o proyecto al que fue asignado

La empresa Smart Data & Automatización cuenta con un área de tecnología que se encarga del desarrollo y mantenimiento de las plataformas BOTAI, Paso Agente y Contáctalos.

La plataforma BOTAI permite la administración de Chatbots, la plataforma Paso Agente permite la comunicación a través de agentes en vivo y la plataforma Contáctalos el envío de comunicaciones masivas.

Para la ejecución de la pasantía, se trabajará dentro del área de tecnología en la definición y estandarización de procesos de documentación y pruebas que garanticen la calidad de los productos realizados dentro del área y brindando soporte al área de desarrollo.

1.2 Diagnóstico inicial de la dependencia asignada

Tabla 1

Matriz DOFA

Debilidades	Oportunidades
<ul style="list-style-type: none"> ● La documentación que se tiene actualmente para orientar el personal en cuanto a los procesos de desarrollo en las diferentes áreas es muy poca o está en proceso de creación. ● El proyecto asignado cuenta con poca información documental y esto tiene un efecto negativo para el correcto desarrollo de las actividades. ● No se tiene estandarización de la documentación y los procesos que rige dentro del área de tecnología y esto causa reprocesos en las diferentes actividades. 	<ul style="list-style-type: none"> ● Aumento en el conocimiento del personal en cuanto al manejo de las normas ISO y su correcta documentación. ● Crecimiento en la variedad de los productos ofertados. ● Se pueden establecer los lineamientos para evitar malos entendidos en la formulación de requerimientos y uso de código limpio. ● Mejoramiento y agilización de procesos internos.

<ul style="list-style-type: none"> ● No hay una guía clara para realizar pruebas de software que permita revisar de forma completa todos los requerimientos a entregar al cliente para garantizar la calidad de los productos ofrecidos. 	<ul style="list-style-type: none"> ● Aseguramiento de calidad en productos ofrecidos
Fortalezas	Amenazas
<ul style="list-style-type: none"> ● Se cuenta con personal capacitado en diferentes áreas. ● Se cuenta con personal dispuesto a suplir fallas o brindar capacitaciones para corregir errores que se producen dentro de los procesos. ● Se dispone de herramientas que nos permiten desarrollar las actividades sin contratiempos. 	<ul style="list-style-type: none"> ● No existe personal encargado para que realice trabajos de documentación y verificación de plan de pruebas. ● Al no contar con unas evaluaciones postproducción, se pueden presentar fallas a los clientes, que afecta la confiabilidad de la empresa. ● No se cuenta con un documento que permita evaluar los resultados de los proyectos.

Nota. Elaboración Propia

Tabla 2

Estrategias DOFA

Estrategias (FO)	Estrategias (DO)
<ul style="list-style-type: none"> ● Brindar las herramientas tecnológicas y capacitaciones necesarias para el mejoramiento del desempeño y conocimiento del personal. ● Establecer una comunicación que permita entender todos los procesos para el mejoramiento de los mismos, donde la comunicación sea clara y precisa. 	<ul style="list-style-type: none"> ● Establecer un equipo capacitado en el tema para poder contextualizar y orientar a los trabajadores de las diferentes normas que existen dentro de la empresa y de las cuales se quieren establecer los lineamientos para su posterior estandarización. ● Asignar equipos de trabajo adecuados, incorporando nuevas tecnologías y metodologías, permitan realizar cambios y modificaciones a las especificaciones de los proyectos.
Estrategias (FA)	Estrategias (DA)
<ul style="list-style-type: none"> ● Aplicar estrategias para obtener una rápida adaptación a tecnologías innovadoras del mercado. ● Planificar métodos y generar estrategias de crecimiento en las relaciones laborales e interpersonales con el equipo de trabajo. ● Contratar personal capacitado en las áreas que se detecten fallas para así mismo obtener mejoras en calidad de los resultados. 	<ul style="list-style-type: none"> ● Asignación de personal adecuado en los diferentes proyectos, que permita la correcta gestión de cada uno de estos. ● Realizar constantes análisis a la competencia, para generar mejores prácticas y tener unos mejores resultados.

Nota. Elaboración Propia

1.2.1 Planteamiento del problema

Para la implementación y el desarrollo de nuevos proyectos de software, es necesario realizar un seguimiento continuo, estandarizado y documentado de los diferentes procesos que se realizan para poder tener un cumplimiento de los objetivos de forma clara, eficiente y con altos niveles de calidad.

Estos proyectos tienen la ventaja de mejorar las operaciones y generar mayores utilidades dentro de la empresa, pero todo esto depende de cómo se planeen desde un comienzo y de cómo se sigan lineamientos existentes para poder tener claras las ideas que se tienen y de las actividades finales que se quieren lograr.

Al no contar con unos lineamientos que ofrezcan a los trabajadores de la empresa una guía única para guiar los procesos de desarrollo, tener claro los documentos o artefactos resultantes de cada tarea del proceso y mantener una comunicación fluida y eficiente entre los actores del proceso, surgen retrasos en las actividades, retrabajo, incumplimiento de tiempos y disminución en la calidad de los productos entregados.

Adicionalmente, al no contar con la documentación completa y un estándar establecido para realizarla, el nuevo personal que ingresa a la empresa requiere de una curva de aprendizaje y entendimiento mayor al realizar el ejercicio de comprender e interpretar los desarrollos existentes o los nuevos requerimientos que sean asignados.

Por otro lado, al no contar con un proceso bien definido y riguroso de evaluación y pruebas de los desarrollos que se realizan, se generan riesgos que pueden afectar negativamente la organización al conseguir insatisfacción del cliente, retrabajo y degradación de los servicios prestados.

Por esta razón se busca estandarizar y establecer lineamientos comunes para la empresa en el área de desarrollo que se apliquen en la creación de todos los proyectos y permitan optimizar recursos, mejorar la calidad de los productos entregados y mantener clientes satisfechos. Así mismo, minimizar riesgos o pérdidas de información y lograr cumplir metas, crecer y obtener resultados de calidad amigables con la mayoría de los clientes.

1.3 Objetivos de la pasantía

1.3.1 General

Definir los lineamientos de documentación y pruebas de los desarrollos de software teniendo en cuenta las normas y estándares internacionales para contribuir con el mejoramiento de la calidad de los productos finales del área de desarrollo.

1.3.2 Específicos

- Definir los lineamientos y estándares de documentación que seguirá la empresa Smart Data & Automation durante el proceso de desarrollo de sus productos.

- Proponer una guía para realizar el proceso de pruebas que permita fomentar la entrega de productos de calidad a los clientes, garantizando la gestión y cumplimiento de los requerimientos post-producción.
- Validar los lineamientos de documentación y la guía para planear y ejecutar las pruebas propuestos a través de un caso de uso que permita brindar soporte al equipo de desarrollo.

1.4 Descripción de las actividades a desarrollar en la misma

Tabla 3

Actividades a desarrollar

Objetivo General	Objetivos Específicos	Actividades a desarrollar en la empresa para hacer posible el cumplimiento de los Obj. Específicos
<p>Definir los lineamientos de documentación y pruebas de los desarrollos de software teniendo en cuenta las normas y estándares internacionales para contribuir con el mejoramiento de la calidad de los productos finales del área de desarrollo.</p>	<p>Definir los lineamientos y estándares de documentación que seguirá la empresa Smart Data & Automation durante el proceso de desarrollo de sus productos.</p> <p>Proponer una guía para realizar el proceso de pruebas que permita fomentar la entrega de productos de calidad a los clientes, garantizando la gestión y cumplimiento de los requerimientos post-producción.</p> <p>Validar los lineamientos de documentación y la guía para planear y ejecutar las pruebas propuestas a través de un caso de uso que permita brindar soporte al equipo de desarrollo.</p>	<ul style="list-style-type: none"> ● Recolectar información por medio de reuniones virtuales para entender los procesos de negocio. ● Investigar sobre lineamientos de documentación de procesos de software y código limpio. ● Analizar la información recolectada. ● Definir los lineamientos que servirán de guía para nuevos proyectos. ● Registrar las actividades realizadas en el documento estipulado por la empresa. ● Definir alcance del plan de pruebas ● Determinar qué tipos de pruebas se implementarán dependiendo los resultados esperados de los proyectos. ● Seleccionar tipos y niveles de pruebas a implementar. ● Identificar herramientas a utilizar. ● Realizar pruebas que sean asignadas por parte del equipo de desarrollo. ● Definir y entender el proceso sobre el cual se realizará el caso de uso. ● Generar la documentación y artefactos requeridos siguiendo los lineamientos de documentación propuestos. ● Aplicar la guía propuesta para realizar las pruebas al caso de uso definido. ● Ajustar lineamientos de documentación y la guía de pruebas con base en resultados obtenidos.

Nota. Elaboración Propia.

2. Enfoques referenciales

2.1 Enfoque conceptual

Para el desarrollo de la propuesta de trabajo, se tuvieron en cuenta dos temas principales: Documentación de procesos de software y Pruebas de software.

2.1.1 Documentación

La documentación de software hace referencia a todo tipo de información que se genera para apoyar las tareas que componen el proceso de desarrollo de software, ayudar a dirigir los esfuerzos del equipo de desarrollo y entender la arquitectura y diseño de la aplicación a lo largo del tiempo. Así mismo, de cara al cliente puede ser toda aquella información que lo guía a la hora usar la aplicación una vez entregada.

La documentación de un programa empieza desde que se inicia el proceso de levantamiento de requerimientos y dependiendo de la metodología de desarrollo que se adapte, la documentación puede ser más o menos extensa y detallada.

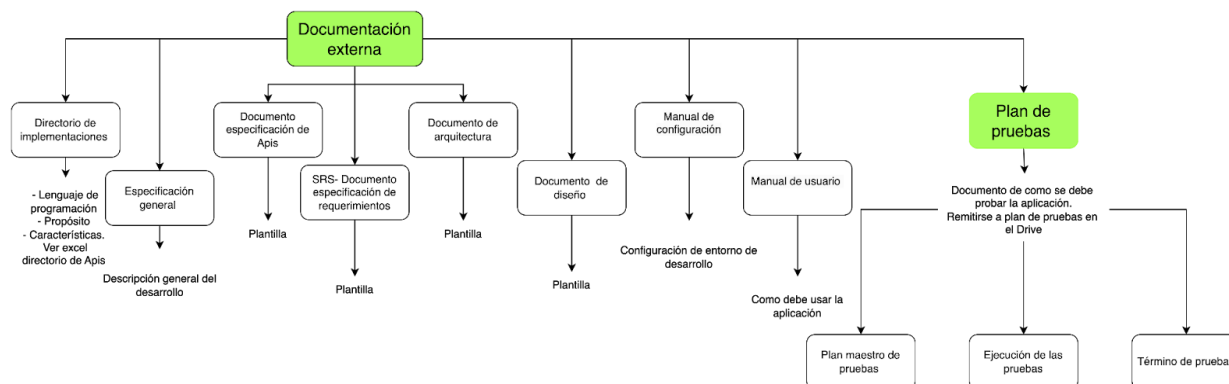
La realización de la documentación no termina con la entrega de la aplicación, pues durante el mantenimiento es necesario actualizarla para reflejar los cambios que se hayan tenido que realizar para crear nuevas funcionalidades.

La calidad del contenido de documentación debe presentar unas características muy particulares para que tenga cierta credibilidad como lo es la legibilidad, precisión, formato, accesibilidad y actualización.

La documentación resultante del proceso de desarrollo de software puede clasificarse en dos tipos: Documentación externa y Documentación interna. La documentación externa corresponde a todos los documentos resultantes de realizar el proceso de construcción de software desde el levantamiento de requerimientos hasta la entrega del producto y su posterior mantenimiento y actualización y la documentación interna corresponde a la documentación que se debe hacer al código fuente.

2.1.1.1 Documentación Externa. Se refiere a todos los documentos que describen la aplicación desarrollada y sus componentes desde la mayor cantidad de puntos de vista permitiendo que un programa no solo puede ser legible y ejecutable por un computador, sino también por cualquier programador o futuros empleados de la empresa soportando el trabajo desarrollado.

La documentación externa es supremamente relevante para entender qué se ha desarrollado, cómo se ha desarrollado, cómo debe funcionar y cómo se debe mantener, es por esto que, dentro de dicha documentación, se deben tener documentos que contenga información de las diferentes fases de desarrollo: levantamiento de requerimientos, definición de arquitectura y diseño, implementación, pruebas, despliegue, mantenimiento y uso.

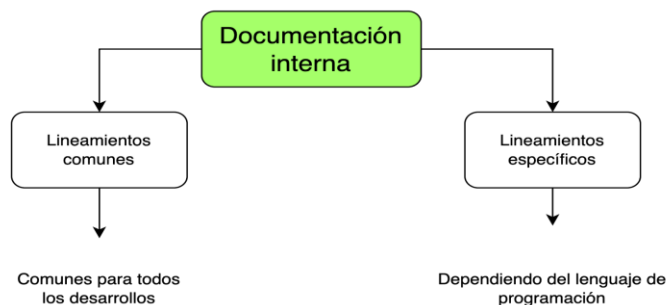
Figura 2*Diagrama documentación externa*

Nota. Elaboración Propia.

2.1.1.2 Documentación Interna. La documentación interna corresponde a la documentación que se hace al código fuente para que sea fácilmente entendible. Esto es de suma importancia al momento de capacitar nuevo personal y también dar soporte a los procesos que existen actualmente. Los estándares de documentación pueden dividirse en dos tipos: lineamientos comunes, que corresponden a los puntos que independiente el lenguaje de programación que se use se deben aplicar siempre y los lineamientos específicos que corresponden a las directrices que dependen del lenguaje de programación a emplear.

Figura 3

Diagrama documentación interna



Nota. Elaboración propia.

Existen 4 pasos importantes para obtener una documentación interna:

1. Identifica los procesos clave: Se deben establecer unas reglas básicas para determinar que se documenta, esto nos permite ser muy asertivos, ya que tendremos unos componentes importantes sobre los cuales se van a trabajar, donde también se deben establecer criterios de lo que se debe documentar, esto con el fin de no documentar cosas que no tengan mayor relevancia en el sistema.
2. Crear una plantilla estándar: Se puede crear una plantilla que puedan usar todos los trabajadores de la empresa, donde se tengan en cuenta los siguientes campos, una descripción del motivo para el cual existe el proceso o del porqué es importante, quienes son los factores claves en el proceso, y lo que necesitará el personal en cuanto al hardware para poder llevar a cabo el proceso.
3. Decidir dónde almacenar los procesos: La documentación interna debe ser accesible para todos los integrantes de la empresa, por lo cual se aconseja crear una carpeta donde se

encuentren todos los archivos o un espacio donde se pueden almacenar los registros.

También se debe tener en cuenta la información que se tiene, que sea clara y precisa para disminuir tiempos en los procesos que se van a realizar.

4. Reserva tiempo para hacer limpieza: Se debe realizar limpieza en cuanto a los documentos nuevos que entren, esto con el fin de eliminar documentos basura que no brindan conocimiento o no tienen importancia en esa sección o documentos que deban actualizarse, se debe establecer unos tiempos para realizar estas limpiezas como por ejemplo cada mes, cada trimestre o el tiempo que se escoja dentro de la empresa; de este modo se contribuirá a que el sistema conserve archivos valiosos y este en buen estado, esto con el fin de que todos hagan uso de los registros.

2.1.1.3 Lineamientos comunes. Son los lineamientos de documentación que se establecen para todos los desarrollos independientemente del tipo de aplicación a implementar. A continuación, se nombran las características comunes de las cuales se pueden generar directrices para poder estructurar los lineamientos de documentación:

2.1.1.3.1 Nombre del proyecto. Se recomiendan nombres fáciles de identificar al momento de realizar una búsqueda rápida en el menú del código, para así mismo traer la información que contenga el proyecto como por ejemplo “ProjectCampaigns, ProjectTemplates”, donde la segunda letra mayúscula indica que hay un espacio entre esas dos palabras o también se puede colocar la raya al piso “_”.

Lo que se busca es que cualquier miembro del equipo comprenda solo leyendo el título del proyecto que puede contener este mismo y minimizar los tiempos en búsqueda del código que se requiera.

2.1.1.3.2 Nombre de funciones. Se recomienda usar nombres que identifiquen el código; evitar abreviaciones como “hp” para hipotenusa o “FH” para fecha y hora, ya que esto no lo entenderá otro desarrollador.

Es de suma importancia utilizar nombres que se puedan pronunciar fácilmente como por ejemplo “sumFunction” para Función Suma. Teniendo en cuenta estos nombres a funciones o variables, cualquier persona o trabajador de la empresa comprenderá fácilmente de que se trata o que puede contener el código.

2.1.1.3.3 Comentarios de código. Comentarios muy acertados y descriptivos con la información que se maneja en el código.

- Se debe comentar la arquitectura general, vista de alto nivel del código.
- Utilización de funciones.
- Soluciones importantes, especialmente cuando no son inmediatamente obvias.
- Se debe denotar muy clara la información del cómo se usa y para qué. Todo el código no se puede estar comentando, por ese motivo, se debe construir un código limpio y entendible para que cualquier persona pueda entenderlo rápidamente sin necesidad de leer un comentario.

El comentario debe ser claro en qué parámetros pide, y que devuelve, esto se comenta si es gran dificultad entender el código; también se debe comentar la funcionalidad para la cual se creó, o también para brindar información sobre versiones, donde está ubicada alguna extensión y demás.

Ejemplo:

```
/**  
  
 * Devuelve x elevado a la potencia de n.  
  
 *  
 * @param {number} x El número a elevar.  
 * @param {number} n La potencia, debe ser un número natural.  
 * @return {number} x elevado a la potencia de n.  
  
 */  
  
function pow(x, n) {  
  
  ...  
  
}
```

Este comentario nos permite saber el propósito de la función y poder usarla de manera correcta.

También comentarios informativos como:

```
/**  
  
 * Esta clase crea una instancia de las clases de motores de...
```

* Registrado en la clase java.security.Security object.

* @autor Juan Perez

* @version 1.50, 04/14/2015

* @desde 1.4 */

- Se debe tener en cuenta el tipo de lenguaje en el cual se está desarrollando, ya que cada uno de ellos tiene una forma de iniciar los comentarios y de finalizarlos.
- Por ejemplo, en java y php, es con “//” para comentar una sola línea, ejemplo:

//esto es un comentario.

pero si queremos comentar varias líneas se hará de la siguiente manera /* */

/* Así se comentan varias

líneas en java sin necesidad de

estar comentando una por una */

- En Django y php también se usa así “# Esto es un comentario en Django.” Se comenta como un #
- En React: “/*Esto es un comentario*/”. Se comenta con /* */
- En Laravel: “{{-- Esto es un comentario --}}”. Se comenta con dos líneas al principio y dos líneas al final --

- En angular: “<! – Esto es un comentario –>”. Se comenta con signo de menor o igual, signo de exclamación y raya al medio, y se cierra con raya al medio y signo de mayor o igual. <!-- -->

Figura 4

Comentarios relativos

```
//COMENTARIOS RELATIVOS A LA FUNCIÓN DEL PROGRAMA
int main(){
    int n1,n2,n3,n4;
```

Nota. Elaboración propia.

- Los comentarios deben ser útiles, sino aportan algo importante al código, evitar realizarlos, como en este caso donde se ve claramente cada funcionalidad del código. Al usar nombres descriptivos el código no necesita comentarios adicionales.

Figura 5

Código sin necesidad de comentarios

```
1var cliente = servicioClientes.recuperarDatosCliente(id_cliente);
2if (cliente.estaActivo)
3{
4    cliente.UltimoAcceso=Date.Now;
5    servicioDatos.guardaCliente(cliente);
6}
```

Nota. Elaboración propia.

2.1.1.3.4 Nombre de variables. Para los nombres de las variables se aplicarán las siguientes reglas:

- Cada nombre de variable debe ser exclusivo; no se permiten duplicados.
- Las variables no pueden contener espacios.
- No debe utilizar caracteres especiales como por ejemplo %, &, \$...
- Se deben evitar los nombres de variable que terminan con un carácter de subrayado, ya que tales nombres pueden entrar en conflicto con los nombres de variable creados automáticamente por comandos y procedimientos.
- Las palabras reservadas no se pueden utilizar como nombres de variable. Las palabras reservadas son ALL, AND, BY, EQ, GE, GT, LE, LT, NE, NOT, OR, TO y WITH.
- Los nombres de variable se definen en inglés, se deben usar palabras que definan lo que se está almacenando en la variable y si son compuestas se separan a través de raya al piso (_).

2.1.1.3.5 Estilo y formato. Se recomienda dejar espacios entre diferentes métodos o acciones dentro de una función, con el fin de que se pueda observar mejor y mejorar la legibilidad del código, puesto que a simple vista se puede observar las diferentes partes de la clase o función.

También se deben tener en cuenta que las funciones a invocar deben estar verticalmente próximas, y es posible la que invoca por encima de la otra; esto para evitar declarar las funciones

o variables al principio, al medio o al final del código, obstruyendo así una construcción limpia y entendible del código y muy tediosa para muchos desarrolladores que hagan uso de este código. Se debe tener en cuenta el formato horizontal del código teniendo en cuenta que actualmente las pantallas son más grandes y tiene mejor resolución se recomienda que 100 a 120 podría estar bien, pero no más, esto con el fin de que todos lo que hagan uso de código, puedan ver claramente en sus pantallas, sin sobrepasarse de los límites y hacer más ameno del manejo del mismo.

Tenemos un ejemplo claro del estilo que se debe manejar, donde es más explícito y organizado, mostrando un orden vertical de un módulo o parte de código bien estructurado, ejemplo:

```
final String word = ctx.method1().method2().method3();
```

Esto denota descuido y es difícil de comprender para otros desarrolladores. Teniendo en cuenta lo anteriormente mencionado, una correcta implementación sería:

```
final String word = ctx.method1();
```

```
final Object1 obj1 = word.method2();
```

```
final Object2 obj2 = obj1.method3();
```

2.1.1.3.6 Lineamientos específicos. Documentar el código de un programa es poder añadir la suficiente información en un documento para poder explicar que se hace, cómo funciona y tener en cuenta algunos errores que se pueden presentar en su desarrollo; esto con el fin de que otras personas sepan que hacer al momento de tener algunos inconvenientes en la redacción del código, y que se entienda muy clara cuál es la finalidad de este código.

Todos los programadores y desarrolladores tienen una forma muy particular de escribir el código teniendo en cuenta la situación en la que se está trabajando, donde se busca que la máquina comprenda ciertos parámetros que el desarrollador debe interpretar a su manera pero que esto también puede traer problemas ya que si no se tiene una documentación del código, pues será muy difícil comprender cuales son las funcionalidad del aplicativo o de si ocurre algún error no se puede solucionar porque no sabemos interpretar el código con el que fue construida.

Entonces, documentar un programa no solo es un acto de buen hacer del programador, sino de ser muy profesional en su área y dejar claro todos los parámetros que se establecen para que otras personas puedan entender cada punto del código y hacer las tareas más fáciles y de poder actuar rápidamente frente a errores que pueden suceder en el transcurso del día a día.

Tenemos dos reglas que no debemos olvidar:

1. Todos los programas tienen errores y descubrirlos sólo es cuestión de tiempo y de que el programa tenga éxito y se utilice frecuentemente.
2. Todos los programas sufren modificaciones a lo largo de su vida, al menos todos aquellos que tienen éxito.

Algo que nos debe quedar claro es que debemos hacer la documentación del código con cada función que se tenga, para que en un futuro los demás ingenieros que hagan uso del mismo sepan cuál es su funcionamiento, donde también se pueda mejorar y corregir errores en tiempos menores a los esperados.

Lo más importante a documentar es añadir explicaciones a todo lo que no es evidente, no hay que repetir lo que se hace, sino explicar por qué se hace; también debemos documentar los métodos, variables, algoritmo que se usa, limitaciones, que se deberían documentar, entre otros.

Se realizará esta documentación con la herramienta swagger que es una serie de reglas, especificaciones y herramientas que nos ayudan a documentar nuestras APIs.

Por lo tanto, en esta herramienta podemos ver todos los endpoint que hemos desarrollado en nuestra API Swagger. Además, nos demuestra cómo son los elementos o datos que debemos pasar para hacer que funcione y nos permite probarlos directamente en su interfaz.

2.1.2 Pruebas de software

Cuando se realiza el desarrollo de una aplicación o software, independientemente de la metodología que se utilice, es de gran importancia realizar un completo y adecuado proceso de pruebas para garantizar que el producto final sea un producto de buena calidad y se pueda determinar si los entregables de desarrollo cumplen con las especificaciones propuestas, los

requisitos de su uso y las necesidades del usuario final. El alcance de las pruebas incluye sistemas basados en software, hardware y sus interfaces.

Las pruebas de software pueden realizarse de manera manual o automatizadas. Las pruebas manuales son las que realiza una persona, insertando y organizando todos los datos e interactuando con el software y las API, teniendo en cuenta las herramientas adecuadas para cada caso.

Anteriormente solo existía este mecanismo de pruebas, y se acudía a un equipo de control de calidad que se encargaba de desarrollar una colección de planes de prueba y determinaba si el software funcionaba correctamente. Luego de este proceso, se entregaba un informe final a los dueños o encargados del área para que realizaran los cambios pertinentes y mejorar la capacidad de sus programas o servicios. Realizar las pruebas de esta manera, terminaba siendo muy costoso para las empresas ya que se requiere de personas capacitadas en el área para configurar el entorno y ejecutar los procesos, se generan retardos y el proceso es propenso a muchos errores humanos.

A diferencia de lo anterior, las pruebas automatizadas brindan mejores beneficios ya que a través de la aplicación de herramientas de software se logra automatizar el proceso manual de revisión y validación de un producto desarrollado en un menor tiempo y con menos margen de error. Para realizar las pruebas automáticas, se han creado herramientas para cada lenguaje de programación y cada caso en particular según se requiera.

Existen diferentes tipos de pruebas de software y dependiendo del tipo, algunas se pueden automatizar y otras no. Entre los tipos más relevantes y que permiten ser automatizados tenemos las siguientes: Pruebas de extremo a extremo que simulan una experiencia a nivel de usuario en todo el producto del software, pruebas unitarias que abarcan unidades individuales de código o segmentos, pruebas de integración que se aseguran de que los módulos de código puedan interactuar entre sí y no tener restricciones y pruebas de rendimiento que se usan para describir la velocidad y capacidad de respuestas del hardware y software.

Así mismo, dentro de los tipos de pruebas que no se pueden automatizar, pero se pueden realizar manualmente se encuentran, las pruebas exploratorias que son más aleatorias y prueban secuencias sin script con la finalidad de encontrar errores o comportamientos inesperados y pruebas de regresión visual que son aquellas se pueden apreciar mejor por el ojo humano como errores de interfaz de usuario, colores erróneos, una fuente incorrecta etc.

A continuación, se presenta una especificación más detallada de los tipos de pruebas.

2.1.2.1 Pruebas funcionales. Verifican cada función de una aplicación o software, su funcionalidad con un conjunto específico de requisitos. Existen diferentes tipos de pruebas funcionales entre ellas:

2.1.2.1.1 Pruebas unitarias. Las pruebas unitarias se aseguran de que todos los componentes de un sistema funcionen como se espera. Estas pruebas son más económicas y rápidas de realizar, ya que se realizan pruebas a los métodos y funciones individuales de las clases para ver cómo se comportan y si están siendo de buen rendimiento. Este tipo de pruebas se pueden ejecutar mediante un servidor de integración continua.

2.1.2.1.2 Pruebas de extremo a extremo. Las pruebas de extremo a extremo replican el comportamiento de un usuario con el servidor en un entorno de aplicación completo. Permiten verificar que los flujos de usuarios funcionen según lo previsto. Este tipo de enfoque de prueba comienza desde la perspectiva del usuario final y simula un escenario del mundo real. Deben centrarse en la eficacia con la que la aplicación resuelve los problemas de sus usuarios.

Lo que se busca probar es que la aplicación o software trabaje conjuntamente y acceda a todas las funcionalidades que se tienen, interactuando así entre diferentes módulos y también dependiendo del tipo de usuario muestre en pantalla las opciones a las que puede acceder y las que no.

Las pruebas de extremo a extremo son una excelente manera de probar el software, pero también presentan algunos desafíos ya que toman tiempo, se deben diseñar para identificar y reproducir los flujos reales para lo cual se debe tener un buen entendimiento del objetivo de la aplicación a probar y son un poco costosas y difíciles de mantener cuando están automatizadas.

2.1.2.1.3 Pruebas de integración. Este tipo de prueba nos permite verificar los distintos módulos o servicios utilizados en la aplicación y ver si trabajan bien en conjunto. Se puede probar la interacción con la base de datos o asegurarnos de que los pequeños servicios funcionen bien en conjunto y según lo esperado.

Estas pruebas son un poco más costosas debido a que se deben tener varios elementos de la aplicación en marcha para obtener los resultados.

Al realizar este tipo de pruebas, se usa la aplicación y analizan los resultados sin examinar su funcionamiento interno. Lo que se busca es que se asegure que la comunicación entre los servicios y componentes funcionen correctamente.

2.1.2.1.4 Pruebas de humo. Las pruebas de humo son aplicadas en sistemas que deben salir rápidamente a producción o funcionamiento ya que se comprueba el funcionamiento básico de la aplicación, focalizándose en ser muy eficaces. Su objetivo es ofrecer la seguridad de que las funciones principales actúan y proceden según lo previsto.

2.1.2.1.5 Pruebas de aceptación. Son pruebas que permiten definir si el desarrollo cumple con los requisitos que fueron solicitados o contratados. Permiten asegurar que la calidad y el diseño del producto cumplan con los requisitos en términos de funcionalidad, usabilidad, durabilidad y seguridad. También se puede tantear el rendimiento del sistema, si cumple o no, si falta hardware y demás aspectos internos, donde se pueden rechazar cambios si no se han cumplido con los objetivos determinados. Generalmente se realizan antes de la comercialización o entrega.

2.1.2.1.6 Pruebas de interfaz de usuario. Las pruebas de la interfaz de usuario (IU) garantizan que la aplicación cumpla con sus requisitos funcionales y cumpla con los estándares de alta calidad, lo que aumenta las posibilidades de una adopción exitosa por parte del usuario.

2.1.2.2 Pruebas no funcionales. Consideran parámetros como la confiabilidad, la usabilidad y el rendimiento. Dentro de los tipos de pruebas NO funcionales encontramos las siguientes:

2.1.2.2.1 Pruebas de rendimiento. En estas se evalúa el rendimiento del sistema con una carga de trabajo predeterminada por una persona, donde se ayuda a medir la fiabilidad, la velocidad, la escalabilidad y la capacidad de respuesta de una aplicación.

Acá podremos medir el tiempo en que el sistema ejecuta un gran número de tareas, o de cómo se comporta con una gran cantidad de datos al mismo tiempo. Se puede determinar si se

cumple con los requisitos de rendimiento, localizar las áreas afectas o errores que surgen, medir la estabilidad durante los picos de tráfico y demás.

2.1.2.2 Prueba de carga (Load testing). Son un tipo de prueba de rendimiento utilizadas para medir o comprobar el comportamiento del sistema en condiciones normales o en condiciones de pico, donde se mida el tráfico de usuarios que acceden y su desempeño. Esto con el fin de asegurar el correcto funcionamiento del sistema cuando hay muchas interacciones. Durante la prueba se debe asegurar que cumple continuamente los estándares de rendimiento a medida que surgen cambios en el código.

2.1.2.3 Pruebas de estrés (Stress Testing). Esta prueba se usa para probar los límites del sistema, donde se busca sobrecargarlo y esperar los resultados. El objetivo principal es verificar la estabilidad y fiabilidad del sistema en condiciones extremas, teniendo en cuenta que el sistema no falle y sea capaz de dar excelentes resultados con las diferentes cargas que se le asignen, que no se bloquee.

2.1.2.4 Pruebas de volumen (Volume Testing). Esta prueba también llamada prueba de inundación, son pruebas no funcionales que se realizan para poder verificar el rendimiento del software frente a grandes cantidades de datos, donde se espera que el sistema pueda interactuar de la mejor manera y no quedarse obsoleto o no saber interpretar grandes volúmenes de datos; donde la base de datos se estira hasta un punto del umbral agregando gran cantidad de datos y luego se prueba la respuesta del sistema a los mismos.

2.1.2.2.5 Pruebas de seguridad (Security Testing). Verifican si el sistema está protegido contra ataques repentinos o deliberados de fuentes internas y externas. Es una parte integral del software que busca garantizar la seguridad en las aplicaciones web, que estén libres de amenazas, vulnerabilidades, lagunas y riesgos que pueden causar gran pérdida en la empresa u organización, y siempre verificando si los datos y recursos están protegidos de terceros.

2.1.2.2.6 Pruebas de compatibilidad (Compatibility Testing). Comprueban si la aplicación es compatible con diferentes entornos. Se busca verificar el correcto funcionamiento en gran variedad de dispositivos, como entornos de red, versiones de navegador y sistemas operativos.

2.1.2.2.7 Pruebas de instalación (Install Testing). Se debe ver si el sistema está funcionando correctamente luego de la instalación, que todos los componentes estén trabajando correctamente y sin errores, esto se hace con el fin de que el usuario final se sienta satisfecho con la aplicación y se deje trabajando correctamente en cualquier dispositivo de preferencia.

2.1.2.2.8 Pruebas de recuperación (Recovery Testing). Determina si un sistema puede recuperarse de fallas como de software, hardware o cualquier falla de red. Estas fallas pueden ocurrir debido a diferentes causas como condiciones físicas, falla de energía, servidor no accesible, y muchas más. Con este tipo de pruebas se evalúa si el sistema es capaz de recuperarse automáticamente reiniciando sus componentes o de si es una falla en el código se pueda detectar rápidamente para que el equipo de soporte la pueda solucionar con prontitud. Pero primordialmente se busca que el sistema sea capaz de identificar la falla y corregirla. En el proceso de recuperación, para evitar grandes pérdidas de datos, se deben tener planes de respaldo para que haya un impacto mínimo en el sistema en llegado evento.

2.1.2.2.9 Pruebas de confiabilidad (Reliability Testing). Garantiza que el software funciona de manera consistente realizando una tarea sin fallar dentro un período específico. Mientras que se realizan este tipo de pruebas se debe verificar las limitaciones del entorno como pérdida de memoria, batería baja, red baja, errores de bases de datos y demás limitaciones que afecten a la misma.

2.1.2.2.10 Pruebas de usabilidad (Usability Testing). Testean la facilidad de uso del usuario en términos de operación, aprendizaje y preparación de entradas y salidas. Se busca evaluar las condiciones a las que es sometido el sistema y la interpretación por parte de los usuarios, si es entendible, si es fácil de usar, complejo etc.

2.1.2.2.11 Pruebas de conformidad (Compliance Testing). Determina si un programa o sistema de software cumple con un conjunto definido de estándares internos o externos antes de su lanzamiento a producción, donde se debe pasar por un proceso de aceptación, donde se evalúan estándares y características definidas dentro de la organización para salir a producción.

2.1.2.2.12 Pruebas de localización (Localization Testing). Verifican el comportamiento de un producto de acuerdo con los entornos locales o culturales específicos. Aquí se define como hacer el contenido de un producto o aplicación, cumpliendo con los requisitos culturales, lingüísticos, y de otro tipo de región o lugar en específico donde se quiere lanzar el aplicativo. Esto con el fin de evitar confrontaciones con las culturas que van hacer uso del mismo.

2.1.3 Herramientas utilizadas para el cumplimiento de los objetivos

2.1.3.1 Git. Es un repositorio de versiones que se utiliza a través de comandos, donde este nos permite descargar a través de cmd los proyectos en los cuales se quiere trabajar en cualquier pc. Esto lo que hace es bajar los proyectos a través de links y clonarlos para su posterior funcionamiento y edición de código. Se puede editar código, compararlos con el existente en el repositorio o subir las modificaciones mediante los comandos básicos como los siguientes:

- git checkout: En que rama se encuentra.
- git clean: Eliminar los archivos sin seguimiento.
- git clone: Crear una copia del repositorio existente.

- `git commit`: Confirma la instantánea preparada en el historial del proyecto.
- `git log`: Permite explorar las revisiones anteriores de un proyecto (Atlassian. (s. f.).

Comandos básicos de Git).

2.1.3.2 Insomnia. Es una aplicación de escritorio multiplataforma gratuita que simplifica la interacción y el diseño de API basadas en HTTP. Funciona como un repositorio de APIs que se utilizan dentro de la empresa y organización, este nos permite de forma rápida, ágil y visual consumir APIS, donde la tarea de anidar peticiones se hace muy sencilla. (Insomnia: Open Source API Client)

2.1.3.3 Visual Studio Code. Es un editor de código fuente desarrollado por Microsoft para Windows, Linux, macOS y Web.

Visual Studio Code se puede extender a través de complementos, disponible a través de un repositorio central. Esto incluye adiciones al editor y soporte de lenguajes. Una característica notable es la capacidad de crear extensiones que analizan código, como linters y herramientas para análisis estático, utilizando el Protocolo de Servidor de Idioma. (Visual Studio Code - Code Editing. Redefined, 2021).

2.1.3.4 Docker Desktop. Es una herramienta para se usa para definir y ejecutar aplicaciones de Docker de varios contenedores, de esta forma se puede decir que es muy parecido a una máquina virtual ya que este nos permite descargar componentes o contenedores de código de una rama en específico, editarla o construir código y luego subir los cambios sin afectar los demás componentes que se encuentran almacenados; se puede utilizar los contenedores de Docker como bloque de construcción principal a la hora de crear aplicaciones y plataformas modernas. (AWS. (s. f.). Amazon Web Services, Inc.).

2.1.3.5 AWS (Amazon). Es una plataforma en la nube que brinda almacenamiento, infraestructura como código, bases de datos, I.A, y muchos otros servicios que las empresas tecnológicas necesitan para desempeñar sus funciones mucho más rápido, teniendo en cuenta que es una de las nubes más seguras para guardar la información como el código que se utiliza en el día a día en las empresas de desarrollo. (¿Qué es AWS? (s. f.). Amazon Web Services, Inc.)

2.1.3.6 Metodología SCRUM. Es un marco de gestión de proyectos de metodología ágil que ayuda a los equipos dentro de las empresas a estructurar y gestionar el trabajo mediante un conjunto de valores y buenas prácticas.

Al aplicar esta metodología se deben tener en cuenta que todas las actividades realizadas que se entregan, serán evaluadas por sprints, donde estos se realizan con el fin de poder tener un seguimiento claro en todos los procesos.

2.1.3.6.1 Product Owner. Es la persona que tiene contacto con el cliente, donde este tiene la capacidad de tomar decisiones en el proyecto ya que conoce muy detalladamente todo lo que el cliente quiere y este mismo puede realizar la gestión del product backlog.

2.1.3.6.2 SCRUM Master. Realiza control y seguimiento de los procesos o tareas que se llevan a cabo dentro del equipo de desarrollo por medio de sprints.

2.1.3.6.3 Equipo de desarrollo. Este equipo se encarga de desarrollar las actividades propuestas, con el fin de dar cumplimiento a los requerimientos establecidos, donde se completan entregas en los sprints con el fin de tener aprobaciones en cada tarea que se realiza.

Cada empresa define el tipo de encuentro para ver los avances y socializaciones que se tengan; dentro de estos se encuentran los siguientes:

- **SCRUM Daily:** Se realiza todos los días con una duración máxima de 15 minutos donde interactúan el líder o colaborador de cada área del proyecto, se debe socializar el trabajo que se adelantó el día anterior.
- **Sprint:** Interactúa principalmente el equipo de desarrollo, con la finalidad de mostrar los avances que se tengan o del cumplimiento de las actividades propuestas como la planificación de las actividades a realizar, revisión de sprint y la retroalimentación.
- **Sprint Retrospective:** Se comparten puntos de vista sobre lo realizado en los sprints y se toman decisiones para mejorar los procesos, donde también se pueden tener

aprobaciones y seguir trabajando con los sprints; se puede contar con la participación de todos los integrantes del proyecto. (Atlassian, s. f.).

2.1.4 Antecedentes

Diseño e implementación de un asistente virtual (chatbot) para ofrecer atención a los clientes de una aerolínea mexicana por medio de sus canales conversacionales. Este proyecto busca poder entender las IA y los beneficios que se pueden brindar para una aerolínea mexicana, con el fin de poder realizar chatbots que permitan tener una comunicación más rápida y efectiva con todos sus clientes, disminuyendo así los tiempos de respuesta a las solicitudes o quejas presentadas por los usuarios. (“Diseño e implementación de un asistente virtual (chatbot) para ofrecer atención a los clientes de una aerolínea mexicana por medio de sus canales conversacionales”, 2020).

Sobre los métodos de prueba utilizados por los probadores de software principiantes. Se realiza estudios a grupos ingenieros que apenas están iniciando su vida laboral o que están terminando la universidad, donde se busca estudiar el tipo de test o pruebas que realizan para probar los diferentes sistemas que se crean o se manejan en las diferentes plataformas que usan en el día a día; lo que se busca entender era si existían similitudes entre los diferentes grupos de evaluados y del tipo de pruebas que desempeñaban. (SIB Digital, s. f.).

Estimación y control de costos en métodos ágiles para desarrollo de software: un caso de estudio. Beneficios de las metodologías ágiles para el desarrollo de proyectos, donde se muestra

la flexibilidad en cuanto a la productividad en equipos pequeños. Se elabora una propuesta de estimación y control de costes en metodologías ágiles para solucionar carencias en cuanto al desarrollo de los proyectos, donde se especifican los elementos a intervenir, buenas prácticas en el desarrollo de los productos y aplicaciones. (SIB Digital, s. f.)

Influencia de la publicidad digital basada en WhatsApp Marketing en el nivel de satisfacción para el fortalecimiento de la fidelización de clientes de las tiendas de barrio de la localidad de Chapinero. Este proyecto nos habla de la importancia que tiene el canal de WhatsApp para hacer crecer negocios, facilitar tareas, promocionar a través de imágenes y demás componentes audiovisuales, y la influencia en la toma de decisiones de los clientes de barrio chapinero en Bogotá. (Influencia de la publicidad digital basada en WhatsApp Marketing en el nivel de satisfacción para el fortalecimiento de la fidelización de clientes de las tiendas de barrio de la localidad de Chapinero, 2021)

2.2 Enfoque legal

2.2.1 Protección de datos personales o ley 1581 de 2012

Cuando hablamos de datos personales, nos referimos a cualquier información que pueda vincularse a una persona con la que pueda identificarse. Por ejemplo, su documento de identidad, lugar de nacimiento, estado civil, edad, dirección residencia, nivel de estudios, trabajo o formación profesional. También hay información más sensible como aspectos de su salud, características físicas, ideología política, vida sexual, etc.

Se debe reconocer y proteger el derecho de todas las personas a conocer, actualizar y corregir la información recabada sobre ellas en bases de datos o archivos que puedan ser tratados por entidades públicas o privadas.

Todos los datos obtenidos se registrarán bajo esta norma, garantizando así la seguridad e integridad de los mismos. Se aplicará a todas las bases de datos y archivos que contengan datos personales.

Se es muy riguroso en este proceso ya que se pueden manejar bases de datos externas a la empresa de donde los programas adquieren información (Política de Protección de Datos Personales, 2023).

2.2.2 Ley 23 de 1982 Sobre derechos de autor.

El derecho de autor es un conjunto de normas y principios legales que confirman los derechos morales y patrimoniales que la ley otorga a los autores por el simple hecho de crear una obra literaria, artística, musical, científica o educativa, independientemente de que sea publicada o inédita. (User, s. f.).

Toda la información que se maneje dentro de la empresa y los lineamientos que se quieren proponer, son referenciados correctamente y haciendo un buen uso de la información que cada uno contenga.

2.2.3 Normas para el desarrollo de software y pruebas

El objetivo de todas las empresas y profesionales es proporcionar productos y servicios de calidad que satisfagan las expectativas del cliente. En este sentido, el conocimiento de los estándares de calidad es fundamental para entender qué pautas se deben seguir para asegurar que dichos productos y servicios son aptos para su propósito.

2.2.3.1 ISO/IEC 12207. Es un estándar de organización ISO para los procesos del ciclo de vida del software. Un conjunto de tareas como un proceso, un conjunto de pasos que incluye actividades, restricciones y recursos produce un determinado resultado, es decir, un proceso es un conjunto de actividades y tareas relacionadas que, cuando se realizan juntas, transforman la entrada en salida. (ISO/IEC/IEEE 12207:2017, 2021).

2.2.3.2 IEEE 829. Establece un marco común para los procesos, actividades y tareas de prueba para respaldar todos los procesos del ciclo de vida del software, incluidos la adquisición, la entrega, el desarrollo, las operaciones y el mantenimiento. Definir las tareas de prueba, las entradas requeridas y las salidas requeridas. Definir la integridad mínima recomendada que correspondan a los niveles de integridad planteados en la misma.

El estándar IEEE 829 define unos pasos para realizar el plan de pruebas al software y consta de 3 partes y a su vez cada una de ella tiene sus componentes que permiten sustentar lo planteado en cada paso

A. Preparación de pruebas.

1. Plan de pruebas.
2. Especificación del diseño de pruebas.
3. Especificación de casos de pruebas.
4. Procedimiento de prueba.

B. Reporte de transmisión de ítems de prueba.

5. Ejecución de las pruebas.
6. Logs de pruebas.
7. Reportes de incidentes de pruebas.

C. Término de pruebas.

8. Reporte de las pruebas.

2.2.3.3 ISO/IEC/IEEE 29119. Esta norma es una actualización a las que ya existían anteriormente y lo que busca es unificar los procesos de prueba y estándares que rigen la correcta documentación del código. Están compuestas por 5 normas o estándares. (ISO/IEC/IEEE 29119, 2017).

1. Conceptos y definiciones BS 7925-1. Que se basa en un vocabulario de términos de pruebas de software.
2. Modelo de procesos de prueba. Nos dice cómo debe ser el proceso de las pruebas y como se debe ir avanzando con sus respectivos entregables.
 - a. Procesos de prueba de organización.

- b.** Procesos de gestión de las pruebas que esta comprende la planificación, control y seguimiento y finalización.
 - c.** Procesos de pruebas dinámicas que comprende el diseño e implementación, gestión del entorno, ejecución y reporte de incidencias.
- 3.** La Documentación IEEE 829, nos dice cómo va estructurado el plan de pruebas y el orden que se debe seguir para su posterior ejecución.
- 4.** Técnicas de pruebas BS 7925-2, Esta norma define el proceso para las pruebas de componentes de software utilizando técnicas de medición y diseño de casos de prueba específicos. Esto permitirá a los usuarios del estándar mejorar directamente la calidad de sus pruebas de software y mejorar la calidad de sus productos de software.
- 5.** Revisiones de software IEEE-1028 El propósito de esta norma es definir revisiones sistemáticas y auditorías aplicables a la adquisición, suministro, desarrollo, operación y mantenimiento de software. Esta norma describe cómo realizar una revisión.
 - a.** Revisión de la gestión.
 - b.** Revisión de técnica.
 - c.** Inspección.
 - d.** Paso a paso.
 - e.** Auditoría.

3. Informe de cumplimiento de trabajo

3.1 Descripción del trabajo

Para el mejoramiento de los procesos de documentación y pruebas para la empresa Smart Data & Automation, se diseñaron los lineamientos de documentación que se deben seguir durante todo el proceso de desarrollo de los proyectos que se implementen y se creó una guía para realizar las pruebas a los productos resultantes.

Inicialmente se realizó un entendimiento del negocio que permitió definir los procesos que sigue la empresa, los actores que participan en cada una de las actividades y los artefactos de entrada y salida de cada actividad.

Con base en dicha información, se identificaron los diferentes tipos de productos y/o requerimientos que pueden conformar los proyectos (desarrollo de nuevos servicios o módulos, creación de APIs, etc) y para cada uno de ellos, se definió la documentación interna y externa que se debe generar creando en cada caso las plantillas requeridas para construir dicha documentación.

Así mismo, se respondió a las siguientes preguntas: Quién debe generar el documento (área responsable de generar la documentación), Cuándo se debe generar el documento (Actividad del proceso en la cual se debe generar la documentación) y Cómo (plantilla que se debe usar para crear un nuevo documento o indicación para actualizar uno existente).

Posteriormente, se procedió a definir el formato de plan de pruebas, y al igual que con la documentación, dependiendo los diferentes tipos de productos y/o requerimientos que pueden conformar los proyectos, se definió la guía a seguir para determinar qué tipo de pruebas realizar y en qué formato presentar los resultados.

Finalmente, para validar los lineamientos propuestos, se aplicó un caso de uso que permitió evaluar y retroalimentar la propuesta realizada.

3.1.1 Recolección de información

Para conocer las buenas prácticas que se deben implementar al momento de desarrollar código, encontrar el diseño o formato de un plan de pruebas bien formulado basado en normas internacionales, definir los lineamientos para el equipo de desarrollo y todo lo demás relacionado para el cumplimiento de los objetivos propuestos.

Se realiza una investigación muy amplia en cuanto a normas y estándares internacionales, donde se evalúa cual se adapta mejor a la empresa para comprender y generar unos lineamientos más ajustados a la mecánica empresarial que permita generar unos lineamientos más ajustados a la mecánica empresarial.

Se tuvieron en cuenta varios elementos de investigación como artículos, páginas web, proyectos y demás componentes de donde se obtuvo la información, teniendo así todas las referencias de estos.

Se realizó una investigación profunda sobre las normas que rigen la documentación del código, estructura, diseño de plantillas y demás componentes que nos permiten definir el proceso de las pruebas. Para esta etapa fue crucial el primer mes de la pasantía, ya que sobre estos elementos se trabajaría todos los demás objetivos.

3.1.2 Etapas de trabajo

Figura 6

Etapas de trabajo

Objetivos	Actividades	Agosto	Septiembre	Octubre	Noviembre
1. Definir los lineamientos y estándares que seguirá la empresa SmartData & Automation durante el proceso de desarrollo de sus productos	Realizar el plan de trabajo donde se estipulan las fases y los procesos.	■			
	Investigar sobre los lineamientos que permitan mejorar la creación de código limpio.	■			
	Registrar las actividades realizadas en el documento estipulado por la empresa.	■	■	■	■
	Definir dentro de la empresa cuál es el tipo de lineamientos para poder empezar aplicar la estandarización que se busca implementar.	■			
	Recolectar información por medio de reuniones virtuales.	■	■	■	■
	Asistir a las reuniones de seguimiento de todas las fases estipuladas.	■	■	■	■
2. Proponer una guía para realizar el proceso de pruebas que permita fomentar la entrega de productos de calidad a los clientes, garantizando la gestión y cumplimiento de los requerimientos post-producción.	Definir alcance del plan de pruebas	■			
	Determinar qué tipos de pruebas se implementará para cumplir con el alcance establecido	■			
	Seleccionar tipos y niveles de pruebas a implementar	■			
	Ejecutar casos de prueba planteados		■	■	
	Generar actividades de seguimiento y estabilización con creación de documentación		■	■	■
	Identificar herramientas a utilizar		■		
	Ajustar plan en base a resultados.		■	■	■
3. Validar los lineamientos de documentación y la guía para planear y ejecutar las pruebas propuestos a través de un caso de uso que permita brindar soporte al equipo de desarrollo.	Detallar los casos de prueba		■	■	
	Realizar pruebas que sean asignadas por parte del equipo de desarrollo				■
	Reportar hallazgos de pruebas implementadas				■
	Identificar y mapear los resultados obtenidos versus los requerimientos iniciales		■	■	

Nota. Elaboración propia.

3.1.3 Entendimiento empresarial

3.1.3.1 Plataformas. Smart Data & Automation brinda soluciones tecnológicas a través de la automatización de procesos de cara al cliente haciendo uso de Chatbots con Inteligencia Artificial y envío de notificaciones para comunicaciones masivas, donde estos componentes ayudan a los clientes a tener una comunicación más rápida y asertiva con cada usuario final, atendiendo todas las solicitudes en tiempos muy cortos.

Esto permite a las empresas poder direccionar a sus clientes a las áreas específicas donde se necesitan las soluciones, es decir, si un cliente tiene una queja o solicitud, a través de la plataforma BOTAI, se construye el flujo del chatbot y se entrena con inteligencia artificial para permitirle ser capaz de entablar una conversación casi humana y poder entender cuál es la solicitud requerida.

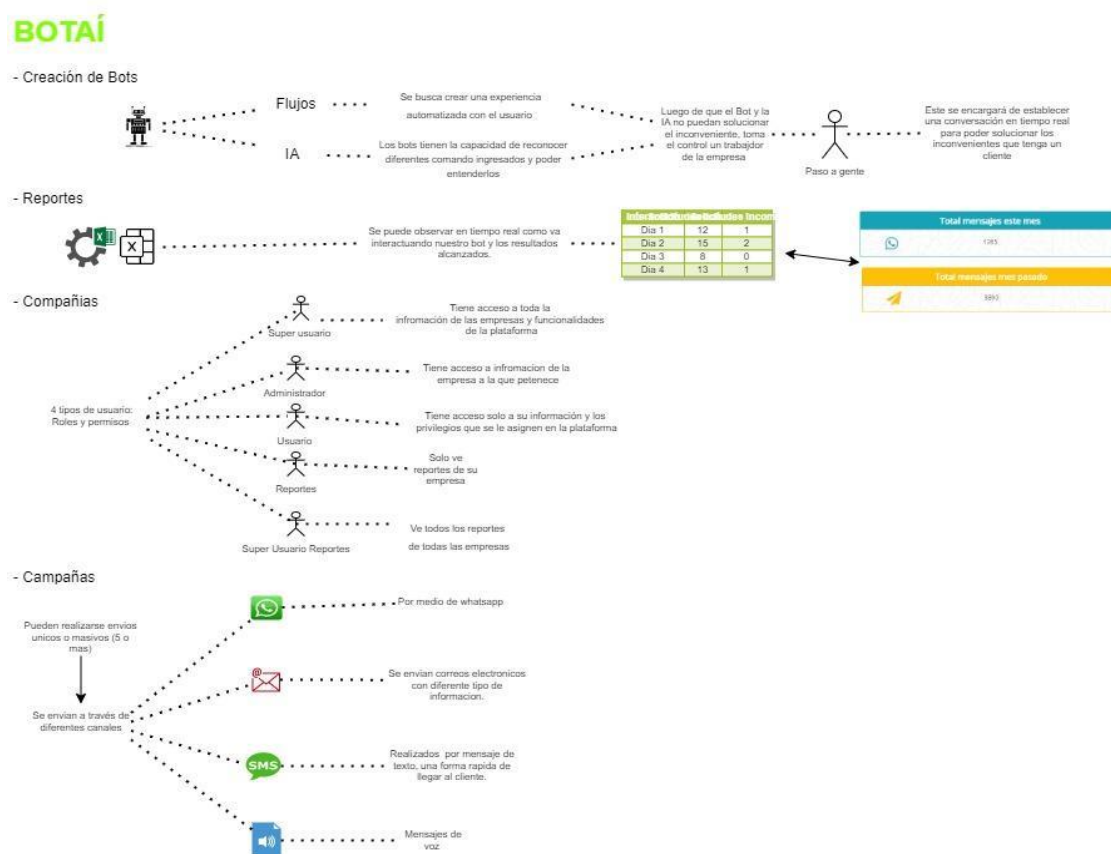
Estos chatbots pueden ser utilizados a través de múltiples canales, como los son: WhatsApp, Web, Facebook, Instagram, Telegram, Google Business Message. Si el chatbot por sí solo no logra solucionar la necesidad del cliente, es posible transferir la comunicación a un asesor humano a través de la plataforma PASO AGENTE, en la cual los trabajadores de las empresas o personal humano especializados en un área, pueden prestar la atención personalizada a los usuarios.

También, BOTAI permite enviar campañas únicas y masivas con información promocional, mensajes personalizados como recordatorios de citas, pago de deudas etc. Esto se

realiza a través del Módulo de campañas, donde el cliente puede interactuar con la plataforma y enviar los mensajes que se necesiten a los usuarios a través de canales como WhatsApp, SMS, EMAIL y VOZ.

Figura 7

Diagrama contexto BOTAI



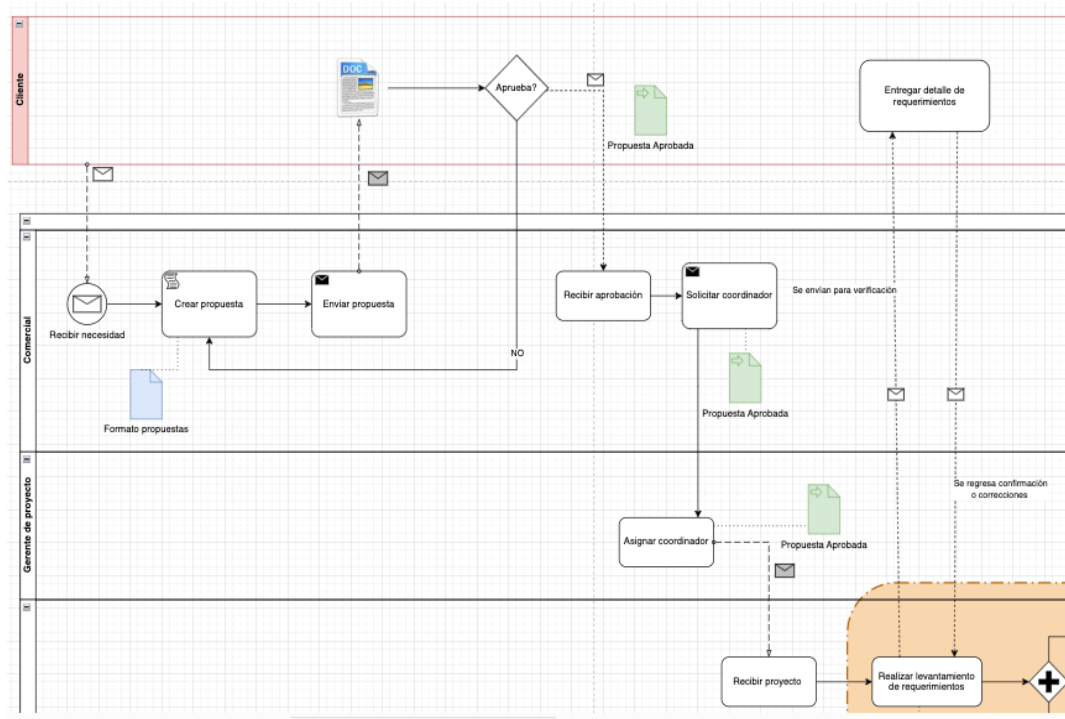
Nota. Obtenido de (Botai, 2022).

3.1.3.2 Proceso de negocio Smart Data & Automation. A continuación, se describe el proceso de negocio que sigue la empresa Smart Data & Automation para la implementación de proyectos.

El proceso inicia con la identificación de una necesidad de los clientes, esto puede ser porque el cliente llega a la empresa con un requerimiento específico o porque el área comercial de la empresa identifica la oportunidad de negocio en alguna compañía.

Independientemente de la forma como llegue la necesidad, la primera tarea por parte del equipo comercial es concretar una propuesta de trabajo con los servicios que se ofrecen y lograr llegar a un acuerdo con el cliente para iniciar con el proyecto.

Teniendo ya la aprobación del cliente, el área comercial traslada la propuesta al área de coordinación de proyectos, donde se le es asignado un coordinador responsable quien será el encargado de hacer un levantamiento detallado de requerimientos e identificar las necesidades que se tendrán en el proyecto para dar cumplimiento.

Figura 8*Diagrama proceso de negocio**Nota.* Elaboración Propia.

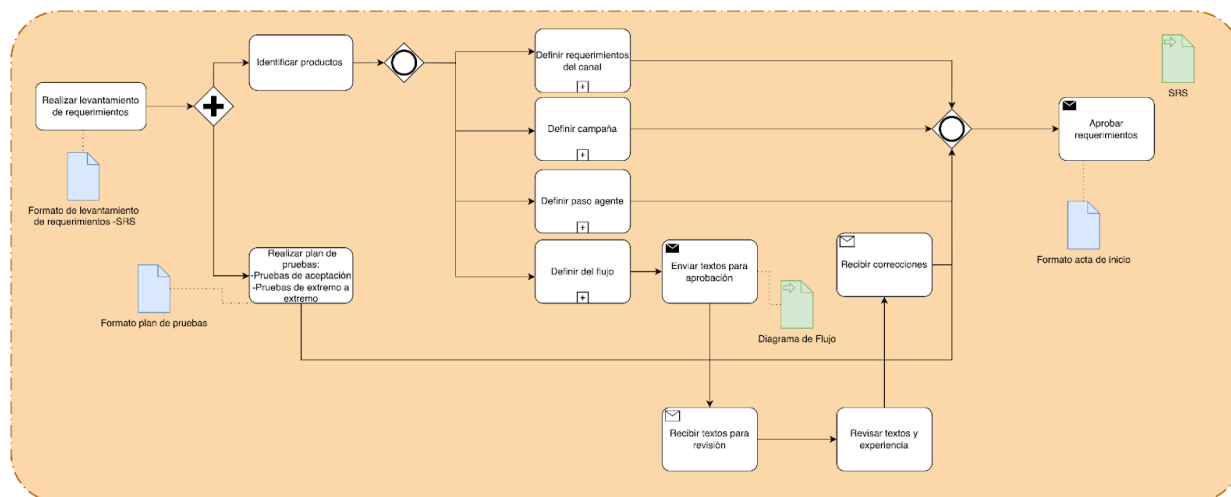
Dentro de este proceso, el coordinador identifica los productos que se van a entregar y se hace una definición detallada que pasa por la aprobación del cliente.

Ya teniendo la aprobación, y dependiendo de los productos o servicios requeridos, se identifica que equipos de trabajo dentro de la empresa deben participar en la implementación.

Cada proyecto puede tener uno o más servicios (envíos de campañas de comunicación, Chatbots, Paso Agente, integraciones, etc.), por ejemplo, puede que un proyecto solo requiera el envío de campañas masivas por WhatsApp y otro requiera campañas y chatbot.

Figura 9

Diagrama identificación de productos

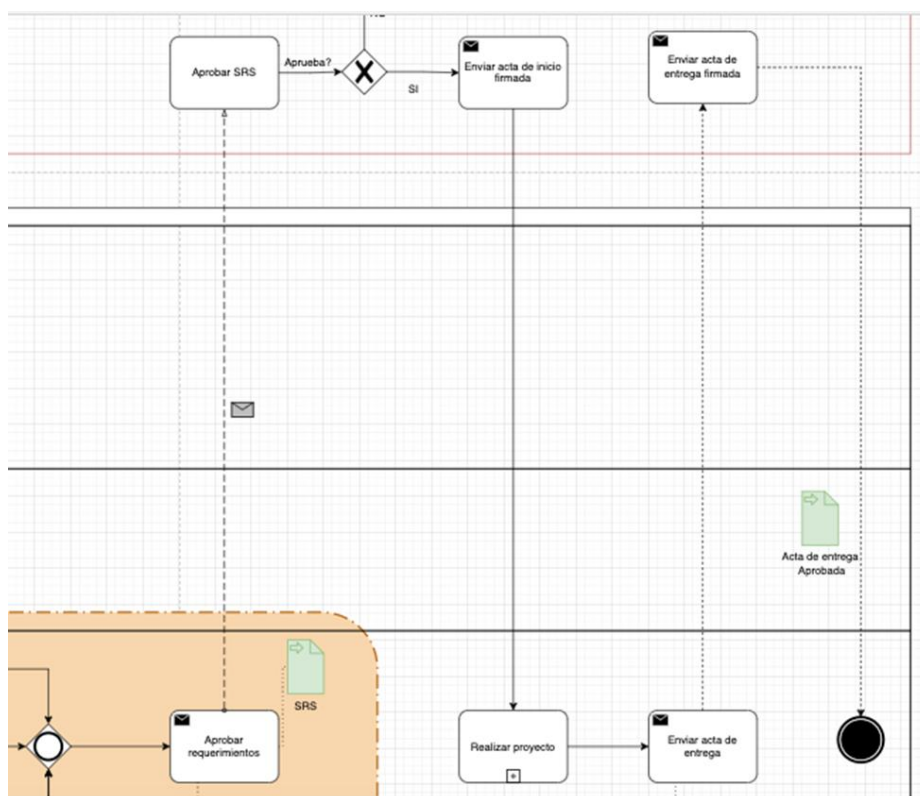


Nota. Elaboración propia.

Si los requerimientos que se tienen dentro del proyecto ya están todos desarrollados, el proyecto pasa al equipo de operaciones quienes serán los encargados de planear e implementar los productos a entregar al coordinador. Si, por el contrario, se identifica que se requiere de nuevas características, mejoras o integraciones, se pasarán estos requerimientos al equipo de desarrollo.

Figura 10

Diagrama proceso de negocio SMD&A - aprobaciones por parte de cliente



Nota. Elaboración propia.

Al finalizar la ejecución por parte del equipo de operaciones y/o desarrollo, se realizan pruebas y el coordinador hace la entrega al cliente concluyendo el proyecto con el acta de entrega aprobada.

Cabe resaltar que los proyectos pueden reabrirse si el cliente quiere realizar alguna mejora a sus productos o tiene nuevas necesidades, para lo cual se evalúa por parte del coordinador el requerimiento y dependiendo de la complejidad y el esfuerzo necesario puede pasar nuevamente a comercial para iniciar el proceso, o ser implementado directamente.

3.2 Definición de lineamientos y estándares de documentación

Para la definición de los lineamientos de documentación de la empresa Smart Data & Automation, se basó en el proceso descrito en el capítulo anterior para determinar qué documentos se deben realizar, quien los debe crear y actualizar, en qué momento se deben crear, donde se deben almacenar y qué formatos se deben seguir.

Cuando un proyecto es designado a un coordinador y él recibe la propuesta con el acuerdo al que se llegó con el cliente, empieza a materializar el proyecto realizando el levantamiento de requerimientos e identificando los tipos de servicios o productos que se deben generar. Es en este punto, donde dependiendo el resultado del análisis y si en la identificación se determina que hay que hacer algún tipo de desarrollo se evalúa cual es el primer documento a realizar.

plantilla de diseño de software entreguen al área de desarrollo los lineamientos de diseño e interacción del usuario que deberá seguir la implementación que se hará.

Cuando el equipo de desarrollo ya cuenta con los documentos que le definen de forma precisa el desarrollo a implementar, los desarrolladores deben seguir los Lineamientos de documentación interna tanto genéricos como específicos para el lenguaje de programación que se emplea.

Al culminar los desarrollos requeridos, el equipo de desarrollo deberá realizar la siguiente documentación:

Figura 12

Excel con información que se debe llenar para cada desarrollo – parte 2

PARA CREACIÓN DE NUEVOS PROYECTOS, SE DEBEN REALIZAR CIERTOS DOCUMENTOS DEPENDIENDO DE LO QUE CONTENGA	DOCUMENTOS	DIRECTORIO DE IMPLEMENTACIONES	ESPECIFICACIÓN GENERAL	DOCUMENTO DE ARQUITECTURA	Manual de configuración
RESPONSABLES	AREA ENCARGADA	DESARROLLO	DESARROLLO	DESARROLLO	DESARROLLO
FORMATOS	LINKS	<input type="checkbox"/> Plantilla Directorio APIS	<input type="checkbox"/> Plantilla especificación general	<input type="checkbox"/> Plantilla arquitectura del Software	<input type="checkbox"/> Plantilla Manual de Configuración
API ESPECIFICA	FRONTEND	X			
	BACKEND	X			X
API GENERICA	FRONTEND	X			
	BACKEND	X			
NUEVA FUNCIONALIDAD	FRONTEND				
	BACKEND				
CORRECCION DE ERROR	FRONTEND				
	BACKEND				
NUEVO SERVICIO	FRONTEND		X		X
	BACKEND		X		X
	CAMBIO DE ARQUITECTURA			X	X
NUEVO MODULO	FRONTEND	X			X
	BACKEND	X			X
	CAMBIO DE ARQUITECTURA			X	X

Nota. Elaboración propia.

Figura 13

Excel con información que se debe llenar para cada desarrollo – parte 3

PARA CREACIÓN DE NUEVOS PROYECTOS, SE DEBEN REALIZAR CIERTOS DOCUMENTOS DEPENDIENDO DE LO QUE CONTENGA	DOCUMENTOS	PLAN DE PRUEBAS		
RESPONSABLES	AREA ENCARGADA	OPERACIONES Y SOPORTE		OPERACIONES Y SOPORTE
FORMATOS	LINKS	Plantilla Especificaciones de pruebas funcionales	Plantilla Pruebas de interfaz de usuario.xlsx	Plantilla Plan de pruebas BOTAI módulo campañas whatsapp
API GENERAL	BACKEND	x		
API GENERICA	FRONTEND			
	BACKEND	x		
NUEVA FUNCIONALIDAD	FRONTEND		x	
	BACKEND	x		
CORRECCION DE ERROR	FRONTEND		x	
	BACKEND	x		
NUEVO SERVICIO	FRONTEND		x	X
	BACKEND	x		X
	CAMBIO DE ARQUITECTURA			X
NUEVO MODULO	FRONTEND		x	X
	BACKEND	x		X
	CAMBIO DE ARQUITECTURA			X

Nota. Elaboración propia.

3.2.1 Documentación de API específicas

Estas APIs son mecanismos que permiten al FrontEnd y BackEnd comunicarse entre sí, permitiendo así el flujo de información continuo que permite programar ciertos funcionamientos de una aplicación determinada dentro del desarrollo del software. Estas APIs se desarrollan en FrontEnd y BackEnd.

Para FrontEnd se deben ocupar los documentos plantilla directorio de APIs y plantilla especificación de Apis. Plantilla directoria de Apis:

Figura 14

Excel plantilla directorio de APIs donde se especifica toda la información.

A	B	C	D	E	F	G
11	EC2	Directorio	Proyecto	Tipo	Dominio	Repositorio
52.4.88.69	Didactic City	/var/www/html/finatic/			finatic.smartdataautomation.com	
52.4.88.69	Didactic City	/var/www/html/bancov/		Desarrollo a la medida	bancov.site	https://git-codecommit.us-east-1.amazonaws.com

Nota. Elaboración propia.

Este documento lo realiza el equipo de desarrollo en cada momento se realiza una nueva API o que se requiere documentar una ya existente, con el fin de que quede correctamente documentada cada creación.

Este documento contiene toda la información importante que se requiere para documentar, a continuación, se explica cuál es la función de cada columna y como se debe llenar.

- 11: Es la instancia donde se encuentra desplegado el desarrollado, esta cuenta con las columnas, la IP y el Nombre.
- EC2: Es el nombre de la instancia donde se encuentra el desarrollo.
- Directorio: Link o URL del directorio donde se encuentra.
- Proyecto: Nombre del proyecto a cuál pertenece.
- Tipo: Si es de tipo API, Landing, Desarrollo a la medida, Servicio.
- Dominio: Link de donde pertenece.
- Repositorio: Link del repositorio donde se encuentra guardado.
- Estado: Se especifica si está en Producción, Desarrollo, Demo o, Ya no se usa.
- Propósito: Para qué tipo de plataforma se usa.

- Observaciones: Se dan breves observaciones y se anexan imágenes de ser necesario.
 - Curl prueba: link donde se prueba la API.
 - URL bot donde se usa: Link o url del bot que la usa actualmente.
 - Coordinador Solicitó/A cargo: Nombre de quien solicita o está a cargo de la información.
 - Desarrollador(es): Nombre de quien desarrolló la API.
 - Link de documentación: URL o LINK sobre la documentación que la sustenta.
 - Lenguaje de programación: Tipo de lenguaje en el cual está desarrollada. (PHP, JAVA, C, C++, etc.)
 - Conexión a BD: A qué tipo de base de datos corresponde. (MySQL, NoSQL, SQLite, MongoDB, etc.).
- Plantilla especificación de APIs:

Figura 15

Imagen documento estructura

Ejemplos de Estructura para realizar la solicitud al equipo de desarrollo

Nombre del Proyecto	Movii biometria	
Link al flujo del bot	https://app.diagrams.net/#G1vUFz_Ry8nq1yJDqiR4BddF2hE7L32TII	
Fecha límite entrega	Tiene fecha límite () _____	Solicitud de Estimación de tiempos (X)
Implementador	Pablo	

Proceso	Envío de información inicial - 4 intentos diarios				
Tipo de Solicitud	API Especifica (X)	API Genérica ()	Cron ()	Funcionalidad BotAI ()	Funcionalidad Paso Agente ()
Objetivo	COMO	NECESITO		PARA	
	Implementador del bot	Validar en la BD de registro inicial si el usuario con el documento digitado tiene más de 4 intentos		Permitir o denegar que se continúe con el proceso de cambio de número	
Datos de entrada	- Tipo de documento (Obligatorio) - Número de documento (Obligatorio)				
Escenarios	Casos		Salidas esperadas		
	El usuario tiene más de 4 intentos		mensaje: "Tiene más de 4" codigo:111		
	El usuario tiene menos de cuatro intentos		mensaje: "No tiene más de 4" codigo:112		
	El usuario no está registrado		mensaje: "Usuario no registrado" codigo:113		
Procesos	N/A				

Nota. Elaboración propia.

Este documento lo realiza la coordinación de proyectos en el momento que empieza la implementación del proyecto. En este documento se detallan elementos importantes que componen la API como nombres de proyectos, link de flujo o comunicación con el Bot, fechas estimadas de entrega en caso de ser un nuevo api, implementar que usa o desarrollar.

- Se define el proceso que realiza, tipo de API si es específica, genérica, cron,etc.
- Objetivo del API, como, necesito, para, donde se debe definir cada uno de ellos.
- Datos de entrada, funcionalidad del API, que datos se necesitan para que funcione.
- Escenarios esperados con casos y salidas esperadas

Para BackEnd se debe ocupar la plantilla directoria de apis, plantilla especificación de APIs y plantilla manual de configuración.

Para plantilla directorio de APIs y plantilla especificación de APIs se deben seguir los lineamientos de APIs específicas. Para plantilla manual de configuración lo realiza el equipo de desarrollo especificando campos como:

- Pasos para configurar una nueva máquina: se realiza el paso a paso del proceso que se deben desplegar para desplegar el desarrollo en una nueva máquina.
- Pasos para hacer un release a producción: Se realiza el paso a paso de cómo se deben subir los cambios y actualizar las versiones de producción.
- Configuración del entorno de desarrollo: Se debe colocar los espacios sugeridos para que un desarrollador pueda implementar los ajustes requeridos.

Figura 16

Documento manual de configuración



Nota. Elaboración propia.

3.2.2 Documentación API genérica

Para estos desarrollos que se usan en FrontEnd y BackEnd se definen los documentos para detallar cada uno de ellos.

Para FrontEnd se deben llenar documentos de plantilla directorio de APIs que lo realiza el equipo de desarrollo y plantilla especificación de APIs que lo realiza coordinación de proyectos.

Para estos documentos se deben seguir los lineamientos establecidos en documentación de APIs específicas.


Para BackEnd se deben llenar documentos de plantilla directorio de APIs que lo realiza el equipo de desarrollo y plantilla especificación de APIs que lo realiza coordinación de proyectos. Para estos documentos se deben seguir los lineamientos establecidos en documentación de APIs específicas.

3.2.3 Documentación de nueva funcionalidad

Para este tipo de desarrollo se debe realizar el documento de plantilla para un nuevo requerimiento tanto para FrontEnd como para BackEnd y es realizado por la coordinación de proyectos.

Figura 17

Documento especificación de nuevo requerimiento

Smart Data & Automation	ESPECIFICACIÓN DE NUEVO REQUERIMIENTO	Smart Data & Automation																										
 <p>Especificación de un nuevo requerimiento</p>																												
<table border="1"> <thead> <tr> <th>Smart Data & Automation</th> <th>ESPECIFICACIÓN DE NUEVO REQUERIMIENTO</th> <th>Smart Data & Automation</th> </tr> </thead> <tbody> <tr> <td colspan="3"> <table border="1"> <thead> <tr> <th>Nombre del Caso de Uso</th> <th>Autenticarse en el sistema</th> </tr> </thead> <tbody> <tr> <td>Código del Caso de Uso:</td> <td>CU01</td> </tr> <tr> <td>Versión:</td> <td>0.0.0.1</td> </tr> <tr> <td>Descripción:</td> <td>Este caso de uso permite identificar los pasos necesarios para iniciar sesión en la plataforma web.</td> </tr> <tr> <td>Actor(es):</td> <td>Usuario - administrador</td> </tr> <tr> <td>Precondición:</td> <td> <ul style="list-style-type: none"> El usuario debe ingresar a la plataforma web El usuario debe estar registrado previamente en la plataforma web </td> </tr> <tr> <td>Escenario Básico:</td> <td> 1. El caso de uso comienza cuando el usuario ingresa a la plataforma web desde su computador o Tablet. <ol style="list-style-type: none"> En la pantalla podrá identificar la opción: <ol style="list-style-type: none"> Iniciar Sesión Accede a la opción "Iniciar Sesión" y llena los espacios <ul style="list-style-type: none"> Usuario Contraseña El usuario selecciona la opción "Iniciar Sesión" </td> </tr> <tr> <td>Escenarios Alternativos:</td> <td> <ul style="list-style-type: none"> Error en el inicio de sesión por problemas en las opciones: <ul style="list-style-type: none"> Usuario Contraseña </td> </tr> <tr> <td>Puntos de Extensión:</td> <td></td> </tr> <tr> <td>PostCondición:</td> <td>El usuario inicia sesión en la plataforma y puede comenzar a administrar su plataforma de acuerdo al rol que tenga asignado</td> </tr> </tbody> </table> </td> </tr> </tbody> </table>			Smart Data & Automation	ESPECIFICACIÓN DE NUEVO REQUERIMIENTO	Smart Data & Automation	<table border="1"> <thead> <tr> <th>Nombre del Caso de Uso</th> <th>Autenticarse en el sistema</th> </tr> </thead> <tbody> <tr> <td>Código del Caso de Uso:</td> <td>CU01</td> </tr> <tr> <td>Versión:</td> <td>0.0.0.1</td> </tr> <tr> <td>Descripción:</td> <td>Este caso de uso permite identificar los pasos necesarios para iniciar sesión en la plataforma web.</td> </tr> <tr> <td>Actor(es):</td> <td>Usuario - administrador</td> </tr> <tr> <td>Precondición:</td> <td> <ul style="list-style-type: none"> El usuario debe ingresar a la plataforma web El usuario debe estar registrado previamente en la plataforma web </td> </tr> <tr> <td>Escenario Básico:</td> <td> 1. El caso de uso comienza cuando el usuario ingresa a la plataforma web desde su computador o Tablet. <ol style="list-style-type: none"> En la pantalla podrá identificar la opción: <ol style="list-style-type: none"> Iniciar Sesión Accede a la opción "Iniciar Sesión" y llena los espacios <ul style="list-style-type: none"> Usuario Contraseña El usuario selecciona la opción "Iniciar Sesión" </td> </tr> <tr> <td>Escenarios Alternativos:</td> <td> <ul style="list-style-type: none"> Error en el inicio de sesión por problemas en las opciones: <ul style="list-style-type: none"> Usuario Contraseña </td> </tr> <tr> <td>Puntos de Extensión:</td> <td></td> </tr> <tr> <td>PostCondición:</td> <td>El usuario inicia sesión en la plataforma y puede comenzar a administrar su plataforma de acuerdo al rol que tenga asignado</td> </tr> </tbody> </table>			Nombre del Caso de Uso	Autenticarse en el sistema	Código del Caso de Uso:	CU01	Versión:	0.0.0.1	Descripción:	Este caso de uso permite identificar los pasos necesarios para iniciar sesión en la plataforma web.	Actor(es):	Usuario - administrador	Precondición:	<ul style="list-style-type: none"> El usuario debe ingresar a la plataforma web El usuario debe estar registrado previamente en la plataforma web 	Escenario Básico:	1. El caso de uso comienza cuando el usuario ingresa a la plataforma web desde su computador o Tablet. <ol style="list-style-type: none"> En la pantalla podrá identificar la opción: <ol style="list-style-type: none"> Iniciar Sesión Accede a la opción "Iniciar Sesión" y llena los espacios <ul style="list-style-type: none"> Usuario Contraseña El usuario selecciona la opción "Iniciar Sesión" 	Escenarios Alternativos:	<ul style="list-style-type: none"> Error en el inicio de sesión por problemas en las opciones: <ul style="list-style-type: none"> Usuario Contraseña 	Puntos de Extensión:		PostCondición:	El usuario inicia sesión en la plataforma y puede comenzar a administrar su plataforma de acuerdo al rol que tenga asignado
Smart Data & Automation	ESPECIFICACIÓN DE NUEVO REQUERIMIENTO	Smart Data & Automation																										
<table border="1"> <thead> <tr> <th>Nombre del Caso de Uso</th> <th>Autenticarse en el sistema</th> </tr> </thead> <tbody> <tr> <td>Código del Caso de Uso:</td> <td>CU01</td> </tr> <tr> <td>Versión:</td> <td>0.0.0.1</td> </tr> <tr> <td>Descripción:</td> <td>Este caso de uso permite identificar los pasos necesarios para iniciar sesión en la plataforma web.</td> </tr> <tr> <td>Actor(es):</td> <td>Usuario - administrador</td> </tr> <tr> <td>Precondición:</td> <td> <ul style="list-style-type: none"> El usuario debe ingresar a la plataforma web El usuario debe estar registrado previamente en la plataforma web </td> </tr> <tr> <td>Escenario Básico:</td> <td> 1. El caso de uso comienza cuando el usuario ingresa a la plataforma web desde su computador o Tablet. <ol style="list-style-type: none"> En la pantalla podrá identificar la opción: <ol style="list-style-type: none"> Iniciar Sesión Accede a la opción "Iniciar Sesión" y llena los espacios <ul style="list-style-type: none"> Usuario Contraseña El usuario selecciona la opción "Iniciar Sesión" </td> </tr> <tr> <td>Escenarios Alternativos:</td> <td> <ul style="list-style-type: none"> Error en el inicio de sesión por problemas en las opciones: <ul style="list-style-type: none"> Usuario Contraseña </td> </tr> <tr> <td>Puntos de Extensión:</td> <td></td> </tr> <tr> <td>PostCondición:</td> <td>El usuario inicia sesión en la plataforma y puede comenzar a administrar su plataforma de acuerdo al rol que tenga asignado</td> </tr> </tbody> </table>			Nombre del Caso de Uso	Autenticarse en el sistema	Código del Caso de Uso:	CU01	Versión:	0.0.0.1	Descripción:	Este caso de uso permite identificar los pasos necesarios para iniciar sesión en la plataforma web.	Actor(es):	Usuario - administrador	Precondición:	<ul style="list-style-type: none"> El usuario debe ingresar a la plataforma web El usuario debe estar registrado previamente en la plataforma web 	Escenario Básico:	1. El caso de uso comienza cuando el usuario ingresa a la plataforma web desde su computador o Tablet. <ol style="list-style-type: none"> En la pantalla podrá identificar la opción: <ol style="list-style-type: none"> Iniciar Sesión Accede a la opción "Iniciar Sesión" y llena los espacios <ul style="list-style-type: none"> Usuario Contraseña El usuario selecciona la opción "Iniciar Sesión" 	Escenarios Alternativos:	<ul style="list-style-type: none"> Error en el inicio de sesión por problemas en las opciones: <ul style="list-style-type: none"> Usuario Contraseña 	Puntos de Extensión:		PostCondición:	El usuario inicia sesión en la plataforma y puede comenzar a administrar su plataforma de acuerdo al rol que tenga asignado						
Nombre del Caso de Uso	Autenticarse en el sistema																											
Código del Caso de Uso:	CU01																											
Versión:	0.0.0.1																											
Descripción:	Este caso de uso permite identificar los pasos necesarios para iniciar sesión en la plataforma web.																											
Actor(es):	Usuario - administrador																											
Precondición:	<ul style="list-style-type: none"> El usuario debe ingresar a la plataforma web El usuario debe estar registrado previamente en la plataforma web 																											
Escenario Básico:	1. El caso de uso comienza cuando el usuario ingresa a la plataforma web desde su computador o Tablet. <ol style="list-style-type: none"> En la pantalla podrá identificar la opción: <ol style="list-style-type: none"> Iniciar Sesión Accede a la opción "Iniciar Sesión" y llena los espacios <ul style="list-style-type: none"> Usuario Contraseña El usuario selecciona la opción "Iniciar Sesión" 																											
Escenarios Alternativos:	<ul style="list-style-type: none"> Error en el inicio de sesión por problemas en las opciones: <ul style="list-style-type: none"> Usuario Contraseña 																											
Puntos de Extensión:																												
PostCondición:	El usuario inicia sesión en la plataforma y puede comenzar a administrar su plataforma de acuerdo al rol que tenga asignado																											
<p>¹Autor ²Nombre del proyecto al cual está asignado o se busca modificar²</p>																												
<p>Se realiza un nuevo requerimiento y se actualiza el SRS.</p>																												

Nota. Elaboración propia.

Este documento debe ir inicialmente el autor y el nombre del proyecto al cual está asignado o se busca modificar. Se debe llenar una tabla con características como:

- Nombre del caso de uso: Que hace referencia en el sistema.
- Código del caso de uso.
- Versión.
- Descripción: Se detalla información breve de que se busca hacer.
- Actor(es): quienes intervienen como, por ejemplo, usuario-administrador.
- Precondición: Que debe realizar el usuario para acceder al sistema.
- Escenario básico: Que realiza el usuario dentro del sistema.
- Escenarios alternativos: Errores que pueden surgir al momento que el usuario haga uso del sistema.
- Puntos de extensión.
- postCondición: Se detalla que hace el usuario después de acceder al sistema.

3.2.4 Documentación nuevo servicio

Cuando se crea un nuevo servicio dependiendo de las necesidades del usuario se debe tener en cuenta la documentación para FrontEnd, BackEnd y cambio de arquitectura.

Para FrontEnd de deben llenar documentos como plantilla de especificación general (hoja de vida) por parte del equipo de desarrollo, plantilla especificación de requerimientos-SRS por parte de coordinación de proyectos, plantilla diseño del software por parte de equipo de diseño y

experiencia, plantilla manual de configuración por parte del equipo de desarrollo y plantilla plan de pruebas Botai por parte de operaciones y soporte.

Para BackEnd se deben llenar documentos como plantilla de especificación general (hoja de vida) por parte del equipo de desarrollo, plantilla especificación de requerimientos-SRS por parte de coordinación de proyectos, plantilla manual de configuración por parte del equipo de desarrollo y plantilla plan de pruebas Botai por parte de operaciones y soporte.

Para cambio de arquitectura se deben llenar los documentos de SRS- especificación de requerimientos que lo realiza coordinación de proyectos, plantilla arquitectura del software que lo realiza el equipo de desarrollo, plantilla manual de configuración que lo realiza el equipo de desarrollo y el plan de pruebas botai que lo realiza el equipo de operaciones y soporte.

3.2.4.1 Para FrontEnd.

Figura 18

Documento plantilla especificación general

Documento	URL de acceso
SRS- Documento especificación de requerimientos	Plantilla SRS- Especificación de requerimientos
Documento de arquitectura	Plantilla arquitectura del Software
	Plantilla Modelos Arquitecturales
Documento de diseño	Plantillas diseño del Software
Manual de configuración	Plantilla Manual de Configuración
Manual de usuario	Plantilla Manual de usuario módulo campañas
Plan de pruebas	Plan de pruebas BOTAI módulo campañas whatsapp

Nota. Elaboración propia.

Figura 19

Plantilla especificación general



Smart Data & Automation
Let's Automate the world

Nombre del proyecto
Plantilla de especificación general

Descripción breve del proyecto
Este proyecto nace de la necesidad que presenta el cliente al momento de atender varias solicitudes en su empresa...

Tipo de lenguaje
Java
Python
Laravel
Visual Basic

Tipo de desarrollo
Backend
Frontend
Api

Indicaciones relevantes del proyecto
Explicar que contiene el proyecto en sus archivos, como está dividido etc.

Objetivo del proyecto
Realizar un paso agente que permita la correcta comunicación de las personas con un miembro especializado de la empresa

Cliente para el cual va dirigido el proyecto
Nombre del cliente
Datos de la empresa
Descripción breve de la empresa

Coordinador del proyecto
NOMBRE DE COORDINADOR
Área
correo electrónico

Equipo de trabajo
Nicolas Lopez
Teléfono: (800) 751 0090
Correo electrónico: nicola@stoncreble.com
Raquel Quintero
Teléfono: (56 1) 0450
correo electrónico: smart@data.com

Nota. Elaboración propia.

En este documento se da una primera vista sobre lo que contiene el proyecto como nombre del proyecto, objetivo del proyecto, cliente para el cual va dirigido, coordinador de proyecto, equipo de trabajo.

Datos más específicos de desarrollo del proyecto como descripción breve del proyecto, tipo de lenguaje, tipo de desarrollo, indicaciones relevantes del proyecto.

Y, por último, se dejan registrados los documentos que están anexos del desarrollo y dan soporte con los links específicos de cada uno.

3.2.4.1.1 Plantilla SRS- Especificación de requerimientos. Este documento se puede visualizar completamente en anexos A, SRS- Especificación de requerimientos, donde se explica que debe ir en cada paso y es realizado coordinación de proyectos.

Este documento explica minuciosamente todos los requerimientos, donde se debe plasmar la descripción global, requerimientos específicos, requerimientos no funcionales y entregables del desarrollo.

Figura 20

Plantilla especificación de requerimientos

Contenido	
1. INTRODUCCIÓN	5
Objetivo general	5
1.1. Propósito	5
1.2. Alcance	5
1.3. Definiciones, acrónimos y abreviaciones	5
2. DESCRIPCIÓN GLOBAL	6
2.1. Descripción general (¿Que es?)	6
2.2. Producto (Que se entrega)	6
2.3. Diseño conceptual de software (¿Que es y cómo funciona?)	7
2.4 Características de usuario	8
3. REQUERIMIENTOS ESPECÍFICOS	8
3.1 Métricas y avance	8
3.2 Tabla de requerimientos funcionales	9
4. REQUERIMIENTO NO FUNCIONALES	9
4.1. Usabilidad	9
4.2. Disponibilidad	9
4.3. Desempeño	10
4.4. Portabilidad	10
4.6. Seguridad	11
4.7. Ambiente, sistema operativo	11
4.8. Restricciones de diseño	11
4.9. Requerimientos de documentación y ayuda en línea del sistema	11
4.10. Interfaces	11
4.11. Estandarización	12
5. Entregables	13
5.1. Documentos de diseño.	13
5.2. Manual de instalación.	13
5.3. Manual de usuario.	13
5.4. Código fuente.	13
5.5. Archivos ejecutables.	13

Nota. Elaboración propia.

3.2.4.1.2 Plantilla diseño del software. Este documento es realizado por parte del equipo de diseño y experiencia donde se detalla una descripción general de todo lo que se requiere para el diseño del software, teniendo en cuenta cada área.

Se muestran el diseño de los mockups con los requerimientos establecidos por el cliente para mostrar una vista previa del proyecto y esperar la aprobación del cliente para avanzar con el proyecto acorde a las condiciones que el mismo establezca y se muestra el diagrama de interfaz gráfica de cómo funciona si se requiere.

Figura 21

Documento de diseño



Nota. Elaboración propia.

Plantilla manual de configuración:

Se deben seguir los lineamientos establecidos en APIs específicas en desarrollo de BackEnd.

3.2.4.1.3 Plan de pruebas Botai. Se realiza por parte de operaciones y soporte, donde se especifican las áreas a probar, el criterio de pruebas, ítems a evaluar, herramientas usadas en el proceso y demás componentes de pruebas.

Este documento consta de tres elementos importante:

- 1. Plan maestro de pruebas:** Se identifican los elementos a probar, problemas de prueba, características, entregables, aprobaciones etc.
- 2. Ejecución de pruebas:** luego de identificar todos los elementos a probar, ítems de prueba, criterios y demás, se realiza la ejecución de cada una de las pruebas establecidas en el plan maestro. Aquí se muestran los logs de pruebas que son los detalles de cada prueba y el reporte de incidentes de las pruebas que salen mal.
- 3. Término de pruebas:** Se anexan todos los reportes que surgen de las pruebas como documentos en Word, formatos de Excel con pruebas etc.

Figura 22

Documento plan de pruebas



PLAN DE PRUEBAS DE SOFTWARE Plataforma BOTAI

Tabla de contenido

Plan Maestro de pruebas (MTP)	2
Test Plan Identifier (Identificador de plan de pruebas)	2
Introduction (Introducción)	2
Test Items (Elementos de prueba)	2
Software Risk Issues (Problemas de riesgo de software)	3
Features to be Tested (Características a probar)	4
Features not to be Tested (Características que no deben probarse)	4
Approach	4
Item Pass/Fail Criteria (Elementos criterios de aprobación/rechazo)	7
Suspension Criteria and Resumption Requirements (Criterios de suspensión y requisitos de reanudación)	8
Test Deliverables (Entregables de prueba)	8
Remaining Test Tasks (Tareas de prueba restantes)	8
Environmental Needs (Necesidades ambientales)	8
Staffing and Training Needs (Necesidades de personal y capacitación)	8
Responsibilities (Responsabilidades)	9
Schedule (Calendario)	9
Planning Risks and Contingencies (Planificación de riesgos y contingencias)	9
Approvals (Aprobaciones)	9
Glossary (Glosario)	9
Especificación del diseño de pruebas de nivel (LTD)	9
Especificación de los casos de pruebas de nivel (LTC)	9
Procedimiento de pruebas de nivel ()	9
Reporte de transmisión de ítems de pruebas (LTL)	9
Ejecución de pruebas	9
Logs de Pruebas: Se registran con detalle qué pruebas se han ejecutado, orden y resultados de los mismos	9
Reporte de incidentes de pruebas	9
Término de pruebas	9
Reporte de pruebas	9
References	9

Nota. Elaboración propia.

Este documento que se definió en base a la norma IEEE 829 se puede observar en **Apéndice A**, formato plan de pruebas Botai.

3.2.4.2 Para BackEnd. Se realizan documento como plantilla especificación general, plantilla srs-especificación de requerimientos, plantilla manual de configuración, plantilla plan de pruebas botai.

Para estos documentos se deben seguir los lineamientos establecidos anteriormente mencionados para el FrontEnd.


3.2.4.2.1 Para cambio de arquitectura. Para los documentos de SRS-especificación de requerimientos, plantilla manual de configuración y el plan de pruebas botai se deben seguir los lineamientos anteriormente mencionados para el FrontEnd (documentación nuevo servicio).

Para plantilla arquitectura del software se deben tener en cuenta tres aspectos que son:


- 1. Restricciones arquitecturales:** este comprende los motivadores de negocio, restricciones de tecnología, atributos de calidad y árbol de utilidad. Se debe detallar toda la información dentro de cada ítem para poder llevar un proceso exitoso en el cambio de una arquitectura.
- 2. Puntos de vista y modelos arquitecturales:** se tiene los puntos de vista lógicos, modelo funcional, punto de vista física, modelo de red, modelo de dependencia tecnológica, modelo de estructura estática de datos y de flujos de información.
- 3. Relaciones entre los puntos de vista:** se da una descripción sobre a que están orientadas las vistas, donde y con que se relacionan, ser muy detallado para poder tener procesos claros y minimizar errores.


Figura 23

Plantilla documento de arquitectura





Plantilla Arquitectura del software





Plantilla Arquitectura del software





Plantilla documento de arquitectura

1. RESTRICCIONES ARQUITECTURALES

1.1. Motivadores de Negocio

Tabla ejemplo:

Tabla Motivador de negocio f	
Nombre del Motivador de Negocio	Descripción del Motivador de Negocio
Asociación con Dengue Grave	Conocer el serotipo y genotipo circulante para inferir el origen de estos aislados, reconstruir las vías de introducción/desplazamiento del virus dentro de la región y servir como marcadores epidemiológicos tempranos para conocer si los genotipos circulantes se asocian con Dengue Grave.
Métrica del negocio	
100% de conocimiento de la asociación con dengue grave de los genotipos circulantes	
Rango:	Costo Mínimo:
Máximo: 0%	0%
Medio: 10%	
Mínimo: 20%	
0%	
Asociación del Motivador con el Negocio:	Definido Por: Analista
	Estimado Por: Técnico
	Ubicación: en el servicio
	Impacto: en el motivador del negocio

Pueden existir varias tablas con varios motivadores de negocio

1.2. Restricciones de Tecnología

Tabla Restricciones de Tecnología f	
ID Restricción - RT 001	Tipo - Tecnológica
Descripción:	Las revisiones a los cambios pueden basarse tanto de base y al nivel técnico a nivel de implementación con la base de datos.
Estimado por:	Área de Investigación de Biología - base
Afectados:	Podrá cargar la información apropiada de un evento cuando se haga acceso a internet

Nota. Elaboración propia.

Este documento se puede observar en Anexos A, Arquitectura del software.

3.2.5 Documentación de nuevo módulo

La creación de un nuevo módulo lo vemos reflejado dentro de un servicio, es decir, tenemos un módulo de campañas que sería un servicio y el módulo que son los componentes que lo acompañan y permiten hacer uso de módulos como listas de contactos, plantillas etc.

Cuando se crea un nuevo módulo dependiendo de las necesidades que se tengan en la plataforma, se debe tener en cuenta la documentación para FrontEnd, BackEnd y cambio de arquitectura.

Para fronted se deben llenar documentos como plantilla directorio de Apus que es realizada por el equipo de desarrollo, plantilla SRS-especificación de requerimientos que es realizada por coordinación de proyectos, plantilla diseño del software que es realizada por diseño y experiencia, plantilla manual de configuración que es realizada por el equipo de desarrollo, plantilla manual de usuario que es realizada por el equipo de operaciones y soporte y plantilla plan de pruebas Botai que se realizó por operaciones y soporte.

Para BackEnd se deben llenar documentos como plantilla directoria de Apus que es realizada por el equipo de desarrollo, plantilla SRS-especificación de requerimientos que es realizada por coordinación de proyectos, plantilla manual de configuración que es realizada por el equipo de desarrollo y plantilla plan de pruebas Botai que es realizado por operaciones y soporte.

Para cambio de arquitectura se deben llenar documentos como SRS-especificación de requerimientos que es realizado por coordinación de proyectos, plantilla arquitectura del software que es realizado por el equipo de desarrollo, plantilla manual de configuración que es realizado por el equipo de desarrollo y plantilla plan de pruebas Botai que es realizado por operaciones y soporte.

3.2.5.1 Para FrontEnd

Plantilla directoria de apis, se deben seguir los lineamientos establecidos para API específica-FrontEnd para plantilla SRS-especificación de requerimientos, se deben seguir los lineamientos establecidos para documentación de nuevo servicio-FrontEnd.

Para plantilla diseño del software, se deben seguir los lineamientos establecidos para documentación de nuevo servicio-FrontEnd.

Para manual de configuración, se debe seguir los lineamientos establecidos para API específica-BackEnd para plantilla plan de pruebas, se deben seguir los lineamientos establecidos para documentación de nuevo servicio-FrontEnd.

Para plantilla manual de usuario:

Se realiza por parte de operaciones y soporte, donde se busca tener un manual claro de los usos que brinda la plataforma, siendo muy claro en cada paso y en cada opción que se presente, esto con la finalidad de que el cliente pueda tener una guía clara sobre los procesos que brinda la aplicación y su correcto manejo.

Figura 24

Documento manual de usuario



Nota. Elaboración propia.

Este documento consta de cinco elementos muy importantes para el desarrollo del manual de usuario que son:

- **Objetivo del documento:** Breve descripción de lo que se quiere realizar.
- **Descripción general:** Se especifica todo lo relacionado con el proyecto, funcionamiento, acceso, interfaz etc.
- **Perfil de usuario:** Se debe colocar los tipos de usuarios que tienen acceso y los roles que desempeñan con los permisos que le asignan para su posterior interacción.
- **Funcionamiento:** Se debe colocar toda la información correspondiente para el buen uso de la aplicación, un manual donde se detalla lo más mínimo para que así cuando otra

persona haga uso, pueda entender fácilmente los pasos que debe seguir, se debe ser muy riguroso con los pantallazos y la información que se brinda ya que este es un manual de usuario para la aplicación que sale a producción

- Preguntas frecuentes: En este apartado van las preguntas frecuentes que se presentan y sus posibles soluciones, se debe mostrar pantallazos con los errores y las correcciones que realizan

Este documento se puede observar en Anexos A, formato manual de usuario.

3.2.5.2 Para BackEnd. Para la plantilla directorio de Apis, se deben seguir los lineamientos establecidos en APIs específicas.

Para plantilla SRS-especificación de requerimientos, manual de configuración y plantilla plan de pruebas Botai, se deben seguir los lineamientos establecidos en documentación de nuevo servicio-FrontEnd.

3.2.5.2.1 Para cambio de arquitectura. Para plantilla srs-especificación de requerimientos, manual de configuración y plan de pruebas, se deben seguir los lineamientos establecidos en documentación de nuevo servicio- FrontEnd.

Para planificar la arquitectura del software, se deben seguir los lineamientos establecidos en documentación de nuevo servicio-cambio de arquitectura.

3.3 Guía para realizar el proceso de pruebas

Cuando se realiza el desarrollo de una aplicación o software, independientemente de la metodología que se utilice, es de gran importancia realizar un completo y adecuado proceso de pruebas para garantizar que el producto final sea un producto de buena calidad y se pueda determinar si los entregables de desarrollo cumplen con las especificaciones propuestas, los requisitos de su uso y las necesidades del usuario final. El alcance de las pruebas incluye sistemas basados en software, hardware y sus interfaces.

Las pruebas de software pueden realizarse de manera manual o automatizadas. Las pruebas manuales son las que realiza una persona, insertando y organizando todos los datos e interactuando con el software y las API, teniendo en cuenta las herramientas adecuadas para cada caso. Se define el proceso de pruebas:

Figura 25

Proceso de pruebas dependiendo del tipo de desarrollo

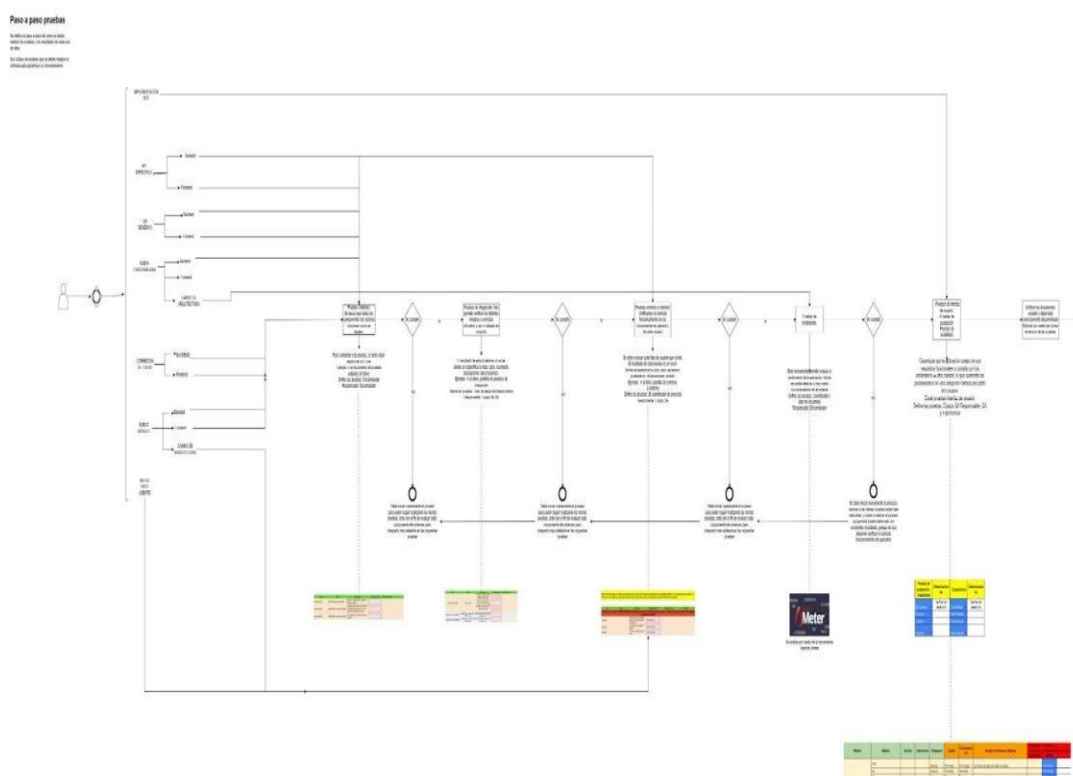
TIPO DE REQUERIMIENTO	SUBTIPO	PRUEBAS						
		PRUEBAS UNITARIAS	PRUEBAS DE INTEGRACIÓN	PRUEBAS DE EXTREMO A EXTREMO	PRUEBAS DE INTERFAZ DE USUARIO	PRUEBAS DE RENDIMIENTO	PRUEBAS DE USABILIDAD	PRUEBAS DE ACEPTACIÓN
		DESARROLLADOR	EQUIPO DE QA	EQUIPO DE QA	EQUIPO DE QA	DESARROLLADOR	EXPERIENCIA	EXPERIENCIA
IMPLEMENTACION BOT							X	X
API ESPECIFICA	FRONTEND	X	X	X	X		X	X
	BACKEND	X	X	X				X
API GENERICA	FRONTED	X	X	X	X		X	X
	BACKEND	X	X	X				X
NUEVA FUNCIONALIDAD	FRONTEND	X	X	X	X		X	X
	BACKEND	X	X	X				X
	CAMBIO DE ARQUITECTURA					X		
CORRECCION DE ERROR	FRONTEND	X	X	X				X
	BACKEND	X	X	X	X		X	X
NUEVO MODULO	FRONTEND	X	X	X				X
	FRONTEND	X	X	X	X		X	X
	CAMBIO DE ARQUITECTURA					X		
NUEVO PASO AGENTE				X	X			

Nota. Elaboración propia.

Para los diferentes tipos de desarrollos que se realiza una tabla en Excel que nos permite ver el tipo de desarrollo o proyecto que se debe implementar y las pruebas que deben ser realizadas para probar los sistemas y brindar unos mejores resultados a los clientes, minimizando errores e incrementando con confiabilidad de los productos entregados.

Figura 26

Diagrama proceso de pruebas



Nota. Elaboración propia.

Se realiza el proceso de las pruebas dependiendo del tipo de desarrollo, especificando qué pruebas se deben realizar para cada uno.

3.3.1 Desarrollos de tipo implementación de BOT

Los BOTS los utilizan las empresas para impulsar las ventas y automatizar procesos con el fin de minimizar tiempos y ser más efectivo con las necesidades de los clientes. Estos están basados en inteligencia artificial y son capaces de mantener una conversación en tiempo real con los clientes ya sea por voz o texto. Se implementan para cualquier tipo y se le dan diferentes usos dependiendo del servicio que presta la empresa como lo son viajes, bancos, ventas etc.

Se realizan pruebas de usabilidad y pruebas de aceptación por parte del equipo de experiencia, donde los resultados de la prueba se plasman en una tabla del documento pruebas interfaz de usuario en la columna L, especificando observaciones si se hayan errores para posteriormente pasamos al equipo de desarrollo y realizar los cambios correspondientes.

Figura 27

Especificación Excel

A Módulo	L Prueba de aceptación - Usabilidad	M Observaciones
Administración de campañas Whatsapp masivas y únicas	Cumple	
	Pasó Revisión	
	Requiere Corrección	
	Sin Verificar	
	Es una mejora	
Cumple		
No cumple		
Cumple		
Cumple		
Cumple		
No cumple		Verificar en desarrollo

Nota. Elaboración propia.

3.3.2 Desarrollos de APIs específicas

Estas APIs son mecanismos que permiten al FrontEnd y BackEnd comunicarse entre sí, permitiendo así el flujo de información continuo que permite programar ciertos funcionamientos de una aplicación determinada dentro del desarrollo del software.

Tipo FrontEnd: Para este tipo de producto se requieren pruebas unitarias, integración, extremo a extremo, interfaz de usuario, usabilidad y aceptación.

Para pruebas unitarias, las realiza del desarrollador del proyecto llenando el formato especificaciones de pruebas, pruebas unitarias.

Figura 28

Plantilla pruebas unitarias

VISTA	CASO	ESCENARIO	PRUEBA/ERROR	OBSERVACIONES	Se deben realizar pruebas a cada elemento que lo componga, el caso que se evalua, escenario, pasa o no. Se anexan ejemplos. Acá se prueban criterios o funciones en específico		
Nueva creación	Permite crear nueva campaña	El boton + debe permitir y mostrar el menú de creación					
Nueva plantilla	Tipo de envío	Se debe seleccionar el tipo de envío para poder crear	PASO				
Listas de envío	Ver que redireccione al menú	Se debe mostrar en pantalla las diferentes configuraciones que se tienen para listas de envíos	NO PASO				

Nota. Elaboración propia.

Se llenan los campos con los ejemplos propuestos, prueba y error especificación si pasa o no la prueba y una casilla de observaciones donde se detallan los casos para luego corregirlos.

Para pruebas de integración, las realiza el equipo de QA luego de realizar las pruebas unitarias, verificando el correcto desempeño del sistema. Se llena el formato especificaciones de pruebas funcionales en la pestaña - pruebas de integración.

Figura 29

Plantilla prueba de integración

VISTA	CASO	ESCENARIO	PRUEBA/ERROR	OBSERVACIONES	Se prueban las integraciones entre módulos como peticiones del frontend al backend y se muestra un ejemplo		
Menú modulo campaña	Subir imagen	Al entrar al modulo de "subir imagen", el sistema debe interactuar con el modulo de APIS para cargar la imagen y retornar la URL donde queda archivada la imagen para usarla en cualquier campaña					
Integraciones con proveedores	ESCENARIO 1: Conexón con el BSP smooch	Envío de campaña unica a través del número "xxx"	PASO				
Integración con bases de datos	Base de datos Posgres	Mostrar listas de campañas, se verifica la correcta comunicación con la base de datos	NO PASO				
Integración entre modulos	Comunicación entre módulo campaña y módulo creación de plantillas	Estando en el módulo de campaña, se permita crear una nueva lista de plantillas, verificando la correcta comunicación entre estos módulos					

Nota. Elaboración propia.

Se llenan los campos con los ejemplos establecidos, pasa o no y una observación en caso de encontrar errores.

Para pruebas de extremo a extremo, las realiza el equipo de QA luego de realizar las pruebas de integración, verificando el correcto desempeño y avance del sistema en cada prueba realizada. Para esta prueba se llena el formato especificaciones de pruebas funcionales en la pestaña - pruebas de extremo a extremo.

En esta prueba evalúa todos los roles de los usuarios que existan en sistema, para probar las funcionalidades están permitidas y cuáles no.

Figura 30*Plantilla prueba extremo a extremo*

VISTA	CASO	ESCENARIO	PRUEBA/ERROR	OBERVACIONES	ESTADO
Tipo de usuario a evaluar: ADMINISTRADOR de la empresa, este tipo de usuario puede observar, crear y modificar toda la información únicamente de su empresa "Compañía"					
Campañas		Permite ver listas de campañas masivas y unicas enviadas por un # telefónico	PASO REVISION		
Listas de contactos		Listar listas de contacto de un número	PASO REVISION		
Plantillas		Crear nueva lista de contacto	NO PASO REVISION ES UNA MEJORA		
Tipo de usuario					
Este tipo de usuario tiene todos los permisos, con todas las que se puedan realizar en cada módulo					
VISTA	CASO	ESCENARIO	PRUEBA/ERROR	OBERVACIONES	ESTADO
Campañas		Sale menú con listado de compañías por ser super usuario	PASO REVISION		
Listas de contactos		Seleccionar compañía que tenga inactivo el servicio pero que lo tuvo activo. Debe mostrar las campañas que creo cuando tuvo activo el servicio	NO PASO REVISION	No muestra el historico de campañas	
Plantillas		Al escoger una compañía, debe actualizar automáticamente las plantillas	PASO REVISION		

Nota. Elaboración propia.

Se llenan los campos establecidos con los ejemplos, prueba/error, observaciones de pruebas que no pasan o mejoras, y el estado que se encuentra es decir que, si ya arreglo, está pendiente o está en proceso por parte del equipo de desarrollo.

Para pruebas de interfaz de usuario, las realiza el equipo de QA luego de realizar la prueba de extremo a extremo, esto con el fin de llevar un proceso exitoso en las pruebas verificando que el sistema está cumpliendo con los objetivos propuestos. Para esta prueba se llena el formulario pruebas de interfaz de usuario.

Figura 31

Plantilla prueba interfaz de usuario

Módulo	Módulo	Sección	Sub Sección	Obligatorio	Estado	Funcionamiento	Arreglos Pendientes o Mejoras	Problemas detectados en pruebas	Estado en pruebas para release	Revisado por
Administración de campañas Whatsapp masivas y únicas	Cancelar			Opcional	Terminado	Terminado			Pasó Revisión	
	Desplegar empresas			Opcional	Terminado	Terminado			Pasó Revisión	
	Clic en empresas y visualiza las campañas (si tiene creadas)			Opcional	Por Arreglar	Por Arreglar	cuando es nueva no permite y se queda cargando		Requiere Corrección	
	Buscar empresa con número de celular			Obligatorio	Por Arreglar	Por Arreglar	cuando es nueva no permite y se queda cargando		Sin Verificar	
	Busca			Opcional	Terminado	Terminado			Es una mejora	
	Descargar CSV			Opcional	Terminado	Terminado			Cumple	
	Filtrar tabla			Opcional	Por Arreglar	Por Arreglar	Solo se puede filtrar por la flecha		No cumple	
	Dar clic en el botón de + para									

Nota. Elaboración propia.

Se llenan los campos con los ejemplos establecidos, estado de prueba, es decir, si pasa o no, sin verificar, es una mejora, revisado por, y por último se explica en una casilla los errores que se encuentran para posteriormente corregirlos en el equipo de desarrollo.

Pruebas de usabilidad y pruebas de aceptación por parte del equipo de experiencia al finalizar el proceso, esto con el fin de verificar si cumple con los requerimientos y con las interacciones que puede tener el cliente con la plataforma. El proceso de estas pruebas es el mismo definido anteriormente para implementación de BOT.

Tipo BackEnd: Se realizan pruebas unitarias, integración, extremo a extremo, aceptación.

Para pruebas unitarias, integración, extremo a extremo se deben tener en cuenta los lineamientos dados en desarrollos de APIs específicas.

Para pruebas de aceptación se deben seguir los lineamientos de implementación de BOT.

3.3.3 Desarrollos de APIs genérica

Estas las utilizan personas ajenas a la organización pueden acceder a ellas, pero solo aquellos con permisos exclusivos. Por lo general, este acceso especial se otorga a ciertos terceros para facilitar colaboraciones comerciales estratégicas.

3.3.3.1 Tipo FrontEnd. Para este tipo de producto se requieren pruebas unitarias, integración, extremo a extremo, interfaz de usuario, usabilidad y aceptación.

Para pruebas unitarias, integración y extremo a extremo se deben seguir los lineamientos de desarrollo APIs específicos.

Para pruebas de interfaz de usuario, usabilidad y aceptación se deben seguir los lineamientos de desarrollo de APIs específicas.

3.3.3.2 Tipo BackEnd. Se realizan pruebas unitarias, integración, extremo a extremo y aceptación.

Para pruebas unitarias, integración, extremo a extremo se deben seguir los lineamientos de desarrollo de APIs específicas.

Para la prueba de aceptación se debe seguir los lineamientos de desarrollos de tipo implementación BOT.

3.3.4 Desarrollos de nuevas funcionalidades

Dentro del sistema se pueden crear nuevas funcionalidades teniendo en cuenta las necesidades de los clientes, como nuevas funcionalidades en canales de difusión o poder subir otro tipo de archivos como PDF, Word etc. Para poder aceptar nuevos procesos, se deben realizar las pruebas que se requieran para verificar que la nueva funcionalidad cumpla con su objetivo y así se minimizan errores y reprocesos.

3.3.4.1 Tipo FrontEnd. Para este tipo de producto se requieren pruebas unitarias, integración, extremo a extremo, interfaz de usuario, usabilidad y aceptación.

Para pruebas unitarias, integración y extremo a extremo se deben seguir los lineamientos de desarrollo APIs específicos.

Para pruebas de interfaz de usuario, usabilidad y aceptación se deben seguir los lineamientos de desarrollo de APIs específicas.

3.3.4.2 Tipo BackEnd. Se realizan pruebas unitarias, integración, extremo a extremo y aceptación.

Para pruebas unitarias, integración, extremo a extremo se deben seguir los lineamientos de desarrollo de APIs específicas.

Para la prueba de aceptación se debe seguir los lineamientos de desarrollos de tipo implementación BOT.

3.3.4.3 Tipo cambio de arquitectura. Se realizan las pruebas de rendimiento por parte de desarrollador para verificar el estado del sistema, utilizando la herramienta Apache Jmeter.

3.3.5 Corrección de error

Cuando se presentan errores en el sistema, se procede a corregir cada uno de ellos sin importar si son de FrontEnd o BackEnd y se deben realizar las pruebas para verificar el estado de la corrección, si cumple o no.

3.3.5.1 Tipo FrontEnd. Para este tipo de producto se requieren pruebas unitarias, integración, extremo a extremo y aceptación.

Para pruebas unitarias, integración y extremo a extremo se deben seguir los lineamientos de desarrollo APIs específicos.

Para pruebas de aceptación se deben seguir los lineamientos de desarrollo de APIs específicas.

3.3.5.2 Tipo BackEnd. Se realizan pruebas unitarias, integración, extremo a extremo, interfaz de usuario, usabilidad y aceptación.

Para pruebas unitarias, integración, extremo a extremo se deben seguir los lineamientos de desarrollo de APIs específicas.

Para la prueba de usabilidad y aceptación se debe seguir los lineamientos de desarrollos de tipo implementación BOT.

3.3.6 Nuevo módulo

Son creaciones que pueden surgir dentro de un sistema como por ejemplo el servicio de campañas, que contiene varios módulos dentro el que prestan diferentes servicios. Se deben realizar las pruebas para ver el correcto funcionamiento y empalme con módulos existentes y no causen problemas o conflictos con otros componentes.

3.3.6.1 Tipo FrontEnd. Para este tipo de producto se requieren pruebas unitarias, integración, extremo a extremo y aceptación.

Para pruebas unitarias, integración y extremo a extremo se deben seguir los lineamientos de desarrollo APIs específicos.

Para pruebas de aceptación se deben seguir los lineamientos de desarrollo de APIs específicas.

3.3.6.2 Tipo tipo BackEnd. Se realizan pruebas unitarias, integración, extremo a extremo, interfaz de usuario, usabilidad y aceptación.

Para pruebas unitarias, integración, extremo a extremo se deben seguir los lineamientos de desarrollo de APIs específicas.

Para la prueba de interfaz de usuario se deben seguir los lineamientos de desarrollo de APIs específicas

Para la prueba de usabilidad y aceptación se debe seguir los lineamientos de desarrollos de tipo implementación BOT.

3.3.6.3 Tipo cambio de arquitectura. Se realizan las pruebas de rendimiento por parte de desarrollador para verificar el estado del sistema, utilizando la herramienta Apache Jmeter.

3.3.7 Nuevo paso agente

Dependiendo de la necesidad de la empresa de pasar del chatbot a un trabajador de la empresa para solucionar errores que el Bot no pudo, se implementan el paso agente, donde se

puede utilizar uno existente o si se requiere uno nuevo, se deben implementar las pruebas correspondientes para verificar el correcto desempeño del mismo.

Para este tipo de producto se requieren pruebas de extremo a extremo y interfaz de usuario.

Para pruebas de extremo a extremo, que son realizadas por el equipo de QA, verificando el correcto desempeño del sistema. Para esta prueba se llena el formato especificaciones de pruebas funcionales en la pestaña - pruebas de extremo a extremo.

En esta prueba se evalúa todos los roles de los usuarios que existan en sistema, para probar las funcionalidades están permitidas y cuáles no.

Figura 32

Plantilla prueba extremo a extremo

VISTA	CASO	ESCENARIO	PRUEBA/ERROR	OBERVACIONES	ESTADO
Tipo de usuario a evaluar: ADMINISTRADOR de la empresa, este tipo de usuario puede observar, crear y modificar toda la información únicamente de su empresa "Compañía"					
Campañas		Permite ver listas de campañas masivas y unicas enviadas por un # telefónico	PASO REVISION		
Listas de contactos		Listar listas de contacto de un número	PASO REVISION		
Plantillas		Crear nueva lista de contacto	NO PASO REVISION		
Tipo de usuario					
Este tipo de usuario tiene todos los permisos, con todas las funcionalidades que se puedan realizar en cada módulo					
VISTA	CASO	ESCENARIO	PRUEBA/ERROR	OBERVACIONES	ESTADO
Campañas		Sale menú con listado de compañías por ser super usuario	PASO REVISION		
Listas de contactos		Seleccionar compañía que tenga inactivo el servicio pero que lo tuvo activo. Debe mostrar las campañas que creo cuando tuvo activo el servicio	NO PASO REVISION	No muestra el historico de campañas	
Plantillas		Al escoger una compañía, debe actualizar automáticamente las plantillas	PASO REVISION		

Notas. Elaboración propia.

Se llenan los campos establecidos con los ejemplos, prueba/error, observaciones de pruebas que no pasan o mejoras, y el estado que se encuentra es decir que, si ya arreglo, está pendiente o está en proceso por parte del equipo de desarrollo.

Para pruebas de interfaz de usuario, las realiza el equipo de QA luego de realizar la prueba de extremo a extremo, esto con el fin de llevar un proceso exitoso en las pruebas verificando que el sistema está cumpliendo con los objetivos propuestos. Para esta prueba se llena el formulario pruebas de interfaz de usuario.

Figura 33

Plantilla prueba interfaz de usuario

Módulo	Módulo	Sección	SubSección	Obligatorio	Estado	Funcionamiento	Arreglos Pendientes o Mejoras	Problemas detectados en pruebas	Estado en pruebas para release	Revisado por
Administración de campañas Whatsapp masivas y únicas	Cancelar			Opcional	Terminado	Terminado			Paso Revisión	
	Desplegar empresas			Opcional	Terminado	Terminado			Paso Revisión	
	Clic en empresas y visualiza las campañas (si tiene creadas)			Opcional	Por Arreglar	Por Arreglar	cuando es nueva no permite y se queda cargando		Requiere Corrección	
	Buscar empresa con número de celular			Obligatorio	Por Arreglar	Por Arreglar	cuando es nueva no permite y se queda cargando		Sin Verificar	
	Busca			Opcional	Terminado	Terminado			Es una mejora	
	Descargar CSV			Opcional	Terminado	Terminado			Cumple	
	Filtrar tabla			Opcional	Por Arreglar	Por Arreglar	Solo se puede filtrar por la flecha		No cumple	
	Dar clic en el botón de + para									

Nota. Elaboración propia.

Se llenan los campos con los ejemplos establecidos, estado de prueba, es decir, si pasa o no, sin verificar, es una mejora, revisado por, y por último se explica en una casilla los errores que se encuentran para posteriormente corregirlos en el equipo de desarrollo.

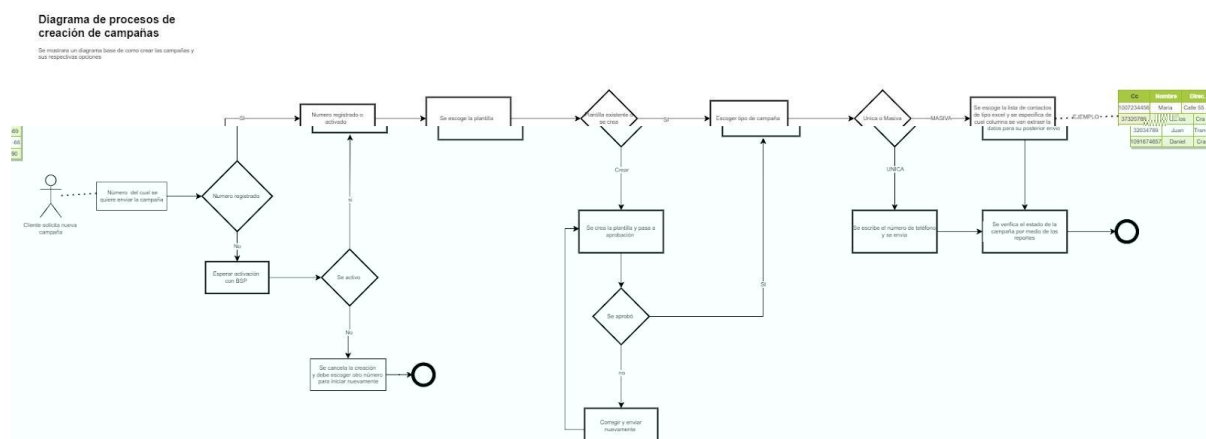
3.4 Casos de uso

3.4.1 Proceso creación de campañas

En este diagrama se puede observar cual es el proceso que se debe seguir para la creación de una nueva campaña, donde se especifica el paso a paso y la finalidad que se le debe dar.

Figura 34

Diagrama proceso creación de campañas



Nota. Elaboración propia.

Esto va ser de mucha utilidad para el nuevo personal que ingrese a la empresa ya que se están detallando los procesos que se deben realizar.

Entonces, tenemos el cliente cuando requiere el servicio de una nueva campaña. Se procede a ingresar a la plataforma y se selecciona el número de teléfono. Si el número de

teléfono no está registrado, se procede a activarlo por medio de BSP y se espera confirmación, de no confirmarse se debe seleccionar otro número y realizar el mismo proceso.

Cuando se tenga el número registrado o activado en la plataforma, se procede a escoger la plantilla que es donde va el mensaje detallado que le llegará a los usuarios finales, sino existe la plantilla se debe crear la plantilla y esperar aprobación.

Luego de que ya se tenga la plantilla o de que también se haya aprobado una nueva, se escoge el tipo de campaña, donde hay campañas únicas y masivas, esto depende de la necesidad del cliente.

Si se escoge campaña única, se deben llenar los datos como número de teléfono al cual va dirigida, se anexa la plantilla y se envía, luego de esto se puede verificar el estado de la campaña por medio de los reportes de campañas.

Si se escoge campaña masiva, se debe escoger la lista de contactos a los cuales va dirigida, donde se sube un archivo de tipo Excel a la plataforma donde contenga todos los datos que el sistema puede tomar automáticamente y se le indica de cuál columna se deben seleccionar los datos para extraerlos y crear la lista con los números telefónicos que se tienen.

Luego de este proceso se envía la campaña y se puede verificar el envío y estado actual de la misma en el módulo de reportes, donde se verá a detalle a cuáles números de envío, si llego a todos, o si hubo errores en los envíos.

Validar los lineamientos de documentación y la guía para planear y ejecutar las pruebas propuestos a través de un caso de uso que permita brindar soporte al equipo de desarrollo.

3.5 Diseño de plan de pruebas para validar los lineamientos

Ver **Apéndice C**, caso de uso, plan de pruebas.

En **Apéndice C**, se encuentran los documentos de caso de uso donde se implementaron plantillas que se crearon.

4. Diagnostico final

Luego de determinar los lineamientos y pruebas que se buscan implementar en Smart Data & Automation, se obtienen conocimientos sobre plataformas organizacionales que están siendo implementadas en grandes empresas a nivel nacional e internacional que permiten el manejo de clientes de una manera más rápida y efectiva, entendiendo las necesidades que se tienen y poder actuar en menores tiempos con información precisa.

Se realiza una entrega de documentos y plantillas lo cual permite a la empresa analizar toda la información y adaptarla según las necesidades de cada proyecto y de esta manera tener unos desarrollos de software con soporte de documentación y pruebas para poder mejorar la entrega en los productos a los clientes, minimizando riesgos y daños en los sistemas gracias a las pruebas que se plantean para cada uno de ellos.

Estos lineamientos permiten realizar nuevos procesos internos de una forma más rápida, segura y eficiente, ya que al momento de cumplir los requerimientos de los clientes se brinda un seguimiento continuo a los proyectos y sus avances, minimizando los riesgos y aumentando la productividad.

5. Conclusiones

- Los lineamientos de documentación permiten cumplir a cabalidad todos los requerimientos de los clientes, identificando cada riesgo y tomando en cuenta cada posible solución al mismo, para que los requerimientos post-producción, estén completamente cubiertos.
- Los formatos entregados se realizan con base a estándares solicitados en la empresa y con aprobación de la misma. Donde para cada tipo de desarrollo se implementa cada formato con sus respectivas características.
- El diseño del plan de pruebas, permitió determinar los tipos de pruebas a los cuales deben ser sometidos los sistemas para poder certificar su integridad y confiabilidad, dependiendo del tipo de desarrollo que se esté realizando.
- Se elaboró el proceso completo de documentación y pruebas, mostrando en diagramas cual debe ser el flujo a seguir dentro de la empresa, aplicando las etapas en cada proceso y los documentos resultantes de cada uno.


6. Recomendaciones

Se recomienda analizar la posibilidad de fortalecer el equipo de pruebas con el fin de poder estudiar todo el software que va ser entregado a un cliente y poder implementar todas las pruebas propuestas, con el fin de brindar productos de máxima calidad.

Se recomienda fortalecer el manejo de metodologías ágiles para gestiones de nuevos proyectos y creación de nuevos módulos en la plataforma Botai, esto con el fin de minimizar tiempos, reducir los riesgos y mejorar los equipos de trabajo, lo cual permite alcanzar todos los objetivos propuestos para cada nuevo desarrollo.

Se recomienda realizar estudios y seguimientos a los sistemas que se tienen actualmente con el fin de verificar el estado de los mismos, garantizando así un software actualizado y confiable.

Referencias

Agustin Navarro Galdon. (2022, 20 mayo). Django documentación automática con Swagger .

YouTube. https://www.youtube.com/watch?v=IZ_mIYSw_Vc

Atlassian. (s. f.). Importancia de la documentación | The Workstream.

<https://www.atlassian.com/es/work-management/knowledge-sharing/documentation/importance-of-documentation>

Colaboradores de Wikipedia. (2022, 13 septiembre). ISO/IEC 12207. Wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/ISO/IEC_12207

Diseño e implementación de un asistente virtual (chatbot) para ofrecer atención a los clientes de una aerolínea mexicana por medio de sus canales conversacionales. (2020, mayo).

INFOTEC. Recuperado 6 de diciembre de 2022, de

https://infotec.repositorioinstitucional.mx/jspui/bitstream/1027/402/1/INFOTEC_MGITIC_FAGO_27082020.pdf

Documentación de código. (s. f.). <http://www.dit.upm.es/%7Eepe/doc/adsw/base/doc/doc.htm>

documentación de código en react - Google Zoeken. (s. f.).

<https://www.google.com/search?q=documentacion+de+codigo+en+react>

IEEE Standard for Software and System Test Documentation. in IEEE Std 829-2008 , vol., no., pp.1-150, 18 July 2008, doi: 10.1109/IEEESTD.2008.4578383.

Influencia de la publicidad digital basada en WhatsApp Marketing en el nivel de satisfacción para el fortalecimiento de la fidelización de clientes de las tiendas de barrio de la localidad de Chapinero. (2021, 17 noviembre). repository.javeriana.edu.co. Recuperado 8 de enero de 2023, de

<https://repository.javeriana.edu.co/bitstream/handle/10554/58913/Camila%20Andrea%20Parada%20Tuta%20-%20Trabajo%20de%20Grado.pdf?sequence=3>

ISO/IEC/IEEE 12207:2017. (2021, 4 febrero). ISO. <https://www.iso.org/standard/63712.html>

ISO/IEC/IEEE 29119. (2017, 16 marzo). ISO/IEC/IEEE 29119 El nuevo estándar internacional para pruebas de software. Recuperado 3 de octubre de 2022, de

<https://in2test.lsi.uniovi.es/gt26/presentations/Tuya-ISO29119-Seguridad-20170316.pdf>

John Ortiz Ordoñez. (2022, 9 marzo). Lógica de Programación: 51 Metodología - Etapa 9 - Documentación Externa de un Programa. YouTube.

https://www.youtube.com/watch?v=b4ZA9F_vPpE

Medina, I. F. (2023, 17 febrero). Los estándares de calidad del software más importantes. Blog de Hiberus Tecnología. <https://www.hiberus.com/crecemos-contigo/los-estandares-de-calidad-del-software-mas-importantes/>

MoonCode. (2021, 2 mayo). Aprende A Documentar Tu API Con Swagger 📖 🍷 - Learn To Document Your API With Swagger 🚀. YouTube.

<https://www.youtube.com/watch?v=SdsaZ-t1QwA>

Nombres de variable. (s. f.). <https://www.ibm.com/docs/es/spss-statistics/saas?topic=view-variable-names>

Política de Protección de Datos Personales. (2023, 6 marzo). Ministerio de Ambiente y Desarrollo Sostenible. <https://www.minambiente.gov.co/politica-de-proteccion-de-datos-personales/>

R. (2014, 24 enero). Diez consejos para mejorar tus comentarios de código fuente. Genbeta.

Rubenfa. (2014b). Diez consejos para mejorar tus comentarios de código fuente. Genbeta.

<https://www.genbeta.com/desarrollo/diez-consejos-para-mejorar-tus-comentarios-de-codigo-fuente>

SIB Digital. (s. f.).

<https://login.sibdigital.ufpso.edu.co/login?qurl=https://www.sciencedirect.com%2fscience%2farticle%2fabs%2fpii%2fS0950584903001988%3fvia%3dihub>

SIB Digital. (s. f.).

<https://login.sibdigital.ufpso.edu.co/login?qurl=https://www.sciencedirect.com%2fscience%2farticle%2fpii%2fS1405774314703506>

SIB Digital. (s. f.).

<https://login.sibdigital.ufpso.edu.co/login?qurl=https://www.sciencedirect.com%2fscience%2farticle%2fpii%2fS0212656714000067>

Smart Data & Automation. (2020). Smart Data & Automation. Smart Data & Automation.

Recuperado 11 de noviembre de 2022, de <https://www.smartdataautomation.com/>

User, S. (s. f.). CECOLDA - Centro Colombiano del Derecho de Autor - LEY 23 DE 1982

SOBRE DERECHO DE AUTOR. <http://www.cecolda.org.co/index.php/derecho-de-autor/normas-y-jurisprudencia/normas-nacionales/124-ley-23-de-1982-sobre-derecho-de-autor>

Apéndice A. Formatos

a) Arquitectura del software

Cuando se tiene un nuevo proyecto que involucre empezar desde cero, se realiza el formato arquitectura del software ya que este sería el plan del software, donde se puede ver cómo está compuesto, como es su funcionalidad, bases de datos con las cuales trabaja, y demás elementos que intervienen para su correcto funcionamiento.



Plantilla documento de arquitectura



HISTORIAL DE CAMBIOS

ID	Fecha	Usuario	Descripción	Revisado por	Revisado en
1.0	2018/02/28	Tade	Creación del documento		



Tabla de contenido

1. RESTRICCIONES ARQUITECTURALES	4
1.1. Motivaciones de Negocio	4
1.2. Restricciones de Tecnología	5
1.3. Restricciones de Negocio	5
1.4. Atributos de Calidad	6
1.4.1. Árbol de Utilidad	6
1.4.1.1. Distribución de atributos de calidad por prioridad	6
1.4.1.2. Atributos de calidad eficiencia	7
1.4.1.3. Atributos de calidad mantenimiento	7
1.4.1.4. Atributos de calidad seguridad	8
1.4.1.5. Atributos calidad usabilidad	8
1.4.1.6. Atributos de calidad funcionalidad	8
1.4.2. Escenarios de calidad	8
2. PUNTO DE VISTA Y MODELO ARQUITECTURALES	10
2.1. Punto de vista de cliente	11
2.1.1. Modelo analítico	11
2.1.2. Modelo de red	12
2.1.3. Modelo de dependencia tecnológica	13
2.2. Punto de vista de información	14
2.2.1. Modelo de estructuras estadísticas de datos	14
2.2.2. Modelo flujo de información	15
3. RELACIONES ENTRE LOS PUNTO DE VISTA	16
3.1. Punto de vista funcional	16



1. RESTRICCIONES ARQUITECTURALES

1.1. Motivadores de Negocio

Tabla ejemplo:

Tabla Ejemplo de negocio 1

Descripción	Impacto/Justificación del negocio
Implementación de un nuevo sistema	Conectar al sector y generar ingresos para cubrir el costo de estos atributos, recuperar los costos de implementación/mantenimiento del área de negocio y generar nuevos ingresos. Se generará una nueva línea de productos para mejorar la competitividad de los generados actualmente. Se generará una nueva línea de productos.

Pueden existir varias tablas con varios motivadores de negocio.

1.2. Restricciones de Tecnología

Tabla Ejemplo de tecnología 1

Restricción	Tip	Justificación
Implementación de un nuevo sistema	Tip	Justificación



Pueden existir varias tablas con varias restricciones de tecnología.

1.3. Restricciones de Negocio

Tabla Ejemplo de negocio 2

Restricción	Tip	Justificación
Implementación de un nuevo sistema	Tip	Justificación

1.4. Atributos de Calidad

1.4.1. Árbol de Utilidad



1.4.1.1. Distribución de atributos de calidad por prioridad

Tabla Atributos de Calidad

Tip	ID	Descripción
Accesibilidad	AC-001	Accesibilidad de usuarios
Seguridad	SG-001	Seguridad de usuarios
Usabilidad	US-001	Usabilidad de usuarios
Funcionalidad	FN-001	Funcionalidad de usuarios
Funcionalidad	FN-002	Funcionalidad de sistemas
Funcionalidad	FN-003	Funcionalidad de procesos
Funcionalidad	FN-004	Funcionalidad de datos
Funcionalidad	FN-005	Funcionalidad de interfaces
Funcionalidad	FN-006	Funcionalidad de servicios
Funcionalidad	FN-007	Funcionalidad de productos
Funcionalidad	FN-008	Funcionalidad de soluciones
Funcionalidad	FN-009	Funcionalidad de experiencias
Funcionalidad	FN-010	Funcionalidad de relaciones
Funcionalidad	FN-011	Funcionalidad de alianzas
Funcionalidad	FN-012	Funcionalidad de asociaciones
Funcionalidad	FN-013	Funcionalidad de consorcios
Funcionalidad	FN-014	Funcionalidad de redes
Funcionalidad	FN-015	Funcionalidad de grupos
Funcionalidad	FN-016	Funcionalidad de comunidades
Funcionalidad	FN-017	Funcionalidad de organizaciones
Funcionalidad	FN-018	Funcionalidad de instituciones
Funcionalidad	FN-019	Funcionalidad de empresas
Funcionalidad	FN-020	Funcionalidad de organizaciones
Funcionalidad	FN-021	Funcionalidad de instituciones
Funcionalidad	FN-022	Funcionalidad de empresas
Funcionalidad	FN-023	Funcionalidad de organizaciones
Funcionalidad	FN-024	Funcionalidad de instituciones
Funcionalidad	FN-025	Funcionalidad de empresas

1.4.1.2. Atributos de calidad eficiencia

Tabla Atributos de Calidad Eficiencia

Restricción	ID	Justificación	Impacto
Implementación de un nuevo sistema	TI-001	Implementación de un nuevo sistema	1
Implementación de un nuevo sistema	TI-002	Implementación de un nuevo sistema	2
Implementación de un nuevo sistema	TI-003	Implementación de un nuevo sistema	3
Implementación de un nuevo sistema	TI-004	Implementación de un nuevo sistema	4



1.4.1.3. Atributos de calidad mantenimiento

Tabla Atributos de Calidad Mantenimiento			
Atributo de Calidad	Id	Descripción	Importancia
Facilidad de mantenimiento	SM-001	Una nueva funcionalidad no debe disminuir el desempeño más de un 5% respecto al anterior.	3
Flexibilidad	SM-002	El sistema debe ser fácil de probar por parte de los desarrolladores y usuarios.	2
Flexibilidad de cambios	SM-003	El sistema debe permitir cambios rápidos e a bajo costo.	2

1.4.1.4. Atributos de calidad seguridad

Tabla Atributos de Calidad Seguridad			
Atributo de Calidad	Id	Descripción	Importancia
Disponibilidad	SC-001	Los usuarios deben poder acceder al sistema en todo momento.	3
Integridad	SC-002	Cada usuario debe tener un rol asignado que permita acceder a las acciones de la aplicación.	3

1.4.1.5. Atributos de calidad usabilidad

Tabla Atributos de Calidad Usabilidad			
Atributo de Calidad	Id	Descripción	Importancia
Facilidad de aprendizaje	US-001	La interfaz debe ser amena y fácil de usar por el usuario.	3
Facilidad de uso	US-002	Los usuarios deben poder realizar sus tareas de manera eficiente en la aplicación.	3



1.4.1.6. Atributos de calidad funcionalidad

Tabla Atributos de Calidad Funcionalidad			
Atributo de Calidad	Id	Descripción	Importancia
Fiabilidad	FC-001	El usuario debe poder acceder desde un dispositivo móvil a la aplicación.	3

1.4.2. Escenarios de calidad

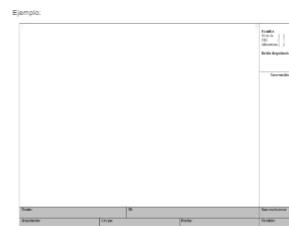
Formulario de Calidad 1			
Atributo de Calidad	Id	Requisición	Modelo de campo
Disponibilidad	FC-001	Permite a los usuarios de campo visualizar la ubicación de los tiempos en el mapa, desde la lista por orden alfabético o interactivo en la ubicación deseada.	Mapa
Facilidad de uso	FC-002	Visualización de tiempos a mostrar	Tabla
Facilidad de uso	FC-003	Columna	Tabla
Facilidad de uso	FC-004	Según condiciones normales	Tabla
Facilidad de uso	FC-005	Ubicación geográfica de tiempos a mostrar	Tabla
Facilidad de uso	FC-006	Temperatura	Tabla

Pueden existir varios escenarios de calidad.

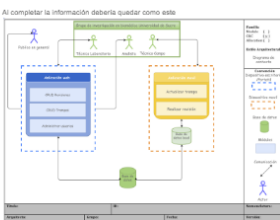


2. PUNTOS DE VISTA Y MODELOS ARQUITECTURALES

Se deben presentar los diagramas con la arquitectura propuesta enmarcado en los diferentes puntos de vista. Para definir estos modelos de arquitectura se deben tener en cuenta los escenarios de calidad.



Link de la plantilla modelos arquitecturales: [Plantilla Modelos Arquitecturales](#)

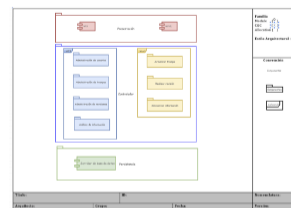


2.1 Puntos de vista

2.1.1 Punto de vista lógico

2.1.1.2 Modelo funcional

Ejemplo:



Se debe explicar detalladamente cada componente, dando así un breve resumen de lo que hay en el diagrama que se presenta.

Ejemplo:
A continuación se encuentran los modelos relacionados con el punto de vista funcional, se maneja un esquema de arquitectura de 3 niveles: modelo, vista y controlador. El nivel superior está compuesto únicamente por el componente de presentación, el nivel intermedio está compuesto por el componente de análisis y por el componente de operaciones básicas y el nivel inferior está compuesto por el componente de comunicaciones y por el componente de persistencia.

2.1.2. punto de vista física

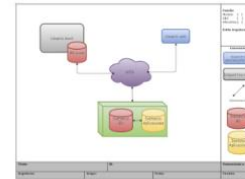
2.1.2.1 Despliegue

Ejemplo:



2.1.2.2 Modelo de red

Ejemplo:



2.1.4. Modelo de dependencia tecnológica

Ejemplo:

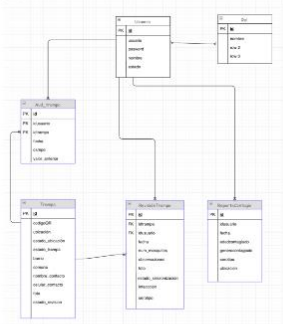
Componente	Solución
Servidor de aplicaciones	Tomcat?
Persistencia	PostgreSQL
Plataforma	Erafe - Java
Seguridad y autenticación	spring-security-core



2.2. Punto de vista de información

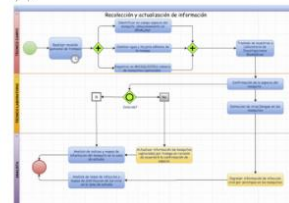
2.2.1. Modelos de estructuras estáticas de datos

Ejemplo:



2.2.2. Modelos flujo de información

Ejemplo:



I. Modelo arquitectural del software

Es la estructura del software donde podremos ver la estructura del mismo, que componentes tiene, propiedades, componentes y cómo se relacionan.

			Familia () Module () Cdc () Allocation ()
			Estilo Arquitectural : Convención
Titulo:		ID:	Nomenclatura:
Arquitecto:	Grupo:	Fecha:	Versión: <input type="text"/>

b) Especificación general

Se encuentra toda la información correspondiente al desarrollo, como nombre, objetivo del proyecto, descripción breve del proyecto, tipo de lenguaje, tipo de desarrollo, indicaciones relevantes, cliente al cual para dirigió, coordinador de proyecto, equipo de trabajo.

Lo que se busca es tener una guía clara sobre los desarrollos que se implementan, haciendo más fácil la comprensión del sistema para saber su propósito con el cual se creó y detalles de su funcionamiento.

d) Documento especificación de APIs

Cuando se realiza una nueva creación de APIs, se debe solicitar al equipo de desarrollo la verificación de la misma a través de unos formatos establecidos, ya sea para una nueva creación o solicitar al equipo de desarrollo un cambio de una existente.



DOCUMENTO ESPECIFICACIÓN APIS



Índice:
 1. Estructura para realizar la solicitud al equipo de desarrollo
 2. Ejemplos de Estructura para realizar la solicitud al equipo de desarrollo



1. Estructura para realizar la solicitud al equipo de desarrollo

Nombre del Proyecto			
Link a Ruta del bot			
Fecha de entrega	Tiene fecha límite ()	Solicitud de Estimación de Recursos ()	
Implementador			
Proceso			
Tipo de Solicitud	API Rest/GraphQL ()	API GraphQL ()	Don ()
	Funcionalidad ()	Funcionalidad ()	Funcionalidad ()
Objetivo	COMO	NECESITO	QUISIERA
Datos de entrada			
Escenarios	Casos	Salidas esperadas	
Procesos adicionales			
Datos requeridos			



2 Ejemplos de Estructura para realizar la solicitud al equipo de desarrollo

Nombre del Proyecto		Módulo de Iniciar	
Link al Ruta del bot		https://www.smartdata.net/colabora/colaborador/2164424	
Fecha de entrega	Tiene fecha límite ()	Solicitud de Estimación de Recursos ()	
Implementador	Pablo		
Proceso	Envío de información inicial -> mensaje de bienvenida		
Tipo de Solicitud	API Rest/GraphQL ()	API GraphQL ()	Don ()
	Funcionalidad ()	Funcionalidad ()	Funcionalidad ()
Objetivo	COMO	NECESITO	QUISIERA
Datos de entrada	*Tipo de documento (Obligatorio) *Nombre de documento (Obligatorio)		
Escenarios	Casos	Salidas esperadas	
	El usuario hace más de 4 intentos mensaje: "Te he más de 4" 2022/11/11		
	El usuario hace menos de cuatro intentos mensaje: "Te he más de 4" 2022/11/11		
Procesos	N/A		



adicionales			
Datos requeridos	N/A		
Proceso	Envío de información inicial -> mensaje de info en la EC de registro inicial		
Tipo de Solicitud	API Rest/GraphQL ()	API GraphQL ()	Don ()
	Funcionalidad ()	Funcionalidad ()	Funcionalidad ()
Objetivo	COMO	NECESITO	QUISIERA
Datos de entrada	*Autentificación (Obligatorio) *Tipo de documento (Obligatorio) *Nombre de documento (Obligatorio) *Fecha de nacimiento (Obligatorio) *Fecha de expedición del documento (Obligatorio) *Correo electrónico (Obligatorio) *Número de celular actual (Obligatorio) *Número de celular nuevo (Obligatorio)		
Escenarios	Casos	Salidas esperadas	
	Dirección de la tarea programada para enviar info inicial. De cada crear una tarea en donde se debe de enviar información de cada caso por empresa y resultado de cada caso de la EC de inicio de sesión con una respuesta de Smart Data & Automation desde donde se debe de crear una tarea programada de la manera que el 100% se recibe cada 20 min o cada 1h. En las cuales cuando se ejecuta debe de programarse de manera que el 100% de los datos se envíen cada 20 min.		
Procesos	N/A		

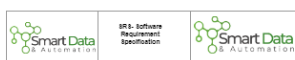


adicionales			
Datos requeridos	Dirección de acceso a SS de Modulo de inicio de sesión y perfil de usuario.		

I. SRS- Especificación de requerimientos

Para la construcción de nuevos proyectos, se debe realizar un levantamiento de requerimientos muy detallado ya que de esto depende el buen desarrollo del mismo.

- Se detalla el objetivo general del proyecto.
 - Descripción global como diseño, características etc.
 - Flujos de uso de negocio.
 - Escenarios operacionales.
 - Requerimientos específicos.
 - Requerimientos no funcionales.
- Entregables como documentos de diseño, manuales, código fuente, archivos ejecutables.



SRS - SOFTWARE REQUIREMENT SPECIFICATION

Autor de proyecto
Número del proyecto

Especificación de requerimientos

Especificación de requerimientos

Especificación de requerimientos



PROYECTO	"Smart Data & Automation"
Actividades	"creación de nueva campaña"

1. INTRODUCCIÓN

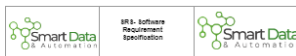
Objetivo general

1.1. Propósito

1.2. Alcance

1.3. Definiciones, acrónimos y abreviaciones

Especificación de requerimientos



Historial de cambios

	Fecha	Sección Modificada	Descripción cambios	Responsable (s)
1.0	10/10/01	"Índice"	"Creación de formato..."	"Rubén Parada"
1.1				
1.2				



Contenido

1. INTRODUCCIÓN	6
Objetivo general	6
1.1. Propósito	6
1.2. Alcance	6
1.3. Definiciones, acrónimos y abreviaciones	6
2. DESCRIPCIÓN GLOBAL	8
2.1. Descripción general (¿Que es?)	8
2.2. Producto (que se entrega)	8
2.3. Diseño conceptual de software (¿Que es y cómo funciona?)	7
2.4 Características de usuario	8
3. REQUERIMIENTOS E SPECIFICOS	8
3.1 Métodos y eventos	8
3.2 Tabla de requerimientos funcionales	8
4. REQUERIMIENTO NO FUNCIONALES	8
4.1. Usabilidad	8
4.2. Disponibilidad	8
4.3. Desempeño	10
4.4. Portabilidad	10
4.5. Seguridad	11
4.7. Ambiente, sistema operativo	11
4.8. Restricciones de diseño	11
4.9. Requerimientos de documentación y ayuda en línea del sistema	11
4.10. Interfaces	11
4.11. Estandarización	12
5. Entregables	12
5.1. Documentos de diseño	12
5.2. Manual de instalación	12
5.3. Manual de usuario	12
5.4. Código fuente	12
5.5. Archivos ejecutables	12



2. DESCRIPCIÓN GLOBAL

2.1. Descripción general (¿Que es?)

Cual es el propósito general del proyecto, que se hace, cómo se hace y para que se hace?

2.2. Producto (Que se entrega)

Se debe redactar toda la información que se va entregar al cliente final como diseño, desarrollo, parámetros específicos y demás aspectos importantes...

Se evidencia el proceso de desarrollo	
Documento de diseño detallado	- Documento de especificación de requerimientos - Documento de diseño de software - Y demás documentos importantes en el desarrollo
Entrega final	- Implementación final - Manuales de usuarios - Instalaciones de los elementos de software del servidor - Instalaciones del software de cliente - Información importante para el cliente - Etc.



2.3. Diseño conceptual de software (¿Que es y cómo funciona?)

Se realiza una tabla con información detallada, ejemplo:

Definición de sistemas	
Tipo del sistema	¿Que tipo de aplicación es, donde se instala, donde está el servidor, cuales navegadores son compatibles, se entrega incluye instalaciones y código fuente?
Requerimientos del sistema para operación	Son especificaciones mínimas para las terminales de acceso como: ¿Tipo de procesador? ¿Ram mínima? ¿Tamaño de resolución de pantalla? ¿Compatible con varios sistemas operativos, cuales?
Requerimientos de red	¿Que tipo de conexión se usa, TCP/IP, BGP etc.? ¿Qué firewalls deben estar habilitados? ¿Ancho de banda mínimo?
Capacidad	¿Cuántos usuarios se pueden crear? ¿Cuántos usuarios pueden acceder al mismo tiempo?
Licenciamiento	¿Es propio, que tipo de licencia, derechos?
Sistema de mesa de ayuda y gestión	¿Se brinda soporte, quien lo brinda? ¿Cuan brinda mantenimiento, correctivo, preventivo? Sistemas de contacto: Intranet Chat E-mail Horarios de atención

Especificación de requerimientos

Especificación de requerimientos

Especificación de requerimientos



2.4 Características de usuario

Tabla ejemplo:

Table with 2 columns: Tipo de usuario, Habilidades. Rows include Administrator del sistema, Administrador de información, and Analista.



3. FLUJOS DE USO

3.1. Flujos de negocio



4. ESCENARIOS OPERACIONALES

4.1. Tabla de escenarios operacionales



Table with 2 columns: Descripción, Condiciones de Ejecución. Rows describe user login, password change, and system status checks.

5. REQUERIMIENTOS ESPECÍFICOS

5.1 Métricas y avance

Se establecerán parámetros básicos de verificación de requerimientos, donde se pueda observar el estado actual de los mismos. Estos ejemplos se usan para entender cómo deben ir escritos dependiendo de su estado.

Requerido: El requerimiento fue escrito y está en proceso de validación con clientes.
Autorevisado: El requerimiento fue validado con cliente y está listo para su implementación.
Implementado: El requerimiento fue implementado en código y está listo para pruebas.
Probado: El requerimiento fue probado y está pendiente para verificación con el cliente.
Verificado: El requerimiento fue verificado por el cliente y termina su ciclo.

Se debe escribir cómo se evaluarán los criterios, cómo se medirá para establecer el cumplimiento, cada uno debe tener una prioridad para ser el estado del proyecto en el día a



Es, teniendo en cuenta el estado de cada requerimiento se establece el avance del proyecto en general.

5.2 Tabla de requerimientos funcionales

Table with 3 columns: No, TIPO, Requerimiento. Rows describe information administration, update, visualization, and graphic elements.

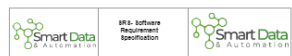
5.3 casos de uso

Cual es el caso de uso, se debe detallar a través de tabla y diagrama



Tabla con especificación ejemplo:

Table with 2 columns: Nombre de Caso de Uso, Autenticarse en el sistema. Rows include description, address, preconditions, basic scenario, and alternatives for user login.



6. REQUERIMIENTO NO FUNCIONALES

6.1 Usabilidad

Ejemplo: Las interfaces de la aplicación deben ser intuitivas, claras y concisas y la capacitación sobre la misma no debe ser superior a un día.

6.2 Disponibilidad

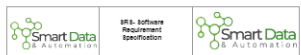
Ejemplo: La aplicación debe tener una disponibilidad al 100%, el tiempo de respuesta a incidentes no debe ser mayor a 30 min desde el reporte...

6.3 Desempeño

Ejemplo: Capacidad permitir el acceso a cuatro mil de usuarios (300). Restricciones de la máquina

El sistema debe estar en el sistema operativo:

- Linux Ubuntu 13.05.
- MAC OSx
- Windows Vista, 6, 8 o posterior.
Para máquinas o móviles se debe contar con características mínimas como:
- Procesador Intel dual
- 1.5 GB de RAM



- 2 GB en Disco Duro.
- Tarjeta de red.

Para dispositivos móviles Android De comer en servidores:

- Tomcat.
El sistema debe autenticar los usuarios con servidor LDAP.

6.4 Portabilidad

Ejemplo: Al ser desarrollado en lenguaje de programación JAVA, la aplicación debe ejecutarse en ambientes que cuenten con JRE (Java Runtime Environment).

6.6 Seguridad

Para los aspectos de seguridad que se llevará en el proyecto se tendrán en cuenta lo siguiente:
- Se debe trabajar un módulo administrador al cual sólo accedan personas autorizadas para realizar los cambios que se necesitan.
- Se trabajará un módulo de visualización el cual estará disponible para todos los usuarios.

6.7 Ambiente, sistema operativo

Se tendrán en cuenta las siguientes especificaciones:
- Dual base de datos ómnibus.
- Método de conexión a la base de datos.
- Lenguajes de programación que se usen.



6.8. Restricciones de diseño

Restricciones de diseño que tiene el sistema que se va a construir

- Tipo de lenguaje de programación
- Tipo de arquitectura como MVC

6.9. Requerimientos de documentación y ayuda en línea del sistema

- Tipo de ayuda que se va a prestar. E-mail, ayuda en línea, línea tja, etc.

6.10. Interfaces

(Si no se usan interfaces en algún tipo, colocar no se usa)

Cual tipo de interfaz será soportada por el sistema

- TCP/IP, GUI, CLI, etc.

Interfases de usuario

Cuales interfaces de usuario se van utilizar como por ejemplo:
- Equipos de pc.
- Dispositivos móviles.
- de lenguaje natural.

Interfases de hardware

Las interfaces que se van utilizar son:
Comprende todos los elementos que permitan ingresar, procesar, y entregar datos.

Interfases de software

Las interfaces que se van a utilizar son:
Ejemplo:
- Una API

Interfases de comunicación

Las interfaces que se van utilizar son:



Ejemplo:
- LDAP para usuario de administración



6.11. Estandarización



Los estándares que se van a utilizar en este sistemas son:

- El proceso de desarrollo del software se hace con base en metodologías SCRUM, SCRUMBAN, KANBAN, etc.

e) Formato para nuevo requerimiento

Cuando el proyecto se va ejecutando y nace un nuevo requerimiento se llena la siguiente tabla y luego se envía la solicitud para añadir el requerimiento al coordinador del proyecto.

	ESPECIFICACIÓN DE NUEVO REQUERIMIENTO	
---	---	---

	ESPECIFICACIÓN DE NUEVO REQUERIMIENTO	
---	---	---

No	TIPO	REQUERIMIENTO	TIPO DE REQUERIMIENTO	Estado actual
1	Administración de información	Permite ingresar datos relacionados con el proceso...	Nuevo requerimiento, Modificación de requerimiento	Aprobado, en desarrollo, producción



Especificación de un nuevo requerimiento

Autor
Nombre del proyecto al cual está asignado

f) Formato diseño del software

Este formato requiere de una descripción general de todo lo que se requiere para el diseño del software, donde se especifica cada área.

Se muestra a primera vista a través de mockups (si se requiere), esto sirve para que el cliente vea un modelado y pueda sugerir los cambios que más se adapten a sus requerimientos.

Se realiza el diagrama de interfaz gráfica para entender mejor las interacciones que se tengan en el sistema.



g) Formato especificación de pruebas funcionales

Consta de 5 columnas donde se especifica:

- La vista, es el módulo sobre el cual se va trabajar.
- Caso, es la opción que se evalúa.
- Escenario, es lo que se espera que el sistema realice.
- Prueba/Error, es donde se da resultado si pasa o no la prueba.

- Observaciones, es un espacio donde se permite que el probador deje sus comentarios ya sean de errores que se encuentra o de algunas mejoras que se pueden realizar y posteriormente pasar los resultados al equipo de desarrollo y corregir.

VISTA	CASO	ESCENARIO	PRUEBA/ERROR	OBSERVACIONES	Se deben realizar pruebas a cada elemento que lo componga, el caso que se evalúa, escenario, pasa o no. Se anexan ejemplos. Acá se prueban criterios o funciones en específico		
Nueva creación	Permite crear nueva campaña	El boton + debe permitir y mostrar el menú de creación	<input type="text"/>				
Nueva plantilla	Tipo de envío	Se debe seleccionar el tipo de envío para poder crear	<input type="text"/>				
Listas de envío	Ver que redireccione al menú	Se debe mostrar en pantalla las diferentes configuraciones que se tienen para listas de envíos	<input type="text"/>				

h) Formato pruebas de interfaz de usuario, aceptación y usabilidad

- Se especifica el módulo sobre el cual se evalúa.
- Sección y subsección.
- Obligatorio (Obligatorio, opcional).
- Funcionamiento (terminado, por arreglar, no aplica, sin verificar).
- Arreglos pendiente o mejoras, (se detallan textualmente).
- Problemas detectados en pruebas (se especifica textualmente).
- Estado en pruebas para release (Paso revisión, requiere corrección, sin verificar, es una mejora).
- Revisado por, (nombre de la persona que realiza la prueba).
- Prueba de aceptación para verificar si cumple los requerimientos (Pasó revisión, requiere corrección, sin verificar, es una mejora).
- Prueba de usabilidad, se verifica si es entendible la plataforma para el cliente. (Pasó revisión, requiere corrección, sin verificar, es una mejora).
- Observaciones (Se describe textualmente los errores o mejoras que se deben realizar)

Módulo	Módulo	Sección	SubSección	Obligatorio	Estado	Funcionamiento	Arreglos Pendientes o Mejoras	Problemas detectados en pruebas	Estado en pruebas para release	Revisado por	Pruebas de aceptación	Pruebas de usabilidad	Observaciones
				-	-	-			Sin Verificar		Sin Verificar	Sin Verificar	
				-	-	-			Pasó Revisión		Pasó Revisión	Pasó Revisión	
				-	-	-			Pasó Revisión		Pasó Revisión	Pasó Revisión	

i) Formato manual de usuario

Se describe toda la funcionales que se encuentran, realizando un documento que permita a los usuarios comprender cada módulo y cada interacción que se tenga en el sistema.



j) Manual de configuración

- Este manual describe toda la información que se debe tener en cuenta para configurar una nueva máquina.
- Se describen los pasos para hacer un release de producción
- Se describen los pasos para realizar la configuración del entorno de desarrollo.



k) Formato plan de pruebas

Se define un plan de pruebas para generarlo en todos los desarrollos que lo requieran, teniendo en cuenta el estándar IEEE 829 de documentación y pruebas. Este plan se conforma en tres partes:

a) Plan maestro de pruebas:

Se encuentra toda la especificación del proceso que se va realizar, como

Introducción,

Características a probar

Riesgos, aprobaciones

Criterios de aprobación/rechazo

Criterios de suspensión

Criterios de reanudación

Entregables de pruebas realizadas

Responsabilidades

Riesgos y contingencias

Especificación del diseño

Especificación de los casos de pruebas

Procedimientos de pruebas

Reporte de transmisión de pruebas.

b) Ejecución de las pruebas:

Logs de pruebas: registro de pruebas ejecutadas, orden y resultados.

Reporte de incidentes de pruebas, retroalimentación de los errores encontrados y posibles soluciones para transmitir al equipo de desarrollo.

c) Término de pruebas: se muestran los resultados obtenidos a través de imágenes, textos, código etc. **Ver Apéndice C.**



PLAN DE PRUEBAS DE SOFTWARE
Plataforma BOTAI



Tabla de contenido

Plan Maestro de pruebas (MTP)	2
Test Plan Identifier (Identificador de plan de pruebas)	2
Introduction (Introducción)	2
Test Items (Elementos de prueba)	2
Software Risk Issues (Problemas de riesgo de software)	3
Features to be Tested (Características a probar)	4
Features not to be Tested (Características que no deben probarse)	4
Approach	4
Item Pass/Fail Criteria (Elementos criterios de aprobación/rechazo)	7
Suspension Criteria and Resumption Requirements (Criterios de suspensión y requisitos de reanudación)	8
Test Deliverables (Entregables de prueba)	8
Remaining Test Tasks (Tareas de pruebas restantes)	8
Environmental Needs (Necesidades ambientales)	8
Staffing and Training Needs (Necesidades de personal y capacitación)	8
Responsibilities (Responsabilidades)	8
Schedule (Calendario)	9
Planning Risks and Contingencies (Planificación de riesgos y contingencias)	9
Approvals (Aprobaciones)	9
Glossary (Glosario)	9
Especificación del diseño de pruebas de nivel (LTD)	9
Especificación de los casos de pruebas de nivel (LTC)	9
Procedimiento de pruebas de nivel (I)	9
Reporte de transmisión de items de pruebas (ITL)	9
Ejecución de pruebas	9
Logs de Pruebas: Se registran con detalle qué pruebas se han ejecutado, orden y resultados de los mismos	9
Reporte de incidentes de pruebas	9
Términos de pruebas	9
Reporte de pruebas	9
References	9



Plan Maestro de pruebas (MTP)

Test Plan Identifier (Identificador de plan de pruebas)

"Dar un nombre donde se identifique claramente el plan y un número (01.02.03...), esto de un orden consecutivo a los diferentes planes que se realicen"

Introduction (Introducción)

Que es el plan, con que fin se hace etc...

Test Items (Elementos de prueba)

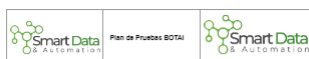
"Especificación del diseño de pruebas, establecer los items a evaluar"

Software Risk Issues (Problemas de riesgo de software)

- ¿Qué problemas pueden perjudicar?

Features to be Tested (Características a probar)

De todos los procesos que se llevan a cabo, ¿cuáles se van a probar? Especificar cada uno.



Features not to be Tested (Características que no deben probarse)

No se probarán secciones como, no se tendrá en cuenta puesto que estos están pasando por un periodo de integración a la plataforma, porque aún no son funcionales, etc.

Approach

- Que tipo de pruebas se aprobarán
- Describir cada una de ellas
- Describir las aplicaciones por las cuales se realizan las pruebas

Item Pass/Fail Criteria (Elementos criterios de aprobación/rechazo)

Estos criterios o condiciones son las tareas que debe cumplir un producto de software para ser aceptado por los usuarios, clientes o partes interesadas.

Ejemplo:

- ¿Me permite el sistema acceder en ser un superusuario?
- ¿Puedo editar perfiles exitosamente?

Suspension Criteria and Resumption Requirements (Criterios de suspensión y requisitos de reanudación)

Ejemplo:

- La prueba se suspende cuando exista un mal funcionamiento del sistema debido a problemas de software. (Se reanuda cuando se solucionen los problemas del software)

Test Deliverables (Entregables de prueba)

Dentro del plan de pruebas se deben definir los documentos que darán soporte al mismo, ejemplo:

- Documento de requerimientos



- Documento de configuración
- Manuales

Remaining Test Tasks (Tareas de prueba restantes)

Tareas que se omiten, pero que deben dejarse plasmadas para su posterior análisis y desarrollo ya que existen partes que el plan no aborde

Environmental Needs (Necesidades ambientales)

Staffing and Training Needs (Necesidades de personal y capacitación)

Quiénes tendrán acceso al plan, cuáles áreas en específico
Se utilizará para capacitar personal nuevo?

Responsibilities (Responsabilidades)

Quiénes son los responsables, áreas, personas, etc. y funciones que se desempeñan en cada una.

Schedule (Calendario)

Fecha en las cuales se programan las tareas

Planning Risks and Contingencies (Planificación de riesgos y contingencias)

- Definir los riesgos que se pueden presentar
- ¿Qué cosas fueron asumidas?
- Definir los elementos que permiten suplantar los riesgos

Approval (Aprobaciones)

Determinar quien debe aprobar y revisar el plan

Especificación del diseño de pruebas de nivel (ITD)

Especificación detallada del diseño de las pruebas, casos en específico, y criterios de aprobación



Especificación de los casos de pruebas de nivel (LIC)

Especificación detallada de los casos que se van a ser probados

Procedimiento de pruebas de nivel (I)

Cual es el procedimiento que se llevará a cabo para probar los items, criterios, casos, etc.

Reporte de transmisión de items de pruebas (ITI)

Análisis de las pruebas, aprobación, evaluación de calidad de las pruebas

Ejecución de pruebas

Logs de Pruebas

Se registran con detalle qué pruebas se han ejecutado, orden y resultados de los mismos (pass/fail).
Si hay inconformidades, se levanta o actualiza un reporte de incidentes

Reporte de incidentes de pruebas

Descripción de los detalles encontrados cuando la prueba no pasó

Término de pruebas

Reporte de pruebas

Reporte detallado sobre las pruebas realizadas, con imágenes, texto explicativo etc.

Apéndice B. Documentos lineamientos

a) Lineamientos de documentación



Documentación interna

Esta documentación conserva y utiliza registros para fundamentar decisiones que se toman dentro de una empresa u organización, donde se puede documentar infinidad de procesos o de información, desde planificaciones hasta las políticas importantes de la empresa.

Cada uno de los procesos que se documentan son muy importantes para la empresa como por ejemplo la documentación de un programa o servicio, donde se especifica el correcto paso a paso y de cómo funciona, esto es suma importancia al momento de capacitar nuevo personal y también dar soporte a los procesos que existen actualmente.

En este caso por ejemplo, se quiere documentar el lenguaje de programación o el funcionamiento de un sistema, donde los pasos a seguir son:

- Este cubre aspectos al programa realizados a la sintaxis de lenguaje, esta documentación está contenida en los comentarios entre llaves, paréntesis o asteriscos, algunos temas a considerar son:
 - cabecera de programa (breve descripción del programa)
 - nombres significativos para describir identificadores
 - comentarios relativos a la función del programa (Que hace el código)
 - cantidad de estilo y formato, líneas en blanco para separar módulos (Ser lo más claro posible en cuanto a la estructura del código)
 - comentarios significativos (Realizar comentarios importantes de las funciones)

Tenemos 4 pasos importantes para elaborar una documentación interna:

1. Identificar los procesos clave: Se deben establecer una reglas básicas para determinar que se documenta, esto nos permite ser muy asertivos, ya que tendremos unos componentes importantes sobre los cuales se van a trabajar, donde también se deben establecer criterios de lo que se debe documentar, esto con el fin de no documentar cosas que no tengan mayor relevancia en el sistema.
2. Crear una plantilla estándar: Se puede crear una plantilla que puedan usar todos los trabajadores de la empresa, donde se tengan en cuenta los siguientes campos, una descripción del motivo para el cual existe el proceso o del porqué es importante, quienes son los factores claves en el proceso, y que necesitará el personal en cuanto al hardware para poder llevar a cabo el proceso.



El nombre del proyecto será un nombre compuesto por tres palabras escritas en inglés. Con solo leerlo, se debe permitir entender qué tipo de aplicación se desarrolló, que función principal tiene y a qué proyecto o compañía pertenece.

tipoFuncionProyectooCualPertenece

Ejemplo:

frontCampaignaBotai

Se debe tener en cuenta el tipo de desarrollo que se va realizar, como por ejemplo si es un proyecto de frontend, backend, así, etc.

La primera palabra deberá ir en minúscula y corresponde a el tipo, a continuación se describen los tipos que se pueden manejar:

Tipo de aplicación	Acronimo
Aplicación tipo api rest	api
Aplicación de frontend	front
Aplicación de backend	back
Aplicación de servicios	services
Aplicación module	module

La segunda palabra se inicia con mayúscula y corresponde a la **Función** principal de la aplicación desarrollada, por ejemplo campaigns, api, etc.

Por último se debe colocar el proyecto o compañía para el cual fue desarrollada, ejemplo, botai, gajo agente, Botiok. Si el desarrollo es genérico y se puede utilizar en varias aplicaciones, esta tercera parte se debe nombrar **SmartData**.

- **frontCampaignaBotai**



- **backMainPasajente**
- **apiAPI_SmartData**

Nombre de funciones

- Se recomienda usar nombres que identifiquen el código; evitar abreviaciones como "fp" para "función" o "FH" para "fecha y hora", ya que esto no lo entenderá otro desarrollador.
- Es de suma importancia utilizar nombres que se puedan pronunciar fácilmente como por ejemplo **sumFunction** para Función Suma. Teniendo en cuenta estos nombres o funciones o variables, cualquier persona o trabajador de la empresa comprenderá fácilmente de que se trata o que puede contener el código.

SDAA

Deben ser nombres claros y concisos, que no tengan doble sentido, se debe dar nombres que identifiquen el código y estar escritos en inglés.

Para aplicaciones desarrolladas en **React, Laravel, Angular, PHP**, siga el siguiente estándar:

Si es una sola palabra debe ir en minúscula, si son palabras compuestas, se deben escribir la primera palabra toda en minúscula y a partir de la segunda iniciar con mayúscula.

- **sumFunction** - Rápidamente se entiende que es una función suma.
- **webhook** - Van todas en minúsculas porque es una sola palabra.
- **sendMsgFacebook** - La primera palabra en minúscula y las siguientes se separan por medio de las letras mayúsculas.

Para aplicaciones desarrolladas en **Python**, se deben seguir los mismos parámetros, sin embargo para separar las palabras se debe hacer por medio de "_", y todas las letras deben ir en minúscula.

Ejemplo:

Describe el objetivo de la función.

- **webhook**
- **send_Msg_Facebook**



3. Decidir dónde almacenar los procesos: La documentación interna debe ser accesible para todos los integrantes de la empresa, donde se aconseja crear una carpeta donde se encuentren todos los archivos o un espacio donde se puedan almacenar los registros. También se debe tener en cuenta la información que se tiene, que sea clara y precisa para disminuir tiempos en los procesos que se van a realizar.

4. Reservar tiempo para hacer limpieza: Se debe realizar limpieza en cuanto a los documentos nuevos que entran, esto con el fin de eliminar documentos basura que no brindan conocimiento o no tienen importancia en esa sección o documentos que deben actualizarse, se debe establecer unos tiempos para realizar estas limpiezas como por ejemplo cada mes, cada trimestre o el tiempo que se elija dentro de la empresa, de este modo se contribuirá a que el sistema conserve archivos válidos y este en buen estado, esto con el fin de que todos hagan uso de los registros.

Cada uno de los procesos que se documentan son muy importantes para la empresa como por ejemplo la documentación de un programa o servicio, donde se especifica el correcto paso a paso y de cómo funciona, esto es suma importancia al momento de capacitar nuevo personal y también dar soporte a los procesos que existen actualmente.

Lineamientos comunes

Son los lineamientos de documentación que se establecen para todos los desarrollos independientemente del tipo de aplicación a implementar.

A continuación se darán los ejemplos de cómo se debe estructurar el código.

Nombre del proyecto

Se recomiendan nombres fáciles de identificar al momento de realizar un búsqueda rápida en el menú del código, para así mismo traer la información que contiene el proyecto como por ejemplo **ProjectCampaigna, ProjectTemplate**, donde la segunda letra mayúscula indica que hay un espacio entre esas dos palabras o también se puede colocar el rayo al paso " _".

La que se busca es que cualquier miembro del equipo comprenda solo leyendo el título del proyecto que puede contener este mismo y minimizar los tiempos en búsqueda del código que se requiere.

SDAA



Comentarios del código

Comentarios muy acertados y descriptivos con la información que se maneja en el código.

- Se debe comentar la arquitectura general, vista de alto nivel del código.
- Utilización de funciones.
- Situaciones importantes, especialmente cuando no son inmediatamente obvias.
- Se debe denotar muy clara la información del cómo se usa y para que. Todo el código no se puede estar comentando; por ese motivo, se debe consultar un código limpio y entendible para que cualquier persona pueda entenderlo fácilmente sin necesidad de leer un comentario.
- El comentario debe ser claro en qué parámetros pide, y qué devuelve, esto se comenta si es gran dificultad entender el código, también se debe comentar la funcionalidad para la cual se creó, o también para brindar información sobre versiones, donde está ubicada alguna extensión y demás.

```

Ejemplo:
/**
 * Devuelve n elevado a la potencia de m.
 *
 * @param {number} n El número a elevar.
 * @param {number} m La potencia, debe ser un número
    natural.
 * @return {number} n elevado a la potencia de m.
 */
function pow(n, m) {
  ...
}
    
```

Este comentario nos permite saber el propósito de la función y poder usarla de manera correcta.

También comentarios informativos como:

```

/**
 * Esta clase crea una instancia de las clases de notificaciones.
    
```



```

 * Registrado en la clase core.security.Security object.
 *
 * Autor: Juan Perez
 *
 * Versión: 1.00, 24/14/2015
 *
 * Versión: 1.4
    
```

• Se debe tener en cuenta el tipo de lenguaje en el cual se está desarrollando, ya que cada uno de ellos tiene una forma de iniciar los comentarios y de finalizarlos.

• Por ejemplo en java y php, es con **/*** para comentar una sola línea, ejemplo:

Esto es un comentario, pero si queremos comentar varias líneas se hará de la siguiente manera **/***

Esto se comentan varias líneas en java sin necesidad de estar comentando una por una **/***

• En Django y php también se usa así **/*** Esto es un comentario en Django. Se comenta como un **/***

• En React: **/*Esto es un comentario*/** Se comenta con **/***

• En Laravel: **/* Esto es un comentario */** Se comenta con dos líneas al principio y dos líneas al final **/***

• En angular: **/* Esto es un comentario */** Se comenta con signo de menor o igual, signo de exclamación raya al medio, y se cierra con raya al medio y signo de mayor o igual: **/***

```

//COMENTARIO RELATIVO A LA FUNCIÓN DEL PROBLEMA
let res={
  id:1,
  name:'John'
}
    
```

• Los comentarios deben ser útiles, sino aportan algo importante al código, evitar realizarlos, como en este caso donde se ve claramente cada funcionalidad del código. Al usar nombres descriptivos el código no necesita comentarios adicionales.



```

has_cliente = servicioClientes.recuperarDatosCliente(id_cliente);
if (cliente.estaActivo)
{
    clienteUtilizarCreditoNuevo;
    servicioClientes.gestionarCliente(cliente);
}
    
```

Nombre de variables

Para los nombres de las variables se aplicaron las siguientes reglas:

- Cada nombre de variable debe ser exclusivo; no se permiten duplicados.
- Las variables no pueden contener espacios.
- No debe utilizar caracteres especiales como por ejemplo %,&,@,...
- Se debe evitar los nombres de variable que terminen con un carácter de subrayado, ya que tales nombres pueden entrar en conflicto con los nombres de variables creados automáticamente por compiladores y procesadores.
- Las palabras reservadas no se pueden utilizar como nombres de variable. Las palabras reservadas son ALL, AND, BY, EQ, GE, GT, LE, LT, NE, NOT, OR, TO y WITH.
- Los nombres de variable se definen en inglés, se deben usar palabras que definan lo que se está almacenando en la variable y si son compuestas se separan a través de rayas o guión_

Teniendo en cuentas estas características se nombró así en python, **tercer**

```

- datos_cliente
- id_cliente
- cliente_tercer
    
```

incorrecto y no se debe realizar ya que entrarán en conflicto con otros aspectos dentro del sistema al momento de realizar búsquedas y hacer llamadas a las variables

```

- _datos_cliente
- numeroTerminos_cliente
    
```



- int (mai)
- Base

Estilo y formato

- Se recomienda usar espacios entre diferentes palabras o acciones dentro de una función, con el fin de que se pueda observar mejor y mejorar la legibilidad del código, puesto que a simple vista se puede observar las diferentes partes de la clase o función. También se deben tener en cuenta que las funciones a invocar deben estar verticalmente alineadas, y es posible que invite por encima de la otra, esto para evitar desordenar las funciones o variables al principio, al medio o al final del código, observando así una construcción limpia y entendible del código y muy fáciles para muchos desarrolladores que hagan uso de este código. Se debe tener en cuenta el formato horizontal del código teniendo en cuenta que actualmente las pantallas son más grandes y tiene mejor resolución se recomienda que 100 a 120 puntos estar bien, pero no más, esto con el fin de que todos lo que hagan uso de código, puedan ver claramente en sus pantallas, sin sobrepasarse de los límites y hacer más ameno del mundo del mismo. Tenemos un ejemplo claro del estilo que se debe manejar, donde es más explícito y organizado, mostrando un orden vertical de un módulo o parte de código bien estructurados, ejemplos:

SOA:

- Un ejemplo de lo que no se debe hacer se denomina código de trenes, y es algo que puede denotar después en un código y que se debe evitar: **Real String word = obj.method();return word;** Este tipo de código puede ser difícil de comprender para otros desarrolladores y se debe corregir como está en la siguiente parte del código.



- Una mejor implementación sería: **Real String word = obj.method(); final Object obj = obj.method(); final Object obj = obj.method();**

Donde se especifica cada función y se puede entender mucho más fácil para otros desarrolladores que manejan otras prácticas de documentación diferentes a las de la persona que documenta primera el código.

Se debe usar la indentación con la tecla **TAB**, con el fin de evitar muchos errores al momento de ejecutar el código, esto nos ayuda a ver claramente a qué función o variables pertenece una línea de código.

```

function bucle() {
    for (i = 0; i < 10; i++) {
        console.log("Iteración #", i);
    }
    if (i == 9) {
        console.log("Estoy en la última iteración");
    } // for
} // function
    
```

El código siempre debe estar indentado de esa forma, utilizando la tabulación para tener un código claro y limpio.

Cuando existe la falta de indentación se puede ver así:

```

function bucle() { for (i = 0; i < 10; i++) {
    console.log("Iteración #", i);
    if (i == 9) console.log("Estoy en la última iteración"); }
}
    
```

Se puede ver que usa menos espacio pero es menos legible que el ejemplo anterior.



Lineamientos específicos

Documentar el código de un programa es poder añadir la suficiente información en un documento para poder explicar que se hace, cómo funciona y tener en cuenta algunos errores que se pueden presentar en su desarrollo; esto con el fin de que otras personas sepan que hacer al momento de tener algunos inconvenientes en la revisión del código, y que se entienda muy clara cual es la finalidad de este código.

Todos los programadores y desarrolladores tienen una forma muy particular de escribir el código teniendo en cuenta la situación en la que se está trabajando, donde se busca que la máquina comprenda ciertos argumentos que el desarrollador debe interpretar a su manera, pero que esto también puede traer problemas ya que sino se tiene una documentación del código, pues será muy difícil comprender cuáles son las funcionalidades del aplicativo o de si ocurre algún error no se puede solucionar porque no sabemos interpretar el código con el que fue construido.

Entonces, documentar un programa no solo es un acto de buen hacer del programador, sino de ser muy profesional en su área y dejar claro todos los parámetros que se establecen para que otras personas puedan entender cada punto del código y hacer las tareas más fáciles y de poder actuar rápidamente frente a errores que pueden suceder en el transcurso del día a día.

tenemos dos reglas que no debemos olvidar:

1. Todos los programas tienen errores y descubrirlos solo es cuestión de tiempo y de que el programa tenga éxito y se utilice frecuentemente
2. Todos los programas sufren modificaciones a lo largo de su vida, al menos todos aquellos que tienen éxito

Algo que nos debe quedar claro es que debemos hacer la documentación del código con cada función que se haga, para que en un futuro los demás programadores que hagan uso del mismo sepan cuál es su funcionamiento, donde también se pueda mejorar y corregir errores en tiempos menores a los esperados.

Lo más importante a documentar es añadir explicaciones a todo lo que no es evidente, no hay que olvidar lo que se hace, sino explicar por qué se hace; también debemos documentar los métodos, variables, algoritmos que se usen, limitaciones, que se desearán documentar, entre otros.



Para la documentación de APIs dependiendo el lenguaje de programación está definido en el documento lineamientos de documentación y estandarización sección de documentación de APIs.

Swagger:

Es una serie de reglas, especificaciones y herramientas que nos ayudan a documentar nuestras APIs. Por lo tanto, en esta herramienta podemos ver todos los endpoints que hemos desarrollado en nuestra API Swagger. Además, nos demuestra cómo son los elementos o datos que debemos pasar para hacer que funcione y nos permite probarlos directamente en su interfaz.

A continuación se darán unos pasos a seguir para tener una correcta documentación teniendo en cuenta el lenguaje de programación que se está utilizando.

Documentación API PYTHON DJANGO

<https://www.django-rest-framework.org/api-guide/serializers/#swagger>

La documentación se realiza con el fin de que el cliente sepa y tenga documentación sobre lo que se ha creado.

En este caso se mostrará el paso a paso para generar la documentación en este lenguaje.

Se realizará la documentación con Swagger:

1. Lo primero que se debe hacer es descargar la dependencia **drf-yasg** con el comando **pip install drf-yasg** en el terminal que se está usando.
2. Luego se añade la dependencia descargada en nuestras aplicaciones instaladas, se procede a editar el código en el apartado de **INSTALLED_APPS** para poder hacer funcionar esta dependencia, donde nos situamos en la aplicación principal (ejemplo: Django demo), settings.py, installed_apps, y se añade **drf_yasg**, que este será el apartado de las aplicaciones instaladas y de las que se van creando.
3. Luego de realizar este proceso vamos al archivo **urls.py** y se añade el siguiente código.

Así podemos observar los dependencias que tenemos y las que acabamos de instalar.

```

from rest_framework import permissions
from drf_yasg.views import get_swagger_view
from drf_yasg import swagger
    
```



4. Después de verificar que tenemos las dependencias instaladas vamos a editar el siguiente código antes del apartado de **urlpatterns**

```

swagger_view = get_swagger_view({
    'urlpatterns': urlpatterns,
    'title': 'Swagger API',
    'description': 'Swagger API',
    'terms_of_service': 'https://www.django-rest-framework.org/api-guide/serializers/#swagger',
    'contact_name': 'Smart Data Automation',
    'contact_email': 'info@smartdataautomation.com',
    'contact_url': 'https://www.django-rest-framework.org/api-guide/serializers/#swagger',
})
urlpatterns += swagger.urls(
    'swagger',
    default_swagger_urls,
)
    
```

En este apartado podemos modificar el título, la descripción, correo y demás.

En Django quedará de la siguiente manera

```

from django.conf.urls import url
from django.urls import path, include
from rest_framework import permissions
from drf_yasg.views import get_swagger_view
from drf_yasg import swagger

urlpatterns = get_swagger_view({
    'urlpatterns': urlpatterns,
    'title': 'Swagger API',
    'description': 'Swagger API',
    'terms_of_service': 'https://www.django-rest-framework.org/api-guide/serializers/#swagger',
    'contact_name': 'Smart Data Automation',
    'contact_email': 'info@smartdataautomation.com',
    'contact_url': 'https://www.django-rest-framework.org/api-guide/serializers/#swagger',
})

urlpatterns += swagger.urls(
    'swagger',
    default_swagger_urls,
)
    
```

5. Ahora se tiene que añadir la **url** para la documentación en la parte de **urlpatterns**

```

urlpatterns += [
    path('swagger', swagger.urls('swagger', default_swagger_urls)),
    path('api/', include(router.urls)),
]
    
```



```

swagger_view = get_swagger_view({
    'urlpatterns': urlpatterns,
    'title': 'Swagger API',
    'description': 'Swagger API',
    'terms_of_service': 'https://www.django-rest-framework.org/api-guide/serializers/#swagger',
    'contact_name': 'Smart Data Automation',
    'contact_email': 'info@smartdataautomation.com',
    'contact_url': 'https://www.django-rest-framework.org/api-guide/serializers/#swagger',
})
urlpatterns += swagger.urls(
    'swagger',
    default_swagger_urls,
)
    
```

6. Por consiguiente y después de realizar todos los pasos mencionados anteriormente ya tenemos nuestra documentación creada y funcionando.

7. Para probar su correcto funcionamiento vamos al terminal y copiamos **python manage.py runserver**, donde esto nos sirve para reiniciar nuestro servidor

```

python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...
System check identified no issues (0 silenced).
May 28, 2022 - 17:25:17
Django version 4.0.4, using settings 'django_demo.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C
    
```

se copia la url que nos brinda al terminal, y vamos a nuestra pantalla en el navegador



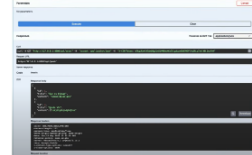
Donde se puede observar que tenemos dos url nuevas que son **3.docx** y **4.docx**, donde si se quiere ver sólo una de ellas, se clican entre barras rectas en la dirección de la página.



Se puede observar como se nos ha generado toda la documentación de manera automática, donde tenemos el GET, POST, PUT, PATCH, DELETE.



Se aconseja utilizar esta documentación ya que nos facilita escribir el código en la misma página, donde podemos ver para qué nos sirve, ejemplo:



Documentación API PYTHON

<https://www.django-rest-framework.org/api-guide/serializers/#swagger>

Se deben seguir los pasos 1, 2, 3, 4, 5, 6, 7 de la documentación de API PYTHON DJANGO

Documentación API PHP

<https://www.django-rest-framework.org/api-guide/serializers/#swagger>

La documentación se realiza a través de **swagger**. La documentación en swagger se basa en un archivo .YAML que se usa como un archivo de configuración:

1. Vamos al editor en línea de swagger 'editor.swagger.io'
2. Se procede a editar y probar los valores correspondientes para identificar la API que se quiere documentar y que ya está creada.



```
Swagger Editor
...
// Ejemplo de cómo se muestra la información de la base de datos
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
```

Luego de especificar la API y dar unas breves indicaciones se harán los paths y las url que se usarán para llamar nuestra API.

```
...
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
```

Esto se hace dentro de la propiedad schema escribiendo la propiedad type

```
...
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
```

3. Si este parámetro se va usar para diferentes url hay dos opciones, una es escribir cada url al mismo código del parámetro y la segunda es definir todos los parámetros que vayamos usando al final del archivo y solo hacer referencia cuando se vaya a usar. Se aconseja usar la segunda porque aplica tanto para los parámetros de las url como las respuestas y los esquemas que son un conjunto de objetos que se pueden referenciar en cualquier momento.



Entonces, el código del parámetro bien que ya existe lo vamos a cortar y pegar dentro del parámetro de **response**, y luego se procede a referenciar donde estaba antes con la propiedad **ref** en un lenguaje como una cadena el nombre del parámetro al que se quiere hacer referencia

```
...
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
```

4. De realizar las respuestas en el apartado de **responses** para ver si al momento de llamar o buscar una API esta funciona correctamente o hay errores.

```
...
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
```

Y luego se va a mandar a referenciar donde estábamos anteriormente



```
...
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
```

5. Para verificar la correcta creación vamos en el apartado derecho de swagger la descripción de nuestra API con todos los parámetros que se establecieron, donde está el encabezado donde se ve el nombre de la API, la versión, el tipo de protocolo que usa, una breve descripción y las url base, donde si se da clic en una url podemos visualizar el apartado de parámetros con los parámetros que se indicaron.



Documentación APIS LARAVEL
<https://laravel.com/docs/5.4/api>
 Se deben seguir los pasos 1, 2, 3, 4, de la documentación de APIS CON PHP.

Documentación de ReactJS con react-boostrap

- Se debe instalar el paquete por medio de la terminal con la finalidad de instalar el paquete de manera global.
 npm install -g react-boostrap

Ejemplo:



```
...
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
```

2. Se buscan generar comentarios en cada uno de los componentes de una manera muy breve, simplemente una descripción en la parte de arriba de las propiedades para que automáticamente se genere la documentación.



Se puede observar como nos genera la documentación a través del terminal, si se quiere reestructurar para que lo muestre en un orden distinto y más agradable se debe colocar `pretty` en el terminal y tenemos como resultado

```
...
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
// Respuesta de la API se envía de manera que la información de la base de datos
// sea accesible a través de Swagger
```



Documentación externa

Un programa no solo puede ser leído y ejecutado solo por un computador, se busca hacer entender a los programadores o futuros empleados de la empresa el funcionamiento que logra el trabajo desarrollado.

Esta tarea que varía con documentos digitales que describen los componentes, el código en sí, este sirve tanto para programadores que vayan hacer uso de este a futuro a trabajadores de la empresa, está los datos, componentes base como datos del programa, que hace, datos de entrada y demás, esto ayuda a entender el código desarrollado, donde también ayuda a los investigadores, así como tener acceso a los datos que se han realizado como las clases de los programas, variables y demás.

La documentación externa es fundamentalmente necesaria para entender qué se ha desarrollado, cómo, algoritmo, función, componente, tenerle sentido a una documentación específica y entendiendo esta documentación se va a facilitar el trabajo a otros programadores el código desarrollado.

- Unos pasos para la documentación externa serían:
- Listado actual del programa fuente y especificación del programa.
 - Diagrama de estructura que represente la organización jerárquica de los módulos que componen el programa.
 - Explicaciones de fórmulas complejas.
 - Descripción de los datos a procesar.
 - Plantillas de plantillas utilizadas para estructurar con los usuarios.
 - Cualquier información especial que pueda servir a los programadores que deban mantener el programa.

- **Directorio de implementaciones:**
 Este documento contiene toda la información importante que se requiere para documentar; a continuación se explica cuál es la función de cada columna y como se debe leer:
 - ➔ **IT:** Es la instancia donde se encuentra desplegado el desarrollador, esto cuenta con las columnas, la IP y Nombre.



- **EC2:** Es el nombre de la instancia donde se encuentra el desarrollador.
- **Directorio:** Link o Url del directorio donde se encuentra.
- **Proyecto:** Nombre del proyecto a cual pertenece.
- **Tipos:** Si es de tipo API, Landing, Desarrollo o la medida, Servicio.
- **Quemado:** Link de donde pertenece.
- **Repositorio:** Link del repositorio donde se encuentra guardado.
- **Estado:** Se especifica si está en Producción, Desarrollo, Demo 0, No se usa.
- **Producto:** Para qué tipo de plataforma se usa.
- **Observaciones:** Se dan breves observaciones y se anexan imágenes de ser necesario.
- **Curl prueba:**
- **URL:** link donde se usa: Link o url del bot que lo usa actualmente.
- **Coordinador:** Servicio/cargo: Nombre de quien solicita o está a cargo de la información.
- **Desarrollador(es):** Nombre de quien desarrolló la API.
- **Link de documentación:** URL o LINK sobre la documentación que la sustenta.
- **Lenguaje de programación:** Tipo de lenguaje en el cual está desarrollada. (PHP, JAVA, C, C++, etc.)
- **Conexión a BD:** A qué tipo de base de datos corresponde. (MySQL, NoSQL, SQLite, MongoDB, etc.)

- **Especificación general:** Para cada desarrollo que se realice, se debe ingresar a la plantilla de especificación general y hacer todos los campos requeridos, esto con el fin de tener en un primer plano la hoja de vida de un proyecto y en la parte inferior, se encuentran los links de acceso a los demás plantillas que se deben documentar para todos los desarrollos. **Especificación general**
- **IRI - Documento especificación de requerimientos:** En este documento se deben detallar todos los requerimientos de los clientes de una forma muy detallada como está planteado en la plantilla, esto con el fin de minimizar errores en las



entrega final y que el cliente está 100% satisfecho con lo que se le brinda.

- Plantilla SRS- Especificación de requerimientos:**
- Documento de arquitectura:** Se debe detallar muy bien toda la información sobre el software, como motivadores, restricciones, puntos lógico, funcional, modelos de red etc. **Plantilla arquitectura del Software**
- Documento de diseño:** Todos los proyectos o en su gran mayoría, cuentan con unas vistas previas donde se pueden mostrar al cliente con el fin de que se puedan realizar cambios antes de entrar al desarrollo, minimizando así los tiempos en la creación de los productos y satisfacer las necesidades del cliente, para ellos se debe realizar una descripción de todo lo requerido para el software, diseñar los mockups del sistema y la interacción gráfica. **Plantilla diseño del Software**
- Manual de configuración:** Se establecen las normas básicas de configuración de los equipos para el posterior funcionamiento. Se debe dirigir a la plantilla Manual de configuración y detallar toda la información. **Plantilla Manual de Configuración**
- Manual de usuario:** Se muestra al usuario como son los pasos que debe seguir para que no tenga errores o malas conductas al momento de iniciar un proceso o proyecto. **Plantilla Manual de usuario**
- Plan de pruebas:** Ramifícate al documento Plan de pruebas Botas, donde se encuentra una plantilla donde especifica los campos que deben ser ocupados. **Plantilla Plan de pruebas BOTAS módulo campañas whatsapp**
- Plantilla de pruebas:** Se debe plasmar toda la información de las pruebas que se vayan a realizar con datos específicos como la vista, caso, escenario, prueba/entorno y observaciones. Esta plantilla cuenta con 3 casos de pruebas en específico, como pruebas unitarias, pruebas de integración y pruebas de extremo a extremo.
 - Plantilla Especificaciones de pruebas funcionales.**
 - Vista:** Módulo en el cual se trabaja.
 - Caso:** Opciones dentro del módulo como creaciones de campañas, lista de contactos y demás elementos que los pueden componer.
 - Escenario:** Se detalla que se va a evaluar como por ejemplo: El botón debe permitir y mostrar el menú de creación
- Plantilla pruebas de interfaz de usuario:** Es un documento a parte de las pruebas anteriormente mencionadas ya que esta es un tipo de pruebas muy importante y se



debe realizar a todos los proyectos y desarrollos que se realizan, ya que de esta depende garantizar una excelente experiencia al usuario.
Plantilla Pruebas de Interfaz de usuario UX



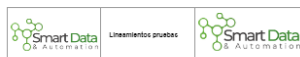
Referencias

- Documentación de python:
<https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>
<https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>
<https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>
<https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>
- <https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>
- <https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>
- <https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>
- <https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>
- <https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>
- <https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>
- <https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>
- <https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>
- <https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>
- <https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>
- <https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>
- <https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>
- <https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>
- <https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>
- <https://www.python.com/7aa31gurnh8g33A92f%3Fwww.python.com%3F>

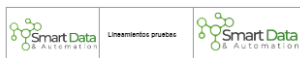
b) Lineamientos de pruebas

Se realiza un documento donde se especifican el tipo de pruebas que pueden ser aplicadas a los diferentes desarrollos que se realizan en AmartData & Automation, esto con el fin de tener una guía clara de qué tipos de pruebas se realizan, como se realizan y por medio de qué herramientas se implementan.

TIPO DE REQUERIMIENTO	SUBTIPO	PRUEBAS						
		PRUEBAS UNITARIAS	PRUEBAS DE INTEGRACIÓN	PRUEBAS DE EXTREMO A EXTREMO	PRUEBAS DE INTERFAZ DE USUARIO	PRUEBAS DE RENDIMIENTO	PRUEBAS DE USABILIDAD	PRUEBAS DE ACEPTACIÓN
		DESARROLLADOR	EQUIPO DE QA	EQUIPO DE QA	EQUIPO DE QA	DESARROLLADOR	EXPERIENCIA	EXPERIENCIA
IMPLEMENTACION BOT							X	X
API ESPECIFICA	FRONTEND	X	X	X	X		X	X
	BACKEND	X	X	X				X
API GENERIC	FRONTEND	X	X	X	X		X	X
	BACKEND	X	X	X				X
NUEVA FUNCIONALIDAD	FRONTEND	X	X	X	X		X	X
	BACKEND	X	X	X				X
	CAMBIO DE ARQUITECTURA					X		
CORRECCION DE ERROR	FRONTEND	X	X	X				X
	BACKEND	X	X	X	X		X	X
NUEVO MODULO	FRONTEND	X	X	X				X
	FRONTEND	X	X	X	X		X	X
	CAMBIO DE ARQUITECTURA					X		
NUEVO PASO AGENTE				X	X			



Lineamientos Pruebas



Pruebas unitarias de React

Se utiliza la herramienta JEST, que es una biblioteca de ejecución de pruebas de JavaScript con acceso al DOM a través de JSDOM. JSDOM hace que el navegador realmente un navegador web, pero generalmente es suficiente para probar los componentes de React. Jest ofrece una gran velocidad de iteración combinada con funciones potentes como módulos simulados y temporizadores, lo que le brinda más control sobre cómo se ejecuta su código.

La biblioteca de pruebas para React es una biblioteca de utilidades que lo ayudan a probar los componentes de React en dependencias de los estilos de implementación. Si bien este enfoque simplifica la refactorización y también conduce a las mejores prácticas de accesibilidad, no proporciona una forma "usual" de representar un componente sin elementos secundarios, lo que permite las capacidades de simulación de Jest.



Criterios de evaluación

1. Permitir crear nueva campaña, mostrando en pantalla el menú de crear nueva campaña
2. Nombre de campaña, línea telefónica de la cual se va enviar la campaña, tipo de envío (masivo o único)
3. Permitir seleccionar lista de contactos al seleccionar
4. Permitir realizar envío único y escribir los números en pantalla
5. Crear nueva plantilla y ver que todas sus funcionalidades permitan la correcta creación de la misma. Se debe escoger canal de envío, línea celular, idioma, categoría, lenguaje, tipo de mensaje (imagen, archivo, video, texto), texto embebido, día de salida, mensaje
6. Presionar botón lista de envíos y que muestre el menú
7. Permitir escoger listas de contactos



Pruebas de aceptación:

Se brindan unos pasos en específico para que un usuario pueda realizarlos, esto con el fin de evaluar componentes específicos del sistema y ver si está correctamente configurado, para detectar errores etc.
Se enfocan en asegurarse de que el sistema sea "robustado para su propósito". Se diseñan principalmente a partir de especificaciones de requisitos, casos de uso y procesos comerciales definidos.
Esta prueba se realiza manualmente.

Pruebas unitarias de Django-Python

Se utiliza la herramienta Coverage.py que nos permite medir la cobertura del código en programas de Python. Ejecutamos los programas para determinar qué partes del código se han ejecutado y analiza el código fuente para identificar el código que podría haberse ejecutado pero no se ejecutó.

Las mediciones de cobertura se usan comúnmente para medir la efectividad de la prueba. Puede indicar qué partes de su código son abarcadas por la prueba y cuáles no. La cobertura tiene la capacidad de generar informes en formato de línea de comandos o HTML. Estos informes se generan en la carpeta `htmlcov` del propio proyecto.



Tabla de contenido

Pruebas unitarias	3
Pruebas unitarias de React	4
Criterios de evaluación	4
Pruebas de integración React	5
Pruebas de extremo a extremo	5
Pruebas de rendimiento React	6
Pruebas interfaz de usuario:	6
Pruebas de usabilidad:	6
Pruebas de aceptación:	7
Pruebas unitarias de Django-Python	7
Referencias	7



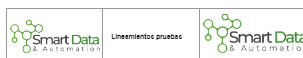
Pruebas unitarias

La prueba o prueba de código se refiere al proceso de verificar el comportamiento del software de un programa o aplicación en particular. Una característica de este mecanismo es asegurar la calidad del sistema, para lo cual se utilizan recursos como pruebas unitarias de software. Estos recursos sirven como validaciones de que el fragmento de código fuente funciona correctamente. Así que implementando este tipo de pruebas a nuestros sistemas, podrían resultar de gran ayuda para asegurar el buen funcionamiento del código.

Es importante aclarar que este tipo de prueba es importante para detectar errores. Porque sin estas pruebas, solo se descubrirán en etapas más avanzadas de desarrollo, como la fase de integración. Esto significa que las pruebas unitarias de software detectan errores en su código antes de tiempo, lo que evita la escalada de errores.

Una prueba de unidad de software contiene un conjunto de características y propiedades tales como: Esto se hace, por ejemplo, escribiendo un fragmento del código fuente de la aplicación o programa para que esta unidad de código pueda ser probada. Uno de los propósitos principales de este tipo de pruebas es, por lo tanto, garantizar que cada unidad de software analizada funcione de manera adecuada e independiente.

Una de las ventajas que se tienen con este tipo de pruebas es que se puede demostrar la capacidad lógica del código fuente de una aplicación para verificar que se encuentre en buen estado y trabajando de manera normal. Otra ventaja sería que permite el aumento de la legibilidad del código fuente, por lo que al detectar errores, el equipo tendrá la capacidad de realizar los cambios pertinentes para suprir las fallas de una forma más sencilla y evitando la pérdida de tiempo y recursos al buscar el error.



8. Permitir visualizar las listas de contactos
9. Permitir crear una nueva lista de contactos
10. Evaluar los datos como título de lista de contactos, seleccionar canal de envío, integración de número, columna con número telefónico
11. Finalizar la creación de lista y ver en pantalla el mensaje de éxito

Documento tipo excel con especificaciones de prueba:
- [Especificaciones de pruebas funcionales](#)

Pruebas de integración React

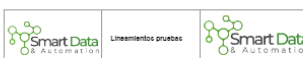
Este tipo de prueba nos permite verificar los distintos módulos o servicios utilizados en la aplicación y ver si trabajan bien en ese conjunto. Se puede probar la interacción con la base de datos o asegurarnos de que los pequeños servicios funcionan bien en conjunto y según lo esperado pero con un poco más de costos siendo posteriormente se deben tener varios elementos de la aplicación en marcha para obtener los resultados esperados. Esto ayuda que al usar la aplicación y analizar los resultados sin examinar su funcionamiento interno, se asegura de que la comunicación entre los servicios suceda correctamente.

Se puede usar el mismo modelo de las pruebas unitarias ya que acá se van a comunicar los módulos de diferentes áreas y de propio elección para observar su comportamiento, determinar fallas entre módulos y por último suplírselas.
Esta prueba se realiza manualmente, verificando así el correcto funcionamiento entre el frontend y el backend con las opciones que el mismo sistema nos brinda.
Se anexa resultado.
- [Especificaciones de pruebas funcionales](#)

Pruebas de extremo a extremo

Esta es el proceso de probar el software de principio a fin a medida que lo usan los usuarios reales. Para una aplicación web, se debe iniciar un navegador, ir a la URL correcta, usar la aplicación según lo previsto y verificar la operación. Para una aplicación de escritorio, debe ejecutar la aplicación, usarla y también verificar su funcionamiento.

Lo que se busca probar es que la aplicación o software trabaje conjuntamente y acceda a todas las funcionalidades que se tienen, interactuando así entre diferentes módulos y también dependiendo del tipo de usuario se tenga en cuenta las opciones a las que puede acceder y las que no, viendo en pantalla si aparece error, una ventana flotante que indique



que no está disponible el servicio para ese tipo de usuario y ver que todo funciona dependiendo del tipo de usuario con el que se está accediendo como por ejemplo: usuario administrador, usuario reportes, usuario superusuario, usuario OI

Resultados: [Especificaciones de pruebas funcionales](#)

Pruebas de rendimiento React

En estas se evalúa el rendimiento del sistema con una carga de trabajo predeterminada por una persona, donde se ayuda a medir la latencia, la velocidad, la escalabilidad y la capacidad de respuesta de una aplicación.

Acá podemos medir el tiempo en que el sistema ejecuta un gran número de tareas, o es cómo se comporta con una gran cantidad de datos al mismo tiempo. Se puede determinar si se cumple con los requisitos de rendimiento, localizar las áreas afectadas o áreas que surgen, medir la estabilidad durante los picos de tráfico y demás.

La herramienta a utilizar es APACHE JMeter, que nos brinda una interfaz gráfica muy entendible, donde se puede especificar qué módulo se quiere probar y qué cual es el rendimiento del mismo, donde se pueden integrar grandes cantidades de código y sobrecargar el sistema para observar su comportamiento y sacar nuestras propias conclusiones para suprir errores o mejorar procesos.

Pruebas interfaz de usuario:

Las pruebas de la interfaz de usuario (UI) garantizan que la aplicación cumpla con sus requisitos funcionales y cumple con los estándares de alta calidad, lo que aumenta las posibilidades de una adopción exitosa por parte del usuario.

Resultados: [Pruebas interfaz de usuario](#)

Pruebas de usabilidad:

Son aquellas que verifican cuánto es de fácil el usar el software de cara a los usuarios finales, de esta manera consideramos un software adecuado y que, posiblemente, sea posición por encima de sus competidores principales.
Esta prueba se realiza manualmente.



Referencias

- <https://reactjs.org/docs/testing-components.html>
- <https://reactjs.org/docs/testing-components.html>
- <https://reactjs.org/docs/testing-components.html>

Apéndice C. Casos de uso

a) Diagrama Botai

<https://drive.google.com/file/d/1gg9HnFfN1nAYLQiFuaYete5WzoV1QXp6/view?usp=sharing>

b) Especificación general

https://docs.google.com/document/d/1MkcFf3byTmOTd3jSu5gFZz1fbbMo_4Tnj6jRMNmIO9E/edit?usp=sharing

c) Directorio de APIs

https://docs.google.com/spreadsheets/d/1sGXkYg_0UEEkirCn5REIHeUrC1WUOuCawh1aMOMdxBc/edit?usp=sharing

d) Manual de usuario módulo campañas

https://docs.google.com/document/d/1FznXFK7dBl-9H95WB9rdVdHHeTFa0WV9oF0xjqK_jp4/edit?usp=sharing

e) Manual de configuración módulo campañas

https://docs.google.com/document/d/1WxAWtJZTNifmGWCQE18_jTkc9Bukf3Tm_L6rDVqg9c/edit?usp=sharing

f) Plan de pruebas

<https://docs.google.com/document/d/1xOebpbdyY2DzEYU2oWaccHp16RpQ19WyZMV RgvW1XIA/edit?usp=sharing>

g) Ejecución de pruebas

<https://docs.google.com/spreadsheets/d/1k4yILDgjR3GVj8R3ppUOc46aTEZREjOqsbI6KQSJTEg/edit?usp=sharing>

https://docs.google.com/spreadsheets/d/1CaXO2waViTZkIXfvUTfnGE114V_DIxhd3LC IEmVWoEk/edit?usp=sharing

h) Manual de usuario módulo campañas

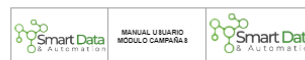


MÓDULO CAMPAÑAS



Tabla de contenido

1. OBJETIVO DEL DOCUMENTO 2
 2. DESCRIPCIÓN DEL MÓDULO 4
 2.1. Acceso a la plataforma 5
 2.2. Recuperación de contraseña 5
 3. PERFIL DE USUARIO 6
 4. FUNCIONAMIENTO 6
 3.1. Creación de campañas 6
 3.1.1. Creación de campaña masiva 9
 3.1.1.1. Creación de campaña masiva y única por el canal de whatsapp 10
 3.1.1.2. Creación de campaña masiva y única por el canal de SMS 24



1. OBJETIVO DEL DOCUMENTO

Campañas es un módulo de la plataforma BOTAI desarrollada por Smart Data & Automation que integra componentes comunicativos con mensajes de difusión a través de plataformas como whatsapp, Facebook, Vioo, Instagram, Twitter, Telegram y Google Business. Estos mensajes son de tipo texto, donde también se integran emojis e imágenes que permitan tener una comunicación más amena con los usuarios finales.

Esta plataforma permite a las empresas difundir información con fines publicitarios, recordatorios, informativos y demás usos que se le puedan dar a la información para que así mismo llegue a las personas que se necesita en tiempos muy cortos, ya que para Smart Data & Automation es muy importante la eficiencia y ética de las tareas que se realizan en el día a día en la plataforma.

El objetivo de este manual es proporcionar una guía que describa detalladamente los principales procesos que se le pueden dar a la plataforma, donde se enseña el correcto manejo en cada campo que se muestra.



2. DESCRIPCIÓN DEL MÓDULO

Este módulo de campañas está compuesto por dos tipos de creaciones de campañas, campañas masivas y campaña única.

La campaña masiva: Es aquella cuando se quiere llegar a varios clientes en un mismo mensaje de difusión, donde se envía una lista de contactos de tipo excel que se sube a la plataforma con el fin de obtener los datos de las personas a las cuales va dirigida.

Ejemplo:

"Hola, somos Smart Data & Automation, recuerda que eres muy importante para nosotros y tu bienestar nos interesa mucho, recuerda que siempre estamos a tu disposición para solucionar los inconvenientes que se presenten."

La campaña única:

Es en uso más específico, donde se quiere llegar a un cliente en especial, como por ejemplo, mensaje especificando algún tipo de cuenta de correo o de información que solo es de uso privado del cliente.

Ejemplo:

"Hola, Pablo Celis, te recordamos que tienes un pago pendiente con fecha de corte al día 10/10/2023 por un valor de \$125.000, ponte al día para evitar multas y sanciones por motivos de reconexión. Para más información te dejamos número de radicado (8800762071) para que te pongas en contacto con nuestros asesores a la línea #950, correo: 2".



2.1. Acceso a la plataforma

El equipo Smart Data & Automation, brindará las credenciales de acceso como correo y contraseña, y la URL de acceso a la plataforma BOTAI. Si se quiere realizar cambios en la URL, y/o en las credenciales de acceso se deberán solicitar directamente a Smart Data & Automation.

Luego de esto se procederá a entrar a la plataforma, se llenan las credenciales y se da clic en el cuadro de "¿Eres un robot?".



Figura 1. Login

2.2. Recuperación de contraseña

En caso de haber olvidado u olvidado su contraseña debe enviar un correo electrónico a support@smartdataandautomation.com, especificando el caso que se presenta e identificándose de cuál empresa pertenece para la posterior verificación de los datos.



3. PERFIL DE USUARIO

- **Administrador:** Este tipo de usuario puede observar, crear y modificar toda la información únicamente de su empresa "Compañía".
- **Reportes:** Este tipo de usuario solo puede observar reportes de su empresa "Compañía".
- **Super usuario reportes:** Este tipo de usuario ve reportes de todas las compañías.
- **Usuario:** Este tipo de usuario solo tiene acceso a flujos, creaciones de bots y sus configuraciones, no puede acceder al módulo campañas whatsapp
- **Super usuario:** Este tipo de usuario tiene todos los permisos, con todas las empresas y todas las creaciones que se puedan realizar en cada módulo

4. FUNCIONAMIENTO

Para entrar al módulo de campañas luego de ingresar las credenciales de acceso nos dirigimos al icono **Campañas**.

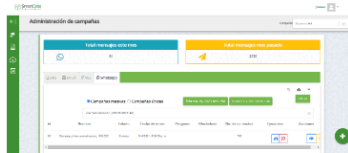


Preparamos a entrar al módulo de campañas

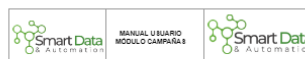


Por defecto aparecerá la de la empresa a la cual están afiliados las credenciales. Para este caso se trabaja con la empresa Guaruma SAS.

Cuando tengamos la empresa asignada en la plataforma ingresamos al menú y vemos las diferentes opciones que tiene



Como podemos ver, se muestran todos los datos de las campañas que se tienen actualmente a las que se han creado, muestra la interacción que ha tenido la campaña en sí mes a través de los medios de difusión como whatsapp y mensaje de texto "SMS", el tipo de campaña que se tiene seleccionado, y las diferentes opciones que muestran el estado actual de la misma como ID, Nombre, Estado, Fecha de envío, Progreso, Bloqueación, No. de contactos, Ejecución, Acciones.



4.1. Creación de campañas

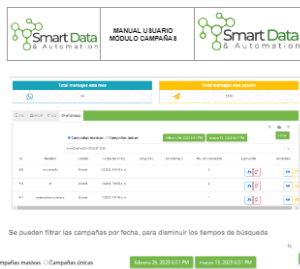
4.1.1. Creación de campaña masiva



Se debe escoger el número de teléfono del cual se va generar la campaña, en este caso la compañía tendrá determinados los números que deben pasar por aprobación para ser usados o en llegado caso, crear o añadir.



Cuando se escoge el número de teléfono se pueden ver las campañas que han sido creadas y el estado actual de cada una de ellas



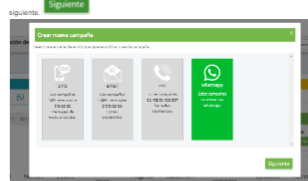
Se pueden filtrar las campañas por fecha, para disminuir los tiempos de búsqueda

4.1.1.1. Creación de campaña masiva y única por el canal de whatsapp

Para crear una nueva campaña, vamos a botón que está ubicado en la parte inferior derecha



Nos desplegará un menú con las opciones donde escogemos whatsapp y presionamos siguiente



Cabe aclarar que los diferentes canales de difusión serán asignados por soporte dependiendo del tipo de plan que se contrata con la empresa, de lo contrario no aparecerán acciones para su posterior funcionamiento.

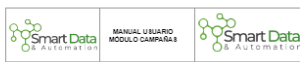


Cuando presionamos en "siguiente", se despliega el menú de la campaña para su creación

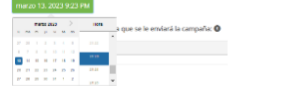


Donde tenemos opciones como el tipo de envío que en este caso será 'Envío masivo con lista de contactos', el nombre de la campaña, de la línea telefónica de la cual se va a realizar el envío, que será enviado inmediato si se quiere enviar al mismo momento de la creación y envío programado donde ingresamos la fecha y hora en la que se quiere enviar la campaña.

Si el envío se quiere realizar programado solo seleccionamos en 'Envío programado' y colocamos la fecha de nuestra preferencia



Seleccione la fecha y hora en la que desea enviar la campaña:



La plantilla es esencial ya que contiene la información que se quiere hacer llegar al cliente y sino se tiene plantillas, se procede a crear una nueva dando clic en el botón

Seleccione la plantilla a utilizar:

Luego nos aparece el menú de las plantillas donde podremos empezar a crearla



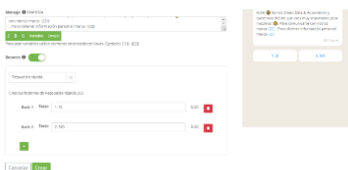
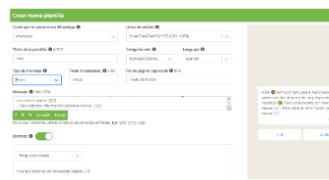
Damos clic en el botón '+' y tenemos este menú que es el de creación de plantillas, donde tenemos todas las opciones de crear una nueva.

Si la parte derecha tendremos una vista previa de la plantilla que se irá modificando dependiendo de los datos que se ingresan.



Completando los campos que existen donde también se pueden añadir variables o si son campañas con el fin de obtener una información de retorno por parte del cliente se activan los botones.

Entonces, seleccionamos el método de envío que sería whatsapp, línea de celular de la cual se va a enviar, se le otorga un título que no permite caracteres especiales, categoría test que tiene opciones Transaccional, Marketing, OTP, lenguaje, tipo de mensaje si es de texto, archivo, video o imagen, texto del encabezado de la plantilla (de algunos opcionales), mensaje (información clara y precisa que va a llegar al cliente donde se pueden anhear emojis, letra en negrita, cursiva o una variable dependiendo si la plantilla lleva opciones).



Luego de crear nuestra plantilla, damos clic en el botón crear y se envía automáticamente para verificación y aprobación.

Cuando nuestra plantilla es aceptada, se verá reflejada de esta manera:

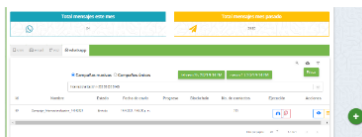


Acá tenemos opciones de visualizar la plantilla, o eliminarla si es el caso.

Luego de que ya tengamos nuestra plantilla creada, regresamos al apartado de creación de campañas y seleccionamos nuestra plantilla.

Llenamos todos los datos, cargamos lista de contacto y plantillas procedemos a darle clic en Enviar

Para ver el estado de nuestro envío, regresamos al menú de campañas y vemos la interacción que ha tenido, mostrando datos en tiempo real



Para seleccionar la opción de campañas únicas y ver el proceso, sería seleccionar envío Único, y nos muestra lo siguiente:

Seleccione el tipo de envío:



Seleccionamos la línea telefónica de cual se va a enviar, se selecciona el tipo de envío si es inmediato o envío programado como se explicó anteriormente, se ingresa el número al cual se va a enviar y de presionamos en agregar único



El número se cargó correctamente y se procede a escoger la plantilla a enviar

Si no se cuenta con plantillas, se debe crear una nueva plantilla como se explicó anteriormente.

Cuando tengamos todos los datos listos, se procede a darle en el botón enviar

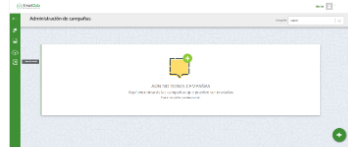
Para verificar que nuestra plantilla fue enviada correctamente vamos al menú de campañas nuevamente y seleccionamos campaña única





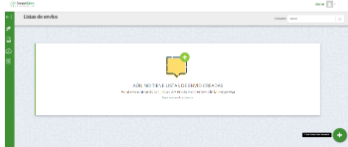
Donde se podrá observar el estado actual de la plantilla y sus datos.

Para enviar campañas masivas se debe contar con listas de envíos o lista de contactos, donde se pueden ver en el apartado de campañas y dentro de este menú vamos a la parte izquierda y seleccionamos



Las listas de envío si no se han creadas se deben crear como se muestra a continuación, dando clic en el botón que se encuentra en la parte inferior derecha.

Presionamos en el botón

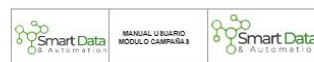


Se debe activar que este servicio debe estar activo por parte de smart Data & Automation para su posterior visualización, de lo contrario aparecerá una ventana emergente como la que se presenta a continuación.



Si se quiere acceder al servicio, se da en la opción de activar activación de campañas y se procede a comunicarse con el equipo de soporte.

Cuando entramos en este apartado nos muestra si tenemos listas de campañas y también las podemos visualizar



Como se quiere crear una nueva damos clic en el siguiente menú de opciones para crear la lista de contactos



Colocamos un título, canal donde se usará esta lista, se sube el archivo que contiene la información de tipo excel preferiblemente ya que la plataforma los reconoce rápidamente y extrae los datos sin problemas, en integración es el número al cual se asociará la lista, se escoge la columna en la cual están ubicados los números telefónicos.

Ejemplo de lista de contactos:



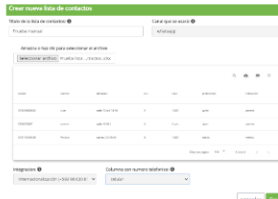
	A	B	C	D	E	F	G
1	Nombre	Apellido	Edad	Sexo	Profesión	Industria	
2	SPYONANNA	JOHN	23	M	profesional	general	
3	ESTHER	JOHANNA	24	F	profesional	general	
4	SPYONANNA	JOHANNA	24	F	profesional	general	

Luego de seleccionar nuestra lista de contactos, nos muestra una vista previa de la misma

Escogemos la integración del número al cual se va a asociar la lista de contactos y la columna que contiene los números telefónicos que en este caso es "teléfono". El sistema detecta el excel y nos muestra el nombre de las columnas que contiene para nosotros elegir.



Finalmente con todos los datos obtenidos creamos la nueva lista de contactos



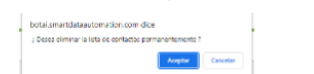
La lista pasa por aprobación rápidamente y se carga automáticamente como se puede apreciar en la imagen.



En esta imagen podemos ver dos listas de contactos creadas donde se pueden visualizar

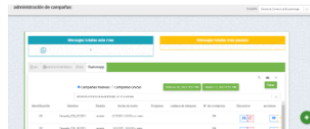
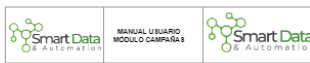


O eliminar según el caso donde nos aparece una ventana emergente de Bata! con el fin de confirmar la opción que se escoge

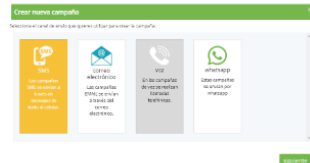


4.1.1.2 Creación de campaña masiva y unica por el canal de SMS

Para crear una campaña por medio de sms, vamos al botón



Seleccionamos SMS y damos en siguiente (Es es el mismo paso de crear campañas de whatsapp)



Nos muestra la interfaz de este modulo, y procedemos a llenar la información



Este tipo de canal dispone de dos métodos de envío que serían "Envío rápido (máximo 10 números telefónicos)".



Este tipo de envío, lleva un nombre, tipo de envío si es simplificado o automatizado, se deben colocar los números de teléfonos a los cuales se quiere hacer llegar la información y se da en "programar" para automático, luego se escoge la plantilla y para su envío, se debe crear una como se muestra anteriormente, se puede enviar de inmediato en "Enviar ahora" o "programador" si se quiere enviar en una fecha o hora diferente como también se enseñó anteriormente.

(Se aclara que ese canal se encuentra en desarrollo y presenta ciertas fallas para su funcionamiento).



3.1.1.3 Creación de campaña masiva y unica por el canal de Email

(Este se encuentra en desarrollo y aún no cuenta con una interfaz gráfica).

5. PREGUNTAS FRECUENTES

(En este apartado van las preguntas frecuentes que se presentan y sus posibles soluciones, se debe mostrar pantallazos con los errores y las conexiones que realzan.)

El manual de usuario del módulo campañas es un documento que se realiza con la finalidad de poder brindarle a los clientes o usuarios de la plataforma una guía rápida de cómo se utilizan los componentes del sistema, los módulos que se tienen, como crear nuevas campañas masivas o únicas, también cómo crear plantillas, asignar números telefónicos, crearlos, ver los reportes de cada campaña.

Este documento también sirve para capacitar a nuevo personal que ingrese a la empresa, reduciendo así los tiempos en capacitaciones y desgaste del personal existente que debe desarrollar otras tareas.

Esto es un claro ejemplo que puede servir de guía para nuevos desarrollos dentro de la empresa ya que teniendo una base como lo es el manual de usuario, se tiene una guía básica de cómo se deben realizar los manuales para otros proyectos que se realicen.

i) Manual de configuración módulo campañas

Se crea el manual de configuración del módulo de campañas con la finalidad de explicar el paso a paso de cómo se debe configurar el entorno en una nueva máquina para poder trabajar en el proyecto. Manual de configuración

Esta manual consta de tres pasos importantes para la configuración del entorno de desarrollo:

El primer paso es configurar una nueva máquina, que se debe dirigir al manual de configuración de BOTAI para ver los pasos a seguir en donde explica detalladamente como configurar el módulo de campañas desde cero.

El segundo paso es realizar un release a producción, y para ejecutar la configuración del módulo de campañas se debe compilar primero el FrontEnd donde se especifican los códigos que se deben ingresar a través de CMD o terminal.

Luego de realizar estos pasos se debe actualizar imagen, en el manual de configuración BOTAI en la plantilla de máquinas escaladas y subir los cambios a producción se encuentran los detalles para este proceso.

El tercer paso luego de compilar el FrontEnd, se procede a la configuración del FrontEnd, donde se especifican los códigos que se deben ingresar a través de un CMD o terminal.

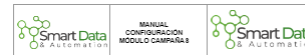
En este mismo paso se debe configurar el BackEnd ya que de este depende poder clonar el proyecto sobre el cual se va a trabajar mediante los comandos Git que se muestran en el manual.

Para finalizar este proceso se debe verificar que todo esté correctamente funcional, entrando a la aplicación y probando los componentes, si algo funciona mal, se recomienda volver a seguir los pasos anteriores y ser muy detallado en la descripción de cada código.



Tabla de contenido

1. PASOS PARA CONFIGURAR UNA NUEVA MÁQUINA	3
2. PASOS PARA HACER UN RELEASE A PRODUCCIÓN	5
2.1. COMPILAR EL FRONTEND	5
2.2. CONFIGURACIÓN DEL BACKEND	5
2.3. ACTUALIZAR IMAGEN	6
3. CONFIGURACIÓN DEL ENTORNO DE DESARROLLO	6
3.1. Configuración del frontend	6
3.2. Configuración del backend	6



1. PASOS PARA CONFIGURAR UNA NUEVA MÁQUINA

Para configurar el módulo de campañas desde cero, diríjase al manual de configuración de BOTAI

2. PASOS PARA HACER UN RELEASE A PRODUCCIÓN

Para ejecutar la configuración del módulo de campañas debe realizar la siguiente configuración en la máquina master

2.1. COMPILAR EL FRONTEND

Paso 1. Ingresar a una terminal dependiendo del sistema operativo

Paso 2. se realiza un git clone al proyecto de fronteras campañas para donarlo "poner comando y url".

Paso 3. compilar el proyecto

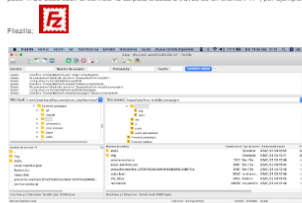
```
cd /frontend/campaigns/
```

```
yarn install es yarn build;
```

al terminal de ejecutar este comando se genera una carpeta llama "build"

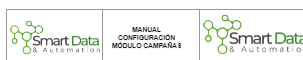


paso 4. Se debe subir al servidor la carpeta creada a través de un cliente FTP. por ejemplo



La ruta en la cual deben quedar los archivos es la siguiente

```
/home/data/frontend-dist/campaigns
```



2.2. CONFIGURACIÓN DEL BACKEND

Paso 1. Conectarse a través de SSH a la máquina master "ingresar comando"

```
sudo ssh -i /Users/SmartData/Downloads/appsflyer.com -p 22 -i user@192.168.1.100 -i 10
```

Paso 2. Ingrese a la carpeta del proyecto y verifique que se encuentre en la rama master

```
cd /services/campaign_backend/ git branch
```

nota: En caso de encontrarse en otra rama se debe mover con el comando

```
cd /services/campaign_backend/ git checkout master
```

Paso 3. Actualice el proyecto del Backend de campañas con el comando

```
cd /services/campaign_backend/ git pull
```

Paso 4. Construya el contenedor con el comando

```
cd /home/ec2-user/ docker-compose build campaign_backend
```

Paso 5. Suba y reinicie con contenedores para tomar los cambios

```
docker-compose up -d, docker-compose restart
```



2.3. ACTUALIZAR IMAGEN

Diríjase al manual de configuración de BOTAI para actualizar la plantilla de las máquinas auto escaladas y subir los cambios a producción

3. CONFIGURACIÓN DEL ENTORNO DE DESARROLLO

3.1. Configuración del frontend

Paso 1. Se debe clonar el proyecto a través de una terminal con el comando

```
git clone https://git-codecommit.us-east-1.amazonaws.com/v1/repos/frontend_campaigns
```

paso 2. Verificar la rama en la que se encuentra ingresando a la carpeta del proyecto con el comando

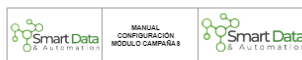
```
cd /frontend/campaigns/
```

Luego de verificar la rama debe estar en development con el comando

```
git branch
```

Paso 3. Realice la configuración del docker, para ello crea el archivo dockerfile dentro de la carpeta frontend_campaigns

```
FROM node:12
WORKDIR /usr/src/app
COPY package.json ./
RUN npm install
COPY . .
CMD ["npm", "start"]
```



```
dockerfile
FROM node:12
WORKDIR /usr/src/app
COPY package.json ./
RUN npm install
COPY . .
CMD ["npm", "start"]
```

Paso 4. Configure el docker compose con el nuevo contenedor

```
services:
  frontend_campaigns:
    build: ./frontend_campaigns/dockerfile
    ports:
      - "3000:3000"
    restart: always
```



Paso 5. Clone el proyecto static frontend con el comando

```
git clone https://git-codecommit.us-east-1.amazonaws.com/v1/repos/static_frontend
```

verificar la rama en la que se encuentra sea la rama de desarrollo

```
cd /static/frontend/
```

Luego de verificar la rama debe estar en development con el comando

```
git branch
```

Paso 6. Realice la configuración del docker, para ello crea el archivo dockerfile dentro de la carpeta static_frontend

```
FROM node:12
WORKDIR /usr/src/app
COPY package.json ./
RUN npm install
COPY . .
CMD ["npm", "start"]
```

Actualice el archivo docker_compose.yml, incluyendo el nuevo contenedor

```
services:
  static_frontend:
    build: ./static_frontend/dockerfile
    ports:
      - "3000:3000"
    restart: always
```



```
dockerfile
FROM node:12
WORKDIR /usr/src/app
COPY package.json ./
RUN npm install
COPY . .
CMD ["npm", "start"]
```

Paso 7. Se debe construir los contenedores creando con el comando

```
cd /home/ec2-user/ docker-compose build campaign_backend
```

Paso 8. suba y reinicia los contenedores con el comando

```
docker-compose up -d, docker-compose restart;
```

3.2. Configuración del backend

Paso 1. Se debe clonar el proyecto a través de una terminal con el comando

```
git clone https://git-codecommit.us-east-1.amazonaws.com/v1/repos/backend_campaigns
```

paso 2. Verificar la rama en la que se encuentra ingresando a la carpeta del proyecto con el comando

```
cd /backend/campaigns/
```

Luego de verificar la rama debe estar en development con el comando

```
git branch
```

Paso 3. Realice la configuración del docker, para ello crea el archivo dockerfile dentro de la carpeta backend_campaigns

