

UNIVERSIDAD CENTRAL DE VENEZUELA
FACULTAD DE CIENCIAS
ESCUELA DE COMPUTACIÓN



DESARROLLO DE UNA SOLUCIÓN CON TECNOLOGÍA WEB PARA LA ELABORACIÓN DE CONTRATOS EN EMPRESAS INMOBILIARIAS.

Trabajo Especial de Grado presentado ante la ilustre

Universidad Central de Venezuela por

Br. Nebraska Estephfany Meléndez Azócar

Para optar al título de Licenciado en Computación

Tutor:

Prof. Franky Uzcátegui

Caracas, Julio 2018

UNIVERSIDAD CENTRAL DE VENEZUELA

FACULTAD DE CIENCIAS

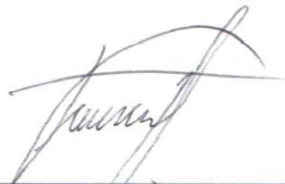
ESCUELA DE COMPUTACIÓN

ACTA

Quienes suscriben, miembros del jurado designado por el Consejo de la Escuela de Computación, para examinar el Trabajo Especial de Grado titulado **“Desarrollo de una solución con tecnología web para la Elaboración de Contratos en Empresas Inmobiliarias.”** y presentado por la Br. **Nebraska Estephfany Meléndez Azócar** titular de la Cédula de Identidad **V-22.014.392**, a los fines de optar al título de **Licenciado en Computación**, dejamos constancia de lo siguiente:

Leído como fue dicho trabajo, por cada uno de los miembros del jurado, se fijó el día 28 de MAYO de 2018, a las 3:30 PM horas, para que la autora lo defendiera en forma pública, lo que esta hizo en PB3 de la Escuela de Computación, mediante una presentación oral de su contenido, luego de lo cual respondiera a las preguntas formuladas. Finalizada la defensa pública del Trabajo Especial de Grado, el jurado decidió aprobar con la nota de 18 puntos.

En fe de lo cual se levanta la presente Acta, en Caracas el día 28 de MAYO de 2018.



Prof. Franky Uzcátegui

(Tutor)



Profa. Mercy Ospina

(Jurado)



Profa. Yusneyi Carballo Barrera

(Jurado)

DEDICATORIA

A mis padres, Debora y Omar por siempre apoyarme y creer en mí.

A mis amigos y familiares por siempre darme ánimos y fuerzas cuando más lo necesité.

A mis compañeros de trabajo por no preguntarme “¿Cómo va la tesis?” y por ayudarme a ser una mejor profesional.

A mis ángeles en el cielo, este logro también es de ustedes.

Nebraska Meléndez.

AGRADECIMIENTOS

Principalmente a mi madre, por ser mi ejemplo a seguir, por enseñarme que las metas se pueden lograr si trabajamos por ellas, gracias por motivarme, por nunca dejar que me rindiera y por inculcarme este espíritu Ucevista. A mi padre, por enseñarme a ser una mujer fuerte e independiente, gracias por todas las enseñanzas.

A mis familiares, en especial a mi hermana Bianca, por enseñarme de sus errores y por siempre estar allí para mí cuando la necesito. Gracias a todos por confiar en mí y en que alcanzaría mis metas.

A mis amigos, gracias por ofrecerme su amistad, por impulsarme a seguir y brindarme momentos de alegría y tranquilidad cuando más lo necesité, por su compañía y comprensión a lo largo de todo este camino. En especial quiero agradecer a mis hermanas que me dio la vida, Silvia, Giovanna, Hayleen y Ligia, gracias por estos once años de amistad y su apoyo incondicional. A mis amigos de la universidad, amigos que conservaré y valoraré el resto de mi vida, colegas y futuros colegas. ¡Si se puede!

A Juan Vásquez, gracias por ser mi compañero en este camino, gracias por apoyarme, y nunca dejarme caer, estoy segura que sin tu apoyo incondicional este camino hubiese sido mucho más difícil y sin tan buenos y hermosos momentos, este logro también te pertenece.

A mis profesores, gracias por darme las herramientas no solo para triunfar en mi carrera, sino en la vida, gracias por ayudarme a desarrollarme como persona y como profesional, sin duda tienen una de las labores más admirables que conozco. Gracias en especial a mi tutor Franky Uzcategui, por su paciencia, su orientación y sin duda apoyo en este trabajo especial de grado. Gracias por su guía hacia uno de mis más grandes sueños.

Finalmente pero no menos importante a mi amada Universidad Central de Venezuela, gracias por ser mi segunda casa, por brindarme una de las mejores experiencias de mi vida, por verme crecer como persona y profesional, espero dejar en alto el nombre de tan grande institución.

Gracias a Dios, y a todos, mi más sincero agradecimiento.

UNIVERSIDAD CENTRAL DE VENEZUELA

FACULTAD DE CIENCIAS

ESCUELA DE COMPUTACIÓN

**DESARROLLO DE UNA SOLUCIÓN CON TECNOLOGÍA WEB PARA LA
ELABORACIÓN DE CONTRATOS EN EMPRESAS INMOBILIARIAS.**

Autor: Nebraska Estephfany Meléndez Azócar.

Tutor: Prof. Franky Uzcategui Polo.

Fecha: Caracas, Mayo 2018.

RESUMEN

En la actualidad las empresas inmobiliarias buscan soluciones de flujo de trabajo probadas para automatizar sus procesos, con el fin de introducir el uso de tecnologías eficientes en el área legal. Para ello, debe automatizarse el proceso de creación de contratos legales para incorporarle la agilidad necesaria. El presente trabajo de investigación propone elaborar una solución con tecnologías web que permita a los abogados y/o representantes legales realizar este proceso. Para la planificación y desarrollo de esta solución se realizó una adaptación de la metodología Scrum. Se utilizaron diversas herramientas, entre ellas el *framework* Angular en su versión 4, el *framework* minimalista Python Flask y como sistema manejador de base de datos MongoDB.

Palabras clave: Proceso, Tecnología Web, Contrato legal, Inmobiliaria, Scrum, Proceso Ágil de Desarrollo.

ÍNDICE DE CONTENIDO

INTRODUCCIÓN.....	1
Capítulo 1	3
Planteamiento del Problema	3
1.1 Planteamiento del problema	3
1.2 Objetivos del TEG	4
1.2.1 Objetivo general	4
1.2.2 Objetivos específicos.....	4
1.3 Justificación	4
1.4 Alcance	5
Capítulo 2	6
Marco Conceptual.....	6
2.1 Gerencia de procesos de negocio	6
2.2 Proceso de negocio	6
2.3 Base de Datos	7
2.4 Bases de datos relacionales.....	7
2.5 Sistema manejador de base de datos	7
2.5.1 PostgreSQL	8
2.5.2 Oracle Database.....	9
2.5.3 Microsoft SQL server.....	9
2.5.4 MySQL.....	9
2.6 Bases de datos No SQL	10
2.7 Tipos de Bases de datos NoSQL	10
2.7.1 Bases de datos clave-valor	10
2.7.2 Bases de datos Documentales	10
2.7.3 Bases de datos de Grafos.....	11
2.8 Tipos de Bases de datos NoSQL	11
2.8.1 Bases de datos clave-valor	11
2.8.2 Bases de datos Documentales	11
2.8.3 Bases de datos de Grafos.....	11
2.9 Ejemplos de Bases de datos No SQL	12

2.9.1	Apache Cassandra	12
2.9.2	MongoDB.....	12
2.9.3	Neo4j.....	13
2.10	Tecnologías de desarrollo Web	13
2.10.1	Lenguaje de Marcas de Hipertexto (HTML).....	13
2.10.2	Hojas de Estilo en Cascada (CSS).....	13
2.10.3	JavaScript	14
2.10.4	Ruby on Rails	14
2.10.5	Angular.....	15
2.10.6	Python Flask.....	15
2.10.7	Node.js.....	15
2.10.8	Django	15
2.10.9	CakePHP	17
2.10.10	React	18
2.10.11	Froala	18
2.10.12	TinyMCE	18
2.10.13	CKeditor	19
2.10.14	Alfresco	19
2.10.15	Nuxeo	20
2.10.16	XML	20
2.10.17	JSON.....	21
2.10.18	Aplicación de página única	22
2.10.19	Modelo - Vista – Controlador.....	22
2.10.20	Servicio Web	23
2.10.21	Protocolo SOAP	23
2.10.22	REST	24
2.11	Contrato	25
2.11.1	Arrendador	26
2.11.2	Arrendatario	26
2.11.3	Contrato de arrendamiento	26
2.11.4	Clases de arrendamientos	26
2.11.5	Contrato de renovación	27
2.11.6	Bases legales del contrato de arrendamiento de locales comerciales.....	27

Capítulo 3	29
Marco Metodológico	29
3.1 Scrum.....	29
3.1.1 Características	30
3.1.2 Fases	30
3.2 Metodología utilizada para el desarrollo	31
Capítulo 4	32
Marco Aplicativo.....	32
4.1 Descripción general de la solución.....	32
4.2 Aplicación de la metodología Scrum.....	34
4.2.1 Lista de Objetivos.....	34
4.2.2 Lista de actividades de la iteración (Pila de Sprint)	36
4.3 Requerimientos del sistema	37
4.3.1 Requerimientos funcionales	37
4.3.2 Requerimientos no funcionales	37
4.4 Descripción del flujo asociado a la solución	38
4.4.1 Proceso de creación de plantillas	38
4.4.2 Proceso de creación de contratos	39
4.5 Análisis del modelo de datos y definición.....	40
4.5.1 Listado de colecciones de la aplicación	40
4.5.2 Modelo de datos	40
4.6 Servicios web.....	43
4.7 Descripción de los módulos del sistema y sus interfaces	45
4.7.1 Descripción de los módulos	45
4.7.2 Interfaces del sistema web.....	46
4.8 Fase de pruebas.....	56
4.8.1 Pruebas funcionales.....	56
4.8.2 Pruebas de aceptación	60
CONCLUSIONES Y RECOMENDACIONES	66
BIBLIOGRAFÍA	68

ÍNDICE DE FIGURAS

Figura 1. <i>Proceso SCRUM. (Tomada de Softeng, 2017)</i>	30
Figura 2. <i>Arquitectura propuesta Fuente: La Autora</i>	33
Figura 3. <i>Proceso de creación de plantillas de contratos. Fuente: Elaborado por la autora.</i>	38
Figura 4. <i>Proceso de creación de contratos. Fuente: Elaborado por la autora.</i>	39
Figura 5. <i>Módulos del sistema web. Fuente: Elaborado por la autora</i>	45
Figura 6. <i>Interfaz Inicio de Sesión. Fuente: Captura de pantalla.</i>	46
Figura 7. <i>Interfaz Inicial creación de contratos. Fuente: Captura de pantalla</i>	46
Figura 8. <i>Interfaz (Modal) Creación de plantilla. Fuente: Captura de pantalla</i>	47
Figura 9. <i>Interfaz Editor de texto para la creación de plantilla de contrato. Fuente: Captura de pantalla.</i>	48
Figura 10 . <i>Interfaz de creación de cláusulas de contratos. Fuente: Captura de pantalla.</i>	48
Figura 11. <i>Notificación cláusula creada. Fuente: Captura de pantalla.</i>	49
Figura 12. <i>Interfaz de creación de plantilla de contrato con información agregada. Fuente: Captura de pantalla.</i>	49
Figura 13. <i>Notificación de plantilla guardada exitosamente. Fuente: Captura de pantalla.</i>	50
Figura 14. <i>Interfaz de tabla de selección de participantes de un contrato. Fuente: Captura de pantalla.</i>	50
Figura 15. <i>Interfaz siguiente luego de la selección de participantes de un contrato. Fuente: Captura de pantalla.</i>	51
Figura 16. <i>Interfaz Creación de contrato. Fuente: Captura de pantalla.</i>	51
Figura 17. <i>Notificación de contrato guardado exitosamente. Fuente: Captura de pantalla.</i>	52
Figura 18. <i>Interfaz modal de la opción de finalizar contrato. Fuente: Captura de pantalla.</i>	52
Figura 19. <i>Notificación de contrato guardado exitosamente. Fuente: Captura de pantalla.</i>	53
Figura 20. <i>Interfaz que Lista los contratos actuales. Fuente: Captura de pantalla</i>	53
Figura 21. <i>Interfaz Opción de borrar contrato. Fuente: Captura de pantalla</i>	54
Figura 22. <i>Interfaz modal de la opción de borrar contrato. Fuente: Captura de pantalla.</i>	54
Figura 23. <i>Notificación de contrato borrado exitosamente. Fuente: Captura de pantalla.</i>	55
Figura 24. <i>Interfaz que Lista los contratos actuales. Fuente: Captura de pantalla</i>	55
Figura 25. <i>Interfaz de contrato en formato pdf. Fuente: Captura de pantalla.</i>	56
Figura 26. <i>Resultados de la pregunta N° 1. Fuente: Pruebas de aceptación con usuarios finales.</i>	61

Figura 27. Resultados de la pregunta N° 2. Fuente: Pruebas de aceptación con usuarios finales.....	62
Figura 28. Resultados de la pregunta N° 3. Fuente: Pruebas de aceptación con usuarios finales.....	62
Figura 29. Resultados de la pregunta N° 4. Fuente: Pruebas de aceptación con usuarios finales	63
Figura 30. Resultados de la pregunta N° 5. Fuente: Pruebas de aceptación con usuarios finales.....	63
Figura 31. Resultados de la pregunta N° 6. Fuente: Pruebas de aceptación con usuarios finales.....	64
Figura 32. Resultados de la pregunta N° 7. Fuente: Pruebas de aceptación con usuarios finales.	64

ÍNDICE DE TABLAS

Tabla 1. <i>Pila del producto. Fuente: Elaborada por la autora.</i>	34
Tabla 2. <i>Pila del producto (Continuación). Fuente: Elaborado por la autora.....</i>	35
Tabla 3. <i>Instrumento evaluación de módulos del sistema. Fuente: Elaborado por la autora.</i>	57
Tabla 4. <i>Matriz de ciclos de pruebas. Fuente: Elaborado por la autora.....</i>	58
Tabla 5. <i>Encuesta de aceptación. Fuente: Elaborado por la autora.</i>	60
Tabla 6. <i>Encuesta de aceptación (Continuación). Fuente: Elaborado por la autora....</i>	61

INTRODUCCIÓN

Actualmente las empresas inmobiliarias necesitan hacer sus procesos legales más rápido, a un menor costo y con mayor transparencia. Se requiere la capacidad de automatizar el proceso de creación de contratos inmobiliarios para mantener la competitividad de la industria. Las empresas inmobiliarias buscan ofrecer soluciones de flujo de trabajo probadas para automatizar sus procesos. Además, de introducir el uso de tecnologías eficientes en el área legal. Estas tecnologías podrían ser aplicadas a diferentes tipos de procesos legales, los cuales ya poseen una estructura previamente establecida que generalmente se mantienen durante su creación. Se propone el uso de tecnologías web con la finalidad de desarrollar una aplicación que permita a sus usuarios conectarse mediante internet para posteriormente obtener los datos necesarios para la creación de contratos. Esta aplicación le proporcionará al cliente la posibilidad de almacenar información referente a los contratos previamente creados, así como también cláusulas previamente establecidas.

Para profundizar en el desarrollo de la aplicación web anteriormente descrita, el presente documento de Trabajo Especial de Grado se encuentra dividido en cuatro capítulos que se detallarán a continuación:

- Capítulo 1 (Problema de la Investigación) en este capítulo se expone la problemática que se quiere solventar a través del presente Trabajo Especial de Grado y la solución que se desarrolla para solventarla. De igual forma se presenta el objetivo general, objetivos específicos, justificación y alcance de este trabajo.
- Capítulo 2 (Marco Conceptual) en este capítulo se presentan conceptos relacionados con la Gerencia de procesos de negocio, Bases de Datos, Arquitectura Cliente – Servidor, Aplicaciones Web y las principales tecnologías utilizadas para el desarrollo de la solución.
- Capítulo 3 (Marco Metodológico) en este capítulo se presenta la Metodología de Desarrollo SCRUM, la cual fue utilizada para guiar el desarrollo de la Aplicación Web sobre la cual se centró el Trabajo Especial de Grado.

- Capítulo 4 (Marco Aplicativo) en este capítulo se presenta la Metodología de Desarrollo SCRUM adaptada a las necesidades de la Aplicación Web desarrollada, detallando las iteraciones llevadas a cabo.

Para finalizar se presentan conclusiones, recomendaciones y referencias bibliográficas del presente Trabajo Especial de Grado.

Capítulo 1

Planteamiento del Problema

En el presente capítulo se detalla la problemática por la cual surge el siguiente trabajo especial de grado, además de la definición de su objetivo general y objetivos específicos, incluyendo la justificación por la cual se realiza el trabajo especial de grado y por último el alcance del mismo.

1.1 Planteamiento del problema

Actualmente los procesos de negocios realizados en una empresa inmobiliaria seleccionada como caso de estudio, se tiene que dichos procesos se realizan mediante un sistema BPMN (Modelo y Notación de Procesos de Negocio) a través del cual se realizan diversos procesos pertenecientes a la organización, dentro de los cuales se encuentra el almacenamiento de los datos de los participantes en la creación de contratos inmobiliarios. Sin embargo se tienen algunas limitantes, como es el caso del proceso de creación de contratos inmobiliarios, ya que los abogados encargados de la creación de dichos contratos realizan los contratos utilizando herramientas como el procesador de texto Microsoft Word, pero para obtener los datos de los participantes se debe realizar todo el proceso de consulta de datos del sistema, para luego transcribir los datos al contrato que está siendo creado en Microsoft Word, además de los números formatos de contratos que son creados, generan una gran cantidad de archivos, lo que dificulta el proceso de creación y selección de formato de contratos. También se tiene la necesidad de añadir las cláusulas de contratos inmobiliarios, que puedan ser utilizadas en cualquier contrato inmobiliario que lo requiera.

Ante la situación planteada, el problema observado describe la necesidad de la integración de un módulo de solución de manejo y creación de contratos inmobiliarios, al sistema existente implementado en una solución BPMN, agregándole a este sistema la funcionalidad que permita realizar el proceso de creación de contratos a través del

mismo sistema debido a una implementación del uso de arquitecturas basadas en servicios web. El sistema existente podrá realizar consultas sobre los contratos creados, permitiendo así que los módulos del sistema ya implementado se alimenten de la información generada por este sistema de creación de contratos inmobiliarios.

1.2 Objetivos del TEG

1.2.1 Objetivo general

Desarrollar una solución con tecnología web para la elaboración de contratos en empresas inmobiliarias.

1.2.2 Objetivos específicos

- Analizar requerimientos del proceso de elaboración de contratos inmobiliarios.
- Diseñar los artefactos y vistas de la aplicación web.
- Desarrollar servicios web y módulo de creación y edición de contratos.
- Realizar pruebas funcionales y de aceptación de usuario.

1.3 Justificación

Un sistema con tecnología web, es una herramienta clave para automatizar el proceso de creación de contratos legales debido a que se aprovecha el uso de tecnologías eficientes para transformar un proceso tedioso que puede ser realizado de una manera óptima, utilizando de mejor manera los recursos.

Teniendo en cuenta que en la actualidad y con los avances de la tecnología la mayoría de los abogados y/o representantes legales, utilizan computadoras o dispositivos tecnológicos para desempeñar su actividad laboral o en su cotidianidad, la implementación de esta solución, resulta de utilidad con la finalidad de ayudarles a realizar este proceso con la mayor comodidad posible, ahorrándoles los inconvenientes que puedan presentarse en su desarrollo. En general un sistema de automatización de contratos representa la capacidad de reducir y estandarizar los tiempos de duración del proceso, así como facilitar el proceso convirtiendo la creación de contratos legales en algo práctico y sencillo.

1.4 Alcance

Este proyecto se basa en la realización de una plataforma web, que permitirá la automatización del proceso de elaboración de contratos en empresas inmobiliarias, brindando así un avance tecnológico en esta área, que para los abogados y/o representantes legales esta aplicación será fácil de usar, y que mejorará los tiempos en la creación de dichos contratos.

Capítulo 2

Marco Conceptual

En este capítulo se expondrán los diferentes conceptos asociados a las tecnologías, enfoques y herramientas de desarrollo. Además de los términos orientados al área de Creación de contratos, que es donde está enfocada la solución de este proyecto.

2.1 Gerencia de procesos de negocio

Es un enfoque empresarial operativo basado en la coordinación de las actividades y decisiones que todas las partes involucradas deben realizar durante un proceso de negocio con el objetivo de convertirse en una organización altamente eficiente, ágil, innovadora y adaptable. (Universidad Central de Venezuela, Sistemas de Información, 2012)

Es un enfoque sistemático para identificar, levantar, documentar, diseñar, ejecutar, medir y controlar tanto los procesos manuales como automatizados, con la finalidad de lograr a través de sus resultados en forma consistente los objetivos de negocio que se encuentran alineados con la estrategia de la organización. BPM abarca el apoyo creciente de TI con el objetivo de mejorar, innovar y gestionar los procesos de principio a fin que determinan los resultados de negocio, crean valor para el cliente y posibilitan el logro de los objetivos de negocio con mayor agilidad. (Freund J., Rücker B y Hitpass B, 2011)

2.2 Proceso de negocio

Un Proceso de Negocio (Business Process) es un conjunto de actividades que son realizadas en coordinación en entorno organizacional y técnico. (Weske. M, 2007)

Un conjunto estructurado, medible de actividades diseñadas para producir un producto especificado, para un cliente o mercado específico. Implica un fuerte énfasis en cómo se

ejecuta el trabajo dentro de la organización, en contraste con el énfasis en el que, característico de la focalización en el producto” (Davenport, Thomas, 1993).

Un proceso de negocio contiene actividades con propósito, es ejecutado colaborativamente por un grupo de trabajadores de distintas especialidades, con frecuencia cruza las fronteras de un área funcional, e invariablemente es detonado por agentes externos o clientes de dicho proceso. (Ould. Martyn, 1995).

2.3 Base de Datos

Una base de datos es un conjunto de datos relacionados pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. Actualmente, y debido al desarrollo tecnológico de campos como la informática y la electrónica, la mayoría de las bases de datos están en formato digital, siendo este un componente electrónico, y por ende se ha desarrollado y se ofrece para un amplio rango de soluciones al problema del almacenamiento de datos. (Wikipedia, 2018)

2.4 Bases de datos relacionales

Una base de datos relacional es una base de datos que se trata como un conjunto de tablas y se manipula de acuerdo con el modelo de datos relacional. Contiene un conjunto de objetos que se utilizan para almacenar y gestionar los datos, así como para acceder a los mismos. Las tablas, vistas, índices, funciones, activadores y paquetes son ejemplos de estos objetos. (IBM Knowledge Center, 2018)

2.5 Sistema manejador de base de datos

De acuerdo a Korth, Silberschatz y Sudarshan (2002) los sistemas manejadores de Base de Datos consisten en una colección de datos interrelacionados y un conjunto de programas para acceder a dichos datos. La colección de datos, normalmente denominada base de datos, contiene información relevante para una empresa. El objetivo principal de un SGBD es proporcionar una forma de almacenar y recuperar la información de una base de datos de manera que sea tanto práctica como eficiente.

Los sistemas de bases de datos se diseñan para gestionar grandes cantidades de información. La gestión de los datos implica tanto la definición de estructuras para almacenar la información como la provisión de mecanismos para la manipulación de la

información. Además, los sistemas de bases de datos deben proporcionar la fiabilidad de la información almacenada, a pesar de las caídas del sistema o los intentos de acceso sin autorización. Si los datos van a ser compartidos entre diversos usuarios, el sistema debe evitar posibles resultados anómalos.

Ospina (2009) describe las funciones que debe cumplir un SMBD:

1. **Definición de datos:** El SMBD debe ser capaz de aceptar definiciones de datos en versión fuente y convertirlas en la versión objeto adecuada. Dicho de otro modo, el SMBD debe incluir componentes procesadores de lenguajes para cada uno de los diversos lenguajes de definición de datos (DDL).
2. **Manipulación de datos:** El SMBD debe ser capaz de atender las solicitudes del usuario para extraer, poner al día, datos que ya existen en la base de datos o para agregar en ella datos nuevos. Dicho de otro modo, el SMBD debe incluir componentes procesadores de lenguajes para cada uno de los diversos lenguajes de manipulación de datos (DML).
3. **Seguridad e integridad de los datos:** El SMBD debe supervisar las solicitudes de los usuarios y rechazar los intentos de violar las medidas de control y seguridad definidas por el DBA.
4. **Control de concurrencia de los datos:** El SMBD debe proporcionar un mecanismo para garantizar que la base de datos se actualice correctamente cuando haya varios usuarios realizando modificaciones de manera concurrente en la base de datos.
5. **Recuperación:** El SMBD debe poder recuperar la base de datos en caso de que esta resulte dañada de alguna forma.
6. **Diccionario de Datos accesible por el usuario:** El SMBD debe incluir un catálogo en el que se almacenen las descripciones de los datos y que sea accesible por parte de los usuarios.

A continuación se presentarán algunos ejemplos de SMBD.

2.5.1 PostgreSQL

Es un sistema de gestión de bases de datos con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado,

utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema, por lo que si se presentase algún tipo de fallo en uno de los procesos, esto su vez no afectaría el resto de los mismos y el sistema continuará funcionando con normalidad. Adicionalmente, PostgreSQL funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema. (Martínez. R, 2010)

2.5.2 Oracle Database

Es un sistema de gestión de bases de datos, desarrollado por Oracle Corporation, por lo tanto es propietario. Se considera a Oracle como uno de los sistemas de bases de datos más completos, destacando como cualidades más relevantes su soporte de transacciones, estabilidad, escalabilidad y Soporte multiplataforma. (Oracle, 2015).

Su dominio en el mercado de servidores empresariales ha sido casi total hasta hace poco, recientemente sufre la competencia del Microsoft SQL Server de Microsoft y de la oferta de otros sistemas manejadores de base de datos con licencia libre como PostgreSQL.

2.5.3 Microsoft SQL server

Microsoft SQL Server es un sistema de manejador de bases de datos relacionales (SGBD) basada en el lenguaje Transact-SQL, capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea. Provee al usuario un sistema estable y seguro que soporta transacciones procedimientos almacenados, y también incluye un potente entorno gráfico de administración, que permite el uso de comandos DDL y DML gráficamente. (Microsoft, 2015).

Adicionalmente, permite trabajar en modo cliente-servidor donde la información y datos se alojan en el servidor y las terminales o clientes de la red sólo acceden a la información, incluyendo la posibilidad de administrar información de otros servidores de datos.

2.5.4 MySQL

MySQL es la base de datos de código abierto más popular del mundo. Con su rendimiento comprobado, fiabilidad y facilidad de uso, MySQL se ha convertido en la

principal opción de base de datos para aplicaciones basadas en web. Desarrollada bajo licencia dual GPL/Licencia comercial por Oracle Corporation

Oracle impulsa la innovación de MySQL, brindando nuevas capacidades para impulsar las próximas generaciones de aplicaciones web, en la nube, móviles e integradas. (MySQL, 2018)

2.6 Bases de datos No SQL

Es una amplia clase de sistemas de gestión de bases de datos que difieren del modelo clásico de SGBDR (Sistema de Gestión de Bases de Datos Relacionales) en aspectos importantes, siendo el más destacado que no usan SQL como lenguaje principal de consultas. Los datos almacenados no requieren estructuras fijas como tablas, normalmente no soportan operaciones JOIN, ni garantizan completamente ACID (atomicidad, consistencia, aislamiento y durabilidad), y habitualmente escalan bien horizontalmente. Los sistemas NoSQL se denominan a veces "no sólo SQL" para subrayar el hecho de que también pueden soportar lenguajes de consulta de tipo SQL. (Wikipedia, 2018).

2.7 Tipos de Bases de datos NoSQL

2.7.1 Bases de datos clave-valor

Son el modelo de base de datos NoSQL más popular, además de ser la más sencilla en cuanto a funcionalidad. En este tipo de sistema, cada elemento está identificado por una llave única, lo que permite la recuperación de la información de forma muy rápida, información que habitualmente está almacenada como un objeto binario (BLOB). Se caracterizan por ser muy eficientes tanto para las lecturas como para las escrituras.

2.7.2 Bases de datos Documentales

Este tipo almacena la información como un documento, generalmente utilizando para ello una estructura simple como JSON o XML y donde se utiliza una clave única para cada registro. Este tipo de implementación permite, además de realizar búsquedas por clave-valor, realizar consultas más avanzadas sobre el contenido del documento.

Son las bases de datos NoSQL más versátiles. Se pueden utilizar en gran cantidad de proyectos, incluyendo muchos que tradicionalmente funcionan sobre bases de datos relacionales.

2.7.3 Bases de datos de Grafos

En este tipo de bases de datos, la información se representa como nodos de un grafo y sus relaciones con las aristas del mismo, de manera que se puede hacer uso de la teoría de grafos para recorrerla. Para sacar el máximo rendimiento a este tipo de bases de datos, su estructura debe estar totalmente normalizada, de forma que cada tabla tenga una sola columna y cada relación dos.

Este tipo de bases de datos ofrece una navegación más eficiente entre relaciones que en un modelo relacional.

2.8 Tipos de Bases de datos NoSQL

2.8.1 Bases de datos clave-valor

Son el modelo de base de datos NoSQL más popular, además de ser la más sencilla en cuanto a funcionalidad. En este tipo de sistema, cada elemento está identificado por una llave única, lo que permite la recuperación de la información de forma muy rápida, información que habitualmente está almacenada como un objeto binario (BLOB). Se caracterizan por ser muy eficientes tanto para las lecturas como para las escrituras.

2.8.2 Bases de datos Documentales

Este tipo almacena la información como un documento, generalmente utilizando para ello una estructura simple como JSON o XML y donde se utiliza una clave única para cada registro. Este tipo de implementación permite, además de realizar búsquedas por clave-valor, realizar consultas más avanzadas sobre el contenido del documento.

Son las bases de datos NoSQL más versátiles. Se pueden utilizar en gran cantidad de proyectos, incluyendo muchos que tradicionalmente funcionan sobre bases de datos relacionales.

2.8.3 Bases de datos de Grafos

En este tipo de bases de datos, la información se representa como nodos de un grafo y sus relaciones con las aristas del mismo, de manera que se puede hacer uso de la teoría

de grafos para recorrerla. Para sacar el máximo rendimiento a este tipo de bases de datos, su estructura debe estar totalmente normalizada, de forma que cada tabla tenga una sola columna y cada relación dos. Este tipo de bases de datos ofrece una navegación más eficiente entre relaciones que en un modelo relacional. ACENS (s,f)

2.9 Ejemplos de Bases de datos No SQL

2.9.1 Apache Cassandra

Es una base de datos NoSQL distribuida y basada en un modelo de almacenamiento de «clave-valor» y de columnas, de código abierto que está escrita en Java. Permite grandes volúmenes de datos en forma distribuida. Su objetivo principal es la escalabilidad lineal y la disponibilidad. La arquitectura distribuida de Cassandra está basada en una serie de nodos iguales que se comunican con un protocolo P2P con lo que la redundancia es máxima. Está desarrollada por Apache Software Foundation. (Apache Cassandra, 2018).

2.9.2 MongoDB

MongoDB es una base de datos NoSQL de documentos, de código abierto que ofrece alto rendimiento, alta disponibilidad y escalado automático.

Es una base de datos ágil que permite a los esquemas cambiar rápidamente cuando las aplicaciones evolucionan, proporcionando siempre la funcionalidad que los desarrolladores esperan de las bases de datos tradicionales, tales como índices secundarios, un lenguaje completo de búsquedas y consistencia estricta.

MongoDB ha sido creado para brindar escalabilidad, rendimiento y gran disponibilidad, escalando de una implantación de servidor único a grandes arquitecturas complejas de centros multidados. MongoDB brinda un elevado rendimiento, tanto para lectura como para escritura, potenciando la computación en memoria (in-memory). La replicación nativa de MongoDB y la tolerancia a fallos automática ofrece fiabilidad a nivel empresarial y flexibilidad operativa. (MongoDB, 2018)

2.9.3 Neo4j

Neo4j es una Base de datos de software libre orientada a grafos, implementado en Java y Scala. Neo4j implementa eficientemente las propiedades de un modelo de grafos hasta el nivel de almacenamiento. A diferencia del procesamiento de gráficos o las bibliotecas en memoria, Neo4j proporciona características completas de la base de datos, incluyendo cumplimiento de transacciones ACID, compatibilidad con clústeres y, o *failover* (conmutación por error) de tiempo de ejecución, lo que lo hace adecuado para utilizar datos de gráficos en escenarios de producción. (Ne4j, 2018)

2.10 Tecnologías de desarrollo Web

Desarrollo web es un término amplio que define la creación de páginas web y posteriormente subirlas al Internet. Para conseguirlo se hace uso de tecnologías de desarrollo del lado del servidor y del cliente que involucran una combinación de procesos de base de datos con el uso de un navegador en Internet a fin de realizar determinadas tareas o mostrar información.

Entre las diferentes tecnologías existentes, tenemos:

2.10.1 Lenguaje de Marcas de Hipertexto (HTML)

HTML, siglas de HyperText Markup Language (lenguaje de marcas de hipertexto), hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia para la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, entre otros. Es un estándar a cargo de la W3C, organización dedicada a la estandarización de casi todas las tecnologías ligadas a la web, sobre todo en lo referente a su escritura e interpretación. (Wikipedia, 2018)

2.10.2 Hojas de Estilo en Cascada (CSS)

CSS es un lenguaje de estilos empleado para definir la presentación, el formato y la apariencia de un documento de marcaje, sea html o cualquier otro. Comúnmente se emplea para dar formato visual a documentos html o xhtml que funcionan como espacios Web. También puede ser empleado en formatos xml, u otros tipos de documentos de marcaje para la posterior generación de documentos. (Collell Puig, s.f)

Las hojas de estilos nacen de la necesidad de diseñar la información de tal manera que podemos separar el contenido de la presentación y, así, por una misma fuente de información, generalmente definida mediante un lenguaje de marcaje, ofrecer diferentes presentaciones en función de dispositivos, servicios, contextos o aplicativos. Por lo que un mismo documento html, mediante diferentes hojas de estilo, puede ser presentado por pantalla, por impresora, por lectores de voz o por tabletas braille. Separamos el contenido de la forma, composición, colores y fuentes. (Collell Puig, s.f)

2.10.3 JavaScript

Es un lenguaje utilizado principalmente del lado del cliente, implementado como parte de un navegador web en donde la mayoría de la programación de este lenguaje está centrada en escribir funciones que respondan a eventos, tales como el movimiento del mouse, aperturas de ventanas, utilización de teclas, cargas de páginas, entre otros; lo cual permite mejoras en la interfaz de usuario agregando dinamismo a las páginas web.

El mismo fue diseñado con una sintaxis similar al lenguaje de programación C adoptando nombres y convenciones del lenguaje de programación Java. Sin embargo, Java y JavaScript no están relacionados ya que tienen semánticas y propósitos diferentes. (Wikipedia, 2018)

2.10.4 Ruby on Rails

Ruby on Rails es un entorno de desarrollo web de código abierto que está optimizado para la satisfacción de los programadores y para la productividad sostenible. Permite escribir un buen código evitando que se repita y favoreciendo la convención antes que la configuración.

También es un conjunto de librerías, automatismos y convenciones destinados a resolver los problemas más comunes a la hora de desarrollar una aplicación web, para que el programador pueda concentrarse en los aspectos únicos y diferenciales de su proyecto en lugar de en los problemas recurrentes.

Rails fue creado en 2003 por David Heinemeier Hansson y desde entonces ha sido extendido por el Rails core team, más de 2.100 colaboradores y soportado por una extensa y activa comunidad. (Ruby on Rails (s.f)).

2.10.5 Angular

Angular es una plataforma que facilita la creación de aplicaciones web. Angular combina las plantillas declarativas, inyección de dependencias, de extremo a extremo, herramientas y mejores prácticas integradas para resolver los retos del desarrollo. Angular permite a los desarrolladores crear aplicaciones que residen en la web, móvil o el escritorio. (Angular (s.f), 2018)

2.10.6 Python Flask

Flask es un microframework en python basado en la especificación WSGI de Werkzeug y el motor de templates Jinja2; te permite crear aplicaciones web rápidamente y con un mínimo número de líneas de código. Este framework nos va a permitir crear aplicaciones web rápidamente ya que incluye un servidor web de desarrollo para que podamos probar las aplicaciones desarrolladas sin tener que instalar Apache o Nginx. (Flask (s.f), 2018). Algunas de las características más importantes que presenta este framework son las siguientes:

- Propio servidor web
- Debbuging
- Fácil desarrollo
- Gran cantidad de documentación
- Compatibilidad con WSGI 1.0
- Soporte Integrado para hacer Unit Testing.

2.10.7 Node.js

Node.js es un entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor, basado en el lenguaje de programación ECMAScript, asíncrono, con I/O de datos en una arquitectura orientada a eventos y basado en el motor V8 de Google. Fue creado con el enfoque de ser útil en la creación de programas de red altamente escalables, como por ejemplo, servidores web. (Fundación Node.js (s.f), 2018)

2.10.8 Django

Django es un entorno de desarrollo web escrito en Python que fomenta el desarrollo rápido y el diseño limpio y pragmático. Django es un Framework web de código abierto

escrito en Python que permite construir aplicaciones web más rápido y con menos código.

Django fue inicialmente desarrollado para gestionar aplicaciones web de páginas orientadas a noticias de World Online, más tarde se liberó bajo licencia BSD. Django se centra en automatizar todo lo posible y se adhiere al principio DRY (Don't Repeat Yourself, no te repitas).

Django hace uso del paradigma de programación MTV (Model Template View, Modelo Plantilla Vista), donde hace uso del Modelo para todo lo que tenga que ver con el acceso a la base de datos, la Vista para la lógica de negocio y de la aplicación y la Plantilla como presentación al usuario.

Algunas características que ofrece son:

- Un mapeador objeto-relacional (ORM).
- Aplicaciones "enchufables" (pluggable) que pueden instalarse en cualquier página gestionada con Django.
- Una API de base de datos robusta.
- Un sistema incorporado de "vistas genéricas" que ahorra tener que escribir la lógica de ciertas tareas comunes.
- Un sistema extensible de plantillas basado en etiquetas, con herencia de plantillas.
- Un despachador de URLs basado en expresiones regulares.
- Un sistema "middleware" para desarrollar características adicionales; por ejemplo, la distribución principal de Django incluye componentes middleware que proporciona cache, compresión de la salida, normalización de URLs, protección CSRF y soporte de sesiones.
- Soporte de internacionalización, incluyendo traducciones incorporadas de la interfaz de administración.
- Documentación incorporada accesible a través de la aplicación administrativa (incluyendo documentación generada automáticamente de los modelos y las bibliotecas de plantillas añadidas por las aplicaciones). (Django (s.f), 2018)

2.10.9 CakePHP

CakePHP es un marco de desarrollo (Framework) rápido para PHP, libre, de código abierto. Se trata de una estructura que sirve de base a los programadores para que éstos puedan crear aplicaciones Web. Su principal objetivo es que se pueda trabajar de forma estructurada y rápida, sin pérdida de flexibilidad.

Con CakePHP el desarrollo web ya no es monótono porque ofrece las herramientas para que se empiece a escribir el código que realmente se necesita: la lógica específica de tu aplicación.

CakePHP tiene un equipo de desarrolladores y una comunidad activos, lo que añade valor al proyecto. Con CakePHP, además de no tener que reinventar la rueda, el núcleo de tu aplicación se mejora constantemente y está bien probado. (CakePHP (s.f), 2017).

Esta es una lista breve con las características que ofrece:

- Licencia flexible
- Compatible con PHP4 y PHP5
- CRUD integrado para la interacción con la base de datos
- Soporte de aplicación (scaffolding)
- Generación de código
- Arquitectura Modelo Vista Controlador (MVC)
- Despachador de peticiones (dispatcher), con URLs y rutas personalizadas y limpias
- Validación integrada
- Plantillas rápidas y flexibles (sintaxis de PHP, con ayudantes[helpers])
- Ayudantes para AJAX, Javascript, formularios HTML y más
- Componentes de Email, Cookie, Seguridad, Sesión y Manejo de solicitudes
- Listas de control de acceso flexibles
- Limpieza de datos
- Caché flexible
- Localización
- Funciona en cualquier subdirectorio del sitio web, con poca o ninguna configuración de Apache.

2.10.10 React

Es una biblioteca Javascript de código abierto para crear interfaces de usuario con el objetivo de animar al desarrollo de aplicaciones en una sola página.

React mantiene un virtual DOM propio, en lugar de confiar solamente en el DOM del navegador. Esto deja a la biblioteca determinar qué partes del DOM han cambiado comparando contenidos entre la versión nueva y la almacenada en el virtual DOM, y utilizando el resultado para determinar cómo actualizar eficientemente el DOM del navegador. (React(s.f), 2017)

2.10.11 Froala

Froala Editor es un ligero editor WYSIWYG HTML escrito en Javascript que permite capacidades de edición de texto enriquecido para sus aplicaciones. Su completa documentación, los complementos de framework especialmente diseñados y gran cantidad de ejemplos lo hacen fácil de integrar. (Froala (s.f)).

Características:

- Multiplataforma.
- Funcionalidad completa.
- Autoguardado.
- Retina lista.
- Alto rendimiento.

2.10.12 TinyMCE

Es un editor de texto WYSIWYG para HTML de código abierto que funciona completamente en JavaScript y se distribuye gratuitamente bajo licencia LGPL.

Al ser basado en JavaScript TinyMCE es independiente de la plataforma y se ejecuta en el navegador de internet. Tiene la habilidad de convertir un campo del tipo textarea u otros elementos de html en instancias del editor. Se puede integrar fácilmente a cualquier CMS. (TinyMCE s.f).

2.10.13 CKeditor

CKEditor es un editor de texto HTML/ WYSIWYG de código abierto que proporciona funciones de procesador de texto en páginas web, sin la necesidad de instalarlo como aplicación en la computadora del cliente. (CKEditor s.f).

Características

- Generación de código XHTML 1.0.
- Soporte CSS.
- Incorporar formularios.
- Formateo de Fuente.
- Cortar, copiar, pegar.
- Inserción de imágenes.
- Creación de tablas.
- Menús contextuales con botón derecho.
- Funcionalidad incorporada de corrector ortográfico.

2.10.14 Alfresco

Es una sistema de administración de contenidos de código fuente libre, desarrollado en Java, basado en estándares abiertos y de escala empresarial para sistemas operativos tipo Windows, Unix Solaris y algunas versiones de Linux. Alfresco (s.f), (2018)

Se distribuye en tres variantes diferentes:

- Alfresco Community Edition: Es software libre, con licencia LGPL de código abierto y estándares abiertos.
- Alfresco Enterprise Edition: Se distribuye bajo licencia de código abierto y estándares abiertos, con la posibilidad de soporte comercial y propietario a escala empresarial.
- Alfresco Cloud Edition (Alfresco in the cloud) que es la versión SaaS o Software como Servicio de Alfresco.

Características:

- Gestión de documentos

- Gestión de contenido web (incluyendo aplicaciones web y virtualización de sesiones)
- Gestión de registros
- Gestión de imágenes
- Servidores descentralizados
- Soporte multiplataforma (oficialmente Windows, GNU/Linux y Solaris).[59]

2.10.15 Nuxeo

Es un sistema de administración de contenidos libre, basado en estándares abiertos y de escala empresarial para Windows y sistemas operativos similares a Unix. Está diseñado para usuarios que requieren un alto grado de modularidad y rendimiento escalable. Nuxeo está desarrollado en Java. (Nuxeo (s.f), 2018).

Algunas de sus características son:

- Gestión de documentos
- Gestión de contenido web
- Versionado a nivel de repositorio
- Gestión de registros
- Gestión de imágenes
- Publicación integrada
- Flujo de trabajo basado en jBPM
- Búsquedas implementadas con el motor Lucene
- Servidores descentralizados

2.10.16 XML

XML es un Lenguaje de Etiquetado Extensible muy simple, pero estricto que juega un papel fundamental en el intercambio de una gran variedad de datos. Es un lenguaje muy similar a HTML pero su función principal es describir datos y no mostrarlos como es el caso de HTML. XML es un formato que permite la lectura de datos a través de diferentes aplicaciones.

Las tecnologías XML son un conjunto de módulos que ofrecen servicios útiles a las demandas más frecuentes por parte de los usuarios. XML sirve para estructurar, almacenar e intercambiar información.

Entre las tecnologías XML disponibles se pueden destacar:

XSL: Lenguaje Extensible de Hojas de Estilo, cuyo objetivo principal es mostrar cómo debería estar estructurado el contenido, cómo debería ser diseñado el contenido de origen y cómo debería ser paginado en un medio de presentación como puede ser una ventana de un navegador Web o un dispositivo móvil, o un conjunto de páginas de un catálogo, informe o libro.

XPath: Lenguaje de Rutas XML, es un lenguaje para acceder a partes de un documento XML.

XLink: Lenguaje de Enlace XML, es un lenguaje que permite insertar elementos en documentos XML para crear enlaces entre recursos XML.

XPointer: Lenguaje de Direccionamiento XML, es un lenguaje que permite el acceso a la estructura interna de un documento XML, esto es, a sus elementos, atributos y contenido.

XQL: Lenguaje de Consulta XML, es un lenguaje que facilita la extracción de datos desde documentos XML. Ofrece la posibilidad de realizar consultas flexibles para extraer datos de documentos XML en la Web. (W3C (s.f), 2018)

2.10.17 JSON

JSON (JavaScript Object Notation – Notación de Objetos de JavaScript) es un formato ligero de intercambio de datos. Está basado en un subconjunto del Lenguaje de Programación JavaScript, Standard ECMA-262 3rd Edition - Diciembre 1999. JSON es un formato de texto que es completamente independiente del lenguaje pero utiliza convenciones que son ampliamente conocidos por los programadores de la familia de lenguajes C, incluyendo C, C++, C#, Java, JavaScript, Perl, Python, y muchos otros. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos.

JSON está constituido por dos estructuras:

- Una colección de pares de nombre/valor. En varios lenguajes esto es conocido como un objeto, registro, estructura, diccionario, tabla hash, lista de claves o un arreglo asociativo.

- Una lista ordenada de valores. En la mayoría de los lenguajes, esto se implementa como arreglos, vectores, listas o secuencias.

Estas son estructuras universales; virtualmente todos los lenguajes de programación las soportan de una forma u otra. Es razonable que un formato de intercambio de datos que es independiente del lenguaje de programación se base en estas estructuras. (JSON (s.f), 2017).

2.10.18 Aplicación de página única

Aplicación de página única o single-page application (SPA), es una aplicación web o es un sitio web que cabe en una sola página con el propósito de dar una experiencia más fluida a los usuarios como una aplicación de escritorio. En un SPA todos los códigos de HTML, JavaScript, y CSS se carga de una vez o los recursos necesarios se cargan dinámicamente como lo requiera la página y se van agregando, normalmente como respuesta de las acciones del usuario. La página no tiene que cargar otra vez en ningún punto del proceso tampoco se transfiere a otra página, aunque las tecnologías modernas pueden permitir la navegabilidad en páginas lógicas dentro de la aplicación. La interacción con las aplicaciones de página única pueden involucrar comunicaciones dinámicas con el servidor web que está detrás. (Wikipedia, 2018)

2.10.19 Modelo - Vista – Controlador

El Modelo Vista Controlador (MVC), es un patrón de diseño muy utilizado para el desarrollo de aplicaciones Web, el cual tiene como idea fundamental separar los componentes principales de las aplicaciones, como lo son el sistema de gestión de base de datos, las interfaces de usuario y las tareas de manejar los eventos que se produzcan. (Olivares Carlos, 2012)

De acuerdo a Fernández Romero y Díaz González (2012) las partes del patrón MVC pueden definirse de la siguiente manera:

- El Modelo es el objeto que representa los datos del programa. Maneja los datos y controla todas sus transformaciones. El Modelo no tiene conocimiento específico de los Controladores o de las Vistas, ni siquiera contiene referencias a ellos. Es el propio sistema el que tiene encomendada la responsabilidad de mantener enlaces entre el Modelo y sus Vistas, y notificar a las Vistas cuando cambia el Modelo.

- La Vista es el objeto que maneja la presentación visual de los datos representados por el Modelo. Genera una representación visual del Modelo y muestra los datos al usuario. Interactúa preferentemente con el Controlador, pero es posible que trate directamente con el Modelo a través de una referencia al propio Modelo.
- El Controlador es el objeto que proporciona significado a las órdenes del usuario, actuando sobre los datos representados por el Modelo, centra toda la interacción entre la Vista y el Modelo. Cuando se realiza algún cambio, entra en acción, bien sea por cambios en la información del Modelo o por alteraciones de la Vista. Interactúa con el Modelo a través de una referencia al propio Modelo

2.10.20 Servicio Web

Alegsa (2016) indica que la W3C define "Servicio Web" como un sistema de software diseñado para permitir interoperabilidad máquina a máquina en una red. En general, suelen ser API's Web que pueden ser accedidas en una red, como Internet, y son ejecutadas en un sistema de alojamiento remoto. Los servicios Web permiten la comunicación entre diferentes máquinas, con diferentes plataformas y entre programas distintos. Esta comunicación se logra a través de la adopción de diversos estándares abiertos.

Los protocolos más comunes a la hora de implementar servicios Web son el protocolo SOAP y REST. El primero hace referencia al protocolo usado para la comunicación entre cliente y servidor donde se intercambian mensajes basados en XML (WSDL), mientras que en el segundo caso los mensajes son por lo general en formato JSON. En ambos casos tanto el cliente como el servidor deben conocer el formato y lenguaje de los mensajes para poder encapsular y desencapsular peticiones y respuestas. (Amaya, Arciniegas, Campo y Chanchi, 2011).

2.10.21 Protocolo SOAP

Es un protocolo para el intercambio de mensajes sobre redes de computadoras, generalmente usando HTTP, el cual está basado en XML lo cual facilita la lectura por parte de las personas, pero también los mensajes resultan más largos y considerablemente más lentos de transferir. Existen múltiples tipos de modelos de

mensajes en SOAP, pero el más utilizado es RPC, en donde un nodo de red (el cliente) envía un mensaje de solicitud a otro nodo (el servidor) y el servidor inmediatamente responde el mensaje al cliente.

Los mensajes SOAP son independientes del sistema operativo y pueden transportarse en varios protocolos de internet como SMTP, MIME y HTTP. Actualmente, la especificación SOAP es mantenida por el XML Protocol Working Group de la W3C.

Básicamente SOAP es un paradigma de mensajería de una dirección sin estado, que puede ser utilizado para formar protocolos más complejos según las necesidades de las aplicaciones que lo implementan. Puede formar y construir la capa base de una "pila de protocolos de web service", ofreciendo un framework de mensajería básica en el cual los web services se pueden construir. Este protocolo se conforma de tres partes:

- Sobre (envelope): el cual define qué hay en el mensaje y cómo procesarlo
- Conjunto de reglas de codificación para expresar instancias de tipos de datos
- La Convención para representar llamadas a procedimientos y respuestas.
- Adicionalmente, posee tres características principales:
- Extensibilidad. - Neutralidad (SOAP puede ser utilizado sobre cualquier protocolo de transporte como HTTP, SMTP, TCP o JMS).
- Independencia (SOAP permite cualquier modelo de programación). (Protocolo SOAP (s.f), 2017)

2.10.22 REST

De acuerdo a Amaya et al (2011), el protocolo REST (Representational State Transfer), plantea un estilo de arquitectura cliente-servidor en la cual un servicio es visto como un recurso y es identificado a través de una dirección URL, mediante la cual puede ser accedido o consumido. Para acceder a estos servicios Web, se hace uso de mensajes en formato simple, los cuales se intercambian entre el cliente y el servidor. REST define a partir de HTTP cuatro métodos: GET, PUT, DELETE y POST. Para la comunicación e intercambio de información entre cliente y servidor a través de REST, se puede hacer uso de diversos formatos y lenguajes: XML, HTML, JSON; aunque debido a la sencillez de los mensajes JSON, este es el tipo de mensajes más difundido en diferentes servicios propios de redes sociales como Facebook y Twitter, y comunidades en Internet. Cada mensaje intercambiado contiene la información necesaria para el

funcionamiento adecuado del servicio, de tal forma que para cada servicio el cliente y servidor conocen el formato o protocolo interno de los mensajes.

Hoy en día en la actualidad se usa en el sentido más amplio para describir cualquier interfaz web simple que utiliza XML y HTTP, sin las abstracciones adicionales de los protocolos basados en patrones de intercambio de mensajes como el protocolo de servicios web SOAP. Los sistemas que siguen los principios REST se llaman con frecuencia RESTful. REST afirma que la web ha disfrutado de escalabilidad como resultado de una serie de diseños fundamentales clave:

Un protocolo cliente/servidor sin estado: cada mensaje HTTP contiene toda la información necesaria para comprender la petición. Como resultado, ni el cliente ni el servidor necesitan recordar ningún estado de las comunicaciones entre mensajes. Sin embargo, en la práctica, muchas aplicaciones basadas en HTTP utilizan cookies y otros mecanismos para mantener el estado de la sesión (algunas de estas prácticas, como la reescritura de URLs, no son permitidas por REST).

Un conjunto de operaciones bien definidas que se aplican a todos los recursos de información: HTTP en sí define un conjunto pequeño de operaciones, las más importantes son POST, GET, PUT y DELETE. Con frecuencia estas operaciones se equiparan a las operaciones en bases de datos (Actualizar, Borrar, Modificación y Consulta) que se requieren para la persistencia de datos, aunque POST no encaja exactamente en este esquema.

Una sintaxis universal para identificar los recursos: En un sistema REST, cada recurso es direccionable únicamente a través de su URI.

El uso de hipertextos, tanto para la información de la aplicación como para las transiciones de estado de la aplicación: la representación de este estado en un sistema REST son típicamente HTML o XML. Como resultado de esto, es posible navegar de un recurso REST a muchos otros, simplemente siguiendo enlaces sin requerir el uso de registros u otra infraestructura adicional. (Wikipedia, 2018)

2.11 Contrato

- Convenio en virtud del cual se produce o se transfiere una obligación o un derecho. (Rafael de Pina Vara, 2006).

- El acuerdo entre dos o más personas para crear o transmitir derechos y obligaciones. El convenio en sentido especial es el acuerdo que modifica o extingue derechos y obligaciones. (Manuel Borja Soriano, 1962)

2.11.1 Arrendador

El propietario de un bien inmueble que, a través de un contrato de arrendamiento o alquiler, promete arrendar la totalidad o una porción del inmueble a otra persona (un "arrendatario") para su uso exclusivo, generalmente, por un determinado período de tiempo y a cambio de un monto de dinero acordado. (Abogado (s,f), 2018)

2.11.2 Arrendatario

Se cataloga como arrendatario a aquel individuo que toma una determinada cosa o servicio en arrendamiento. (Definicion.de (s,f), 2018)

2.11.3 Contrato de arrendamiento

Es un contrato por el cual una de las partes, llamada arrendador, se obliga a transferir temporalmente el uso y goce de un bien mueble o inmueble a otra parte denominada arrendatario, quien a su vez se obliga a pagar por ese uso o goce un precio cierto y determinado. (Antonio López Álvarez, Alfonso Ortega Giménez, 2012)

2.11.4 Clases de arrendamientos

Arrendamiento de cosas: Crea un vínculo personal, por virtud del cual puede exigir el arrendatario, el uso y disfrute de aquellas, en tanto pesa sobre éste la obligación de pagar la merced convenida.

Prestación de servicios: El arrendatario se obliga a trabajar o a prestar determinados servicios al arrendante en forma, lugar y tiempo convenidos mediante un pago. El arrendatario está obligado a retribuir los servicios. Este tipo de contrato concluye por incumplimiento de obligaciones, por terminación de contrato o por la muerte.

Arrendamiento de obra o locación de obras: Una persona se compromete con otra a realizar una obra o un trabajo determinado mediante el pago de un precio. Esto recae sobre el resultado de un trabajo, sobre el producto del mismo, ya acabado. Ejemplo: la confección de un traje o la construcción de una casa. (Wikipedia, 2018)

2.11.5 Contrato de renovación

Hace referencia a realizar de nuevo un contrato o realizarle modificaciones al que ya se venía manejando, es decir comenzar a manejar nuevos términos y condiciones en un contrato que ya se venía desarrollando con anterioridad. (Mis abogados (s,f))

2.11.6 Bases legales del contrato de arrendamiento de locales comerciales.

- El código Civil Venezolano Vigente el cual establece en que consiste un contrato de arrendamiento.
- Ley de Regulación del Arrendamiento Inmobiliario para el Uso Comercial.

Gaceta 40.418: del 23 de Mayo del 2014 a través del cual se publica el Decreto N° 929 que decreta el Rango, Valor y Fuerza de la Ley de Regulación de Arrendamiento Inmobiliario para Uso Comercial

Ley especial que regula la materia de arrendamiento de locales comerciales

- **Objeto**

El Decreto, tiene por objeto regir las condiciones y procedimientos para regular y controlar la relación entre arrendadores y arrendatarios, para el arrendamiento de inmuebles destinados al uso comercial (artículo 1).

- **Ámbito Subjetivo**

A los fines de su aplicación, se entenderá por “inmuebles destinados a uso comercial”, aquellos en los cuales se desempeñen actividades comerciales o de prestación de servicio como parte del giro ordinario del establecimiento que allí funciona, independientemente de que dicho inmueble constituya una unidad inmobiliaria por sí solo, forme parte de un inmueble de mayor magnitud, o se encuentre anexado a este.

Constituirán inmuebles destinados a uso comercial, salvo prueba en contrario, los locales ubicados en centros comerciales, en edificaciones de viviendas u oficinas , o en edificaciones con fines turísticos, de uso médico asistencial distintos a consultorios, laboratorios o quirófanos, o educacional , así como los que forman parte, sin ser solo depósitos, de un galpón o estacionamiento, los quioscos, stands, y

establecimientos similares, aun cuando éstos no se encuentren unidos de manera permanente al inmueble donde funcionan o se ubiquen en áreas de dominio público

Capítulo 3

Marco Metodológico

Para desarrollar un software de calidad uno de los primeros pasos a seguir es conocer el software a realizar y todo lo que se debe satisfacer para cumplir el objetivo, para así lograr un desarrollo satisfactorio, donde una de las decisiones más importantes es el de escoger a la metodología apropiada para un equipo en un determinado proyecto.

3.1 Scrum

Scrum es un proceso ágil y liviano que sirve para administrar y controlar el desarrollo de software. El desarrollo se realiza en forma iterativa e incremental (una iteración es un ciclo corto de construcción repetitivo). Cada ciclo o iteración termina con una pieza de software ejecutable que incorpora nueva funcionalidad. Las iteraciones en general tienen una duración entre 2 y 4 semanas. Scrum se utiliza como marco para otras prácticas de ingeniería de software como RUP o Extreme Programming.

Se focaliza en priorizar el trabajo en función del valor que tenga para el negocio, maximizando la utilidad de lo que se construye y el retorno de inversión. Está diseñado especialmente para adaptarse a los cambios en los requerimientos, por ejemplo en un mercado de alta competitividad. Los requerimientos y las prioridades se revisan y ajustan durante el proyecto en intervalos muy cortos y regulares. De esta forma se puede adaptar en tiempo real el producto que se está construyendo a las necesidades del cliente. Se busca entregar software que realmente resuelva las necesidades, aumentando la satisfacción del cliente.

Scrum tiene un conjunto de reglas muy pequeño y muy simple y está basado en los principios de inspección continua, adaptación, auto-gestión e innovación. El cliente se entusiasma y se compromete con el proyecto dado que ve crecer el producto iteración a iteración y encuentra las herramientas para alinear el desarrollo con los objetivos de negocio de su empresa. Por otro lado, los desarrolladores encuentran un ámbito propicio

para desarrollar sus capacidades profesionales y esto resulta en un incremento en la motivación de los integrantes del equipo. Oliver A. Perez (2011).

3.1.1 Características

- Equipos autodirigidos.
- Utiliza reglas para crear un entorno ágil de administración de proyectos.
- No prescribe prácticas específicas de ingeniería.
- Los requerimientos se capturan como ítems de la lista Product Backlog.
- El producto se construye en una serie de Sprints de un mes de duración.

3.1.2 Fases

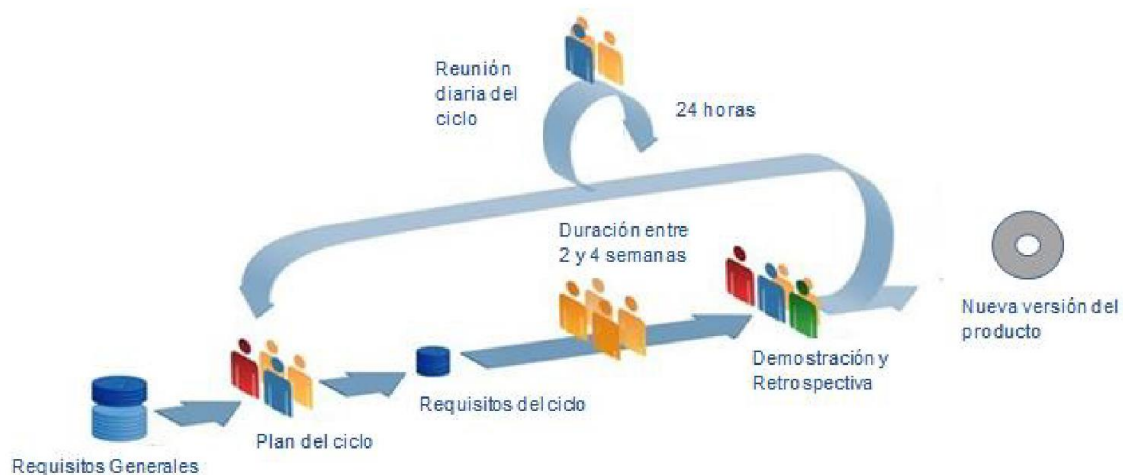


Figura 1. Proceso SCRUM. (Tomada de Softeng, 2017)

Revisión de planes de Release: Corresponde a la “planificación del Sprint”. Ésta fase se ejecuta una vez establecida la pila de producto y es llevada a cabo por el equipo a fin de evaluar las diferentes factibilidades de los requerimientos y estimaciones, basándose en la funcionalidades.

Distribución, revisión y ajustes de estándares de producto: Corresponde a la “Pila de Sprint”. En ésta fase los desarrolladores realizan los ajustes de los estándares y requerimientos mínimos, dejando todo listo para comenzar con la fase de Sprint.

Sprint: Ésta Fase de aproximadamente 30 días es donde se efectúa el desarrollo del software y se llevan a cabo las reuniones, consta de las siguientes subfases: elaborar, integrar, revisar y ajustar.

Revisión del Sprint: Corresponde al “incremento”. En ésta fase se revisa el Sprint y si es necesario se añaden nuevos ítems a la pila de producto. Éste proceso se repite hasta que el producto esté listo para la fase de cierre.

Cierre: En ésta fase se da lugar a la depuración y correcciones de errores, éste procedimiento se repite hasta alcanzar la calidad en el producto. Posterior a las correcciones y pruebas se realiza el Marketing y promoción del producto y al terminar ésta fase el proyecto queda cerrado.

3.2 Metodología utilizada para el desarrollo

El desarrollo de la aplicación Web se realizará bajo la metodología Scrum, ya que está orientada a la productividad, posee gran capacidad de adaptación, permite desarrollar soluciones con estándares de calidad, y gracias a la comunicación permanente con el cliente se pueden realizar los ajustes necesarios en cada etapa del proyecto para garantizar que el producto final cumpla los requerimientos establecidos. Para el desarrollo de la aplicación, se realizarán cuatro iteraciones (Sprints), donde cada una tendrá una duración de cuatro semanas aproximadamente.

Siguiendo los lineamientos de la metodología se definieron tres roles básicos:

Product Owner: son los beneficiarios y las personas interesadas en el desarrollo de la solución, en este caso, serían las empresas inmobiliarias que se encuentran involucradas en el proyecto.

Team: Personas encargadas del desarrollo de la aplicación, en este caso la Bachiller y tesista Nebraska Estephfany Meléndez Azócar.

Scrum Master: Persona encargada de liderar el equipo de desarrollo, este rol le correspondería al profesor Franky Uzcategui, tutor del TEG.

Capítulo 4

Marco Aplicativo

A continuación, para el siguiente capítulo se presenta una descripción general de la solución en conjunto con las respectivas tecnologías utilizadas para su elaboración. Además se detalla la metodología de desarrollo SCRUM que se aplicó para la elaboración de este proyecto, así como los diferentes requerimientos, perfiles de usuarios que interactúan con el sistema, explicación de las interfaces de la aplicación y las pruebas realizadas que contribuyeron a un mejor desempeño y una mayor aceptación de la aplicación.

4.1 Descripción general de la solución

Se propone el uso de tecnologías web, para el desarrollo de una solución que tiene como finalidad la automatización de creación de contratos inmobiliarios, que les permita a los usuarios ahorrar tiempo y facilitar la obtención de los datos y la creación de contratos.

Realizando las solicitudes de los datos mediante un Servicio Web que les permita intercambiar información con un servidor web que le proporcione los datos requeridos para el inicio de la creación del contrato. De tal manera que el usuario no requiere cargar la información, solo consultarla, con la finalidad de minimizar los tiempos de creación evitando generación de retrasos para el cliente.

Por esa razón, se propone la siguiente arquitectura:

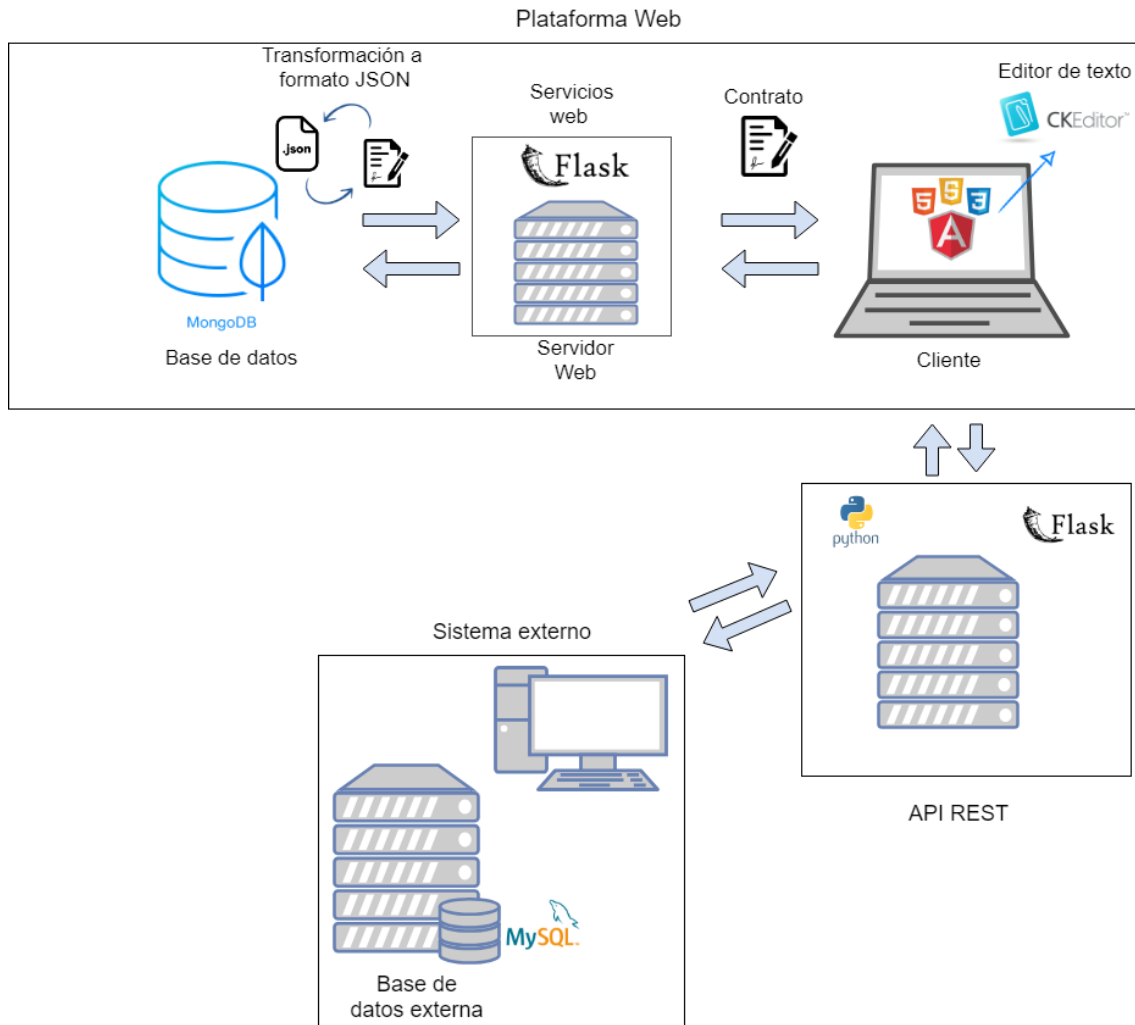


Figura 2. Arquitectura propuesta Fuente: La Autora.

En esta arquitectura se mantienen los principios establecidos por la arquitectura base de un sistema web, dentro de esta, se proponen la utilización de las siguientes herramientas para el desarrollo de cada una de las áreas del trabajo:

Para el desarrollo al área correspondiente al cliente, se utilizó el framework Angular en su versión 4, el cual permitió desarrollar una aplicación web de una sola página, con el propósito de dar una experiencia más fluida al usuario. En esta aplicación el usuario podrá crear y editar contratos inmobiliarios, utilizando CKEditor un editor de texto para HTML que estará integrado en el sistema para brindar funcionalidades de edición de documentos. Una vez finalizado el proceso de edición de contrato el mismo es enviado en formato json a través de un servicio para posteriormente ser almacenado en una base de datos.

Para el desarrollo de la capa de servicio web, se propone el uso del micro framework Flask, que nos permitió construir los servicios para la comunicación del cliente con la base de datos. Este framework minimalista, permite el uso del patrón modelo vista controlador (MVC) que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones. La base de datos asociada a la aplicación web será implementada utilizando MongoDB, debido a que la misma está orientada a documentos además de ser fácilmente escalable. Una vez almacenados los contratos, se procede a generar un contrato en formato pdf.

4.2 Aplicación de la metodología Scrum

A continuación, en la siguiente Tabla se muestran las actividades programadas para los Sprint en pro del desarrollo del sistema.

4.2.1 Lista de Objetivos

#	Sprint	Actividad	Duración
1	Analizar requerimientos del proceso de elaboración de contratos inmobiliarios.	• Definición de los requerimientos del negocio.	2 Semanas
		• Instalación de las herramientas de desarrollo necesarias.	1 Semana
		• Configuración de los ambientes de desarrollo.	1 Semana
2	Diseñar los artefactos y vistas de la aplicación web.	• Creación de prototipos de la aplicación web.	1 Semana
		• Maquetado de las interfaces.	2 Semanas
		• Elaboración del modelo de datos de la base de datos.	1 Semana

Tabla 1. Pila del producto. Fuente: Elaborada por la autora.

#	Sprint	Actividad	Duración
3	Desarrollar módulo para la creación, edición de contratos y servicios web.	• Desarrollo del módulo de creación de contratos. • Desarrollo de la funcionalidad de autenticación. • Desarrollo de la funcionalidad de obtención de datos de los participantes del contrato. • Desarrollo de la funcionalidad de crear contratos, crear plantillas de contrato y cláusulas de contrato. • Desarrollo de la funcionalidad de guardar contratos.	2 Semanas
		• Desarrollo del módulo de edición de contratos. • Desarrollo de la funcionalidad de listar contratos, plantillas de contrato y cláusulas. • Desarrollo de la funcionalidad de editar contratos y editar plantillas de contrato. • Desarrollo de la funcionalidad de eliminar contratos y plantillas de contratos. • Desarrollo de la funcionalidad de generar el contrato en formato pdf.	2 Semanas
4	Realizar pruebas funcionales, y de aceptación de usuarios.	• Realización de pruebas funcionales. • Realización de pruebas de calidad.	4 Semanas

Tabla 2. Pila del producto (Continuación). Fuente: Elaborado por la autora.

4.2.2 Lista de actividades de la iteración (Pila de Sprint)

- **Sprint 1:** Se inició con el análisis y definición de los requerimientos del negocio, una vez definido el alcance se procedió a la elección de las herramientas necesarias para el desarrollo de la aplicación y posteriormente su instalación, las cuales fueron Python Flask 3.6.5, Angular 4, CKeditor 4 y MongoDB 3.6.4. Finalmente se instalaron todas las dependencias necesarias para la realización del sistema.
- **Sprint 2:** En este Sprint se realizó el diseño y creación de prototipos de la aplicación web. También se realizó el maquetado de la aplicación a partir de los componentes provistos por el Framework Angular 4 como base para el desarrollo de interfaces de usuario. Luego se procedió a la creación del modelo de datos para la base de datos.
- **Sprint 3:** En el siguiente Sprint se desarrollaron las funcionalidades de los módulos de creación y edición de contratos, con sus respectivas validaciones para el correcto funcionamiento del sistema. En los cuales los módulos se dividieron en las siguientes actividades:
 - **Módulo de creación de contratos:** En el cual se desarrollaron las funcionalidades de autenticación de usuarios, creación de contratos, creación de plantillas de contratos y creación de cláusulas de contratos. Además del desarrollo de la funcionalidad de almacenamiento de contratos, plantillas de contratos y cláusulas de contrato.
 - **Módulo de edición de contratos:** Se desarrollaron las funcionalidades de listar los contratos, plantillas de contratos y cláusulas existentes. También se desarrolló la funcionalidad de editar contratos y plantillas de contratos, así como la funcionalidad de eliminar contratos y plantillas de contratos y la funcionalidad de generar contratos en formato pdf.
- **Sprint 4:** Se realizaron las pruebas funcionales necesarias para validar el correcto funcionamiento de las funcionalidades desarrolladas. Así como la

realización de pruebas no funcionales como lo son disponibilidad, accesibilidad, usabilidad, seguridad y rendimiento y de calidad al sistema desarrollado.

4.3 Requerimientos del sistema

De acuerdo a los requerimientos definidos, se establecieron las siguientes clasificaciones:

4.3.1 Requerimientos funcionales

- Autenticación de usuarios.
- Consulta de los participantes del contrato.
- Creación de plantilla de contrato.
- Creación de contrato.
- Creación de cláusulas de contrato.
- Consulta de plantillas creadas.
- Consulta de contratos creados.
- Consulta de cláusulas creadas.
- Edición de contrato.
- Edición de plantilla de contrato.
- Eliminar contrato.
- Eliminar plantilla.
- Eliminar cláusula.
- Almacenar contrato.
- Almacenar plantilla de contrato.
- Generar contrato en formato pdf.

4.3.2 Requerimientos no funcionales

- Validar las entradas de los datos para el correcto funcionamiento del sistema.
- Usar tecnologías multiplataforma para el desarrollo del sistema.
- Interfaces intuitivas y de fácil uso.

4.4 Descripción del flujo asociado a la solución

Para el desarrollo de este trabajo especial de grado, es importante detallar dos de los procesos principales que se cumplen dentro de la aplicación y que dan forma lo que se podría definir como el flujo de la solución.

A continuación se detallan los procesos de: creación de plantillas y creación de contratos.

4.4.1 Proceso de creación de plantillas

En Figura 3 se describe el proceso de creación de plantillas de contratos, el cual consiste en que el usuario, una vez que se ha autenticado exitosamente en la aplicación por primera vez, su primera acción para poder crear un contrato es crear una plantilla de contrato, que posteriormente será utilizada para la creación de un contrato, una vez creada la plantilla el usuario procede a escribir o ingresar contenido para armar esta plantilla, posteriormente el usuario puede o no, agregar una cláusula ya previamente creada o también podría crear una nueva cláusula. Cuando el usuario considere que la plantilla está finalizada, procede a guardarla y esta será almacenada en formato Json en la base de datos MongoDB, dando por finalizado el proceso.

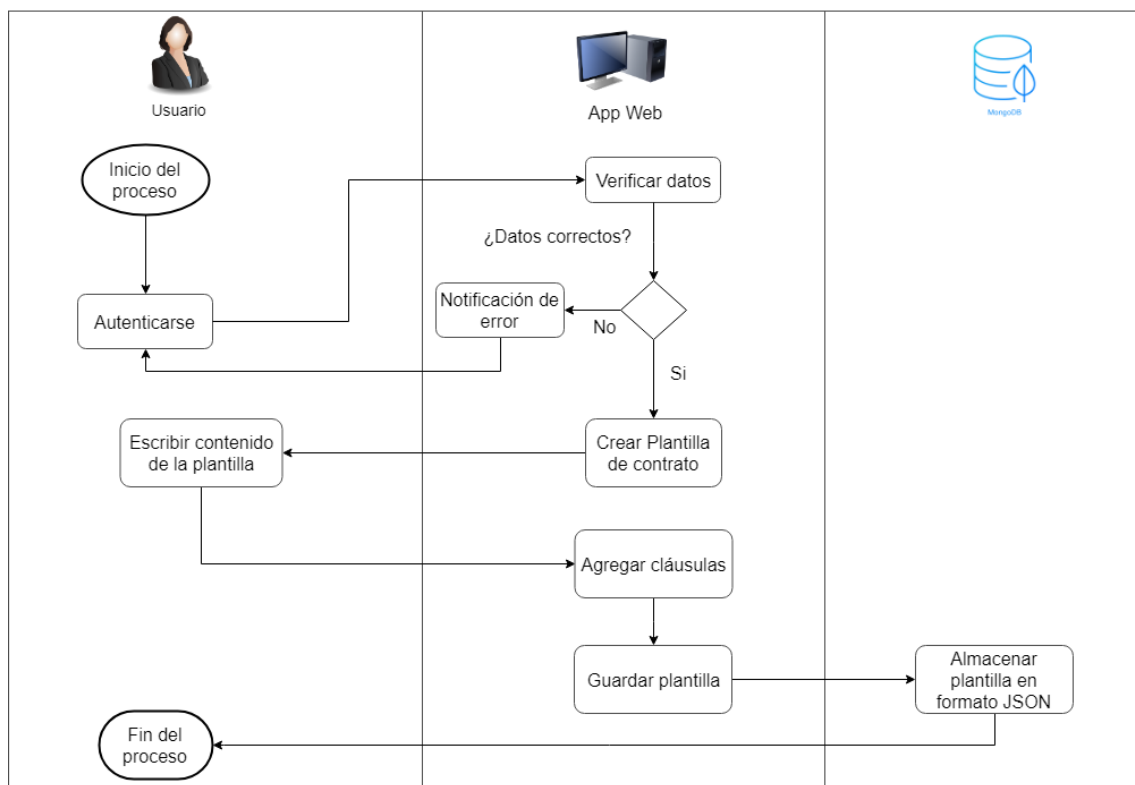


Figura 3. Proceso de creación de plantillas de contratos. Fuente: Elaborado por la autora.

4.4.2 Proceso de creación de contratos

En Figura 4 se describe el proceso de creación de contratos, el cual consiste en seleccionar una plantilla de contrato ya creada, para a partir de esa plantilla proceder a crear un contrato. Una vez seleccionada la plantilla, se deberá seleccionar los participantes del contrato, que han sido cargados previamente en el sistema. Después de seleccionar los participantes, el contrato procederá a cargar con los datos de los participantes seleccionados, de igual forma se podrá modificar el contrato, agregar cláusulas si así se desea, para luego proceder a guardar el contrato el cual será almacenado en la base de datos MongoDB en formato json, además en la creación de contrato se tiene la opción de finalizar contrato, para así proceder a generar el contrato en formato pdf, dando por finalizado el proceso.

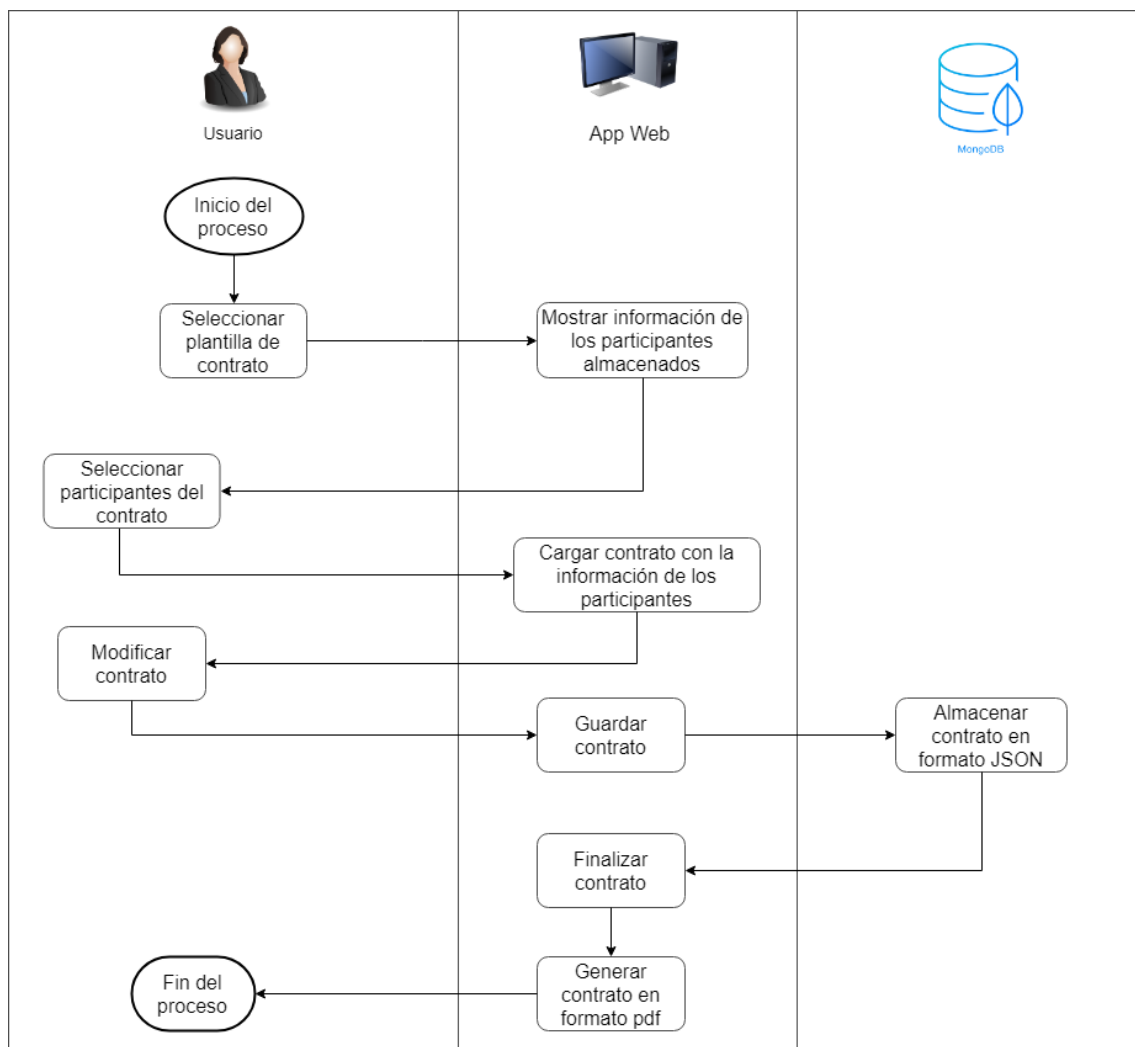


Figura 4. Proceso de creación de contratos. Fuente: Elaborado por la autora.

4.5 Análisis del modelo de datos y definición

En la elaboración del modelo de datos pertenecientes a la aplicación se crearon un total de cuatro (4) colecciones.

4.5.1 Listado de colecciones de la aplicación

A continuación se presenta un listado de las colecciones de la base de datos perteneciente al sistema desarrollado, junto con una breve descripción de cada una:

- **Users (Usuarios):** Permite almacenar los usuarios en el sistema.
- **Contract (Contrato):** Permite almacenar la información de los contratos, como lo son: título y contenido, así como también la información de los participantes de un contrato: arrendador, arrendatario e inmueble, los cuales se modelaron como documentos embebidos dentro de esta colección.
- **Clausule (Cláusula):** Permite almacenar la información de las cláusulas, como lo son: título, descripción y contenido.
- **Template (Plantilla):** Permite almacenar la información de las plantillas de contratos, como lo son: título y contenido.

4.5.2 Modelo de datos

- **Colección: User (Usuario)**

```
{
  username : "NOMBRE DE USUARIO",
  password : "CONTRASEÑA"
}
```

- **Colección: Contract (Contrato)**

```
{
  name : "TITULO DEL CONTRATO",
  content : "CONTENIDO DEL CONTRATO",
  participantes: [
    {
```

```

    type : "ARRENDADOR",
    subType: "NATURAL",
    nombre_arrendador: "NOMBRE DEL ARRENDADOR",
    ci_arrendador: "CEDULA DEL ARRENDADOR",
    estado_civil_arrendador: "ESTADO CIVIL",
    domicilio_arrendador: "DOMICILIO",
    registro_mercantil_arrendador: "REGISTRO
MERCANTIL"
  },
  {
    type : "ARRENDADOR",
    subType: "JURIDICO",
    nombre_empresa_arrendador: "NOMBRE EMPRESA",
    rif_empresa_arrendador: "RIF EMPRESA",
    registro_mercantil_arrendador_juridico: "REGISTRO
MERCANTIL",
    domicilio_arrendador_juridico: "DOMICILIO",
    nombre_representante_juridico: "NOMBRE
REPRESENTANTE JURIDICO",
    ci_representante_juridico: "CEDULA REPRESENTANTE
JURIDICO"
  },
  {
    type : "ARRENDATARIO",
    nombre_empresa_arrendatario: "NOMBRE EMPRESA",
    rif_empresa: "RIF EMPRESA",
    registro_mercantil_arrendatario: "REGISTRO
MERCANTIL",
    ubicacion: "UBICACION",

```

```

        nombre_representante: "NOMBRE REPRESENTANTE
        JURIDICO",

        ci_representante: "CEDULA REPRESENTANTE
        JURIDICO"

    }],

    inmueble: {

        name : "NOMBRE INMUEBLE",

        local_Number: "NUMERO O IDENTIFICADOR DEL LOCAL",

        dimension : "DIMENSIONES",

        address : "DIRECCION"

    }

}

```

- **Colección: Template (Plantilla)**

```

{

    name : "TITULO DE PLANTILLA",

    content : "CONTENIDO DE PLANTILLA"

}

```

- **Colección: Clausules (Clausulas)**

```

{

    title : "TITULO DE CLAUSULA",

    description : "DESCRIPCION DE LA CLAUSULA",

    content : "CONTENIDO DE LA CLAUSULA"

}

```

4.6 Servicios web

El sistema web es alimentado a través de servicios web, los cuales fueron desarrollados siguiendo el enfoque REST. A continuación se describen los servicios desarrollados para el sistema:

1. **Nombre:** Autenticación.

1.1. **Método HTTP:** POST

1.1.1 **Descripción:** Servicio que permite al usuario autenticarse o abrir sesión en el sistema.

1.1.2 **Entrada:** Nombre de usuario y contraseña.

1.1.3 **Salida:** “Token” de autenticación e información asociada al usuario.

2. **Nombre:** Contrato.

2.1. **Método HTTP:** GET

2.1.1 **Descripción:** Servicio que permite obtener un listado de los contratos existentes en el sistema.

2.1.2 **Entrada:** Usuario autenticado.

2.1.3 **Salida:** Listado de contratos.

2.2 **Método HTTP:** POST

2.2.1 **Descripción:** Servicio que permite registrar un contrato en el sistema.

2.2.2 **Entrada:** Título del contrato, contenido del contrato, el cual debe contener los participantes de dicho contrato.

2.2.3 **Salida:** Notificación de éxito o de error.

2.3 **Método HTTP:** PUT

2.3.1 **Descripción:** Servicio que permite editar o modificar un contrato.

2.3.2 **Entrada:** Contrato, ID del contrato que se desea editar.

2.3.3 **Salida:** Notificación de éxito o de error.

2.4 **Método HTTP:** DELETE

2.4.1 **Descripción:** Servicio que permite borrar un contrato.

2.4.2 **Entrada:** Contrato, ID del contrato que se desea borrar.

2.4.3 **Salida:** Notificación de éxito o de error.

3. **Nombre:** Plantilla de contrato.

3.1. **Método HTTP:** GET

3.1.1 **Descripción:** Servicio que permite obtener un listado de las plantillas existentes en el sistema.

3.1.2 **Entrada:** Usuario autenticado.

3.1.3 **Salida:** Listado de plantillas de contratos.

3.2 **Método HTTP:** POST

3.2.1 **Descripción:** Servicio que permite registrar una plantilla de contrato en el sistema.

3.2.2 **Entrada:** Título de la plantilla, contenido de la plantilla.

3.2.3 **Salida:** Notificación de éxito o de error.

3.3 **Método HTTP:** PUT

3.3.1 **Descripción:** Servicio que permite editar o modificar una plantilla de contrato.

3.3.2 **Entrada:** Plantilla, ID de la plantilla de contrato que se desea editar.

3.3.3 **Salida:** Notificación de éxito o de error.

3.4 **Método HTTP:** DELETE

3.4.1 **Descripción:** Servicio que permite borrar una plantilla de contrato.

3.4.2 **Entrada:** Plantilla, ID de la plantilla de contrato que se desea borrar.

3.4.3 **Salida:** Notificación de éxito o de error.

4. **Nombre:** Cláusulas.

4.1. **Método HTTP:** GET

4.1.1 **Descripción:** Servicio que permite obtener un listado de las cláusulas existentes en el sistema.

4.1.2 **Entrada:** Usuario autenticado.

4.1.3 **Salida:** Listado de las cláusulas existentes en el sistema.

4.2 **Método HTTP:** POST

4.2.1 **Descripción:** Servicio que permite registrar una cláusula en el sistema.

4.2.2 **Entrada:** Título de la cláusula, contenido de la cláusula y descripción.

4.2.3 **Salida:** Notificación de éxito o de error.

4.3 **Método HTTP:** DELETE

4.3.1 **Descripción:** Servicio que permite borrar una cláusula.

4.3.2 **Entrada:** Cláusula, ID de la cláusula que se desea borrar.

4.3.3 **Salida:** Notificación de éxito o de error.

4.7 Descripción de los módulos del sistema y sus interfaces

4.7.1 Descripción de los módulos

El sistema web se encuentra dividido en dos módulos que se describen a continuación y se presentan en la figura:

- **Módulo de creación de contratos:** Este módulo está dedicado a la creación de contratos inmobiliarios, a través de este el usuario podrá crear contratos en base a plantillas de contratos que también podrán ser creadas a través del sistema, así como también agregar cláusulas previamente creadas a través del sistema esto facilitara la creación de contratos y permitirá además seleccionar los participantes ya existentes en el sistema.

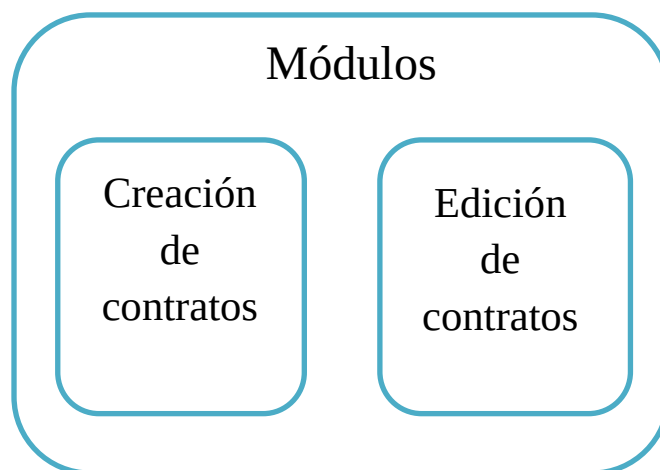


Figura 5. Módulos del sistema web. Fuente: Elaborado por la autora.

- **Módulo de edición de contratos:** A través de este módulo se podrá realizar la edición de un contrato, además de también permitir la edición de plantillas de contratos. También se permitirá eliminar un contrato, plantilla o cláusula.

4.7.2 Interfaces del sistema web

- **Interfaz correspondiente al inicio de sesión:** En la figura 4 se presenta la interfaz asociada al inicio de sesión en el sistema. En esta vista el usuario introduce su nombre de usuario y contraseña para realizar la autenticación.

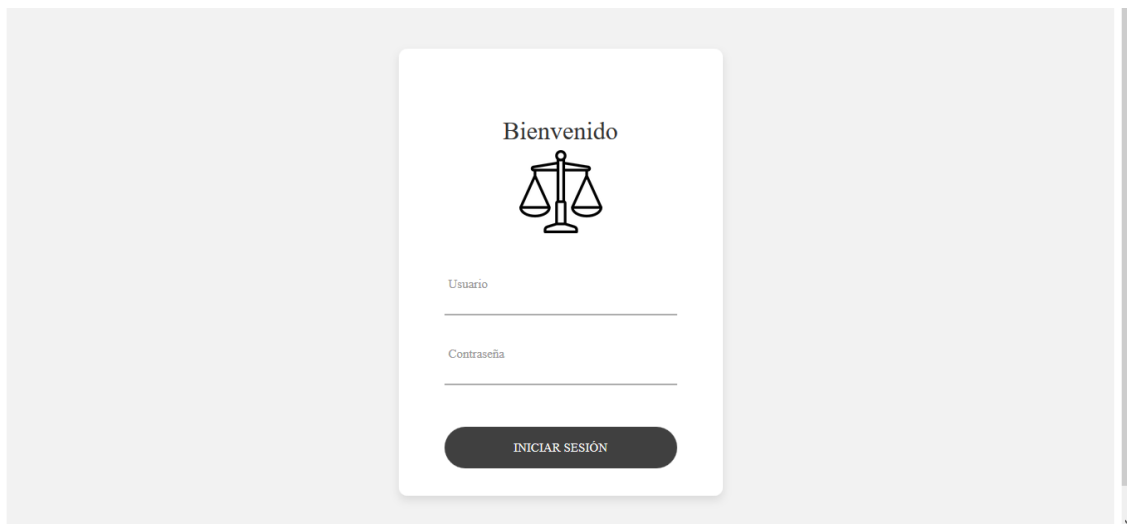


Figura 6. *Interfaz Inicio de Sesión. Fuente: Captura de pantalla.*

- **Interfaz inicial de Creación de contratos:** En la figura 5 se presenta la primera interfaz que se visualiza al iniciar sesión en el sistema. Esta muestra los contratos y plantillas de contratos ya creados, y los contratos finalizados. También muestra en la parte izquierda la opción de “Crear nueva plantilla de contrato”, opción utilizada para la creación de nuevas plantillas de contratos.



Figura 7. *Interfaz Inicial creación de contratos. Fuente: Captura de pantalla*

- **Interfaz (modal) correspondiente a la creación de plantilla de contrato:** En la figura 6 se presenta la interfaz utilizada para llevar a cabo la creación de plantillas de contratos, introduciendo los datos solicitados.

The image shows a web application interface with a dark blue header containing a scales of justice icon. A modal window titled 'INTRODUZCA LOS DATOS DE LA PLANTILLA' is open. On the left side of the modal, there is a sidebar with the text 'CREAR CONTR...' and a large plus sign, with the text 'CREAR NUEVA PL...' and 'DE CONTRA' below it. The main area of the modal contains three input fields: 'Nombre de la plantilla' with a placeholder 'Titulo...', 'Descripción' with a placeholder 'Descripción...', and 'Tipo de contrato.' which is a dropdown menu. At the bottom of the modal is a dark blue button labeled 'Crear Plantilla'.

Figura 8. Interfaz (Modal) Creación de plantilla. Fuente: Captura de pantalla

- **Interfaz editor de texto, creación de plantilla de contrato:** En la figura 7 se presenta la interfaz siguiente a la modal de creación de plantillas de contratos, donde se muestra el editor de texto donde se podrá crear una plantilla para contratos, en la parte izquierda se tiene una lista de cláusulas que se pueden agregar a la plantilla, en la barra superior a la derecha se tienen tres opciones, una opción de ir al inicio, una opción para agregar una cláusula, es decir crear una nueva cláusula y una opción de guardar. Además se tienen tres opciones de menús desplegables, que son arrendatario, arrendador e inmueble, las cuales contienen campos parametrizables para agregar a tu plantilla de contrato.

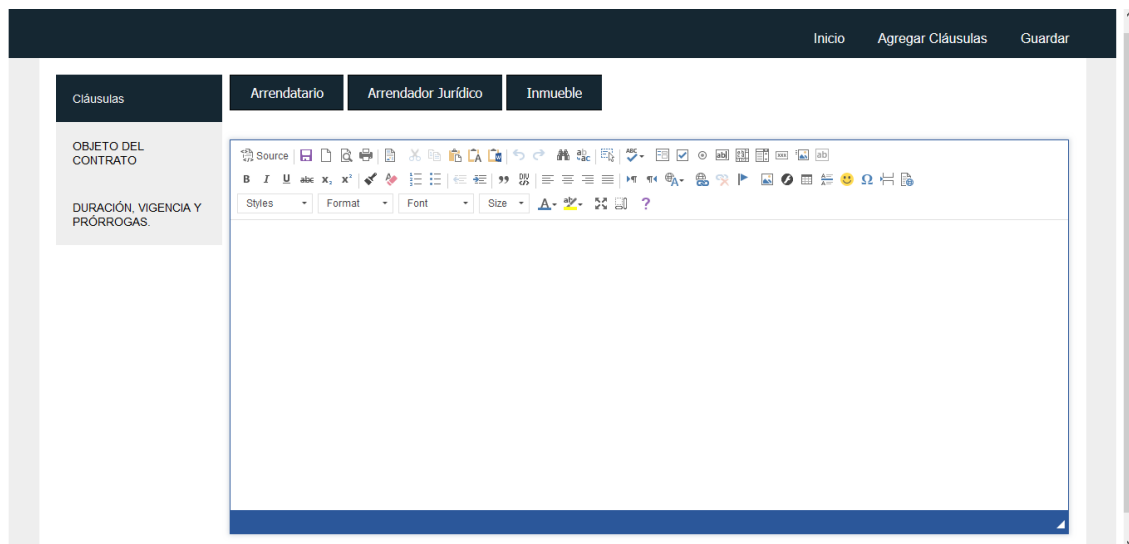


Figura 9. Interfaz Editor de texto para la creación de plantilla de contrato. Fuente: Captura de pantalla.

- **Interfaz correspondiente a la creación de cláusulas de contrato:** En la figura 8, se presenta nuevamente un editor de texto donde se podrá crear la cláusula que queremos agregar, en la parte izquierda tenemos un formulario en el cual indicaremos el título de la cláusula y una descripción de la misma y en la barra superior derecha se tienen las opciones de ir atrás y de guardar respectivamente.

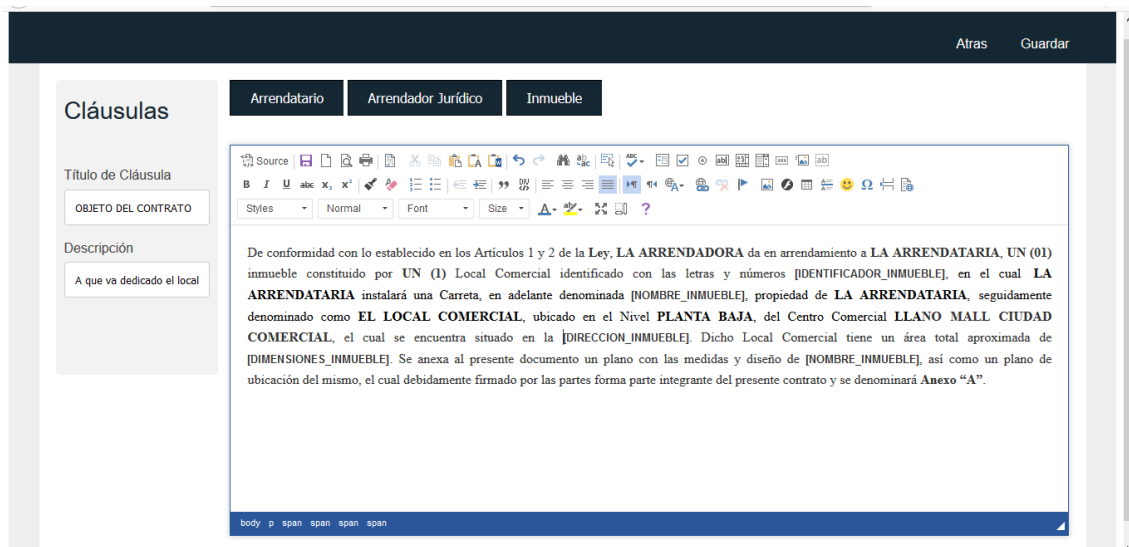


Figura 10 . Interfaz de creación de cláusulas de contratos. Fuente: Captura de pantalla.

- **Interfaz siguiente luego de la creación de una cláusula:** En la figura 9 se presenta la interfaz siguiente luego de crear una cláusula. En esta se informa sobre la exitosa creación de la misma y se actualiza a la interfaz de creación de una plantilla de contrato que era la interfaz en la cual nos encontrábamos anteriormente.

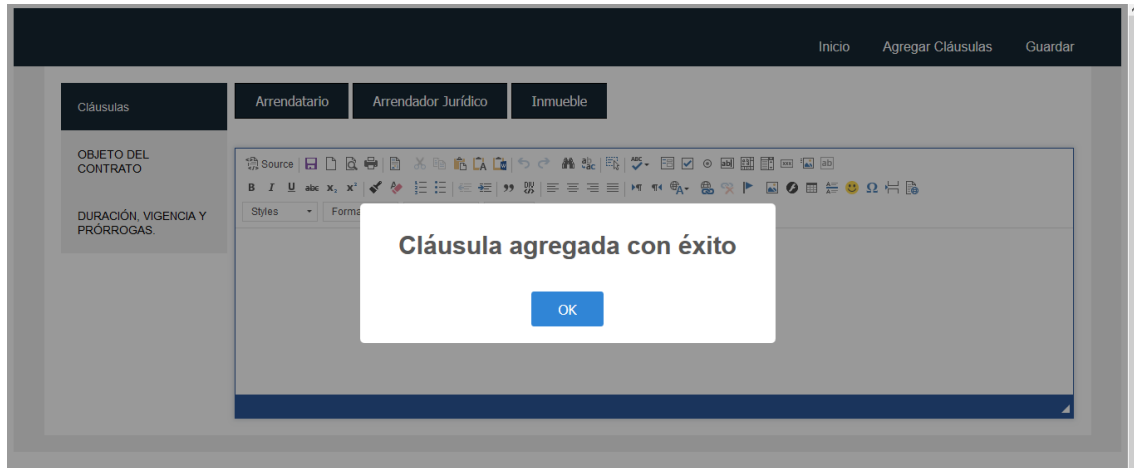


Figura 11. Notificación cláusula creada. Fuente: Captura de pantalla.

- **Interfaz correspondiente a la creación de una plantilla de contrato con los campos parametrizables añadidos:** En la figura 10 se presenta el editor de texto para la creación de plantilla de contrato con la información de un contrato y los campos requeridos de los participantes añadidos como campos genéricos parametrizables, estos campos son seleccionados en el menú desplegable superior izquierdo.

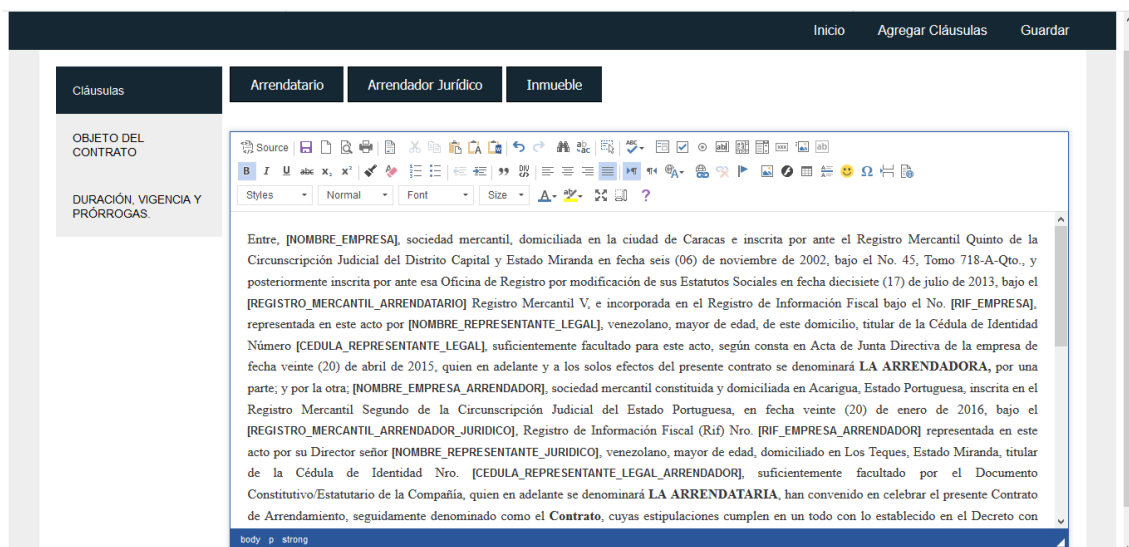


Figura 12. Interfaz de creación de plantilla de contrato con información agregada. Fuente: Captura de pantalla.

- **Interfaz siguiente luego de la creación de una plantilla de contrato:** En la figura 11 se presenta la interfaz siguiente luego de crear una plantilla. En esta se informa sobre la exitosa creación de la misma. Esta aparece luego de seleccionar la opción de guardar.

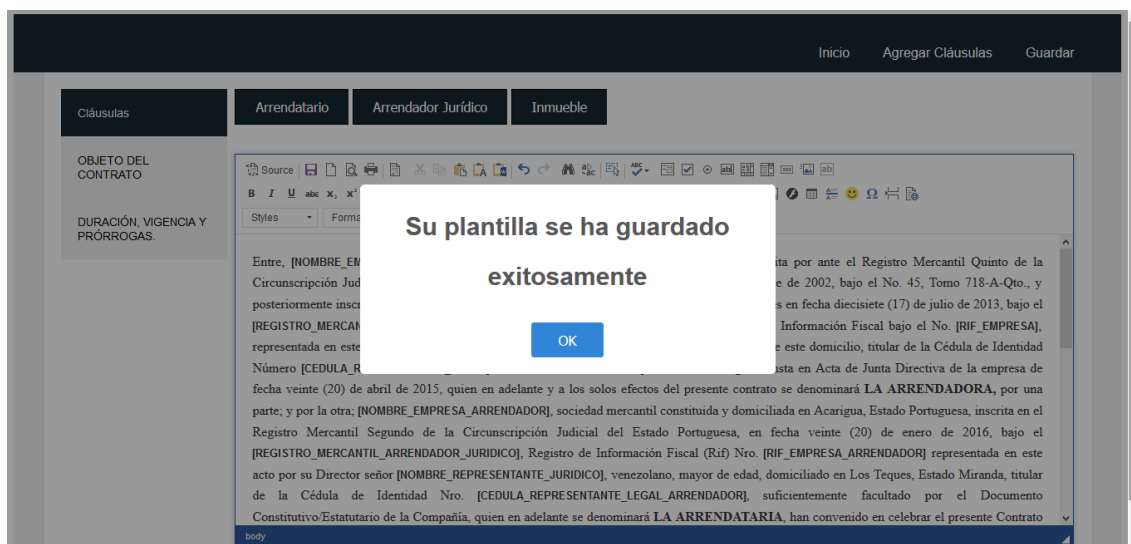


Figura 13. Notificación de plantilla guardada exitosamente. Fuente: Captura de pantalla.

- **Interfaz siguiente a la selección de una plantilla para crear un contrato:** En la figura 12 se muestra una tabla que contiene el nombre del participante o nombre de la empresa participante, cédula o rif y si se refiere a un arrendador o arrendatario. En la cual se procederán a seleccionar los participantes de un contrato inmobiliario, luego de seleccionarlos se procede a dar siguiente en el botón que se encuentra en la parte inferior derecha.

NOMBRE Y/O NOMBRE DE LA EMPRESA	CÉDULA O RIF	ARRENDADOR/ARRENDATARIO
NUTRIFITNESS LLANO MALL, C.A	J-40726069-3	arrendador
ADMINISTRADORA IN SIGN, C.A.	J-30986892-6	arrendatario
COMERCIOS IN SIGN, C.A.	J-55986892-6	arrendatario

Siguiente

Figura 14. Interfaz de tabla de selección de participantes de un contrato. Fuente: Captura de pantalla.

- **Interfaz (modal) correspondiente a la selección de participantes de un contrato:** En la figura 13 se presenta la interfaz utilizada para llevar a cabo la creación de contratos, introduciendo los datos solicitados.

Figura 15. Interfaz siguiente luego de la selección de participantes de un contrato. Fuente: Captura de pantalla.

- **Interfaz siguiente a la selección de participantes, correspondiente a creación de contrato:** En la figura 14 se visualiza la información del contrato, la cual ha sido cargada automáticamente debido a la plantilla seleccionada, y en los cuales los campos parametrizables han sido reemplazados por los datos de los participantes del contrato.

Figura 16. Interfaz Creación de contrato. Fuente: Captura de pantalla.

- **Interfaz siguiente luego de la creación de contrato:** En la figura 15 se presenta la interfaz siguiente luego de crear un contrato. En esta se informa sobre la exitosa creación del mismo. Esta aparece luego de seleccionar la opción de guardar.

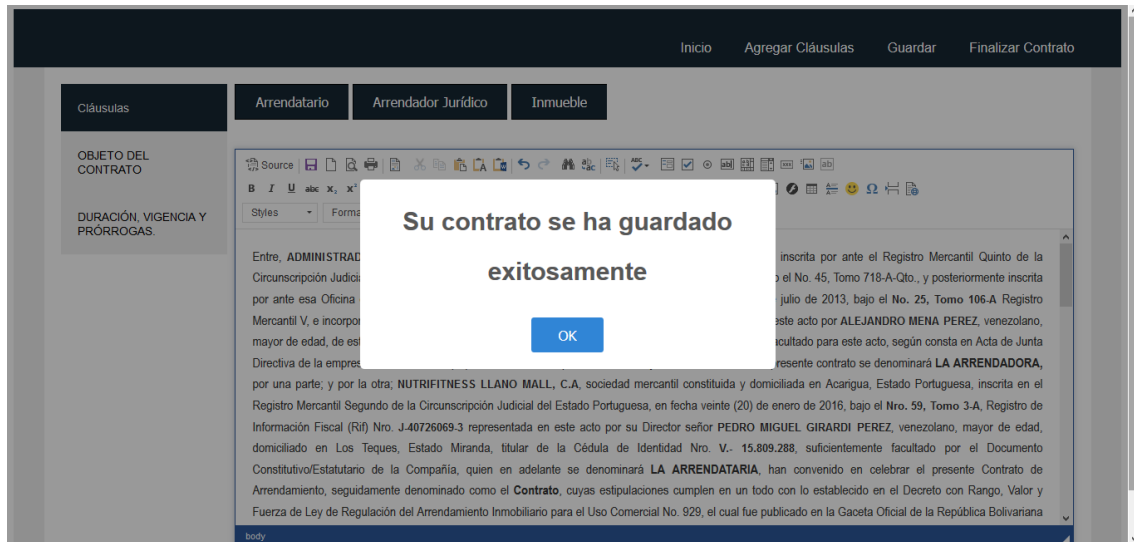


Figura 17. Notificación de contrato guardado exitosamente. Fuente: Captura de pantalla.

- **Interfaz siguiente luego de seleccionar la opción de finalizar contrato:** En la figura 16 se presenta la interfaz siguiente luego de seleccionar la opción que aparece en la parte superior derecha de “Finalizar contrato”. En esta se pregunta si desea finalizar o no el contrato, ya que una vez finalizado no se podrá seguir editando.

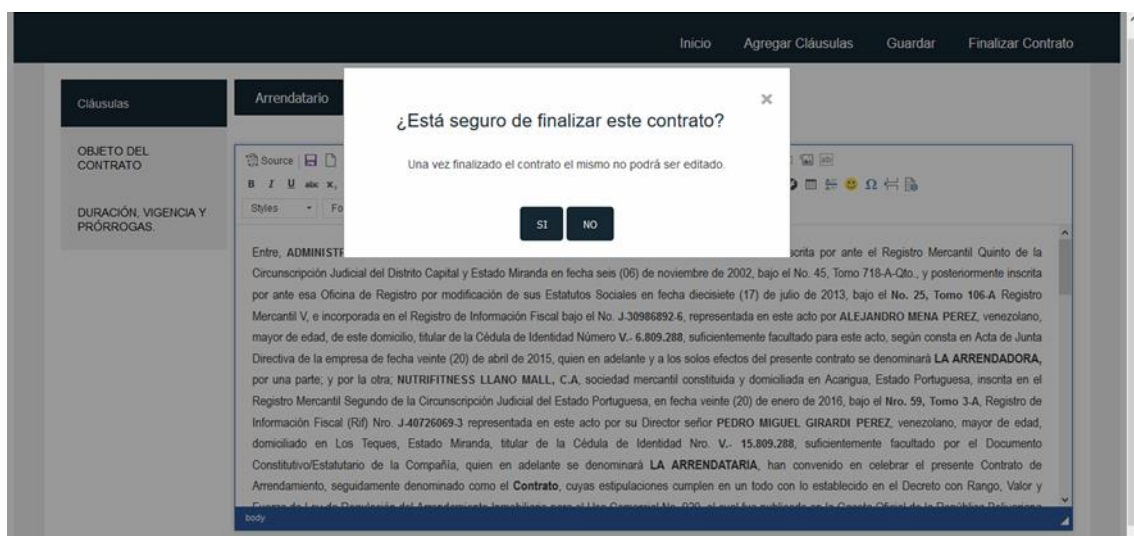


Figura 18. Interfaz modal de la opción de finalizar contrato. Fuente: Captura de pantalla.

- **Interfaz siguiente luego de la creación de contrato:** En la figura 15 se presenta la interfaz siguiente luego de crear un contrato. En esta se informa sobre la exitosa creación del mismo. Esta aparece luego de seleccionar la opción de guardar.

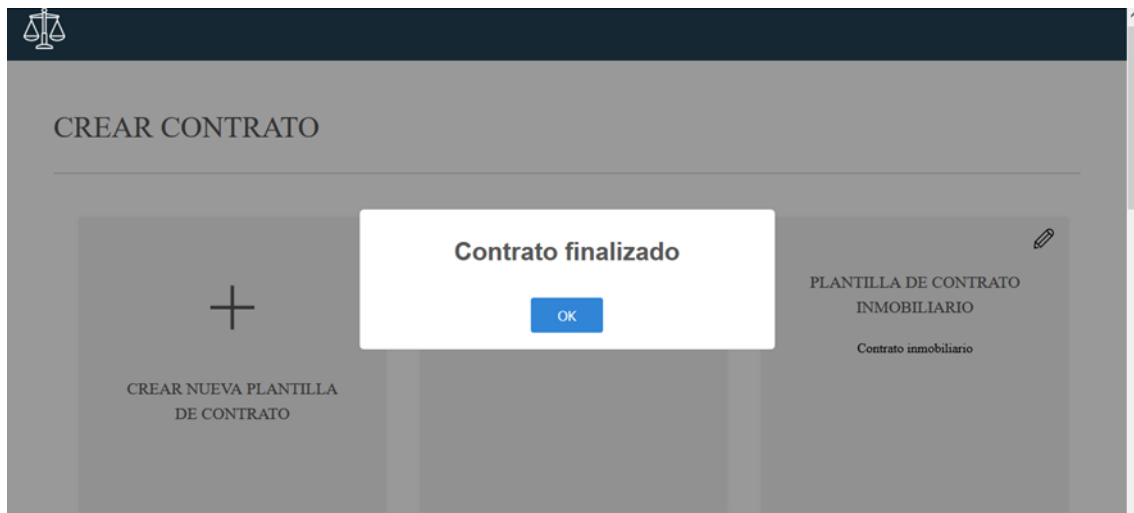


Figura 19. Notificación de contrato guardado exitosamente. Fuente: Captura de pantalla.

- **Interfaz correspondiente a los contratos actuales:** En la figura 18 se presenta la interfaz donde se listan los tres últimos contratos actuales del usuario. Además se visualiza la opción de editar, la cual desplegará la opción de borrar un contrato, y seleccionando el contrato, se redirigirá a la interfaz del editor de texto. En la parte inferior derecha se encuentra la opción de “Más”, la cual redirigirá a la interfaz en la cual se listan todos los contratos actuales.



Figura 20. Interfaz que Lista los contratos actuales. Fuente: Captura de pantalla.

- **Interfaz correspondiente a la opción de borrar contrato:** En la figura 19 se presenta la interfaz con la opción de borrar un contrato.



Figura 21. Interfaz Opción de borrar contrato. Fuente: Captura de pantalla.

- **Interfaz siguiente luego de seleccionar la opción de borrar contrato:** En la figura 20, se visualiza la modal correspondiente a la opción de borrar contrato, en la cual se le pregunta al usuario si desea borrar un contrato y que de borrarlo no podrá recuperarlo.



Figura 22. Interfaz modal de la opción de borrar contrato. Fuente: Captura de pantalla.

- **Interfaz siguiente luego de seleccionar la opción de borrar contrato:** En la figura 21 se presenta la interfaz siguiente luego de borrar un contrato. En esta se informa si el contrato fue borrado de manera exitosa. Esta aparece luego de seleccionar la opción de borrar.

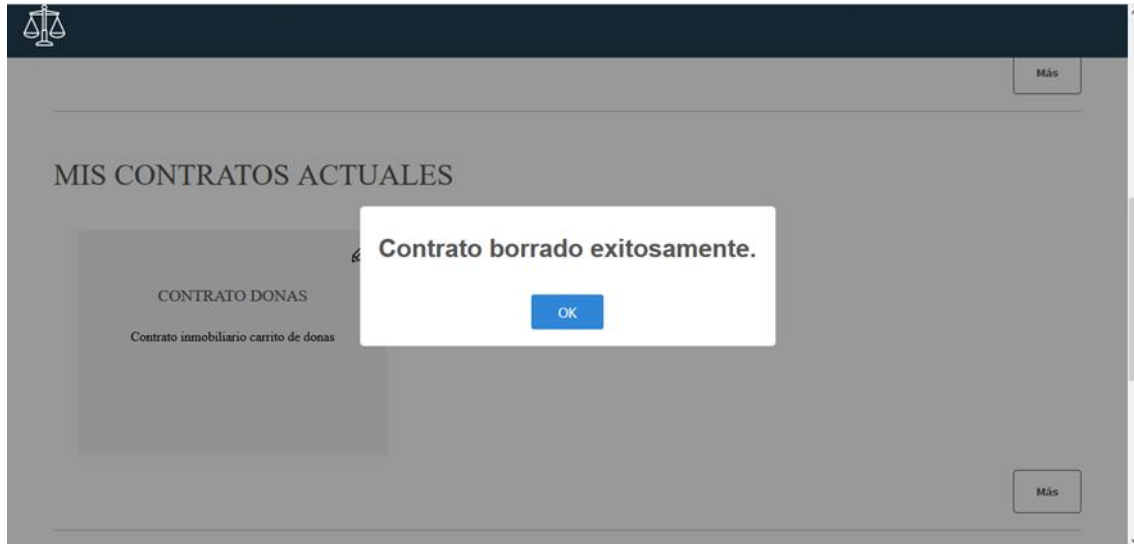


Figura 23. Notificación de contrato borrado exitosamente. Fuente: Captura de pantalla.

- **Interfaz correspondiente a los contratos finalizados:** En la figura 22 se presenta la interfaz donde se listan los tres últimos contratos finalizados del usuario. Además seleccionando el contrato finalizado, se generara el contrato en formato pdf, y se redirigirá a esa interfaz. En la parte inferior derecha se encuentra la opción de "Más", la cual redirigirá a la interfaz en la cual se listan todos los contratos finalizados.



Figura 24. Interfaz que Lista los contratos actuales. Fuente: Captura de pantalla.

- **Interfaz correspondiente a un contrato finalizado en formato Pdf:** En la figura 23 se visualiza el contrato finalizado en formato pdf.

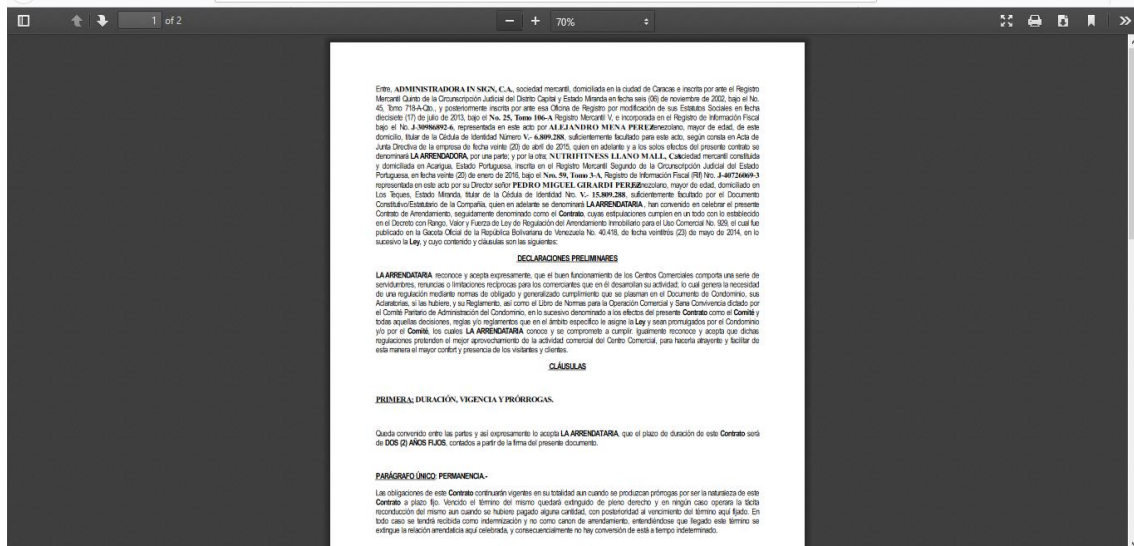


Figura 25. Interfaz de contrato en formato pdf. Fuente: Captura de pantalla.

4.8 Fase de pruebas

Las pruebas realizadas sobre el sistema se encuentran clasificadas en dos: Pruebas Funcionales y Pruebas de Aceptación de Usuario. Las Pruebas Funcionales se realizaron para verificar el cumplimiento de los requerimientos funcionales planteados para el sistema, mientras que las Pruebas de Aceptación de Usuario se realizaron para medir que tan fácil e intuitivo es el sistema para los usuarios finales, y así seguir mejorando el mismo en cuanto a aspectos relacionados a la Interacción Humano – Computador.

4.8.1 Pruebas funcionales

Para estas pruebas se realizó una tabla mostrada a continuación, en la cual se especifican los módulos pertenecientes al sistema y las funcionalidades que deberían tener cada uno de ellos, estas fueron probadas por usuarios especialmente seleccionados que poseyeran un criterio de evaluación asertivo y crítico en cuanto a los resultados. De los cuales tres (3) de ellos desempeñan el cargo de abogados y uno (1) de analista de sistemas.

Módulo	Funcionalidades probadas	Resultado
Creación de contratos	• Autenticación de usuarios. • Consulta de los participantes del contrato. • Creación de plantilla de contrato. • Creación de contrato. • Creación de cláusulas de contrato. • Consulta de plantillas creadas. • Consulta de contratos creados. • Consulta de cláusulas creadas.	
Edición de contratos	• Edición de contrato. • Edición de plantilla de contrato. • Eliminar contrato. • Eliminar plantilla. • Eliminar cláusula. • Almacenar contrato. • Almacenar plantilla de contrato. • Generar contrato en formato pdf.	

Tabla 3. Instrumento evaluación de módulos del sistema. Fuente: Elaborado por la autora.

Las pruebas presentadas en esta tabla se fueron realizando mediante ciclos hasta lograr la completa aprobación de los módulos.

	Ciclo de prueba		
Usuario	1	2	3
1	0 módulos aprobados. 2 módulos no aprobados.	1 módulo aprobado. 1 módulo no aprobado.	2 módulos aprobados.
2	0 módulos aprobados. 2 módulos no aprobados.	1 módulo aprobado. 1 módulo no aprobado.	2 módulos aprobados.
3	0 módulos aprobados. 2 módulos no aprobados.	1 módulo aprobado. 1 módulo no aprobado.	2 módulos aprobados.
4	0 módulos aprobados. 2 módulos no aprobados.	1 módulo aprobado. 1 módulo no aprobado.	2 módulos aprobados.

Tabla 4. Matriz de ciclos de pruebas. Fuente: Elaborado por la autora.

Ciclo 1:

Usuario 1: De los 2 módulos probados, 2 terminaron en un estatus de no aprobación.

Las funcionalidades que fallaron fueron las siguientes:

- No se realizaba de manera correcta la consulta de los datos de los participantes.
- No se creaba la plantilla de contrato de manera correcta.
- No se listaban los contratos del más reciente al más antiguo.
- No se generaba el contrato en formato pdf.

Usuario 2: De los 2 módulos probados, 2 terminaron en un estatus de no aprobación.

Las funcionalidades que fallaron fueron las siguientes:

- No se realizaba de manera correcta la consulta de los datos de los participantes.
- Error al editar un contrato, no guardaba los cambios.
- No se creaba el contrato de manera correcta.

Usuario 3: De los 2 módulos probados, 2 terminaron en un estatus de no aprobación.

Las funcionalidades que fallaron fueron las siguientes:

- No se listaban las plantillas de la más reciente a la más antigua.
- No se creaba la plantilla de contrato de manera correcta.
- No se generaba el contrato en formato pdf.
- No eliminaba la cláusula seleccionada.

Usuario 4: De los 2 módulos probados, 2 terminaron en un estatus de no aprobación.

Las funcionalidades que fallaron fueron las siguientes:

- No se generaba el contrato en formato pdf.
- No se creaba el contrato de manera correcta.
- No se realizaba de manera correcta la consulta de los datos de los participantes.

Ciclo 2:

Usuario 1: De los 2 módulos probados, 1 terminó en un estatus de no aprobación.

Las funcionalidades que fallaron fueron las siguientes:

- No se almacenaba el contrato de manera correcta.
- No se editaban las plantillas.

Usuario 2: De los 2 módulos probados, 1 terminó en un estatus de no aprobación.

Las funcionalidades que fallaron fueron las siguientes:

- No se almacenaban las plantillas.
- No se editaban las plantillas.

Usuario 3: De los 2 módulos probados, 1 terminó en un estatus de no aprobación.

Las funcionalidades que fallaron fueron las siguientes:

- No se almacenaba el contrato de manera correcta.

- No se editaban las plantillas.

Usuario 4: De los 2 módulos probados, 1 terminó en un estatus de no aprobación.

Las funcionalidades que fallaron fueron las siguientes:

- No se almacenaban las plantillas.
- No se editaban las plantillas.

Ciclo 3: En este ciclo todos los usuarios aprobaron todas las funcionalidades del sistema de todos los módulos.

4.8.2 Pruebas de aceptación

Para verificar el cumplimiento de los requerimientos no funcionales especificados para el sistema y el nivel de aceptación de los usuarios finales en cuanto a las interfaces de usuario, se elaboró una encuesta de siete preguntas presentada en la siguiente tabla. Esta encuesta fue aplicada a un grupo de seis personas correspondientes al conjunto de usuarios finales para así analizar la retroalimentación obtenida por parte de estos, y detectar en que aspectos se pueden mejorar las interfaces con la intención de mejorar la experiencia del usuario. La siguiente encuesta está basada en el modelo de usabilidad definido por Nielsen, J. (1999). y la cual fue aplicada a un grupo de seis (6) personas correspondientes al conjunto de usuarios finales para así analizar la retroalimentación obtenida por parte de estos, y detectar en que aspectos se pueden mejorar las interfaces con la intención de mejorar la experiencia del usuario. De los seis (6) encuestados, cuatro (4) desempeñan el cargo de abogados en empresas inmobiliarias y dos (2) desempeñan el cargo de analistas de sistemas.

Nº	Pregunta	Totalmente en desacuerdo	En desacuerdo	De acuerdo	Totalmente de acuerdo
1	¿Considera usted que los colores crean un buen contraste?				

Tabla 5. Encuesta de aceptación. Fuente: Elaborado por la autora.

#	Pregunta	Totalmente en desacuerdo	En desacuerdo	De acuerdo	Totalmente de acuerdo
2	El diseño y estilo de la aplicación es consistente a lo largo de sus interfaces				
3	La aplicación ofrece mensajes acordes a las operaciones realizadas.		.		
5	El lenguaje usado es claro y conciso.				
6	¿El sistema es de navegación intuitiva?				
7	¿Las acciones realizadas son fáciles de memorizar?				

Tabla 6. Encuesta de aceptación (Continuación). Fuente: Elaborado por la autora.

A continuación se presentan los resultados obtenidos para cada pregunta de la encuesta:

Pregunta 1

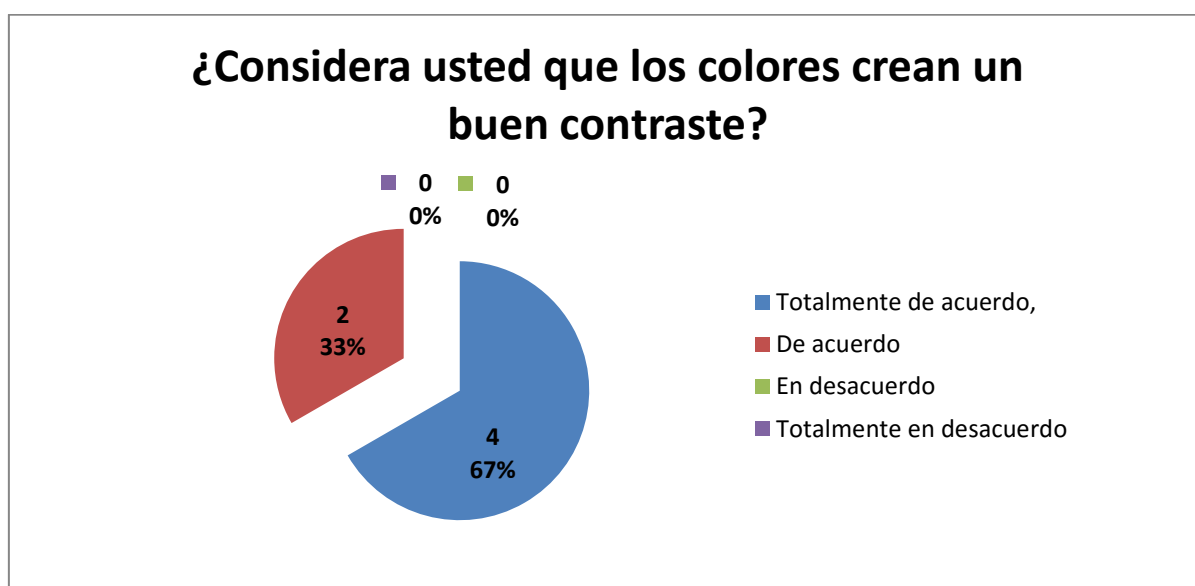


Figura 26. Resultados de la pregunta N° 1. Fuente: Pruebas de aceptación con usuarios finales.

Pregunta 2

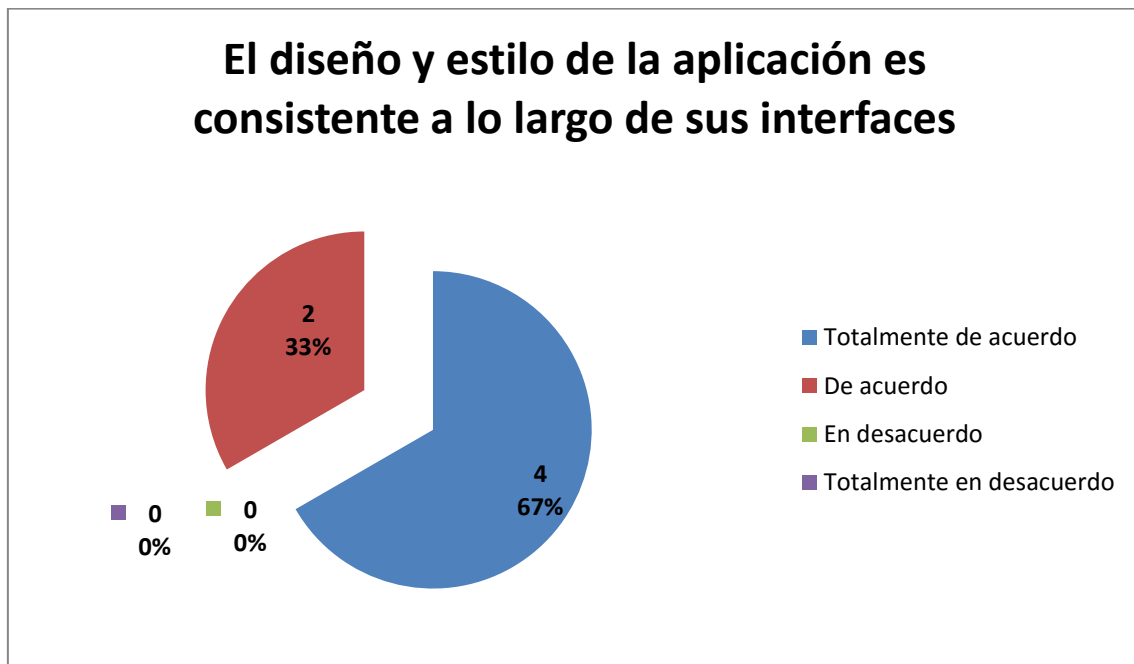


Figura 27. Resultados de la pregunta N° 2. Fuente: Pruebas de aceptación con usuarios finales

Pregunta 3

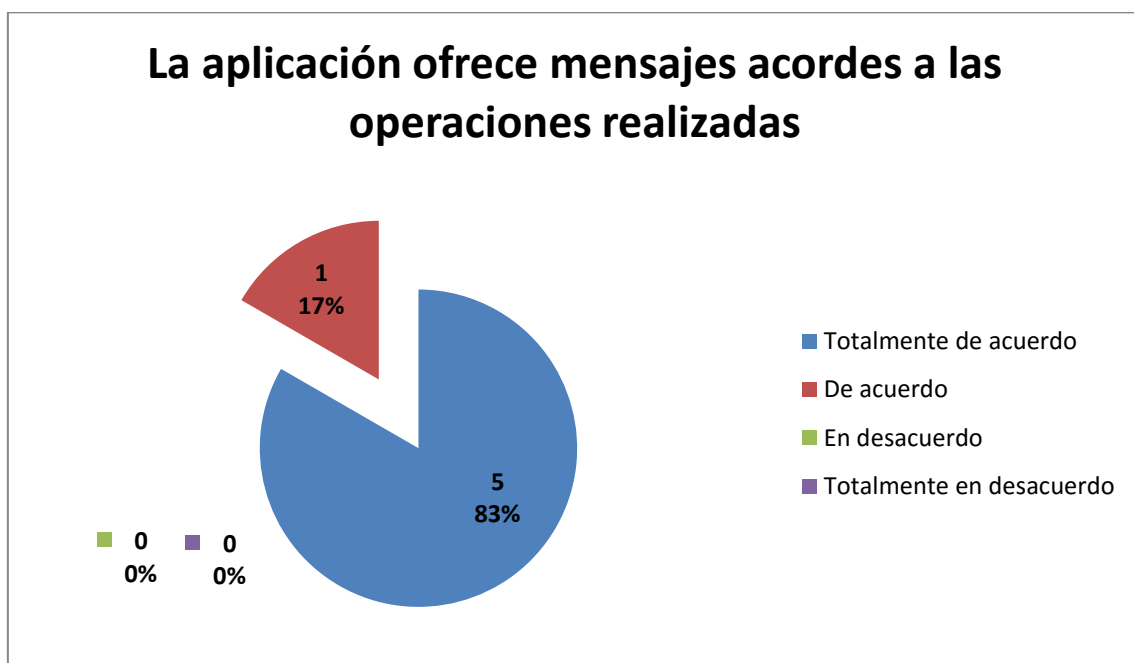


Figura 28. Resultados de la pregunta N° 3. Fuente: Pruebas de aceptación con usuarios finales

Pregunta 4

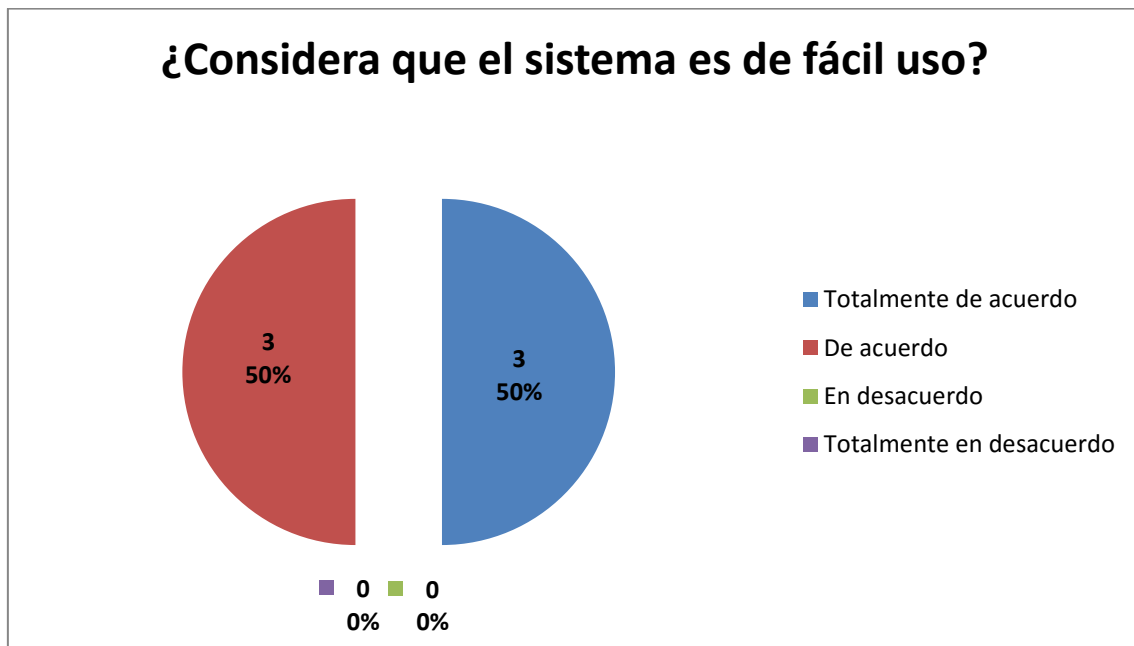


Figura 29. Resultados de la pregunta N° 4. Fuente: Pruebas de aceptación con usuarios finales

Pregunta 5

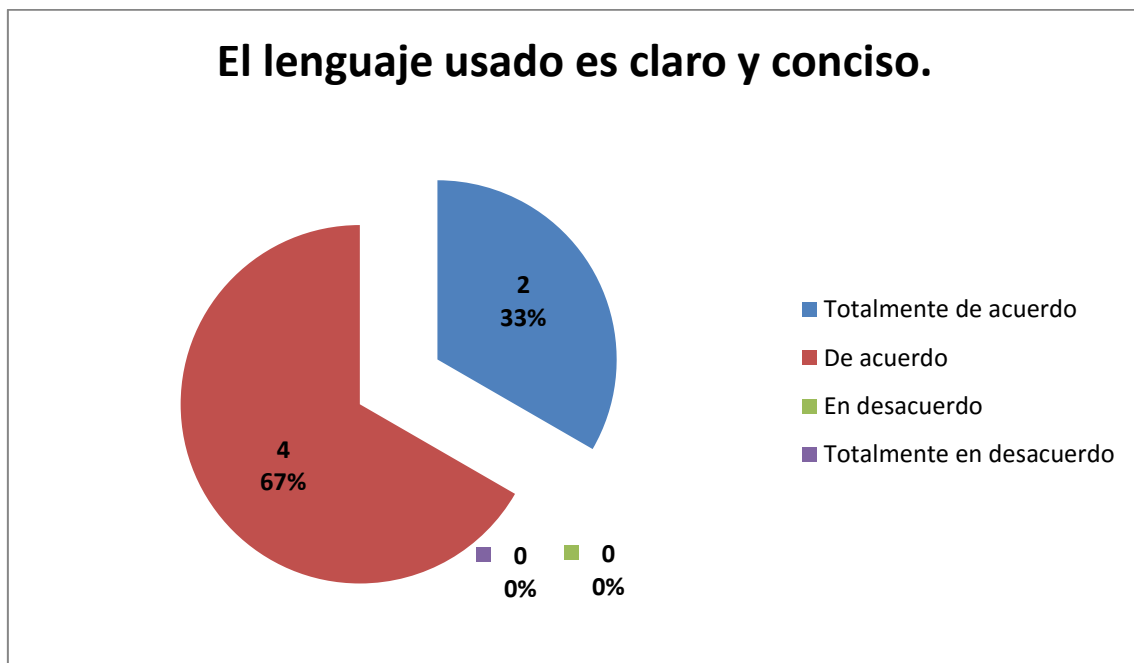


Figura 30. Resultados de la pregunta N° 5. Fuente: Pruebas de aceptación con usuarios finales

Pregunta 6

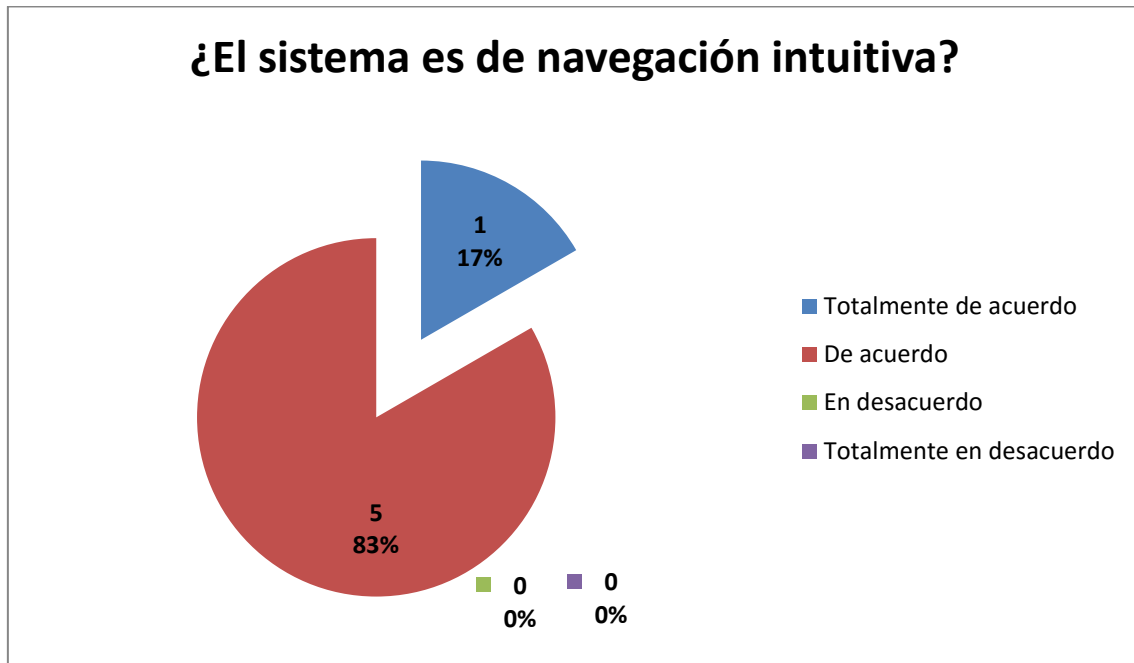


Figura 31. Resultados de la pregunta N° 6. Fuente: Pruebas de aceptación con usuarios finales

Pregunta 7

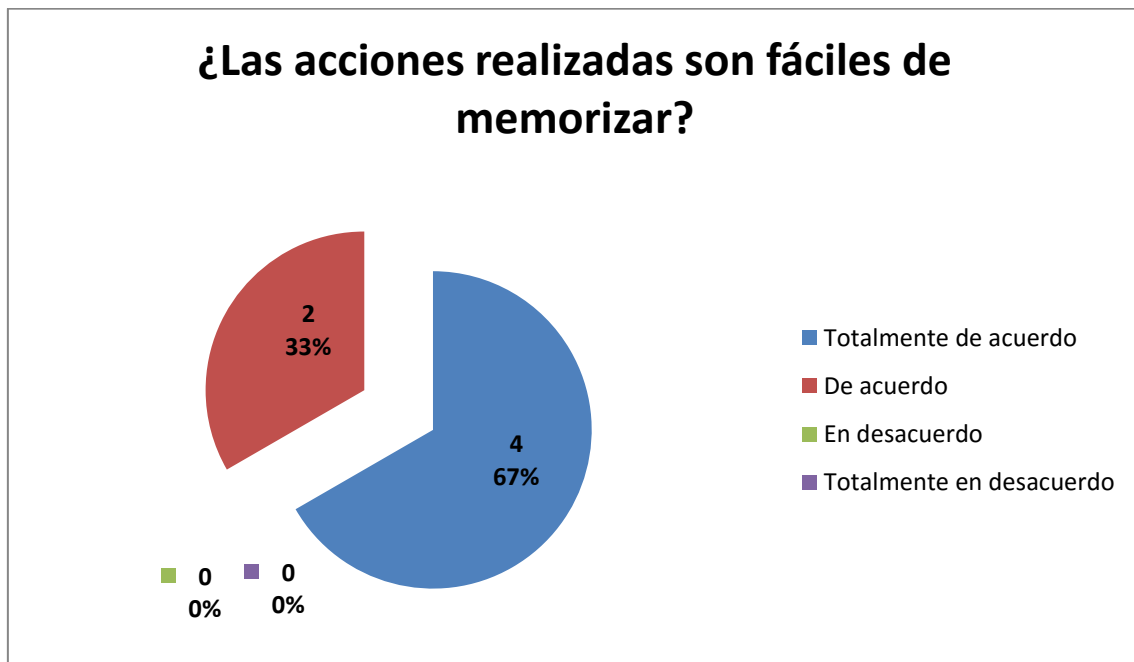


Figura 32. Resultados de la pregunta N° 7. Fuente: Pruebas de aceptación con usuarios finales.

Según los resultados obtenidos a través de la encuesta realizada, se puede concluir que los usuarios perciben el sistema como fácil de usar, de navegación intuitiva, interfaces con un contraste adecuado, interfaces organizadas y consistentes, además de utilizar un lenguaje claro, conciso e informar claramente acerca de los eventos que están ocurriendo en el sistema.

CONCLUSIONES Y RECOMENDACIONES

A través del presente Trabajo Especial de Grado se desarrolló un Sistema de Elaboración de contratos inmobiliarios que cumplió con los objetivos y requerimientos planteados para agilizar de creación de contratos de una empresa inmobiliaria. Para el cumplimiento de los objetivos y requerimientos fue crucial el análisis y entendimiento previo del proceso de negocio que se quería agilizar, detectando así las debilidades y fortalezas que poseía inicialmente para luego plantear una solución que optimizase la ejecución del mismo.

Una vez planteada la solución fue importante evaluar y analizar el conjunto de tecnologías que podían utilizarse para la implementación de la misma con la intención de decidir acertadamente las tecnologías de desarrollo de acuerdo a los requerimientos y especificaciones definidas. Para el desarrollo de este Sistema se utilizaron tecnologías como HTML, CSS, Angular 4, Python Flask, y MongoDB, ya que el uso de estas se adaptaban a los requerimientos y especificaciones definidas para la solución desarrollada. El desarrollo de la solución se planteó mediante la utilización de una Metodología de Desarrollo que se adaptara a las necesidades de la solución y que permitiera ir cumpliendo adecuadamente los objetivos de la misma. La solución desarrollada fue guiada mediante una adaptación de la Metodología de Desarrollo SCRUM.

Se debe tener en cuenta que es de suma importancia para todo sistema adaptarse a nuevas necesidades e incluir con rapidez funcionalidades que permitan satisfacer estas necesidades, por esta razón se debe de tener en cuenta las siguientes recomendaciones:

- Mantener actualizadas las tecnologías utilizadas en el desarrollo del sistema para así aprovechar ampliamente las bondades que puedan ofrecer.

- Considerar en todo momento una posible extensión al sistema, como podría ser una aplicación móvil. Cualquier funcionalidad que se añada al sistema web debería poder ser utilizada también por otras aplicaciones.
- Mantener los estándares del API de los servicios web. Cualquier cambio que se haga en los servicios web implementados en el sistema desarrollado no debería alterar la interfaz del servicio, de lo contrario también deberían ajustarse todos los clientes que consumen dichos servicios.
- Por último incorporar un módulo de gestión documental de manera que se pueda administrar el flujo de los documentos.

BIBLIOGRAFÍA

Abogado (s,f), Diccionario de Arrendador-Arendatario. Artículo en Línea. Disponible en: <https://www.abogado.com/recursos/ley-del-propietario-y-del-arrendatario/diccionario-de-arrendador-arrendatario.html>

ACENS (s,f) . Tipos de Base de datos NoSQL. Artículo en línea. Disponible en: <https://www.acens.com/wp-content/images/2014/02/bbdd-nosql-wp-acens.pdf>

Alegsa, L. (2006). Diccionario de Informática y Tecnología: Definición de Servicio Web. Recuperado de http://www.alegsa.com.ar/Dic/servicio_web.php

Alfresco (s.f). About Alfresco. Artículo en línea. Disponible en:

<https://web.archive.org/web/20060326193416/http://www.alfresco.com:80/about/>

Angular (s.f). What is Angular? Artículo en línea. Disponible en: <https://angular.io/docs>

Antonio López Álvarez, Alfonso Ortega Giménez. (2012) Glosario Jurídico Básico. Página 43.

Apache Cassandra. (2018). Artículo en línea. Disponible en: <http://cassandra.apache.org/>

CakePHP (s.f). CakePHP. Artículo en línea. Disponible en: <http://book.cakephp.org/1.3/es/The-Manual/Beginning-With-CakePHP/What-is-CakePHP-Why-Use-it.html>.

Chanchi, G. E., Campo, W. Y, Amaya, J. P. y Arciniegas, J. L. (2011) Esquema de servicios para Televisión Digital Interactiva, basados en el protocolo REST-JSON. Universidad de Cauca, Colombia. Recuperado de :

<http://www.seer.ufgrs.br/index.php/cadernosdeinformatica/article/view/v6n1p233-240/11807>

CKEditor (s.f). What is CKEditor. Artículo en línea. Disponible en: https://docs.ckeditor.com/ckeditor4/latest/guide/dev_basics.html#what-ckeditor-is

Collell Puig, Jordi (Sin fecha). CSS3 y JavaScript Avanzado. Universitat Oberta de Catalunya, Cataluña, España

Davenport. Thomas, (1993). Process Innovation, Harvard Business School Press, USA.

Definicion.de (s,f). Arrendatario. Artículo en Línea. Disponible en: <https://definicion.de/arrendatario/>

Django (s.f). Django. Artículo en línea. Disponible en: <http://django.es/>.

Fernández Romero, Y. y Díaz González, Y. (2012). Patrón Modelo-Vista-Controlador. Revista Telem@tica. Vol. 11. Recuperado de: <http://revistatelematica.cujae.edu.cu/index.php/tele/article/view/15/10>

Flask (s.f). *About Flask*. Artículo en línea. Disponible en: <http://flask.pocoo.org/docs/0.12/>

Freund J., Rücker B y Hitpass B. (2011) BPMN 2.0 Manual de referencia y guía rápida. Edición Hispana. Primera Edición.

Froala (s.f). *About Froala*. Artículo en línea. Disponible en: <https://www.froala.com/wysiwyg-editor>

Fundación Node.js (s.f). Node.js Artículo en línea. Disponible en: <https://nodejs.org/es/>

IBM Knowledge Center (s,f). Bases de datos relacionales. Artículo en línea. Disponible en: https://www.ibm.com/support/knowledgecenter/es/SSEPGG_8.2.0/com.ibm.db2.udb.doc/admin/c0004099.htm

JSON (s.f). JSON. Artículo en línea. Disponible en: <http://www.json.org/json-es.html>.

Korth, H. F., Silberschatz, A. y Sudarshan, F. S. (2002). Fundamentos de bases de datos, Editorial Mc Graw Hill. Recuperado de: <https://unefazuliasistemas.files.wordpress.com/2011/04/fundamentos-de-bases-de-datos-silberschatz-korth-sudarshan.pdf>

Manuel Borja Soriano. (1962) *Teoría General de las Obligaciones*. Porrúa. Página 129.

Martínez, R (2010). *Información sobre postgresQL*. Documento en línea. Disponible en: http://www.postgresql.org/es/sobre_postgresql.

Microsoft (2015). Información general de SQL Server Compact. Artículo en línea. Disponible en: <http://technet.microsoft.com/es-es/library/ms172448.aspx>.

Mis abogados (s,f). Renovación de contrato de arrendamiento. Artículo en línea. Disponible en: <https://misabogados.com.co/blog/renovacion-de-contrato-de-arrendamiento/>

MongoDB (s,f). What is mongodb? Artículo en línea. Disponible en: <https://www.mongodb.com/what-is-mongodb>

MySQL (s.f), *Sobre MySQL*. Artículo en línea. Disponible en: <https://www.mysql.com/about/>

Neo4j (s,f). What is a Graph Database?. Artículo en línea. Disponible en: <https://neo4j.com/developer/graph-database/>

Nielsen, J. (2003). Usability 101: Introduction to Usability. Recuperado de: <http://www.useit.com/alertbox/20030825.html>

Nielsen, J. (1999). Designing web usability: the practice of simplicity. Indianapolis: New Riders Publishing.

Nuxeo (s.f). About Nuxeo. Artículo en línea. Disponible en:

<https://www.nuxeo.com/es/about/why-nuxeo/>

Olivares Carlos. (2012). GENCI-2 Gestor de Contenido Modular Para la Coordinación de Investigación De la Facultad de Ciencias de la UCV (tesis de pregrado). Escuela de Computación, Facultad de Ciencias de la Universidad Central de Venezuela, Caracas, Venezuela

Oliver A. Perez (2011). *Cuatro enfoques metodológicos para el desarrollo de Software RUP – MSF – XP - SCRUM*. Documento en línea. Disponible en: <http://biblioteca.uniminuto.edu/ojs/index.php/Inventum/article/download/9/9>.

Oracle (2015). *Oracle Database 12c*. Documento en línea. Disponible en: <http://www.oracle.com/lat/products/database/overview/index.html>.

Ospina Torres, M. (2009) *Introducción al manejo de Base de Datos*. Facultad de Ciencias, Universidad Central de Venezuela. Recuperado de: <http://www.ciens.ucv.ve:8080/genasig/sites/bases-de-datos/archivos/GuiaTeoricaTema1BaseDatos.pdf>

Ould. Martyn, (1995). [Proceso de negocio(Business Process)], Documento en línea. Disponible en: http://www.ugr.es/~mnoguera/collaborative_systems-business_processes_10-11.pdf

Protocolo SOAP (s.f). Protocolo SOAP. Artículo en línea. Disponible en: <http://www.alegsa.com.ar/Dic/soap.php>.

Rafael de Pina Vara. (2006). *Diccionario de Derecho*. Porrúa. Página 188

React (s.f). React - A JavaScript library for building user interfaces. Artículo en línea. Disponible en: <https://reactjs.org/>

Ruby on Rails (s.f). ¿Qué es Rails? Artículo en línea. Disponible en: <http://www.rubyonrails.org.es/>

TinyMCE (s.f). *TinyMCE Docs*. Artículo en línea. Disponible en: <https://www.tinymce.com/docs/>

Universidad Central de Venezuela, Sistemas de Información, (2012). Recuperado de: <https://www.scribd.com/doc/123464875/Introduccion-a-BPMS-ppt>

W3C (s.f). XML. Artículo en línea. Disponible en: <http://www.w3c.es/Divulgacion/GuiasBreves/TecnologiasXML>.

Weske. M. (2007) [Proceso de negocio(Business Process)], Documento en línea. Disponible en: <http://alarcos.esi.uclm.es/per/fruiz/cur/santander/fruiz-pn.pdf>

Wikipedia (s.f). *Base de datos*. Artículo en línea. Disponible en: http://es.wikipedia.org/wiki/Base_de_datos.

Wikipedia (s.f). *NoSQL*. Artículo en línea. Disponible en: <https://es.wikipedia.org/wiki/NoSQL>

Wikipedia (s,f). *HTML*. Artículo en línea. Disponible en:
<http://es.wikipedia.org/wiki/HTML>

Wikipedia (s,f). *JavaScript*. Artículo en línea. Disponible en:
<http://es.wikipedia.org/wiki/JavaScript>

Wikipedia (s,f). *Single-page application*. Artículo en línea. Disponible en:
https://es.wikipedia.org/wiki/Single-page_application

Wikipedia (s,f). *Clases de arrendamiento*. Artículo en línea. Disponible en:
https://es.wikipedia.org/wiki/Contrato_de_arrendamiento#Clases_de_arrendamientos