Servicio automatizado para elegir la modalidad de selección de un contratista para entes territoriales de sexta categoría - SAC

Oscar Alexander Rojas Linares
10892218658
Fabián Santiago Carrillo Otálora
10892214782

Universidad Antonio Nariño
Especialización Ingeniería de Software
Facultad de Ingeniería de Sistemas
Bogotá, Colombia
2022

Servicio automatizado para elegir la modalidad de selección de un contratista para entes territoriales de sexta categoría - SAC

Oscar Alexander Rojas Linares
Fabián Santiago Carrillo Otálora

Proyecto de grado presentado como requisito parcial para optar al título de:
Especialista en Ingeniería de Software

Director (a):

MSC. Iván Romero Flórez
MSC. Dianalin Neme Prada

Universidad Antonio Nariño
Especialización Ingeniería de Software
Facultad de Ingeniería de Sistemas
Bogotá, Colombia
2022

NOTA DE ACEPTACIÓN

El trabajo de grado titulado

_____,

Cumple con los requisitos para optar

Al título de _____.

Firma del Tutor

Firma Jurado

Firma Jurado

Bogotá, 04 de noviembre del 2022.

Tabla de contenido

Pág.

Lista de Figuras

Lista de tablas

Resumen

El sistema permitirá al usuario (persona designada para la elaboración de contratos estatales) ingresar una serie de parámetros, los cuales le indicará la modalidad de selección de un contratista del estado, luego de esto el sistema desplegará los campos que debe diligenciar para descargar la plantilla del contrato.

La implementación de este servicio automático innovará en la estandarización de formatos para la selección de contratistas para los entes territoriales de sexta categoría, además de esto, permitirá tener control de la cantidad de acuerdos que se elaboren y reducirá el tiempo de este proceso, manteniendo las plantillas centralizadas y estandarizadas.

Se espera que brindando este servicio se disminuya el tiempo de elaboración de contratos y se estandarice los formatos de los procesos de selección, esto será de gran ayuda para las alcaldías para tener control de los procesos, ya que al automatizarlos y tener centralizada la información permitirá más transparencia en los mismos.

Palabras clave: Automatización; Contratista, Contratos, Plantillas, Servicio.

Abstract

The system will allow the user (person designated for the elaboration of state contracts) to enter a series of parameters, which will indicate the modality of selection of a state contractor, after this the system will display the fields that must be filled out to download the template of the contract.

The implementation of this automatic service will innovate in the standardization of formats for the selection of contractors for the sixth category territorial entities, in addition to this, it will allow control of the number of agreements that are drawn up and will reduce the time of this process, maintaining the centralized and standardized templates.

It is expected that by providing this service, the time for preparing contracts will be reduced and the formats of the selection processes will be standardized. This will be of great help to the municipalities to have control of the processes, since by automating them and having the information centralized, it will allow more transparency in them.

Key words: Automation; Contractor, Contracts, Templates, Service.

Introducción

Según el decreto 1082 de 2015 del departamento nacional de planeación, en el que se compilan la normatividad y los detalles de las cláusulas de la contratación estatal por las cuales los municipios se deben regir para hacer las contrataciones que se requieran, y siguiendo esta normatividad, es necesario hacer documentos para que estos procesos se realicen acorde a la normatividad y con los lineamientos que establece la ley. Dicho esto, se evidencia que estos procesos pueden ser demorados, demandan mucho tiempo (aproximadamente 2 días en la elaboración completa) y conocimiento de la norma; por ello es necesario implementar un sistema automatizado de plantillas, para que el desarrollo de estas actividades se haga de manera más eficiente y rápida, en la que podamos disminuir el tiempo de elaboración y que se minimicen los errores humanos que se puedan cometer al hacer la documentación una por una y manual.

En el desarrollo de este proyecto se identificaron los problemas que abordan la falta de estandarización, corrupción, el clientelismo y la dificultad para entender y elegir las modalidades de selección de un contratista del estado; en el que se justificaron los temas abordados y se planteó como solución, desarrollar una aplicación web expuesta como servicio, la cual permitirá estandarizar los formatos de modalidad de selección de un contratista del estado, además, esta herramienta permite diligenciar, guardar, editar y descargar los formatos completos para que puedan ser publicados en las diferentes herramientas de contratación estatal. Por esto, se determinaron los objetivos a los que se quería llegar, para apreciar los resultados y evaluarlos y así determinar el alcance y las ayudas que estos pueden generar.

Para esto fue necesario desarrollar la teoría y la explicación de los temas tratados, describiendo las herramientas utilizadas para el desarrollo de la aplicación, metodología y la arquitectura seleccionada para hacer software óptimo y eficiente, finalmente se muestran el proceso de construcción, pruebas de seguridad y calidad, proceso de despliegue.

Formulación y descripción del problema

Descripción del problema

Actualmente, los entes municipales de sexta categoría, no tienen los medios necesarios tecnológicos para hacer el proceso de selección de un contratista del estado de una manera eficiente, ya que la falta de estandarización, corrupción, el clientelismo y la dificultad para entender y elegir los métodos de selección, impiden que este proceso se desarrolle correctamente.

La elección de la modalidad de selección de contratistas del estado tiene problemas serios, puesto que es necesario conocer muy bien la normativa para tomar la mejor decisión dependiendo de la necesidad que se requiera satisfacer, además, este procedimiento toma aproximadamente un día en su elaboración y se pueden presentar varios errores humanos en la elaboración de estos documentos por problemas en la tipología y estandarización.

Por tanto, es necesario automatizar este proceso para reducir tiempos, tener control y registro de los contratos, minimizar riesgos de corrupción, disminuir el error humano y ayudar a que este proceso sea más sencillo de entender.

Formulación del problema

¿A través de que mecanismo se puede implementar un servicio en el cual se disminuya el tiempo en la elaboración de las modalidades de selección de un contratista del estado?

Objetivos

Objetivo General

Ofrecer un servicio de plantillas automatizado para elegir la modalidad de selección de un contratista del estado para entes territoriales de sexta categoría para el año 2022.

Objetivos Específicos

- Construir un generador de plantillas para la elección de la modalidad de selección de contratistas automáticamente.
- Facilitar un servicio de plantillas para el diligenciamiento de los formatos de las modalidades de selección, disminuyendo el tiempo en su elaboración.
- Estandarizar los formatos usados en los procesos de elección de modalidad de contratistas.
- Elaborar una estadística por tipo de modalidad, por medio de un reporte descargable.

Marco de Referencia

Estado del Arte

En la tabla 1, se observa una comparación entre las aplicaciones actuales del mercado que ofrecen algún servicio de virtualización de plantillas, o virtualización de contratos, donde se puede analizar las diferentes funcionalidades que ofrece cada una.

Aplicaciones de plantillas	Contiene Plantillas de contratos gubernamentales	Cargar Plantillas Prediseñadas	Editar, descargar plantillas	Fácil uso de plantillas	Costos anual (USD)
trackado	X	X	✓	✓	299
Pandadoc	X	✓	✓	✓	No registra
Gatekeeper	✓	✓	✓	✓	10.500
Legito	✓	✓	✓	✓	11.520 (un solo usuario)
Icertis Suite	✓	✓	✓	✓	NO registra
Servicio automatizado para la selección de contratos	✓	✓	✓	✓	Sin definir (un solo usuario)

Tabla 1: Cuadro comparativo entre las aplicaciones que ofrecen servicios similares

Fuente: Elaboración propia.

El estado del arte nos permite observar que las diferentes aplicaciones en el mercado actual ofrecen un producto igual, con funcionalidades similares, pero a un alto costo, ya que ofrecen como tal todo el portafolio, incluyendo funcionalidades adicionales como firma digital, alertas de vencimiento de contratos, gestión y cobro por usuario, etc; las cuales no forman parte del problema.

Componente de Innovación

La implementación de este servicio automático innovará en la estandarización de formatos para la selección de contratistas para los entes territoriales de sexta categoría, además de esto, permitirá tener control de la cantidad de acuerdos que se elaboren y reducirá el tiempo de este proceso, manteniendo las plantillas centralizadas y estandarizadas.

Impacto

Se espera que brindando este servicio se disminuya el tiempo de elaboración de contratos y se estandarice los formatos de los procesos de selección, esto será de gran ayuda para las alcaldías para tener control de los procesos, ya que al automatizarlos y tener centralizada la información permitirá más transparencia en los mismos.

Marco teórico

- Entes Territoriales Sexta Categoría

Ente Territorial: “Se refiere a las entidades político – administrativas públicas del orden territorial, tales como Departamentos, Municipios, Distritos y Territorios Indígenas.”[1]

En Colombia los municipios se clasifican en categorías uno a seis y categoría especial de acuerdo con su número de habitantes, como lo indica la Ley 617 de 2000 en su artículo 6, en la Figura 1 se puede apreciar la tabla de categorización de los municipios.

Sexta categoría: “Todos aquellos distritos o municipios con población igual o inferior a diez mil (10.000) habitantes y con ingresos corrientes de libre destinación anuales no superiores a quince mil (15.000) salarios mínimos legales mensuales.”[2]

CATEGORIZACIÓN MUNICIPIOS

CATEGORÍA	POBLACIÓN (HABITANTES)	I.C.L.D. (S.M.L.V.)
Especial	Superior o igual a 500.001	Superior a 400.000
Primera	Entre 100.001 y 500.000	Superior a 100.000 y hasta 400.000
Segunda	Entre 50.001 y 100.000	Superior a 50.000 y hasta 100.000
Tercera	Entre 30.001 y 50.000	Superior a 30.000 y hasta 50.000
Cuarta	Entre 20.001 y 30.000	Superior a 25.000 y hasta 30.000
Quinta	Entre 10.001 y 20.000	Superior a 15.000 y hasta 25.000
Sexta	Inferior a 10.000	Inferior a 15.000

LEY 617 DE 2000 – ART. 6°

Figura 1: Categorización municipios de Colombia según la Ley 617 de 2000

Fuente: https://www.quienesquien.co/cuales-las-categorias-los-municipios-colombia/

- Decreto 1082 de 2015 Capítulo 2 - Sección 1

DISPOSICIONES ESPECIALES DEL

SISTEMA DE COMPRAS Y CONTRATACIÓN PÚBLICA

Modalidades de Selección

Subsección 4

- Contratación directa

“Artículo 2.2.1.2.1.4.4. Convenios o contratos inter administrativos. La modalidad de selección para la contratación entre Entidades Estatales es la contratación directa; y en consecuencia, le es aplicable lo establecido en el artículo 2.2.1.2.1.4.1 del presente decreto.” [3]

“Artículo 2.2.1.2.1.4.9. Contratos de prestación de servicios profesionales y de apoyo a la gestión, o para la ejecución de trabajos artísticos que solo pueden encomendarse a determinadas personas naturales. “[3]

“Las Entidades Estatales pueden contratar bajo la modalidad de contratación directa la prestación de servicios profesionales y de apoyo a la gestión con la persona natural o jurídica que esté en capacidad de ejecutar el objeto del contrato, siempre y cuando la Entidad Estatal verifique la idoneidad o experiencia requerida y relacionada con el área de que se trate. En este caso, no es necesario que la Entidad Estatal haya obtenido previamente varias ofertas, de lo cual el ordenador del gasto debe dejar constancia escrita.

Los servicios profesionales y de apoyo a la gestión corresponden a aquellos de naturaleza intelectual diferentes a los de consultoría que se derivan del cumplimiento de las funciones de la Entidad Estatal, así como los relacionados con actividades operativas, logísticas, o asistenciales.”[3]

Subsección 5

- Mínima Cuantía

“Artículo 2.2.1.2.1.5.2. Procedimiento para la contratación de mínima cuantía.

Las siguientes reglas son aplicables a la contratación cuyo valor no excede del diez por ciento (10%) de la menor cuantía de la Entidad Estatal, independientemente de su objeto:

1. La Entidad Estatal debe señalar en la invitación a participar en procesos de mínima cuantía la información a la que se refieren los numerales 2, 3 y 4 del artículo anterior, y la forma como el interesado debe acreditar su capacidad jurídica y la experiencia mínima, si se exige esta última, el cumplimiento de las condiciones técnicas exigidas, incluyendo las obligaciones de las partes del futuro contrato.

2. La Entidad Estatal podrá exigir una capacidad financiera mínima cuando no hace el pago contra entrega a satisfacción de los bienes, obras o servicios. Si la Entidad Estatal exige capacidad financiera debe indicar cómo hará la verificación correspondiente en la invitación.

3. La invitación se publicará por un término no inferior a un (1) día hábil para que los interesados se informen de su contenido y formulen observaciones o comentarios, los cuales serán contestados por la Entidad Estatal antes del inicio del plazo para presentar ofertas. De conformidad con el parágrafo del presente artículo, dentro del mismo término para formular observaciones se podrán presentar las solicitudes para limitar la convocatoria a Mipyme colombianas.

4. La Entidad Estatal incluirá un cronograma en la invitación que deberá tener en cuenta los términos mínimos establecidos en este artículo. Además de lo anterior, en el cronograma se establecerá: i) el término dentro del cual la Entidad Estatal responderá las observaciones de que trata el numeral anterior. ii) El término hasta el cual podrá expedir adendas para modificar la invitación, el cual, en todo caso, tendrá como límite un día hábil antes a la fecha y hora prevista para la presentación de ofertas de que trata el último plazo de este numeral, sin perjuicio que con posterioridad a este momento pueda expedir adendas para modificar el cronograma del proceso; en todo caso, las adendas se publicarán en el horario establecido en el artículo 2.2.1.1.2.2.1. del Decreto 1082 de 2015. iii) El momento en que publicará un aviso en el SECOP precisando si el proceso efectivamente se limitó a Mipyme o si podrá participar cualquier otro interesado. iv) Finalmente, se dispondrá un término adicional dentro del cual los proponentes podrán presentar sus ofertas, el cual será de mínimo un (1) día hábil luego de publicado el aviso en que se informe si el proceso se limita o no a Mipyme.

5. La Entidad Estatal debe revisar las ofertas económicas y verificar que la de menor precio cumple con las condiciones de la invitación. Si esta no cumple, la Entidad Estatal debe verificar el cumplimento de los requisitos de la invitación de la oferta con el segundo mejor precio, y así sucesivamente. Lo anterior sin perjuicio de la oportunidad que deberán otorgar las Entidades Estatales para subsanar las ofertas, en los términos del artículo 5 de la Ley 1150 de 2007, para lo cual establecerán un término preclusivo en la invitación para recibir los documentos subsanables, frente a cada uno de los requerimientos. En caso de que no se establezca este término, los proponentes podrán subsanar sus ofertas hasta antes de que finalice el traslado del informe de evaluación.

6. La Entidad Estatal debe publicar el informe de evaluación durante mínimo un (1) día hábil, para que durante este término los oferentes presenten las observaciones que deberán ser respondidas por la Entidad Estatal antes de realizar la aceptación de la oferta seleccionada.

7. La Entidad Estatal debe aceptar la oferta de menor precio, siempre que cumpla con las condiciones establecidas en la invitación a participar en procesos de mínima cuantía. En la aceptación de la oferta, la Entidad Estatal debe informar al contratista el nombre del supervisor o interventor del contrato.

8. En caso de empate, la Entidad Estatal aplicará los criterios de que trata el artículo 35 de la Ley 2069 de 2020, conforme a los medíos de acreditación del artículo 2.2.1.2.4.2.17. del presente Decreto o las normas que los modifiquen, adicionen o sustituyan.

9. La oferta y su aceptación constituyen el contrato estatal.”[4]

“Artículo 2.2.1.2.1.5.3. Adquisiciones en grandes almacenes cuando se trate de mínima cuantía

Las Entidades Estatales deben aplicar las siguientes reglas cuando decidan adquirir bienes hasta por el monto de su mínima cuantía en establecimientos que correspondan a la definición de "gran almacén" señalada por la Superintendencia de Industria y Comercio:

1. La invitación debe estar dirigida a los grandes almacenes. Esta invitación deberá publicarse en el SECOP y en la página web de la entidad, y contendrá como mínimo: a) la descripción técnica, detallada y completa del bien, identificado con el cuarto nivel del Clasificador de Bienes y Servicios, de ser posible, o de lo contrario con el tercer nivel; b) la forma de pago; c) el lugar de entrega; d) el plazo para la entrega de la cotización que debe ser de mínimo un (1) día hábil; e) la forma y el lugar de presentación de la cotización, y f) la disponibilidad presupuestal.

2. La Entidad Estatal debe evaluar las cotizaciones presentadas y seleccionar a quien, con las condiciones requeridas, ofrezca el menor precio del mercado y aceptar la mejor oferta.

3. En caso de empate, la Entidad Estatal aplicará los criterios de desempate de que trata el artículo 35 de la Ley 2069 de 2020, conforme a los medios de acreditación del artículo 2.2.1.2.4.2.17 del presente Decreto o las normas que los modifiquen, adicionen o sustituyan.

4. La oferta y su aceptación constituyen el contrato estatal.”[4]

- Contratación

“La contratación administrativa es un sistema que emplean las Administraciones Públicas para poder ejecutar buena parte de los trabajos de los que no se pueden encargar.”[5]

- Contratos Estatales

“Son actos jurídicos que celebran las entidades estatales generadoras de obligaciones, determinadas en normas de derecho público y en lo previsto en estas, se debe acudir al derecho privado, dentro de los cuales se encuentran: Contrato de obra, contrato de consultoría, contrato de prestación de servicios, contrato de prestación de servicios profesionales, contrato de prestación de servicios de apoyo a la gestión, contrato de suministro, contrato de concesión, encargos fiduciarios y de fiducia pública, entre otros.”[6]

- Contratación Directa

“Es el procedimiento mediante el cual la entidad estatal contrata directamente con una persona natural o jurídica, la prestación de servicios profesionales, la prestación de servicios de apoyo a la gestión o la adquisición de un bien o servicio, que tenga un proveedor exclusivo o por ser titular de los derechos del mismo.”[7]

- Mínima Cuantía

“La modalidad de selección de mínima cuantía es un procedimiento con términos cortos para escoger al contratista, siempre que el valor de la adquisición de los bienes, obras y/o servicios, no exceda el diez por ciento (10%) de la menor cuantía de la Entidad Estatal.”[8]

- Contratista del estado

“El contratista se constituye en un colaborador o instrumento de la entidad estatal para la realización de actividades o prestaciones que interesan a los fines públicos”[9]

- Plantilla

Es un tipo de documento que crea una copia de sí mismo cuando se abre. También se puede definir como modelo que se puede utilizar para crear otros documentos. En software, una plantilla agiliza el trabajo de reproducción o de muchas copias idénticas o casi idénticas (que no tiene que ser tan elaborado, sofisticado o personal).

Para este sistema se utilizarán las siguientes tecnologías definidas en una arquitectura por capas.

- Java Spring Boot

Es una de las principales herramientas backend con Java del ecosistema del desarrollo WEB. Tiene fácil integración con Spring framework, Spring Data, Spring Security, Spring Cloud lo que hace que los despliegues de las aplicaciones sean bastante simples y eficaces. [10]

- Mysql

“MySQL es el sistema de gestión de bases de datos relacional más extendido en la actualidad al estar basado en código abierto.”[11]

MySql, al ser de código abierto, ofrece una gran comunidad de usuarios, lo que lleva a Mysql a ser fácilmente accesible y ampliamente extendido en sus funcionalidades.

Además de esto, cuenta con las siguientes características, lo que nos llevó a tomar la decisión de optar por esta tecnología para guardar la información que requiramos.

Arquitectura Cliente y Servidor: Comunicación diferenciada entre cliente y servidor para un mejor rendimiento.

Compatibilidad con SQL: Lenguaje generalizado.

Vistas: Capacidad de ver distintas bases de datos basadas en SQL.

Procedimientos almacenados: Eficiencia al procesar información por medio de procesos almacenados.

Desencadenantes: Posibilidad de automatizar tareas dentro de la base de datos.

Transacciones: Posee buena integridad de datos resguardando información.

- React

“Es una librería open source de JavaScript para desarrollar interfaces de usuario. Fue lanzada en el año 2013 y desarrollada por Facebook, quienes también la mantienen actualmente junto a una comunidad de desarrolladores independientes y compañías.

Hoy en día muchas empresas de primer nivel utilizan React para el desarrollo de sus aplicaciones, y es que entre ellas podemos encontrar Facebook, Instagram y el cliente web de WhastApp (todas propiedad de Facebook), y otras como AirBnb, Uber, Netflix, Twitter, Reddit o Paypal.

Desde su lanzamiento, su uso ha ido incrementando notablemente, convirtiéndose, al día de hoy, en una de las tecnologías front-end más utilizadas.”[12]

- Servicios Rest

“Una API de REST, o API de RESTful, es una interfaz de programación de aplicaciones (API o API web) que se ajusta a los límites de la arquitectura REST”[13]

Las API son protocolos y definiciones que son usados par integrar software de aplicaciones. Lo que nos permite interactuar con un sistema para obtener información o ejecutar acciones. En nuestro caso, la utilizaremos para que la interfaz gráfica pueda gestionar información, ya sea de consulta o para almacenar.

Metodología

Para realizar la solución a la problemática planteada, se ha tomado la decisión de usar la metodología KANBAN, esta metodología es fácil de implementar, permitirá conocer el estado real de la ejecución del proyecto, realizar un mayor seguimiento a las tareas realizadas cumpliendo con entregas pequeñas, pero significativas y es cómoda para el grupo de trabajo. Se utilizará el tablero KANBAN de la herramienta de Trello para llevar el seguimiento de las actividades.

Estas están divididas en los siguientes estados:

- En reserva
- Diseño
- ¿Qué está pendiente por hacer?
- En ejecución
- En pruebas
- Finalizado

Cabe resaltar que el primer paso para implementar dicha metodología consiste en listar todas las tareas que a futuro en el proyecto se van a realizar, y a medida que avanza el proyecto las vamos moviendo de posición según el estado que corresponda.

- **En reserva:**
  En este estado definimos todas las tareas a realizar y las organizamos su prioridad de forma ascendente, las más prioritarias y descendente, las de menor importancia. Ver Figura 2.

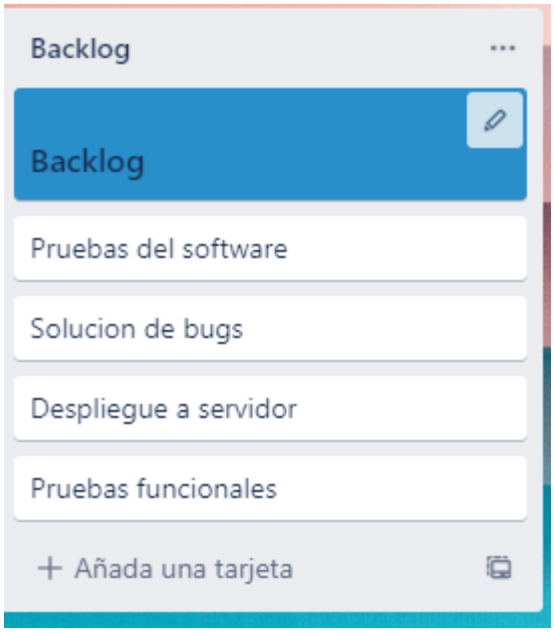

Figura 2: Kanban Backlog

Fuente: Elaboración propia

- **Diseño:**
  En este estado manejamos las tareas que todavía se encuentran en diseño, y dependen algún tema o tarea previa para dar inicio. Ver Figura 3

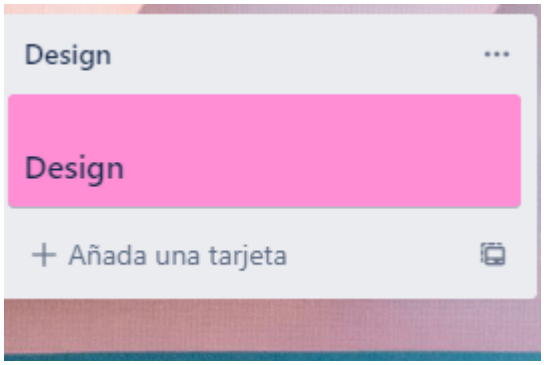

Figura 3: Kanban Design

Fuente: Elaboración propia

- **¿Qué está pendiente por hacer?:**
  Son tareas que no se han podido iniciar por falta de recurso humano o por falta de alguna tarea anterior pendiente. Ver Figura 4

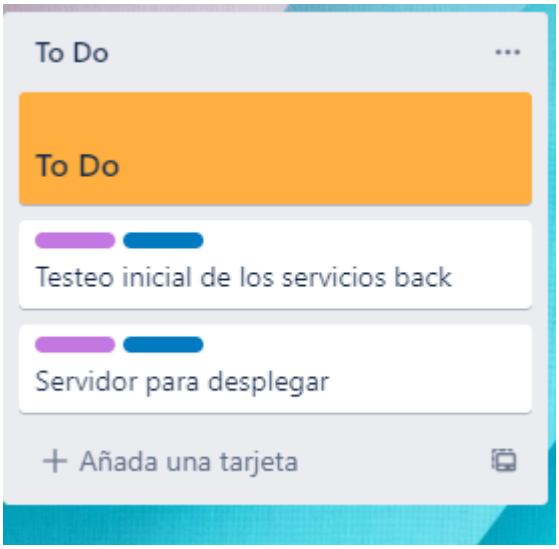

Figura 4: Kanban To Do

Fuente: Elaboración propia

- **En ejecución:**
  Son las tareas que actualmente está trabajando el equipo. Ver Figura 5

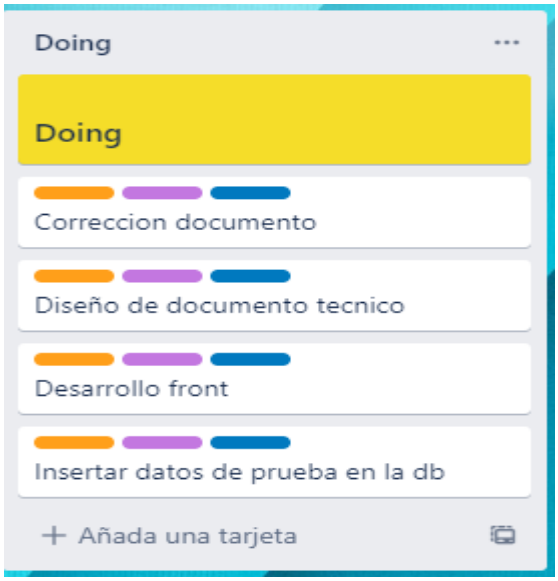

Figura 5: Kanban Doing

Fuente: Elaboración propia

- **En pruebas:**
  Son las tareas que se están probando por parte del equipo para verificar que si se cumple con el resultado esperado. Ver Figura 6

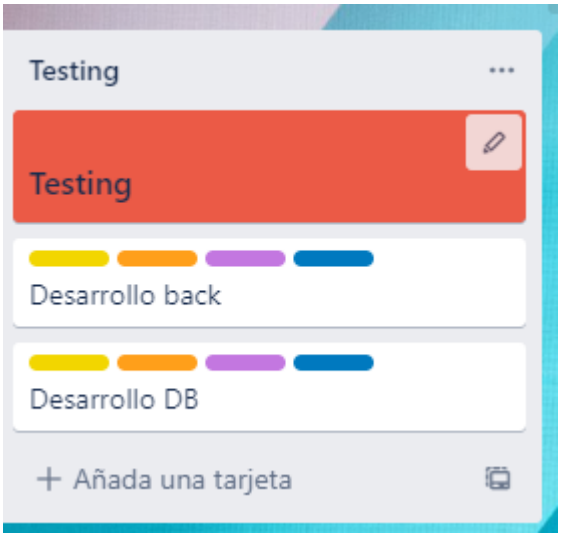

Figura 6: Kanban Testing

Fuente: Elaboración propia

- **Finalizado:**

  Tareas que el equipo considera que ya finalizo de forma satisfactoria. Ver Figura 7

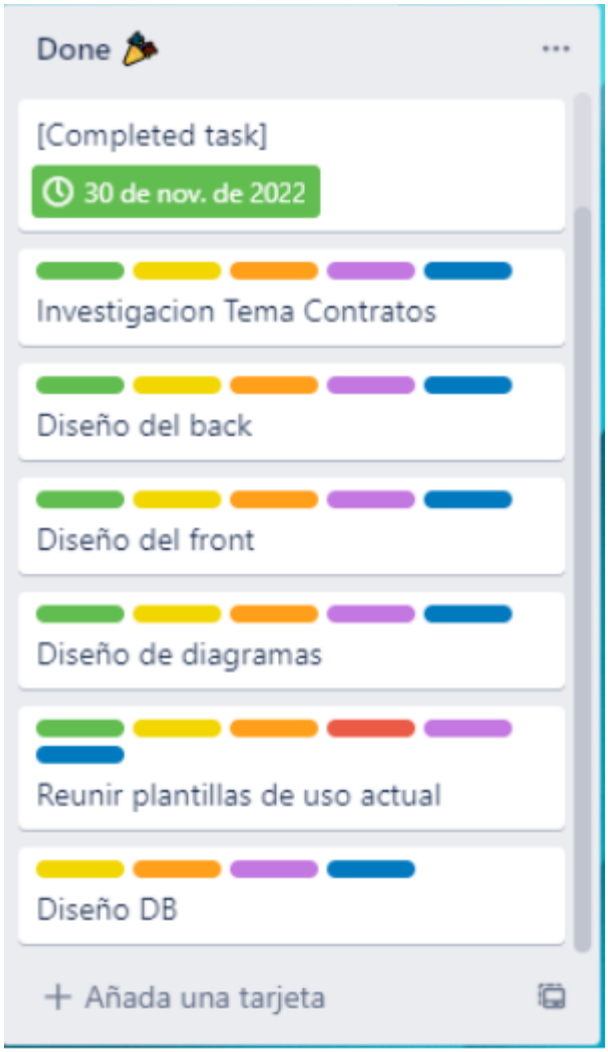

Figura 7: Kanban Backlog

Fuente: Elaboración propia

Cada tarea que paso por cada estado, le agregamos la etiqueta del color correspondiente, con la finalidad de tener una trazabilidad de la tarea, no todas las tareas contienen todos los colores, ya que pueden existir tareas que no sean necesario que pasen por todos los estados, de esta forma tenemos presente cada tarea en que estado se encuentra actualmente en el proyecto.

Proceso de Software

A continuación describiremos las funcionalidades, diagramas y arquitectura de los componentes tecnológicos de nuestro servicio para el desarrollo del planteado.

Detallaremos los requerimientos funcionales y no funcionales que se plantearon para la aplicación, además daremos a conocer cada uno de los diagramas para que sea de mayor entendimiento el propósito del servicio y presentaremos la decisión arquitectural que se tomó para un óptimo desarrollo de la aplicación.

Requerimientos Funcionales

De acuerdo con la metodología usada, las historias de usuario se acomodan perfectamente al marco ágil que nos brinda Kanban. Las historias de usuario nos darán una explicación de la funcionalidad del Software descrita desde la perspectiva del usuario. Lo que nos ayuda a tener orden en el desarrollo para entregar valor al usuario, impulsar el trabajo diario, trabajo en equipo y la mejora de nuestro servicio.

Los requerimientos funcionales fueron generados junto a la persona especialista en Contratación, quien específicamente nos indicó las funcionalidades generales de la aplicación. Estas las describiremos en las siguientes tablas.

Identificación del requerimiento	SAC-1
Título	Seleccionar tipo de modalidad de un contratista
Rol	Usuario
Prioridad	Alta
Descripción	**COMO** usuario **QUIERO** poder ingresar el tipo de contrato y el valor del contrato **PARA** que el sistema me indique la modalidad de selección que necesito diligenciar.
Criterios de aceptación	Se requiere que el sistema valide el tipo de contrato y el valor del contrato para que entregue la modalidad de selección del contratista.

Tabla 2: Historia de usuario SAC-1

Fuente: Elaboración propia.

Identificación del requerimiento	SAC-2
Título	Diligenciamiento de datos
Rol	Usuario
Prioridad	Alta
Descripción	**COMO** usuario **QUIERO** poder ingresar a través de un formulario los datos necesarios **PARA** elaborar un contrato.
Criterios de aceptación	Se requiere que ofrezca un formulario con los parámetros que se deben diligenciar para poder diligenciar un contrato de forma correcta

Tabla 3: Historia de usuario SAC-2

Fuente: Elaboración propia.

Identificación del requerimiento	SAC-3
Título	Descargar plantillas
Rol	Usuario
Prioridad	Alta
Descripción	**COMO** usuario **QUIERO** descargar las plantillas ya diligenciadas **PARA** poder publicarlas en el SECOP
Criterios de aceptación	El sistema debe entregar un archivo comprimido con

	todos los documentos por modalidad de contrato. -10 documentos formato .docx para Contratación Directa -11 documentos formato .docx para Mínima Cuantía

Tabla 4: Historia de usuario SAC-3

Fuente: Elaboración propia.

Identificación del requerimiento	SAC-4
Título	Descarga de reportes
Rol	Admin
Prioridad	media
Descripción	**COMO** administrador **QUIERO** poder descargar reportes **PARA** tener la trazabilidad de los contratos elaborados dependiendo el tipo de contrato.
Criterios de aceptación	El sistema debe presentar un archivo Excel que contenga los reportes de los contratos elaborados según la modalidad que se seleccione.

Tabla 5: Historia de usuario SAC-4

Fuente: Elaboración propia.

Identificación del requerimiento	SAC-5
Título	Adición de usuarios
Rol	Admin
Prioridad	media

Descripción	**COMO** administrador **QUIERO** poder registrar usuarios **PARA** que puedan hacer uso de la creación de contratos.
Criterios de aceptación	El sistema debe permitir al usuario administrador la creación de usuarios normales para que puedan hacer generar contratos.

Tabla 5: Historia de usuario SAC-5

Fuente: Elaboración propia.

Requerimientos no funcionales

Los requerimientos no funcionales del servicio se especifican a continuación.

- El registro y acceso de usuarios debe estar bajo la normativa ISO 27001 - A9 Control de acceso.
- El usuario administrador es el único que puede crear usuarios para que puedan generar contratos.
- La aplicación debe poder ejecutarse en cualquier navegador.
- La aplicación debe ser “responsiva” para garantizar su adecuada visualización en cualquier equipo de cómputo.

Diseño y arquitectura

De acuerdo con los requerimientos funcionales y no funcionales y validando todos los factores que contemplan el desarrollo del software, hemos seleccionado la arquitectura basada en capas, la cual se basará en 3 capas; capa de presentación, capa de lógica de negocio y capa de datos.

Esta arquitectura nos brinda, separación de responsabilidades, seguridad, pruebas y es fácil de desarrollar. De acuerdo con estas características y, ya que el equipo de trabajo es

pequeño y cuenta con experiencia trabajando en Capas, esta arquitectura se acopla adecuadamente al desarrollo.

La siguiente tabla dará a conocer un resumen de lo que nos brinda esta arquitectura.

Requerimiento	**Arquitectura Capas**
Tiempo de implementación y desarrollo (5 Meses)	Alto
Minimizar Costos	Medio
Escalabilidad	Bajo
Mantenimiento	Alto
Disponibilidad	Alto
Testeabilidad	Alto

Tabla 6: Cuadro de ventajas y desventajas de la arquitectura usada.

Fuente: Elaboración propia.

Diagrama de despliegue

Este diagrama muestra la arquitectura de ejecución de nuestro software, describe los diferentes nodos, la relación entre ellos y la forma como se comunican las diferentes capas. Ver Figura 8

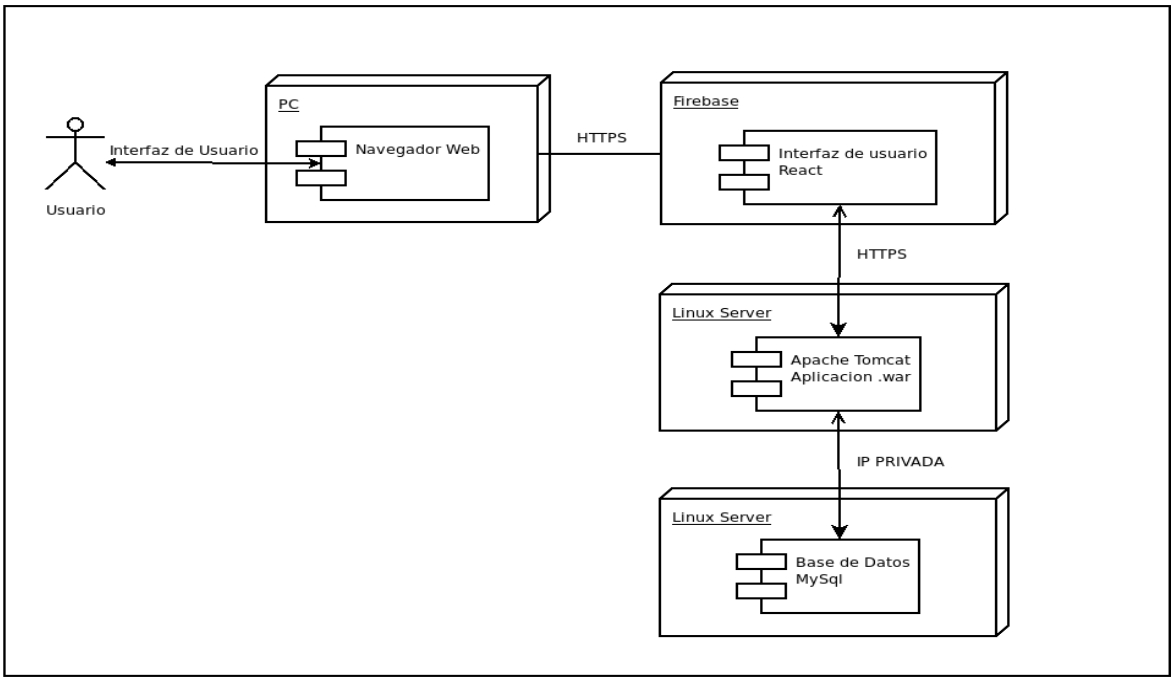

Figura 8: Diagrama de Despliegue

Fuente: Elaboración propia

Podemos apreciar que cada una de las capas tendrá servidor propio sobre distinto sistema operativo, esto para garantizar la disponibilidad, y la rápida recuperación del servicio si se presenta alguna falla. Los servidores estarán bajo la misma red privada, para garantizar seguridad en la comunicación. La capa de presentación y la capa de lógica será vía REST.

Diagrama de Casos de Uso

El siguiente diagrama expresa las principales funcionalidades del sistema. Ver Figura 9.

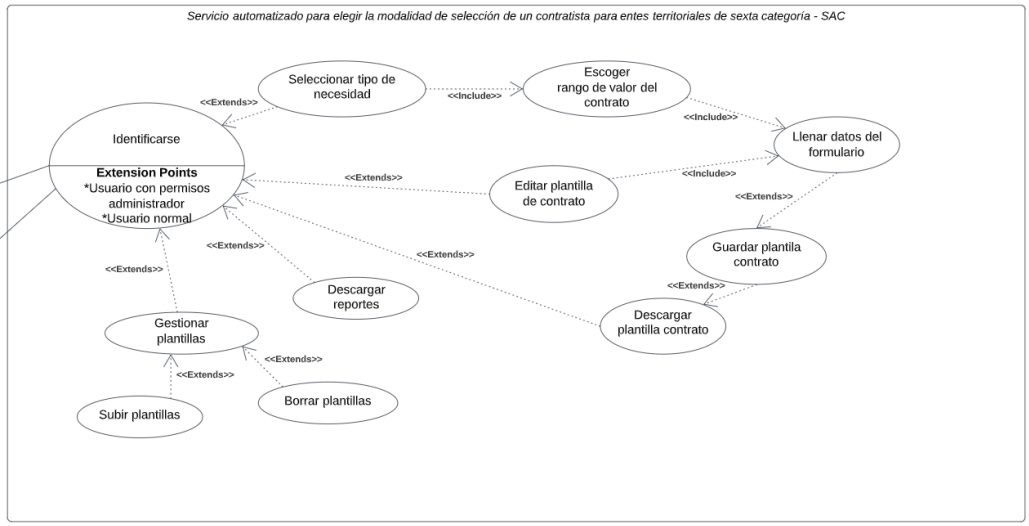

Figura 9: Diagrama de Casos de Uso

Fuente: Elaboración propia

Actores

- Usuario Administrador:

Es el usuario encargado de administrar plantillas y usuarios.

- Usuario Normal:

Es el usuario encargado de diligenciar los campos del formulario que solicita la aplicación.

Casos de Uso

- El usuario o administrador debe estar identificado en el sistema para poder ejecutar cualquier acción.
- El usuario puede escoger la necesidad a satisfacer del contrato(Compra o servicio).
- Luego de esto, debe escoger el rango del valor total del contrato.
- El sistema indicará la modalidad de selección de acuerdo con la necesidad y el usuario podrá seleccionar el tipo de plantilla.
- Se deberá llenar los campos del formulario que el sistema le solicite.

- El usuario podrá guardar o descargar las plantillas diligenciadas.
- El usuario podrá seguir diligenciando una plantilla guardada.
- El usuario podrá descargar un reporte con la cantidad de plantillas de contratos elaborados.
- El administrador de la plataforma, es el único que puede modificar, agregar y borrar plantillas.

Diagrama de secuencia

Este diagrama describe la concurrencia del sistema, permitiendo identificar las partes que pueden ejecutarse de manera simultánea y la sincronización de unidades de ejecución. Ver Figura 10.

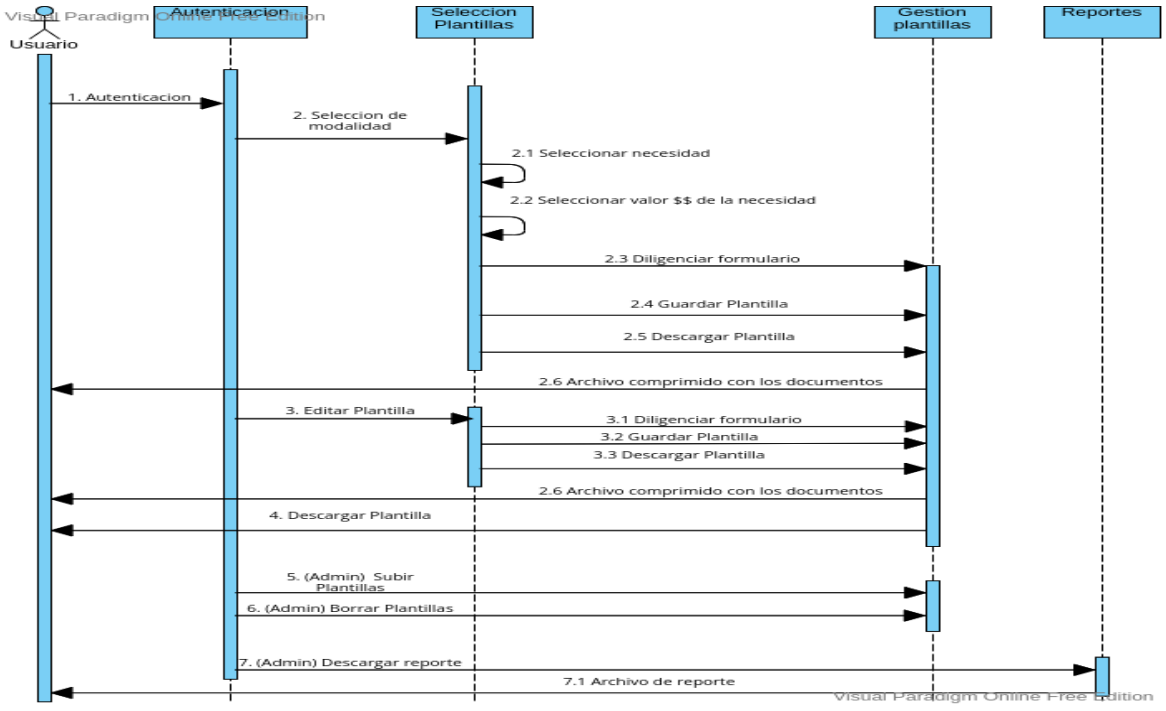

Figura 10: Diagrama de secuencia

Fuente: Elaboración propia

Diagrama de clases

Este diagrama permite abstraer las entidades más relevantes, que expresan la interacción entre los elementos y la cobertura de los requerimientos funcionales. Ver Figura 11.

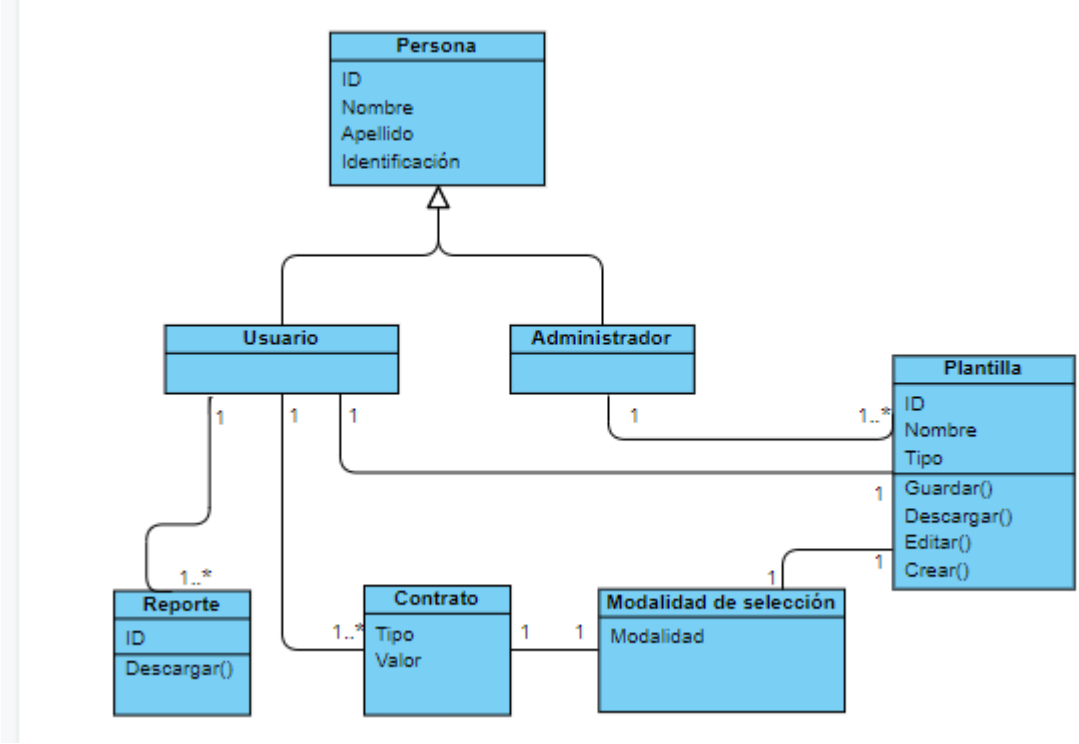

Figura 11: Diagrama de clases

Fuente: Elaboración propia

- Plantillas:

  Es el componente encargado de toda la administración referente a plantillas

- Reportes:

Es el componente encargado de generar el reporte correspondiente a las plantillas descargadas.

- Cuenta Usuario:

  Es el componente encargado de la autenticación y gestión de la cuenta del usuario.

- Contrato:

  Es el componente encargado de recibir los datos de entrada del usuario y procesar qué tipo de contrato se adapta a la solicitud.

Arquitectura de alto nivel

A continuación proporcionaremos una vista general del aplicativo completo, donde se mostrarán los componentes principales para su funcionamiento y la forma en la que interactúan cada uno de ellos. Ver Figura 12

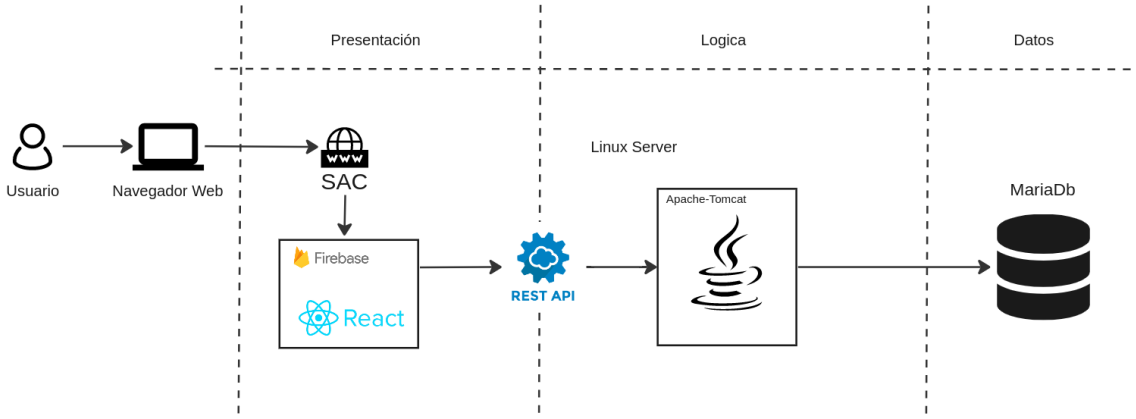

Figura 12: Arquitectura de alto nivel.

Fuente: Elaboración propia

Construcción

A continuación presentaremos un resumen del proceso de desarrollo donde se mencionan las tecnologías utilizadas y la manera en la que construimos las diferentes capas de la aplicación.

- Capa de Datos

Para la capa de datos, hemos usado el motor de base de datos MariaDb, ya que este es de código abierto y lo podemos utilizar de manera adecuada sobre el servidor Linux, algunas de las ventajas por las cuales elegimos este motor de base de datos, son su seguridad, puesto que cuenta con la posibilidad de cifrar la base de datos, su rendimiento y eficiencia en la carga de datos y transacciones.

Para satisfacer las necesidades del software, se creó una base de datos llamada SAC, la cual contiene 4 tablas para almacenar información. A continuación describiremos cada una:

- TEMPLATES TABLE

Esta tabla guarda la información completa de las plantillas generadas por el usuario. Almacena datos como el nombre de usuario, el tipo de modalidad de la plantilla, fechas y objeto tipo Json, que es el que almacena la información del formulario de la plantilla. Ver Figura 13

Column Name	#	Data Type	Not Null	Auto Increment	Key
TEMPLATE_ID	1	int(11)	[v]	[v]	PRI
TEMPLATE_NAME	2	varchar(100)	[v]	[ ]	
TEMPLATE_USER_ID	3	int(11)	[v]	[ ]	
TEMPLATE_TYPE	4	varchar(50)	[v]	[ ]	MUL
TEMPLATE_OBJECT	5	mediumtext	[v]	[ ]	
TEMPLATE_DESCRIPTION	6	varchar(100)	[ ]	[ ]	
TEMPLATE_CREATE_DATE	7	date	[v]	[ ]	
TEMPLATE_LAST_MODIFIED_DATE	8	date	[ ]	[ ]	

Figura 13: Propiedades de la tabla Templates

Fuente: Elaboración propia

- TEMPLATE_TYPE TABLE

Esta tabla guarda los tipos de modalidades de selección. Si la norma cambia y agrega nuevas modalidades de selección, en esta tabla podemos agregarlas y esto nos permite no tener que hacer modificaciones sobre el código. Ver Figura 14

Column Name	#	Data Type	Not Null	Auto Increment	Key
TEMPLATE_TYPE_ID	1	int(11)	[v]	[v]	PRI
TEMPLATE_TYPE_NAME	2	varchar(100)	[v]	[ ]	

Figura 14: Propiedades de la tabla Templates_Type

Fuente: Elaboración propia

- USERS TABLE

Esta tabla guarda la información de los usuarios registrados y nos indica si el usuario tiene privilegios de administración. Ver Figura 15

Column Name	#	Data Type	Not Null	Auto Increment	Key
USER_ID	1	int(11)	[v]	[v]	PRI
USER_NAME	2	varchar(50)	[v]	[ ]	
USER_PASSWORD	3	varchar(50)	[v]	[ ]	
USER_ADMIN	4	int(1)	[ ]	[ ]	
IS_LOGIN	5	varchar(1)	[v]	[ ]	

Figura 15: Propiedades de la tabla Users

Fuente: Elaboración propia

- REPORTS TABLE

La tabla reportes guarda información de cada una de las descargas que se hagan de los templates generados. Esta información nos permite generar los reportes solicitados por el administrador, en los cuales podrá ver tipos de modalidad, fechas, cantidad y el nombre del usuario que los generó. Ver Figura 16

Column Name	#	Data Type	Not Null	Auto Increment	Key
REPORT_ID	1	int(11)	[v]	[v]	PRI
TEMPLATE_TYPE	2	varchar(100)	[v]	[ ]	
TEMPLATE_DOWNLOAD_DATE	3	date	[v]	[ ]	
USER_NAME	4	varchar(100)	[v]	[ ]	
TEMPLATE_NAME	5	varchar(100)	[v]	[ ]	
TEMPLATE_DESCRIPTION	6	varchar(100)	[v]	[ ]	
TEMPLATE_CREATE_DATE	7	date	[v]	[ ]	

Figura 16: Propiedades de la tabla Reports

Fuente: Elaboración propia

- Capa de Lógica

La capa de Lógica, está constituida en JAVA, este lenguaje que es Open Source; una de sus grandes ventajas, nos permite elaborar una aplicación estable, accesible para cualquier S.O y fácil de integrar. Este lenguaje fue elegido también por la experiencia que se tiene trabajando en él. Para la ejecución y desarrollo usamos el IDE Eclipse, el cual es una herramienta muy completa y fácil de utilizar. Para el control de versiones, nos apoyamos en GIT, y para el repositorio usamos las herramientas que nos brinda GITHUB, de esta manera podíamos tener registro de los cambios que hacía alguno de los integrantes del equipo de trabajo, mantener la versión principal estable y tener el código en el repositorio, por si se necesitaba desplegar el código en algún ambiente local para pruebas. Ver Figura 17

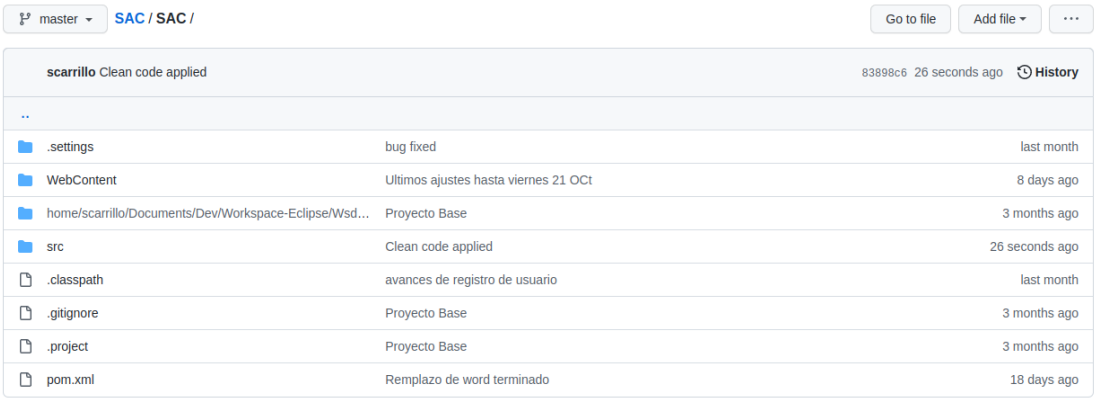

Figura 17: Repositorio GITHUB

Fuente: Elaboración propia

El proceso de construcción del backend se basó en 3 fases. Integración con la base de datos, diseño del API REST y lógica de proceso.

Para la integración con la base de datos se usó el driver jdbc, el cual nos permite interactuar con la base de datos MariaDB. Al tener conexión se crearon los métodos para consultar, actualizar, borrar e insertar datos en cada una de las tablas de la base de datos. Ver Figura 18

```
public class DatabaseTemplates {

    private static String databaseName="SAC";

    private static DatabaseConnectionInformation databaseConnectionInformation;
    private static DateTimeFormatter dateTimeFormatter = DateTimeFormatter.ofPattern("yyyy/MM/dd HH:mm:ss");
    protected static DbAccess dbAccess;


    static {
        try {
            databaseConnectionInformation=new DatabaseConnectionInformation(
                    PropertiesHandler.getPropertyValue("database."+databaseName+".Url"),
                    PropertiesHandler.getPropertyValue("database."+databaseName+".Username"),
                    PropertiesHandler.getPropertyValue("database."+databaseName+".Password"),
                    PropertiesHandler.getPropertyValue("database."+databaseName+".DriverName"),
                    Integer.parseInt(PropertiesHandler.getPropertyValue("database."+databaseName+".Connections")),
                    databaseName);
            dbAccess=new DbAccess(databaseConnectionInformation);
        }
        catch(Exception e) {
            e.printStackTrace();
            Logger.log("Unable to connect to the database "+databaseName);
        }
    }
```

Figura 18: Ejemplo de conexión con la base de datos.

Fuente: Elaboración propia

Para que la capa de lógica se comunicara con la capa de presentación, se construyó una API en el cual se expusieron diferentes métodos para poder cumplir con las necesidades de los requerimientos. Se establecieron 10 métodos, los cuales permiten acciones como, registrarse, obtener el listado de plantillas, guardar cambios, iniciar sesión, entre otros.
La siguiente figura muestra los métodos del API modo documentación. Ver Figura 19.

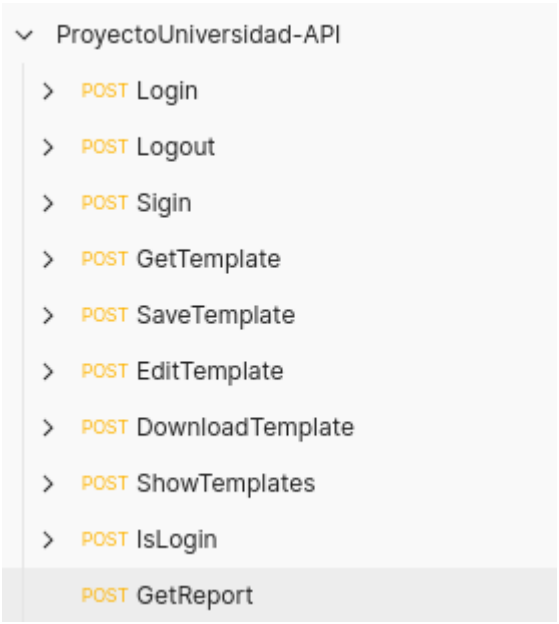

Figura 19: Documentación Api Rest

Fuente: Elaboración propia

Una vez los métodos del API fueron definidos, se dispuso a hacer el desarrollo de cada uno de los métodos, para que procesaran cada solicitud de manera correcta y entregarán respuestas válidas para que la capa de presentación supiera como manejar las solicitudes.

El método DownloadTemplate, es el método encargado de procesar los documentos en formato .docx. Para el manejo de documentos, usamos la librería Apache POI, la cual nos brinda diferentes métodos para manejar los formatos de Office y así poder interactuar con cada uno de los documentos. Esta librería nos permite acciones como, leer, escribir,

modificar y crear documentos office. En este caso solo vamos a manejar documentos formato .docx, ya que fue uno de los requerimientos de la aplicación. Ver Figura 20

```
private static void processingNewDocx(String originalFilePath, String newFilePath, Map<String, String> keyValueList) {

    try {

        Path path = Paths.get(originalFilePath);
        byte[] byteData = Files.readAllBytes(path);

        XWPFDocument doc = new XWPFDocument(new ByteArrayInputStream(byteData));

        for(String keyFi: keyValueList.keySet()) {

            String customFieldKey = Objects.isNull(keyFi) || keyFi.trim().isEmpty() ? null: "«" + keyFi.toUpperCase() + "»";
            String customFieldValue = Objects.isNull(keyValueList.get(keyFi)) ? null : keyValueList.get(keyFi);

            TextReplacer textReplacer = new TextReplacer(customFieldKey, customFieldValue);

            textReplacer.replace(doc);

            if(Objects.nonNull(doc.getTables())) {
                textReplacer.replaceTable(doc);
            }

        }

        FileOutputStream fos = new FileOutputStream(new File(newFilePath));
        doc.write(fos);
```

Figura 20: Ejemplo del método para sobreescribir documentos .docx

Fuente: Elaboración propia

- Capa de presentación.

Para la capa de presentación se utilizó el libreria de React JS con Redux 18.2.0 escrito en formato javascript, la cual es una librería de javascript de código abierto, esta tecnología actualmente cuenta con el respaldo de facebook y la comunidad de software libre, se eligió esta tecnología ya que los miembros del equipo tiene una experiencia previa y así se podría facilitar el tema de implementación. Para el uso de esta librería se utilizó Visual Studio Code, el cual es un editor de código, ligero en el cual también los miembros del equipo tienen una experiencia previa en su utilización.

Esta capa se dividió por componentes, aproximadamente 10 componentes en los cuales se encuentran estructurados los principales módulos de la aplicación, el componente de Redux se utilizó para la administración de los métodos de consumo hacia la parte de la capa de lógica. Ver Figura 21 y 22

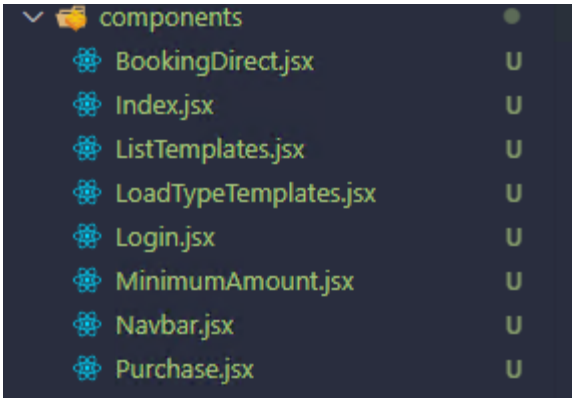

Figura 21: Parte de los componentes creados.

Fuente: Elaboración propia

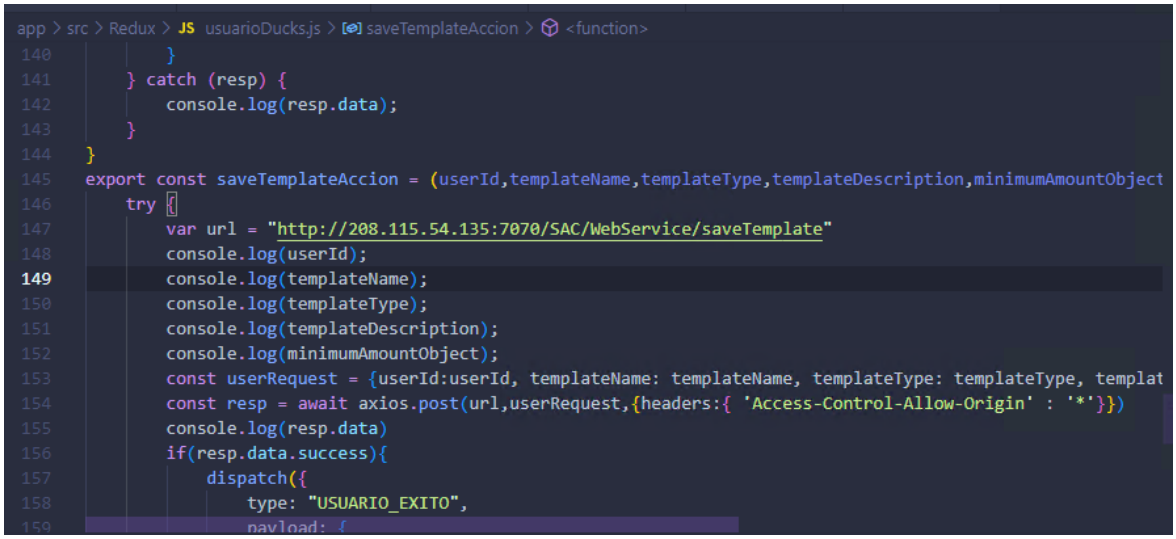

Figura 22: Ejemplo del consumo de los servicios de la capa de lógica.

Fuente: Elaboración propia

Para el manejo de estilos se utilizó la biblioteca de Material UI, para construir las interfaces de usuario, y posteriormente que la aplicación se ajuste dinámicamente a diferentes tipos de pantalla (responsive). Ver Figura 23

app > src > components > Index.jsx > ...

```
1   import React, {useEffect, useState } from 'react'
2   import CssBaseline from '@mui/material/CssBaseline';
3   import AppBar from '@mui/material/AppBar';
4   import Box from '@mui/material/Box';
5   import Container from '@mui/material/Container';
6   import Toolbar from '@mui/material/Toolbar';
7   import Paper from '@mui/material/Paper';
8   import Stepper from '@mui/material/Stepper';
9   import Step from '@mui/material/Step';
10  import StepLabel from '@mui/material/StepLabel';
11  import Button from '@mui/material/Button';
12  import Typography from '@mui/material/Typography';
13  import { createTheme, ThemeProvider } from '@mui/material/styles';
14  import FormControl from '@mui/material/FormControl';
15  import InputLabel from '@mui/material/InputLabel';
16  import Select from '@mui/material/Select';
17  import MenuItem from '@mui/material/MenuItem';
18  import {withRouter} from "react-router-dom";
19  import MinimumAmount from './MinimumAmount';
20  import BookingDirect from './BookingDirect';
```

Figura 23: Ejemplo de la importación de los componentes de Material UI.

Fuente: Elaboración propia

Finalmente, obtenemos los diferentes módulos ya renderizados, los cuales se ven de la siguiente forma. Ver Figura 24 y 25

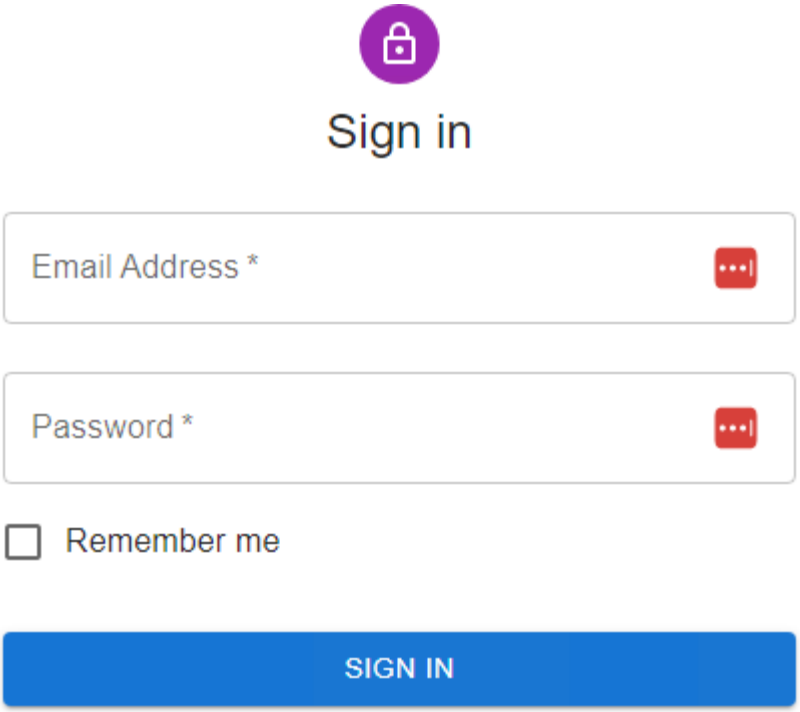

Figura 24: módulo de logueo.

Fuente: Elaboración propia

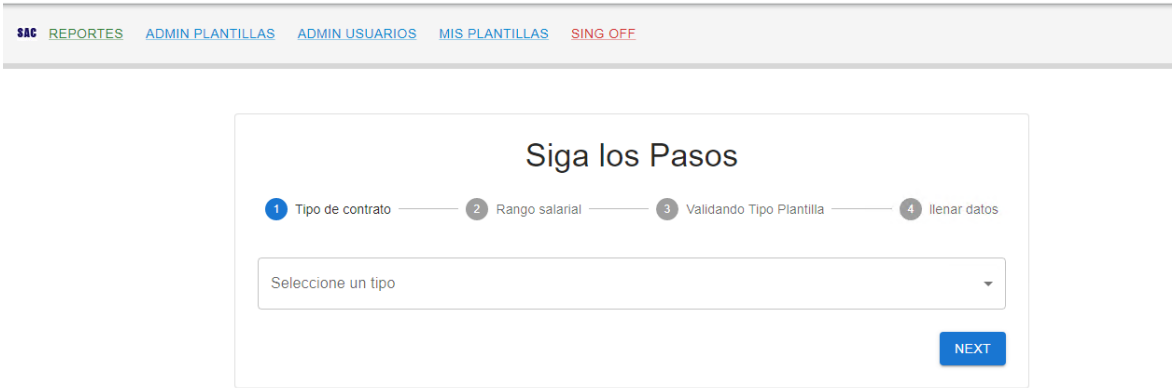

Figura 25: Módulo del home, navbar, index.

Fuente: Elaboración propia

Al finalizar el proceso de selección de tipo de contrato y diligenciamiento de los datos de la plantilla, obtendremos una URL la cual nos permite descargar los archivos en formato .docx. A continuación un ejemplo del archivo generado por la aplicación. Ver Figura 26

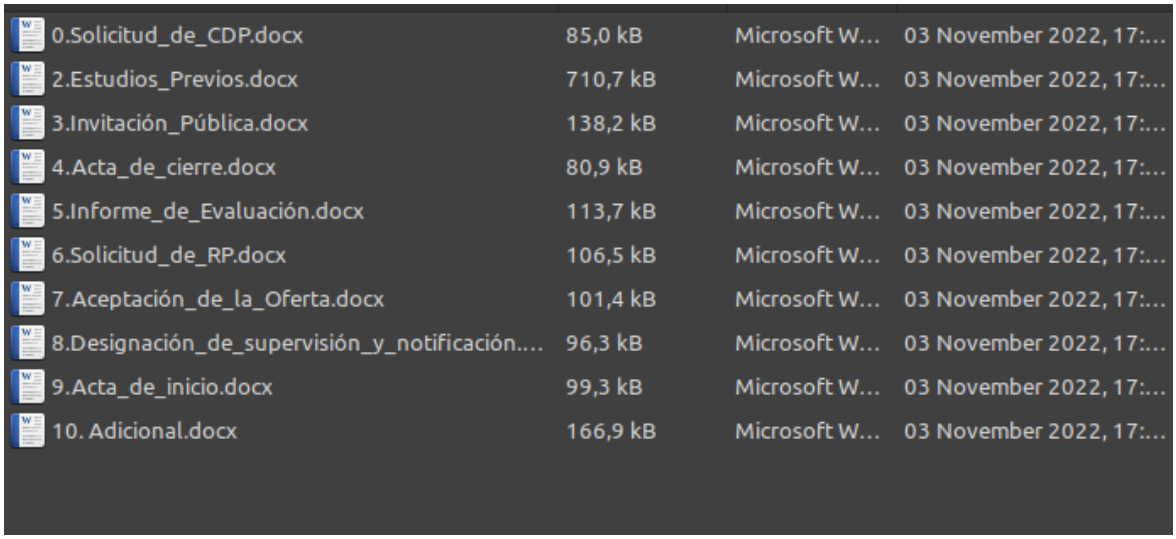

Figura 26: Documentos entregados por la aplicación al finalizar todo el proceso.

Fuente: Elaboración propia

Pruebas de Calidad

En esta sección mostraremos las pruebas de calidad ejecutadas sobre la capa de lógica. Para esto usamos la herramienta SonarQube, la cual nos permite hacer análisis de código estático y nos brinda integraciones para hacer técnicas automatizadas de DevOps.

Primero, instalamos la versión gratuita que brinda SonarQube, con esta herramienta corriendo en local, podemos hacer el análisis del código de manera rápida y realizar las correcciones pertinentes, con eso podemos asegurarnos que no se suban errores de seguridad, bugs y vulnerabilidades al ambiente de producción.

A continuación mostraremos el proceso realizado con el código inicial y el código final.

En la primera validación de calidad, el análisis del código fue con status Failed, ya que contaba con 17 Bugs, los cuales pueden causar bloqueos de la aplicación y que la aplicación no funcione correctamente. Ver Figura 27

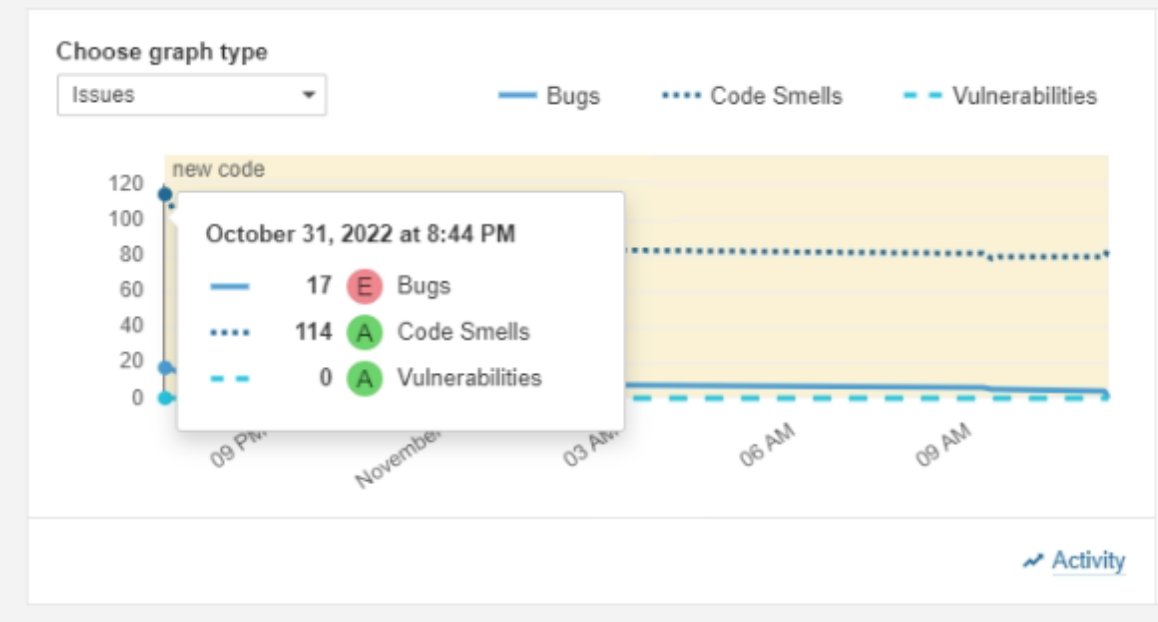

Figura 27: Resultado de SonaQube con el código inicial.

Fuente: Elaboración propia

Luego de obtener el resultado fallido, se procedió a hacer los ajustes necesarios y el code review para que la herramienta nos aprobara la calidad del código y fuese instalada en producción.

La siguiente imagen presenta la corrección de estos bugs y alguna corrección de code smells. Ya con esto, el programa puede ser desplegado en producción con la calidad que la herramienta nos brinda. Ver Figura 28

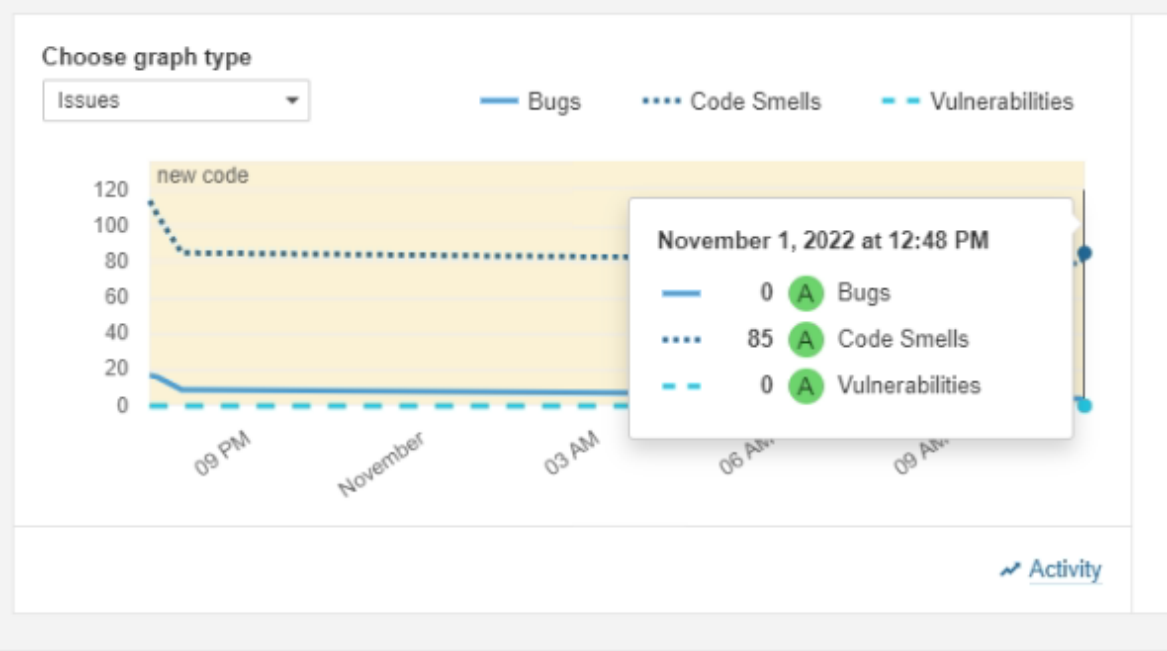

Figura 28: Resultado de SonaQube con el código corregido.

Fuente: Elaboración propia

La siguiente figura evidencia que el código final el cual fue colocado en el ambiente de producción. La herramienta nos dio el estatus de Success y nos mostró que el código cuenta con 0 bugs, 0 vulnerabilidades.

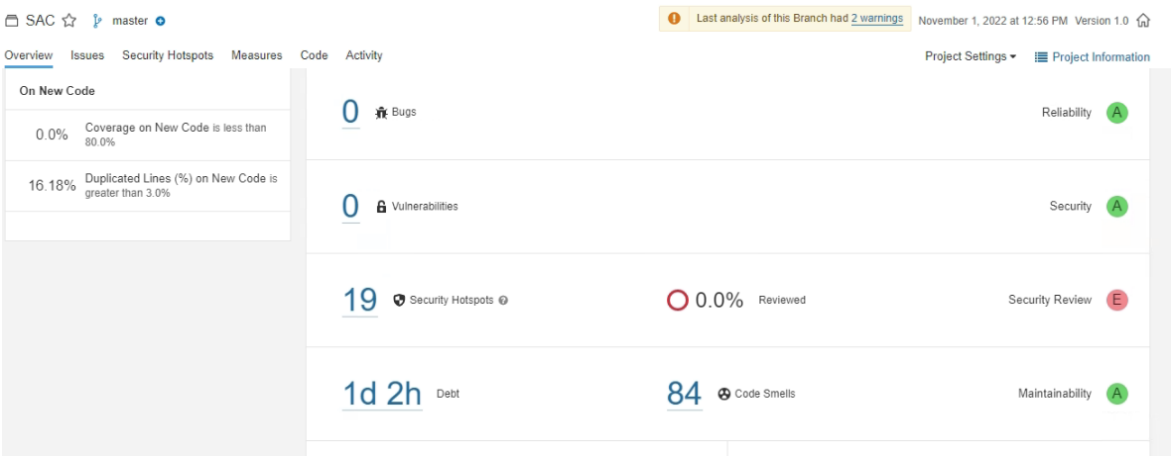

Figura 29: Resultado de SonaQube con el código final.

Fuente: Elaboración propia

Pruebas de Seguridad

Para las pruebas de seguridad usaremos el aplicativo OWASP para validar vulnerabilidades del sistema. En el cual podemos observar que no tenemos riesgos de alto impacto, el gran porcentaje de riesgos encontrados son de tipo informativo. Solo se obtienen 2 riesgos medios y 2 bajos. La siguiente imagen muestra el resultado obtenido por la aplicación. Se entrega el documento completo de los resultados encontrados en pdf como anexo. Ver Figura 30

		Confidence				
		User Confirmed	Alto	Medio	Bajo	Total
Risk	Alto	0 (0,0 %)	0 (0,0 %)	0 (0,0 %)	0 (0,0 %)	0 (0,0 %)
	Medio	0 (0,0 %)	1 (12,5 %)	1 (12,5 %)	0 (0,0 %)	2 (25,0 %)
	Bajo	0 (0,0 %)	0 (0,0 %)	1 (12,5 %)	1 (12,5 %)	2 (25,0 %)
	Informativo	0 (0,0 %)	0 (0,0 %)	2 (25,0 %)	2 (25,0 %)	4 (50,0 %)
	Total	0 (0,0 %)	1 (12,5 %)	4 (50,0 %)	3 (37,5 %)	8 (100%)

Figura 30: Resultado de Owasp.

Fuente: Elaboración propia

Instalación y Configuración

A continuación describiremos el proceso de instalación y configuración de cada una de las capas que usamos para el desarrollo de la aplicación.

- Capa de Datos

La capa de datos, está alojada sobre un servidor Centos y con un motor de base de datos de MariaDb.

Pará su instalación, es necesario contar con un servidor con 8gb de memoria RAM, 100 gb de espacio de disco y 4 cores, como mínimo de requerimientos, para garantizar que la base de datos procese de manera rápida y sea estable en cada transacción que se haga.

Al tener el servidor con estas características, nos disponemos a instalar el sistema operativo, puede ser Ubuntu server o Centos server. Con cualquiera de estos dos S.O, la base de datos puede ser instalada de manera correcta. Luego de tener el sistema operativo instalado y correctamente configurado, instalamos el motor de base de datos MaríaDb con el siguiente comando.

Luego de instalada, procedemos con la creación de usuarios, permisos, configuraciones y con la creación de la base de datos que vamos a usar.

Luego de esto, podemos interactuar sobre algún IDE, como puede ser workbeanch, heidiSQL o Dbeaver, para visualizar y crear de manera más fácil, las tablas y los registros que sean necesarios.

- Capa de Lógica

La aplicación Java, está instalada sobre un servidor Apache Tomcat alojado sobre un servidor Linux.

Al igual que la capa de datos, es necesario contar con un servidor con 16gb de memoria RAM, 50 gb de disco duro y 4 cores, como mínimo de requerimientos, y podremos instalar un SO linux ya sea Ubuntu server o centos server.

Al tener el servidor configurado, instalamos Apache-Tomcat-9 para poder correr el aplicativo Java.

Pará poder correr la aplicación en el servidor, desde eclipse, nos disponemos a generar el archivo .war. Ver Figura 31.

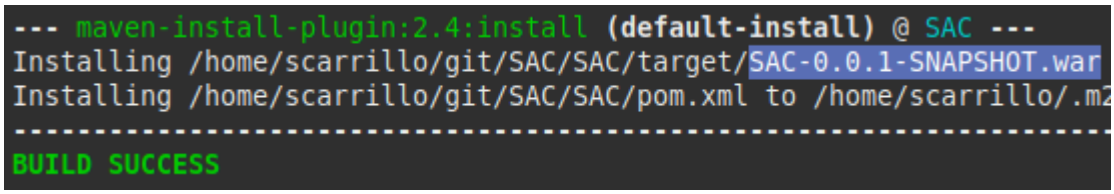

Figura 31: Compilación de la aplicación para generar .war

Fuente: Elaboración propia

Al tener el archivo .war generado, lo transferimos al servidor y allí ejecutamos el Tomcat para correr la aplicación. Para poder ejecutar la aplicación, necesitamos estar ubicados en la ruta del tomcat y alli ejecutar “sh bin/startup.sh” para correr la aplicación. Lo cual se verá como se muestra en la Figura 32.

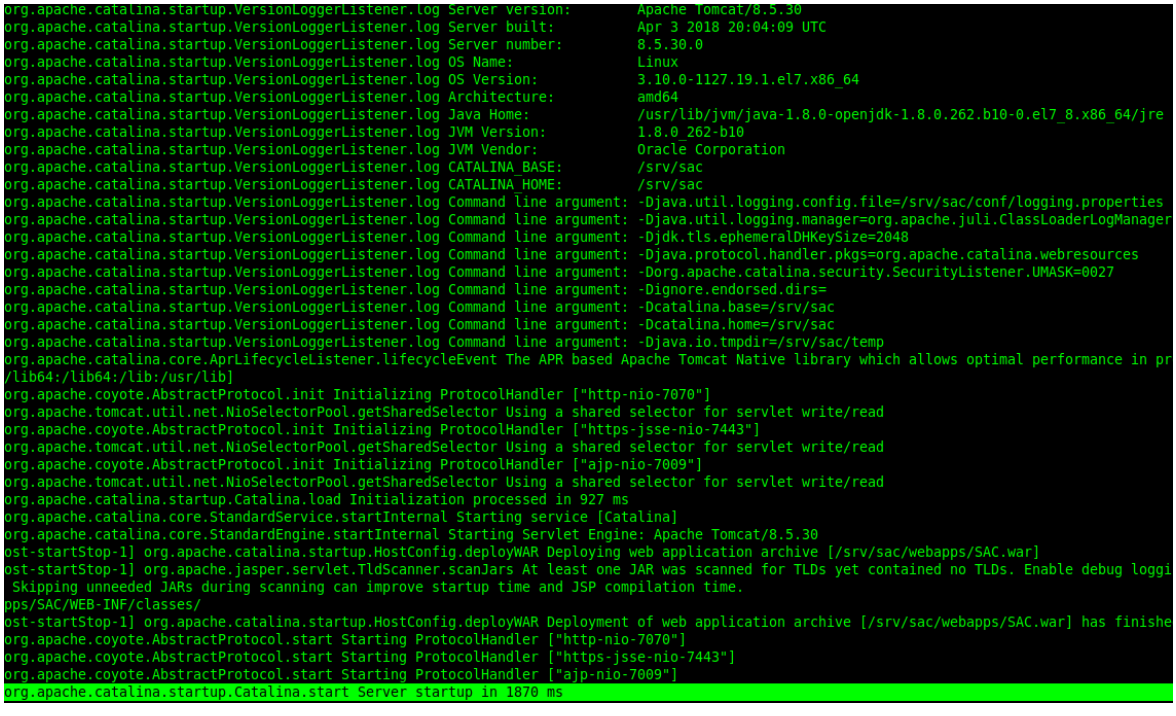

Figura 32: Ejecución Tomcat en el servidor para correr la aplicación

Fuente: Elaboración propia

- Capa de presentación.

Para este proceso instalamos NodeJs versión 14.18, para poder ejecutar la librería de react js, instalamos con el siguiente comando, npm install react@17 react-dom@17. Ver Figura 33.

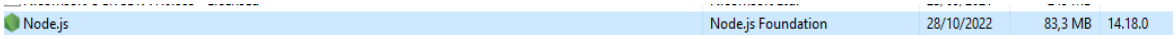

Figura 33: Ejemplo instalando NodeJs.

Fuente: Elaboración propia

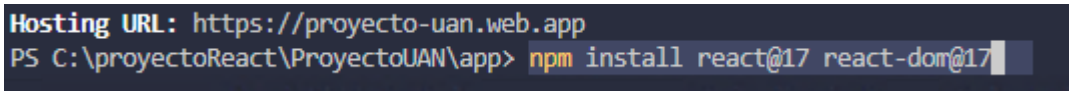

Figura 34: Ejemplo instalando react js en el proyecto.

Fuente: Elaboración propia

Conclusiones

- Se ofreció un servicio automatizado para generar plantillas que permite elegir la modalidad de selección de un contratista del estado para entes territoriales de sexta categoría para el año 2022, de acuerdo a la necesidad.
- Se facilitó un servicio en el cual pueden diligenciar los datos necesarios para elaborar un contrato según la modalidad de selección que se indique. Este servicio permite reducir los tiempos de elaboración entre un 80% y 87%.
- Se desarrolló una plataforma en la cual se pueden estandarizar los formatos usados en el proceso de elección de modalidad de un contratista del estado, con esto se minimiza el error humano a la hora de crear los documentos de un contrato desde el inicio.
- Se ofreció un servicio de reportes, el cual puede ser descargado por el encargado administrativo para validar la cantidad de contratos que se han realizado y saber de qué tipo de modalidad.
- Se elaboró un repositorio donde pueden encontrar las diferentes plantillas que se hayan elaborado, para verificación de datos o edición de parámetros.

Referencias Bibliográficas

[1] ENTIDAD TERRITORIAL (ET). (n.d.). Minciencias. Retrieved March 22, 2022, from https://minciencias.gov.co/glosario/entidad-territorial-et

[2] LEY 617 DE 2000. (2017, October 6). CONGRESO DE LA REPÚBLICA. Retrieved March 22, 2022, from https://www.itrc.gov.co/observatorio/wp-content/uploads/2017/10/Ley_617_2000_ley_presupuesto.pdf

[3] Decreto 1082 de 2015 | DNP. (n.d.). Departamento Nacional de Planeación. Retrieved March 22, 2022, from https://www.dnp.gov.co/normativas/decreto-%C3%BAnico-reglamentario-1082-de-26-de-mayo-2015#subseccion4

[4] Decreto 1082 de 2015 | DNP. (n.d.). Departamento Nacional de Planeación. Retrieved March 22, 2022, from https://www.dnp.gov.co/normativas/decreto-%C3%BAnico-reglamentario-1082-de-26-de-mayo-2015#subseccion5

[5] Contratación administrativa: todo lo que necesitas saber. (n.d.). BeeDIGITAL. Retrieved November 4, 2022, from https://www.beedigital.es/contratacion-laboral/contratacion-administrativa-todo-lo-que-necesitas-saber/

[6] Curso de inducción a los gerentes públicos de la administración colombiana. (n.d.). Curso de inducción a los gerentes públicos de la administración colombiana. Retrieved November 4, 2022, from https://www.funcionpublica.gov.co/eva/gerentes/Modulo4/tema-1/1-concepto.html

[7] *Curso de inducción a los gerentes públicos de la administración colombiana*. (n.d.). Curso de inducción a los gerentes públicos de la administración colombiana. Retrieved November 4, 2022, from https://www.funcionpublica.gov.co/eva/gerentes/Modulo4/tema-2/1-modalidades.html

[8] *Curso de inducción a los gerentes públicos de la administración colombiana*. (n.d.). Curso de inducción a los gerentes públicos de la administración colombiana. Retrieved November 4, 2022, from https://www.funcionpublica.gov.co/eva/gerentes/Modulo4/tema-2/1-modalidades.html

[9] *Curso de inducción a los gerentes públicos de la administración colombiana*. (n.d.). Curso de inducción a los gerentes públicos de la administración colombiana. Retrieved

November 4, 2022, from https://www.funcionpublica.gov.co/eva/gerentes/Modulo4/tema-2/1-modalidades.html

[10] Qué es y para qué sirve Java Spring Boot. (n.d.). Platzi. Retrieved March 22, 2022, from https://platzi.com/blog/que-es-spring-boot/

[11] Robledano, A. (2019, September 24). Qué es MySQL: Características y ventajas. OpenWebinars. Retrieved March 22, 2022, from https://openwebinars.net/blog/que-es-mysql/

[12] Coalla, J. L. (2021, June 23). React | Qué es, para qué sirve y cómo funciona | Descúbrelo todo. Tribalyte Technologies. Retrieved November 4, 2022, from https://tech.tribalyte.eu/blog-que-es-react

[13] Temas Integración ¿Qué es una API de REST? (2020, May 8). Red Hat. Retrieved March 22, 2022, from https://www.redhat.com/es/topics/api/what-is-a-rest-api

Anexos.

Se anexa resultado de la validación de seguridad de la aplicación entregada por el aplicativo OWASP. Nombre: ZAPScanningRepor.pdf