OPTIMIZACIÓN DE LOS TIEMPOS DE CONSULTA A LA BASE DE DATOS EN LA PLATAFORMA INTELCOST

JAVIER ALBERTO ACOSTA LEÓN

UNIVERSIDAD SANTO TOMÁS
DIVISÓN DE INGENIERÍAS
INGENIERÍA DE TELECOMUNICACIONES
BOGOTÁ D.C.
2020
MONOGRAFÍA OPCIÓN DE GRADO

JAVIER ALBERTO ACOSTA LEÓN

MONOGRAFÍA PARA OPTAR POR EL TÍTULO DE INGENIERO DE TELECOMUNICACIONES

RAFAEL ORLANDO CUBILLOS SÁNCHEZ
ING. ELECTRÓNICO

UNIVERSIDAD SANTO TOMÁS
DIVISIÓN DE INGENIERÍAS
INGENIERÍA DE TELECOMUNICACIONES
BOGOTÁ D.C.
2020

Nota de Aceptación

_____

_____

_____

_____

Presidente del Jurado

_____

Jurado

_____

Jurado

Bogotá D.C. 18 marzo 2020

AGRADECIMIENTOS

Gracias a Intelcost por confiar en mi profesionalismo, por brindarme las herramientas para crecer personal y profesionalmente. Agradezco a mis compañeros de trabajo y mis jefes por apoyarme, brindarme la paciencia, darme las fuerzas a desarrollar este proyecto y compartirme el conocimiento de cada uno. Agradezco a la Universidad Santo Tomás por formarme como ser humano y como un profesional integro y competitivo, a los profesores que dedicaron su tiempo y brindaron sus habilidades blandas para formarme como profesional. Finalmente, y no menos importante, agradezco a mi familia que siempre confiaron en mí, estuvieron orgullosos de mi en cada etapa de mi formación y me dieron fuerzas cuando más las necesite tanto en mi carrera como para desarrollar este proyecto.

CONTENIDO

LISTA DE TABLAS

Pág

LISTA DE FIGURAS

GLOSARIO

PRIMERA PALABRA: escribe aquí la definición de la primera palabra ordenada por orden alfabético de forma similar a un diccionario.

FRAMEWORK: es una herramienta que trae consigo funcionalidades estandarizadas con el fin de simplificar a los desarrolladores el uso de funciones o procesos comunes en el ámbito de la programación

ORM: es una técnica utilizada para representar la información proveniente de una base de datos como un objeto y pueda ser manipulado por medio de un framework.

SaaS: Software as a Service, es una forma de prestar servicio de software donde los clientes no requieren la instalación de ningún programa en su equipo, es decir, puede acceder a este por medio de Internet

RESUMEN

Este documento busca optimizar las consultas a la base de datos través del ORM Eloquent integrado con el framework de Laravel, se busca identificar las rutas críticas de navegación por medio de una herramienta de analítica como Google Analytics. En este documento se identifica un problema al momento de descargar un reporte en la plataforma Intelcost y con la implementación del ORM en mención se agiliza la consulta a la base de datos, mejorando la experiencia del usuario y corrigiendo un error al descargar este reporte.

PALABRAS CLAVE: Consultas ágiles, optimización de tiempos, Eloquent, implementar ORM, herramienta analítica.

1. INTRODUCCIÓN

Intelcost es una empresa que presta los servicios de e-procurement como Software as a Services (SaaS), el objetivo es brindar un portafolio de servicios que permita cumplir con todo el proceso de abastecimiento y adquisición de productos y servicios en la compañía. El modelo de negocio de Intelcost es segmentar todo el proceso en distintos módulos para ofrecer a sus clientes la oportunidad de contratar los módulos o procesos requeridos según la necesidad.

Intelcost permite dentro su modelo brindar la visibilidad a los proveedores para que los clientes los tengan en cuenta en sus procesos de licitación, por lo tanto, cuenta también con una plataforma para los proveedores, la cual permite que los proveedores accedan, diligencien toda la información referente a su empresa y apliquen a ofertas y subastas publicadas por los clientes.

Uno de los módulos más robustos y solicitados por los clientes, es el módulo de **ofertas**, este módulo permite conectar proveedores y clientes para realizar las licitaciones. Esto permite que el proveedor pueda aplicar a diferentes ofertas de cada cliente y también permite que el cliente pueda obtener mejores precios y ahorrar recursos a la hora de realizar una compra.

2. OBJETIVOS

2.1 OBJETIVO GENERAL

Optimizar los tiempos de respuesta al consultar la base de datos en el módulo de ofertas de la plataforma Intelcost, herramienta tecnológica para el proceso de procurement.

2.2 OBJETIVOS ESPECÍFICOS

Implementar una herramienta tecnológica que permita establecer las rutas de navegación que habitualmente utilizan los usuarios.

Cuantificar los tiempos de respuesta promedio obtenidos al realizar cada una de las peticiones críticas identificadas en la plataforma.

Desarrollar y aplicar una solución que permita mejorar los tiempos de respuesta de mayor impacto en la velocidad de las peticiones de la plataforma.

3. PLANTEAMIENTO DEL PROBLEMA

3.1 DEFINICIÓN DEL PROBLEMA

Actualmente, al momento de generar el reporte del módulo de ofertas en la plataforma Intelcost, presenta tiempos de respuesta muy altos y en ocasiones no se puede descargar este reporte por exceso de tiempo de espera, esto ocurre debido a que la petición supera los 5 minutos y por seguridad el servidor rechaza la petición. Por otro lado, la consulta no puede simplificarse, debido a que necesariamente tiene que consultar distintas tablas en la base de datos y hacer consultas personalizadas para ciertos clientes, esto incrementa el tiempo de respuesta de la base de datos, afectando la descarga del reporte antes de 5 minutos.

3.2 JUSTIFICACIÓN

El reporte de ofertas permite a los usuarios realizar seguimiento sobre la documentación, el estado de participación de los oferentes y el cronograma de la oferta. Es un documento que permite resumir el proceso que se llevó a cabo para realizar la compra. Varios usuarios reportaron la incidencia presentada afectando sus procesos de seguimiento y auditoria.

Para esta problemática la herramienta de analítica ayuda a identificar donde los usuarios están navegando recurrentemente y el análisis de velocidad desempeño ayuda a determinar donde los usuarios tardan más tiempo. Adicionalmente, luego de identificar la problemática, una de las herramientas que permite agilizar consultas es implementar un ORM, debido a que son diseñados para escatimar los tiempos de consulta a la base de datos.

4. MARCO TEÓRICO

4.1 E-procurement

Inicialmente, procurement se puede definir como el proceso de gestión para la adquisición o compra en el cual una compañía se abastece de diferentes recursos o servicios a través de proveedores. Por otro lado, el concepto de e-procurement se refiere a la automatización e implementación de la tecnología, con el fin de agilizar el proceso de compra mediante la estandarización y optimización de las necesidades de la compañía o también se puede definir como ”una plataforma permite la realización de los procesos de compras a través de internet” (1). Este concepto e-procurement se diferencia del término e-commerce debido a que su enfoque se da en las negociaciones B2B (Business to Business), donde el objetivo es maximizar las posibilidades de compra y minimizar los costos como lo indica Miguel Palacios “de tal forma que un pequeño ahorro en las mismos puede suponer un significativo aumento de los beneficios” (2). De igual manera, se diferencia de e-purchasing debido a que la compra o la adquisición se prevé estratégicamente, determinando el impacto holístico de comprar basados en una economía de escala a nivel departamental.

Dentro de las principales ventajas del e-procurement permite automatizar los procesos de compra y cotización” reducir gastos, concentrar las compras sobre proveedores y catálogos de productos homologados y reducir el tiempo del ciclo de compra.” (2), minimizando el desgaste del recurso humano y tareas de gran envergadura, adicionalmente, permite identificar productos y servicios de una manera objetiva. Sin embargo, es importante determinar los productos y servicios aptos para gestionar, debido a que es posible que el recurso por su naturaleza no sea óptimo o no requiera este proceso.
Para cumplir con el proceso de compra es necesario pasar por una serie de etapas, en Intelcost se representan una o varias etapas como un módulo. A través de estos módulos los clientes pueden ajustar sus procesos de procurement dependiendo su necesidad. A continuación, se compara a grandes rasgos las etapas de la compra y los módulos que Intelcost ha diseñado para cumplir con el proceso de e-procurement.

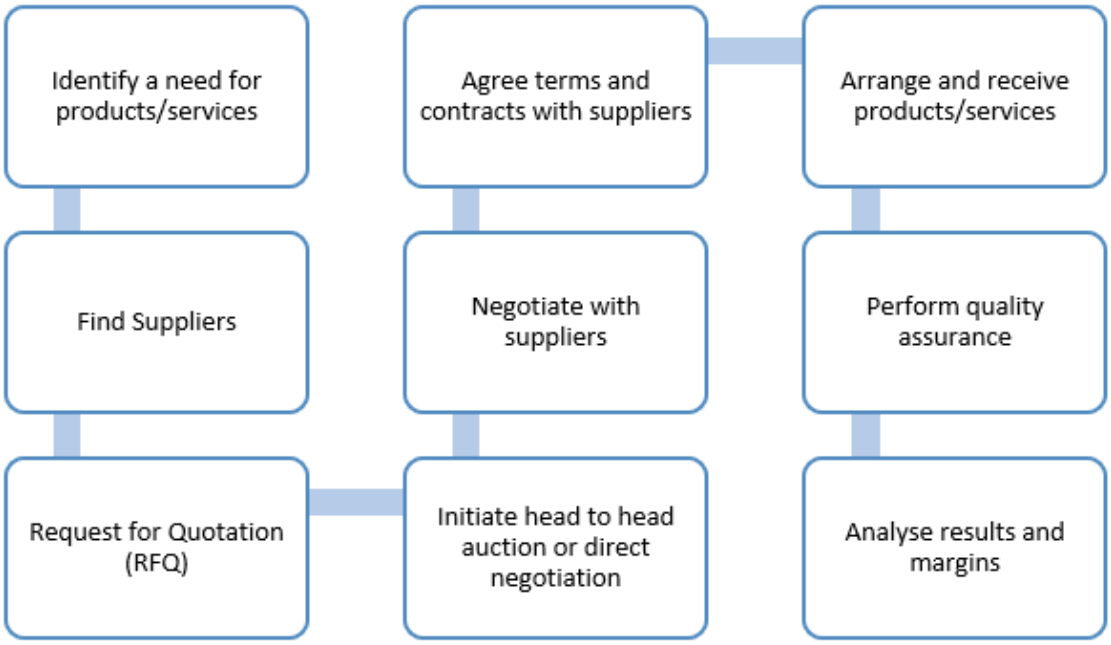


Figura 1. Flujo de compra en un proceso de abastecimiento.[1]

4.1.1 PAA

Plan de adquisición anual de compras. El cliente realiza una planeación anual de abastecimiento, determinando presupuesto anual y los encargados de cada uno de los departamentos o áreas.

4.1.2 Solicitud

El usuario genera una solicitud para cubrir una necesidad, ya sea de un servicio o producto. En este módulo el cliente identifica las necesidades de la compra.

4.1.3 Proveedores

Cuando el proveedor realiza el registro de la información de su empresa. Tanto Intelcost como el cliente verifica esta información a través de este módulo para su invitación a la oferta.

4.1.4 Precalificaciones

El cliente realiza una evaluación sobre ciertos aspectos puntuales a los proveedores, determinando los más aptos para invitarlos posteriormente a una oferta.

[1] *Fuente:https://www.procurehere.com/wp-content/uploads/2018/05/buyer-flow.jpg*

4.1.5 Estudio de mercado

Es una invitación masiva a oferentes con el fin de determinar los precios del mercado y las propuestas de los proveedores.

4.1.6 AOC

Este módulo generalmente es utilizado por empresas públicas, en este módulo el cliente determina si la compra es oportuna y conveniente de acuerdo con el estudio de mercado realizado.

4.1.7 Ofertas (RFQ)

Es una oferta donde los proveedores invitados envían sus propuestas y el cliente evalúa la documentación enviada para determinar el proveedor del producto o servicio

4.1.8 Subastas

Es una oferta donde compiten varios oferentes ajustando sus precios, de tal forma que, al finalizar el tiempo estimado por el comprador, el mejor postor gane la licitación.

4.1.9 Contratos

Posterior a la oferta o la subasta se realiza la contratación del proveedor seleccionado, con cláusulas y condiciones acordadas. En este módulo el cliente tiene la capacidad de dar inicio al contrato, informar avances en la ejecución del contrato y evaluar la ejecución del contrato.

4.1.10 Catálogos Virtuales

Una vez iniciado el contrato, el cliente puede solicitar cantidades parciales de los productos o servicios solicitados dependiendo de lo pactado en el contrato. En esta sección se deben poner las teorías y conceptos que pretendes usar como apoyo para el desarrollo de tu trabajo. Un trabajo investigativo debe contar con abundantes referencias y citas en esta sección.

5 ANÁLISIS GENERAL

5.1 DESCRIPCIÓN GENERAL DEL NEGOCIO

Actualmente, Intelcost presta los servicios de e-procurement como Software as a service a clientes activos como Mansarovar Energy, Metro de Medellín, CaracolTv, entre otros. Dentro del portafolio de servicios ofrece una plataforma modular la cual puede ser personalizada dependiendo de la necesidad del cliente.

Para este estudio inicialmente se determinaron los módulos más contratados por los clientes, se realizó un análisis donde se puede establecer cuántos clientes utilizan cada módulo, esto permite delimitar el objetivo de este estudio con el fin de abordar una solución a los problemas de velocidad puntuales y optimizar el desempeño de la plataforma Intelcost. En la siguiente figura se realiza la comparación de los módulos desarrollados por Intelcost y cada uno de sus clientes activos a cierre del año 2019.

		CLIENTE											TOTAL
		MECL	Masa Stork	GEB	Conconcreto	ODL	MDM	Terpel	TGI	Ocensa	Savia	Akiri	CaracolTv	
MÓDULO	Proveedores	1	1	1	1	1	1	1	1	1	1		1	11
	Proceso/Eventos/RFQ	1	1	1	1	1	1	1	1		1			9
	Precalificación	1	1	1		1		1	1	1			1	8
	Subastas	1		1	1	1	1	1	1		1			8
	Contratos		1	1		1	1							4
	Catálogos Virtuales				1		1					1		3
	PAA - Plan de adquisiciones					1	1							2
	Solicitudes						1							1
	AOC						1							1
	Estudios de Mercado						1							1
	Desempeño							1						1
	Reportes						1							1
	Comité			1										1

Figura 2. Módulos contratados por los clientes. Autor.

Como se puede evidenciar, los módulos más contratados por los clientes son: Proveedores, Procesos/Eventos y Precalificaciones. Con el fin de abordar un módulo objeto de estudio se utilizó la herramienta de Google con el fin de determinar el módulo más utilizado por los usuarios en la plataforma (Ver figura 2).

Figura 2. Tendencia de Módulos contratados por los clientes. Autor.

Luego del informe de Google Analytics se determinó que el módulo objeto de este estudio es Proceso/Eventos (Ofertas). Esto se determinó debido a que es un módulo transversal en proceso de compras y abastecimiento, es contratado por la mayoría de los clientes y es uno de los módulos más utilizados por los clientes según las estadísticas dispuestas por Google Analytics.

5.2 DESCRIPCIÓN DEL MÓDULO OBJETO DE ESTUDIO

Dentro de las etapas de e-procurement es importante recordar que el objetivo principal de este proceso es realizar una compra. Para suplir esta necesidad Intelcost ha diseñado el módulo Proceso/Evento, donde los clientes a través de compradores pueden detallar las necesidades y requerimientos que la compañía tiene. Luego, el comprador puede realizar una oferta a los proveedores, en este módulo pueden enviar la información solicitada por el comprador. El módulo de ofertas genera un valor para la compañía, agrupando las necesidades de la compra y las propuestas de los proveedores.

5.2.1 Análisis de flujos en el módulo de estudio

Inicialmente se identificaron las rutas de navegación que el usuario puede hacer en el módulo de Proceso/Evento, se puede observar el proceso en el siguiente diagrama.

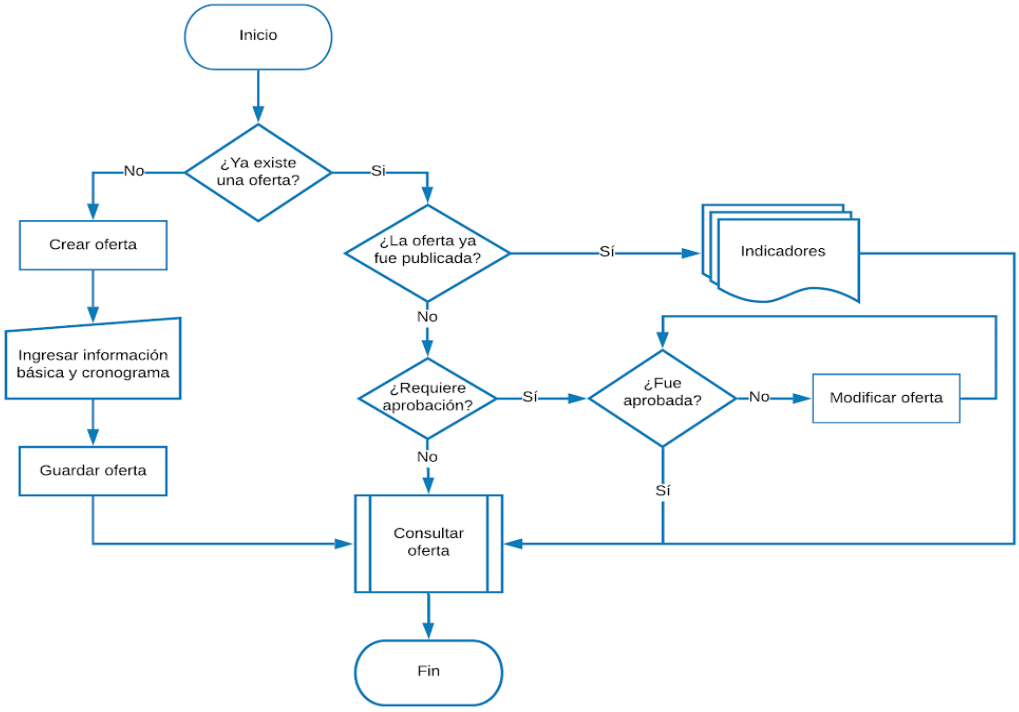


Figura 3.

Figura 3. Diagrama de flujo módulo de ofertas. Autor.

Luego de crear la oferta el comprador o el encargado del proceso puede generar un reporte global de todas las ofertas, buscar una oferta en específico (esto lo puede realizar ingresando filtros o seleccionándola en la lista de las 10 últimas creadas). Luego, si la oferta está activa puede modificarla y si desea puede publicarla. Una vez publicada los proveedores son notificados para que participen en la oferta enviado sus propuestas. Al finalizar el tiempo de publicación, el comprador, los evaluadores o responsable de la oferta pueden realizar la evaluación de cada una de las propuestas enviadas, rechazando o aprobando cada uno de los proveedores. Finalmente, luego de la evaluación el comprador puede elegir uno o más proveedores para adjudicar la oferta y continuar el proceso en el módulo de contratos.

Es importante destacar que este es un flujo normal y básico del módulo, debido a que actualmente cada cliente tiene pequeñas modificaciones dependiendo de la necesidad, en muchos casos se requiere una aprobación antes de publicar un oferta, se puede inactivar o cancelar una oferta cuando no se quiere realizar la compra, también se puede declarar desierta la oferta si ningún proveedor cumple con los requerimientos establecidos por el cliente o el presupuesto no se ajusta a lo que demanda el mercado. A continuación, se puede observar en la Figura 4 el diagrama de flujo básico del módulo de ofertas.

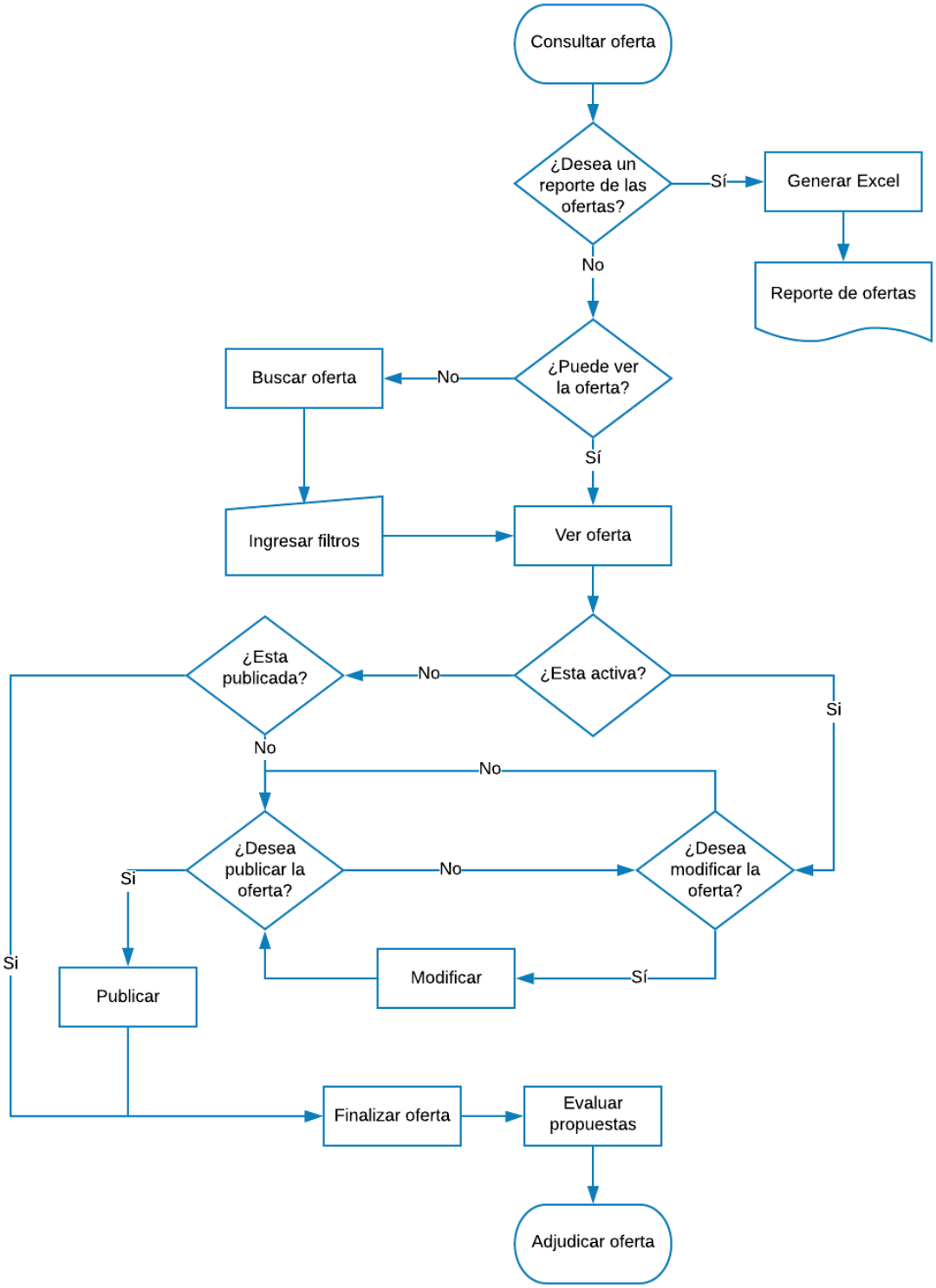


Figura 4. Diagrama de flujo consultar y publicar una oferta. Autor.

6. HERRAMIENTA ANALYTICS

6.1 ANÁLISIS

Luego de establecer las rutas de navegación en el módulo de Ofertas, como se mencionó anteriormente se utilizó la herramienta de Google Analytics debido a sus ventajas frente a otras herramientas detalladas a continuación:

Figura 5. Cuadro comparativo herramientas de analítica web. Autor.

Inicialmente para determinar la herramienta de analítica web, se tuvieron en cuenta los aspectos mencionados en la imagen anterior, tales como:
Tiempo real: Esto nos permite monitorear el tráfico en tiempo real, este fue un factor decisivo al momento de elegir en un primer filtro a estas tres herramientas, debido a que la mayoría no realizan estos análisis en tiempo real. Es importante que se obtenga la información con un bajo tiempo de respuesta, ya que en ocasiones se evidencia que si la navegación del usuario en la plataforma es rápida la herramienta de analítica no podrá realizar el registro de las páginas a las que navegan los usuarios de forma adecuada.

Informes personalizados: En Google Analytics solo se permiten filtrar resultados comparados con el total de resultados. Si se desea generar un informe sobre uno o dos datos particulares y estos no cumplen con al menos el 50% del total de resultados, el informe no se visualizará bien y por consiguiente no serán resultados comparables. Por otro lado, Kissmetrics permite personalizar los informes de tal manera que el usuario pueda generarlos a su necesidad. Finalmente, Matomo permite generar informes de manera limitada, no permite utilizar más de 6 datos para generar el informe en su versión freemium.
Detalle flujo de tráfico: En este caso Matomo permite ver las rutas de navegabilidad a través de mapas de calor en las vistas que más frecuentan los usuarios, esta funcionalidad permite ver el detalle del flujo de tráfico o la frecuencia de los usuarios al utilizar un botón o un elemento de la vista. Esto facilita de manera exponencial el análisis por cada vista.
Segmentación: Permite dimensionar los resultados dependiendo de la necesidad del informe. Variables como ubicación, páginas visitadas, dispositivo móvil utilizado, entre otros. Revisando Google Analytics permite hasta 1000 segmentos en toda la herramienta, Matomo en su capa gratuita permite hasta un máximo de 10 segmentos y finalmente Kissmetrics no tiene ninguna limitación.
Servidor Remoto: Esta característica es una de las más importantes, debido a que si la herramienta se instala en un servidor propio se tiene la responsabilidad de destinar recursos de procesamiento y almacenamiento. Por tal razón, Google Analytics y Matomo son una excelente opción.
Retención de información: Esta característica es importante a la hora de elegir la herramienta a utilizar, debido a que define el tiempo que las estadísticas van a estar disponibles. Tanto para Google Analytics como para Kissmetrics esta funcionalidad es por tiempo ilimitado, es decir que los datos obtenidos no se borrarán o no estarán disponible cada cierto tiempo, a diferencias de Matomo que su tiempo máximo de retención de información es de 6 meses en capa gratuita y un año contratando los servicios.

Como se mencionó anteriormente y luego de analizar las comparativas se optó por utilizar Google Analytics, debido a que ofrece muchas de las funcionalidades como: Ver estadísticas tiempo real, los informes personalizados son limitados, pero manualmente se pueden realizar filtros para obtener la información necesaria, la segmentación es un factor que para el alcance del proyecto se ajusta a la necesidad. Por otro lado, el servidor remoto ayuda a que el rendimiento de la plataforma no se va afectado por la implementación de esta herramienta. Adicionalmente, la información puede ser retenida sin límite de tiempo y finalmente

Google ofrece esta herramienta sin costo, lo cual no incurre en un gasto para la compañía.

6.2 IMPLEMENTACIÓN

Google Analytics es la herramienta que permitirá identificar cuantos usuarios están activos en la plataforma y las rutas de navegación, para implementar esta herramienta Intelcost solicitó a Google una cuenta para administrar las propiedades, estas propiedades se pueden entender como diferentes plataformas para un cliente o diferentes entornos de desarrollo, a su vez, cada propiedad puede contener varias vistas lo cual corresponde a un subdominio por propiedad.

Para implementar Google Analytics a la plataforma Intelcost fue necesario configurar la propiedad como se muestra en la siguiente imagen:

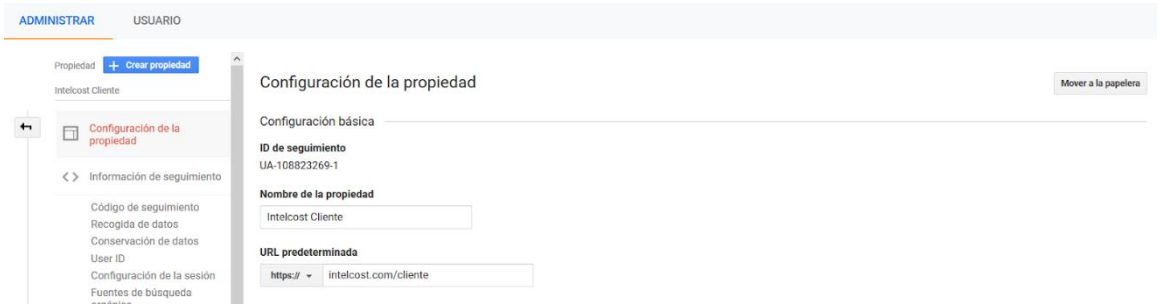


Figura 6. Configuración de la propiedad en Google Analytics. Autor.

Una vez configurada la propiedad en Google Analytics, se debe implementar el siguiente script en el código dentro de la etiqueta <HEAD> como se muestra a continuación:

```
<!-- Global site tag (gtag.js) - Google Analytics -->

<script async src="https://www.googletagmanager.com/gtag/js?id=UA-108823269-1"></script>

<script>

  window.dataLayer = window.dataLayer || [];
  function gtag(){dataLayer.push(arguments);}
  gtag('js', new Date());
  gtag('config', 'UA-108823269-1');

</script>
```

Figura 7. Implementación de script de seguimiento de Google Analytics en código. Autor.

Este script permite que cada vez que se cargue el <HEAD> inicie una nueva conexión a la propiedad configurada anteriormente. Esto permite determinar cuántos usuarios están conectados, debido a que cada vez que un usuario ingresa

a la plataforma el <HEAD> se ejecuta en el navegador y envía la petición al servidor de Google.

Una vez realizado el procedimiento anterior, se garantiza que se tiene una conexión con la herramienta como se muestra en la siguiente imagen:

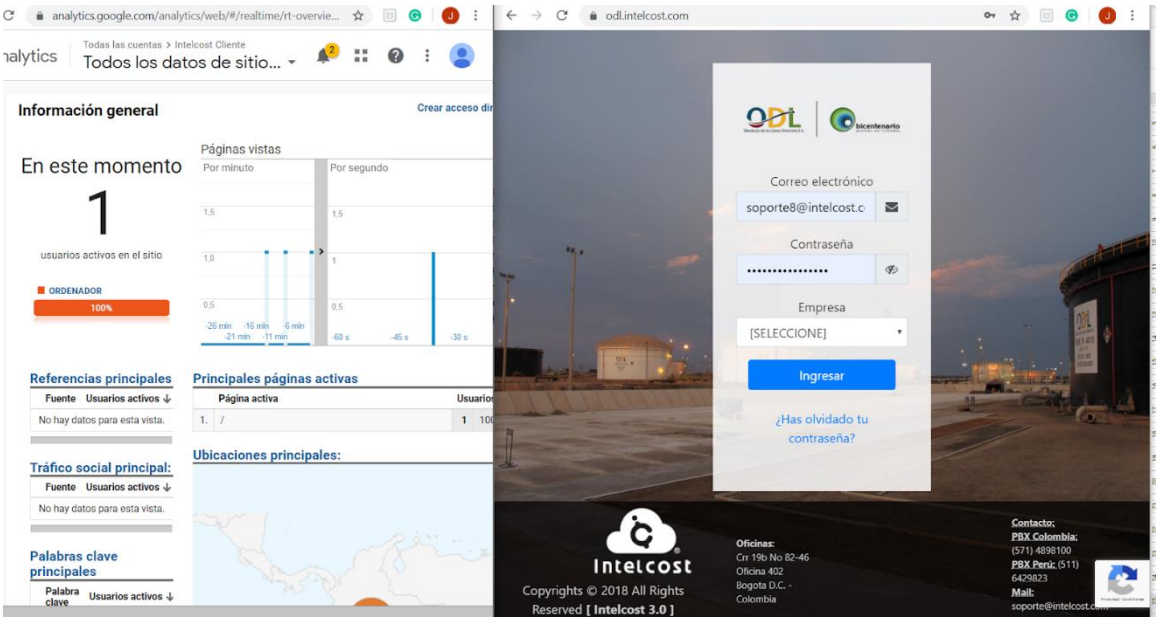


Figura 8. Evidencia de conexión de la herramienta Google Analytics con la plataforma. Autor.

Intelcost fue implementado bajo la premisa Single Page Application, lo cual permite cargar dinámicamente el contenido necesario a los usuarios sin necesidad de cargar toda la página. Al inicio de implementar la herramienta de Google Analytics todas las visitas a diferentes vistas se interpretaban como la URL “/”, tal y como se muestra en la anterior imagen, por lo tanto, fue necesario modificar los parámetros que se enviaban a Google Analytics cuando se carga un contenido en específico.

Para implementar la funcionalidad se utilizó la función que carga el contenido en específico y consultar el objeto sessionStorage, el cual almacena información en el navegador de cada usuario.

```
947    var cargarContenido = function(pagina, titulo, params){
948
949        sessionStorage.setItem("navegacionActual", pagina);
950
```

Figura 9. Función cargarContenido. Autor.

El anterior fragmento de código muestra la función que carga el contenido y como la variable navegacionActual almacenada en el sessionStorage es modificada. Esta variable cambia cada vez que el usuario da clic en un botón y ejecuta la función cargarContenido como se muestra en la siguiente imagen:

```
<button class="btn btn-warning" onclick="cargarContenido('subastas/subastas','Subastas')">
Volver a mis subastas</button>
```

Figura 10. Ejemplo de un botón utilizando la función cargarContenido. Autor.

Por lo tanto, se creó la variable actual la cual toma el valor almacenado en la variable navegacionActual del objeto sessionStorage y envía este valor junto con el parámetro 0 a la función sendDataGoogleAnalytics.

```
958         let actual=sessionStorage.getItem("navegacionActual");
959         sendDataGoogleAnalytics(actual,0);
```

Figura 11. Envío de parámetros a la función sendDataGoogleAnalytics. Autor.

En el transcurso de la implementación de la herramienta se vio la necesidad de identificar exactamente dónde se encuentra el usuario en cada instante, la anterior funcionalidad permite establecer la ubicación del usuario en la plataforma solo cuando interactuaba con un botón. En consecuencia, se implementa la siguiente solución a los demás elementos con los cuales el usuario puede interactuar.

```
<li class="col col-md-3 col-sm-4 col-xs-4 classGoogle " googleAnalytics="visita_informativa" href="#tab_2"
data-toggle="tab">
```

Figura 12. Ejemplo de llamado de clases y atributos para uso de Google Analytics. Autor.

En la anterior imagen, se implementa la herramienta de seguimiento para un ítem de una lista, solo es necesario incluir la clase classGoogle, la cual se muestra dentro del recuadro rojo de la imagen y definir el nombre del parámetro googleAnalytics como se muestra en el recuadro naranja de la imagen para enviar la url deseada a Google Analytics. Dicho lo anterior, es necesario referenciar la función al final del archivo para poder enviar los datos de la vista al JavaScript. Esta referencia o llamado se puede observar en la siguiente imagen:

```
<script>

        getDataJsGoogleAnalytics();

</script>
```

Figura 13. Llamada a la función getDataJsGoogleAnalytics. Autor.

A continuación, se define la función getDataJsGoogleAnalytics, la cual toma cada elemento que contenga la clase classGoogle y agrega al evento clic la función

sendDataGoogleAnalytics con el nombre definido en el atributo googleAnalytics y el parámetro 1.

```
//Función para obtener atributo de JS y enviarlos a Google Analytics
var getDataJsGoogleAnalytics = function(){
    $(this).each(function() {
        $(".classGoogle").on('click', function(event){
            var route= $(this).attr('googleAnalytics');
            sendDataGoogleAnalytics(route,1);
        });
    });
}
```

Figura 14. Función getDataJsGoogleAnalytics. Autor.

Posteriormente, se describe la función sendDataGoogleAnalytics, la cual dependiendo del segundo parámetro (0 o 1), construirá la ruta o no. Es decir, si la función fue llamada desde cargarContenido no deberá modificar la URL y esta se enviará a Google Analytics. Ejemplo: "/ofertas/listado_ofertas". Mientras que, si la ruta proviene de un elemento distinto a un botón, es necesario construir la ruta con la variable de sesión navegacionActual. Ejemplo: De la URL "visita_informativa" la función permite modificar la URL a "/ofertas/detalle/visita_informativa".
Finalmente, luego de obtener la URL, en la función ga('set','page','nombreRuta') se define la URL en el campo nombreRuta para definir los datos a enviar a Google Analytics, con la función ga ('send', 'pageview'); se envía la solicitud a la herramienta de analitica. En resumen, se puede observar la función en la siguiente imagen:

```
var sendDataGoogleAnalytics = function($route,$source){
    //Si la petición proviene de CargarContenido
    if($source==0){
        ga('set','page',route);
        ga('send' , 'pageview');

    //Se construye ruta si proviene de un elemento distinto a un botón
    }else if($source==1){
        let url=sessionStorage.getItem("navegacionActual")+'/'+route;
        ga('set','page',url);
        ga('send' , 'pageview');
    }else{
        console.log('Origen desconocido Google Analitics');
    }
}
```

Figura 15. Función sendDataGoogleAnalytics. Autor.

Finalmente, se muestra el resultado de la implementación de la herramienta, en la siguiente imagen se puede ver que los datos se están enviando correctamente al servidor de Google Analytics.

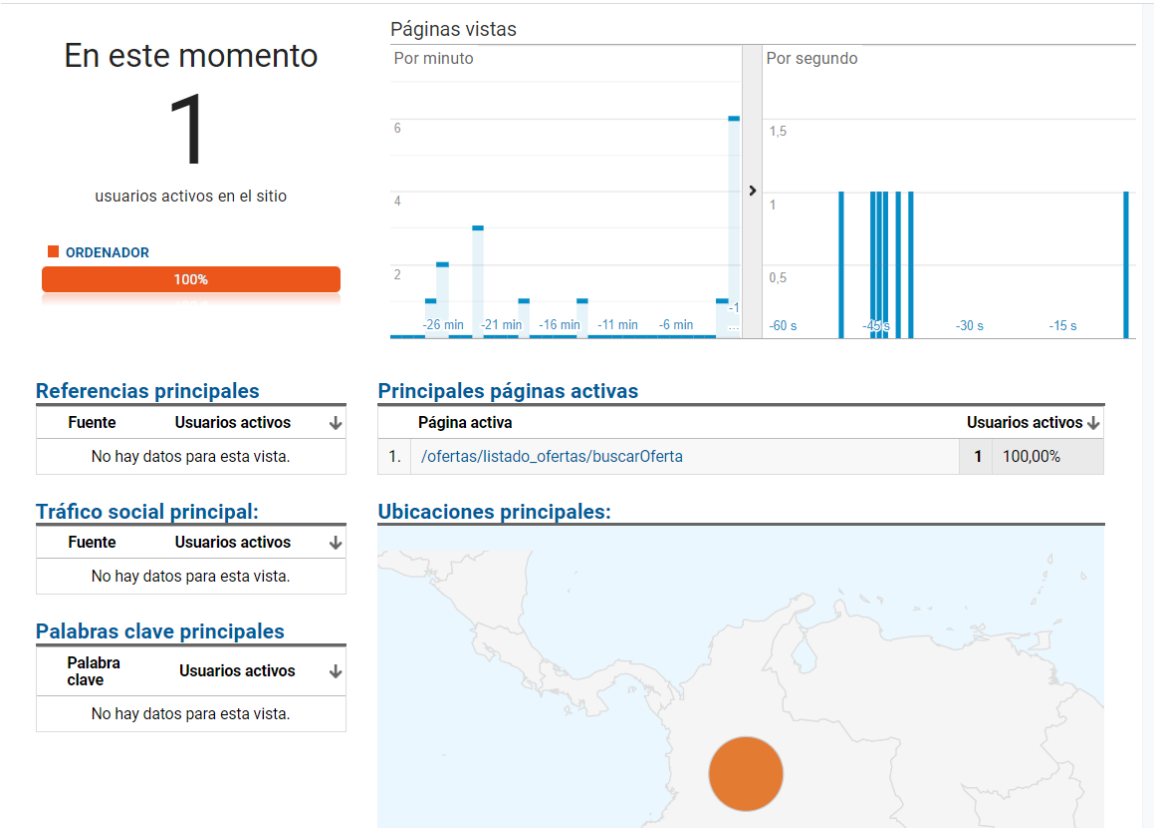


Figura 16. Resultado de seguimiento de rutas de navegación Autor.

7 ANÁLISIS DE RUTAS DE NAVEGACIÓN

Luego de implementar Google Analytics como herramienta de analítica web y configurar las respectivas vistas, se tomó un periodo de recolección de datos de tres meses de testeo, se tomaron únicamente los resultados mayores a 100 visitas para minimizar los resultados y enfocar el estudio en problemáticas globales y no particulares. A partir de la anterior implementación se obtuvieron las siguientes estadísticas en Google Analytics:

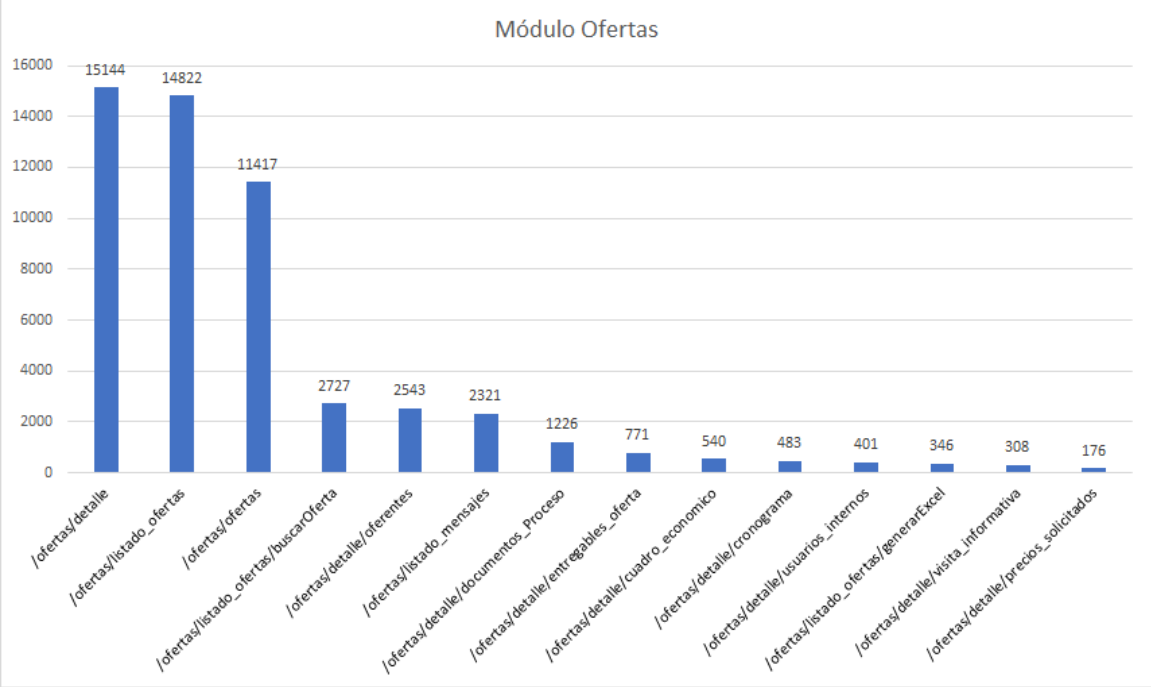


Figura 17. Rutas de navegación frecuentes por los usuarios en el módulo de ofertas. Autor.

En esta imagen se logra apreciar las rutas de navegación más comunes que los usuarios utilizan en el módulo de ofertas, gracias al seguimiento realizado estas estadísticas servirán de base para constatar con el análisis de velocidad, y posteriormente determinar en donde el usuario está presentando mayor lentitud, con el fin de proponer una solución para optimizar las respuestas del servidor y mejorar la experiencia del usuario.

7.1 Descripción de las rutas definidas

A continuación, se especifica cada una de las rutas con su respectiva descripción e imagen de referencia:

7.1.1 /ofertas/ofertas

Dentro del módulo de ofertas se encuentra dos modalidades, ofertas cerradas o públicas. Cada una contiene las opciones de crear, copiar, consultar o ver indicadores.

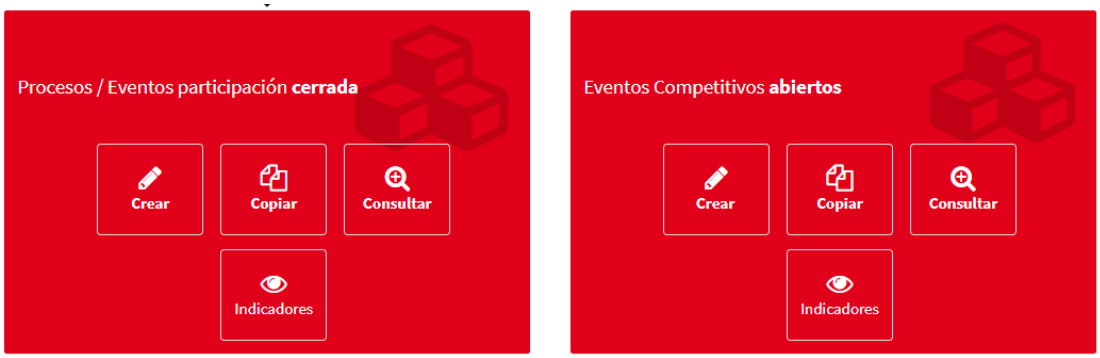


Figura 18. Módulo de ofertas. Autor.

7.1.2 /ofertas/listado_ofertas

Una vez el usuario quiera consultar una oferta se listarán por defecto las últimas ofertas. En esta vista se muestra la información básica sobre las ofertas como: Id, objeto, descripción, fecha de incio, fecha de de cierre, estado, responsable del evento y la opción de ver detalle.

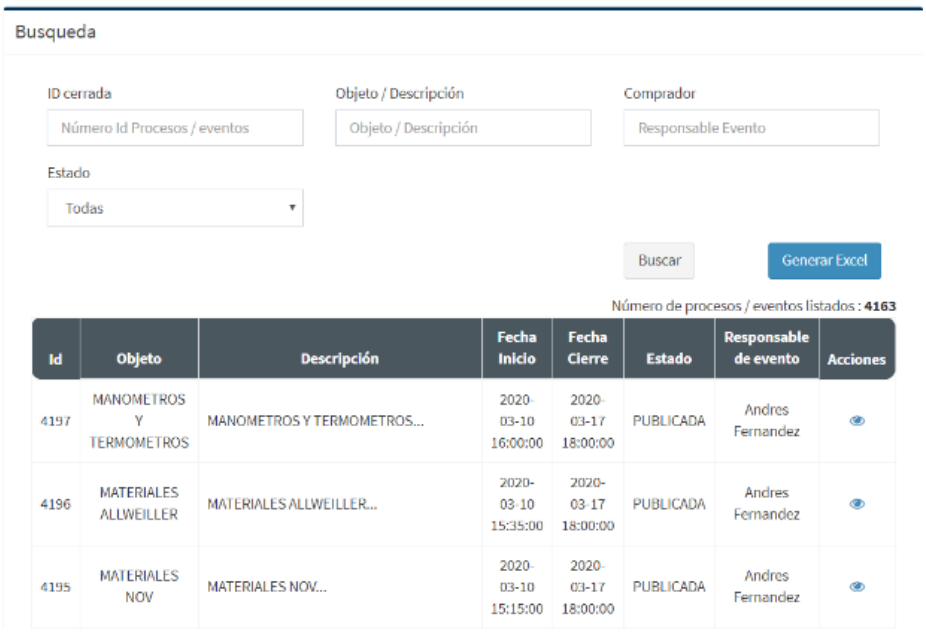


Figura 19. Listado de ofertas. Autor.

7.1.3 /ofertas/listado_ofertas/buscarOferta

El usuario también podrá buscar un proceso en especial indicando el id de la oferta, objeto, descripción, comprador o estado.

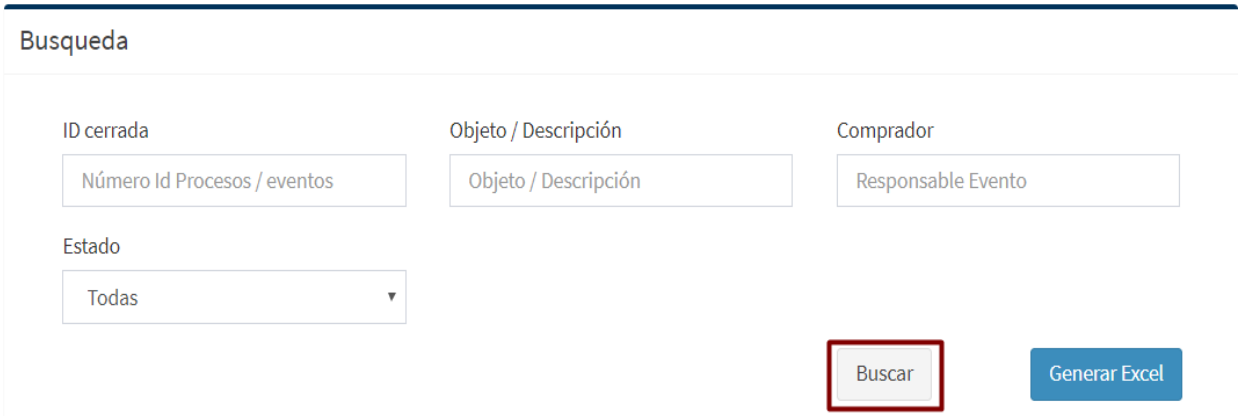


Figura 20. Buscar oferta. Autor.

7.1.4 /ofertas/listado_ofertas/generarExcel

El usuario podrá generar un reporte en excel donde se muestra la información de todas las ofertas que se están listando. También puede realizar filtros para generar el reporte personalizado.

Figura 21. Generar Excel. Autor.

7.1.5 /ofertas/detalle

Cuando el usuario desee ver el detalle de una oferta, puede visualizar la información en la opción consultar, se representa como un ojo en la columna de acciones.

Id	Objeto	Descripción	Fecha Inicio	Fecha Cierre	Estado	Responsable de evento	Acciones
4197	MANOMETROS Y TERMOMETROS	MANOMETROS Y TERMOMETROS...	2020-03-10 16:00:00	2020-03-17 18:00:00	PUBLICADA	Andres Fernandez	

Figura 22. Ver detalle de oferta. Autor.

8. ANÁLISIS DE VELOCIDAD

Antes de realizar cualquier prueba se revisó la conexión a internet con el fin de garantizar que los datos obtenidos sean más confiables y no se vean alterados en gran medida por la velocidad de la red. Las velocidades desde donde se realizaron las pruebas se aproximan a una velocidad de 340 Mbps de descarga y 170 Mbps de subida en promedio como se muestra en la siguiente imagen:

Figura 23. Test de velocidad[2]

Posterior a esta revisión y luego de establecer las rutas de navegación que los usuarios frecuentan en el módulo de ofertas, se realizaron las pruebas con el fin de determinar los tiempos de ejecución que se toman la mayoría de las peticiones por cada cliente que tiene contrata este módulo.

Para la ejecución de estas pruebas se utilizó la herramienta de DevTools de Google Chrome, con el fin de determinar el tiempo que el servidor responde a cada petición realizada desde este módulo. A continuación, se puede observar un ejemplo de la respuesta de la herramienta DevTools:

[2] *Fuente: http://www.speedtest.com/*

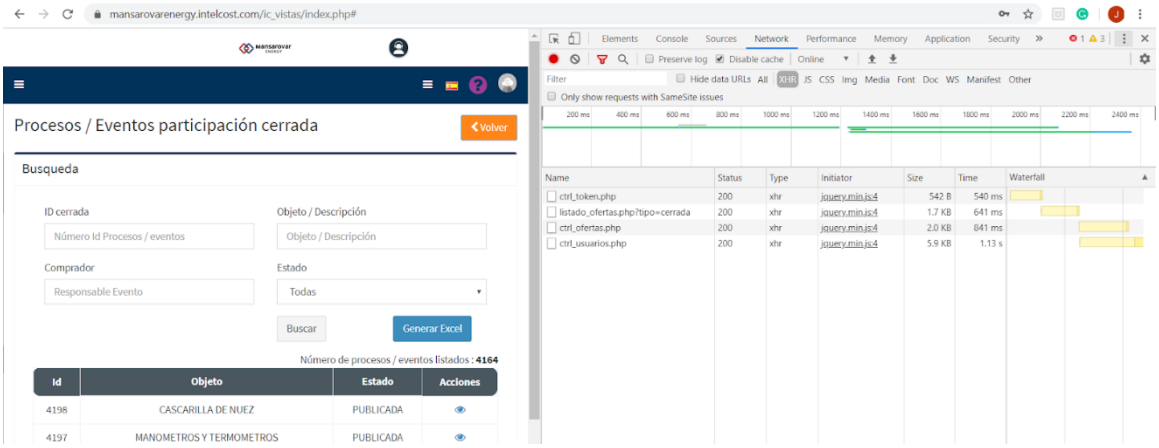


Figura 24. Herramienta DevTools. Autor.

Como se puede apreciar en la siguiente imagen, la respuesta de cada petición tarda un tiempo distinto en ser recibida por el cliente. es en este punto en el cual se determinarán en donde los usuarios pueden presentar mayores dificultades o mayores tiempos de respuesta por parte del servidor. El cuadro naranja indica el nombre de la petición y el cuadro rojo el tiempo que el servidor se demora en dar respuesta a la petición.

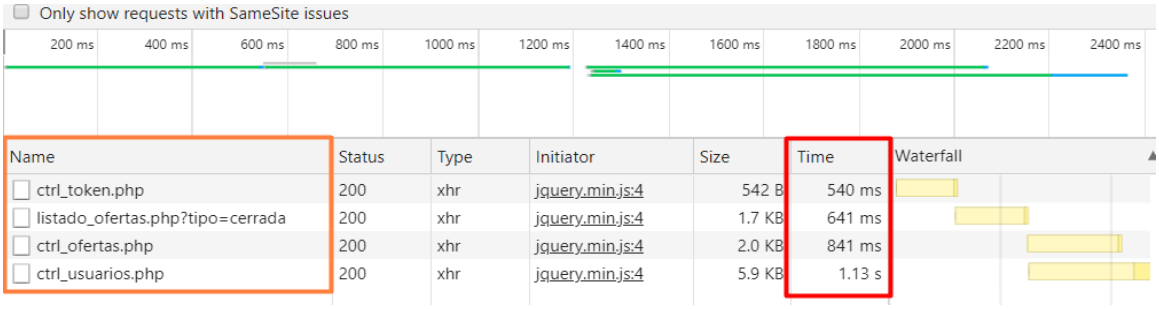


Figura 25. Peticiones al servidor y sus correspondientes tiempos de ejecución. Autor.

Luego de realizar el análisis de rutas se identificaron las peticiones que consultan las bases de datos, con el fin de utilizar mecanismo de optimización al realizar estas consultas. A continuación, se muestra el consolidado de los tiempos de demora promedio en realizar las peticiones al servidor:

Tabla 1.Tiempos de demora en cada petición realizada al servidor.

Ruta/Descripción	***Tiempo (s)***
/ofertas/ofertas	Entre 0,463 a 0.954
/ofertas/listado_ofertas	0,148
/ofertas/listado_ofertas/buscarOferta	0.618
/ofertas/listado_ofertas/generarExcel	Mayor a 5 minutos
/ofertas/detalle	0,432
/ofertas/ listado_mensajes	0,212

Fuente: Autor.

Se puede identificar que las peticiones realizadas al servidor tienen una duración promedio de 500 segundos, excepto para la ruta /ofertas/listado_ofertas/generarExcel, la cual excede los 5 minutos en responder el servidor. Cabe resaltar que por seguridad Intelcost no permite peticiones mayores a 5 minutos, por lo tanto, este reporte no se puede generar por time out.

9 PROPUESTA DE MEJORA VELOCIDAD

9.1 ANÁLISIS

Inicialmente se quería determinar una relación entre las rutas de navegación y los tiempos de respuesta en el servidor, con el fin de generar estadísticas y ajustar los objetivos de la compañía en optimizar estos tiempos. Luego de la fase de adquisición de información, comenzaron los usuarios a reportar errores con el reporte mencionado en el análisis de velocidad, por lo tanto, se tomó la iniciativa de incluir en este proyecto la optimización del reporte.

Para comenzar, se realizaron pruebas con varios clientes y se determinó que cuando el cliente ha realizado más de 2000 ofertas, el reporte no se puede generar debido a que supera el tiempo máximo de 5 minutos en generarse. A continuación, se muestra la petición en distintos clientes.

9.1.1 Metro de Medellín (403 procesos)

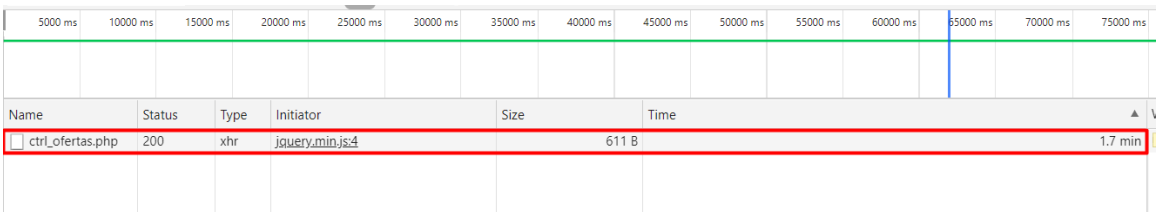


Figura 26. Tiempo de respuesta de la petición en Metro de Medellín. Autor.

9.1.2 Terpel (3367 procesos)

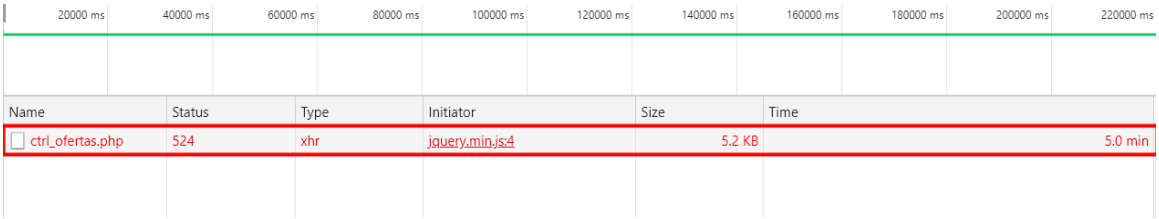


Figura 27. Tiempo de respuesta de la petición en Terpel. Autor.

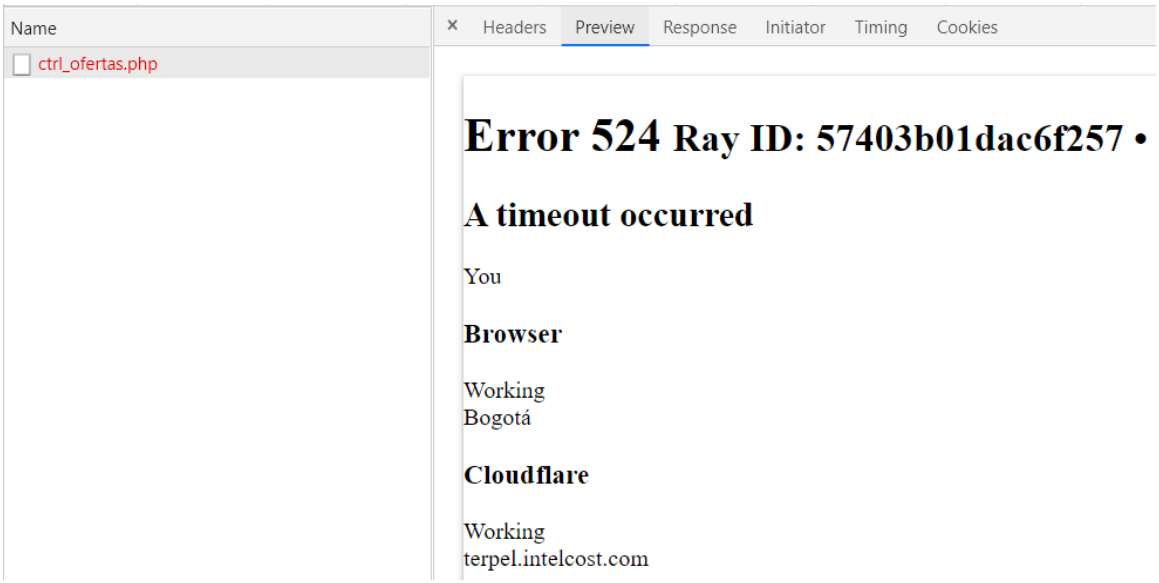


Figura 28. Información recibida de la petición en Terpel. Autor.

9.1.3 Mansarovar (4169 procesos)

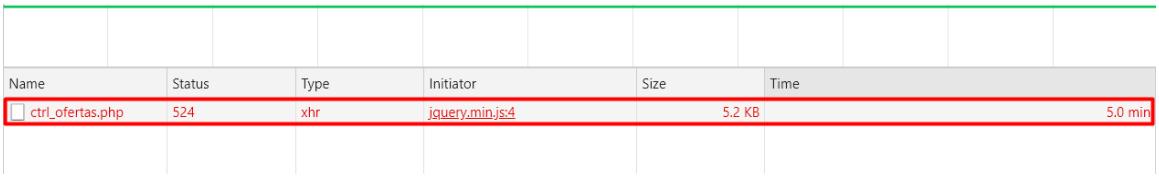


Figura 29. Tiempo de respuesta de la petición en Mansarovar.

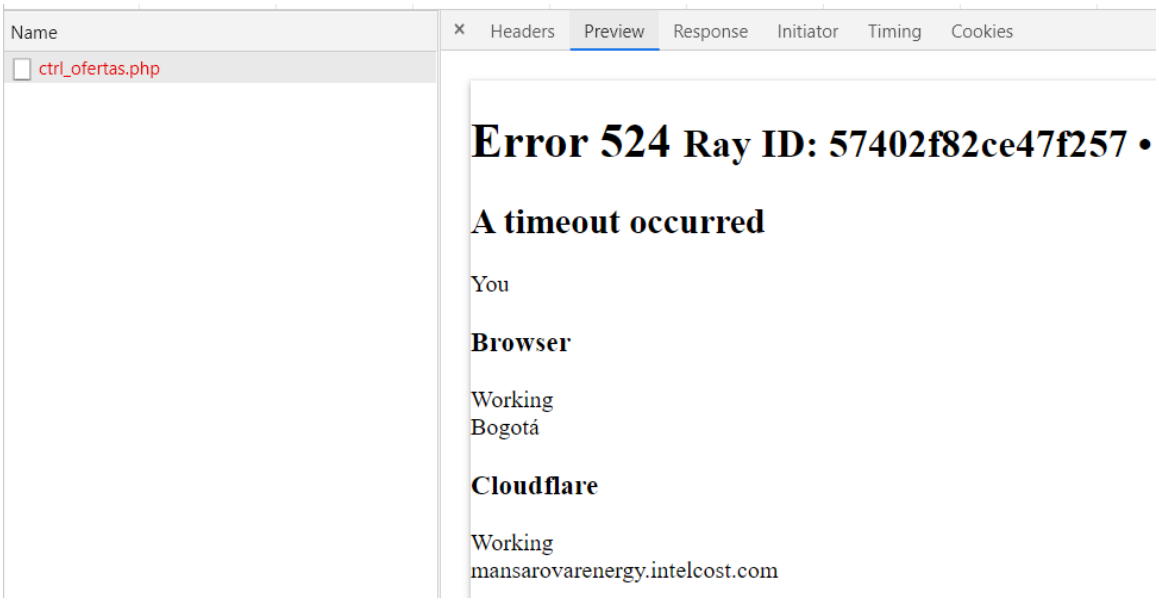


Figura 30. Información recibida de la petición en Mansarovar. Autor.

9.1.4 Masa Stork (2353 procesos)

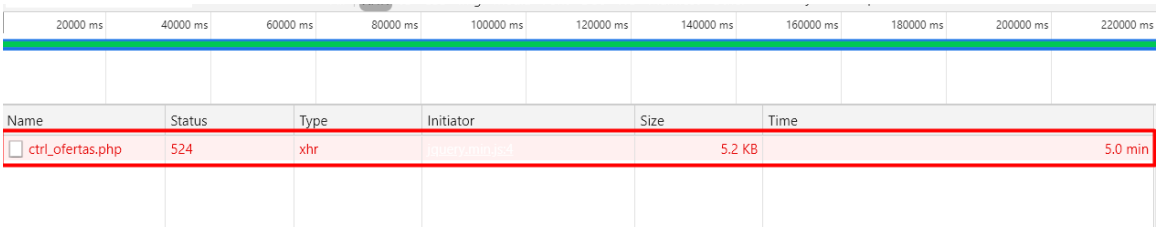


Figura 31. Tiempo de respuesta de la petición en Masa Stork. Autor.

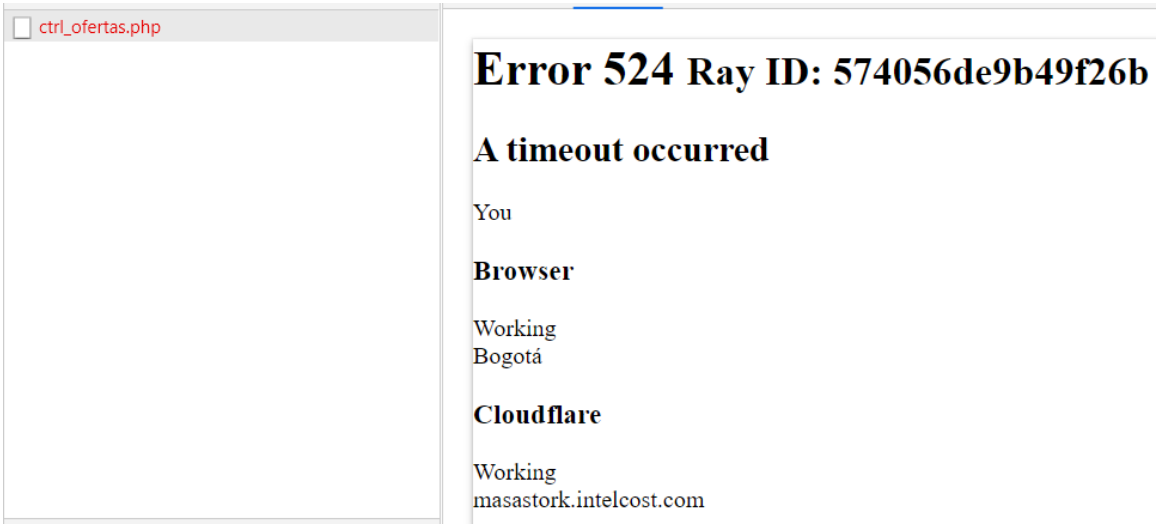


Figura 32. Información recibida de la petición en Masa Stork. Autor.

9.2 IMPLEMENTACIÓN

Como se puede observar, los reportes en la mayoría de los clientes que exceden las 2000 ofertas no se pueden generar por tiempo excedido en realizar la petición. Por lo tanto, como la plataforma Intelcost está desarrollada en PHP y está en proceso de migración a Laravel, se decidió utilizar el ORM Eloquent para optimizar los tiempos al realizar esta petición a la base de datos.

Inicialmente se define el modelo, el cual tendrá la información de la tabla a consultar en la base de datos, es decir, en este archivo se indica: la base de datos a consultar, la tabla, las columnas y las relaciones con otras tablas. A continuación, se describe cómo fue la implementación de los modelos:
Inicialmente, se realizó la referencia de las bibliotecas y librerías a utilizar, estas librerías deben estar incluidas en el archivo composer.json archivo el cual contiene todos los paquetes que se requieren.

```php
<?php
//Alias del archivo
namespace Models\Ofertas;

//Librerias y bibliotecas a utilizar
use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\SoftDeletes;
use Illuminate\Support\Facades\DB;
```

Figura 33. Bibliotecas y librerías por utilizar en cada modelo. Autor.

Luego, se crea una clase la cual debe tener el mismo nombre del archivo y el nombre de la tabla correspondiente en la base de datos. La variable protegida table hace referencia al nombre de la tabla, la variable fillable contiene el nombre las columnas de esta tabla. Es importante resaltar que los nombres de estas variables no se pueden modificar, debido a que Eloquent no reconocería el nombre de la tabla y sus respectivas columnas.

```php
class Oferta extends Model
{
    //Nombre de la tabla en BD
    protected $table = "ofertas";
    //Campos de la base de datos
    protected $fillable = [
        "id",
        "old_id",
        "seq_id",
        "solpeds_relacionadas",
        "tipo",
        "id_cliente",
        "objeto",
        "actividad",
        "descripcion",
        "moneda",
        .
        .
        .
```

Figura 34. Descripción de la tabla en cada modelo. Autor.

Una vez se hayan nombrado todas las columnas de la tabla, se realizan las relaciones con otras tablas, las cuales se definen por medio de funciones públicas, dentro de estas funciones se hace un llamado a las funciones hasOne o hasMany, las cuales determinan si es una relación uno a uno o uno a muchos. Para utilizar alguna de estas funciones, se debe indicar dentro de los parámetros el archivo de donde se encuentra la tabla y la llave foránea. Adicionalmente, se puede definir una condición para esta relación tal y como se muestra en la función infoRequisición, donde luego de utilizar la función hasOne y se utiliza la función where para indicar una condición. En este caso se define que el campo estado debe ser diferente a 'cancelado' para cada registro encontrado.

```
        "fecha_autorizacion_ajudicacion",
        "modalidad_seleccion"
    ];

    public function infoRequisicion(){
        return $this->hasOne('Models\Requisiciones\Requisicion', 'id', 'id_requisicion')
                ->where('estado','!=','cancelada');
    }
    public function infoAoc(){
        return $this->hasOne('Models\Aoc\Aoc', 'id_estudio_mercado', 'id')
                ->where('estado','!=','cancelado');
    }
    public function infoContrato(){
        return $this->hasOne('Models\Contratos\Contratos', 'oferta_id', 'id')
                ->where('estado','!=','cancelada');
    }
    public function infoUsuarioCreacion(){
        return $this->hasOne('Models\Usuarios\Usuario', 'id', 'usuario_creacion');
    }
    public function infoUsuarioInternos(){
        return $this->hasMany('Models\Usuarios\OfertaUsuariosInternos', 'id_oferta', 'id');
    }
    public function infoAocRfq(){
        return $this->hasOne('Models\Aoc\Aoc', 'id_aoc', 'id_aoc');
    }
    public function infoProveInviOfertas(){
        return $this->hasMany('Models\Ofertas\InformacionProveedoresInvitados', 'id_oferta', 'id');
    }
    public function actividades(){
        return $this->hasOne('Models\Maestras\Actividades\MstActividades', 'producid', 'actividad');
    }
}
```

Figura 35. Bibliotecas y librerías por utilizar en cada modelo. Autor.

Luego de describir la estructura del archivo que hace referencia a la base de datos, se muestran a continuación cada una de las tablas y relaciones que se requieren para generar el reporte.

```php
<?php
namespace Models\Usuarios;

use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\SoftDeletes;
use Illuminate\Support\Facades\DB;

class Usuario extends Model
{
    protected $table = "usuarios";
    protected $fillable = [
        "id",
        "id_old",
        "usuario_sap",
        "documento",
        "nombre",
        "email",
        "telefono",
        "username",
        "password",
        "password_nocrip",
        "tipo",
        "id_perfil",
        "empresa_id",
        "ciudad_id",
        "cargo",
        "evaluador_hse",
        "estado",
        "fecharegistro",
        "fecha_actualizacion",
        "token",
        "aprobador_catalogos",
        "foto_perfil",
        "fecha_nacimiento",
        "usuario_modificacion",
        "fecha_modificacion",
    ];
```

Figura 36. Modelo Usuarios. Autor.

```php
<?php
namespace Models\Aoc;

use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\SoftDeletes;
use Illuminate\Support\Facades\DB;

class Aoc extends Model
{
    protected $table = "aoc";

    protected $fillable = [
        "id_aoc",
        "seq_id",
        "id_cliente",
        "id_estudio_mercado",
        "id_solicitud_compra",
        "id_paa_linea",
        "id_abogado",
        "objeto",
        "id_solicitante",
        "id_centro_gestor",
        "urgencia_manifiesta",
        "requiere_presupuesto",
        "numero_acto",
        "fecha_acto",
        "moneda_acto",
        "valor_acto",
        "alcance",
        "contratacion",
        "justificacion",
        "alternativas",
        "antecedentes",
        "moneda",
        "valor_sin_iva",
        "iva",
        "valor_con_iva",
        "recurso_porcentaje",
        "proyecto",
        "modalidad",
        "fundamentos_norma",
        "tipo_supervision",
        "numero_comite",
        "fecha_comite",
        "fecha_creacion",
        "usuario_creacion",
        "fecha_actualizacion",
        "usuario_actualizacion",
```

Figura 37. Modelo Aoc. Autor.

```php
<?php
namespace Models\Ofertas;

use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\SoftDeletes;
use Illuminate\Support\Facades\DB;

class InformacionProveedoresInvitados extends Model
{
    protected $table = "informacion_proveedores_invitados_ofertas";

    protected $fillable = [
        'id',
        'id_oferta',
        'nit_invitado',
        'contacto_invitado',
        'email_invitado',
        'fecha_invitacion',
        'usuario_registro',
        'estado',
        'estado_participacion',
        'fecha_actualizacion',
        'observaciones',
        'nombre_usuario',
        'email_usuario',
        'fecha_registro',
        'id_usuario',
        'nit',
        'razon_social',
        'estado_proveedor',
        'carta_invitacion',
        'moneda',
        'valor'=> 'integer' ,
        'porcentaje',
        'observacion'
    ];
}
```

Figura 38. Modelo InformaciónProveedoresInvitados. Autor.

```php
<?php
namespace Models\Maestras\Actividades;

use Illuminate\Database\Eloquent\Model;
use Illuminate\Database\Eloquent\SoftDeletes;
use Illuminate\Support\Facades\DB;

class MstActividades extends Model
{
    protected $table = "mstactividades";
    protected $connection = "proveedores";
    protected $fillable = [
        'id',
        'cod_cliente',
        'segmenid',
        'segmendesc',
        'familiaid',
        'familiadesc',
        'claseid',
        'clasedesc',
        'producid',
        'producdesc',
        'categoria',
        'complejidad',
        'riesgo',
        'ficha_evaluacion',
        'estado'
    ];


}
```

Figura 39. Modelo Actividades. Autor.

Como se puede observar en este último modelo se adiciono la variable connection, esto quiere decir que se especifica la conexión a una base de datos, esto fue necesario debido a que Intelcost cuenta con dos bases de datos y en ocasiones es necesario realizar una relación con tablas en ambas bases de datos.

Luego de tener los modelos definidos junto con sus relaciones, en el archivo de Javascript donde se ejecuta la función cuando el usuario da click en el botón Generar Excel, se modifica la url a la cual la petición Ajax está enviando la información de filtros:

```
var generarExcelOfertas= function(){
    var filtros = new Object();
    filtros.id = $('#idOferta').val();
    filtros.objeto = $('#objetoOFerta').val();
    filtros.estado = $('#estadoOferta').val();
    filtros.solped = $('#solicitudesPedido').val();
    filtros.tipo = $('#tipooferta_listado').val();
    filtros.comprador = "";
    if(validarVariableExistente($("#responsableEvento").data("seleccionado"))){
        filtros.comprador = $("#responsableEvento").data("seleccionado");
    }
    $.ajax({
        type: "POST",
        url: "../ic_controladores/ctrl_ofertas.php",
        data: {
            act: 'generarExcelOfertas',
            filtros: filtros
```

Figura 40. Función generarExcelOfertas (Antes). Autor.

Se debe realizar el cambio de la url de la siguiente forma:

```
var generarExcelOfertas= function(){
    var filtros = new Object();
    filtros.id = $('#idOferta').val();
    filtros.objeto = $('#objetoOFerta').val();
    filtros.estado = $('#estadoOferta').val();
    filtros.solped = $('#solicitudesPedido').val();
    filtros.tipo = $('#tipooferta_listado').val();
    filtros.comprador = "";
    if(validarVariableExistente($("#responsableEvento").data("seleccionado"))){
        filtros.comprador = $("#responsableEvento").data("seleccionado");
    }

     mostrar_modal_general();

    $.ajax({
        url: '../Apis/reporteGeneralOfertas',
        type: 'POST',
        data: {
            filtros: filtros
```

Figura 41. Función generarExcelOfertas (Después). Autor.

Cuando la petición se realice, será direccionado al índex de Apis en donde se hace referencia a la función generarExcelOfertas del archivo DescargarOfertas.php como se muestra en la siguiente imagen:

```
['reporteGeneralOfertas' => 'reportes/DescargarOfertas.php', 'function' => 'DescargarOfertas@generarExcelOfertas'],
```

Figura 42. Archivo Índex en el directorio Apis. Autor.

Se crea un archivo con el nombre de DescargarOfertas.php para ser consultado como API, el cual contendrá toda la lógica y la utilización de estos modelos para generar el reporte, de igual forma se agregan las librerías que permitirá crear el reporte en excel, comunicarse con el modelo Ofertas y conectarse con la base de datos.

DescargarOfertas.php

```php
<?php
use Models\Ofertas\Oferta;
use Rap2hpoutre\FastExcel\FastExcel;
use Box\Spout\Writer\Style\StyleBuilder;
use Rap2hpoutre\FastExcel\SheetCollection;
use Illuminate\Database\Eloquent\Builder;

class DescargarOfertas{
    //Genera Excel general de las ofertas
    public function generarExcelOfertas($filtros){
        //Se toma la variable de sesión donde se indica el cliente
        $idempresa=$_SESSION["empresaid"];
        //Se toman los filtros que retorna el JS
        $filtros = $filtros->inputs();
        //Se convierte en objeto el array de filtros
        $filtros = $filtros["filtros"];
        //Se verifica que las variables necesarias existan y contengan algún valor
        if((isset($idempresa)||!$idempresa==null)&&(isset($filtros['tipo'])||!$filtros['tipo']==null)&&
            (isset($filtros['estado'])||!$filtros['estado']==null)){
            //Se inicializa conexión a BD
            new ConnectionSQL();
            //Se crea la variable con la información de la tabla ofertas donde id_cliente sea el cliente que genera el reporte
            $ofertas= Oferta::where('id_cliente',$idempresa)
            //Verifica que las ofertas estén en un estado distinto a INACTIVO
            ->where('estado','!=','INACTIVO')
            //Verifica si es una oferta cerrada, pública, estudio de mercado o RFQ
            ->where('tipo',$filtros['tipo'])
            //Obtiene los datos de la tabla a través de un select
```

Figura 43. Archivo DescargarOfertas (1/6). Autor.

En este punto se crea una clase con el nombre DescargarOfertas luego de esta clase se crea la función generarExcelOfertas ingresando el parámetro filtros que se envía desde la petición Ajax. Luego, se crea la variable de sesión de empresa para identificar qué cliente está realizando la petición, es importante antes de realizar la consulta a la base de datos verificar que los datos básicos para generar el reporte existan, por lo tanto, en la condicional se valida si existen y si son diferentes a nulo.

Una vez validado que los datos básicos existen, se crea la conexión a la base de datos inicializando el objeto ConnectionSQL como se muestra en la línea 21, en este punto, se puede observar que en la línea 23 se hace referencia al modelo Oferta creado anteriormente, también se condiciona donde el id_cliente sea igual a la variable de sesión tomada anteriormente (ver línea 12). Posteriormente, se colocan las condiciones de la consulta como se muestran en las líneas 25 y 27.

```
28          //Obtiene los datos de la tabla a través de un select
29          ->select(
30              'id',
31              'seq_id',
32              'solpeds_relacionadas',
33              'id_cliente',
34              'objeto',
35              'actividad',
36              'descripcion',
37              'moneda',
38              'presupuesto',
39              'maestra1',
40              'maestra2',
41              'fecha_inicio',
42              'hora_inicio',
43              'fecha_cierre',
44              'hora_cierre',
45              'ronda',
46              'id_aoc',
47              'id_requisicion',
48              'fecha_limite_msg_observacion',
49              'usuario_creacion',
50              'estado',
51              'fecha_creacion'
52          )
53          //Utiliza la relación infoUsuarioCreacion para obtener el nombre del usuario en creación
54          ->with(['infoUsuarioCreacion' =>function($query){
55              $query->select(
56                  'id',
57                  'nombre'
58              );
59          }])
60          //Si el filtro id existe, entonces busca ofertas donde el seq_id sea igual al digitado por el usuario
61          ->when(isset($filtros['id']), function($query) use ($filtros) {
62              return $query->where('seq_id','like','%'.$filtros['id'].'%');
63          }
64      )
65          //Si el filto objeto existe busca en la columna objeto y descripción el texto ingresado por el usuario
66          ->when(isset($filtros['objeto']), function($query) use ($filtros) {
67              $query->where('objeto','like','%'.$filtros['objeto'].'%');
68              $query->orWhere('descripcion','like','%'.$filtros['objeto'].'%');
69          }
```

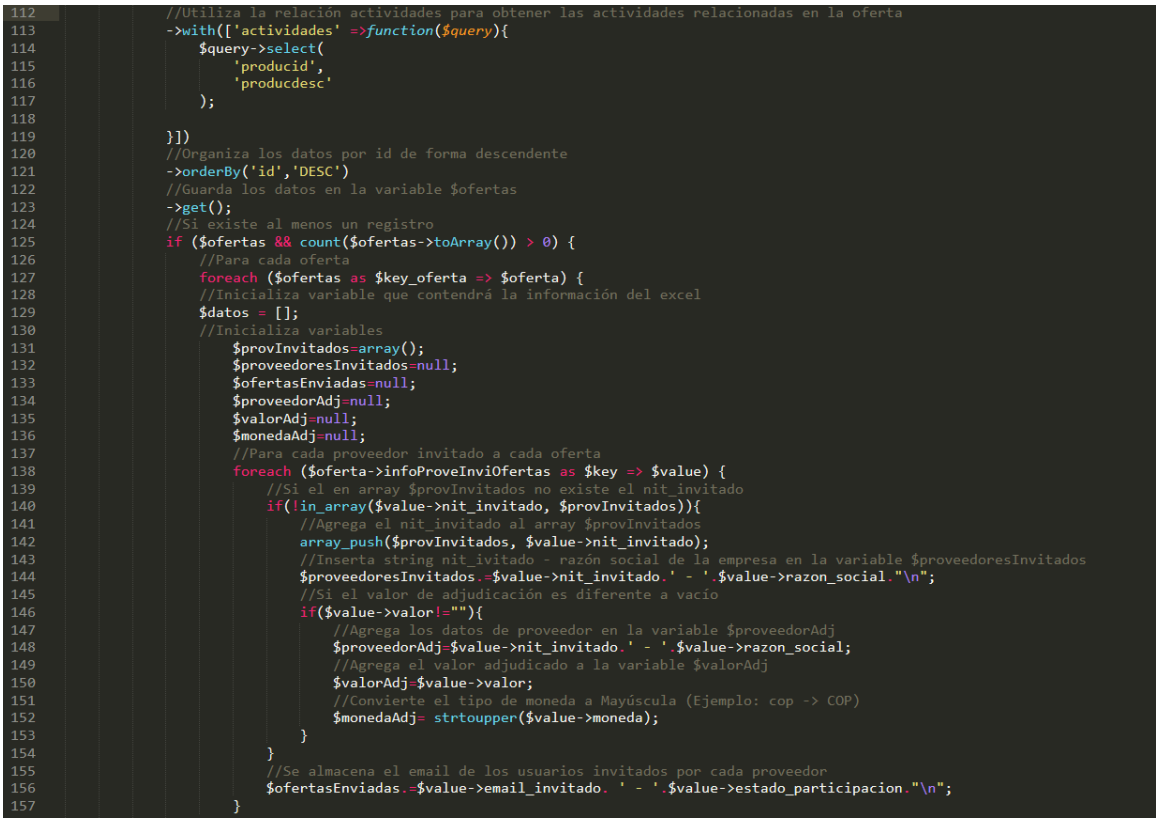
Figura 44. Archivo DescargarOfertas (2/6). Autor.

En este punto es posible indicar qué campos se requieren traer de la base de datos para todos los registros, esto se especifica con un select como se muestra en la anterior imagen. Para utilizar las relaciones y acceder a datos o registros de otras tablas relacionadas, se utiliza la palabra reservada with y se enviar como parámetro el nombre de la función descrita en el modelo. Es importante que, si se requiere mostrar solo algunas columnas de esta relación o realizar alguna condicional sobre esta relación, se debe incluir la sentencia function y realizar la consulta o la condicional como se muestra entre las líneas 54 y 59. Nota: Siempre se debe incluir la llave foránea (Como se muestra en la línea 56). Eloquent reconoce condicionales como IF, se realiza por medio de la palabra reservada when, como primer parámetro debe tener una respuesta de un booleano, en este caso, se está validado si la variable filtros[‘id’] existe, en dado caso de ser true condicionarán los resultados donde el seq_id contenga el valor de esta variable. Es importante incluir la variable filtros con la palabra reservada use (Como se muestra en la línea 61). El procedimiento continúa para cada una de las condicionales y relaciones.

```
71              //Si el usuario requiere el reporte con un estado en específico
72              ->when(($filtros['estado']!='ALL'), function($query) use ($filtros) {
73                  $query->where('estado',$filtros['estado']);
74              }
75          )
76              //Si se quiere algún número de solped en específico
77              ->when(isset($filtros['solped']), function($query) use ($filtros) {
78                  return $query->where('solpeds_relacionadas','like','%'.$filtros['solped'].'%');
79              }
80          )
81              //Si se desea buscar las ofertas creadas por cada comprador
82              ->when(isset($filtros['comprador']), function($query) use ($filtros) {
83                  return $query->where('usuario_creacion',$filtros['comprador']);
84              }
85          )
86              //Utiliza la relación infoAocRfq para obtener el número del AOC si lo tiene
87              ->with(['infoAocRfq' =>function($query){
88                  $query->select(
89                      'id_aoc',
90                      'seq_id AS seq_id_aoc'
91                  );
92
93              }])
94
95              //Utiliza la relación infoProveInviOfertas para obtener los proveedores invitados a la oferta
96              ->with(['infoProveInviOfertas' =>function($query){
97                  $query->select(
98                      'id_oferta',
99                      'nit_invitado',
100                     'contacto_invitado',
101                     'email_invitado',
102                     'fecha_invitacion',
103                     'estado_participacion',
104                     'observaciones',
105                     'razon_social',
106                     'moneda',
107                     'valor',
108                     'observacion'
109                 );
110
111             }])
```

Figura 45. Archivo DescargarOfertas (3/6). Autor.

Luego de realizar las condicionales y las relaciones, en la línea 121 se organizan los registros por orden de id de forma descendente y finalmente con la función get (), se realiza la consulta a la base de datos y los registros se almacenan en la variable ofertas creada inicialmente. Debido a que es un objeto que contiene los registros y es posible que contenga uno más registros (condicional de la línea 125), es necesario utilizar un foreach para obtener la información de cada registro (ver línea 127). Debido a que se realizaron relaciones uno a muchos, es necesario implementar un foreach anidado para acceder a la información como se muestra en la línea 138 para acceder a cada proveedor que se invitó a cada oferta, luego, cuando el proveedor tenga un valor adjudicado tomará nuevamente los valores y los almacenará en las variables $proveedorAdj, $valorAdj y $monedaAdj.



```
//Utiliza la relación actividades para obtener las actividades relacionadas en la oferta
->with(['actividades' =>function($query){
    $query->select(
        'producid',
        'producdesc'
    );

}])
//Organiza los datos por id de forma descendente
->orderBy('id','DESC')
//Guarda los datos en la variable $ofertas
->get();
//Si existe al menos un registro
if ($ofertas && count($ofertas->toArray()) > 0) {
    //Para cada oferta
    foreach ($ofertas as $key_oferta => $oferta) {
    //Inicializa variable que contendrá la información del excel
    $datos = [];
    //Inicializa variables
        $provInvitados=array();
        $proveedoresInvitados=null;
        $ofertasEnviadas=null;
        $proveedorAdj=null;
        $valorAdj=null;
        $monedaAdj=null;
        //Para cada proveedor invitado a cada oferta
        foreach ($oferta->infoProveInviOfertas as $key => $value) {
            //Si el en array $provInvitados no existe el nit_invitado
            if(!in_array($value->nit_invitado, $provInvitados)){
                //Agrega el nit_invitado al array $provInvitados
                array_push($provInvitados, $value->nit_invitado);
                //Inserta string nit_ivitado - razón social de la empresa en la variable $proveedoresInvitados
                $proveedoresInvitados.=$value->nit_invitado.' - '.$value->razon_social."\n";
                //Si el valor de adjudicación es diferente a vacío
                if($value->valor!=""){
                    //Agrega los datos de proveedor en la variable $proveedorAdj
                    $proveedorAdj=$value->nit_invitado.' - '.$value->razon_social;
                    //Agrega el valor adjudicado a la variable $valorAdj
                    $valorAdj=$value->valor;
                    //Convierte el tipo de moneda a Mayúscula (Ejemplo: cop -> COP)
                    $monedaAdj= strtoupper($value->moneda);
                }
            }
            //Se almacena el email de los usuarios invitados por cada proveedor
            $ofertasEnviadas.=$value->email_invitado. ' - '.$value->estado_participacion."\n";
        }
```

Figura 46. Archivo DescargarOfertas (4/6). Autor.

Luego de obtener y almacenar la información obtenida al realizar la consulta en la base de datos, se crea un objeto para almacenar esta información obtenida y estructurar el excel que el usuario desea descargar. En la siguiente imagen se muestra un objeto donde se define una llave y una variable, estos datos representan el nombre de la columna en excel y el valor que se colocará en esta columna respectivamente.

```
158                     //Objeto que obtiene la información que se visualizará en el excel
159                     $datos[] =
160                     [
161                         //Asigna los valores con las columnas
162                         'ID oferta' => $oferta->seq_id,
163                         'Objeto' => $oferta->objeto,
164                         'Actividad' => $oferta->actividades->producdesc,
165                         'Descripción' => $oferta->descripcion,
166                         'Moneda' => $oferta->moneda,
167                         'Presupuesto' => $oferta->presupuesto,
168                         'Fecha de inicio' => $oferta->fecha_inicio,
169                         'Hora de inicio' => $oferta->hora_inicio,
170                         'Fecha de cierre' => $oferta->fecha_cierre,
171                         'Estado' => $oferta->estado,
172                         'Observación límite de mensajes' => $oferta->fecha_limite_msg_observacion,
173                         'Comprador' => $oferta->infoUsuarioCreacion->nombre,
174                         'Fecha de creación' => $oferta->fecha_creacion,
175                         'Proveedores invitados' => $proveedoresInvitados,
176                         'Proveedores evaluados' => $ofertasEnviadas,
177                         'Proveedor adjudicado' => $proveedorAdj,
178                         'Moneda adjudicación' => $monedaAdj,
179                         'Valor adjudicación' => $valorAdj,
180                     ];
181                     //Si la empresa es Metro de medellín adiciona dos columnas
182                     if ($idempresa == 10) {
183                         $datos[$key_oferta]['AOC'] = $oferta->infoAocRfq->seq_id_aoc ;
184                         $datos[$key_oferta]['Requisición'] = $oferta->id_requisicion;
185                     //Si la empresa es Terpel o Confa adiciona las dos columnas
186                     }else if($idempresa == 14 || $idempresa == 26){
187                         $datos[$key_oferta]['Ronda'] = $oferta->ronda ;
188                         $datos[$key_oferta]['Solpeds'] = $oferta->solpeds_relacionadas;
189                     }
190                 }
```

Figura 47. Archivo DescargarOfertas (5/6). Autor.

Para poder crear el archivo de excel es necesario crear un nombre único para el archivo (línea 192), especificar el conjunto de datos (línea 194), definir el estilo de las celdas y la presentación del excel (líneas 196-200). Luego de obtener estos datos, se procede a generar el excel como se muestra en la línea 202, es en este punto donde el excel se descarga en el servidor. Finalmente, solo queda que el archivo se envíe al usuario y se descargue por el navegador, este proceso se realiza en las líneas 204 a 216, una vez realizado esto el usuario obtendrá el reporte descargado y el archivo almacenado el servidor es eliminado finalmente.

```php
                //Nombre y configuración del archivo Excel
                $nombre_archivo = 'Reporte ofertas Intelcost'.date('d-m-Y').' '.uniqid().'.xlsx';
                //Convierte el objeto en Collect
                $list = collect($datos);
                //Configura el diseño del excel
                $style = (new StyleBuilder())
                ->setFontBold()
                ->setFontSize(11)
                ->setShouldWrapText(false)
                ->build();
                //Crea el archivo de excel
                (new FastExcel($list))->headerStyle($style)->export('reportes_temp/'.$nombre_archivo);
                //Crea y almacena el archivo de forma temporal
                    header("Cache-Control: max-age=$seconds_to_cache");
                    header('Content-Description: File Transfer');
                    header('Content-Type: application/octet-stream');
                    header('Content-Disposition: attachment; filename="'.basename('reportes_temp/'.$nombre_archivo).'"');
                    header('Content-Transfer-Encoding: binary');
                    header('Expires: 0');
                    header('Cache-Control: must-revalidate, post-check=0, pre-check=0');
                    header('Pragma: public');
                    header('Content-Length: ' . filesize('reportes_temp/'.$nombre_archivo));
                    ob_clean();
                    flush();
                    //Descarga el archivo al usuario que lo genera
                    readfile('reportes_temp/'.$nombre_archivo);
                    //Elimina el archivo almacenado en el servidor
                    unlink('reportes_temp/'.$nombre_archivo);
                    //Cierra la conex
                    exit;
            }else{
                var_dump("No se encontraron ofertas");
            }

        }else{
            var_dump("Parámetros insuficientes para generar reporte");
        }}
    }
?>
```

Figura 48. Archivo DescargarOfertas (6/6). Autor.

10. RESULTADOS

Luego de implementar la optimización en entornos tanto locales como en los ambientes de prueba (Develop, Alpha y UAT) se obtuvieron los siguientes resultados:

10.1 TIEMPOS DE RESPUESTA

10.1.1 Metro de Medellín

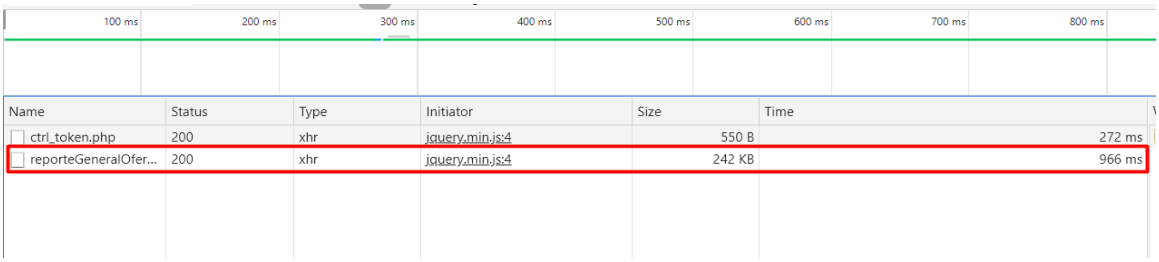


Figura 49. Tiempo en realizar la petición optimizada a Metro de Medellín. Autor.

10.1.2 Terpel

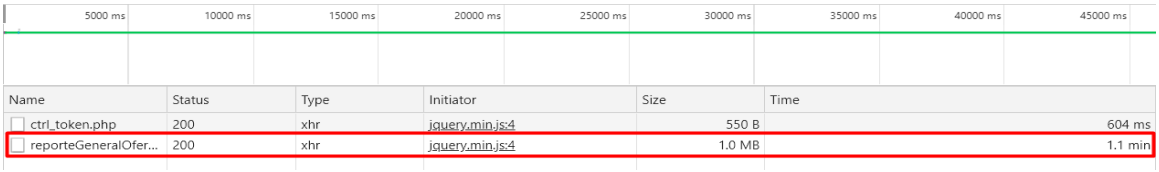


Figura 50. Tiempo en realizar la petición optimizada a Terpel Autor.

10.1.3 Mansarovar

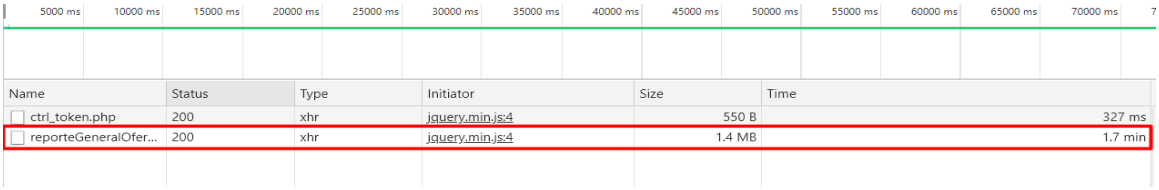


Figura 51. Tiempo en realizar la petición optimizada a Mansarovar. Autor.

10.1.4 Masa Stork

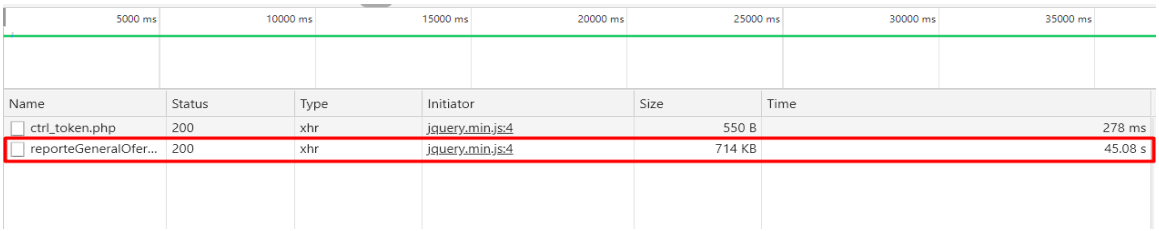


Figura 52. Tiempo en realizar la petición optimizada a Masa Stork. Autor.

En resumen, se puede observar que la mayoría de los reportes antes no se podían descargar, pero con la implementación del ORM de Laravel se logró optimizar los tiempos en obtener la información de la base de datos. A continuación, puede observar un cuadro comparativo con los tiempos antes y después de la implementación, y con su respectivo porcentaje de optimización calculado.

Tabla 2. Resultado de la optimización.

Cliente	***Tiempo en generar reporte sin Eloquent***	***Tiempo en generar reporte con Eloquent***	***Diferencia de tiempo***	***Porcentaje de optimización***
MDM	1.7 min	0,966 s	1699,034 s	99.9431%
Terpel	Más de 5 min	1.1 min	3,9 min	Más del 78%
Mansarovar	Más de 5 min	1.7 min	3,3 min	Más del 66%
Masa Stork	Más de 5 min	45,08 s	4954,92 s	Más del 99.098%

11. CONCLUSIONES

Luego de implementar Google Analytics como herramienta de analítica y configurarla para su funcionamiento en la plataforma Intelcost, el personal de Intelcost puede realizar seguimiento en vivo de donde se encuentran los usuarios al navegar en la plataforma, esto permite generar estadísticas para mejorar la experiencia del usuario. Sin embargo, al momento de obtener los reportes generados por la herramienta no siempre se pueden personalizar dependiendo la necesidad del cliente.

Gracias a la implementación de la herramienta de Google Analytics, se pudo identificar que aproximadamente 350 usuarios estaban intentando acceder a un reporte que no se podía descargar por tiempo de espera excedido. Por lo tanto, la herramienta proporciona información de valor para identificar la cantidad de usuarios afectados por un bug de programación.

Luego de contrastar los tiempos en descargar el reporte con cada cliente y posterior a la implementación de Eloquent, se puede concluir que los ORM son una herramienta poderosa al momento de realizar consultas más ágiles en entornos de programación basada en un framework como PHP. Se recomienda finalmente utilizar Eloquent e implementar vistas en la base de datos para agilizar consultas de este volumen

12. BIBLIOGRAFÍA

1. **Fernández, Nazario García.** [En línea] Noviembre de 2012. [Citado el: 02 de 01 de 2020.] https://www.researchgate.net/publication/328174732_e-Procurement_Importancia_y_Aplicacion.

2. **Palacios, Miguel.** Researchgate. *Researchgate.* [En línea] Enero de 2010. [Citado el: 03 de 02 de 2020.] https://www.researchgate.net/publication/47900940.

3. **Laravel (Taylor Otwell).** Eloquent: Getting Started. [En línea] 2011. https://laravel.com/docs/5.7/eloquent#introduction.

4. **Kissmetrics.** Kissmetrics. [En línea] 27 de 01 de 2020. https://www.kissmetricshq.com/product/.

5. **Google.** Marketing Google. [En línea] 02 de 03 de 2020. https://marketingplatform.google.com/intl/es/about/analytics/benefits/.