CIS1830CP01

Deep Dark Web & Social Crawler (DDW&SC): Aplicativo para apoyar la gestión de Ciberinteligencia

Gina Daniela Colmenares Malaver
Nicolás Méndez González
Oscar David Virgüez Castro

PONTIFICIA UNIVERSIDAD JAVERIANA
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA DE SISTEMAS
BOGOTÁ, D.C.
2019

CIS1830CP01

Deep Dark Web & Social Crawler (DDW&SC): Aplicativo para apoyar la gestión de Ciberinteligencia

Autor(es):

Gina Daniela Colmenares Malaver
Nicolás Méndez González
Oscar David Virgüez Castro

MEMORIA DEL TRABAJO DE GRADO REALIZADO PARA CUMPLIR UNO DE LOS REQUISITOS Y OPTAR AL TÍTULO DE INGENIERO DE SISTEMAS

Director

Joshsua James González Díaz

Jurados del Proyecto Final

Ing. Wilson Arturo Prieto Hernández

Ing. Germán Romero Gutiérrez

Sitio Web del Proyecto Final

https://livejaverianaedu.sharepoint.com/sites/Ingsis/TGCIS/183001

PONTIFICIA UNIVERSIDAD JAVERIANA
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA DE SISTEMAS
BOGOTÁ, D.C.
DICIEMBRE, 2019

PONTIFICIA UNIVERSIDAD JAVERIANA
FACULTAD DE INGENIERÍA
INGENIERÍA DE SISTEMAS

Rector de la Pontificia Universidad Javeriana

Jorge Humberto Peláez Piedrahita, S.J.

Decano de la Facultad de Ingeniería

Ing. Lope Hugo Barrero Solano

Directora del programa Ingeniería de Sistemas

Ing. Mariela Josefina Curiel Huérfano

Director del departamento de Ingeniería de Sistemas

Ing. Efraín Ortíz Pabón

Artículo 23 de la Resolución No. 1 de junio de 1946

"La Universidad no se hace responsable de los conceptos emitidos por sus alumnos en sus proyectos de grado. Sólo velará porque no se publique nada contrario al dogma y la moral católica y porque no contengan ataques o polémicas puramente personales. Antes bien, que se vean en ellos el anhelo de buscar la verdad y la Justicia".

AGRADECIMIENTOS

Agradecemos a la Pontificia Universidad Javeriana y a sus docentes por la formación a nivel personal y profesional, en el transcurso de la carrera. Igualmente agradecemos a nuestro director de trabajo de grado por la confianza, dedicación y acompañamiento a lo largo del desarrollo del trabajo de grado.

También le agradecemos a Alejandro Sierra por el apoyo en el modelo de arquitectura de datos, al teniente Juan Carlos García por su acompañamiento y asesoría durante el proceso a, Carlos Hernández por su dedicación y apoyo en el diseño gráfico y a Angélica Vergara por su asesoría y ayuda para la arquitectura del proyecto.

Finalmente, agradecemos a Wilson Prieto, coordinador del Grupo de Atención a Emergencias Cibernéticas de Colombia ColCERT y al ingeniero Germán Romero, profesor de maestría de la Universidad de los Andes, por su asistencia a la sustentación y comentarios para la mejora del trabajo de grado.

TABLA DE CONTENIDO

ABSTRACT

Dado el gran reto de las organizaciones por mitigar las amenazas cibernéticas hacia la información que reside en medios electrónicos. *Deep Dark Web & Social Crawler* (*DDW&SC*) es un aplicativo para apoyar la gestión de Ciberinteligencia, que permite la búsqueda y recolección de información en la *Deep Web*, *Dark Web* y algunas redes sociales preseleccionadas, para reforzar la estrategia de seguridad de la información y toma de decisiones en las compañías y organizaciones.

One of the biggest challenge of organizations is to mitigate cybersecurity threats and protect to the information that resides on electronic media. *Deep Dark Web* & Social Crawler (DDW&SC) is an application to support cyberintelligence, which allows searching and collecting information on the *Deep Web*, *Dark Web* and a few *Social Networks*, to strengthen information security strategies in companies and organizations.

INTRODUCCIÓN

Eric Hoffer, escritor y filósofo estadounidense cita *"En tiempos de cambio, quienes estén abiertos al aprendizaje se adueñarán del futuro, mientras que aquellos que creen saberlo todo estarán bien equipados para un mundo que ya no existe"* (Manuel & Santiago, 2016), mostrándonos que la globalización y las innovaciones tecnológicas están afectando las actividades económicas y por supuesto el desarrollo de las naciones. Los avances tecnológicos son abrumadores viéndose reflejados en los estilos de vida que tienen las personas, teniendo como precedente que la adopción y operación exitosas de cualquier nueva tecnología depende de la gestión adecuada de los riesgos asociados con la misma.

En Colombia se están comenzando a usar conceptos como *Blockchain* para el registro y aseguramiento de las transacciones (Rubio & Huertas, 2018); *Big Data* y *Data Analytics* para el almacenamiento de grandes volúmenes de información y la toma de decisiones; *Machine Learning* para el otorgamiento de créditos (ACIS, 2019), reconocimiento de siniestros y prevención del fraude; *Algorithmic Trading* para la compra y venta de valores en los mercados electrónicos; C*loud Computing* para el desarrollo, pruebas y operación de aplicaciones administrativas y misionales; *Artificial Intelligence* para manejar portafolios financieros (Portafolio, 2019); *Biometrics* para el reconocimiento y autenticación de los clientes; IoT o Internet de las cosas para lograr una tarifación adecuada de las pólizas de seguros; *Smart Contracts,* o contratos inteligentes, en operaciones de comercio electrónico; *API (Application Programming Interface)* y *Web Services* para proveer información a otras organizaciones sin depender de elementos computacionales particulares. Estas tecnologías ya se han consolidado, están al alcance de personas y entidades de todo tipo y tamaño y su aplicación en diferentes sectores y actividades solo está limitada por la creatividad.

Basado en lo anterior, las organizaciones a nivel mundial ven con preocupación el accionar ante ataques cibernéticos que tienden a poseer una naturaleza sistémica afectando a todo un ecosistema cibernético. Por tal razón es necesario comenzar a migrar a planes tanto de respuesta como recuperación que sean concebidos mucho antes que una materialización de un incidente y de ahí la necesidad de ser preventivos antes que reactivos.

Para garantizar una toma de decisiones acertada en las organizaciones, surge la necesidad de una herramienta de apoyo para la ciberinteligencia, que permita la búsqueda, recolección de datos no solo en la red superficial, sino en la *Deep Web*, *Dark Web* y redes sociales.

En el presente documento se expone la solución planteada a la problemática relacionada con la seguridad de la información en la actualidad. Este consta de varias secciones en las cuales se abarcará desde el proceso de contextualización, planeación y diseño hasta el desarrollo y despliegue del aplicativo.

DESCRIPCIÓN GENERAL

Oportunidad, Problema

Contexto del problema.

La interconectividad y la adopción de las nuevas tecnologías traen consigo un incremento en la exposición a los riesgos cibernéticos que están demandado por parte de las organizaciones tanto públicas como privadas, y autoridades, esfuerzos importantes para identificarlos y gestionarlos. Uno de los grandes retos de las organizaciones lo constituye la seguridad de la información, en particular, la que reside o se procesa en medios electrónicos, la cual, ahora más que nunca, se encuentra expuesta a las amenazas cibernéticas, dada la naturaleza global de Internet y de los sistemas de información, que no tienen una limitación fronteriza (FireEye, 2012). En los últimos meses hemos podido ratificar la importancia del tema, donde casos como los ataques a entidades financieras tanto en Centro América como Sur América llega a colocarnos en la posición de realizar esfuerzos tanto a nivel técnico como procedimental y legal.

El comunicado emitido en la reunión de marzo de 2017, los ministros y gobernadores del G20 en Baden-Baden señalaron que el uso malicioso de las tecnologías de la información y las comunicaciones podría alterar los servicios financieros nacionales e internacionales cruciales para el desarrollo de la sociedad, desmejorando la seguridad, la confianza y poniendo en peligro la estabilidad financiera. Mencionaron además que promoverían la capacidad de recuperación de los servicios e instituciones financieras en las jurisdicciones del G20 contra el uso malicioso de las tecnologías de la información y la comunicación, incluso de países fuera del G20 (*G20—2017—Communiqué G20 Finance Ministers and Central Bank .pdf*, s. f.).

Lo anterior es basado en los ciberataques, los cuales han llegado a posicionarse como una de las amenazas más latentes y significativas a nivel mundial siendo el sistema financiero uno de los más afectados. Informes recientes muestran un sinnúmero de ataques significativos y exitosos tanto dentro como fuera del sector. Ataques como lo han sido el de 2016 al Banco de Bangladesh que resultó en el robo de 81 millones de dólares (bdnews24, 2019), el ataque del *Ransomware WannaCry* que infectó más de doscientos cincuenta mil (250,000) sistemas informáticos en 150 países diferentes (AS, 2019), y la fuga de información de Equifax que ha comprometido hasta 143 millones individuos (The New York Times, 2017), son muestra de una realidad preocupante frente a una hiper conectividad con la cual convivimos al día de hoy. La naturaleza cambiante del riesgo cibernético para las instituciones financieras está impulsada por varios factores, incluida la evolución de la tecnología, que puede generar vulnerabilidades nuevas; interconexiones entre instituciones financieras y partes externas (p. e. a través de proveedores de computación en la nube y *FinTech* que pueden estar fuera de un marco regulatorio); esfuerzos de los ciberdelincuentes para encontrar nuevos métodos de ataque y comprometer los sistemas de TI; y acciones delictivas mediante uso de medios informáticos que buscan ganancias financieras ilícitas. Reconociendo la amenaza de los riesgos cibernéticos y la naturaleza crítica de mejorar la resiliencia de las instituciones financieras a estos riesgos, las autoridades de todo el mundo han tomado medidas regulatorias y de supervisión diseñadas para facilitar tanto la mitigación del riesgo cibernético por parte de las instituciones financieras y su respuesta efectiva y recuperación de los ciberataques.

El "riesgo cibernético" se refiere a los riesgos operativos que pueden resultar en la pérdida de confidencialidad, integridad y disponibilidad de datos o información; y el riesgo que puede afectar negativamente la infraestructura de las tecnologías de la información (TI) o las operaciones de una entidad financiera. Por otro lado, el riesgo operativo se entiende comúnmente como el riesgo de

pérdida resultante de procesos internos inadecuados o fallidos, personas y sistemas o de eventos externos. Basado en esto, el riesgo cibernético se entiende generalmente como parte integral del riesgo operacional y surge de eventos "cibernéticos" intencionales o maliciosos en los que algo sale mal en el entorno de la organización y de su infraestructura de TI que se encuentra interconectada, ya sea físico o virtual. Adicional a esto se considera a la vez el riesgo de hacer operaciones en el entorno "cibernético" o virtual que comprende Internet, comunicaciones inalámbricas o computación en la nube, en otras palabras, la exposición de acciones u operaciones en el ambiente conocido como el "ciberespacio" (*EY - 2017—EY-recuperando-la-ciberseguridad.pdf.pdf*, s. f.). Ha habido intentos de definir este riesgo con mayor precisión. Por ejemplo, el *CRO Forum* lo define como "cualquier riesgo que surja del uso de datos electrónicos y su transmisión, incluidas herramientas tecnológicas como Internet y las redes de telecomunicaciones. También abarca los daños físicos que pueden ser causados por los ciberataques, el fraude cometido por el mal uso de los datos, cualquier responsabilidad derivada del almacenamiento de datos y la disponibilidad, integridad y confidencialidad de la información electrónica, ya sea relacionada con personas, empresas o gobiernos" (*CRO Forum—2016—CRO Forum Concept Paper on a proposed categorisati.pdf*, s. f.).

Los orígenes del riesgo cibernético pudieran encontrarse desde la organización o desde una fuente externa. Un evento de riesgo cibernético a menudo es intencional, deliberado o malicioso, pero podría ser involuntario, vinculado a una falla de software, mal funcionamiento del hardware o configuración incorrecta de un componente de TI. Comprender la causalidad del riesgo cibernético es relevante desde la perspectiva de una institución financiera que establece los controles internos del sistema y los componentes de seguridad de TI. El impacto de un evento de riesgo cibernético converge con varios otros dominios de riesgo. El riesgo cibernético generalmente se discute de

forma independiente, ya que se ha convertido en un riesgo importante en los últimos años. En realidad, el impacto de un evento de riesgo cibernético no está aislado o confinado, pero a menudo desencadena un conjunto consecuente de eventos que impacta y se superpone con otros riesgos. La evaluación de impacto del posible riesgo cibernético, dentro del marco más amplio de gestión de riesgos, es por lo tanto un desafío

Adicionalmente a este contexto, se realizó un documento sobre el estado del arte (Anexo 3 – Estado del arte), en el que se menciona el funcionamiento de la red *TOR*, comparativa con otras redes similares y sistemas recomendados para navegar en estas redes, entre otros trabajos en el área.

Formulación del problema.

Dada la ausencia de herramientas de búsquedas parametrizadas en la *Deep Web* y redes sociales, se dificulta la obtención de información en páginas pertenecientes a este tipo de red, ya que buscadores convencionales como Google© se limitan a la web superficial y no alcanzan a llegar a páginas con dominios de servicios ocultos *(.onion)*. Estas páginas solo son accedidas mediante la instalación de extensiones a los navegadores convencionales o mediante la conexión a través de navegadores como *TOR*.

Lo anterior significaría un desconocimiento de la información que allí reside, y si se está haciendo un uso malintencionado, ¿cómo se podría mitigar ese riesgo una vez encontrada la información?.

Propuesta de solución.

Aprovechando la revolución de la información, surge la necesidad de implementar una herramienta de apoyo para la gestión de ciberinteligencia, con el fin de mitigar la problemática encontrada; se propone desarrollar un aplicativo de software que permita a los usuarios realizar búsquedas parametrizadas. El aplicativo se encargará de listar las páginas que cumplan con los filtros establecidos

para búsqueda. Además, se mostrarán resultados acertados dadas unas palabras clave. El detalle de cada página ofrecerá, información de éstas, y un análisis estadístico sencillo del contenido de las mismas. Una vez el aplicativo inicie la búsqueda entregará en tiempo real los resultados obtenidos, mostrando los enlaces a las páginas o archivos encontrados, una descripción y material gráfico relacionado con el contenido encontrado.

Justificación de la solución.

La necesidad principal de una herramienta para las búsquedas parametrizadas en *Deep Web*, *Dark Net* y redes sociales, surge de la ausencia de estas. Regularmente este proceso se realiza de forma manual, accediendo a un navegador como *TOR* y posteriormente a diferentes páginas y/o blogs donde se encuentran los enlaces a otros sitios web. Este proceso se considera poco efectivo, ya que no se alcanza a acceder a ciertos niveles de profundidad en la red y, por ende, el usuario no obtiene un gran volumen de información.

En el caso de plataformas sociales como *Reddit®, Twitter® y Pastebin®* se dispone de una barra de búsqueda, sin embargo, se debe hacer la búsqueda independiente en cada una de ellas, no hay una integración de una misma búsqueda y todas las plataformas.

Igualmente, a partir de las problemáticas anteriormente descritas, surge la oportunidad de la creación de una herramienta que, mediante la búsqueda parametrizada, apoye la gestión de ciberinteligencia, mediante el análisis de la información obtenida.

Descripción del Proyecto

Objetivo general.

Desarrollar una herramienta para apoyar la gestión de ciberinteligencia y que permita realizar búsquedas parametrizadas, ofreciendo a las organizaciones la posibilidad de un levantamiento de información disponible en *Deep Web*, *Dark Web* y redes sociales.

Objetivos específicos.

- Identificar los componentes, elementos y arquitecturas usadas para el acceso a *Deep/Dark web*.
- Diseñar los componentes requeridos para las búsquedas dentro de las fuentes de información seleccionadas.
- Implementar los componentes requeridos para las búsquedas.

Entregables, Estándares utilizados y Justificación.

Tabla 1. Entregables y estándares utilizados

Entregable	Estándares asociados	Justificación
SPMP	IEEE 1058.1-1987 (IEEE, 1988)	Para el contenido del entregable SPMP se tomó como base el estándar IEEE 1058.1-1987, haciendo modificaciones para adaptarlo a este proyecto, se describe una vista general, el contexto y la forma en que se va a administrar el proyecto. En este documento es donde se establecen los estándares, herramientas y forma en que el equipo va a trabajar durante el desarrollo del trabajo de grado.

Tabla 2. (continuada).

Entregable	Estándares asociados	Justificación
SRS	IEEE 830-1998 (IEEE, 1998)	El documento SRS adaptado para este proyecto está basado en el estándar IEEE 830-1998, en este documento se describen las bases establecidas entre el director (Joshsua González) y el equipo de trabajo. Se especifican todas las funcionalidades y atributos del sistema a desarrollar.
SDD	IEEE 1016-2009 (IEEE, 2009)	Este documento está basado y adaptado del estándar IEEE 1016-2009, en él se especifica la arquitectura y las interfaces lógicas y de usuario que serán diseñadas e implementadas.
Manuales de Usuario	-	Este documento es una guía de uso de la herramienta para facilitar el despliegue y brindar una ayuda al usuario final para que pueda realizar las búsquedas.
Memoria	Documento propuesto por la Pontificia Universidad Javeriana	Este documento presenta la información del proyecto de grado DDW&SC, tanto la planeación como el desarrollo de este.
Documento de pruebas	-	Con este documento se pretende mostrar toda la fase de pruebas realizadas con el fin de asegurar el correcto funcionamiento de cada uno de los módulos desarrollados, como también del producto final entregado, en el cual se asegure una salida satisfactoria luego de ingresar búsquedas por parte del cliente final.
Aplicativo	-	Finalmente, el equipo de trabajo entregará un producto de software acorde a los requerimientos especificados y consignados en el documento SRS.

CONTEXTO DEL PROYECTO

Contexto

Este proyecto se realizó con la intención de brindar una herramienta para realizar búsquedas en la red oculta *TOR*, pues a pesar de que existen buscadores como Google©, Yahoo!© o Bing© que son capaces de indexar páginas e información expuesta en Internet, hay que ser conscientes que sólo se está buscando aproximadamente en el 4% del Internet (Panda Security, 2016).

La red *TOR*, fue creada bajo una creencia común: *"Los usuarios de Internet deben tener acceso privado a una web sin censura"* (Tor Project, 2015). En los años 90, miembros del *US Naval Research Lab* (NRL) iniciaron con el diseño e investigación del enrutamiento de cebolla *(The Onion Router)* (Tor Project, 2015).

Poco a poco la red fue adquiriendo fuerza y apoyo de voluntarios, tanto así que para el año 2007, ya era posible navegar saltando *firewalls* y cortafuegos que impedían la libre conexión de forma anónima en algunas zonas geográficas que eran censuradas por los gobiernos (Tor Project, 2015).

Este anonimato dio pie para que usuarios malintencionados empezaran a usar la red con fines delictivos, lo que ha obligado a las autoridades a tratar de regular la red, pero aún ha sido un reto muy grande para ellos.

Debido a la gran volatilidad de la red, no existen buscadores sofisticados que puedan indexar y mantener una gran base de datos histórica para realizar búsquedas rápidas y acertadas. Por esta razón, se vio la oportunidad de crear una herramienta que apoyara en la búsqueda de información sobre la red *TOR*.

Adicionalmente a la *Deep Web*, el proyecto incorpora un módulo de búsqueda en plataformas sociales, inicialmente se realizó el desarrollo para búsqueda en *Facebook*® y *Twitter*®. Sin embargo, por los problemas legales que presentó *Facebook*® desde el caso de *Cambridge Analytica*® en el año 2018, la red social de Mark Zuckerberg decidió dar de baja a mediados del mes de junio de 2019 la herramienta de *Graph Search*, la cual permitía realizar búsquedas de contenidos públicos alojados en *Facebook*® (Nguyen, 2019).

Esto afectó en gran magnitud el proyecto, dejando obsoleto el desarrollo realizado para búsquedas en esta red social. Por tal motivo, el equipo DDW&SC decidió reemplazarlo por búsquedas en *Reddit*®, una red social pensada para hablar de cualquier tema de interés entre los usuarios y que tiene grandes comunidades alrededor del mundo (Reddit, 2005). Sin embargo, debido que el problema con *Facebook*® significaba una gran baja para el proyecto, también se vio la posibilidad de implementar búsquedas dentro de *Pastebin*®.

Pastebin® es un sitio web, en donde los usuarios pueden copiar, pegar y compartir cualquier contenido en texto plano, es usado principalmente por programadores para almacenar código fuente, configuraciones, etc. Sin embargo, existe una población que lo usa para compartir ideas o comunicar cualquier contenido textual (Pastebin, 2002).

Análisis del contexto

Dentro de la *Deep Web* existe una gran variedad de buscadores que facilitan de una u otra forma la búsqueda de contenido e información en esta red, últimamente gobiernos y organizaciones han puesto interés y esfuerzo por desarrollar plataformas que puedan mejorar la búsqueda y clasificar los resultados que se encuentren. Algunos de los buscadores más conocidos y utilizados en la *Deep*

Web son ***Torch, Ahmia, VisiTOR, Not Evil*** y todas las Wikis en donde en su mayoría son páginas que clasifican diferentes *URL* por su contenido (Neoteo, 2019).

Recientemente una *start-up* francesa desarrolló un sistema para hacer búsquedas en la *Deep Web*, al cual planean incorporarle procesamientos con inteligencia artificial, sin embargo, por el cuidado de la herramienta, no es pública y su uso es muy restringido (Milenio, 2018).

Por parte de las redes sociales, antes de la desactivación de *Graph Search* existían diferentes herramientas *online* para hacer búsquedas en esta plataforma, luego de la desactivación todas ellas dejaron también de funcionar y hoy en día sólo es posible encontrar resultados vinculados con las relaciones de amistad y páginas de interés asociados a una cuenta. Con *Twitter*®, *Pastebin*® y *Reddit*® existen *API,* algunas versiones de pago y otras herramientas *online* que permiten buscar usuarios, publicaciones entre otros.

DDW&SC reúne distintos buscadores y *API* para realizar búsquedas de información en un meta-buscador, centralizando todos los resultados en una sola herramienta que además permite realizar un filtro sobre ellos para destacar aquellos que tengan más relevancia según el perfilamiento que entregue el usuario sobre sus intereses en la búsqueda. Además, permite hacer el proceso de *crawling* sobre un resultado, es decir que empieza a buscar coincidencias partiendo desde una URL y paseando por el contenido de la página y sus enlaces encontrados hasta el límite que estipule el usuario.

Los resultados que son encontrados pueden ser exportados a un archivo PDF, en donde queda evidencia de estos y además se provee de un *dashboard* con el resumen de la búsqueda en general.

ANÁLISIS DEL PROBLEMA

En esta sección se muestran las especificaciones en las que se basó el proyecto para cumplir con el objetivo general. Se describen los requerimientos, restricciones y toda la especificación funcional, los cuales son detallados en el documento SRS (Especificación de requerimientos de software).

Requerimientos

Entendido el contexto del problema y la solución propuesta por el equipo, se establecieron los diferentes requerimientos que sirvieron como base inicial y guía durante todo el desarrollo del proyecto.

Dentro de los requerimientos establecidos se tuvieron en cuenta las características funcionales de la herramienta y algunas características no funcionales que le brindan estabilidad y mejoran la experiencia del usuario final.

Durante la implementación de las funcionalidades el equipo de trabajo optó por manejar historias de usuario por su flexibilidad para realizar cambios y tener un modelo de desarrollo muy cercano al de una metodología ágil, las historias de usuario principales son listadas a continuación en el tercer numeral de esta sección, y pueden ser consultados con su detalle en el Anexo 1 – Especificación de historias de usuario.

En la especificación de requerimientos aparecen dos (2) actores principales: El sistema y el usuario final. El sistema como actor, se encarga de procesar todas las peticiones y ejecutar cada uno de los procesos necesarios para poder satisfacer las solicitudes, dando la respuesta esperada por el usuario final, ya sean búsquedas, reportes, *crawling*, etc.

El usuario final, es el administrador y cliente del sistema, es quien inicia las solicitudes y hace uso de los resultados para encontrar información de acuerdo con sus intereses, tal y como lo haría en un buscador convencional como lo es Google©, Yahoo!© o Bing©.

Como el objetivo principal del sistema es realizar búsquedas en la *Deep/Dark Web* y sobre algunas plataformas usadas por comunidades de personas, se hizo uso de buscadores y *API* ya existentes para obtener los resultados según los intereses de usuario final. Esto conllevó algunas demoras en los tiempos de respuesta de cada plataforma además de exigencias de hardware, limitando el proyecto con las siguientes restricciones.

Restricciones

Desde el inicio del proyecto se tenían contempladas algunas restricciones, sin embargo, durante el desarrollo de este fueron surgiendo otras restricciones que eran de gran importancia para tener en cuenta y buscar alternativas de tal forma que no se viera afectado negativamente el desarrollo y poder cumplir con la propuesta inicial. A continuación, se listan las diferentes restricciones encontradas y una descripción de cada una de ellas.

Tiempo.

El tiempo total para desarrollar el proyecto de grado es de 12 meses el cual se divide en dos (2) partes de seis (6) meses cada uno (ver el Anexo 5 – Cronograma de proyecto).

La primera es de planeación y definición del proyecto. Durante este periodo se estableció la forma en que el equipo debería trabajar, la metodología a utilizar, así como los requerimientos que debía cumplir la entrega para dar por culminado y aprobado el proyecto de grado.

La segunda parte es la parte de desarrollo e implementación, durante este periodo el equipo tuvo reuniones semanales en donde se designaban tareas y se mostraban avances de lo realizado. Con el director se tenían reuniones quincenales en las cuales se le presentaban los desarrollos y se escuchaban sugerencias acerca de estos para aplicar correcciones de la manera más pronta posible y poder continuar con el desarrollo general.

Dinero.

Al principio se pretendió hacer un aplicativo *stand-alone*, pero a lo largo del tiempo, el equipo notó que para una máquina de recursos medios, estas se veían cargadas por los servicios corriendo, y por ende no se lograba un buen rendimiento. Por lo que fue necesario separar el sistema en diferentes nodos para luego proponer un sistema distribuido que en el mejor de los casos fuera desplegado en un ambiente virtualizado en la nube con el fin de tener una escalabilidad según lo requieran las máquinas en medio de su funcionamiento.

Otros de los aspectos en que se vio restringido por dinero fue haciendo uso de las *API* de las plataformas, específicamente en el caso de *Twitter*® y *Pastebin*®, con *Twitter*® se pudo utilizar una versión gratuita que limita el número de consultas de publicaciones, y con *Pastebin*®, tomamos como alternativa hacer uso de herramientas como *Selenium* para simular una búsqueda humana y de esta forma obtener los resultados por medio de consultas sobre el HTML de la página de búsquedas.

***Captchas* y bloqueadores.**

Muchas de las páginas en las que se hacen búsquedas detectan *bots* e impiden su correcto funcionamiento, aunque durante el desarrollo y pruebas el proyecto no se vio afectado por bloqueos, es posible que, si alguna plataforma que no cuenta con *API* para hacer consultas implementa un

captcha, se requiera de un desarrollo adicional para omitirlos o evitar que se detecte la búsqueda automatizada.

Tiempos de respuesta.

Al tener que hacer búsquedas en la red *TOR* (*Deep/Dark Web*), los tiempos de respuesta de las páginas aumentan considerablemente debido a los saltos entre nodos por el cual funciona esta red. Esto se ve reflejado en la búsqueda, *crawling* y recolección de resultados generando una demora inicial al momento de comenzar la búsqueda en los distintos motores de búsqueda implementados. Esto también sumado a la metodología usada para la indexación de resultados, la cual necesita que todos los motores de destino respondan (o se agoten) antes de enviar el resultado al navegador.

Disponibilidad de las plataformas.

Debido a que a herramienta realiza búsquedas sobre los principales buscadores de la *Deep Web*, y algunas plataformas sociales. Se ve limitado a la disponibilidad de estos servicios y su correcto funcionamiento. Si en dado caso alguno de los buscadores o plataformas deja de funcionar o cambian la estructura en la que presentan los resultados, la herramienta DDW&SC deberá ser intervenida y adaptada a los cambios realizados.

Especificación funcional

Como se mencionó en el primer numeral de esta sección, se trabajaron con historias de usuario para establecer los requerimientos del sistema. Estos requerimientos fueron presentados, validados, priorizados y aceptados por el director.

A continuación, en la Tabla 3. Requerimientos funcionales principalesTabla 3, se listan y detallan los principales requerimientos desarrollados e implementados para el sistema DDW&SC.

Tabla 3. Requerimientos funcionales principales

ID	HISTORIA DE USUARIO	ACTOR	ESCENARIO EXITOSO
HU-1	Como sistema quiero perfilar la búsqueda del usuario para entregar resultados más precisos.	Sistema	1. El usuario debe seleccionar máximo tres (3) categorías de su interés. 2. El sistema relaciona los resultados con las categorías. 3. El sistema clasifica los resultados según la relación.
HU-2	Como usuario quiero realizar una búsqueda con términos para encontrar resultados en la *Deep/Dark Web.*	Cliente	1. El usuario ingresa un término, términos o frase para la búsqueda. 2. El usuario selecciona el botón "Buscar". 3. El sistema inicia el proceso de búsqueda en *Deep Web.* 4. El sistema muestra los resultados de la búsqueda.
HU-4	Como sistema quiero utilizar distintos buscadores para encontrar resultados en la *Deep/Dark Web*.	Sistema	1. Se inicia un proceso de búsqueda por cada buscador. 2. Se aplican los filtros (si los hay) a medida que se encuentran resultados. 3. Se consolidan los resultados para almacenarlos.
HU-5	Como usuario quiero realizar una búsqueda con términos para encontrar resultados en *Twitter®, Reddit® y Pastebin®.*	Cliente	1. El usuario ingresa un término, términos o frase para la búsqueda. 2. El usuario selecciona el botón "Buscar".

Tabla 4. (continuada).

ID	HISTORIA DE USUARIO	ACTOR	ESCENARIO EXITOSO
HU-5	Como usuario quiero realizar una búsqueda con términos para encontrar resultados en *Twitter®, Reddit® y Pastebin®.*	Cliente	3. El sistema inicia el proceso de búsqueda en *Twitter®, Reddit® y/o Pastebin®.* 4. El sistema muestra los resultados de la búsqueda.
HU-6	Como usuario quiero iniciar el proceso de *crawling* partiendo desde un resultado o URL.	Cliente	1. El usuario ingresa una URL de partida o selecciona un resultado encontrado. 2. El usuario selecciona el botón "Iniciar *Crawling*". 3. El sistema muestra los resultados del proceso de *crawling*.

Estos requerimientos pueden ser consultados con más detalle en el Anexo 1 – Especificación de Historias de Usuario, y toda su ingeniería realizada para llegar al resultado final en el documento SRS.

Adicionalmente, se tuvo en cuenta algunos requerimientos no funcionales para mejorar la experiencia de usuario, confiabilidad y usabilidad del sistema. Estos requerimientos se muestran en la Tabla 5. Requerimientos no funcionales Requerimientos no funcionales**Error! Reference source not found.**.

Tabla 5. Requerimientos no funcionales

ID	REQUISITO	Actor	ESPECIFICACIONES Y RESTRICCIONES
NF-1	Se debe proveer una guía para puesta en marcha del sistema.	Sistema	El sistema cuenta con una guía o procedimiento para facilitar la instalación de esta al usuario.
NF-2	El usuario debe poder acceder a todos los componentes de la herramienta en menos de tres (3) clics.	Usuario	La usabilidad es importante, y se debe tener los componentes desde el menú principal a menos de tres (3) clics.
NF-3	El usuario debe aprender a manejar la herramienta a lo mucho en diez (10) minutos.	Usuario	Se debe hacer la interfaz lo más intuitiva posible, y con ayuda del manual de usuario se especificará de manera sencilla el funcionamiento. Logrando una curva de aprendizaje rápida.
NF-4	La herramienta debe permitir recuperar el estado en caso de fallo.	Sistema	La constante escritura en el proyecto permite que se pueda retomar la búsqueda sin mayores complicaciones. Estas escrituras no deben afectar el rendimiento de la herramienta.

De acuerdo con los requerimientos funcionales descritos anteriormente, se muestra mediante diagramas de BPMN el flujo del proceso a alto nivel que se realiza para la ejecución de los mismos.

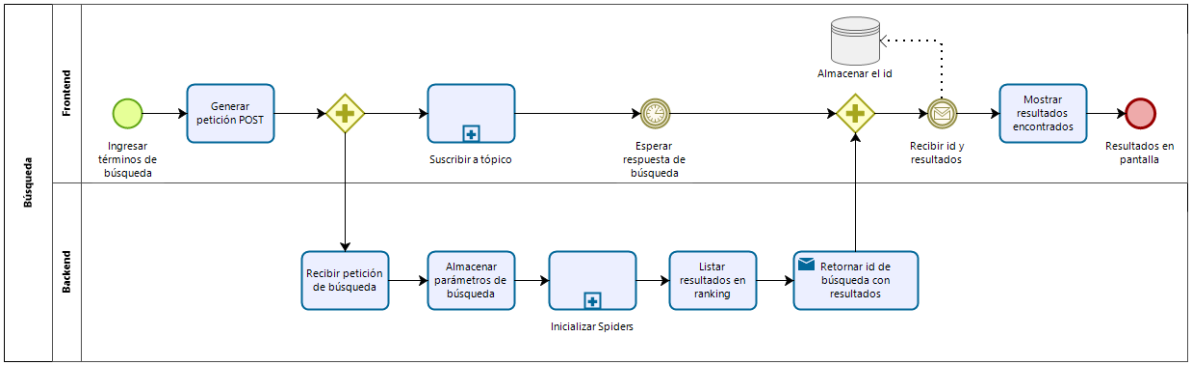

Figura 1. Diagrama BPMN búsqueda DDW&SC

El proceso de búsqueda inicia cuando el cliente por medio de algún navegador web accede al sistema y realiza una búsqueda lo que produce el envío una petición POST, esta petición genera dos (2) tareas en paralelo, la primera es suscribir al tópico el cual es descrito en Figura 2. Diagrama BPMN suscripción a tópico y la segunda en el servidor (*Backend*) cuando se recibe una petición del cliente, en dicha petición viajan los parámetros seleccionados por el usuario como lo son:

Tabla 6. Parámetros de búsqueda

Parámetro	Obligatorio	Descripción
Términos a buscar	Si	Es la palabra o frase que digita el usuario en la barra de búsqueda, el principal objetivo para buscar resultados.
Buscadores para utilizar	Si	Es la lista de buscadores con los cuales el usuario desea hacer la búsqueda entre las opciones a elegir están ***Torch, Ahmia, Not Evil, VisiTOR, Twitter®, Reddit®*** **y** ***Pastebin®***. Si el usuario no selecciona ninguno, por defecto se realiza la búsqueda en todos.
Perfilamiento	No	Consta de tres (3) campos de texto en donde el usuario puede incluir palabras

Tabla 7. (continuada).

Parámetro	Obligatorio	Descripción
Perfilamiento	No	Es la palabra o frase que digita el usuario en la barra de búsqueda, el principal objetivo para buscar resultados.

Estos parámetros de búsqueda (Tabla 6. Parámetros de búsqueda) son almacenados en el servidor y basado en ellos se inicializan los *spiders* requeridos, los cuales se describen en la Figura 3. Diagrama BPMN inicialización de *spiders*.

Una vez los *spiders* han realizado la búsqueda en las distintas plataformas, el servidor retorna un id de búsqueda al cliente el cual le permite acceder a los resultados y solicitarlos de forma paginada según como el usuario seleccione en la página de resultados.

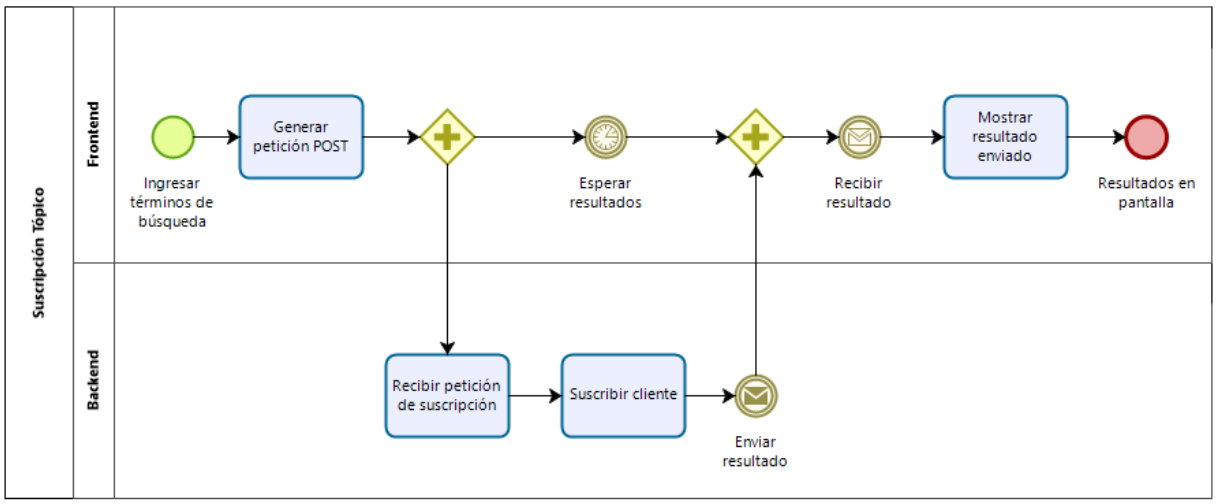

Figura 2. Diagrama BPMN suscripción a tópico

La suscripción se realiza con el objetivo de poder recibir mensajes o resultados de forma asíncrona sin tener que enviar una petición previa para cada resultado. Es utilizado para los casos de resultados acertados o de relevancia y para recibir los resultados cuando se realiza el proceso de *crawling*

Figura 4. Diagrama BPMN *crawling*. Para la suscripción es necesario que el navegador genere una petición POST estableciendo el tópico al cual desea suscribirse dentro del *payload* de esta.

Cuando la petición es recibida en el servidor, el cliente es agregado a la lista de suscriptores y en el momento en que se encuentre un resultado, este es enviado al tópico de tal manera que el navegador lo recibe y lo muestra en la página de resultados de inmediato.

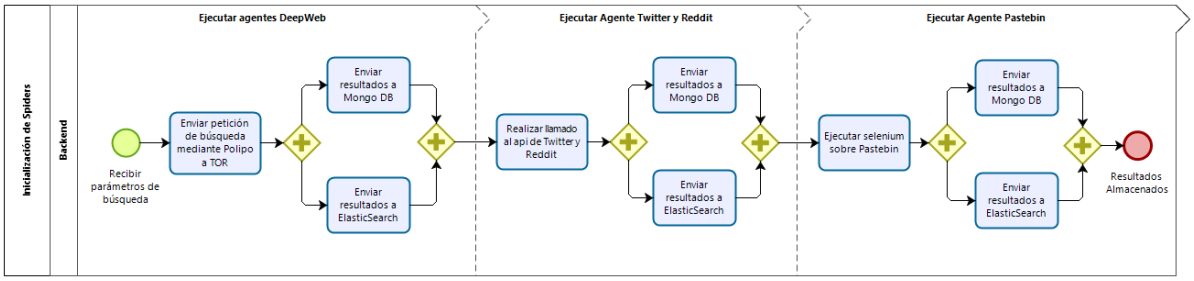

Figura 3. Diagrama BPMN inicialización de *spiders*

En la Figura 3. Diagrama BPMN inicialización de spiders, se muestra el proceso que es llevado a cabo cuando se necesita inicializar algún *spider* para buscar en las distintas plataformas. En esta descripción se toma como ejemplo el caso de búsqueda con todos los *spiders*; sin embargo, cuando no se seleccionan todos, el proceso es muy similar con la diferencia en que se omiten los que no fueron seleccionados por el usuario.

Primero se hace la búsqueda en los buscadores de *Deep Web*, para esto es necesario establecer conexión con la red *TOR* por medio del *proxy Polipo*, cuando se ha establecido se inicia el *spider* asociado y todos los resultados encontrados son enviados a Mongo DB® para su almacenamiento y a *Elasticsearch*® para su procesamiento de texto.

Luego se ejecutan las plataformas que cuentan con un *API*, se envían las consultas y de la misma forma, se almacenan los resultados en Mongo DB® y *Elasticsearch*® finalmente se realiza la búsqueda en *Pastebin*®, que requiere una forma diferente a la de los *spiders*. Hace uso de la librería

Selenium para acceder a la página de resultados y por medio del HTML se extraen los resultados encontrados en ella.

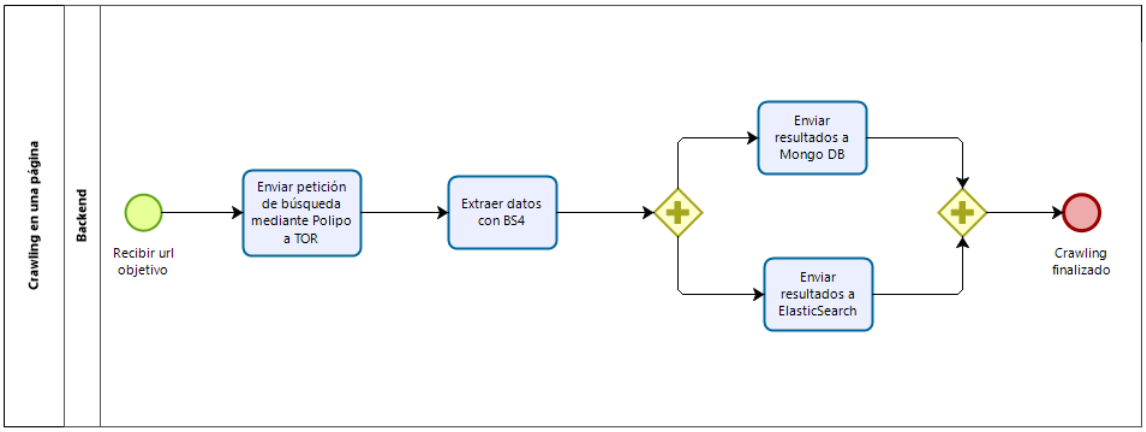

Figura 4. Diagrama BPMN *crawling*

Otra de las características principales del sistema es el proceso de *crawling*, este proceso puede iniciarse de dos (2) formas.

La primera, cuando el usuario accede a los detalles de un resultado encontrado e inicia el *crawling* a partir de este. Se despliega un campo en donde van apareciendo todas las URL encontradas con su título, *referer* y descripción hasta que se encuentre la cantidad máxima de resultados seleccionada por el usuario.

La segunda forma de iniciar este proceso es cuando el usuario desea hacer *crawling* a partir de una URL en específico, se dirige a la página de búsqueda avanzada, selecciona la cantidad máxima de resultados, provee una URL de partida y hace clic en iniciar *crawling*, por lo que el sistema repite el mismo procedimiento descrito anteriormente.

En ambos casos el proceso se maneja mediante suscripciones y los resultados son generados en tiempo real a medida que van siendo encontrados.

DISEÑO DE LA SOLUCIÓN

En esta sección del documento se muestra la solución a la que finalmente se llegó para cumplir con el objetivo del proyecto de grado. Se detalla la arquitectura seleccionada y cómo se comunican los diferentes componentes durante el funcionamiento de la herramienta DDW&SC.

Selección de herramientas

Para el desarrollo del metabuscador se planteó apoyarse en varias herramientas que facilitaran el desarrollo del trabajo. Fundamentalmente se dividieron en las siguientes:

Web Scraping.

Una parte fundamental del metabuscador consiste en la agregación de resultados de múltiples buscadores, es decir la obtención de resultados de los principales motores de búsqueda presentes en la *Deep Web*. La obtención de estos resultados se puede realizar manualmente por medio de *xpaths* y expresiones regulares sobre las páginas. Existen librerías y herramientas para lograr este objetivo como lo son *Beautiful Soup* y *Selenium*, o frameworks más completos como lo es *Scrapy*.

- **Beautiful Soup:** Es una biblioteca de Python para analizar documentos HTML. Esta biblioteca crea un árbol con todos los elementos del documento y puede ser utilizado para extraer información. Por lo tanto, esta biblioteca es útil para realizar *Web Scraping* (Crummy, 2019).
- **Scrapy:** Es un *framework* para realizar funciones de *scraper* como *crawling* sobre sitios web y extraer datos estructurados que pueden utilizarse para una amplia gama de aplicaciones útiles, como la minería de datos, el procesamiento de información o los archivos históricos (Scrapy, 2008b)**.**

- **Selenium:** Es una suite de herramientas destinadas a la automatización aplicaciones web con fines de prueba, pero ciertamente no se limita a eso. Además permite realizar acciones como la obtención de contenidos de las páginas web (Selenium, 2007).

Para estas herramientas, durante la fase de desarrollo se encontró cuál era la adecuada para cada una de las fuentes de información. Por un lado, para la obtención de información de *Pastebin*®, que puede ser listada y fácilmente accesible vía *xpaths*, al momento de utilizar herramientas de alto nivel como *Scrapy* harían que la fuente detectara las mismas y no retornara resultados (Towardsdatascience, 2019).

Tabla 8. Relación fuente por herramienta

Fuente	Herramienta	Justificación
Deep Web engines (**NotEvil, Visitor, Ahmia y Torch**)	*Scrapy*	La mayoría de las páginas en la *Deep Web* crecen de una estructura compleja, y los buscadores no son la excepción. Módulo de *middleware* que permite utilizar el *proxy* de *Polipo* para el uso de ***Socks*.** *Framework* que se encarga de la separación de cargas.
CommonAgent	*Scrapy*	*Scrapy* provee la solución *crawlerspyder* para hacer el *crawling* sobre la página raíz (Scrapy, 2008a).

Tabla 9. (continuada).

Fuente	Herramienta	Justificación
Pastebin®	*Selenium*	Navegación similar a la humana, evitando respuestas fallidas al servidor (no resultados).
Facebook®	*Selenium*	-

Reddit® y Twitter®	*API*	*API* proveído para desarrolladores para la obtención de los resultados

Framework desarrollo web.

Es importante la selección de un *framework* que se acomode a las necesidades del proyecto. Por ello se tuvo que hacer la comparativa entre las siguientes opciones: *Django, Flask* y *Sanic©*. Entre los criterios de evaluación cabe resaltar el requerimiento asociado a la entrega en tiempo real de resultados. Lo cual implica el uso de la comunicación por *sockets* y no sólo la interfaz de una *API REST*.

El *file routing*, que permite el envío de archivos desde el servidor hacia el cliente vía *streaming* o la *API REST*. Es necesario dados los requerimientos (reportes) y (descarga de página).

Cabe establecer que se pueden utilizar tanto *frameworks* como *microframeworks*. Las principales diferencias entre estos son las funcionalidades adicionales que proveen, como el manejo de cuentas y bases de datos, que para las definiciones de la herramienta no son necesarias, y/o sustituidas por otras herramientas en la arquitectura.

- **Django**: Es un marco de trabajo de alto nivel de Python Web que fomenta el desarrollo rápido y el diseño limpio y pragmático. Construido por desarrolladores experimentados, se encarga de gran parte de las molestias del desarrollo web. Es gratuito y de código abierto (Django Project, 2019).
- **Flask**: Es un *microframework* que se centra en la simplicidad, el minimalismo y el control del grano fino. Implementa lo mínimo, dejando al desarrollador con total libertad de elección en cuanto a módulos y complementos (Pallets Projects, 2019).

- **Sanic:** Es un *microframework* asíncrono, similar a *Flask*, una estructura muy parecida a una petaca: es pequeña, liberal y deja mucho espacio para el desarrollador. Su principal característica definitoria es su velocidad (Sanic, 2017).

Tabla 10. Comparación de *frameworks*

Framework	Simplicidad	Síncrono	Asíncrono	*File-routing*	Amplia documentación	Tipo de *framework*
Django	?	✓	✗	✓	✓	*Full-stack*
Flask	✓	✓	✗	✓	✓	*microframework*
Sanic©	✓	✗	✓	✓	?	*microframework*

Como se mencionó anteriormente, la parte asíncrona era vital para alcanzar las funcionalidades propuestas. Por ello tuvo más peso al momento de tomar la decisión del *framework* a utilizar.

A pesar de que *Django* fuera el más utilizado y uno con la documentación más extensa, al momento de implantarlo en el proyecto se vería, por un lado, subutilizado dadas sus amplias funcionalidades, y también escaso al momento de cumplir con la promesa del funcionamiento en parte asíncrono (Daniel Tomaszuk, 2018).

Flask, por otra parte, provee una solución más mínima y ajustada a la mayoría de las necesidades de la herramienta, pero también flaquea en la parte de envío de resultados ya mencionado.

Sanic©, por último, ya que está orientado a ser un *microframework* similar a *Flask*, provee funcionalidades parecidas, tanto una interfaz para implementar los *routings* necesarios para la parte consumida por el *front-end* vía REST, como el uso de llamadas asíncronas. Permitiendo el envío de información conforme es encontrada por el *back-end*.

Framework front-end.

Al momento de seleccionar un *framework* para el desarrollo de *front-end*, se tuvo en cuenta opciones como *Django*, *Angular* o *React*. Sin embargo, la primera en ser descartada fue *Django* dado que a pesar de ser *framework* bastante completo, brinda muchas funcionalidades que no iban a ser utilizadas para este tipo de proyecto lo que generaría gasto de recursos innecesarios. Entre *Angular* y *React* estaba la selección, y luego de discutir entre el equipo qué herramienta utilizar se optó por *Angular* debido a que este *framework* ya se había trabajado durante una asignatura de la carrera y se tenía un ligero conocimiento sobre este, básico, pero que ahorraría tiempo de capacitación y estudio del mismo. De esta forma se pudo iniciar paralelamente el desarrollo del *back-end* junto con el *front-end*.

Además, cabe anotar que *Angular* tiene el soporte de Google©, una gran documentación y comunidades que facilitan el desarrollo y la resolución de problemas, también cuenta con la librería de estilos *Angular Material* que hace más sencilla la construcción de aplicaciones de una sola página teniendo capacidad de adaptarse a pantallas móviles, en tal caso que en un trabajo futuro se deseara incluir soporte completo del sistema DDW&SC para estos dispositivos (Angular, 2016).

Gestión documental.

Una de las características del *metabuscador* es el tratamiento que se le dará a la información en nuestro caso, almacenamiento y posteriormente la búsqueda sobre los mismos. Por lo que fue necesario una base de datos no relacional ya que proveía flexibilidad a la hora de almacenar la información.

1. **Elasticsearch**: Motor de búsqueda basada en la librería de Java: *Lucene*. Es distribuida, *open-source* y de tipo *REST-ful* y es utilizada para el análisis de documentos, búsqueda de

texto completo, con casos de uso de inteligencia operacional. Opera en un entorno distribuido (ELK, 2010).

2. **MongoDB:** Base de datos no relacional en donde se almacenan todos los datos recolectados de las búsquedas e información relacionada con las sesiones de usuario. Permite mantener un registro de las búsquedas que puede ser consultado en cualquier momento (MongoDB, 2009).

Arquitectura

El sistema inicialmente se pretendió hacer *stand-alone*, pero a lo largo del desarrollo surgieron problemas de rendimiento, puesto que se requería de gran poder de procesamiento en una sola máquina para poder llevar a cabo todas las tareas. Esto obligó al equipo a tomar decisiones y cambiar toda la estructura del sistema, pasando a una arquitectura distribuida.

Arquitectura previa.

Como bien se planteó el uso de una arquitectura desplegada en varios nodos, la primera solución propuesta se puede evidenciar en la Figura 5. Arquitectura previa.

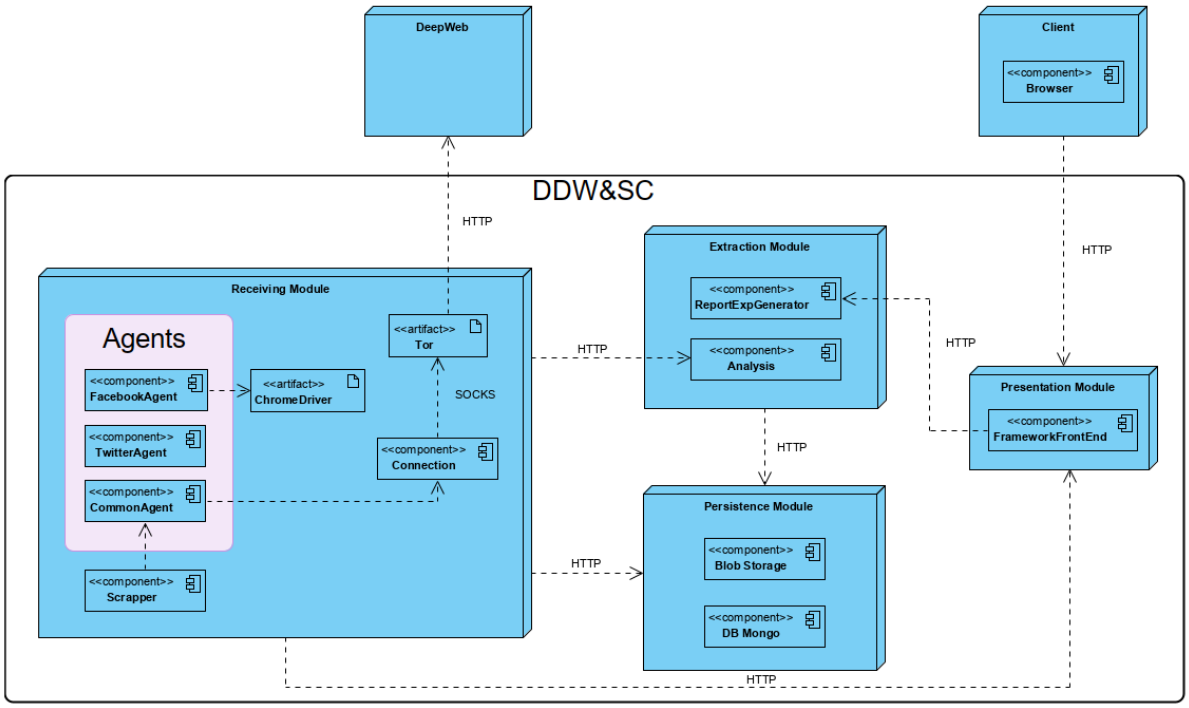

Figura 5. Arquitectura previa

Esta arquitectura se caracteriza por la especificación de cuatro grandes módulos, *Receiving*, *Extraction* y *Persistence*; a continuación, algunas consideraciones para cada uno de ellos y la razón por la cual cambiaron.

Receiving Module.

Este módulo suponía el uso de un agente por cada plataforma social, en aquel entonces no había sucedido lo expuesto en la sección Tabla 2. (continuada).

Entregable	Estándares asociados	Justificación
SRS	IEEE 830-1998 (IEEE, 1998)	El documento SRS adaptado para este proyecto está basado en el estándar IEEE 830-1998, en este documento se describen las bases establecidas entre el director (Joshsua González) y el equipo de trabajo. Se especifican todas las funcionalidades y atributos del sistema a desarrollar.

SDD	IEEE 1016-2009 (IEEE, 2009)	Este documento está basado y adaptado del estándar IEEE 1016-2009, en él se especifica la arquitectura y las interfaces lógicas y de usuario que serán diseñadas e implementadas.
Manuales de Usuario	-	Este documento es una guía de uso de la herramienta para facilitar el despliegue y brindar una ayuda al usuario final para que pueda realizar las búsquedas.
Memoria	Documento propuesto por la Pontificia Universidad Javeriana	Este documento presenta la información del proyecto de grado DDW&SC, tanto la planeación como el desarrollo de este.
Documento de pruebas	-	Con este documento se pretende mostrar toda la fase de pruebas realizadas con el fin de asegurar el correcto funcionamiento de cada uno de los módulos desarrollados, como también del producto final entregado, en el cual se asegure una salida satisfactoria luego de ingresar búsquedas por parte del cliente final.
Aplicativo	-	Finalmente, el equipo de trabajo entregará un producto de software acorde a los requerimientos especificados y consignados en el documento SRS.

CONTEXTO DEL PROYECTO para *Facebook®*, por lo cual el agente de este aún se encontraba presente. Adicionalmente, para la extracción de la información proveniente de las fuentes de la *Deep Web*, se planteó utilizar un agente desarrollado por el equipo y también establecer las funcionalidades necesarias para la comunicación a esta red (Tabla 2. (continuada).

Entregable	Estándares asociados	Justificación

SRS	IEEE 830-1998 (IEEE, 1998)	El documento SRS adaptado para este proyecto está basado en el estándar IEEE 830-1998, en este documento se describen las bases establecidas entre el director (Joshsua González) y el equipo de trabajo. Se especifican todas las funcionalidades y atributos del sistema a desarrollar.
SDD	IEEE 1016-2009 (IEEE, 2009)	Este documento está basado y adaptado del estándar IEEE 1016-2009, en él se especifica la arquitectura y las interfaces lógicas y de usuario que serán diseñadas e implementadas.
Manuales de Usuario	-	Este documento es una guía de uso de la herramienta para facilitar el despliegue y brindar una ayuda al usuario final para que pueda realizar las búsquedas.
Memoria	Documento propuesto por la Pontificia Universidad Javeriana	Este documento presenta la información del proyecto de grado DDW&SC, tanto la planeación como el desarrollo de este.
Documento de pruebas	-	Con este documento se pretende mostrar toda la fase de pruebas realizadas con el fin de asegurar el correcto funcionamiento de cada uno de los módulos desarrollados, como también del producto final entregado, en el cual se asegure una salida satisfactoria luego de ingresar búsquedas por parte del cliente final.
Aplicativo	-	Finalmente, el equipo de trabajo entregará un producto de software acorde a los requerimientos especificados y consignados en el documento SRS.

CONTEXTO DEL PROYECTO).

Extraction Module.

No existía un motor de búsqueda asociado para la generación de los reportes, así como para el análisis no se optó por el uso de artefactos adicionales. Dado el alcance del proyecto, se cambió por el uso de *ElasticSearch*®.

Persistence Module.

En esta arquitectura, el almacenamiento se realizaría en la nube en un *Blob Storage*. Para almacenar en gran medida las páginas y sus contenidos. Una mirada más en detalle de los *blobs* a almacenar, y la naturaleza actual de la herramienta (*in-house*) se descartó la idea.

Arquitectura seleccionada.

La arquitectura seleccionada se ajustó a las necesidades del proyecto, bajo el modelo de cliente servidor (School of Computing Universiti Utara Malaysia Kedah, Malaysia & Oluwatosin, 2014) debido que cada modulo actúa como cliente y servidor, dado el paso de los mensajes entre todos y cada uno de ellos. Este patrón se repite por toda la solución, tanto del usuario al *front-end* como del *front-end* al modulo de extraccion; y para servidores web como para servidores de archivos.

Para mantener una buena escalabilidad frente al incremento de peticiones en el sistema, se sugiere tenerlo desplegado en un servicio de *cloud* como *Microsoft© Azure* o *Amazon© Web Services*. A continuación se describe la arquitectura diseñada y se detalla cada uno de sus nodos con su funcionamiento.

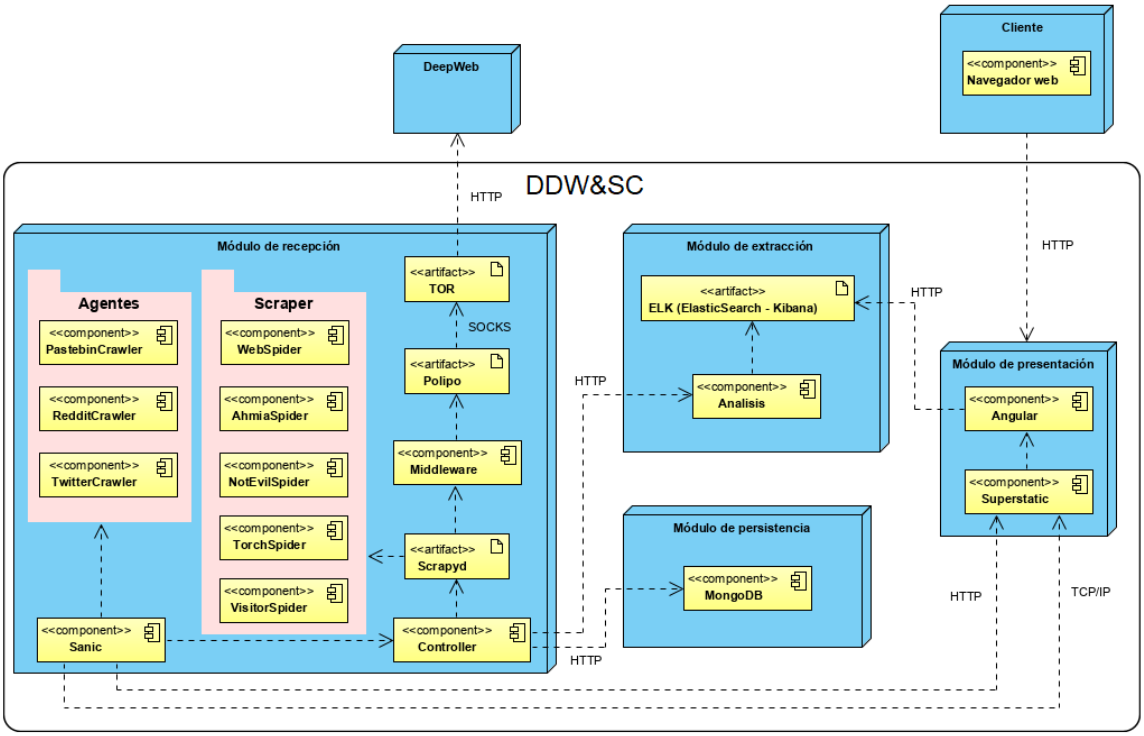

Figura 6. Arquitectura del sistema

Para esta nueva arquitectura se dividieron las tareas en cuatro (4) módulos (nodos) principales:

- **Módulo de Recepción:** Este módulo se encarga de lo relacionado a la inicialización de la búsqueda y búsqueda de resultados haciendo uso de los diferentes buscadores o agentes. Es donde también se establece la conexión con *Deep Web* y se envían los resultados a la base de datos y a *ElasticSearch®*. Contiene el servidor web Sanic© que permite establecer la interfaz de comunicación con el *front-end* por medio de peticiones HTTP y conexiones por *sockets* TCP/IP, para enviar datos de forma asíncrona para el caso del *crawling* y resultados acertados, los cuales se manejan con suscripciones.
    - **Agentes:**
        - **PastebinCrawler:** Componente encargado de hacer búsquedas dentro de la página www.pastebin.com . Su implementación se apoya en *Selenium*, el cual realiza la búsqueda como lo haría un humano. Debido a que esta página realiza las

búsquedas por medio del *API* de Google©, el cual no responde a las peticiones que provengan de un agente diferente al de un *browser*.

- **TwitterCrawler:** Componente que realiza búsquedas en la red social *Twitter*® haciendo uso de las *API* disponibles para Python (*twython* y *tweepy*).
- **RedditCrawler:** Componente que realiza búsquedas en *Reddit*®, el cual también fue apoyado por el *API* de *Reddit*®.

- **Scraper:**
  - **WebSpider:** Componente que se encarga de obtener datos sobre una URL semilla y extraer el texto, además de visitar los enlaces contenidos en cada URL para buscar más información. El límite de visitas está ajustado por la configuración del usuario en la que tiene la posibilidad de seleccionar un valor máximo de doscientos (200) resultados.
  - **AhmiaSpider:** Componente encargado de hacer la búsqueda y obtener los resultados del buscador de *Deep Web* **Ahmia**.
  - **NotEvilSpider:** Componente encargado de hacer la búsqueda y obtener los resultados del buscador de *Deep Web* **Not Evil**.
  - **TorchSpider:** Componente encargado de hacer la búsqueda y obtener los resultados del buscador de *Deep Web* **Torch**.
  - **VisitorSpider:** Componente encargado de hacer la búsqueda y obtener los resultados del buscador de *Deep Web* **Visitor**.
- **Scrapyd:** Aplicación para desplegar y ejecutar *spiders* de *Scrapy*. Permite desplegar los proyectos y controlar los *spiders* por medio de un *API REST*.

- **Controller:** Como su nombre lo indica, este componente se encarga de administrar todo lo relacionado con generadores de resultados, se encarga de enviar peticiones al *API* de *Scrapyd* y es quien recibe los resultados para almacenarlos en Mongo DB® y enviarlos a *Elasticsearch*® para su correspondiente tratamiento.
- **Middleware:** Este componente se encarga de las conexiones del servidor con otros artefactos, como lo es *Polipo* para las peticiones a la *Deep Web*, y para la información enviada por el *controller* para *Elasticsearch*® y Mongo DB®.
- **Polipo:** Servidor *proxy* de reenvío y almacenamiento que facilita la conexión de la herramienta hacia la red *TOR*, en donde se realizan la mayoría de las búsquedas.
- **TOR:** Binario ejecutable del navegador *Tor Browser*, usado para hacer las peticiones a las páginas de la red *TOR* y navegar en ella.

- **Módulo de extracción:** En este módulo se realiza el análisis de los resultados de *Elasticsearch*® para poder hacer el *ranking* y encontrar los resultados con mayor relevancia para el usuario.
  - **ELK:** (*Elasticsearch*®, Kibana©) es un conjunto de herramientas para el análisis, consulta y visualización de datos. En este proyecto se usó solamente *Elasticsearch*® y Kibana©. Con *Elasticsearch*® se realizan las consultas, relación y análisis de los datos extraídos los cuales son enviados al *front-end*, haciendo uso de Kibana© para mostrar un *dashboard* con el resumen de los resultados.
  - **Análisis:** Componente encargado de enviar las peticiones o consultas necesarias a *Elasticsearch*® y retornarlas al *controller* del módulo de recepción para que finalmente sean enviados los resultados hacia el cliente.

- **Módulo de persistencia:** Es en este módulo donde se almacenan todos los resultados encontrados y las sesiones del usuario en la base de datos Mongo DB®.
    - **MongoDB:** Base de datos no relacional en donde se almacenan todos los datos recolectados de las búsquedas e información relacionada con las sesiones de usuario. Permite mantener un registro de las búsquedas que puede ser consultado en cualquier momento.
- **Módulo presentación:** Contiene un servidor web que permite la conexión directa con el navegador web del usuario y con el *front-end* programado con el *framework* de *Angular*.
    - **Angular:** *Framework typescript* para el desarrollo de aplicaciones web de una sola página soportado por Google ©.(«Angular—Architecture overview», 2016)
    - **Superstatic:** Es un servidor web ligero, soportado por *Firebase* con soporte para HTML5(«Superstatic», 2018).
- **Cliente:** Navegador del cliente por el que accede a la plataforma web, se recomienda acceder directamente desde el navegador *Tor Browser*, para que los enlaces de los resultados encontrados puedan ser accedidos directamente.

DESARROLLO DE LA SOLUCIÓN

Metodología

Para el desarrollo de DDW&SC se siguió la metodología DAD, propuesta en el Anexo 2 – Project Management Plan. En el cual se le dio prioridad al desarrollo de las secciones criticas primero. Por ejemplo al módulo de conexión a la *Deep Web*, que en un principio significó el desarrollo de todo un módulo y posteriormente dado el cambio de la arquitectura, terminó siendo apoyado en el *framework* de *Scrapy*.

Para el manejo de actividades del equipo se llevó un tablero en *Trello* con las tareas asignadas, así como el uso de reuniones virtuales ya que por temas de localización y disponibilidad fue mejor llevarlas de esta manera. Para cada una de estas reuniones se generó un acta como se planteó en el documento del plan de proyecto, si bien se ideó la reunión con una periodicidad establecida, esta algunas veces se tuvo que cambiar por la etapa del proyecto.

Implementación

La implementación de los módulos cuenta con una documentación en la plataforma *GitBook*, que puede ser consultada en el siguiente enlace: *https://ddw-sc.gitbook.io/*. En este se encuentra una descripción general del proyecto, al igual que una guía para la ejecución de la herramienta.

Como se ha descrito anteriormente se desarrolló una herramienta similar a un metabuscador que permitiera hacer búsquedas parametrizadas sobre la *Deep Web*, y se apegó el desarrollo a lo propuesto en el cronograma inicial. Las funcionalidades desarrolladas, en el orden que fueron implementadas son las siguientes:

Módulo de recepción.

Este es el módulo principal de la herramienta, aquí se da el manejo de las peticiones entrantes al servidor. La puesta en marcha de este módulo se basó principalmente en la configuración inicial para el *web server*. En un principio se hizo un desarrollo con el *framework* de *Flask*, con las funcionalidades básicas de búsqueda. Más adelante durante el desarrollo se presentó un impedimento de la parte de la integración entre este y el módulo de presentación (*Angular*). Por lo cual, el equipo se vio obligado a cambiar de servidor. Sanic© por otro lado permitió solventar ese impedimento sin gran configuración adicional.

Conexión a Deep Web.

Para la conexión hacia la *Deep web*, como se mostró anteriormente en la arquitectura antes de *Scrapy* era necesario implementar un conjunto de funciones que permitieran la obtención de resultados por parte del *back-end*. Lo que implicaba la desviación de las peticiones al servicio que se ejecuta de *TOR*. Este módulo fue reemplazado por la configuración que se hace para *Scrapy* y el *middleware* de conexión utilizando el *proxy server* de *Polipo* (***socks***).

Agentes.

La definición de distintos agentes discriminados por cada fuente permite compartir una interfaz común para la comunicación hacia las otras capas de la herramienta. Este módulo tiene una relación con los *drivers*, haciendo uso de los ejecutables binarios para el proceso de *crawling*.

CrawlerTwitter.

Este agente no se apoya en artefactos adicionales como binarios o *Jars* para su uso, debido a que se maneja el *API* de *Twitter*® que facilita la búsqueda de publicaciones en esta red social. La *API* se encarga de hacer la búsqueda y retornar los tweets más relevantes según corresponda.

WebCrawler.

Este agente hace uso de la configuración de la conexión y del análisis de los contenidos para realizar el *crawling* según un término ingresado. Este realizará la búsqueda sobre los demás buscadores y sobre una URL raíz, en el caso que aplique.

Pastebin.

Agente que hace uso del *driver* de *Selenium* para realizar el *crawling*. Este, según los términos ingresados, simula el comportamiento humano gracias al *gecko driver* (*driver Selenium* para *Mozilla Firefox*) o *Chrome driver* (*driver Selenium* para Google Chrome©).

Reddit.

Al igual que *Twitter*®, este agente no utiliza los drivers para llevar a cabo el proceso de *crawling*. En su lugar se realiza el respectivo consumo de la *API* de *Reddit*® por medio del *framework* ***PRAW*** (*Python Reddit API Wrapper*), en la cual se crea una instancia de una *subReddit* y se parametrizan los campos a buscar como: título, comentarios, puntuación, id, entre otros.

Scraper.

La razón de tener agentes distintos para cada fuente de información, en este caso cada uno de los buscadores se debe a que no todos ofrecen una interfaz uniforme de la cual consumir. Estos agentes se apoyan de *Scrapy*, una librería para la extracción de información de una página web, bien sea la *metadata* o la generación de objetos a partir del HTML (en este caso cada resultado).

Integración con Elasticsearch® y Kibana©.

Como se mencionó en la sección de herramientas, el uso del motor de búsqueda para mejorar los resultados encontrados era necesario. Para esta integración fue necesaria la creación de un nuevo *pipeline*, similar al utilizado para Mongo DB®, es decir al momento de encontrar cada uno de los ítems, este sería almacenado como un documento en *ElasticSearch*®. Por otro lado, para el envío de esta información como también de las otras fuentes externas a *Scrapy*, se implementó una funcionalidad dedicada al envío de estos a *ElasticSearch*®.

Una vez almacenados estos documentos, es posible realizar una indexación y visualización de esta vía Kibana©. Se generaron dos (2) *dashboard*, uno para el seguimiento de las búsquedas y otro para la visualización de la búsqueda en específico.

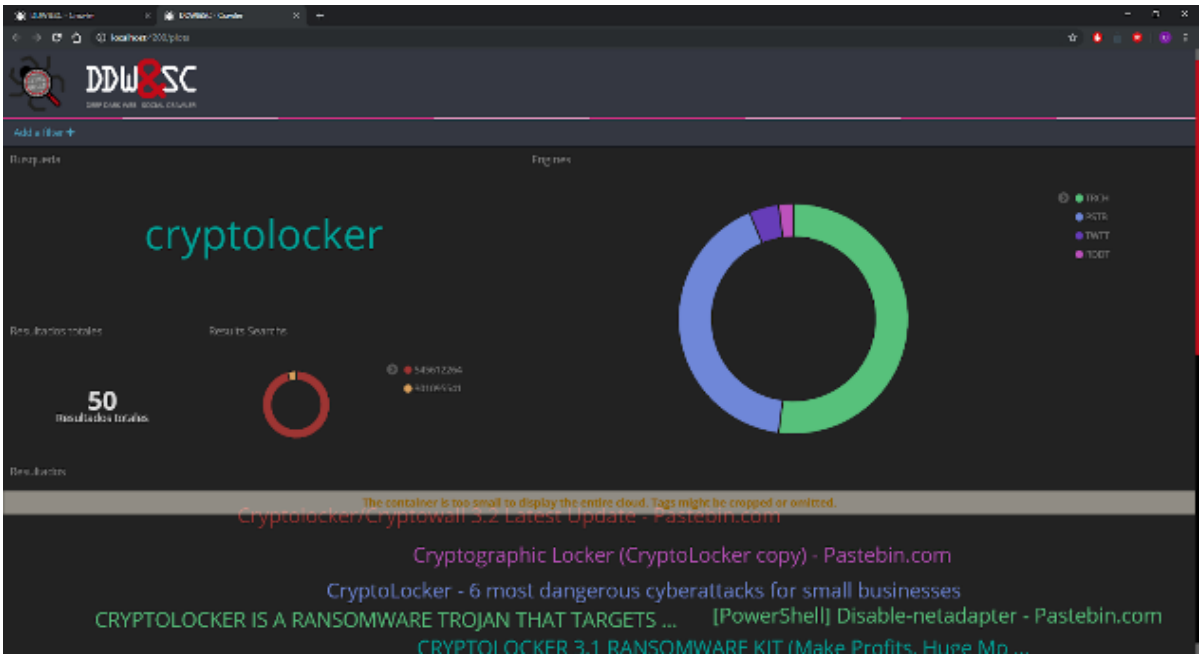

Figura 7. Dashboard para el resultado en específico

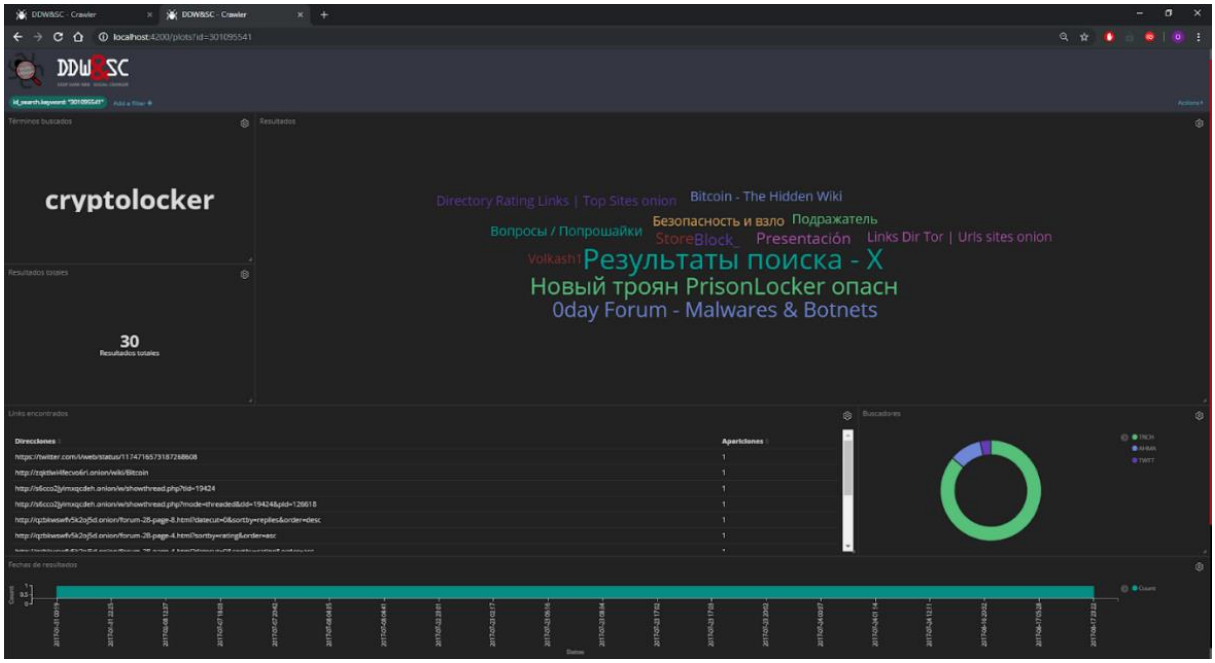

Figura 8. Dashboard búsqueda

Producto final

De cara al usuario, las funcionalidades provistas por el *back-end* se pueden utilizar desde el *front-end*, es decir la interfaz con la que el usuario interactúa con la herramienta DDW&SC, la cual tiene un diseño sencillo que le permite al usuario usarlo sin mayor dificultad. Este se compone de dos (2) secciones, la del establecimiento de la búsqueda como la de la visualización y manejo de los resultados.

Para mayor detalle de la solución y de la interfaz, por favor revisar el manual de usuario disponible en el Anexo 4 – Manual de usuario.

Búsqueda.

Para la búsqueda sobre el metabuscador, se muestra esta primera página (*Figura 9.* Página principal de búsqueda) donde se puede realizar una búsqueda sencilla como agregar términos adicionales o restringir la misma a sólo unos buscadores.

El diseño de la página es el mostrado en la *Figura 9.* Página principal de búsqueda donde se cuenta con el campo de entrada como con las configuraciones adicionales.

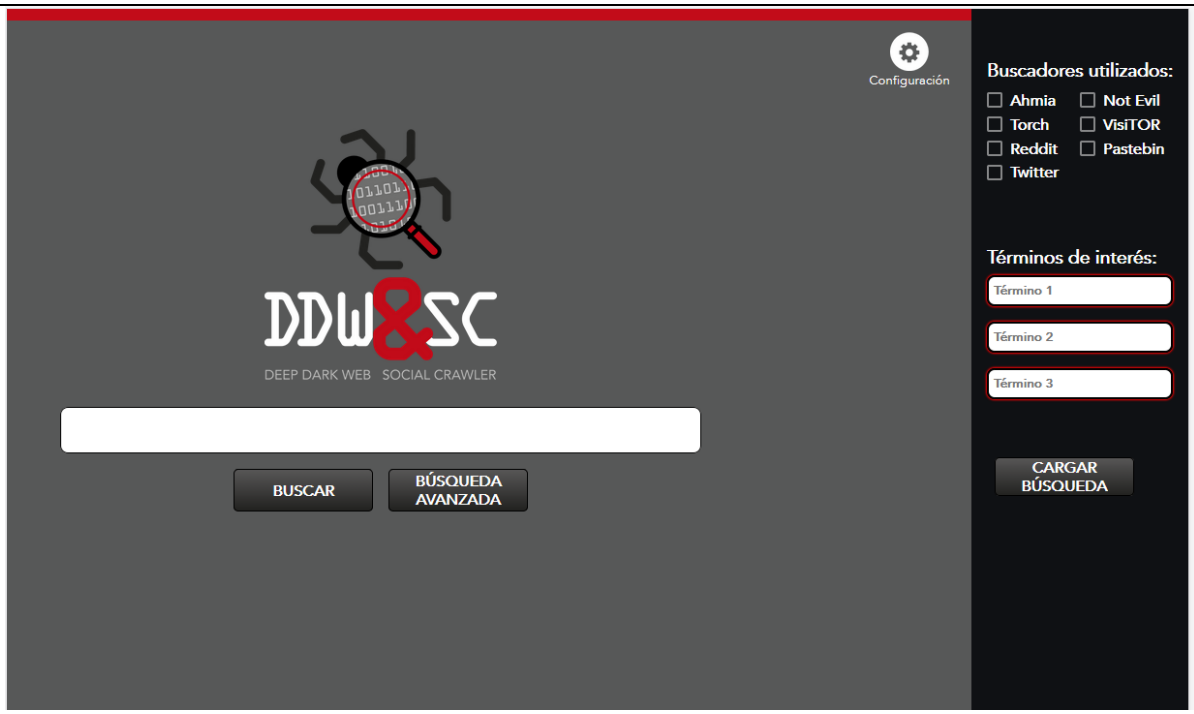

Figura 9. Página principal de búsqueda

Guiado por la historia de usuario 1: *HU-1 Como sistema quiero perfilar la búsqueda del usuario para entregar resultados más precisos.* Se establecen los parámetros adicionales de la página principal, así como toda la página mostrada en la Figura 10. Página de búsqueda avanzada que con campos adicionales permite filtrar aún más los resultados.

En esta misma página es posible ejecutar el *crawling* dado una URL raíz, respondiendo a la historia de usuario HU-*6 Como usuario quiero iniciar el proceso de crawling partiendo desde un resultado o URL*. Este mostrará al costado derecho las paginas conforme se van encontrando (*Figura 10.* Página de búsqueda avanzada).

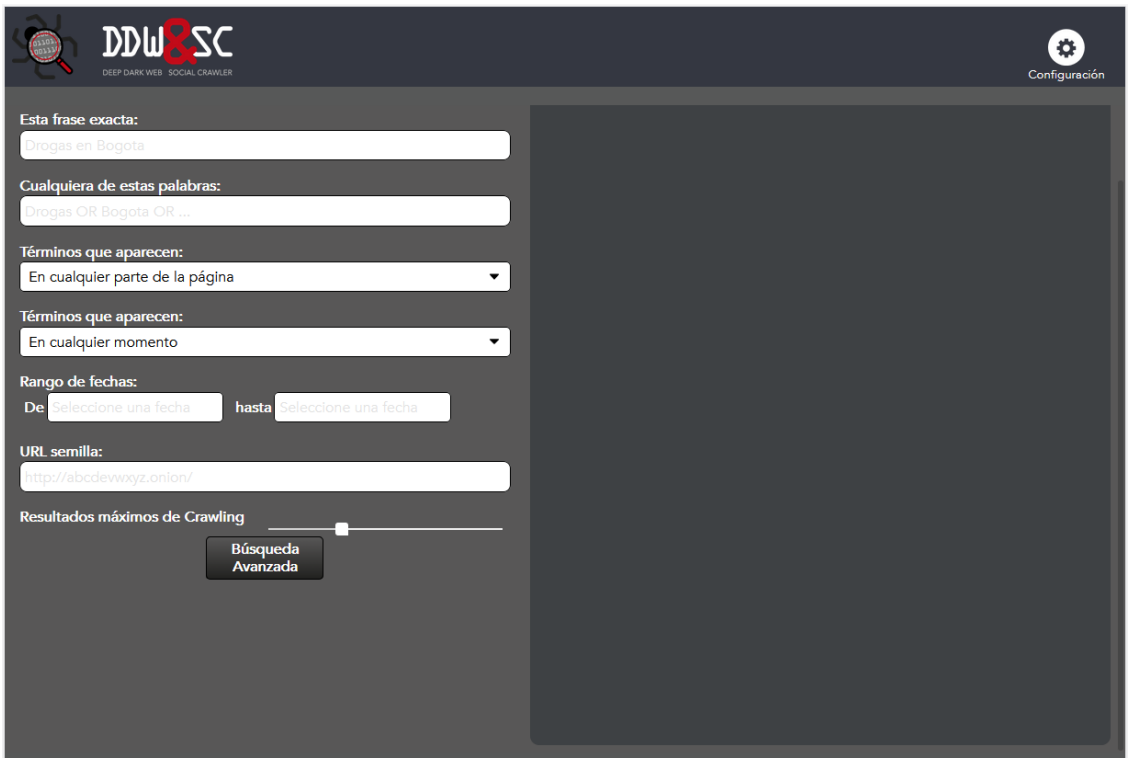

Figura 10. Página de búsqueda avanzada

Generación de reportes.

Una vez obtenidos los resultados, es posible la generación de un reporte, seleccionando los resultados a incluir. Para que el usuario genere el reporte es necesario que se seleccionen resultados, como se muestra en la Figura 11. Página de resultados.

Resultados acertados.

Debido a que dentro de los parámetros de búsqueda es posible establecer términos adicionales, se apoyó en *Elasticsearch*® para la búsqueda sobre los documentos ya alojados para así mostrarlos en las secciones de los resultados acertados. El llenado de esta sección, se hace asíncronamente conforme se filtran los resultados.

Figura 11. Página de resultados

Ver estadística.

Desde la misma vista de la Figura 11. Página de resultados, es posible visualizar el *dashboard* mencionado en la sección Integración con Elasticsearch® y Kibana©. para la visualización de los detalles de la búsqueda realizada.

Exportar Búsqueda.

Siguiendo las necesidades del proyecto, existe la posibilidad de exportar los parámetros de la búsqueda para retomarla posteriormente. Desde el inicio Figura 9. Página principal de búsqueda.

Detalle del resultado.

Por cada uno de los resultados encontrados es posible acceder a una información adicional de la página como se muestra en la Figura 12. Detalle del resultado.

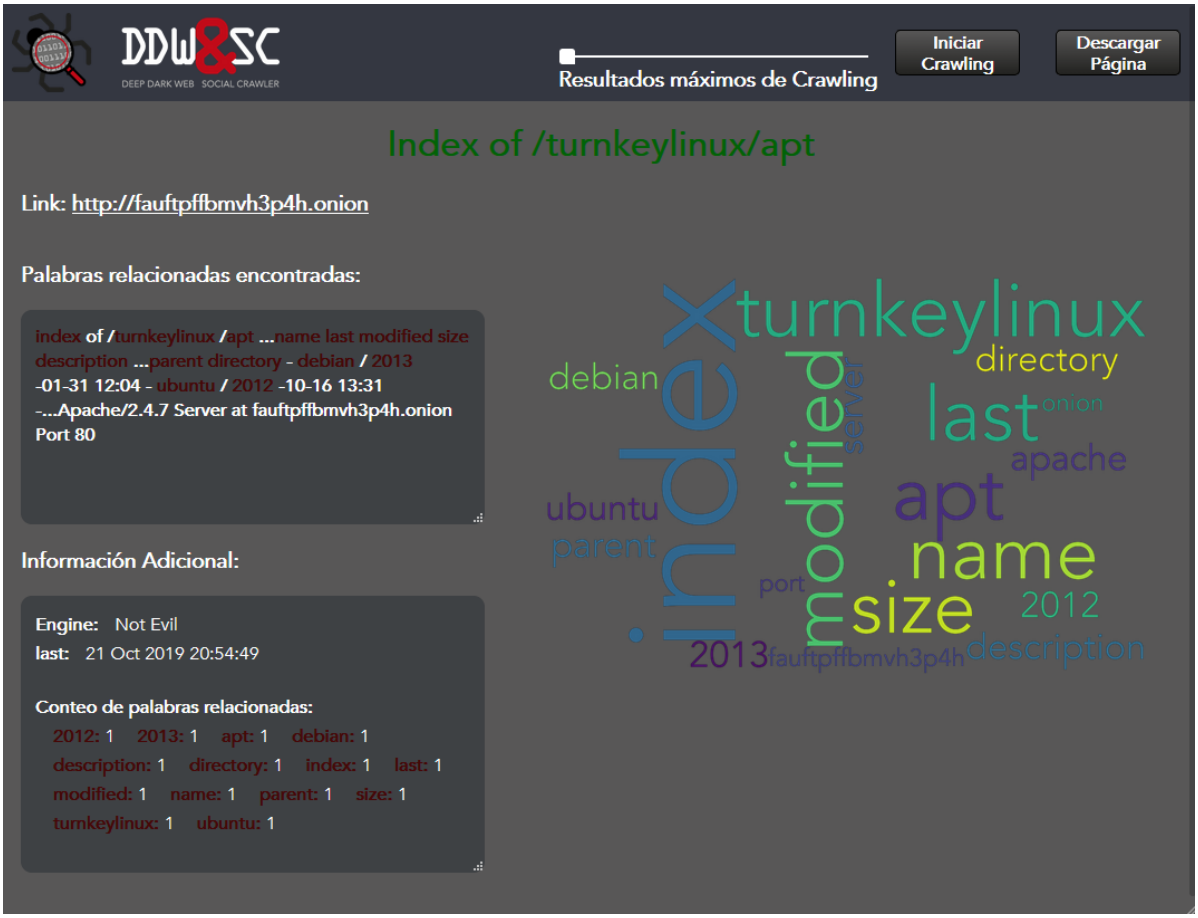

Figura 12. Detalle del resultado

Iniciar Crawling.

Igualmente, desde esta vista es posible realizar el *crawling* desde el resultado actual, conforme se encuentren resultados aparecerán en la lista como se observa en la Figura 13. Crawling sobre el resultado.

Descargar Página.

Desde esta vista, se posibilita también realizar la descarga de la página. En formato .zip, es decir el HTML y sus contenidos.

Figura 13. Crawling sobre el resultado

RESULTADOS

Automatización de pruebas

Contexto

Dado que se decidió trabajar con la metodología ágil DAD (*Disciplined Agile Delivery*) Figura 14. Metodología DAD más específicamente con *Scrum*, *Kanban* y XP, se recomienda la implementación de *agile testing*, concepto que propone una alineación con el área de desarrollo. Es decir, en cada iteración en la que se implementará una funcionalidad o lógica del negocio,

se realizarán las respectivas pruebas. La historia de usuario estará aceptada una vez sea aprobada por el *Product Owner* y el área de calidad.

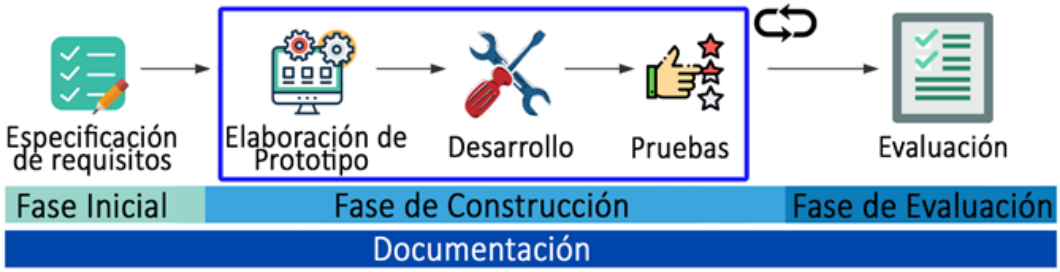

Figura 14. Metodología DAD

Como se evidencia en el gráfico anterior, el proceso de *testing* se encuentra dentro de la iteración, lo cual lo hace partícipe de cada entregable.

Dadas las condiciones del proyecto, se realizó el siguiente cuadro comparativo Tabla 11. Comparativo Agile testing y Waterfall testing para evaluar entre *agile testing* y *waterfall testing*, o *testing* cascada tradicional.

Tabla 11. Comparativo *Agile testing* y *Waterfall testing*

Agile Testing	*Waterfall testing*
Hay una planeación mínima para ejecutar las pruebas.	Proceso más estructurado, se detalla más la descripción de la fase de pruebas.
Se adapta más a proyectos pequeños.	Se adapta a cualquier tipo de proyecto.
Si se encuentran defectos al inicio del proyecto, pueden ser corregidos en el transcurso de este.	El producto es probado al final de la fase de desarrollo, cualquier cambio retrasará la puesta en marcha.
Documentación mínima requerida para iniciar las pruebas.	Requiere documentación robusta de todo el desarrollo realizado para iniciar la fase de pruebas.

Tabla 12. (continuada).

Agile Testing	*Waterfall testing*
Cada iteración tiene su propia fase de pruebas. Se pueden ejecutar las regresiones cada que se añada una nueva funcionalidad.	Las pruebas comienzan después de terminar la fase de desarrollo.
En cada iteración se entrega un mínimo producto viable (ya aceptado) al usuario final.	Todas las funcionalidades son entregadas al final de la fase de implementación y se realiza la certificación del producto completo.
Testers y desarrolladores trabajan en conjunto.	Áreas de desarrollo y QA trabajan por separado.
Las pruebas de aceptación (UAT) son parte de cada iteración.	Las pruebas de aceptación son realizadas al final del proceso de desarrollo.
El equipo de pruebas debe estar en constante comunicación con el equipo de desarrollo, para aclarar requerimientos y reglas de negocio.	El equipo de desarrollo no está involucrado en el proceso de pruebas.

Dados los puntos descritos anteriormente, se decidió por la implementación de *agile testing,* ya que al ser un proyecto pequeño es más viable tener un constante proceso de pruebas para así encontrar los defectos en el transcurso del desarrollo, que realizar la detección y corrección de defectos al final, ya que esto compromete el proceso de entrega del aplicativo, pues dentro de la metodología no se contemplan dichos tiempos para la corrección.

Al elegirse *agile testing*, se contemplan los siguientes *frameworks*:

i. Behavior Driven Development BDD

BDD es un *framework* de automatización de pruebas que facilita la comunicación entre *Stakeholders*, analistas de pruebas y desarrolladores, ya que implementa los casos de prueba en estructuras llamadas escenarios, los cuales se encuentran escritos en lenguaje *Gherkin*. Esta sintaxis similar al lenguaje natural puede ser interpretada más fácilmente por cualquier tipo de usuario (John Ferguson Smart, 2014).

ii. Acceptance Test Driven Development ATDD

ATDD mantiene la comunicación entre el cliente, los desarrolladores y el equipo de pruebas. El concepto principal de ATDD es guiar el desarrollo por las pruebas de aceptación, por dicha razón las pruebas se realizan antes del desarrollo. A diferencia de BDD, no todos los *tests* son automatizados.

iii. Pruebas exploratorias

En esta metodología, el diseño y la ejecución de las pruebas se realizan simultáneamente y su principal finalidad es el aprendizaje, orientado a la optimización de la calidad de los casos. Se basa principalmente en saber cómo funciona el producto a probar mediante la exploración de este.

Al querer implementar *agile testing* se recomienda la automatización de pruebas, ya que aumenta la efectividad, la eficiencia y el *coverage*. Con la automatización de pruebas se ahorra aproximadamente entre el 50% y 70% del tiempo invertido en pruebas manuales.

Dado que para el proyecto se requiere una constante comunicación entre el equipo de desarrollo y pruebas, se optó por la implementación de BDD. De esta manera los casos de prueba podrán ser entendidos por todos los miembros del equipo.

Proceso de automatización

Para realizar la automatización de los casos de prueba se tuvo el siguiente flujo de trabajo:

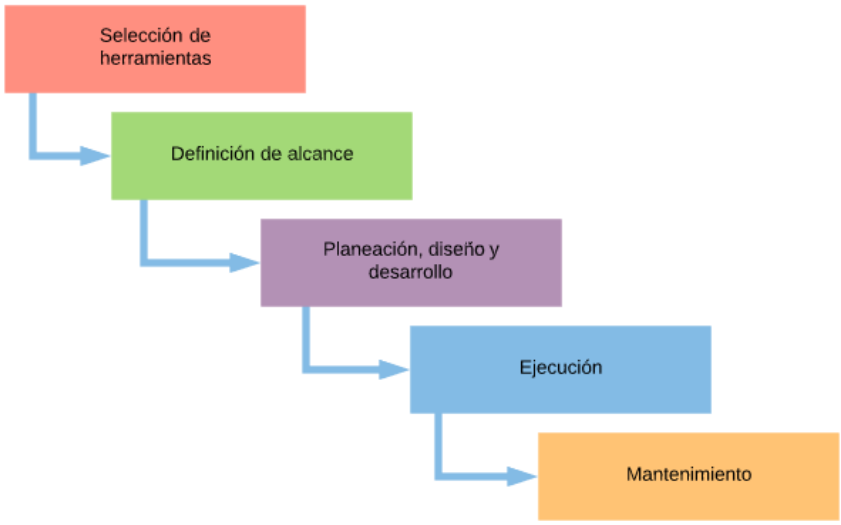

Figura 15. Flujo de trabajo para la automatización

i. Selección de herramientas:

Debido a las características del proyecto se eligieron las siguientes herramientas para la implementación de BDD:

1. **Serenity BDD**: Orientado específicamente a la generación de reportes asociados a la ejecución de casos de prueba (John Ferguson Smart, 2016).

2. **Cucumber:** Framework el cual nos permite ejecutar los casos de prueba en descripciones escritas en texto plano, es decir, permite ejecutar los casos de prueba descritos en lenguaje *Gherkin*.

3. **Selenium:** *Driver* que permite simular el comportamiento humano en pruebas a nivel *front-end*.

ii. Definición de alcance

Para el presente desarrollo se realizó la automatización de las pruebas a nivel funcional, dado que representa los requisitos más significativos de la herramienta, dichas pruebas serán a nivel de *front-end.* De igual forma se contemplaron solamente escenarios positivos, pues a nivel de desarrollo no se encontraba dentro de la prioridad realizar la captura de excepciones o muestra de mensajes de error.

iii. Planeación, diseño y desarrollo

En esta fase se detalló la arquitectura propuesta para la automatización, el número de casos de prueba a cubrir y el flujo para construir los test. En la sección resultados se abarcará con más detalle cada punto.

iv. Ejecución

La ejecución de los casos de prueba se da al momento de terminar la iteración y tener un mínimo producto viable para evaluar la regla de negocio agregada.

v. Mantenimiento

Se contempla el mantenimiento de las pruebas automatizadas o robots, ya que en el transcurso del desarrollo pueden ocurrir cambios en las reglas de negocio y por ende en la implementación. Ya que se tienen explícitos los criterios de comparación que pueden cambiar en el tiempo, se debe realizar el mantenimiento correspondiente, de lo contrario los casos de prueba resultarán fallidos.

Resultados

1. Arquitectura propuesta

Si bien al implementar BDD se ve una relación entre desarrollo y pruebas, el proyecto de automatización no está estrictamente ligado al lenguaje de desarrollo.

Dado que las herramientas escogidas (*Serenity, Cucumber, Selenium*) tienen mayor facilidad de uso e instalación al estar asociadas a un proyecto *Java* (por el IDE *Eclipse* y el manejo de dependencias mediante *Maven*), se escogió este lenguaje para la implementación de las pruebas automatizadas.

A continuación, se muestra el diagrama de paquetes que describe la arquitectura propuesta para la automatización de pruebas.

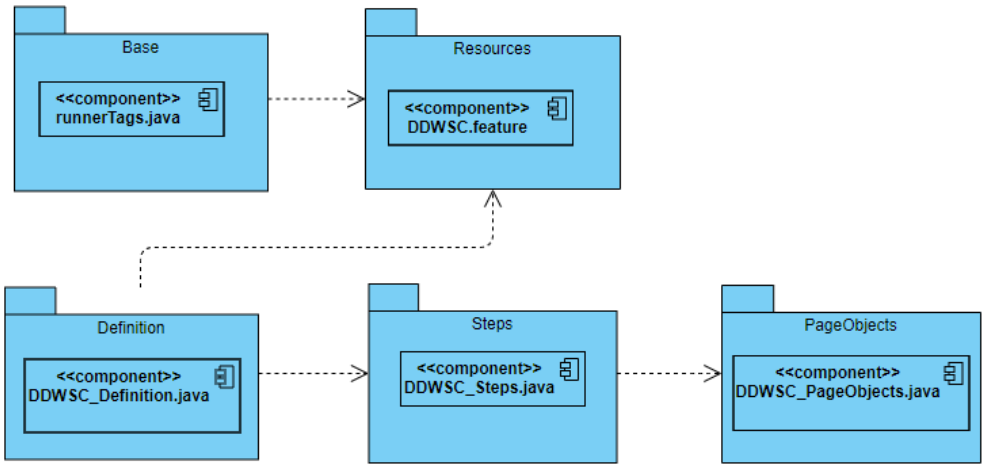

Figura 16. Arquitectura para la automatización de pruebas

- **Base**: Este paquete contiene el componente con el que se podrán ejecutar los casos. En dicho archivo java se hace referencia al *feature* correspondiente, y a los casos que se desean ejecutar.

- **Definition**: Dicho paquete contiene los métodos mapeados del *feature* de cada uno de los pasos *Given, When y Then*. Dichos métodos llaman a su vez a métodos del paquete *steps*.

- **Steps**: Este paquete contiene la clase que llama los métodos necesarios de la clase *PageObjects* y a su vez los encapsula en pasos más específicos que son llamados por la clase del paquete *definition*.

- **PageObjects**: Dicho paquete representa los componentes web a validar, y los métodos para acceder a ellos. Estos métodos son llamados por la clase *steps*.

- **Resources**: Paquete que contiene el archivo *feature* en donde se definen los casos de prueba de la herramienta.

Se escogió la siguiente división de paquetes debido que de esta manera cada clase tiene una responsabilidad única, y es más clara la relación que tienen con los demás paquetes.

2. Casos de prueba

Como se describió previamente, los casos de prueba están escritos en lenguaje *Gherkin* y son almacenados en un archivo *feature*, estos tienen la siguiente estructura:

- **Background**: Describe las precondiciones para cada uno de los escenarios, así se evita escribir este paso en cada uno.
- **Scenario**: Indica el nombre del escenario de prueba que contiene los casos.
- **Scenario Outline**: Indica que en el escenario se pueden recibir variables para correr múltiples casos con diferente *data*. Este escenario va junto con la tabla *Examples*, en donde se detallan los datos a utilizar en cada caso de prueba.
- **Given (dado):** Pone en contexto el escenario y provee los prerrequisitos para ejecutar el test.
- **When (cuando):** Describe las acciones que suceden al lanzar el caso de prueba.
- **Then (entonces):** Especifica el resultado que se espera del caso de prueba.

A continuación, se presenta un ejemplo de caso de prueba realizado:

```
@DDWSCWeb
Feature: DDWSC Web

  Background:
    Given El portal web esta disponible

  @RegresionDDWSCWeb @CasoExitoso
  Scenario Outline: Búsqueda normal en DDWSC
    When para el caso de prueba <idCaso>
    And ingreso los criterios de busqueda correctamente <criterio>
    Then verifico los resultados en la pagina principal

    Examples:
      | idCaso | criterio  |
      |      1 | Bogota    |
      |      2 | Colombia  |
      |      3 | Javeriana |
```

Figura 17. Ejemplo caso de prueba

A partir de esta información se evidencian tres (3) casos de prueba dentro del escenario exitoso, en donde el primer criterio a enviar en la búsqueda normal para el primer caso es “Bogotá”, el segundo es “Colombia” y el tercero “Javeriana”.

3. Análisis del reporte

Serenity provee la creación de un reporte HTML (cuyo diseño puede ser modificable) en el cual se capturan las evidencias paso a paso, adicionalmente, se crea un gráfico en donde se evidencian los casos de prueba totales, diferenciando entre casos pasados, en donde todos los pasos fueron satisfactorios, los fallidos, en donde ocurrió un error en uno de los pasos y los casos en “advertencia”, en donde quedaron pendientes uno o más pasos por ejecutar. Estas evidencias pueden ser consultadas en la carpeta del proyecto, bajo el nombre “*index*”.

Gracias a dicho reporte es posible evidenciar el paso a paso de la ejecución, y en casos fallidos, saber cuál es el error arrojado.

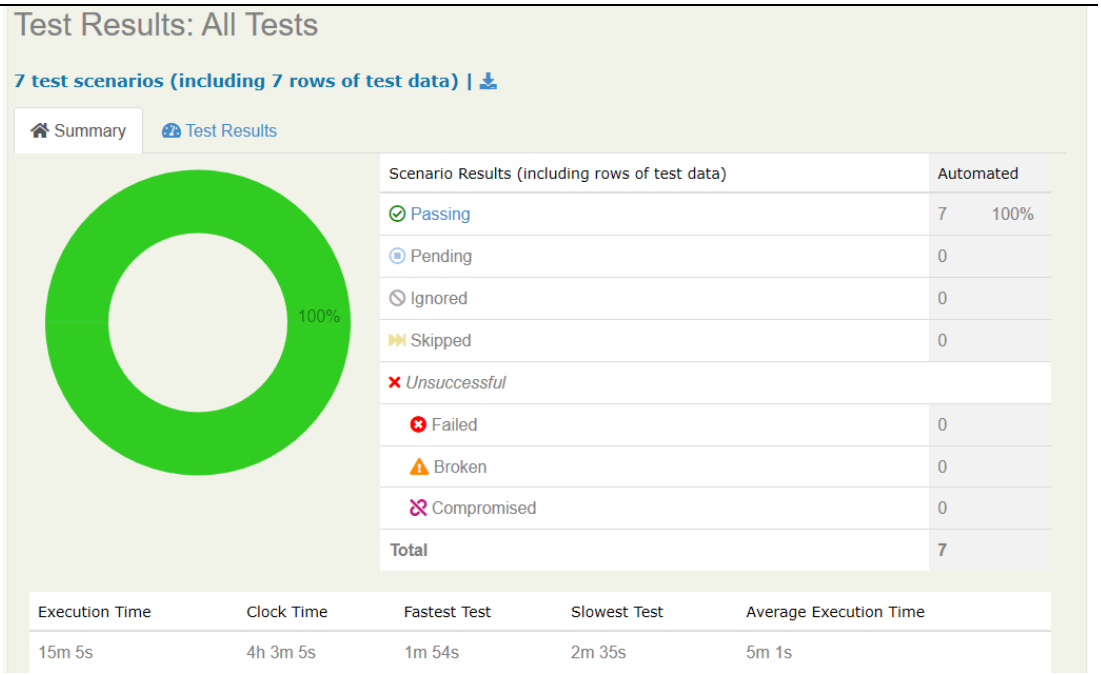

Figura 18. Reporte de pruebas

CONCLUSIONES

Análisis de impacto

A continuación, se muestra el impacto que generará el desarrollo de este proyecto para las organizaciones que hagan uso de la herramienta desarrollada.

Corto plazo.

A corto plazo, aproximadamente en cinco (5) meses se espera que el impacto sea pequeño; puesto que será aplicado en un caso de estudio con La Armada de la República de Colombia (ARC) y posiblemente utilizado por un número reducido de organizaciones en su mayoría PyMES, que no tienen gran cantidad de recursos e infraestructura destinados al aseguramiento de sus activos, y desean realizar algún tipo de ciberinteligencia para tomar medidas de seguridad y proteger los activos que poseen. Además, durante la aplicación del caso de estudio es importante monitorear la

herramienta para detectar aumentos de carga que afecten el rendimiento y así establecer los recursos necesarios para mejorar la disponibilidad del sistema.

Mediano plazo.

A mediano plazo, transcurrido cerca de un (1) año, se espera darle continuidad al proyecto haciendo del sistema una herramienta más robusta mejorando su nivel de seguridad. Para lograr esto es necesario tener un control de acceso a las aplicaciones, usar protocolos seguros y encriptar la información en donde sea posible. Todo con el fin de que pueda soportar el apoyo para ciberinteligencia a grandes organizaciones ayudando a crear modelos proactivos en caso de que se presenten incidentes de seguridad generando impactos negativos en las organizaciones.

Largo plazo.

A largo plazo, después de un lapso aproximadamente de tres (3) años, se busca que la herramienta DDW&SC sea utilizada por diferentes empresas reconocidas, logrando la localización de información que les sea de utilidad, soportando así la implementación de planes de mitigación y respuesta proactiva ante incidentes de seguridad que se generen desde *Deep Web* y redes sociales. Todo esto con el fin de que este aplicativo se convierta en una herramienta asequible para cualquier organización que desee hacer ciberinteligencia y así protegerse de posibles ataques, ahorrando en muchas ocasiones recursos que se deben invertir luego de que se han producido los incidentes.

Aspecto social.

Esta herramienta al igual que *Aleph* (Milenio, 2018), que se encuentra en desarrollo, si no se le da un uso adecuado y se controla su acceso, al caer en manos de menores de edad o usuarios malintencionados generaría un impacto negativo, puesto que en la *Deep Web* y *Dark Web* se encuentra

mucho contenido como pornografía infantil, venta de drogas, venta de armas y otras ilegalidades permitiendo el fácil acceso a cualquier usuario.

Aspecto tecnológico.

En el aspecto tecnológico es un aporte significativo, puesto que la herramienta DDW&SC está proveyendo un mecanismo para realizar búsquedas en una red muy volátil que cambia constantemente y hace difícil su indexación. Además, con trabajos adicionales sobre la herramienta, puede llegar a desarrollarse un sistema que haga uso de *machine learning* o inteligencia artificial ofreciendo funcionalidades para la predicción e identificación de eventos relacionados con ciberseguridad.

Aspecto económico.

El aspecto económico se ve afectado positivamente, ya que al realizar búsquedas mediante DDW&SC, se está economizando tiempo comparado con la forma convencional en que lo haría un usuario visitando distintos buscadores. Pues la herramienta hace este proceso en las diferentes fuentes de resultados una vez el usuario ha enviado la solicitud de búsqueda, y adicionalmente presenta una página de detalles por cada resultado en donde se puede ver un resumen de los datos relevantes sin tener que visitar la página original que en muchos casos demora en cargar debido al enrutamiento de la red *TOR*.

Conclusiones y trabajo futuro

Conclusiones.

El objetivo general del proyecto se cumplió con la implementación del 100% de las historias de usuario, brindando así una herramienta que provee la integración de buscadores como **Torch, Ahmia NotEvil, VisiTOR**, redes sociales como *Twitter*® y plataformas como *Pastebin*®, en una sola

herramienta. Adicionalmente, se proporciona información relevante de la *Deep Web* y *Dark Web* en la web superficial, mediante el acceso desde *browsers* convencionales como *Google Chrome*© y *Mozilla Firefox*©.

De igual manera, se puede concluir que con la herramienta desarrollada se apoya el proceso de obtención de información de fuentes abiertas, acoplando resultados provenientes tanto de plataformas sociales como de la *Deep Web* en una única interfaz. Dichos resultados, algunos sobre delitos informáticos, muchas veces no llegan a la red superficial, por lo cual pueden ser de utilidad como fuente de investigación.

Uno de los grandes retos que surgen a partir del desarrollo de este proyecto es la implementación de búsquedas dentro de la red social *Facebook*®, pues como se mencionó anteriormente el servicio de *Graph Search* fue dado de baja impidiendo su funcionamiento por lo que el equipo se vio en la obligación de reemplazarlo.

Debido a que no hay control de la información presente en esta red, la calidad de esta no expresa un juicio de valor real. En otras palabras presenta información que en muchos casos no es totalmente confiable y su contenido generalmente está por fuera de la ley como la pornografía y las ventas ilegales. Se propone como trabajo futuro implementar un mecanismo para filtrar esta información.

La cantidad de información que se encuentra en la *Deep Web* puede ser de gran ayuda para la prevención y preparación ante posibles ataques. Con la herramienta DDW&SC este proceso de búsqueda en distintas fuentes (que manualmente puede llegar a ser engorroso y con dificultad para usuarios con pocos conocimientos), se facilita al usuario evitando realizar la búsqueda individual en cada uno de los buscadores, ofreciendo una sola interfaz que realiza la búsqueda conjunta para

todas las fuentes, teniendo en una sola pantalla los resultados más relevantes y actuales arrojados por los buscadores seleccionados.

Trabajos futuros.

A lo largo del desarrollo, se observó el potencial que tenía la herramienta y la gran cantidad de mejoras que estaban fuera del alcance de este trabajo de grado que se pueden llegar a implementar. Dentro de los trabajos futuros se proponen las siguientes ocho (8) características que podrían darle un muy buen valor a la herramienta:

1. Desplegar la herramienta en un entorno virtualizado ofreciendo un grado de disponibilidad mayor y seguridad para el usuario final.
2. Implementar un modo de búsqueda segura, para no mostrar contenido abominable como la pornografía infantil.
3. Incrementar el número de agentes de búsqueda de la herramienta.
4. Realizar detección automática de la estructura de la página, con el objetivo de adaptarse a ella ante posibles cambios *(Machine Learning*).
5. Implementación de todas las vistas para adaptarse a cualquier tipo de pantallas como las de dispositivos móviles.
6. Parametrización que permita la flexibilidad de la configuración del entorno, como asignación de recursos, conexiones, *proxies*, entre otros.
7. Desarrollar módulos para el aseguramiento de evidencia digital.
8. Aplicación de *data Analytics* sobre los resultados encontrados para generar vectores sobre posibles ataques.

REFERENCIAS

ACIS. (2019, julio 23). AIS Group presenta a la banca colombiana una solución para mejorar la concesión de créditos mediante inteligencia artificial | ACIS. Recuperado 8 de diciembre de 2019, de https://acis.org.co/portal/content/NoticiaInternacionall/ais-group-presenta-la-banca-colombiana-una-soluci%C3%B3n-para-mejorar-la-concesi%C3%B3n-de-cr%C3%A9ditos

Angular. (2016). Angular. Recuperado 8 de diciembre de 2019, de https://angular.io/

Angular—Architecture overview. (2016). Recuperado 13 de noviembre de 2019, de https://angular.io/guide/architecture

AS. (2019, diciembre 5). 50.000 nuevos virus cada día: Cuánto malware sale en la Red—AS.com. Recuperado 8 de diciembre de 2019, de https://as.com/meristation/2019/12/05/betech/1575527835_046229.html

bdnews24. (2019). Ransomware increases cyberattack risks despite security steps—Bdnews24.com. Recuperado 8 de diciembre de 2019, de https://bdnews24.com/technology/2019/11/18/ransomware-increases-cyberattack-risks-despite-security-steps

CRO Forum—2016—CRO Forum Concept Paper on a proposed categorisati.pdf. (s. f.). Recuperado de https://www.thecroforum.org/wp-content/uploads/2016/06/ZRH-16-09033-P1_CRO_Forum_Cyber-Risk_web.pdf

Crummy. (2019, octubre 6). Beautiful Soup Documentation—Beautiful Soup 4.4.0 documentation. Recuperado 13 de noviembre de 2019, de https://www.crummy.com/software/BeautifulSoup/bs4/doc/

Daniel Tomaszuk. (2018). Python's Frameworks Comparison: Django, Pyramid, Flask, Sanic, Tornado, BottlePy and More. Recuperado 13 de noviembre de 2019, de https://www.net-guru.com/blog/python-frameworks-comparison

Django Project. (2019). The Web framework for perfectionists with deadlines | Django. Recuperado 13 de noviembre de 2019, de https://www.djangoproject.com/

ELK. (2010). Elasticsearch: RESTful, búsqueda y analíticas distribuidas | Elastic. Recuperado 13 de noviembre de 2019, de https://www.elastic.co/es/products/elasticsearch

EY - 2017—EY-recuperando-la-ciberseguridad.pdf.pdf. (s. f.). Recuperado de https://www.ey.com/Publication/vwLUAssets/EY-recuperando-la-ciberseguridad/$File/EY-recuperando-la-ciberseguridad.pdf

FireEye. (2012). *Protección de los datos, la propiedad intelectual y la marca frente a ciberataques*. 7.

G20—2017—Communiqué G20 Finance Ministers and Central Bank .pdf. (s. f.). Recuperado de http://www.g20.utoronto.ca/2017/170318-finance-en.pdf

IEEE. (1988). IEEE Standard for Software Project Management Plans. *IEEE Std 1058.1-1987*, 1-28. https://doi.org/10.1109/IEEESTD.1988.121942

IEEE. (1998). IEEE Recommended Practice for Software Requirements Specifications. *IEEE Std 830-1998*, 1-40. https://doi.org/10.1109/IEEESTD.1998.88286

IEEE. (2009). IEEE Standard for Information Technology–Systems Design–Software Design Descriptions. *IEEE STD 1016-2009*, 1-35. https://doi.org/10.1109/IEEESTD.2009.5167255

John Ferguson Smart. (2014). An Introduction to BDD Test Automation with Serenity and Cucumber-JVM. Recuperado 13 de noviembre de 2019, de http://thucydides.info/docs/articles/an-introduction-to-serenity-bdd-with-cucumber.html

John Ferguson Smart. (2016). The Serenity Reference Manual. Recuperado 13 de noviembre de 2019, de http://thucydides.info/docs/serenity-staging/

Manuel, P. T., José, & Santiago, T. C. (2016). *Ideas para aprender a aprender: Manual de innovación educativa y tecnología*. Editorial UOC.

Milenio. (2018, diciembre 18). Aleph: El buscador similar a Google de la deep web. Recuperado 4 de noviembre de 2019, de https://www.milenio.com/tecnologia/aleph-el-google-de-la-deep-web

MongoDB. (2009). La base de datos líder del mercado para aplicaciones modernas. Recuperado 8 de diciembre de 2019, de MongoDB website: https://www.mongodb.com/es

Neoteo. (2019, marzo 14). Los mejores buscadores de la Deep Web. Recuperado 4 de noviembre de 2019, de NeoTeo website: https://www.neoteo.com/buscadores-deep-web/

Nguyen, K. (2019, junio 13). Facebook accused of siding with violent regimes after blocking access to accountability tool [Text]. Recuperado 4 de noviembre de 2019, de ABC News website: https://www.abc.net.au/news/2019-06-13/facebook-blocks-access-to-graph-search-transparency-tool/11207102

Pallets Projects. (2019). Flask. Recuperado 13 de noviembre de 2019, de Pallets website: https://palletsprojects.com/p/flask/

Panda Security. (2016, junio 6). Los secretos de la Deep Web. Recuperado 4 de noviembre de 2019, de Panda Security Mediacenter website: https://www.pandasecurity.com/spain/mediacenter/seguridad/tor-y-deepweb-todos-los-secretos/

Pastebin. (2002). Pastebin.com—FAQ [Frequently Asked Questions] [Paste Site]. Recuperado 4 de noviembre de 2019, de Pastebin.com website: https://pastebin.com/faq#1

Portafolio. (2019). La inteligencia artificial irrumpe en las finanzas | Tendencias | Portafolio. Recuperado 8 de diciembre de 2019, de https://www.portafolio.co/tendencias/la-inteligencia-artificial-irrumpe-en-las-finanzas-536183

Reddit. (2005). Página de inicio—Reddit. Recuperado 4 de noviembre de 2019, de https://www.redditinc.com/

Rubio, L. A. S., & Huertas, A. B. (2018). *SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO*. 97.

Sanic. (2017). Sanic—Sanic 19.9.0 documentation. Recuperado 13 de noviembre de 2019, de https://sanic.readthedocs.io/en/latest/

School of Computing Universiti Utara Malaysia Kedah, Malaysia, & Oluwatosin, H. S. (2014). Client-Server Model. *IOSR Journal of Computer Engineering*, *16*(1), 57-71. https://doi.org/10.9790/0661-16195771

Scrapy. (2008a). Scrapy 1.8 documentation—Scrapy 1.8.0 documentation. Recuperado 13 de noviembre de 2019, de https://docs.scrapy.org/en/latest/

Scrapy. (2008b). Scrapy at a glance—Scrapy 1.8.0 documentation. Recuperado 12 de noviembre de 2019, de https://docs.scrapy.org/en/latest/intro/overview.html

Selenium. (2007). Selenium—Web Browser Automation. Recuperado 13 de noviembre de 2019, de https://www.seleniumhq.org/

Superstatic. (2018). Recuperado 13 de noviembre de 2019, de Npm website: https://www.npmjs.com/package/superstatic

The New York Times. (2017, septiembre 7). Equifax Says Cyberattack May Have Affected 143 Million in the U.S. - The New York Times. Recuperado 8 de diciembre de 2019, de https://www.nytimes.com/2017/09/07/business/equifax-cyberattack.html

Tor Project. (2015, octubre 29). El Proyecto Tor | Privacidad & Libertad en línea. Recuperado 4 de noviembre de 2019, de https://torproject.org

Towardsdatascience. (2019). Scrapy Vs Selenium Vs Beautiful Soup for Web Scraping. Recuperado 13 de noviembre de 2019, de https://towardsdatascience.com/scrapy-vs-selenium-vs-beautiful-soup-for-web-scraping-24008b6c87b8

APÉNDICES

Anexos

Anexo 1 – Especificación de historias de usuario.

Anexo 2 – Project Management Plan

Anexo 3 – Estado del arte

Anexo 4 – Manual de usuario

Anexo 5 – Cronograma de proyecto

Listado de tablas

Listado de figuras