HERRAMIENTA DE SOFTWARE PARA AUTOMATIZACIÓN DE LOS PROCESOS DE INSERCIÓN Y CAPTURA DE DATOS DE LA EMPRESA LUXITYLAB

ANDRÉS MAURICIO GÓMEZ RODRÍGUEZ
.

UNIVERSIDAD PEDAGÓGICA Y TECNOLÓGICA DE COLOMBIA
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA DE SISTEMAS
TUNJA
2021

HERRAMIENTA DE SOFTWARE PARA AUTOMATIZACIÓN DE LOS PROCESOS DE INSERCIÓN Y CAPTURA DE DATOS DE LA EMPRESA LUXITYLAB

ANDRÉS MAURICIO GÓMEZ RODRÍGUEZ
.

Trabajo de grado en la modalidad de práctica con proyección empresarial o social para optar al título de Ingeniero en sistemas

Director (a)
JORGE ENRIQUE OTALORA LUNA
Doctor en Ingeniería de software

UNIVERSIDAD PEDAGÓGICA Y TECNOLÓGICA DE COLOMBIA
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA DE SISTEMAS
TUNJA
2021

Nota de aceptación:

Aprobado por el Comité de Currículo en cumplimiento de los requisitos exigidos por la Universidad Pedagógica y Tecnológica de Colombia para optar al título de Ingeniero en sistemas, actuando como jurados:

NOMBRES Y APELLIDOS DEL JURADO 1 (EN MAYÚSCULA)
Título Académico de mayor nivel

NOMBRES Y APELLIDOS DEL JURADO 2 (EN MAYÚSCULA)
Título Académico de mayor nivel

Tunja, fecha (día, mes, año)

TABLA DE CONTENIDO

pág.

LISTA DE ILUSTRACIONES

pág.

1. INTRODUCCIÓN

El presente trabajo en modalidad de práctica empresarial para la empresa Luxitylab tiene como objetivo principal la creación de un sistema de software que permita la grabación de procesos ejecutados dentro de un sitio web para poder re ejecutarlos en cualquier momento y las veces que sea necesario, eliminando por completo los errores humanos y permitiendo a las empresas utilizar la creatividad de sus empleados en tareas donde las maquinas aun no pueden llegar.

Su importancia radica en que hoy en día está en auge la tendencia de delegar a los robots y maquinas tareas tediosas y repetitivas en las que los humanos no nos sentimos a gusto, además de que se reduce el tiempo necesario para que dichos procesos sean ejecutados aumentando la productividad de las empresas.

El documento está estructurado por capítulos, en el primero especifica los objetivos que se esperan lograr con el desarrollo del presente trabajo; el segundo resume por qué se tomó la decisión de desarrollar el proyecto; en el tercer capítulo se hace el referente teórico necesario para entender el desarrollo; en el cuarto se describe la metodología usada para el desarrollo; en el quinto capitulo se muestra el desarrollo del proyecto en cada una de sus etapas.

1. OBJETIVOS

A continuación, se enumeran los objetivos que se esperan lograr con el desarrollo del presente proyecto.

1.1. OBJETIVO GENERAL.

Desarrollar una herramienta de software para robotización de procesos, que se integre al navegador Google Chrome permitiendo automatizar la inserción y captura de datos de la empresa “Luxitylab”.

1.2. OBJETIVOS ESPECÍFICOS.

- Desarrollar un módulo que permita la captura de las acciones realizadas por un usuario dentro de un sitio web.
- Diseñar un mecanismo que permita el procesamiento y persistencia de las acciones realizadas por el usuario.
- Construir una extensión de Google Chrome que le permita recrear los procesos almacenados por el usuario.
- Realizar pruebas para comprobar la efectividad de la herramienta implementada.

2. PLANTEAMIENTO DEL PROBLEMA

Actualmente toda organización que apoye sus procesos con software, bien sea genérico o a la medida, debe realizar tareas tediosas y repetitivas, como por ejemplo cargar una lista de 1000 o más usuarios en un formulario que exige la carga uno a uno, o por ejemplo revisar frecuentemente un sitio web para extraer y comparar el valor de una moneda o criptomoneda.

También vemos que la necesidad de delegar tareas tediosas y repetitivas a los robots está presente y en auge hoy en día, y en el ambiente empresarial es común que existan tareas que deben realizarse desde aplicaciones o sitios web, y que demandan gran cantidad de tiempo de un empleado, y las herramientas que existen en el mercado tienen un enfoque diferente ya que trabajan a nivel de sistema operativo, y tienen un alto costo monetario.

De esta manera proponemos buscar una estructura de software cuya interfaz con el usuario sea una extensión web, y que a través de sus componentes capturare las acciones que el usuario realiza en un sitio web, las ordene y guarde en una base de datos para así ser capaces de recrear dicha tarea en cualquier momento determinado, la veces que sea necesario.

3. REFERENTE TEÓRICO

Debido a que este trabajo corresponde a una práctica empresarial desarrollada dentro de la empresa Luxitylab, los referentes teóricos relacionados no contienen apartados tales como estado del arte y marco teórico. Se presenta un referente de los conceptos y herramientas utilizadas durante el desarrollo de la práctica.

A continuación, se hace una breve introducción a los diferentes conceptos necesarios para entender el desarrollo del presente trabajo. Para obtener un mayor contenido teórico al respecto, se recomienda consultar las referencias.

3.1. RPA (robotic process automation).

La automatización robótica de procesos se trata de delegar a un robot de software una tarea compuesta por una serie de acciones que normalmente son ejecutadas por un ser humano, quitando del medio los errores humanos y añadiéndole las ventajas de ser ejecutadas por una máquina, aumentando la productividad y permitiendo que se aproveche la capacidad creativa de los humanos para otro propósito. El robot se enfrenta a una interfaz de usuario en donde ejecuta las acciones que constituyen la tarea a realizar por lo que es necesario que estén bien definidas y programadas por un humano. [1]

Para automatizar un proceso se deben tener en cuenta algunas características del mismo como:

- Altamente manuales y repetitivos: Que sean repetitivos, y, por tanto, sometidos a poca variación o discernimiento.
- Basados en reglas: Las decisiones o alternativas en el flujo están claras, y se pueden reducir a un conjunto de reglas bien definidas.
- Procesos con entradas electrónicas estándar: Los datos de entrada llegan en formato electrónico estándar y estructurado, lo cual redunda en una facilidad de procesamiento y automatización.

- Sistemas no sometidos a grandes cambios: Los sistemas en que se apoya el flujo son también estables.
- Procesos maduros y estables: No se están modificando continuamente esos procesos, sino que son operativas ya bien asentadas y sin cambios.

[2]

3.2. DESARROLLO FRONT-END

Cada vez son más los procesos que muchas empresas apoyan con aplicaciones y sitios web que están construidos en diferentes frameworks y tecnologías disponibles en el mercado. Estas aplicaciones son presentadas a los usuarios a través de un navegador web que es el encargado de interactuar con el usuario; para que un sitio web pueda ser mostrado a sus usuarios, actúan principalmente 3 lenguajes que en conjunto componen lo que es conocido como desarrollo Front-End:

a. HTML (Lenguaje de maquetación)
b. CSS (Lenguaje de diseño)
c. JavaScript (Lenguaje de programación interpretado por los navegadores)

A través de los últimos años que han sido de rápido avance para estas herramientas, surgen diferentes frameworks y tecnologías como Angular js, React js, Vue js, por nombrar los más conocidos; que intentan con éxito integrar y optimizar el uso de las tres herramientas principales (HTML, CSS, JavaScript). El resultado de un sitio web construido con unos de estos Frameworks, aunque es diferente entre cada uno de ellos, comparten la característica de ser un compilado de las tres herramientas principales, y heredar propiedades y ventajas de cada una de ellas. Lo que nos permite generalizar ciertas características como la maquetación por etiquetas, y el manejo de eventos del usuario; sin importar en la herramienta que se haya desarrollado el sitio web en cuestión. [3]

3.3. EXTENSION - GOOGLE CHROME.

Una extensión para navegador es una aplicación escritas en JavaScript que se ejecuta en el contexto de un navegador web teniendo acceso a toda la información que los usuarios ingresan y/o consumen en un sitio web. Según Google: “Las extensiones de navegador son pequeños programas o aplicaciones que se instalan dentro del navegador web (Google Chrome) y añaden o mejoran funciones del navegador desde la propia aplicación. De forma habitual una extensión de navegador pesa muy poco, algunas menos de 1 mega y crean un acceso directo en el menú del navegador desde el que podemos ejecutar este complemento.” [4]

3.4. ETIQUETA HTML

Según [5], un elemento HTML “Es un elemento del lenguaje *HTML* cuyo formato es un fragmento de texto encerrado entre corchetes angulares < >, y cada elemento HTML tiene una etiqueta de inicio del tipo <etiqueta> y suele terminar con una etiqueta de cierre que lleva una barra inclinada al principio </etiqueta>”.
Un elemento HTML puede ser un botón, un enlace, una imagen, o un simple párrafo.

3.5. CÓDIGO COMPILADO:

Según la documentación de Mozilla Firefox, [6] “Compilar es el proceso de transformar un programa informático escrito en un lenguaje en un programa equivalente en otro formato. Al programa que se encarga de compilar se le llama compilador. A veces, a esta tarea se le llama "ensamblar" o "construir", lo que suele implicar otros procesos adicionales, ej. empaquetarlo en formato binario.”
Por ejemplo, un conjunto de archivos en formato HTML, CSS, y JavaScript, pueden ser compilados en un solo archivo en formato HTML, y un solo archivo en formato JavaScript, que es como funciona *React js.*

3.6. EVENTOS JAVASCRIPT:

Tradicionalmente la programación se ejecuta de forma secuencia, es decir, se ejecuta una instrucción después de ejecutar la instrucción inmediatamente anterior; para el desarrollo de aplicaciones web, predomina un modelo de programación basado en eventos en donde el lenguaje de programación está al pendiente de los sucesos en el contexto de la aplicación.

“Los eventos son acciones u ocurrencias que suceden en el sistema que está programando y que el sistema le informa para que pueda responder de alguna manera si lo desea. Por ejemplo, si el usuario hace clic en un botón en una página web, es posible que desee responder a esa acción mostrando un cuadro de información”. [7]

Existen diferentes tipos de eventos que ocurren en un sitio web:

- Eventos del teclado
- Eventos del mouse
- Eventos de carga de información

Algunos de los eventos que JavaScript provee y que podemos enlazar a elementos del sitio web, son:

- *Evento clic,* que se ejecuta cuando el usuario presiona el botón izquierdo del mouse sobre un elemento HTML
- *Evento focusout*, que se ejecuta cuando un usuario quita el cursor de un elemento que estaba seleccionado.
- *Evento change,* que se ejecuta cuando cambia el contenido de un elemento, principalmente se usa sobre elementos de entrada de texto

[7]

3.7. FRAMEWORK

Como su traducción al español lo indica, un Marco de trabajo, se puede ver como una plantilla en la que se encierran y generalizan conceptos principalmente para generar estándares y resumir el contenido de un trabajo.

“En general, con el término framework, nos estamos refiriendo a una estructura software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación. En otras palabras, un framework se puede considerar como una aplicación genérica incompleta y configurable a la que podemos añadirle las últimas piezas para construir una aplicación concreta.” [8]

Para el desarrollo de aplicaciones existe actualmente en el mercado una gran cantidad de frameworks que permiten construir sitios web con diferentes lenguajes de programación, y desde diferentes enfoques.

3.8. ANGULAR JS

Angular es un marco de trabajo para la construcción de aplicaciones web, y una plataforma de desarrollo para crear sitios de una sola página eficientes y sofisticados. Es actualmente uno de los frameworks más usados para construir aplicaciones.

Su principal característica es que proporciona a las aplicaciones un patrón diseño de Modelo Vista Controlador, el cual es ampliamente utilizado en la construcción de software porque conceptualiza las capas que conforman un producto.

Este software está enfocado en la construcción de software web del lado del cliente por lo que es una herramienta útil para la creación de interfases graficas para usuario final

[9]

3.9. API (INTERFAZ DE PROGRAMACIÓN DE APLICACIONES)

Una interfaz de programación de aplicaciones es un conjunto de herramientas que proporciona protocolos para la comunicación entre sí de diferentes sistemas de programación.

Una Api se caracteriza por comunicar componentes o sistemas de software, principalmente se usan como interfaz para solicitar datos de un repositorio o una base de datos propiamente y servirlos en un formato determinado como lo puede ser el formato XML o JSON

[10]

3.10. DJANGO – PYTHON

Es un framework para el desarrollo de aplicaciones web que incluye el despliegue de servicios REST (Transferencia de Estado Representacional) que permiten crear APIs para fácil comunicación entre componentes de Software.

Su principal característica es que permite la comunicación con bases de datos sean relacionales o no relacionales, y proporciona la capacidad de servir la información consultada en diferentes formatos entre los que incluye HTML para presentación a usuario final, y XML y Json para comunicación entre componentes de software. A demás permite crear mecanismos para recepción de información sea desde un usuario final, o desde otro software o componente.

[11]

3.11. BASES DE DATOS DE GRAFOS

Las bases de datos que se orientan en grafos son un tipo de base de datos no relacional que se caracteriza por representar una entidad como un nodo, y una relación como una arista del grafo, de esta forma, un nodo cualquiera puede tener una relación igual o diferente con cualquier otro u otros nodos del grafo. Así, un cambio en la estructura de la base de datos, es decir, un cambio en la relación de

un nodo con otro, no implica hacer un cambio en toda la base de datos, sino resulta siendo un cambio local, con mucho menor impacto en el resto de datos.

Las bases de datos orientadas a grafos presentan algunas ventajas importantes, como permitir aplicar la teoría de grafos para las búsquedas de un nodo especifico, o inclusive alguna de sus propiedades, además, resulta siendo una herramienta útil para visualización de datos ya que permite identificar fácilmente comportamientos en el la relación de los datos.
[12]

3.12. NEO4J

Neo4j es un motor de Base de datos cuya filosofía se basa en los grafos, permitiendo que una entidad sea representada como un nodo del grafo, y la relación que unta entidad relaciona a otra se representa como una línea que une a dos nodos. [13]

Neo4J cuenta con su propio lenguaje de consulta basado en graphSql cuyo nombre es Chypher y según propia documentación de Neo4J: “La sintaxis de Cypher proporciona una forma visual y lógica de hacer coincidir patrones de nodos y relaciones en el gráfico. Es un lenguaje declarativo inspirado en SQL para describir patrones visuales en gráficos utilizando la sintaxis ASCII-Art. Nos permite indicar qué queremos seleccionar, insertar, actualizar o eliminar de los datos de nuestro gráfico sin una descripción de cómo hacerlo exactamente. A través de Cypher, los usuarios pueden construir consultas expresivas y eficientes para manejar la funcionalidad necesaria de creación, lectura, actualización y eliminación.” [14]

3.13. SELENIUM WEB DRIVER

Selenium se compone de una serie herramientas que permiten enviar comandos al navegador para que este los interprete como si un usuario estuviera ejecutando acciones dentro del sitio web.

Selenium cuenta con librerías para diferentes lenguajes de programación incluyendo Python, Java, JavaScript, y varios más. Hoy en día es ampliamente utilizado para automatizar casos de pruebas, y sets completos de pruebas para diferentes herramientas web. [15]

3.14. XPATH

XPath (XML Path Language) es un lenguaje que permite recorrer y navegar archivos en formato XML, incluyendo archivos HTML. Fue desarrollado para permitir la comunicación entre sistemas de información ya que XML fue en su momento la forma más conocida para enviar y recibir datos. Hoy en día es ampliamente utilizado para recorrer y extraer información de sitios web. [16]

3.15. DOM

El DOM es una interfaz de programación que define la estructura jerárquica en la que está formado un archivo en formato HML o XML, esta estructura es esencial para que sea posible recorrer y manipular los elementos pertenecientes a un archivo HTML. [17]

3.16. AJAX JS

Según Mozilla Org, “JavaScript Asíncrono + XML (AJAX) no es una tecnología por sí misma, es un término que describe un nuevo modo de utilizar conjuntamente varias tecnologías existentes. Esto incluye: HTML o XHTML, CSS, JavaScript, DOM, XML, XSLT, y lo más importante, el objeto XML HttpRequest. Cuando estas

tecnologías se combinan en un modelo AJAX, es posible lograr aplicaciones web capaces de actualizarse continuamente sin tener que volver a cargar la página completa. Esto crea aplicaciones más rápidas y con mejor respuesta a las acciones del usuario.” [6]

3.17. JSON

JSON (JavaScript Object Notation) es un formato de texto para representar y compartir información en forma de diccionario, lo que significa que un archivo en formato Json se compone de un elemento o una lista de elementos en la que cada uno de ellos cuenta con diferentes atributos con una llave y un valor. [18]

3.18. PARADIGMA EVOLUTIVO

El paradigma evolutivo para el desarrollo de software es una metodología que surge reuniendo características de las diferentes metodologías llamadas agiles para el desarrollo de proyectos de software, su principal característica es que desde una fase muy temprana del desarrollo (que se denomina primera iteración o primera evolución) se expone al usuario principal o la persona que está al frente del desarrollo por parte del cliente. Así sus comentarios y observaciones serán tenidos en cuenta para las siguientes iteraciones. La ventaja principal de utilizar una metodología evolutiva es que por lo general los clientes no saben realmente sus necesidades, y la mejor forma de identificar una necesidad es enfrentándose a una posible solución. [19]

4. METODOLOGÍA

Para el desarrollo del proyecto se usó una metodología iterativa incremental, en donde en cada iteración se intenta dar solución a una de las diferentes funcionalidades necesarias para cumplir con los objetivos del proyecto.
A continuación, se describen las herramientas utilizadas en el marco del desarrollo del proyecto, para esto, se categorizan de la siguiente forma:

a. *Herramientas para la captura de acciones*: La captura de eventos recae en la extensión para Google Chrome que cuenta con los siguientes componentes:

    i. **Angular/Cli**: en su versión 8.3.20 que permite la interacción con el usuario, aprovechando los componentes estilizados que provee, no es necesario ocupar mucha atención en el diseño.
    ii. **JavaScript**: que soporta la captura y persistencia temporal de las acciones ejecutadas por el usuario. También se encarga de la comunicación con el backed para la persistencia de la información capturada
    iii. **Chrome Tabs Api**: que permite la comunicación entre los diferentes componentes de la extensión

b. *Herramientas para la recreación de acciones:* Para recrear las acciones correspondientes a un proceso, se usa principalmente:

    i. **Selenium Web driver 3.141.0**: En su librería para Python Selenium provee una interfaz para enviar comandos al navegador web, y así es posible interactuar con los elementos del DOM del sitio visitado por un usuario.

c. *Herramientas comunes*: Estas herramientas tienen como función principal la persistencia de las acciones en una base de datos. Están involucradas tanto en la captura de eventos como en la recreación de los mismos-

   i. **Django 2.2.0**: que permite recibir las acciones capturadas, procesarlas y guardarlas en la base de datos para que persistan
   ii. **Neo4J:** es la base de datos que permite guardar las acciones conectadas entre sí en forma de grafo, es decir, una acción conectada con la inmediatamente anterior y también con inmediatamente siguiente. Además permite crear ramas adicionales para caminos alternativos en la recreación de los procesos guardados.

4.1. PRIMERA ITERACIÓN - CAPTURA DE EVENTOS EJECUTADOS POR UN USUARIO.

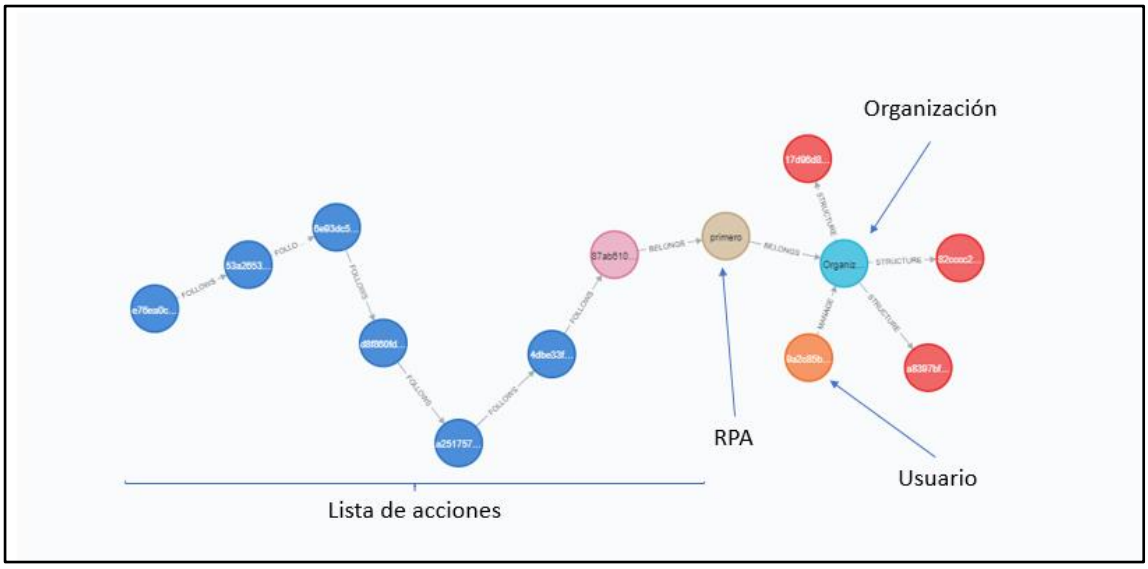
La primera iteración se centra principalmente en capturar los pasos que va ejecutando un usuario mientras interactúa con la aplicación web; esto es posible gracias a *JavaScript* que provee escuchadores de diferentes eventos que ocurren cuando el usuario ejecuta alguna acción.
De los diferentes eventos que *JavaScript* permite escuchar, se identificaron 2 para centrar especialmente la atención que son: ***click,*** y ***focusout***.
De la misma forma se identificó un elemento *HTML* en el cual enfocar la atención: la etiqueta ***input***; los demás elementos HTML pueden ser tratados de la misma manera.

4.2. SEGUNDA ITERACIÓN - PERSISTENCIA DE LAS ACCIONES DEL USUARIO.

La segunda iteración se centra en guardar en una base de datos los diferentes procesos con sus correspondientes acciones, para que puedan estar disponibles

para consulta en cualquier momento. La base de datos está representada por un *grafo* cuyo *nodo inicial* corresponde al *usuario*, el cual puede tener varias *organizaciones*, y a su vez una *organización* puede tener varios *procesos*.

Ilustración 1. Grafo



Fuente: Autor

Una vez el usuario comienza la grabación de un *proceso*, se guardan temporalmente las acciones que éste ejecuta dentro del sitio, y al finalizar el proceso de grabación se realiza una recopilación de las *acciones* temporalmente guardadas para ser enviadas vía *Ajax* al backend, que guarda cada acción como un *nodo* en el *grafo*.

4.3. TERCERA ITERACIÓN - RECREAR UN PROCESO

En esta iteración se busca ejecutar dentro de un sitio web las acciones ya almacenadas que pertenecen a un proceso especifico, para esto se recorre la *rama* del *grafo* que corresponde a dicho proceso (ver ilustración 1). De esta manera se

recopila la totalidad de acciones en una colección, y así es posible ejecutarlas una a una dentro del sitio web correspondiente.
Para la ejecución de las acciones se usa la librería de *Selenium* en su versión para Python. Esta librería se comunica con el navegador, y permite localizar el elemento *HTML* involucrado en la acción a ejecutar, y una vez localizado, Selenium también permite interactuar con él para completar la acción.

4.4. CUARTA ITERACIÓN - CREACIÓN DE LA EXTENSIÓN PARA GOOGLE CHROME

En la cuarta iteración se intenta solucionar la interacción con el usuario, para permitir que se administren los diferentes procesos que un usuario puede guardar.
La extensión para el navegador es el componente que interactúa con el usuario final, permitiéndole crear Rpas (Procesos Robóticos automatizados), y accionar el comienzo y fin de grabación de para cada proceso. Este componente está construido con el framework de desarrollo Angular que facilita la creación de interfaces de usuario.

A demás de interactuar con el usuario, la extensión es la encargada de comunicar los diferentes componentes de software involucrados en el desarrollo del proyecto, ya que interactúa con el sitio web para grabar un proceso, capturando y enviando las acciones ejecutadas por el usuario para que puedan ser guardadas en la base de datos. También es desde la extensión donde se permite al usuario re ejecutar un proceso previamente guardado.

5. DESARROLLO DEL PROYECTO

A continuación, se describirá la forma en que se intenta dar solución a cada una de las funciones necesarias para cumplir con los objetivos del proyecto.

5.1. ETAPAS DEL DESARROLLO

Cada etapa corresponde a cada una de las funciones principales que constituyen el software final.

5.1.1. Primera - Captura de Eventos ejecutados por un usuario.

Debido a la gran cantidad de posibilidades de eventos que un usuario puede ejecutar dentro de un sitio, se intenta resumir y generalizar estas acciones enfocando únicamente dos eventos de los posibles: evento ***click***, y evento ***focusout*** para todos los elementos del DOM.
Para la captura de las acciones ejecutadas por el usuario, se usa principalmente la forma nativa que provee JavaScript para la escucha y captura de eventos

Ilustración 2.Sintaxis para la escucha de eventos Javascript

```
document.addEventListener('focusout', function (e) {
    // Acciones a tomar
})
```

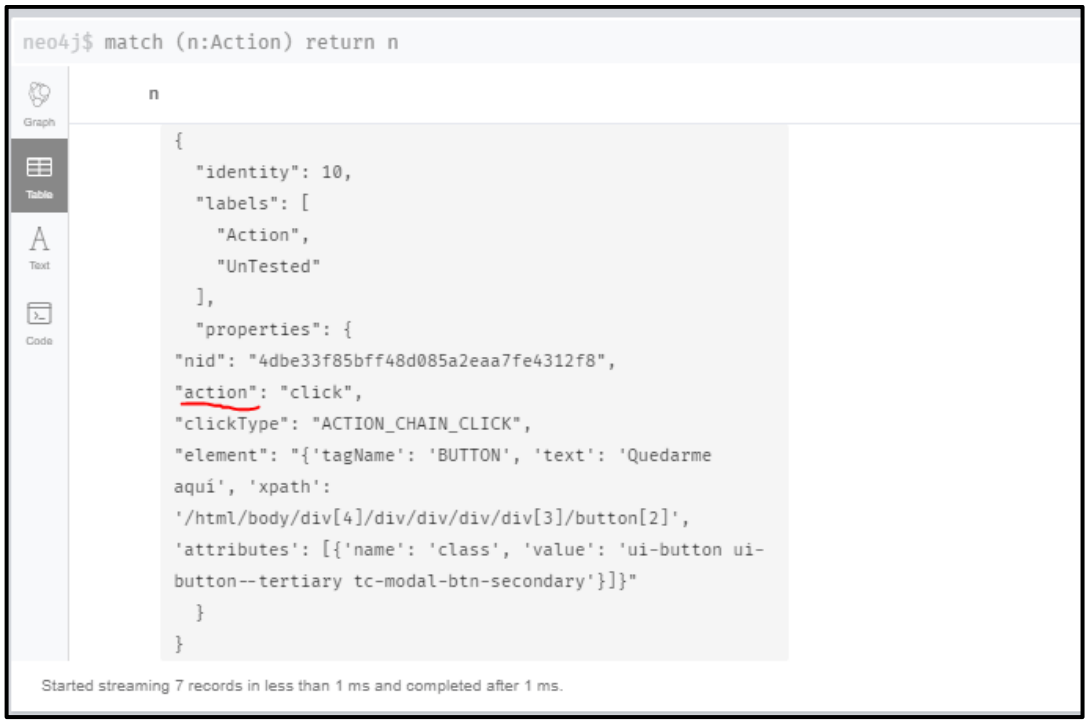
Fuente: Autor

Una vez capturado el evento, se identifica el nombre de la etiqueta, sus atributos *HTML*, el *XPath*, y el *texto* del elemento sobre el cual se ejecuta la acción.

De la misma forma, se realiza una tipificación que se guarda como una propiedad del objeto que corresponde la acción ejecutada (**ver ilustración 3**) llamada "***action***", para guardar el tipo de evento capturado que corresponderá a alguna de las siguientes opciones:

- click
- selected option
- keys enter

Ilustración 3.Datos guardados dentro del nodo



Fuente: Autor

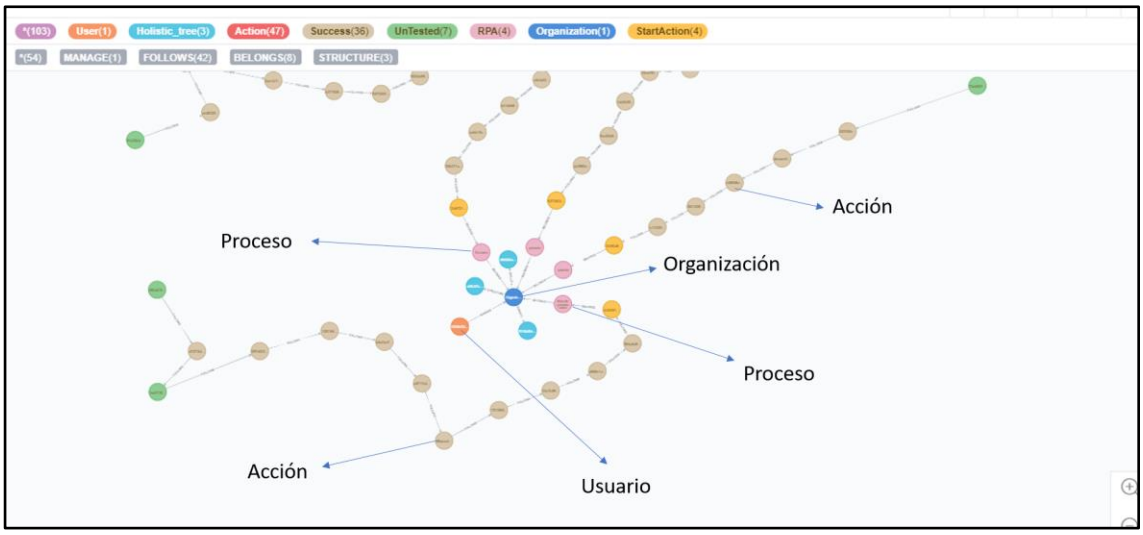
5.1.2. Segunda - Persistencia de las acciones del usuario.

Para realizar el almacenamiento de los datos, se hace uso del framework *Django* con el que se construye una *Api REST* para la información enviada después de capturar las acciones ejecutadas por el usuario.

Como se menciona anteriormente, cada acción se guarda en forma de nodo, así, cada proceso se modela como una rama del *grafo* que compone la base de datos, de esta forma, una acción está conectada con la acción inmediatamente anterior, y con la inmediatamente siguiente; comenzando por la acción de acceder al sitio web específico, guardando la url correspondiente.

Esto hace más sencillo recorrer y recrear un proceso, además, en caso de que haya necesidad de analizar con detalle un proceso, es más comprensible para el ojo humano.

Ilustración 4. Captura de pantalla de la base de datos real



Fuente: Autor

5.1.1. Tercera - Recrear un proceso

Esta tarea está a cargo de la *Api REST* construida con *Django*, que usando la librería *Selenium* ejecuta una a una las acciones que corresponden al proceso que se intenta recrear.
Para recrear un proceso es necesario cumplir dos pasos principales: localizar un elemento, y ejecutar la acción correspondiente

5.1.1.1. Localización de elementos:

Selenium provee diferentes métodos para ubicar un elemento dentro del DOM de un sitio web [18]
(Locating Elements-Selenium). Entre ellos, se centra la atención en dos de ellos: búsqueda por ***XPath,*** y búsqueda por el ***Texto*** que el elemento contiene.

Además, se presta especial atención a los hipervínculos identificados con la etiqueta ***a,*** ya que existe la posibilidad de que diferentes hipervínculos compartan el mismo *XPath*, y esto aumenta la probabilidad de tener errores para al momento de recrear el proceso, porque hacer clic sobre un hipervínculo erróneo, conlleva a ser dirigidos a una página del sitio que no corresponde a las visitadas por el usuario dentro del proceso grabado. Por esta razón la búsqueda por texto para los hipervínculos resulta siendo una mejor opción.

Teniendo en cuenta lo anterior y debido a que existen diferentes factores que pueden evitar que Selenium identifique un elemento, por ejemplo, que el programador haya definido dos elementos con el mismo identificador, se propone realizar dos intentos para ubicar un elemento:
Para el ***primero:*** se realizará una búsqueda por texto para dar prioridad a los hipervínculos, y en caso de fallar, se realizará una búsqueda por *XPath*.

Para el ***segundo intento***: se realiza primero una búsqueda más compleja compuesta por el XPath del elemento concatenado con el *texto* que éste contiene; en caso de fallar se realiza una búsqueda compuesta por el *XPath* del elemento más los diferentes atributos HTML que éste el elemento tiene.

5.1.1.2. Ejecución de acciones:

La ejecución de cada acción depende principalmente del atributo guardado como "***action***" asignado durante la grabación del proceso (ver ilustración 3), cuando el usuario ejecuta dicha acción.

Selenium provee diferentes acciones como *click*, o *keys_enter,* por dar algunos ejemplos, que se usan como métodos para interactuar con un sitio web, además, provee una forma más compleja y a su vez precisa de ejecutar acciones a través de los *action chains*. Gracias a esto es posible implementar una condición en donde se verifica que la acción sea ejecutada a través del método sencillo, y si falla, sea ejecutada a través de los action chains.

5.1.2. Creación de la extensión para Google Chrome

La extensión web es el componente encargado de proporcionar una interfaz gráfica al usuario para administrar los diferentes Rpas, además de ser el encargado de comunicar con la Api REST y así interactuar con la base de datos. La extensión se compone principalmente de tres partes:

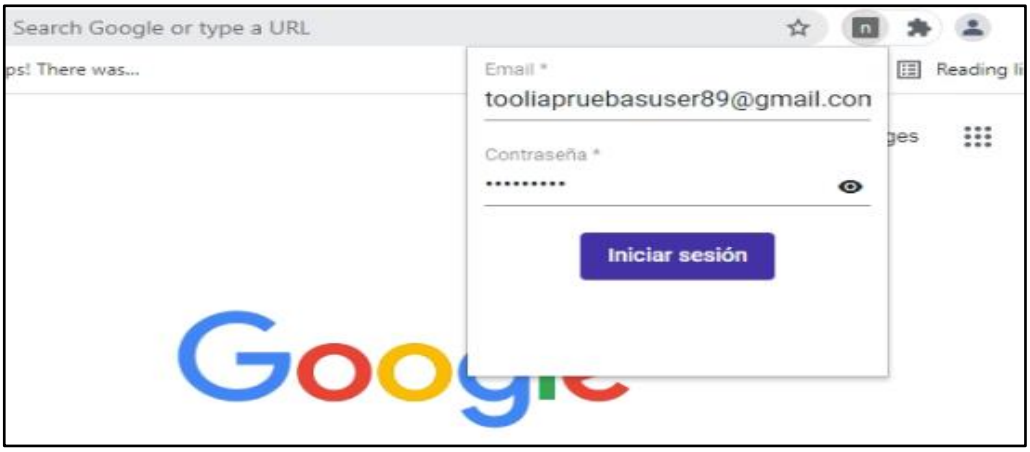
a) Interfaz de usuario

En este caso la herramienta principal es *Angular js*, teniendo en cuenta que proporciona componentes estilizados y listos para usarse.

Usando los componentes de Angular se implementan pantallas para:

- **Iniciar Sesión**

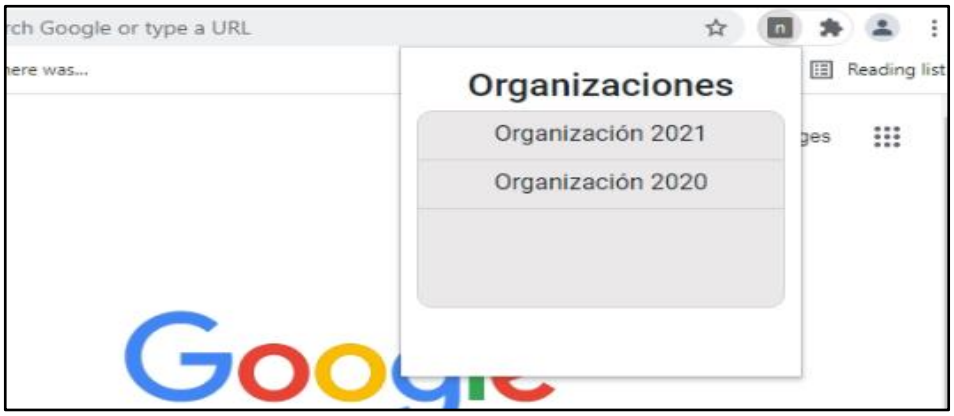
Ilustración 5. Inicio de sesión



Fuente: Autor

- **Seleccionar una organización**
- **Listar organizaciones**

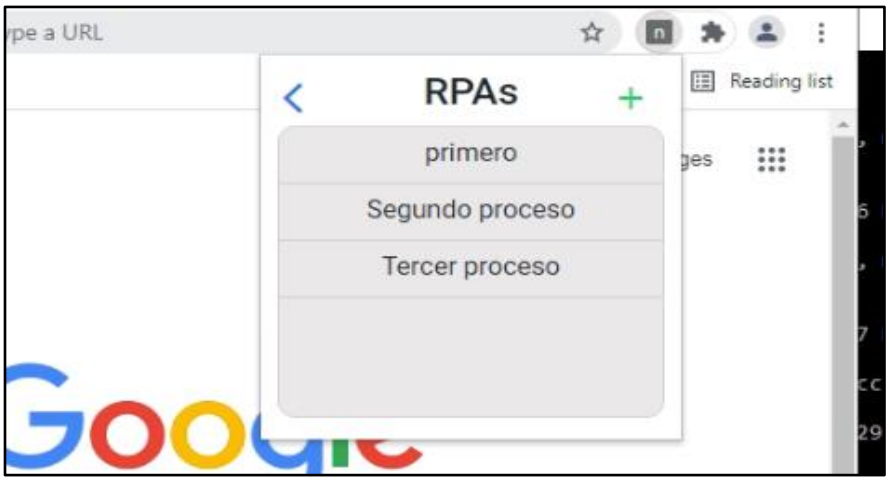
Ilustración 6. Lista de organizaciones



Fuente: Autor

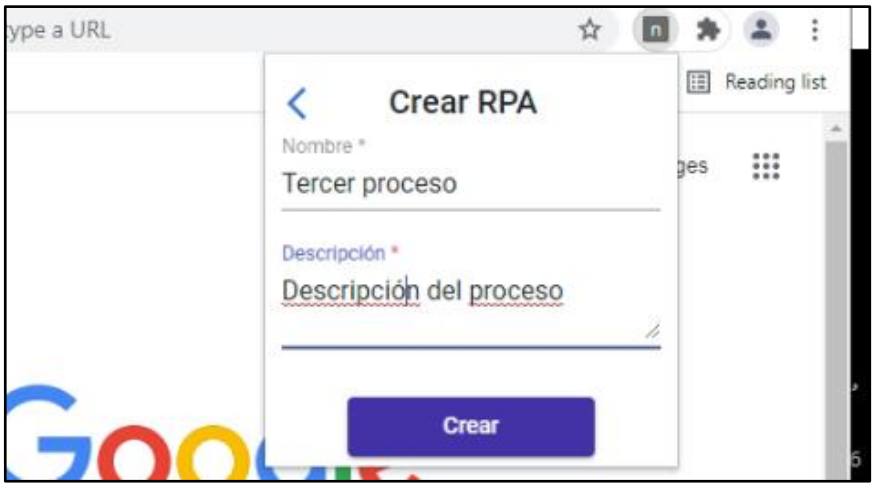
- **Listar, seleccionar, crear y borrar procesos**.

Ilustración 7.Lista de Rpas



Fuente: Autor

Ilustración 8. Creación de Rpa



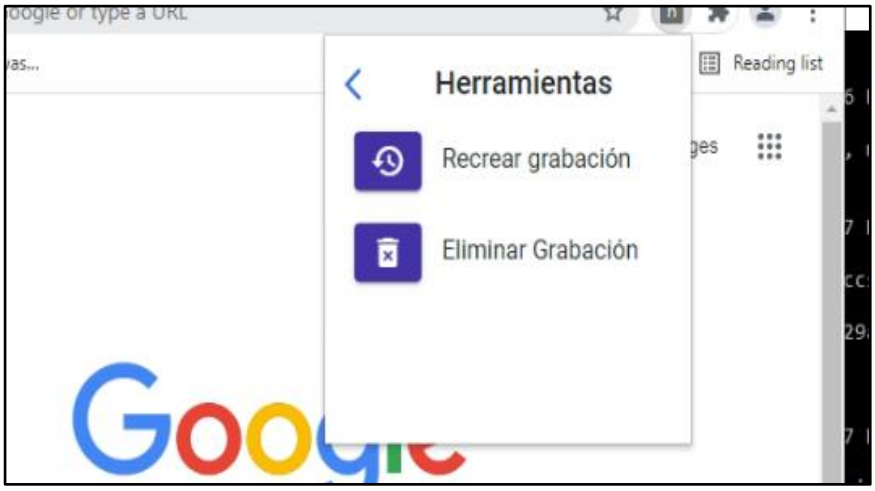
Fuente: Autor

A través de esta interfaz también se permite al usuario enviar los comandos para ***comenzar*** *y* ***finalizar*** un proceso de grabación, además es se permite enviar el comando para ***recrear*** un preso previamente grabado.

Ilustración 9. Comienzo de grabación Rpa

Fuente: Autor

Ilustración 10. Opciones de un Rpa



Fuente: Autor

b) Scripts de contenido

Son porciones de código en *JavaScript* cuyo contexto se limita a la *pestaña* y *página* actual donde el usuario se encuentre en un momento determinado.

Es en esta porción de código en donde se capturan los eventos ejecutados por el usuario dentro de un sitio web, ya que tienen el contexto actual de la página en que se encuentra situado el usuario.

c) Scripts de background

Son porciones de código en JavaScript cuyo contexto abarca todas las pestañas abiertas del navegador, incluyendo los datos que se reciben desde los *scripts de contenido*.

La comunicación entre estos componentes se realiza mediante el API que Google provee para interactuar con los diferentes contextos del navegador.

Se puede resumir el flujo de grabación de un proceso de la siguiente forma:

i. La interfaz de usuario (*Angular*) recibe las órdenes del usuario, y las envía como mensaje al *script de contenido* el cual tiene acceso a interactuar con el *DOM* que en dicho momento esté siendo renderizado para el usuario, y donde se capturan las diferentes acciones que el usuario ejecuta.
ii. Cada evento que es capturado se envía inmediatamente al *script background* que tiene la capacidad de persistir datos durante toda la sesión del navegador, que se cuenta desde que el navegador es abierto hasta que todas sus pestañas estén cerradas.
iii. Una vez el *usuario* termina la *grabación de proceso*, el *script en background* que tiene una *colección* con las diferentes *acciones* capturadas en cada *página* visitada por el *usuario*, y las envía al *backend* para ser procesadas y guardadas en la base de datos.

5.2. ANÁLISIS DEL DESARROLLO DEL PROYECTO

A continuación se describen los principales problemas que se encontraron en el desarrollo del proyecto, y la solución que se plantea para dar una solución

5.2.1. Recorrido del Grafo

Pesando teóricamente, *XPath* debería ser suficiente para ubicar con precisión cualquier elemento dentro de cualquier página de un sitio web, ya que es un lenguaje especializado en recorrer y procesar documentos *XML*, además, el resultado de cualquier *Framework* para *Front-End* es un archivo.html acompañado otros archivos compilados como JavaScript.

Pero en la práctica no lo es para todos los casos porque el toque personal de cada desarrollador se ve reflejado en el producto final que es el sitio web como tal. Entonces pudimos enfrentarnos a situaciones como por ejemplo diferentes elementos *HTML* con el mismo identificador, o por ejemplo comportamiento de visibilidad de componentes que el desarrollador sobre escribe de manera muy personal.

Estos comportamientos inesperados nos afectan tanto en la captura de los eventos como en la ejecución de los mismos, ya que lo primero en ambos casos es ubicar el *elemento HTML* implicado en la acción.

En las pruebas, identificamos también que los métodos ofrecidos por *Selenium Webdriver* para interactuar de forma sencilla con el sitio presentan comportamientos inesperados en algunos casos de prueba, y a su vez, *los actions chains*, ofrecidos también por *Selenium* presentan comportamientos inesperados en otros casos de prueba

Debido a lo anteriormente mencionado, aprovechando la base de datos, nos vimos forzados a implementar una función para recrear una primera vez (de prueba) el proceso y guardar el método provisto por *Selenium Webdriver* para ejecutar la acción (ver sección de métodos) como una propiedad del nodo que corresponde a dicha acción. Así, con esta *ejecución de prueba*, se intenta ejecutar la acción con

uno de los métodos. y en caso de fallar, el nodo que corresponde a la acción, será marcado como invalido, y así se crea una rama en el *grafo* a partir del *nodo* anterior, reemplazando el método de ejecución que ha fallado por la misma acción, pero ejecutada con un *action chain*.

De esta forma, es posible que la rama del grafo que corresponde al proceso guardado, pueda tener diferentes *subramas* inválidas, y una *subrama* válida que será la *rama* que se recorra las siguientes veces que se recree el proceso.

5.2.2. Inserción y captura de datos desde archivos externos

Hasta el momento hemos hablado de un automatizador de procesos que inserta los mismos datos que el usuario inserta en el momento de *grabación del proceso*, y la captura de datos se limita a los archivos que se descargan con la acción de un clic, y que son guardados en la carpeta de descargas del navegador.
Se hace una propuesta para la inserción y captura de datos desde fuentes externas de la siguiente forma:

5.2.2.1. Inserción de datos

Se presenta una propuesta para insertar datos desde un archivo en formato Json en el que la clave de cada ítem corresponde al nombre que el usuario le otorga a cada campo. Lo que requiere que el usuario tenga conocimientos básicos de estructura de datos.
Con ayuda de Angular, ofrecemos al usuario un menú tipo *pop up* que se muestra sobre cada elemento *input* de *HTML*, y le presente la opción al usuario de nombrar el campo, y enseguida insertar el valor o lista de valores que se ingresarán en cada ejecución del proceso.

5.2.2.2. Captura de datos

Se presente una propuesta en donde el usuario puede guardar:

- El *texto* de un elemento *HTML* específico
- Una lista de elementos *HTML que se guarda como un objeto Json* que corresponde a el *texto* de cada elemento *HTML* hijo de cada ítem de la lista a guardar
- Un archivo que se guarda en formato Base64, es decir, texto

6. CONCLUSIONES

- Fue posible desarrollar un componente de software que capture y guarde las acciones ejecutadas por el usuario en un sitio web, con la restricción de las acciones que involucren el evento **mouseover** propio de JavaScript

- Apoyándose del Framework Django y la base de datos Neo4J Se logró construir un componente de software que reciba las acciones capturadas, las organice y las guarde en la base de datos.

- Se logró construir una extensión web que permite al usuario grabar y recrear procesos ejecutados en un sitio web, Aunque se debe aclarar que la extensión web solo cumple con el papel de recibir la orden desde el usuario, y la transmite para se ejecutada por el componente que trabaja con *Selenium*

- No fue posible dejar un software que permita grabar los procesos ejecutados en todos los sitios web, debido principalmente a los errores que comente el desarrollador en cada sitio, que aunque no causan problemas en el funcionamiento normal del mismo, no permite que sea sencillo automatizar procesos dentro de él, pero es posible realizar una personalización en caso de que el sitio web lo requiera.

- El software en su completitud, permite grabar y recrear procesos ejecutados dentro de un sitio web. Se debe tener en cuenta la principal restricción ya que actualmente no se tienen en cuenta las acciones que involucren el evento ***mouseover.***

7. DISCUCIÓN

Es importante tener en cuenta que el esfuerzo llevado a cabo resulta siendo una prueba de concepto en el que se demuestra que es posible automatizar procesos dentro de un sitio web. Pudimos comprobar que no es posible llegar a una única versión del software en el que se generalicen todos los sitios web existentes.

El nivel de limpieza y simplicidad que le da un desarrollador a cada pieza de software, en este caso que tenga que ver con *Front-End,* impacta fuertemente en el desempeño de los diferentes robots y arañas que existen hoy para extraer datos de la web.

La herramienta construida además de ser un automatizador de procesos, podría ser un excelente asistente para Testers nivel junior y senior, así como a algunos automatizadores de pruebas que usen Selenium.

Es posible vender la herramienta teniendo en cuenta que se deben tener claros los casos de uso que se van a procesar, a partir de estos casos se realizaría una sesión de ajuste y puesta a punto, en donde se revise y asegure la ejecución de cada caso de uso.

8. PROPUESTAS DE MEJORA

Hasta ahora la herramienta se limita a sitios donde no se presenten comportamientos de mouseover, sería posible hacer la inversión de tiempo para agregar este comportamiento a la lista de eventos capturados. y así asegurar el funcionamiento de la herramienta en un porcentaje mayor de sitios y aplicaciones web.

Sería interesante intentar aplicar un modelo de *aprendizaje supervisado* en el que se le entreguen para cada proceso la lista de acciones que hoy en día tenemos, y el resultado esperado, y la *red neuronal* intente una y otra vez hasta que llegue a obtener el resultado que el usuario espera para cada proceso.

9. REFERENCIAS

[1] M. B. &. A. H. Wil M. P. van der Aalst, «SpringerLink,» 14 mayo 2018. [En línea]. Available: https://link.springer.com/article/10.1007/s12599-018-0542-4.

[2] D. M. R. Tejada, «Revista CIES,» 2020. [En línea]. Available: http://www.escolme.edu.co/revista/index.php/cies/article/view/286%20:.

[3] J. W. &. Sons, Front-end Development with ASP.NET Core, Angular, and Bootstrap, 2018.

[4] «Ideas SEM,» 2020. [En línea]. Available: https://ideassem.com/que-es/extension-de-navegador/ .

[5] M. Lopez, Enero 2021. [En línea]. Available: https://www.masterseosem.com/diccionario-seo-glosario-de-terminos-de-marketing-digital/etiquetas-html .

[6] «MDN Web docs,» 2021. [En línea]. Available: https://developer.mozilla.org/es/docs/Web/Guide/AJAX.

[7] M. W. Docs, 2021. [En línea]. Available: https://developer.mozilla.org/es/docs/Learn/JavaScript/Building_blocks/Events.

[8] J. J. Gutiérrez., «¿Qué es un framework web?,» 2020. [En línea]. Available: http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf.

[9] «Introduction to the Angular Docs,» [En línea]. Available: https://angular.io/docs.

[10] R. hat, «red hat,» [En línea]. Available: https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces#:~:text=Una%20API%20es%20un%20conjunto,de%20saber%20c%C3%B3mo%20est%C3%A1n%20implementados..

[11] D. Rubio, 2017. [En línea]. Available: https://link.springer.com/chapter/10.1007/978-1-4842-2787-9_12.

[12] W. A. V. Vargas, «Bases de datos orientadas a grafos y su enfoque en el mundo real,» 2014. [En línea]. Available: https://d1wqtxts1xzle7.cloudfront.net/32768000/trabFinalBDOG-BSDT-with-cover-page-v2.pdf?Expires=1628000251&Signature=GI5WtXlDRpxhXuMKVlJhAJhUdQC0Tg5fp0wLOe5WrSSpXvYRQksaQqW1a5Xb72c5mvgot03HQimYgeTUp1vZssjUm0A9sPBVPGuNpgQABGUa-vAIqFQ~1AYn0FsW7g7tj7YHg7mc5.

[13] R. V. Bruggen, Learning Neo4j, 2014.

[14] N. .INC, «neo4J.com,» [En línea]. Available: https://neo4j.com/developer/cypher/.

[15] S. Avasarala, Selenium WebDriver Practical Guide, 2014.

[16] J. Clark, «XML Path Language (XPath),» 1999. [En línea]. Available: http://new-design.renderx.com/files/demos/xmlspec/xpath/REC-xpath-19991116.pdf.

[17] L. W. a. others, «Document Object Model (DOM) Level 1 Specification,» 2000. [En línea]. Available: http://mail.granada.org/intranet/software.nsf/55510b195cfe9e3fc1257fda0032d515/7462079d1f806df7c1256f09003d0138/$FILE/w3c%20DOM.pdf.

[18] «Locating Elements,» 2021. [En línea]. Available: https://selenium-python.readthedocs.io/locating-elements.html.

[19] P. L. y. M. C. Penadés, «Métodologías ágiles para el desarrollo de software:,» [En línea]. Available: https://uvirtual.unet.edu.ve/file.php/419/metodologia_agil_xp_bueno.pdf..