UNIVERSITAT POLITÈCNICA DE VALÈNCIA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Politécnica Superior de Gandia

Desarrollo de una aplicación para automatizar procesos de ofimática usando el software libre Selenium

Trabajo Fin de Grado

Grado en Tecnologías Interactivas

AUTOR/A: Calabuig Chafer, Brian

Tutor/a: Marín-Roig Ramón, José

Cotutor/a externo: BELLO MEDINA, JUAN

CURSO ACADÉMICO: 2021/2022

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCOLA POLITÈCNICA SUPERIOR DE GANDIA

“ Aplicación para automatizar procesos usando el RPA de software open source Selenium ”

TRABAJO FINAL DE GRADO

Autor/a: Brian Calabuig

Tutor/a: José Marín-Roig

GANDIA, 2022

Resumen

A día de hoy ya está muy normalizado el uso de robots y autómatas en la industria, automatizando y facilitando la producción. Además, la irrupción de los software de automatización están trasladando estas praxis al desarrollo de software y empresas digitales, impulsadas por los RPA (Robotic Process Automation) y las Inteligencias Artificiales. Bajo el auge de estas tecnologías, nace este proyecto, aprovechando el poco desarrollo en este campo, para investigar y desarrollar una aplicación capaz de automatizar procesos web en el navegador mediante uno de los RPA más populares actualmente, Selenium. En este documento se explica todo el procedimiento realizado para su desarrollo, desde su estudio previo a su implementación.

Palabras claves: RPA, Selenium, robot, bot, automatización, robótica, proceso

Abstract

Today, the use of robots and automatons in industry is already very standardized, automating and facilitating production. In addition, the emergence of automation software is transferring these practices to the development of software and digital companies, driven by RPA (Robotic Process Automation) and Artificial Intelligences. Under the boom of these technologies, this project was born, taking advantage of the little development in this field, to investigate and develop an application capable of automating web processes in the browser using one of the most popular RPA currently, Selenium. This document explains the whole procedure carried out for its development, from its previous study to its implementation.

Keywords: RPA, Selenium, robot, bot, automation, robotic, process

Índice general

Índice de imágenes

1. Introducción

1.1. Introducción

A día de hoy, uno de los mayores objetivos de las empresas y negocios es aplicar la digitalización y transformar los negocios convencionales en negocios digitales, favorecidos por la globalización que sufre el mundo. Junto a esto, el auge en la automatización y la robotización de las industrias hace que el sector empresarial esté experimentando un cambio radical en los procesos y procedimientos a realizar durante su día a día.

Esta automatización favorece el mayor volumen de tareas realizables en un día. Al igual que la robotización de las industrias favoreció en su momento una mayor producción en las industrias de productos. Por tanto, estamos en un punto en el que las empresas dirigidas al desarrollo de software, gracias al uso de los propios software, incrementan la productividad de manera exponencial, imitando así el modelo de las fábricas.

Este escenario se ve potenciado, en parte por los software RPA o de automatización robótica de procesos, que es en los que nos vamos a centrar en este proyecto. Por tanto, se realizará un estudio de cómo está el mercado actual y cuáles son las herramientas o softwares más populares. De esta selección, nos decantamos por el que más se adapte a lo que necesitamos para el desarrollo de la aplicación propuesta.

Una vez seleccionado uno, y desarrollada la lógica de la aplicación, es momento de la parte visual o UI de esta. Para ello, contamos con las aplicaciones low code que se pueden desarrollar con las herramientas de la empresa colaboradora Aura Quantic, que son aplicaciones las cuales no necesitan código para su desarrollo.

Por último, después de un estudio de mercado, se decide desarrollar un sitio web en el cual poder distribuir los diferentes software de la empresa, incluido este proyecto, con el fin de dar visibilidad a los productos de la empresa y facilitar el acceso de estos a los posibles clientes.

En este documento se detalla todo el estudio previo realizado antes de elegir la herramienta más óptima para el desempeño que tenemos entre manos, así como el proceso de diseño e implementación de la misma, finalizando con las conclusiones del proyecto.

1.2. Motivación

La motivación que hay detrás de este proyecto es la de desarrollar una aplicación que automatice procesos realizados mediante el navegador con el apoyo del software libre llamado Selenium, debido a la escasez de herramientas que proporcionan esta funcionalidad en el mercado, pudiendo así entrar de lleno en el mercado con una aplicación de bajo coste.

Además es una tecnología muy versátil con la que se pueden realizar diversas funcionalidades en un futuro, por tanto la investigación de esta y el estudio a realizar serán muy beneficiosas y enriquecedoras, pudiendo adaptar esta herramienta a otros usos.

Por último, la satisfacción de contribuir a una tecnología en desarrollo es otra de las principales motivaciones para decantarse por este proyecto, ya que da una oportunidad única de aprender y crecer junto a la comunidad y el software en cuestión.

1.3. Objetivo y plan de acción

El objetivo marcado del proyecto es conseguir desarrollar una aplicación para automatizar procesos del navegador usando el RPA de software open source Selenium.

El plan de acción para lograr el objetivo definido está formado por los siguientes pasos:

1. Analizar el marco teórico en el que se encuentra el desarrollo de las herramientas de automatización robótica de procesos (RPA).

2. Estudiar la documentación del software Selenium para manejarse con soltura.

3. Estudiar las diferentes alternativas que hay para implementar servidores con su capa de conexión y decantarse por la opción más óptima.

4. Integrar el software Selenium al servidor desarrollado.

5. Acoplar el sistema terminado a las aplicaciones low-code de Aura Quantic.

6. Diseñar un sistema gratuito de distribución de software o tienda de aplicaciones en la cual poder distribuir el servicio desarrollado, además de otros software de la empresa.

1.4. Metodología

En esta sección se describe la metodología de trabajo empleada durante el desarrollo del proyecto, además de la herramienta de control de versiones usada.

1.4.1. Scrum

Durante el completo desarrollo del proyecto se ha optado por emplear la metodología ágil Scrum, la cual consiste, en un primer momento, en el desarrollo de lo que conocemos como PVM o producto mínimo viable. Este concepto hace referencia a un primer estado del proyecto el cual ya se podría comercializar puesto que cumpliría ya los requisitos mínimos. Lo que enriquece a este método son las posteriores entregas, denominadas sprints, que suelen ser periódicas de un tiempo previamente acordado, en las cuales se van incluyendo funcionalidades al proyecto con el fin de mejorarlo lo máximo posible.

Posterior a las entregas de los sprints se suelen realizar una reuniones con el Product Owner del proyecto (en este caso los tutores del mismo) denominadas Sprint Review, en las cuales se recapitula y evalúa lo realizado hasta la fecha y además poder planear los siguientes hitos a conseguir para el próximo sprint.

Referente a este proyecto, los sprints se han fijado en una fecha límite de 2 semanas, tras las cuales se hace una retrospectiva con el tutor de la empresa y se definen los siguientes pasos, aumentando así la productividad y la gestión del proyecto.

1.4.2. Git. Control de versiones

A la hora del control de versiones del proyecto se ha optado por la herramienta Git, siendo la más popular entre los desarrolladores de software, y teniendo una gran comunidad. Se ha trabajado con un solo repositorio en el que se han incluído todas las partes del proyecto. En lo referente a funcionalidades extra que proporciona la herramienta Git, se ha decidido trabajar mediante Git Flow y el uso de ramas, eligiendo la opción más común de crear 2 ramas denominadas *master* y *develop*, siendo la primera de estas la rama estable en la que solo entra código testeado y funcional, mientras que todo el desarrollo y pruebas se realizan en la rama *develop*, evitando así romper el proyecto y teniendo siempre una versión estable alojada en la rama *master*.

2. Estado del arte

A raíz de la segunda revolución industrial, se empezaron a desarrollar técnicas de automatización de las tareas realizadas en las industrias con tal de mejorar y optimizar la producción de éstas. Más tarde, comenzó la digitalización con la irrupción de las computadoras y ordenadores en el sector industrial. Recientemente, en combinación de ambas tendencias, se busca la automatización de tareas y procesos realizados mediante arquitecturas de procesadores, con tal de abaratar los costes de estas tareas y aumentar su eficiencia. Previo al diseño y desarrollo del proyecto, se realizará un pequeño estudio del estado actual de ambas tendencias mencionadas y de las diferentes tecnologías y herramientas surgidas para que éstas se asienten en el sector industrial de hoy en día.

2.1. Digitalización y automatización

La digitalización es un proceso de transformación en el que las empresas habilitan, mejoran o evolucionan sus funciones empresariales, operaciones comerciales, gestiones de clientes y/o procesos de comunicación mediante las tecnologías digitales.

La automatización refiriéndose a la automatización de los procesos de negocio (**BPA,** *Business Process Automation*), que es la que nos repercute a nosotros, es la automatización de aquellos procesos o tareas que se repiten periódicamente o aquellos que requieran de una alta dificultad para su realización a través de la habilitación de la tecnología.

Encasillando estas tendencias a los tiempos actuales, vemos como la gran mayoría de las empresas han estado los últimos años digitalizando poco a poco sus negocios, tendencia que se ha visto incrementada exponencialmente después de la pandemia, ya que, según la revista Forbes, el 80% de las empresas en el mundo han adelantado su transformación digital, mientras que en España han sido 7 de cada 10 empresas las que se han digitalizado como respuesta a la situación actual forzada por el covid-19. Como vemos, es una tendencia que iba en aumento y que tarde o temprano iba a deparar en la digitalización de la totalidad de las empresas mundiales. La Unión Europea espera que en 2030 se complete la digitalización del 100% de sus empresas. Asimismo, la digitalización implica la aparición de muchas tareas de gestión, contabilidad, etc. realizadas mediante computadoras, lo que hace que los trabajadores tengan que realizar todas estas tareas a diario de manera repetitiva. Por tanto, esta situación concluye en que las empresas busquen automatizarlas, ahorrando así costes, tiempo de sus trabajadores y aumentando la

eficiencia. Siguiendo con la información de antes, según la empresa Kaizen Institute, mediante su estudio y análisis del sector, hasta un 53% de las empresas españolas apuestan por la automatización de sus procesos de negocio.

Una vez observado el panorama actual, vamos a estudiar las tecnologías que hacen posible que las empresas hayan empezado esta transformación digital, en concreto, el deseo de automatizar la gran parte de los procesos que realizan a diario.

La primera tecnología que vamos a ver, son las plataformas de desarrollo low-code o de bajo código, que son una manera de desarrollar aplicaciones sin necesidad de saber programación. Esta tecnología facilita a personas no técnicas implementar procesos de automatización, diseñando un proceso de manera visual, similar a un diagrama de flujo, mediante herramientas *drag and drop* (arrastrar y soltar) situadas en la interfaz gráfica de la plataforma low-code. Esta tecnología permite que la automatización de procesos sea accesible para prácticamente cualquier persona, no solo para personas técnicas o programadores, ampliando enormemente las capacidades de una empresa en conseguir la automatización.

Algunas otras tecnologías que permiten la digitalización y la automatización son aquellas comprendidas en la inteligencia artificial, como pueden ser procesos de machine learning y deep learning, asistentes virtuales o chatbots, big data, etc.

Para finalizar vemos los tipos de automatización de procesos existentes, siendo estos las macros, la automatización de procesos de las tecnologías de la información (**ITPA**) y la automatización robótica de procesos (**RPA**).

2.1.1. Tipos de automatización de procesos

- **Macros**: es la más asentada de las tres, estando disponibles desde hace décadas. Son una serie de funciones que permiten automatizar tareas repetitivas, complejas o habituales en bases de datos o cualquier otro sistema informático. Aunque permiten la automatización de dichos procesos, las macros tienen algunos inconvenientes, siendo algunos de ellos que las secuencias son limitadas y no admiten variación, además de que deben ser establecidas previamente. Añadido a todo esto, mediante las macros no se puede prescindir por completo de la presencia humana ya que necesitan de algún operario que pulse el botón y que aporte la información y se asegure de su calidad.
- **ITPA**: este paradigma consiste en la mejora de la eficiencia al reducir el trabajo manual en la ejecución de tareas de tecnologías de la información

rutinarias, como actualizar el sistema o la configuración de nuevos servidores.

- **RPA**: esta tecnología está en auge, siendo la más reciente y la que mayor proyección de futuro tiene. Permite que robots (siendo estos completamente software, no físicos) se encarguen al 100% de toda clase de tareas administrativas o de ofimática que sean repetitivas. A diferencia de los dos modelos anteriores, los rpa cuentan con una flexibilidad e integración mucho mayor, haciendo parecer que quien se encarga de las tareas es un trabajador, aunque no sea humano.

Una vez conocidos los tipos de automatización presentes en la industria, vamos a profundizar en los RPA, ya que son el modelo más interesante para el desarrollo del proyecto propuesto.

2.2. Automatización robótica de procesos (**RPA**)

Como hemos mencionado antes, un RPA es un tipo de automatización de los procesos de negocio que replica las acciones de un ser humano interactuando con la interfaz de usuario de un sistema informático, como pueden ser los navegadores web, programas de ofimática o de gestión y contabilidad.

Dentro de las soluciones RPA encontramos dos tipos de tipología: por una parte tenemos a los robots que trabajan sobre procesos deterministas empleando datos estructurados y procesos basados en reglas, mientras que por otro están los que permiten gestionar procesos no definidos, mediante técnicas de aprendizaje automático (machine learning) y empleando datos no estructurados, siendo esta última una solución que combina la automatización con el uso de técnicas avanzadas de inteligencia artificial, dando como resultado una tipología mucho más versátil y con más aplicaciones futuras.

2.2.1. Evolución histórica del concepto

El concepto de automatización existe desde hace mucho tiempo, pero el concepto de RPA se considera como una evolución tecnológica significativa de esta actividad puesto que recientemente han aparecido gran cantidad de plataformas de software que hacen posible de manera sencilla que las grandes empresas apuesten por el enfoque de la automatización y empiecen a aumentar el uso de dichas plataformas. Si estas plataformas de software no fueran sencillas y fiables, la gran mayoría de grandes corporaciones serían reacias a implementarlas debido a los posibles riesgos que supondría en la calidad y reputación de la propia empresa.

El concepto de RPA se acuñó en la aparición de robots que facilitan la automatización de los procesos de negocio. Dichos robots no hacen referencia a las típicas máquinas presentes en el sector industrial o a las máquinas con aspecto humanoide, típicas de películas de ciencia-ficción, que se nos vienen a la mente al escuchar la palabra robot. Más bien se adhiere al significado de robot virtual, siendo abstracto para la percepción humana, por tanto, refiriéndose a todo programa informático o software desarrollado para la realización de las tareas propias de la automatización.

2.2.2. Contraposición de RPA con Business Process Management (**BPM**)

Aunque ambos conceptos se pueden confundir y entender como sinónimos, la verdad es que existen grandes diferencias entre ellos.

Anteriormente hemos hablado de la automatización de los procesos de negocio (BPA), que se basa en la automatización de algunos BPM, siendo un tipo de estos. Pero, ¿qué son los BPM? Hablamos de BPM o gestión de procesos de negocio como un conjunto de estrategias que se utilizan para agilizar y optimizar los procesos tanto automatizados (BPA) como manuales. Aunque ambos conceptos operan con una lógica similar basada en eventos, acciones, condiciones y bucles, la diferencia esencial radica ahí, en el contexto sobre el que se aplican.

Mientras que los BPM tienen como objetivo asegurar que la infraestructura operacional sea sólida, afianzando así la base sobre la que la empresa opera, facilitando un flujo de trabajo ordenado el cual integra usuarios, sistemas y datos de manera coordinada, los RPA se usan para realizar tareas repetitivas o complejas tal como las haría una persona, pero realizadas por un software aumentando así la velocidad de ejecución y la eficiencia. Por tanto, observamos que los RPA actúan de manera superficial en la base implantada por los BPM, permitiendo que ciertas tareas dentro del flujo de trabajo se realicen de la forma más eficiente posible.

En definitiva, lo mejor para una empresa es combinar ambos conceptos, estableciendo un flujo de trabajo adecuado y unas bases sólidas gracias a los BPM, y analizando este flujo de trabajo, lo ideal sería poder detectar en qué momento del flujo aparecen los cuellos de botella o elementos que disminuyen la productividad para poder solventarlos, y si es posible, aplicar RPA en la solución.

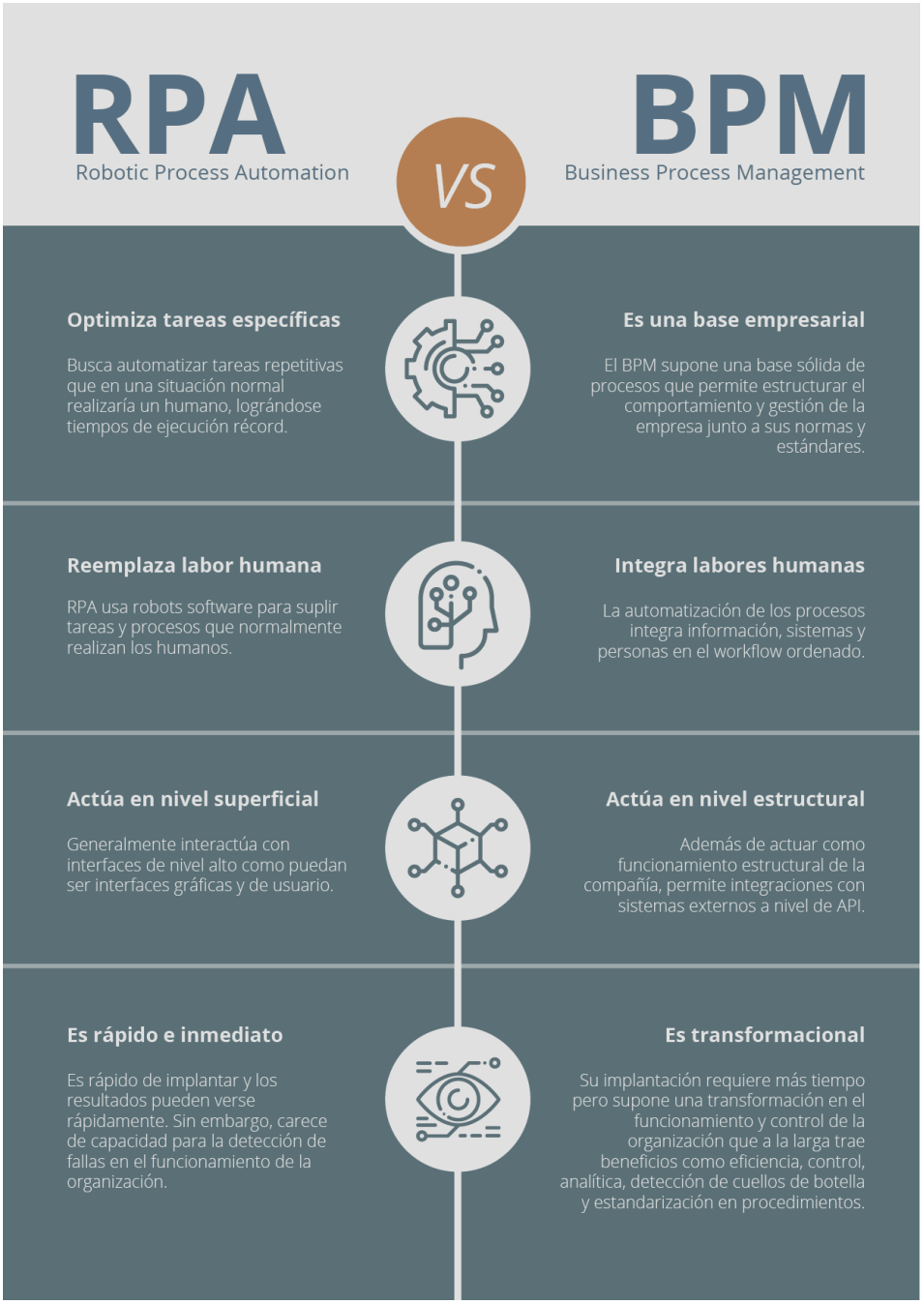


Imagen 1. RPA vs BPM.

2.2.3. Beneficios de la robotización RPA

La aplicación de software RPA trae consigo una gran cantidad de beneficios para las empresas, que son los que consiguen que éstas estén interesadas en aplicar esta transformación digital.

El primero de los beneficios más claro es el ahorro de tiempo y costes que implica, siendo esta tecnología entre 10 y 15 veces más rápido que un humano, además de poder destinar a los trabajadores a otras labores que no puedan ser realizadas por robots aligerando el trabajo que deben realizar los trabajadores.

En adición, los robots y los programas son capaces de trabajar sin descanso durante las 24 horas del día, durante los 7 días de la semana, así todo el año inclusive los días festivos.

Otro gran beneficio que otorga la implementación de tecnologías RPA es la mejora de la productividad, ya que mejora la eficiencia de los procesos entre un 50 y un 90%. Asimismo, esta mejora de la eficiencia también se ve respaldada por la ausencia de errores de estos robots, ya que la probabilidad de que éstos tengan alguna equivocación es prácticamente nula.

Hablando del tema económico otra vez, para implementar la robotización hay que hacer un desembolso inicial. Pero esto no resulta un problema puesto que el ahorro de costes que te otorga a largo plazo hace que la inversión inicial se amortice en tan solo 3 meses.

Para finalizar, otro beneficio de estos software es que no resulta intrusivo, lo que facilita su implementación sin causar ningún tipo de disrupción en los sistemas que ya hay en una institución. Los robots se conectan de forma sencilla a las aplicaciones de la empresa, lo que deriva en el último beneficio más notorio de este tipo de tecnologías, que es su capacidad para la integración, la cual permite combinarlos con otras tecnologías punteras como la inteligencia artificial, el reconocimiento óptico de caracteres, y cualquier otra tecnología o funcionalidades que requiera la empresa.

En resumen, la automatización robótica de procesos promete un ahorro de costes, de tiempo, fiabilidad, y eficiencia en procesos que se repitan de forma periódica o que contengan una gran carga documental, evitando posibles errores humanos.

2.2.4. ¿Cuándo aplicar soluciones RPA?

Una vez conocemos lo que es un RPA y de los beneficios que nos ofrece, las preguntas que debemos hacernos son, ¿cuándo deberíamos aplicar este tipo de soluciones y si a todos los procesos se les puede aplicar? Para poder contestar estas cuestiones, hay que hacer un estudio de los procesos que posee la empresa, pero de manera general se pueden contestar cómo vamos a exponer a continuación.

Según un estudio realizado y publicado por los informáticos *Wil M. P. van der Aalst*, *Martin Bichler* y *Armin Heinzl* en *Springer Fachmedien Wiesbaden GmbH*, parte de S*pringer Nature 2018*, existen 3 tipos de casos de procesos en una empresa, siendo estos la automatización de procesos tradicionales, los procesos candidatos a ser automatizados mediante RPA y por último, aquellas tareas o procesos que solo pueden ser realizadas por trabajadores humanos.

La primera de las mencionadas hace referencia a aquellos procesos que tradicionalmente son automatizados debido a que suelen tener estructuras similares y hacen la automatización más viable económicamente. Por otra parte, la última citada comprende aquellas tareas que, como su propio nombre indica, solo las pueden realizar seres humanos. Por tanto, se concluye que todas las tareas y todos los procesos que no se pueden clasificar como ninguno de estos 2 casos corresponden al caso que resta, que son los otros posibles candidatos a ser automatizados mediante tecnologías de RPA, ya que son tareas repetitivas pero no lo suficiente como para justificar al 100% su automatización.

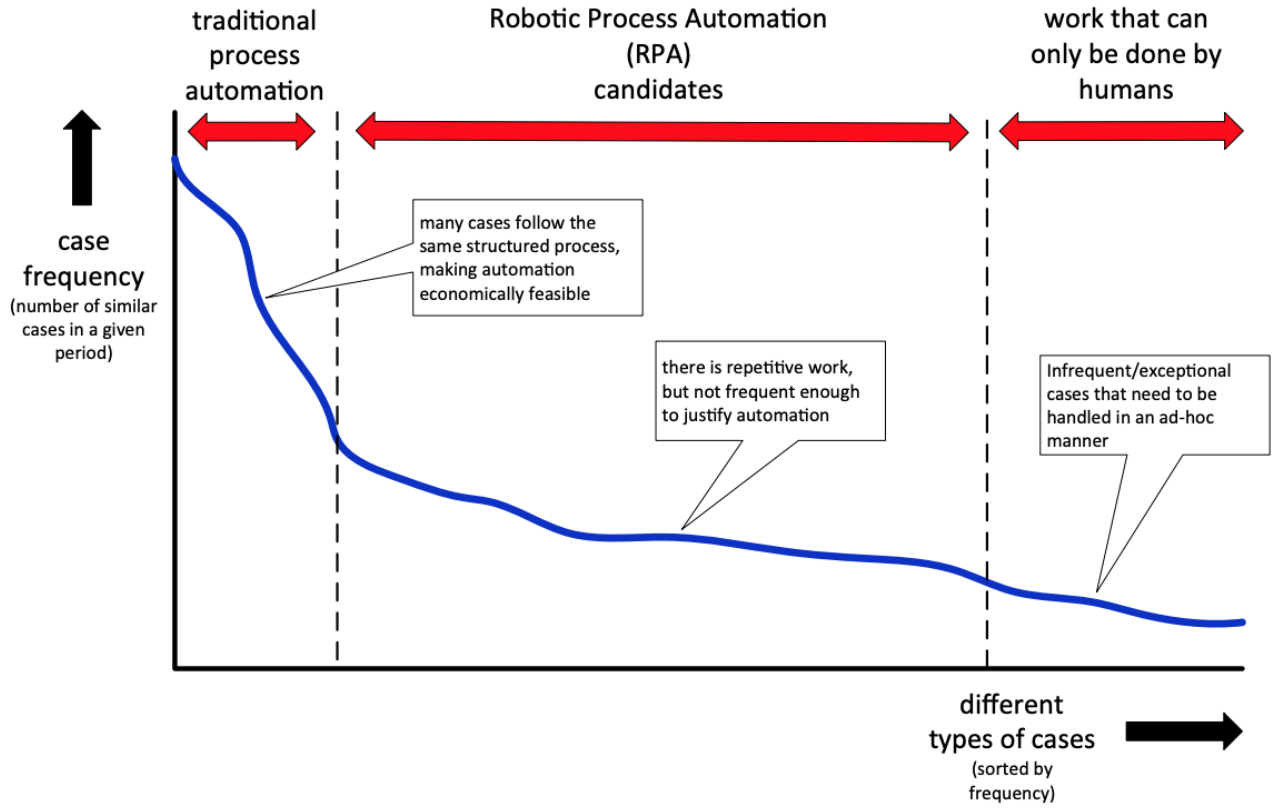


Imagen 2. Gráfica para determinar cuándo aplicar una solución RPA.

Continuando con el artículo publicado por los informáticos teóricos citados anteriormente, estos dicen que en este gráfico se puede aplicar el principio de pareto, el cual dice que el 80% de los casos se puede explicar con el 20% de los tipos de casos. Es decir, el 80% de los procesos de una empresa se incluyen en el primer caso de automatización de procesos tradicionales, y el 20% restante comprendidos por los dos tipos restantes, incluye tanto los posibles procesos automatizables como los que no se pueden automatizar, siendo ambos los menos frecuentes.

Entonces, respondiendo a las cuestiones iniciales, más del 80% de los procesos son automatizables, incluyendo a los que son viables para ser automatizados al 100%, como a los que hay que estudiar si sería posible, tanto económicamente como por recursos, automatizarlos con el fin de mejorar la eficiencia de la empresa sin comprometer los fondos y los recursos de la empresa.

2.3. RPAs en la industria

Durante los años 90 se empezaron a implantar gigantescos sistemas BPM corporativos como los propuestos por IBM o Oracle. Más tarde, con llegada de los 2000, surgió la aparición de las plataformas de software de automatización para complementar los sistemas BPM. Después de una sequía de una década sin avances tecnológicos en la materia, en el último lustro se afianzó la tecnología de los RPA, irrumpiendo con fuerza en el sector industrial contemporáneo. A medida de la evolución de esta tecnología, los líderes de la industria han estado reenfocando a sus empleados para que realicen más tareas "basadas en el ser humano" (como por ejemplo, interacciones cara a cara con el cliente), dejando así las tareas que no requieran de la presencia humana para los software RPA. A continuación veremos la implicación de esta tecnología en los diferentes sectores existentes.

2.3.1. Industria de los seguros

Este sector tiene mucho que ganar con el uso de los RPA debido al gran volumen de demandas de reclamos por las que una compañía de seguros debe navegar en el día a día, aumentando así la velocidad de resolución de éstas. Con los clientes de hoy en día que desean un servicio más rápido y más inteligente, los sistemas antiguos deben actualizarse para cumplir con esas expectativas, por eso, y por la cantidad abrumadora de procesos repetitivos que suelen tener cualquier tipo de de compañia de seguros, la industria se ha volcado hacia el uso del software de automatización de procesos.

2.3.2. Industria bancaria

Con una enorme proporción de documentos de cuenta, el sector de la banca también necesitaba encontrar un sistema de automatización de estos procesos. La integración de los RPA permite que los tiempos de procesamiento se agilicen además de eliminar los errores humanos.

A parte de los beneficios de siempre, este sector se ve beneficiado con la automatización de los procesos referentes al cumplimiento normativo, puesto que la normativa suele ser cambiante, consiguiendo así que los trabajadores no tengan que revisar manualmente la información y que las auditorías sean menos problemáticas.

2.3.3. Sector sanitario

En el sector del cuidado de la salud, los RPA se encargan de las tareas como la programación de pacientes, el procesamiento de reclamos, la entrada de datos y la facturación, evitando que los trabajadores de los hospitales y centros de salud se tengan que ocupar de estas tareas administrativas y puedan centrarse en mejorar la calidad de la relación con el paciente, con todo lo que ello implica.

2.3.4. Industria de las telecomunicaciones

Respecto a este sector, los RPA se concentran en la automatización con el fin de mejorar la comunicación de datos y aumentar la eficiencia operativa, ya que nunca se podría automatizar las tareas pertenecientes a la atención del cliente. Pero, a pesar que las telecomunicaciones implican un diálogo entre dos personas, hay casos en los que una de estas personas se podría sustituir, ya que existen softwares capaces de procesar una entrada y discernir la salida deseada, como sería el caso de los chatbots.

2.3.5. Sector de la fabricación

Debido a que este sector es el más repetitivo y cuenta con una sólida infraestructura, es de los más idóneos para la implantación de tecnologías RPA, haciéndose éstas cargo de las tareas administrativas y de contabilidad y facturación, además de fortalecer los procedimientos de su cadena de suministro.

2.3.6. Aura Quantic

Una vez comprobado el papel de los RPA en los diferentes sectores del mundo empresarial actual, vamos a presentar la empresa Aura Quantic, empresa colaboradora para la realización de este proyecto, y el papel que juegan la automatización y los RPA en ella.

AuraQuantic es una empresa española, fundada en el 2002 como plataforma de automatización de procesos de negocio sin código y con una gran presencia global.

Imagen 3. Presencia de AuraQuantic en todo el mundo.

Inicialmente fue diseñada como una plataforma de soluciones intelligent Business Process Management (iBMP), pero en la actualidad ha evolucionado convirtiéndose en una plataforma que permite la creación de aplicaciones low-code o de bajo código. Presenta una interfaz sencilla e intuitiva para el usuario, además de poseer una gran cantidad de conectores que facilitan la integración de estas tecnologías, como pueden ser Google suite, que permite el uso de las funcionalidades de google, conectores para el uso de los programas de Microsoft Office, o conectores para el uso de API REST.

La tecnología BPM de Aura Quantic posee, por definición, gran parte de las características que se le pueden pedir a una tecnología RPA, por lo que Aura Quantic puede utilizarse como entorno de automatización software, contando con las capacidades siguientes:

- **Control de workflow**: a través de un diagrama de flujo se construye la lógica del proceso. Posteriormente se definen los atributos necesarios para hacerlo funcionar.
- **Personalización completa**: admite el diseño de cualquier proceso, independientemente de su complejidad.
- **Bases de datos**: cuenta con bases de datos propias para poder almacenar cualquier tipo de información.
- **Procesamiento de la información**: todo dato importado al proceso puede ser procesado, almacenado o enviado a otro sistema o proceso.
- **Tareas de sistema**: ejecutan tareas de manera automática, entre las cuales se incluyen desde el envío de correos electrónicos a la ejecución de scripts personalizados, pasando por otras muchas tareas de diferente índole.
- **Conectividad**: permite conexiones con cualquier tipo de dispositivo, sistema, aplicación o base de datos externa mediante servicios web.
- **Conectores nativos**: conectores propios que permiten la integración con otras aplicaciones de software externas como Microsoft Office o Web Services.

En conclusión, la empresa Aura Quantic cuenta con una plataforma de automatización bastante completa, basada en tecnología BPM como sistema estructural, pero tiene la intención de poder implementar la tecnología RPA como una posible solución a problemas específicos, previo análisis de los cuellos de botella que sufre ésta, con el fin de paliarlos y dando así un paso más en la evolución digital que trae consigo la tecnología RPA.

2.4. Principales herramientas RPA

Una vez analizado el impacto de los RPA en los diferentes sectores de la industria, procedemos a exponer las herramientas más importantes y conocidas basadas en la tecnología RPA, siendo éstas softwares privativos pertenecientes a diversas corporaciones. Además de estas herramientas de carácter privado, se analizarán los framework o librerías existentes de código libre o *open source* basadas en la tecnología RPA, con la finalidad de escoger el más óptimo para la realización de este proyecto, evitando así recurrir a softwares de pago. Estas últimas, por lo general, son menos comunes que los softwares privativos y, además, requieren de más trabajo para el entrenamiento de dicha herramienta, a menudo teniendo que escribir código en un editor, pero dadas su versatilidad y capacidad de adaptación, las hacen las más idóneas para la implementación de este proyecto.

2.4.1. Herramientas de software privativo

2.4.1.1. Microsoft Power Automate

Power Automate es una herramienta de Microsoft la cual permite crear flujos de trabajo conectando 2 o más aplicaciones, mediante conectores, con el fin de ahorrar tiempos en las tareas de una empresa. Estos flujos pueden ser para tareas de Office 365 o aplicaciones externas a Microsoft, inclusive pueden ser mejorados mediante trabajos de la inteligencia artificial incorporados por AI Builder o Power BI. Los flujos de trabajo son automatizados y escalables, además no requieren de conocimiento de programación para su uso, aumentando así su eficiencia. Es una mejora sustancial del antiguo producto Microsoft Flow, obteniendo el reconocimiento como líder en el sector de la automatización en el 2021 por grandes corporaciones del asesoramiento como Gartner y Forrester.

2.4.1.2. UiPath

Esta herramienta, de la empresa con el mismo nombre, está diseñada para automatizaciones de escritorio para Windows. Es una de las empresas líderes en el sector, presentando una herramienta muy pulida, que aunque esté limitada a las automatizaciones de escritorio en Windows, funciona a la perfección, aumentando la eficiencia de sus usuarios. Además, UiPath se está extendiendo hacia la inteligencia artificial y herramientas de visión artificial que pueden extraer información de imágenes o de la propia pantalla. Asimismo, permite la integración con infinidad de lenguajes de programación para aquellos usuarios más técnicos, como pueden ser C#, Python, java, etc.

2.4.1.3. Blue Prism

Una de las empresas pioneras en la tecnología de los RPA, siendo ésta la que acuñó el término de *Robotic Process Automation*. Empezó su actividad en el año 2012, convirtiéndose en otra de las empresas líderes del sector, a día de hoy. Su herramienta consiste en una plataforma de bajo código con la cual entrenar a los robots para que emulen a los humanos a la hora de realizar tareas, respaldada mediante inteligencia artificial y machine learning con el fin de suavizar el comportamiento de los robots.

2.4.2. Herramientas de software open source o de código abierto

2.4.2.1. Puppeteer

Esta herramienta es una librería de Node.js, desarrollada y respaldada por Google, que proporciona una API de alto nivel que permite automatizar acciones sobre los navegadores de Google. Algunas de las acciones que puede realizar son: generar capturas de pantalla, automatizar el envío de formularios o entradas de teclado, capturar un seguimiento de la runtime de un sitio web para diagnosticar problemas de rendimiento, crear un entorno de pruebas automatizado, etc. Aunque es una herramienta muy interesante, el hecho que sea solo compatible con Google Chrome y Node.js (javascript), hace que no nos decantemos por ésta para cimentar las bases de nuestro proyecto.

2.4.2.2. Selenium

Es un entorno de pruebas de software para aplicaciones basadas en web. Permite automatizar infinidad de tareas de un navegador, ya sea sin el uso de código mediante el Selenium IDE (interfaz gráfica), o si se requiere, con el uso de scripts personalizados. Dichos scripts admiten una cantidad considerable de lenguajes de programación, entre los que destacan python, javascript, java, ruby, .net o php. Asimismo, este framework es compatible con los navegadores más usados, como Firefox, Microsoft Edge, Google Chrome o Safari, en cualquiera de los sistemas operativos de linux, en Windows incluso en Mac OS.

Este framework contiene una serie de componentes que lo hacen ser una herramienta muy completa. A continuación procederemos a exponerlos y dar una breve explicación de cada uno de ellos.

El primer componente que nos encontramos es el citado anteriormente como Selenium IDE. Es un entorno de desarrollo integrado o interfaz gráfica, diseñado como una extensión del navegador con soporte en Chrome y Firefox, con la que se puede interactuar mediante botones para grabar acciones las cuales se quieran reproducir con exactitud en momentos futuros. Estas acciones pueden ser desde simples cliks en pantalla o el almacenamiento de variables, hasta la grabación de scripts para su futura edición (en el caso de que el usuario tenga conocimientos técnicos de programación).

También podemos encontrar el Selenium Client API, que es el componente que nos permite la compatibilidad con la gran cantidad de lenguajes de programación mencionados recientemente. Esta interfaz de programación de aplicaciones (API) permite poder llamar a los métodos de Selenium Client en los script personalizados que desees crear,

Otro de los componentes más útiles que posee este framework es el Selenium WebDriver. Es un servidor que acepta comandos al navegador vía HTTP de forma local o en remoto, siendo el que permite la conectividad con los navegadores. Es el sucesor de Selenium Remote Control, componente que ya ha quedado desfasado. Este mejora al anterior en que no precisa de un servidor especial para ser ejecutado, sino que cada vez que es llamado WebDriver crea una instancia del navegador y lo controla, lo que implica un gran ahorro de recursos.

Por último, tenemos el Selenium Grid, componente que permite crear las instancias del navegador en máquinas remotas con la finalidad de ejecutar pruebas en paralelo en múltiples máquinas y manejar diferentes versiones y configuraciones de manera centralizada (Concentrador), reduciendo ampliamente el tiempo que tarda un paquete de pruebas en completarse.

En conclusión, vemos como es un framework muy completo, que a pesar de ser muy complejo y tener una curva de aprendizaje muy alta, ofrece una serie de posibilidades de uso muy interesantes.

2.4.3. Herramienta seleccionada: Selenium

Después de analizar las posibles herramientas que nos ofrece el mercado, concluimos en que la más completa, por tanto, la que emplearemos para el desarrollo del proyecto, es el framework Selenium.

Dos de las características principales que nos han hecho decantarnos por este framework son su **versatilidad** a la hora de crear soluciones, ya sea mediante código gracias a sus scripts personalizables, o mediante el IDE (interfaz gráfica) que posee, y su **capacidad de integración** con una gran cantidad de lenguajes de programación y con los principales navegadores y sistema operativos.

En definitiva, tenemos grandes esperanzas con las oportunidades que nos ofrece este framework con la finalidad de desarrollar un proyecto competitivo que cumpla con las expectativas deseadas, ahorrando tiempo y dinero a largo plazo.

3. Desarrollo de la propuesta

A continuación, se desarrolla la propuesta del proyecto, analizando los requisitos y viendo los diseños, tanto a nivel de arquitectura como de interfaz. Para finalizar el capítulo, se expone detalladamente la fase de implementación del mismo.

3.1. Análisis de los requisitos

Uno de los principales objetivos propuestos por la empresa colaboradora Aura Quantic es el de desarrollar una herramienta que pueda ser una alternativa a software de pago, como Power Automate de Microsoft. Esto con el fin de ofrecer una potente herramienta propia de la empresa y así ahorrar costes tanto a ellos como a futuros clientes.

Como es lógico, este proyecto no pretende ser un calco de Power Automate, ya que no se dispone ni del tiempo ni del conocimiento para poder replicar todas las funcionalidades que este ofrece. Solo interesa poder replicar la funcionalidad de automatizar procesos realizados a través del navegador, como por ejemplo una empresa de seguros que tenga que rellenar un formulario mediante la información recibida por correo. En este caso de ejemplo, la información requerida siempre va a estar dispuesta de la misma forma en el correo, por tanto se sabe qué campos se debe extraer de este, y el formulario también es siempre igual, por tanto se sabe qué información poner en cada campo.

Otro de los requisitos propuestos por la empresa es la de poder alojar el servidor en una máquina para que su servicio esté siempre disponible para su uso por cualquier cliente. Además de esto, a la empresa le gustaría que fuese posible la integración de este servicio mediante las aplicaciones low code o de bajo código que ellos mismos desarrollan y ofrecen.

Una vez analizado los requisitos propuestos por la empresa, y realizado el estudio del mercado y el marco teórico de los software de automatización robótica de procesos, se elige la herramienta Selenium y se empieza con la fase de diseño del proyecto.

3.2. Diseño

Una vez analizados los requisitos principales del proyecto, se pasa a definir la estructura del mismo, diseñando la arquitectura que tiene la lógica de la aplicación y presentando los bocetos o mockups seguidos para el desarrollo de la interfaz gráfica.

3.2.1. Arquitectura de la aplicación

Se opta por la tradicional arquitectura de cliente-servidor, con el fin de establecer una capa de conexión o API propia. Como ha sido mencionado previamente, el cliente será constituido por las aplicaciones de bajo código que pueden ser desarrolladas mediante la herramienta Aura Quantic, y el servidor es desarrollado por nosotros mismos.

Se establece la tecnología de servidor Express, framework web alojado dentro del entorno de ejecución de NodeJS, que es un entorno de programación diseñado para el desarrollo de aplicaciones o servidores web. Estas tecnologías son las seleccionadas debido a la gran escalabilidad y simpleza que proporcionan a la hora de desarrollar un servidor web.

Por tanto, la arquitectura de la aplicación queda de la siguiente manera:

- Un archivo que controla la lógica de la aplicación, en el que se encuentran los métodos a utilizar.
- Un archivo que arranca el servidor web.
- Un archivo que alberga las peticiones o endpoints a los métodos definidos en la lógica.

Por último, el lenguaje elegido para el desarrollo completo es JavaScript, ya que es el lenguaje en el que trabaja NodeJS y Express. A esto, cabe destacar que se crea un archivo que sirve como API propia para llamar a los métodos del propio Selenium dentro de nuestra lógica del proyecto.

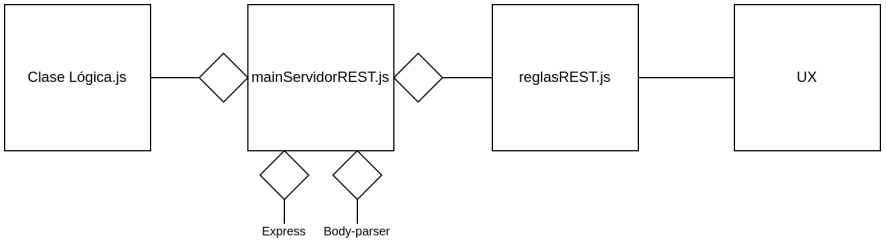


Imagen 4. Diagrama de arquitectura.

3.2.2. Diagramas de la interfaz de usuario

Una vez establecido el diseño de la lógica de la aplicación toca diseñar la interfaz de usuario que queremos que vea el cliente final. En nuestro caso solo desarrollamos versión web, no móvil, ajustándose a lo que pide la empresa colaboradora.

Esta interfaz es cambiante según lo que demande el cliente, además de ser ellos mismos capaces de desarrollarla gracias a las herramientas para aplicaciones de bajo código que ofrece Aura Quantic. Aquí mostramos dos ejemplos de lo que se puede llegar a conseguir.

Como podemos ver ambos ejemplos constan de un formulario simple en el que especificamos los datos requeridos para buscar la información deseada, y una vez hecha la búsqueda, se completa otro formulario simple con la información obtenida.

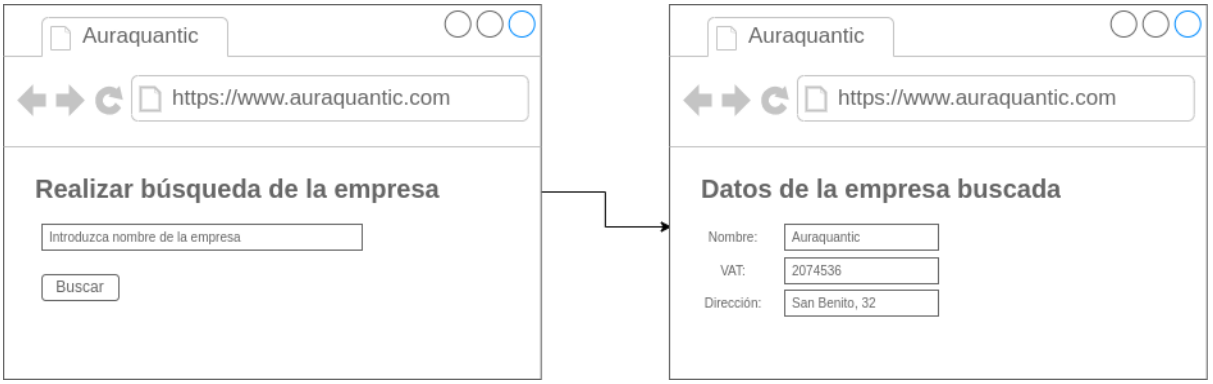


Imagen 5. Diagramas de la interfaz de usuario.

3.3. Implementación

Para finalizar el capítulo, se detalla el proceso entero del desarrollo del proyecto desde el punto de vista técnico.

3.3.1. Backend

En primer lugar tenemos la implementación del backend o lógica del proyecto, en la que destacan por un lado, el servidor hecho en Express y la capa de conexión o API realizada en NodeJS con la que establecer conexión entre la lógica y la interfaz, y por otro lado las clases API_Selenium, para poder usar los métodos de Selenium en nuestra aplicación, y Lógica, con la cual definir los métodos propios necesarios para satisfacer los pedidos de los clientes, implementadas en el lenguaje de programación JavaScript.

3.3.1.1. Clase con métodos para usar Selenium

En esta clase se definen unos métodos para llamar a la API de selenium que podrán ser usados en la clase Lógica. Esta clase nace con el fin de evitar repetir código, ya que en esta clase se definen unos métodos que se pueden repetir muchas veces a la hora de crear funcionalidades.

Lo primero que debemos hacer es importar el requirement o requisito de Selenium WebDriver para poder usarlo en el archivo. A continuación, definimos una variable global denominada driver en la que construimos el objeto necesario para usar Selenium Webdriver. Esta variable la tenemos que usar en cada método si queremos usar el componente WebDriver previamente construído.

```
//Cargamos el selenium-webdriver
const webdriver = require('selenium-webdriver'),
  By = webdriver.By,
  until = webdriver.until;

const driver = new webdriver.Builder().forBrowser('chrome').build();
```

Imagen 6. Dependencias de la clase API_Selenium.js

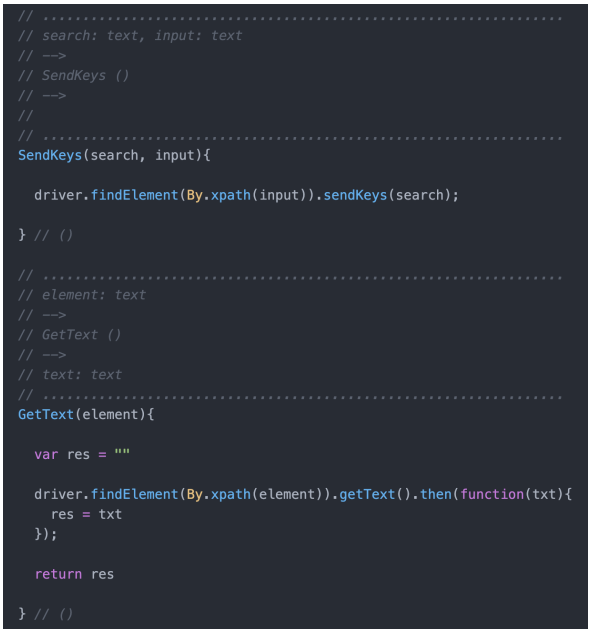
Seguidamente hay que exportar la clase entera para poder usarla en la clase Lógica. Y una vez hecho todo esto pasamos a definir los métodos que vamos a usar, dando como resultado los siguientes:

- **Follow()**: este método sirve para navegar a una url previamente pasada en los argumentos.
- **ClickButton()**: este método sirve para pulsar un botón. Hay que pasarle el identificador del botón en los argumentos.
- **SendKeys()**: este método sirve para rellenar formularios. Hay que pasarle el identificador del formulario y el texto a rellenar en los argumentos.
- **GetText()**: este método sirve para obtener el texto de un elemento de la web. Hay que pasarle el identificador del elemento en los argumentos.
- **OpenNewTab()**: este método sirve para abrir una nueva pestaña en el navegador.
- **ScrollToElement()**: este método sirve para hacer scroll a un elemento de la web. Hay que pasarle el identificador del elemento en los argumentos.
- **ScrollToElementByPixels()**: este es similar al anterior pero además baja el número de píxeles especificados en los argumentos.
- **AssertIsDisplayed()**: este método sirve para comprobar si un elemento aparece en la web. Para ellos hay que pasarle el identificador del elemento en los argumentos.

```
// ....................................................................
// URL: text
// -->
// Follow ()
// -->
//
// ....................................................................
Follow(page){

  driver.get(page)

} // ()

// ....................................................................
// button: text
// -->
// ClickButton ()
// -->
//
// ....................................................................
ClickButton(button){

  driver.findElement(By.xpath(button)).click();

} // ()
```

```
// ....................................................................
//
// -->
// OpenNewTab ()
// -->
//
// ....................................................................
OpenNewTab(){

  driver.switchTo().newWindow('tab');

}

// ....................................................................
// element: Text
// -->
// ScrollToElement ()
// -->
//
// ....................................................................
ScrollToElement(element){

  driver.execute_script("return " + element + ".scrollIntoView(true);");

}
```

```
// ....................................................................
// element: Text, pixels: Text
// -->
// ScrollToElementByPixels ()
// -->
//
// ....................................................................
ScrollToElementByPixels(element, pixels){

  driver.execute_script("return " + element + ".scrollIntoView(true);");
  driver.execute_script("window.scrollBy(0, )" + pixels + ");")

}

// ....................................................................
// element: Text
// -->
// AssertIsDisplayed ()
// -->
//
// ....................................................................
AssertIsDisplayed(element){

  is_element_displayed = driver.findElement(By.xpath(element)).isEnabled();
  if (!is_element_displayed){
    console.log("Element is not displayed")
  }
  console.log("Element is displayed")

}
```



```
// ....................................................................
// search: text, input: text
// -->
// SendKeys ()
// -->
//
// ....................................................................
SendKeys(search, input){

  driver.findElement(By.xpath(input)).sendKeys(search);

} // ()

// ....................................................................
// element: text
// -->
// GetText ()
// -->
// text: text
// ....................................................................
GetText(element){

  var res = ""

  driver.findElement(By.xpath(element)).getText().then(function(txt){
    res = txt
  });

  return res

} // ()
```

Imágenes 7, 8, 9 y 10. Métodos de la clase API_Selenium.js

En nuestro caso, para los identificadores mencionados anteriormente usamos los xpath, que es un código que construimos nosotros mismos para localizar elementos en la estructura de la web o DOM, debido a que nos proporcionan una mayor polivalencia porque no todos los elementos poseen el atributo id. Para construirlos, hay que seguir la siguiente estructura de ejemplo: **//*[@class="enter"]**. Esta estructura siempre sigue el mismo patrón, empieza por 2 contrabarras (//) seguido del tipo de elemento que queremos buscar (para los tipos tenemos que usar etiquetas de HTML), en este ejemplo usamos un asterisco (*) que indica que busque todos los tipos de elementos que cumplan la condición sin importar el propio tipo. A continuación, especificamos la condición entre llaves cuadradas ([ ]), siendo esta condición que el atributo class del elemento se llame enter.

Por el momento, estos son los métodos propuestos, aunque siempre se pueden ampliar en el futuro en caso de necesidad, ya que la herramienta Selenium proporciona un sin fin de oportunidades y opciones.

3.3.1.2. Servidor y capa de conexión (API)

Una vez definida la clase anterior, creamos otra clase denominada Lógica en la cual creamos los procesos demandados por los clientes mediante los métodos definidos en la clase de API_Selenium. En primer lugar, importamos la clase anteriormente mencionada y exportamos la clase entera para poder usarla en el Servidor.

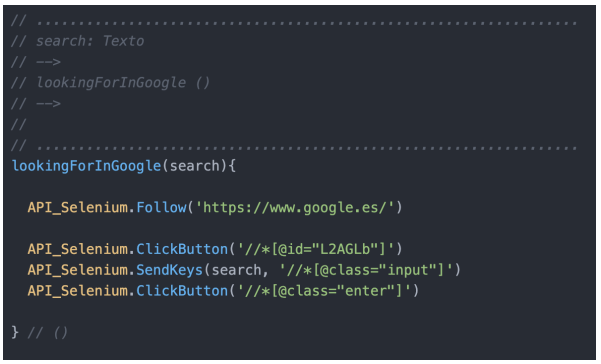
```
// ..........................................................................
// Logica.js
// ..........................................................................
const API_Selenium = require("./API_Selenium.js")
```

Imagen 11. Dependencias de la clase Logica.js

A continuación, a falta de pedidos de posibles clientes, definimos unos métodos de ejemplo de lo que se puede lograr con esta herramienta, siendo estos los siguientes:

- **goToWebPage()**: navega hasta la url especificada en los argumentos.
- **lookingForInGoogle()**: busca un texto pasado en los argumentos en google.
- **lookinfForInfoCompany()**: busca en la web *companyweb.be/en* la empresa que le pasamos en los argumentos y nos devuelve su número de identificación fiscal (VAT) y su dirección.
- **ranking()**: navega a la web *ranking-empresas.eleconomista.es,* obtiene la primera empresa en el ranking y en otra pestaña navega a la web del método anterior y busca la información de dicha empresa, devolviéndonos los mismos datos que antes.

```
// ....................................................................
//
// -->
// openAuraQuanticWeb ()
// -->
//
// ....................................................................
openAuraQuanticWeb(){

  API_Selenium.Follow('https://www.auraquantic.com/')

} // ()

// ....................................................................
// URL: Texto
// -->
// goToWebPage ()
// -->
//
// ....................................................................
goToWebPage(page){

  API_Selenium.Follow(page)

} // ()
```



```
// ....................................................................
// search: Texto
// -->
// lookingForInGoogle ()
// -->
//
// ....................................................................
lookingForInGoogle(search){

  API_Selenium.Follow('https://www.google.es/')

  API_Selenium.ClickButton('//*[@id="L2AGLb"]')
  API_Selenium.SendKeys(search, '//*[@class="input"]')
  API_Selenium.ClickButton('//*[@class="enter"]')

} // ()
```

```
// ....................................................................
// company: Text
// -->
// lookingForInfoCompany ()
// -->
// json { VAT: Text, address: Text}
// ....................................................................
lookingForInfoCompany(company){

  //Asignamos valor vacío a las variables de respuesta
  var vat = ""
  var address = ""

  API_Selenium.Follow('https://www.companyweb.be/en')
  API_Selenium.ClickButton('//*[@class=".btn-lg"]')
  API_Selenium.SendKeys(company, '//*[@class=".search-input"]')
  API_Selenium.ClickButton('//*[@id="freeSearch"]')
  API_Selenium.ClickButton('//*[@class=".btn-lg"]')

  //Asignamos valor deseado a las variables de respuesta
  vat = API_Selenium.GetText('//*[@id="company-description"]')
  address = API_Selenium.GetText('//*[@id="address"]')

  //Creamos estructura de respuesta con las variables deseadas
  var res = {
    vat: vat,
    direccion: address
  }

  return res

} // ()
```

```
// ....................................................................
//
// -->
// ranking ()
// -->
// vat: text
// ....................................................................
ranking(){

  var company = ""
  var vat = ""

  API_Selenium.Follow("https://ranking-empresas.eleconomista.es/")
  API_Selenium.ClickButton('//*[@class="notice-agree-button"]')
  API_Selenium.ClickButton(".tr_hover_even:nth-child(1) > .tal")
  company = API_Selenium.GetText('//*[@class="tit1"]')

  API_Selenium.OpenNewTab()
  API_Selenium.Follow('https://www.companyweb.be/en')
  API_Selenium.ClickButton('//*[@class=".btn-lg"]')
  API_Selenium.SendKeys(company, '//*[@class=".search-input"]')
  API_Selenium.ClickButton('//*[@id="freeSearch"]')
  API_Selenium.ClickButton('//*[@class=".btn-lg"]')

  //Asignamos valor deseadoa las variables de respuesta
  vat = API_Selenium.GetText('//*[@id="company-description"]')

  //Creamos estructura de respuesta con las variables deseadas
  var res = {
    nombre: empresa,
    vat: vat
  }

  return res

} // ()
```

Imágenes 12, 13, 14 y 15. Métodos de la clase Logica.js

Ahora pasamos a definir el archivo ReglasRest, el cual establece una capa de conexión o API Rest, y contiene las peticiones o endpoints de los métodos de la clase Lógica para poder ser llamados desde la interfaz de usuario mediante conectores, siempre y cuando, el servidor tenga el servicio activo. Estas llamadas son realizadas a través del protocolo HTTP, siendo usados solo las peticiones GET y POST.

Para el servidor web, como hemos comentado, usamos Express y Node JS. Para ello, primero debemos instalar Node y su package mediante el comando de shell **npm install**. Posteriormente, definimos un código sencillo en JavaScript en el que importamos las dependencias necesarias, entre ellas la clase lógica, y

definimos una función con la cual cargar esta clase. Por último, en el main() cargamos la clase lógica mediante la función que acabamos de mencionar, después creamos el servidor Express, cargamos el archivo ReglasRest, y finalmente lanzamos el servicio, mediante el comando de shell **npm run servidor**, que queda escuchando en el puerto 8080.

```
// .....................................................
// POST /abrirWeb
// .....................................................
servidorExpress.post('/abrirWeb/', async function(peticion, respuesta) {
  console.log(" * POST /abrirWeb ")
  var datos = JSON.parse(peticion.body)
  await laLogica.goToWebPage(datos.pagina)
  respuesta.send("¡Abierta la web con éxito!")
}) // post /abrirWeb

// .....................................................
// POST /buscar
// .....................................................
servidorExpress.post('/buscar/', async function(peticion, respuesta) {
  console.log(" * POST /buscar ")
  var datos = JSON.parse(peticion.body)
  await laLogica.lookingForInGoogle(datos.busqueda)
  respuesta.send("¡Búsqueda realizada con éxito!")
}) // post /buscar
```

```
// .....................................................
// POST /infoEmpresa
// .....................................................
servidorExpress.post('/infoEmpresa/', async function(peticion, respuesta) {
  console.log(" * POST /infoEmpresa ")
  var datos = JSON.parse(peticion.body);
  var res = await laLogica.lookingForInfoCompany(datos.empresa);
  respuesta.send(JSON.stringify(res));
}) // post /infoEmpresa

// .....................................................
// POST /ranking
// .....................................................
servidorExpress.post('/ranking/', async function(peticion, respuesta) {
  console.log(" * POST /ranking ")
  var res = await laLogica.ranking();
  respuesta.send(JSON.stringify(res));
}) // post /ranking
```

Imágenes 16 y 17. Peticiones o endpoints del archivo Reglas REST.js

Por último, una vez tenemos el servidor funcionando en local, buscamos alojarlo en algún servicio que nos permita tenerlo conectado a todas horas. Aquí barajamos diversas opciones, como hosts de pago, máquinas virtuales, contenedores Docker, etc. Finalmente, consensuado con el tutor de la empresa, se opta por alojar el servidor en las máquinas que tiene la propia empresa siempre activas. Por tanto, se despliega el backend del proyecto en dicha máquina, se vuelve a instalar los paquetes de NodeJS y se ejecuta el servidor para que el servicio quede escuchando continuamente.

3.3.2. Frontend. Aplicación low code

Referente al frontend, se desarrollan los formularios necesarios a través de las herramientas de desarrollo de aplicaciones de bajo código o low code que proporciona AuraQuantic. Estas herramientas consisten en un entorno de creación *drag and drop* o de arrastrar elementos al lienzo de edición. Además contiene diferentes configuraciones.

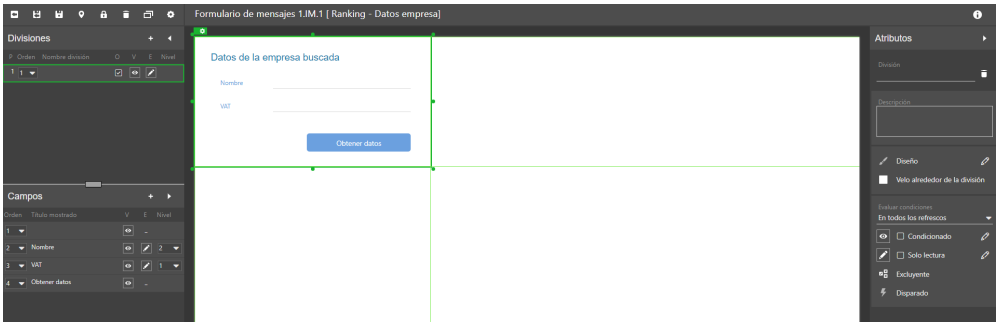


Imagen 18. Entorno de desarrollo de aplicaciones low code de Auraquantic.

Para nuestro proyecto, en primer lugar creamos un proceso en el cual añadimos un evento de mensaje inicial, otro de mensaje final y uno de final de proceso.

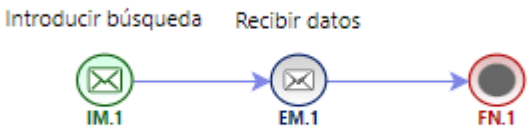


Imagen 19. Diagrama del proceso.

En el panel del evento añadimos las variables necesarias para crear los campos del formulario.

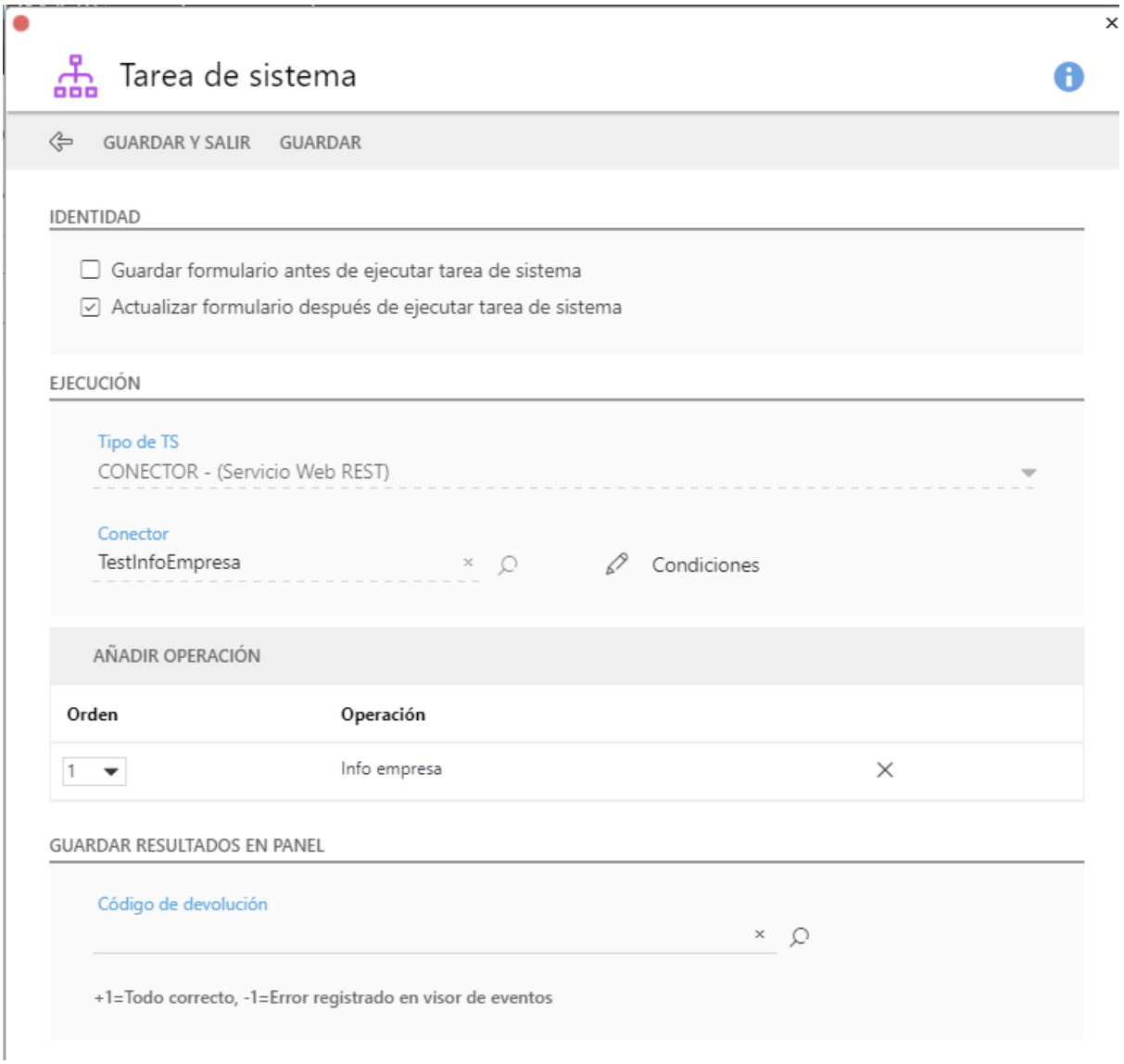


Imagen 20. Tarea del sistema del proceso en el creador de aplicaciones low code.

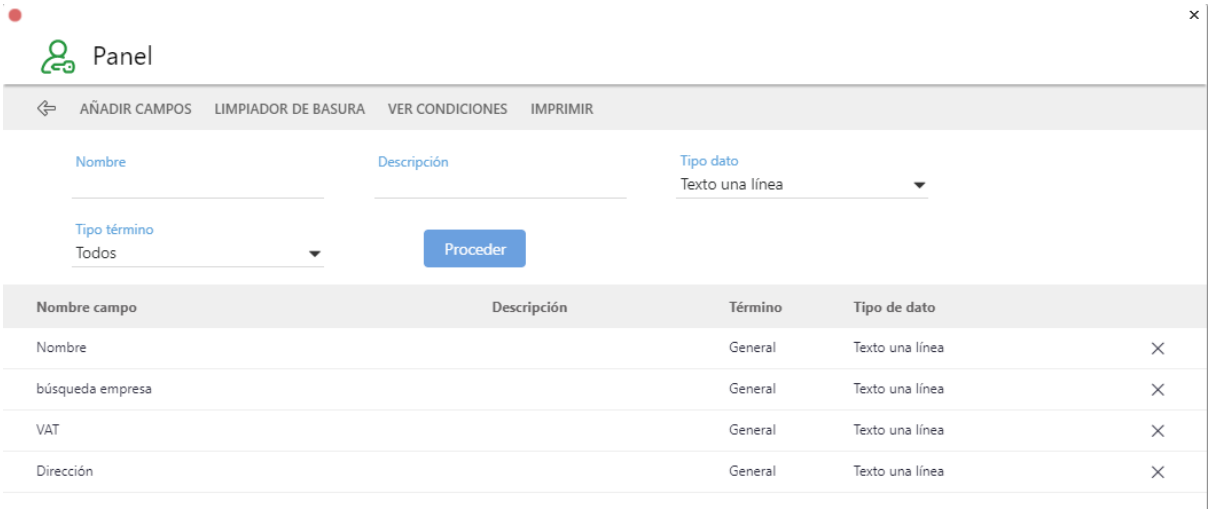
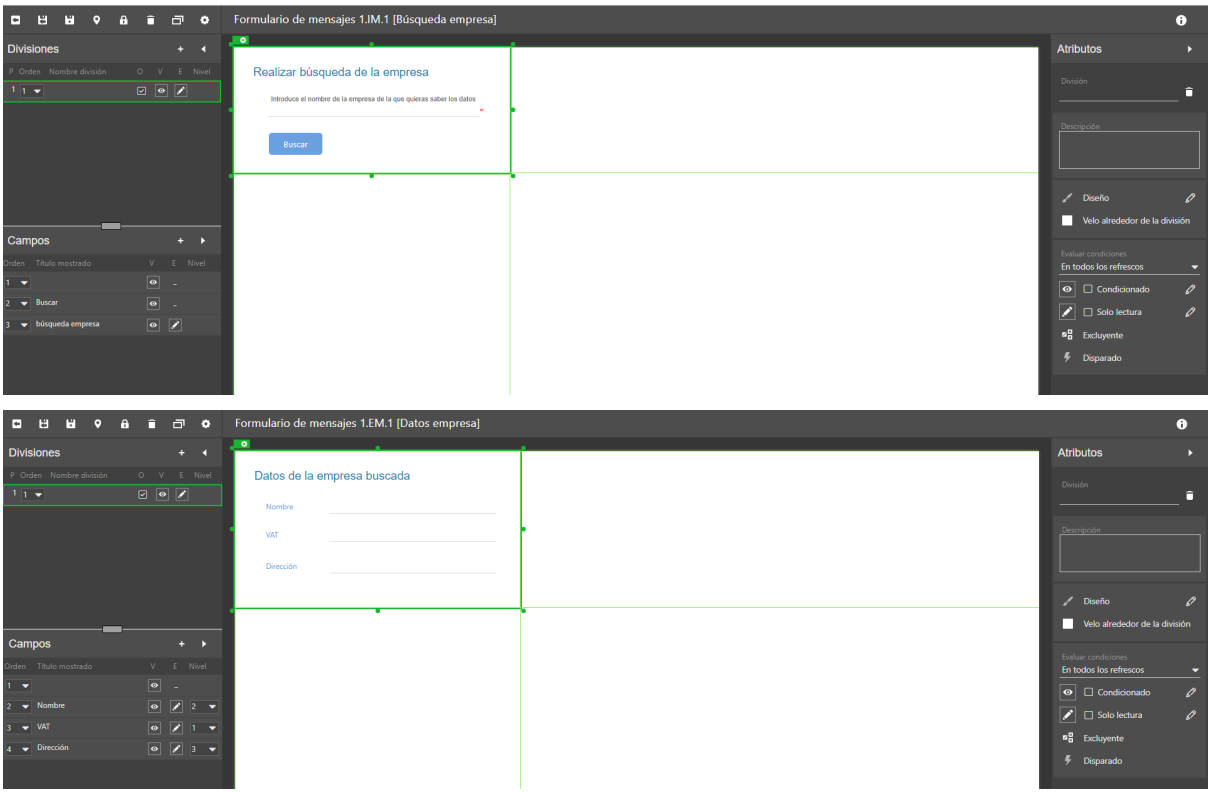


Imagen 21. Panel del proceso del creador de aplicaciones low code.

Por último, montamos el formulario mediante el entorno de creación mencionado previamente.



Imágenes 22 y 23. Formularios para el proceso creado.

3.3.3. Conectores. Integración backend y frontend

Para la integración de la Lógica con las aplicaciones low code de AuraQuantic, la misma empresa ofrece unas herramientas conocidas como conectores, con las que se puede realizar esta funcionalidad.

Para ello, hay que crear un conector en el apartado correspondiente de las herramientas de AuraQuantic, configurarlo especificando un nombre y una url, y por último, definir el tipo de petición (GET o POST), las cabeceras que requiera, y los apartados *request* y *response*.

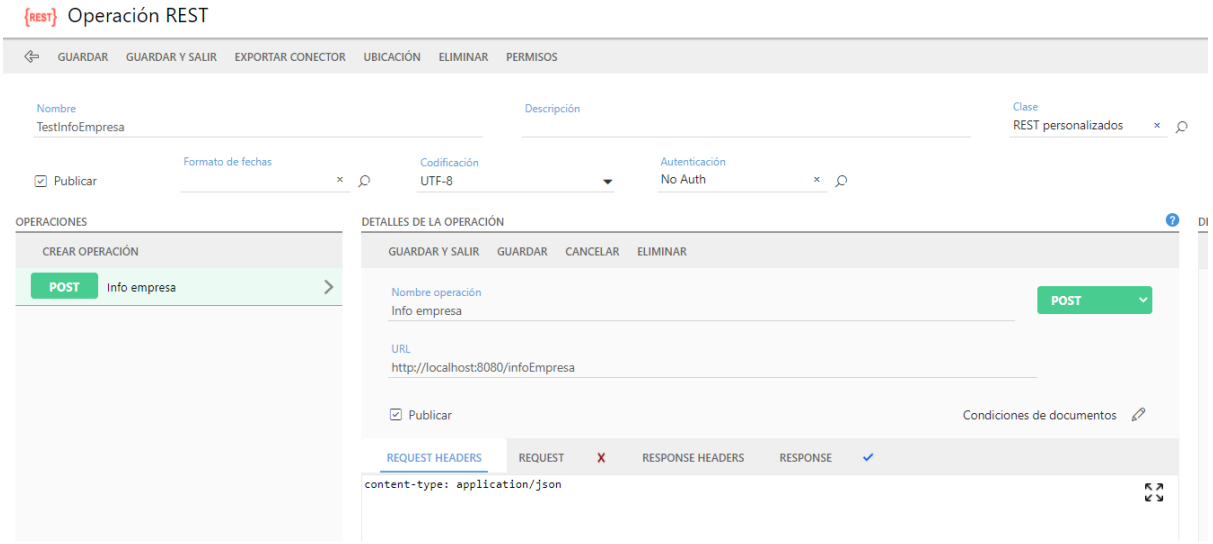
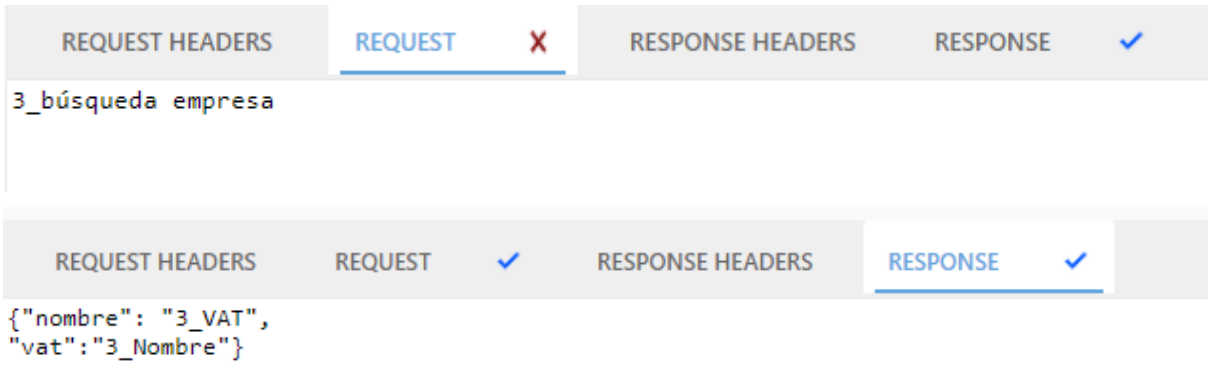


Imagen 24. Conector POST en la herramienta AuraQuantic.

Dichos apartados son muy importantes, ya que en el apartado de request se especifica el campo del formulario del cual la petición extraerá la información a enviar, mientras que en el apartado response se define el resultado esperado, con estructura de JSON, añadiendo los campos del formulario que queremos rellenar.



Imágenes 25 y 26. Apartados request y response del conector.

Por tanto, una vez configurado el conector y realizada una petición, el método de la clase Lógica se procesa obteniendo así la información referente a la empresa con la que se rellena el formulario.

3.3.4. Distribución. Marketplace o tienda de aplicaciones

En este punto, el proyecto ya cumple todos los requisitos que se habían definido previamente, siendo capaz de automatizar procesos web en el navegador mediante la herramienta WebDriver de Selenium. Pero una parte muy importante de un proyecto es la distribución del mismo con el fin de comercializarlo y darle rentabilidad al desarrollo. Por ello, se plantea la creación de una tienda de aplicaciones o MarketPlace asociada a la empresa AuraQuantic y alojada en los dominios web de esta. Este MarketPlace tiene como función la de poder distribuir diversas herramientas, procesos, conectores, y demás software desarrollado por la empresa, entre ellos incluído este proyecto. En un inicio, el acceso a esta tienda es exclusivo de los clientes de AuraQuantic, siendo todo el contenido de ella gratuito, permitiendo a un cliente poder descargar todo aquello que necesite. En un futuro, se estudiará la posibilidad de abrir el acceso al MarketPlace a cualquier usuario y fijar un precio a los servicios ofrecidos en el mismo.

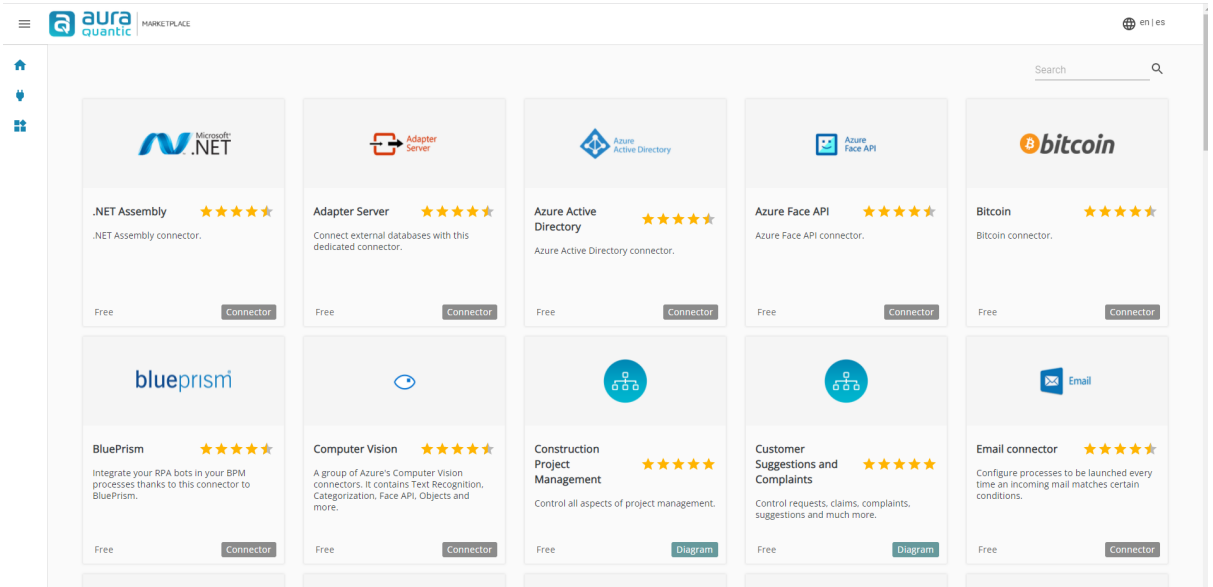
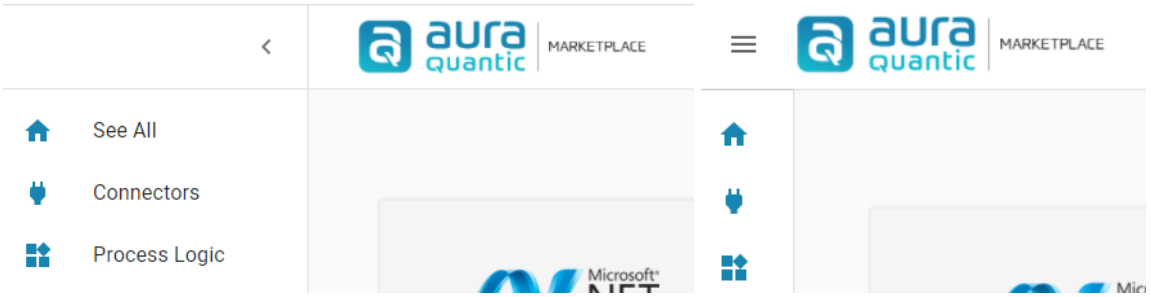


Imagen 27. Marketplace o tienda de aplicaciones.

Para su desarrollo, se opta por el framework de JavaScript, Vue JS, por los siguientes motivos:

- Su curva de aprendizaje es de las más sencillas en cuanto a frameworks de JavaScript, lo cual se adapta a la perfección al perfil que presentamos.
- Usa HTML, CSS y JavaScript, que son lenguajes que ya conocemos.
- Vue JS ofrece un gran rendimiento de las aplicaciones desarrolladas.

Vue JS se caracteriza por el uso de componentes web, por tanto, lo primero es crear un componente para el menú, el cual es un menú lateral formado por iconos con enlaces a la página home, uno a los conectores, que te lleva a la página de productos con un filtro aplicado para que solo se muestren los conectores, y uno que te lleva a la misma página pero filtrando por procesos. El componente del menú tiene un menú de hamburguesa con la funcionalidad de descolapsar el menú, haciendo que aparezcan los nombres al lado de los iconos.



Imágenes 28 y 29. Componente del menú colapsado y desplegado.

A continuación, se define la página de productos, en la cual aparecen todos los productos existentes en la tienda separados por tarjetas. En cada tarjeta aparece un conector con su nombre, su icono y su valoración. Estas tarjetas son clickables, abriendo así la ficha del producto, la cual tiene toda la información del software en cuestión, además de un botón de descargar, el cual descarga el o los ficheros que contiene el producto a través del navegador, como podemos observar en las imágenes.

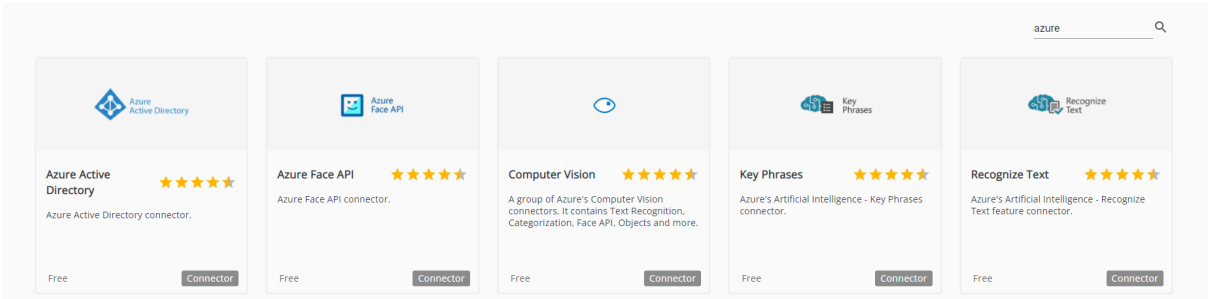
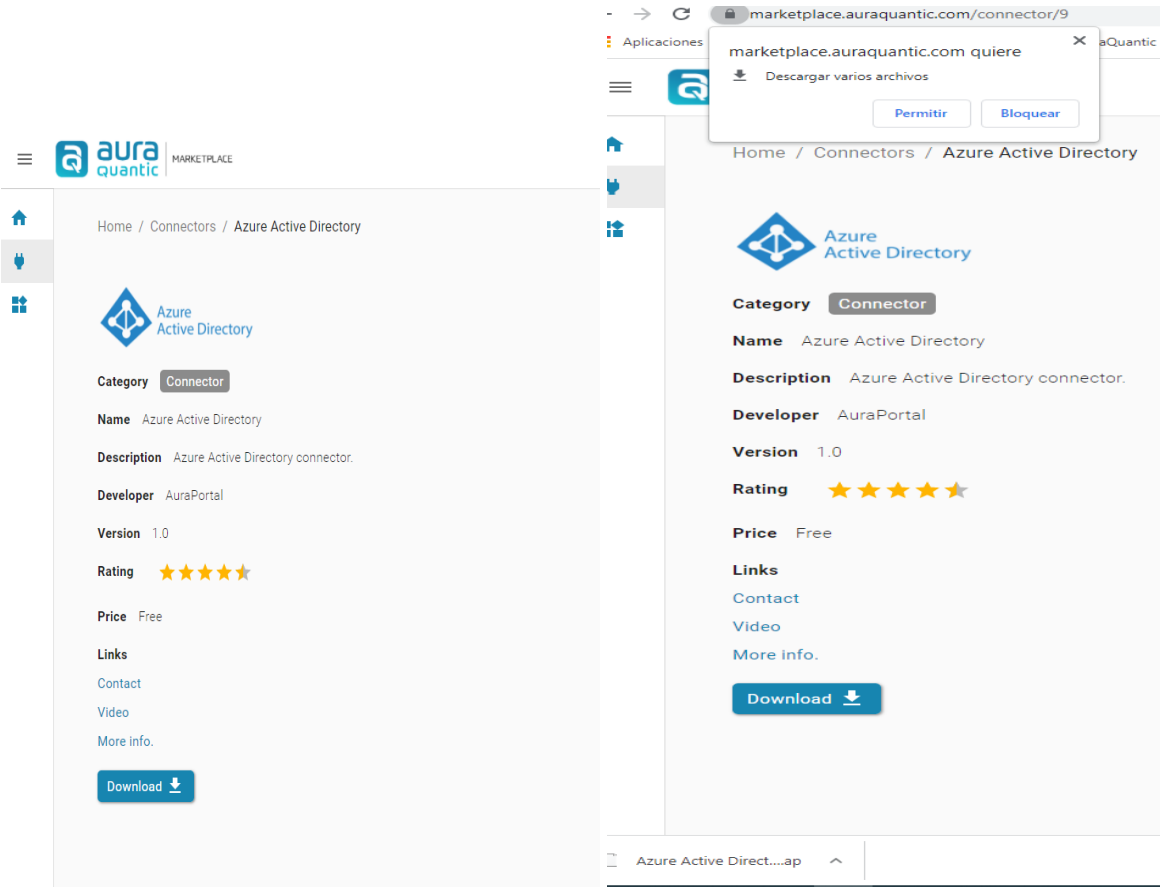


Imagen 30. Buscador en el MarketPlace.



Imágenes 31 y 32. Ficha de producto del MarketPlace y funcionalidad de descargar.

4. Conclusiones

Una vez realizado el proyecto, podemos concluir que se han conseguido los hitos definidos al principio del desarrollo. Se ha estudiado el uso de los RPA en la industria a día de hoy y su evolución a lo largo de los últimos años. Se ha indagado en diferentes herramientas RPA, realizando un estudio intensivo al software Selenium, contribuyendo a su desarrollo y entrando en la gran comunidad que esta tecnología posee.

A nivel de producto, se ha alcanzado una fase beta bastante satisfactoria, la cual la empresa valora ofrecer a sus clientes e integrarla en sus soluciones. En este punto, cobra gran importancia el mantenimiento y soporte que se le dé a la herramienta, puesto que el desarrollo no ha terminado, está en constante crecimiento con tal de satisfacer las peticiones de los clientes.

A nivel personal, el desarrollo de este proyecto ha supuesto una enriquecedora experiencia, entrando de lleno en el campo de la investigación y del desarrollo de nuevas tecnologías emergentes. Además de haber aprendido mucho sobre el software Selenium y su potencial.

Unos meses después de haber terminado el proyecto y haberse desvinculado de la empresa colaboradora, además de haber entrado al mundo laboral, viendo con retrospectiva el desarrollo del proyecto, se puede afirmar que, aunque muy interesante y enriquecedor, el proyecto no aprovecha del todo el potencial que ofrece el software Selenium.

Como posibles mejoras del proyecto, se podría aprovechar el software de selenium para implementar tests de calidad (QA) además de la automatización de ciertos procesos web. Aprovechando la automatización que ofrece, se podría definir unos métodos para comprobar que las funcionalidades de una web o negocio e-commerce funcionan como se espera, como por ejemplo el registro o un proceso de compra. Otra mejora sería abandonar las aplicaciones de bajo código o low code y desarrollar un frontend con Vue JS en el cual representar los resultados de los tests de calidad, informando de sí la funcionalidad testeada funciona correctamente o, en su defecto, tiene algún error y en qué punto de la traza se encuentra, para así poder detectarlo rápidamente para su solución.

De esta manera, estaríamos dando un valor añadido a nuestra aplicación, aumentando el público objetivo de la misma.

5. Referencias y bibliografía

1. Wil M. P. van der Aalst · Martin Bichler · Armin Heinzl - “Robotic Process Automation” (14 May 2018) https://link.springer.com/content/pdf/10.1007/s12599-018-0542-4.pdf
2. Professor Leslie Willcocks · Professor Mary Lacity · Andrew Craig - “The IT Function and Robotic Process Automation” (October 2015) http://eprints.lse.ac.uk/64519/1/OUWRPS_15_05_published.pdf
3. Aleksandre Asatiani · Esko Penttinen - “Turning robotic process automation into commercial success – Case OpusCapita” (May 2016) https://www.researchgate.net/publication/303460189_Turning_robotic_process_automation_into_commercial_success_-_Case_OpusCapita
4. Ning Zhang · Bo Liu - “Alignment of business in robotic process automation” https://www.emerald.com/insight/content/doi/10.1108/IJCS-09-2018-0018/full/pdf
5. Leslie Willcocks · Mary Lacity · Andrew Craig - “Robotic process automation: strategic transformation lever for global business services?” https://link.springer.com/article/10.1057/s41266-016-0016-9

   http://eprints.lse.ac.uk/71146/1/Willcocks_Robotic%20process%20automation_author_2017%20LSERO.pdf
6. Leslie Willcocks · Mary Lacity · Andrew Craig - “Robotic Process Automation at Telefónica O2” http://eprints.lse.ac.uk/64516/1/OUWRPS_15_02_published.pdf
7. Somayya Madakam · Rajesh M. Holmukhe · Durgesh Kumar Jaiswal - “THE FUTURE DIGITAL WORK FORCE: ROBOTIC PROCESS AUTOMATION (RPA)” https://www.scielo.br/j/jistm/a/m7cqFWJPsWSk8ZnWRN6fR5m/?format=pdf&lang=en
8. Aleksandre Asatiani · Esko Penttinen · Henje Kasslin - “How to Choose between Robotic Process Automation and Back-End System Automation?” (June 2018) https://www.researchgate.net/publication/324918928_How_to_Choose_between_Robotic_Process_Automation_and_Back-End_System_Automation
9. Taulli, Tom, author. | Apress | 1st edition | 2020 - “Robotic Process Automation: An Introduction to the RPA Market, Implementation, and How It Can Impact Your Company” https://polibuscador.upv.es/discovery/fulldisplay?docid=alma9970907587037006&context=L&vid=34UPV_INST:bibupv&lang=es&search_scope=MyInst_and_CI&adaptor=Local%20Search%20Engine&tab=BUS_GENERAL&query=any,contains,rpa&offset=25

10. Bryan Andrade Andrade - “El RPA: ¿De dónde viene, para qué sirve y cómo empezar?” https://revistaempresarial.com/tecnologia/el-rpa-de-donde-viene-para-que-sirve-y-como-empezar/
11. Selenium - “Documentación oficial de Selenium” https://www.selenium.dev/documentation/
12. Selenium - “Documentación oficial de Selenium” https://www.selenium.dev/selenium-ide/docs/en/introduction/getting-started
13. Auraquantic - “Cursos y documentación AuraQuantic” https://www.auraquantic.com/
14. Microsoft - “Documentación oficial de Power Automate” https://docs.microsoft.com/en-us/power-automate/