Universidad de Alcalá
Escuela Politécnica Superior

Grado en Ingeniería Telemática

Trabajo Fin de Grado

Evolución de soluciones Low-code: Estudio

Autor: Marius Florin Lungu

Tutor: Antonio Moratilla Ocaña

2023

UNIVERSIDAD DE ALCALÁ

Escuela Politécnica Superior

Grado en Ingeniería Telemática

Trabajo Fin de Grado

Evolución de soluciones Low-code: Estudio

Autor: Marius Florin Lungu

Tutor/es: Antonio Moratilla Ocaña

TRIBUNAL:

Presidente: Dr. Eugenio Fernández Vicente

Vocal 1º: Ing. Ángel Javier Álvarez Miguel

Vocal 2º: Dr. Antonio Moratilla Ocaña

FECHA: 2023

"El éxito no es definitivo, el fracaso no es fatal:
lo que cuenta es el coraje para continuar."

Sir Winston Churchill

Agradecimientos

Quiero expresar mi más profundo agradecimiento a todas las personas que han sido parte de este importante y fundamental capítulo de mi vida.

En primer lugar, a mis padres, cuyo apoyo incondicional, amor y sacrificio han sido la base de mi éxito académico. Su constante aliento y confianza en mí me han impulsado a esforzarme al máximo, y han conseguido que en mis peores momentos consiga no rendirme.

A mi novia, por su comprensión, paciencia y apoyo emocional durante este arduo proceso. Tus palabras de ánimo y tu amor inquebrantable han sido mi refugio en los momentos más desafiantes.

A mis amigos, quienes han estado a mi lado en las buenas y las malas, brindándome su amistad incondicional y haciéndome recordar la importancia de la diversión y el equilibrio en la vida.

A mis amigos de la universidad, con quienes compartí innumerables horas de estudio, muchos logros y risas. Vuestra colaboración y camaradería hicieron que cada desafío académico fuera más llevadero y enriquecedor.

Por último, pero no menos importante, a mi compañero Carlos Moreno, a quien agradezco enormemente por su mentoría y apoyo a lo largo de los últimos años. Tus conocimientos, motivación y consejos han sido invaluables para mi desarrollo académico y profesional.

Este logro no habría sido posible sin el respaldo de cada uno de ustedes. Estoy profundamente agradecido por haber compartido este viaje con personas tan especiales.

¡Gracias por ser parte de este logro!

Resumen

Las soluciones Low-code han surgido como alternativas revolucionarias en el desarrollo de software, permitiendo crear aplicaciones funcionales sin conocimientos profundos de programación. No obstante, estas soluciones son poco conocidas todavía; no se sabe con exactitud el nivel que pueden alcanzar y si todo aquello que prometen se cumple. [1]

Este trabajo de fin de grado (TFG) tiene como principal objetivo explorar su evolución e impacto en el ciclo de desarrollo de software, comprendiendo su transformación y accesibilidad. Se investigarán sus ventajas, limitaciones y potencial para fomentar la agilidad y productividad en el desarrollo de software. Para ello se llevará a cabo un amplio estudio basado en revisión literaria y análisis empírico de casos reales. Se espera ampliar el conocimiento existente y proporcionar información valiosa para profesionales, líderes empresariales y académicos interesados en aprovechar estas soluciones.

Palabras clave: Low-Code, estudio, desarrollo de software, aplicaciones, evolución.

Abstract

Low-code solutions have emerged as revolutionary alternatives in software development, making it possible to create functional applications without in-depth programming knowledge. However, these solutions are still relatively unknown, and the extent to which they can deliver on their promises remains uncertain. [1]

The main objective of this Final Degree Project (TFG) is to explore the evolution and impact of Low-code solutions on the software development cycle, understanding their transformation and accessibility. We will investigate their advantages, limitations, and potential to promote agility and productivity in software development. To achieve this, an extensive study will be conducted, including literature review and empirical analysis of real-world cases. The aim is to expand existing knowledge and provide valuable insights for professionals, business leaders, and academics interested in leveraging these solutions.

Keywords: Low-Code, study, software development, applications, evolution.

Índice

1. Introducción

1.1 Motivación

En la última década, el desarrollo de software ha experimentado una evolución significativa debido a la creciente demanda de soluciones más rápidas y eficientes. En respuesta a este desafío, han surgido las soluciones Low-code, soluciones que prometen agilizar el proceso de desarrollo de aplicaciones al permitir a los usuarios crear software funcional sin requerir un conocimiento profundo de la programación tradicional.

Este enfoque revolucionario ha ganado popularidad en la industria, ofreciendo una alternativa a los métodos tradicionales de desarrollo de software. Los desarrolladores pueden acelerar el tiempo de desarrollo de aplicaciones, ya que gran parte del trabajo se realiza a través de interfaces gráficas intuitivas. Además, estas plataformas suelen ofrecer características como la generación automática de código, integración con sistemas existentes, gestión de bases de datos y colaboración en equipo. [2]

De esta manera, el crecimiento de las soluciones Low-code ha generado un interés creciente en investigar y comprender sus ventajas, limitaciones y su potencial para impulsar la agilidad y la productividad en el desarrollo de software.

En la actualidad toda investigación realizada sobre estas soluciones contribuye a ampliar el conocimiento existente sobre las soluciones Low-code, brindando información valiosa para profesionales de desarrollo de software, líderes empresariales y académicos interesados en aprovechar esta innovadora aproximación al desarrollo de aplicaciones. Con esta investigación, se busca proporcionar una base sólida para futuras decisiones y avances en el ámbito de las soluciones Low-code.

1.2 Objetivos

El objetivo de este Trabajo de Fin de Grado (TFG) es analizar la evolución de las soluciones Low-code y su impacto en el desarrollo de software. A medida que estas herramientas ganan terreno en el mercado, es esencial comprender cómo han evolucionado a lo largo del tiempo y cómo se han adaptado para abordar las necesidades cambiantes de los desarrolladores y las organizaciones. Además, es fundamental evaluar las ventajas y desventajas de las soluciones Low-code en comparación con los enfoques de desarrollo tradicionales, y analizar su eficacia en términos de tiempo, costo y calidad del software resultante.

Para lograr estos objetivos, se realizará una revisión exhaustiva de la literatura existente sobre el tema, abarcando estudios académicos, artículos científicos y casos de uso prácticos. Además, se llevará a cabo un análisis empírico utilizando encuestas tanto a desarrolladores y/o profesionales de la industria, como a personas con poco conocimiento sobre tecnología y este ámbito, con el fin de obtener información valiosa sobre las experiencias y percepciones en torno a las soluciones Low-code.

1.3 Estructura de la memoria

El presente trabajo está estructurado de la siguiente manera:

- En el **Capítulo 1** se introduce la memoria, exponiendo la motivación del trabajo, los objetivos y su estructuración.
- A continuación, se hará una explicación acerca de la metodología utilizada para llevar a cabo la investigación, incluyendo los detalles de la revisión bibliográfica y el enfoque de la investigación empírica, en el **Capítulo 2.**
- En el **Capítulo 3** se hará una comparación entre No-Code y Low-Code, y luego también se presentará una comparación con los modelos convencionales de programación.
- En el **Capítulo 4** se proporcionará un marco teórico que abordará los conceptos fundamentales relacionados con las soluciones Low-code y su evolución. Además, se mencionarán las diferentes ventajas y limitaciones de las soluciones Low-code, así como su impacto en el ciclo de desarrollo software.
- El **Capítulo 5** se centrará en los resultados obtenidos y su análisis, basándonos en el estudio de una encuesta realizada acerca del Low-code.
- En el **Capítulo 6** procederemos a investigar y exponer diferentes herramientas Low-code y No-Code, analizando diferentes aspectos como funciones, facilidad de uso, facilidad de obtención, costes, entre otras.
- A continuación, en el **Capítulo 7** escogiendo una opción entre las vistas en el capítulo 6, procederemos a profundizar en su estudio con mayor detalle, probando y observando diferentes aspectos de la misma (Diseño, Facilidad de uso, Herramientas, etc).
- Por último, en el **Capítulo 8** se presentarán las conclusiones finales del estudio, así como las posibles áreas de investigación futura.

En resumen, este TFG busca explorar en profundidad la evolución de las soluciones Low-code y proporcionar una visión objetiva de su impacto en el desarrollo.

2. Metodología de investigación

La metodología utilizada para llevar a cabo la investigación en este Trabajo de Fin de Grado (TFG) se basa en emplear dos enfoques principales, la investigación bibliográfica y la investigación empírica, acompañados de un estudio práctico de una herramienta Low-Code. Estos métodos se seleccionaron en base a los objetivos planteados y la naturaleza de la investigación, permitiendo abordar de manera exhaustiva el tema de estudio.

En primer lugar, se llevará a cabo una investigación bibliográfica detallada. Este enfoque implica la revisión y el análisis crítico de diversas fuentes bibliográficas, como libros, artículos científicos, informes técnicos y documentos académicos relevantes. Mediante esta investigación, se explorarán las teorías existentes, los avances previos y los enfoques utilizados en el ámbito de las soluciones Low-code. La investigación bibliográfica permitirá establecer un marco teórico sólido y sentar las bases conceptuales necesarias para el estudio.

En segundo lugar, se realizará una investigación empírica que involucrará la recopilación de datos primarios mediante técnicas como encuestas y entrevistas. A través de estos métodos, se recopilará información directa de profesionales del sector y expertos en desarrollo de software, pero también a personas inexpertas y otros actores relevantes en el ámbito de las soluciones Low-code. La investigación empírica permitirá obtener percepciones, experiencias y opiniones de primera mano sobre el uso, las ventajas y las limitaciones de estas soluciones, así como su impacto en el ciclo de desarrollo de software.

La combinación de la investigación empírica y la investigación bibliográfica proporcionará una perspectiva integral y rigurosa sobre el tema de estudio. Ambos enfoques se complementarán mutuamente, permitiendo una validación cruzada de los hallazgos y una comprensión más profunda de las ventajas, limitaciones y el potencial de las soluciones Low-code en el desarrollo de software.

Por último, procederemos a instalar, trastear y analizar una aplicación de Low-code, en este caso, Microsoft PowerApps. Comenzaremos explorando su entorno de desarrollo visual, arrastrando y soltando elementos para diseñar la interfaz y la lógica de la aplicación. Luego, evaluaremos su rendimiento y funcionalidad mediante pruebas exhaustivas, simulando diferentes escenarios de uso. También analizaremos su capacidad para integrarse con otros servicios y sistemas. Durante el proceso, recopilaremos datos sobre la experiencia del usuario, tiempos de respuesta y posibles fallos, y compararemos los resultados con los objetivos planteados.

3. Tecnologías No-Code y Low-Code

3.1. Diferencias entre Low-Code y No-Code

En el ámbito del desarrollo de software, es esencial comprender las diferencias sutiles pero significativas entre las categorías de No-Code y Low-Code. Ambos enfoques tienen como objetivo simplificar la creación de aplicaciones, pero su alcance y nivel de control varían considerablemente. [5]

En el caso del enfoque Low-Code, su audiencia principal está compuesta por desarrolladores profesionales que poseen cierto nivel de conocimientos técnicos. Estas herramientas requieren una base mínima de habilidades de programación, ya que algunas de ellas permiten la incorporación de funciones mediante la codificación de segmentos de código. Para los programadores experimentados, el Low-Code se convierte en un valioso aliado al agilizar la creación de aplicaciones, eliminando la necesidad de enfrentarse a tareas repetitivas o trabajo duplicado. Estas herramientas les permiten diseñar aplicaciones de manera rápida, a través de la manipulación visual de elementos y con un margen mínimo de intervención manual.

Por otro lado, las soluciones No-Code se dirigen a un espectro de usuarios mucho más amplio. A diferencia del Low-Code, están diseñadas para empleados que carecen de destrezas técnicas en informática. La construcción de aplicaciones utilizando el enfoque No-Code implica una serie de pasos visuales, lo que resulta en aplicaciones de menor complejidad y funcionalidades más limitadas. [6]

En términos de flexibilidad tecnológica, las diferencias entre ambas se vuelven aún más pronunciadas. En el caso del Low-Code, se permite la manipulación de su funcionalidad a través de cambios y adiciones de código, lo que afecta directamente el comportamiento de la aplicación. Esta característica ofrece ventajas en términos de personalización, ampliando su aplicabilidad en diversas situaciones. Sin embargo, esta capacidad puede acarrear problemas de compatibilidad con versiones anteriores. Cada actualización implica un proceso de comprobación minucioso, ya que modificaciones en funcionalidades previamente personalizadas podrían generar incompatibilidades.

Por contraste, las herramientas No-Code evitan este escenario. Su estructura cerrada garantiza que las actualizaciones no interfieran en el funcionamiento de las aplicaciones existentes. Esta característica se traduce en una ventaja considerable, ya que las actualizaciones no ponen en riesgo el rendimiento de las aplicaciones creadas mediante este enfoque.

3.2 Mitificación de los Conceptos Tecnológicos

En la misma línea de toda tecnología emergente que despierta expectativas debido a sus características particulares, se observa un fenómeno de exaltación excesiva en los medios de comunicación, lo que resulta en una mitificación. Esta exaltación distorsiona la percepción real de las tecnologías al exagerar sus atributos. Las propuestas de las tecnologías Low-Code y No-Code son particularmente atractivas, especialmente para aquellos sin formación técnica en desarrollo de software. [7]

De hecho, con el fin de aumentar la visibilidad de los artículos, varios sitios web presentan estas tecnologías como una revolución que transformará completamente el panorama informático. Pero, ¿es esto realmente cierto? En el caso del No-Code, su enfoque no pretende ser una solución universal para todas las problemáticas; más bien, se dirige a entornos específicos [3]. Las aplicaciones web representan un entorno idóneo para el No-Code, ya que funciona especialmente bien para aplicaciones con funcionalidades simples. Además, la disponibilidad de numerosas herramientas gratuitas, combinada con la eliminación de la necesidad de conocimientos técnicos, lo hace accesible para una amplia audiencia.

Por otro lado, cuando se trata de aplicaciones con funcionalidades más complejas, el panorama cambia. Estas aplicaciones involucran algoritmos y elementos que no son perfectamente compatibles con el enfoque Low-Code y No-Code. Sin embargo, surge la pregunta: ¿podrían ofrecer un rendimiento adecuado a pesar de los desafíos? Esta cuestión impulsa el estudio de caso en este trabajo.

Es importante recalcar que no se puede afirmar que estas tecnologías sean útiles para todos los casos de uso. Cada situación debe someterse a un análisis previo para determinar su viabilidad en un proyecto específico.

3.3 Entrevista con un profesional

Aprovechando nuestra participación en diferentes eventos de informática, tuvimos la oportunidad de conocer a diversas personas apasionadas por la tecnología. Entre ellas, destacamos la experiencia con un desarrollador excepcional, Alejandro Vicente, quien desempeña un papel crucial en el campo de desarrollo e IT en la empresa Webel. Su generosidad al compartir sus experiencias y conocimientos tanto en el ámbito del "low code" como en la programación convencional resultó ser muy adecuada y beneficiosa para nuestro estudio.

En vista a sus perspectivas y experiencias decidimos plantearle varias preguntas respecto al caso que estamos estudiando, con el objetivo de obtener una opinión valiosa acerca de las soluciones Low-Code, y sus diferencias con los métodos de programación convencional. Las preguntas realizadas se pueden consultar en el **Anexo I**, al final de este documento.

Después de escuchar y analizar en detalle sus respuestas, se podría decir que, las soluciones Low-Code son consideradas por los expertos como soluciones ideales para quienes desean una introducción suave a la programación o necesitan crear aplicaciones sencillas de manera eficiente. En cuanto a la calidad del software, pueden ser robustas y seguras, dado que utilizan componentes previamente probados, pero también pueden limitar la personalización en comparación con enfoques de programación tradicionales.

Además, estas soluciones encuentran un nicho exitoso en la educación y en casos de uso simples, como la enseñanza de conceptos básicos de programación o la creación de aplicaciones internas. Sin embargo, es poco probable que reemplacen por completo los métodos de programación convencionales, especialmente en proyectos complejos y de alto rendimiento, donde la flexibilidad y el control precisos son esenciales.

4. Estado del arte

4.1 Introducción al desarrollo de software y los enfoques tradicionales

El desarrollo software es el proceso de entender, proyectar, especificar, diseñar, programar, comprobar y documentar aplicaciones informáticas, facilitando posteriormente su mantenimiento de manera controlada y segura.

Uno de los enfoques tradicionales más utilizados para diseño y desarrollo software es el *Desarrollo basada en el ciclo de Vida de Software,* enfoque que sigue una serie de etapas/fases secuenciales como la planificación, el diseño, la implementación, pruebas, culminando con las etapas finales de despliegue y mantenimiento del software. Cada una de las etapas se lleva a cabo de manera secuencial, lo que implicará que los cambios realizados en una de las etapas, puedan suponer un impacto significativo en las etapas posteriores. [10]

Asimismo, en los diferentes enfoques tradicionales, se requería un alto nivel de conocimiento de lenguajes específicos de programación y una amplia experiencia en desarrollo de software para poder materializar los proyectos deseados de forma exitosa. Estos enfoques estaban basados en el desarrollo de código fuente a medida, empleando lenguajes de programación variados como Java, C++, PHP o Python. Sin embargo, todo el desarrollo requería mucho tiempo, un esfuerzo considerable, y a menudo era necesario un equipo de desarrollo especializado para llevar a cabo proyectos complejos. [9]

Por lo tanto, si bien estos enfoques tradicionales han demostrado su eficacia a lo largo de los años, también presentan numerosos desafíos. El desarrollo software puede llegar a resultar muy costoso, lento, propenso a errores, sobre todo en proyectos de gran escala, e incluso resultar muy engorroso y requerir un nivel de aprendizaje para programadores y desarrolladores menos experimentados.

En resumen, los enfoques tradicionales de desarrollo de software han demostrado en el pasado toda su valía. Sin embargo, debido a la continua necesidad de una mayor eficiencia y agilidad en el desarrollo de aplicaciones, han surgido nuevas y numerosas soluciones como las soluciones Low-Code, que tratan de superar las limitaciones de los enfoques tradicionales.

4.2 Concepto y fundamentos de las soluciones low-code

Las soluciones Low-code son plataformas de desarrollo de aplicaciones que permiten a todo tipo de usuarios crear software funcional con un mínimo conocimiento sobre codificación manual. Estas herramientas están diseñadas para acelerar y simplificar el proceso de desarrollo proporcionando conceptos de alto nivel y componentes visuales que permiten al usuario plantear, diseñar, desarrollar y desplegar aplicaciones con mayor facilidad.

El objetivo principal y fundamental de las soluciones Low-code es permitir a usuarios con diferentes niveles de conocimiento de programación participar de forma activa en el proceso de desarrollo software de aplicaciones de forma activa, mediante interfaces visuales intuitivas basadas en "arrastrar y soltar", donde los usuarios puedan seleccionar y configurar los detalles y componentes deseados para construir sus aplicaciones como: formularios, botones, tablas, flujos de trabajo, etc. [11]

Además de los componentes visuales, las soluciones Low-code también incorporan motores de reglas de negocio, que permiten definir la lógica y los flujos de trabajo de la aplicación de manera visual. Estos motores ayudan a automatizar procesos comerciales y a tomar decisiones basadas en reglas establecidas previamente, sin necesidad de escribir código manualmente.

Otro aspecto importante de las soluciones Low-code es la generación automática de código. A medida que un usuario diseña y configura su aplicación, la plataforma genera automáticamente el código subyacente necesario para que la aplicación funcione. Esto reduce significativamente la cantidad de codificación manual requerida, acelerando notablemente el proceso de desarrollo y minimizando posibles errores. [12]

En resumen, las soluciones Low-code son herramientas que permiten a los usuarios desarrollar aplicaciones sin requerir un conocimiento profundo de programación tradicional. A través de componentes visuales, motores de reglas de negocio y generación automática de código, estas soluciones ofrecen una forma más cómoda, accesible y eficiente de crear software funcional.

4.3 Evolución de las soluciones low-code

A lo largo de los años, las soluciones Low-code han experimentado una evolución significativa desde su concepción inicial. A medida que las demandas y las necesidades de desarrollo de software han evolucionado, estas soluciones han seguido el ritmo, adaptándose y mejorando para abordar los desafíos emergentes.

Las soluciones Low-code tienen sus raíces en los lenguajes de programación de cuarta generación (4GL), lenguajes desarrollados entre los años 1970 y 1990. Al igual que las soluciones Low-code desarrolladas en la actualidad, los métodos empleados durante el final del siglo XX (p.e 4GL o RAD → Desarrollo rápido de aplicaciones) y principios del siglo XXI (p.e. las plataformas móviles) comparten el mismo propósito general: simplificar y acelerar el proceso de desarrollo de aplicaciones, incluyendo aspectos como el soporte de bases de datos, generación de informes, desarrollo de interfaces gráficos de usuario (GUI), desarrollo web, etc. [27]

En las primeras etapas de desarrollo de las soluciones Low-code, los programadores se centraban principalmente en la creación de aplicaciones rápidas, básicas y sencillas, con entornos fáciles de utilizar, y con un conjunto limitado de componentes y funcionalidades. Estas soluciones iniciales brindaban una introducción temprana a los conceptos de desarrollo visual y generación automática de código, presentando sin embargo ciertas limitaciones en términos de flexibilidad y personalización. [15]

A medida que la implementación de las soluciones Low-code aumentó y su potencial quedó evidente, se produjo una evolución significativa en términos de funcionalidad y capacidad. Las plataformas Low-code adquirieron una amplia gama de componentes y características adicionales, que permitían a los usuarios desarrollar aplicaciones cada vez más complejas y personalizadas. [13] Progresivamente, fueron introducidos nuevos componentes especializados para diferentes tipos de aplicaciones, como formularios dinámicos, integraciones con servicios en la nube, desarrollo web y capacidades de colaboración en tiempo real.

Además, las soluciones Low-code evolucionaron en términos de escalabilidad y capacidad de integración con sistemas y tecnologías existentes. Se agregaron funcionalidades de API y conectividad con bases de datos, permitiendo a los usuarios acceder y manipular datos externos de manera más cómoda y sencilla. Esto abrió la puerta a la integración de aplicaciones Low-code con otros sistemas empresariales, lo que mejoró la utilidad y la viabilidad en entornos empresariales más amplios.

La evolución de las soluciones low-code también ha estado impulsada por avances tecnológicos más amplios. La aparición de tecnologías como la inteligencia artificial (IA) y el aprendizaje automático (machine learning) ha permitido a las soluciones Low-code ofrecer características de automatización y generación de código mucho más sofisticadas. Actualmente la IA puede ayudar sugiriendo al usuario de manera automática componentes, reglas de negocio y flujos de trabajo, acelerando aún más el proceso de desarrollo. [15]

4.4 Ventajas y desventajas de las soluciones Low-code

Las soluciones low-code ofrecen una serie de ventajas que las han convertido en una opción popular y atractiva para el desarrollo de aplicaciones. Sin embargo, también presentan ciertas desventajas que deben tenerse en cuenta. A continuación, exploraremos tanto las ventajas como las desventajas o desafíos asociados con las soluciones Low-code:

Ventajas de las soluciones low-code:

- ✓ **Facilidad y rapidez de desarrollo:** Las soluciones Low-code permiten a los desarrolladores crear aplicaciones de manera más rápida y sencilla mediante el uso de componentes visuales y la generación automática de código. Al utilizar interfaces visuales intuitivas y herramientas de arrastrar y soltar, las aplicaciones son construidas de forma más eficiente, acelerando el proceso de desarrollo, reduciendo sus costes, y reduciendo la dependencia de la codificación manual. [11]
- ✓ **Mayor accesibilidad:** Al proporcionar interfaces visuales intuitivas y abstracciones de alto nivel, las soluciones Low-code permiten que personas con menos experiencia en programación participen en el desarrollo de aplicaciones. Esto amplía el grupo de personas que pueden contribuir al proceso de desarrollo de software.
- ✓ **Mejora de la productividad:** Al automatizar ciertos aspectos del desarrollo de software, como la generación de código y la configuración de componentes, las soluciones Low-code pueden aumentar la productividad de los equipos de desarrollo. De esta manera, los desarrolladores pueden centrarse en aspectos más estratégicos y de alto valor, en lugar de tareas repetitivas y tediosas. [18]
- ✓ **Flexibilidad y adaptabilidad:** Las soluciones Low-code ofrecen una mayor flexibilidad para la personalización y adaptación de aplicaciones. De esta manera, tanto usuarios como desarrolladores pueden aprovechar una amplia gama de componentes y funcionalidades para ajustar la aplicación a sus diferentes necesidades, lo que facilita la iteración y la evolución continua de la aplicación a medida que dichas necesidades vayan cambiando. [16]

Desventajas de las soluciones Low-code:

- × **Limitaciones de personalización:** Aunque las soluciones Low-code ofrecen un alto nivel de flexibilidad, pueden presentar limitaciones en términos de personalización avanzada y desarrollo de características complejas. Algunas aplicaciones pueden requerir código personalizado adicional más allá de las capacidades de la plataforma Low-code.
- × **Curva de aprendizaje inicial:** A pesar de ser más accesibles que los enfoques tradicionales de desarrollo de software, las soluciones Low-code requieren un período de aprendizaje para familiarizarse con las plataformas y su funcionamiento. Además, puede existir limitaciones iniciales en la capacidad de los desarrolladores para aplicar conocimientos avanzados de programación. [17]

- **Dependencia del proveedor:** Al adoptar una solución Low-code específica, los desarrolladores pueden quedar vinculados a un proveedor particular y a las limitaciones impuestas por la plataforma. Por esta razón, cambiar de plataforma puede ser costoso y requerir una reescritura significativa del código, lo que puede suponer una preocupación a largo plazo. [18]
- **Complejidad de integración:** La integración de aplicaciones Low-code con sistemas existentes plantea desafíos debido a la necesidad de interoperabilidad y la compatibilidad con otros sistemas y tecnologías. Para llevar a cabo esta integración, es necesario realizar un esfuerzo adicional y un análisis exhaustivo de los protocolos de comunicación y las interfaces de programación de aplicaciones (API) que se utilizan en los sistemas heredados o en los componentes externos.
- **Limitaciones de escalabilidad:** Las soluciones Low-code son ideales para el desarrollo rápido de prototipos y aplicaciones de tamaño mediano. Sin embargo, pueden enfrentar dificultades en términos de escalabilidad para aplicaciones empresariales de gran envergadura, debido a que la arquitectura y las capacidades de escala de las plataformas Low-code pueden ser limitadas en comparación con enfoques más tradicionales. [11]
- **Dependencia de actualizaciones de la plataforma:** Las soluciones Low-code evolucionan rápidamente, estando sujetas a actualizaciones frecuentes por parte de los proveedores. Aunque esto pueda ser una ventaja en términos de nuevas características y mejoras, también puede generar dependencia en las actualizaciones de la plataforma. Los desarrolladores deben estar preparados para adaptarse a los cambios y asegurarse de que sus aplicaciones sigan siendo compatibles con las versiones actualizadas de la plataforma.

En conclusión, es muy importante tener en cuenta tanto las ventajas como las desventajas de las soluciones Low-code al evaluar su idoneidad para un proyecto o contexto específico. Si bien las soluciones Low-code pueden ofrecer beneficios significativos en términos de desarrollo rápido, accesibilidad y productividad, también es esencial considerar los desafíos o dificultades que puedan presentar en términos de restricciones, limitaciones, o dependencias.

4.5 Impacto de las soluciones Low-code en el ciclo de desarrollo de software

Las soluciones Low-code han revolucionado el ciclo de desarrollo de software, introduciendo cambios y mejoras en diferentes etapas del proceso. A continuación, se explorarán los diferentes efectos que las soluciones low-code han tenido en el ciclo de desarrollo de software:

1. **Agilidad en el desarrollo**: Las soluciones Low-code permiten un desarrollo más rápido al proporcionar herramientas visuales y abstracciones de código, facilitando a los desarrolladores la creación de prototipos y aplicaciones funcionales en menos tiempo, acelerando así el ciclo de desarrollo. La capacidad de arrastrar y soltar componentes predefinidos y utilizar flujos de trabajo automatizados facilita la creación rápida de aplicaciones. [19]
2. **Colaboración multidisciplinaria:** Estas soluciones facilitan la participación en el desarrollo software de personas con diferentes niveles de conocimiento técnico, fomentando la comunicación y colaboración entre equipos y roles.
3. **Retroalimentación acelerada:** Con las soluciones Low-code, es posible obtener rápidamente comentarios y realizar iteraciones en el desarrollo de software. Este proceso se basa en mostrar prototipos o versiones iniciales de la aplicación a los usuarios finales y obtener comentarios tempranos. Esto facilitará a los desarrolladores una mejor comprensión de los requisitos y necesidades del usuario, lo que a su vez conduce a un desarrollo más eficiente y satisfactorio de la aplicación. [20]
4. **Reducción del tiempo de desarrollo y mantenimiento:** Automatizar tareas repetitivas y proporcionar componentes reutilizables, implicará una mayor eficiencia y una clara reducción del tiempo necesario para desarrollo y mantenimiento de aplicaciones, y una mayor eficiencia. [16]
5. **Flexibilidad y adaptabilidad:** Las soluciones Low-code permiten personalizar y extender la funcionalidad de las aplicaciones, adaptándolas a los requisitos específicos del proyecto, y a los cambios que este puede sufrir.
6. **Enfoque centrado en la experiencia del usuario:** Estas soluciones priorizan la creación de interfaces intuitivas y atractivas, mejorando la usabilidad y la experiencia del usuario.

En resumen, las soluciones Low-code han transformado el ciclo de desarrollo de software al acelerar y reducir el tiempo de desarrollo y mantenimiento, fomentando la colaboración multidisciplinaria, facilitando la retroalimentación, brindando flexibilidad y adaptabilidad, y mejorando la experiencia del usuario.

4.6 Conclusiones parciales

En este capítulo se ha examinado el marco teórico relacionado con las soluciones Low-code y su evolución. Se ha explorado el concepto de las soluciones Low-code, sus fundamentos principales, su origen y su crecimiento en popularidad en los últimos años.

Además, se ha analizado su impacto en el ciclo de desarrollo de software, destacando sus ventajas en términos de agilidad, colaboración multidisciplinaria, retroalimentación acelerada, reducción del tiempo de desarrollo y mantenimiento, flexibilidad y adaptabilidad, así como la priorización de la experiencia del usuario. [14]

Asimismo, se han abordado las limitaciones y desafíos asociados con las soluciones Low-code, como la escalabilidad para aplicaciones empresariales de gran envergadura. Si bien las soluciones Low-code son altamente efectivas para el desarrollo rápido de prototipos y aplicaciones de tamaño mediano, es esencial considerar cuidadosamente su idoneidad para proyectos de mayor envergadura, teniendo en cuenta las limitaciones de escalabilidad y las necesidades específicas del negocio.

Estas conclusiones parciales resaltan la importancia de comprender las implicaciones de adoptar soluciones Low-code en el desarrollo de software, evaluando cuidadosamente sus ventajas y limitaciones en función de los requisitos y objetivos del proyecto.

4.7 Estudios posteriores

En los próximos capítulos se proyecta realizar una encuesta ampliada y dirigida a un público más diverso para obtener una visión holística de las percepciones y necesidades relacionadas con las soluciones Low-code. Esta encuesta se distribuirá entre distintos perfiles de usuarios, incluyendo profesionales de diferentes sectores y personas con escasa experiencia en programación.

Además, se tiene previsto realizar una exhaustiva exploración de diversas plataformas de low-code disponibles en el mercado actual. Esta amplia selección incluirá programas como Power Apps, SAP, OutSystems, Mendix, Appian, Zoho Creator y Google App Maker, entre otras destacadas soluciones. La elección de estas herramientas responde al propósito de abarcar un abanico representativo de opciones que aborden diversas necesidades y contextos de desarrollo.

Si bien el análisis se extenderá a múltiples plataformas de Low-code, se ha optado por un enfoque más detallado en un programa concreto que ha demostrado un impacto significativo en la industria y que han ganado popularidad en diversos ámbitos. De este modo, el elegido para el estudio en profundidad es Microsoft Power Apps, respaldado por su integración con Microsoft y su creciente adopción en el desarrollo de aplicaciones.

5. Recopilación y análisis de datos

Las herramientas Low-code son un avance de la informática que poco a poco van siendo conocidas por una amplia multitud de usuarios, sin importar las edades o el sector al que cada uno se pueda dedicar.

En este apartado realizaremos un análisis exhaustivo acerca de las herramientas Low-code, creando un cuestionario con la herramienta Google-Forms. El cuestionario diseñado recopila diferentes tipos de datos incluyendo preguntas iniciales sobre la edad y ocupación de cada persona, preguntas relevantes relacionadas con el conocimiento y experiencia sobre programación y soluciones Low-code, percepciones sobre su utilidad y aplicabilidad, así como expectativas y preocupaciones al considerar su uso.

Es importante resaltar 2 aspectos:

1. El propósito de esta encuesta es establecer un punto de partida para comprender la interacción con el área y su nivel de adopción entre los usuarios profesionales y no profesionales.
2. Durante la resolución del formulario, se explica al público brevemente que son las herramientas Low-code, para facilitar su comprensión en caso necesario y como estas podrían facilitar el uso de herramientas de programación.

5.1 Estructura de la encuesta

La encuesta, que se presenta en el **Anexo II**, aborda una serie de preguntas con el propósito de evaluar el conocimiento y las necesidades de los encuestados en relación con el desarrollo de software y la programación. Las áreas clave que se exploran en la encuesta incluyen:

1. **Conocimientos informáticos y experiencia en desarrollo de software:** Se indaga sobre la experiencia y los conocimientos en la programación y el desarrollo de software, abarcando todo tipo de niveles, desde niveles básicos hasta avanzados.
2. **Interés en crear aplicaciones y tipos de aplicaciones deseadas:** Se explora el deseo de desarrollar aplicaciones adaptadas a necesidades personales o laborales, así como, ejemplos concretos de aplicaciones.
3. **Expectativas y desafíos en el desarrollo sin conocimientos profundos**: Se exploran tanto las expectativas sobre la facilidad de uso y la eficiencia de las herramientas de desarrollo, como las posibles dificultades contempladas al crear aplicaciones sin profundos conocimientos de programación.

4. **Interés en Soluciones Low-code:** Tras una breve introducción de los conceptos básicos de las soluciones Low-Code, se evalúa el interés en poder utilizarlas para desarrollo de aplicaciones.
5. **Deseo de aprendizaje, capacitación o recursos:** Se investiga la disposición y el deseo por recibir capacitación o recursos para aprender a utilizar las soluciones Low-code.
6. **Ventajas y preocupaciones percibidas de Low-code:** En esta segunda parte de la encuesta, dedicada a un personal con cierto nivel de experiencia, se exploran las ventajas percibidas de las soluciones Low-code en comparación con otros métodos de desarrollo, y se evalúan las preocupaciones en torno al uso de estas soluciones.
7. **Funcionalidades Esenciales:** Se identifican las funcionalidades consideradas esenciales en una plataforma Low-code
8. **Disposición a Sustituir Lenguajes Convencionales:** Se indaga si considerarían sustituir lenguajes de programación convencionales por Low-code.
9. **Opinión sobre Colaboración en Proyectos Low-code:** En útlimo lugar, se recoge la opinión sobre la colaboración en proyectos de desarrollo de software mediante Low-code.

Un aspecto importante a resaltar sobre la encuesta es la estructura multipartita, es decir, tendremos una encuesta diseñada en varias secciones. La primera sección está diseñada para ser completada por todos los usuarios, con el propósito de recopilar información general. Seguidamente, se encuentra una segunda parte, dirigida también a todos los encuestados, pero con un enfoque específico en aquellos que han manifestado interés en la tecnología "low-code" después de la presentación de las diapositivas explicativas. Finalmente, la última sección se reserva exclusivamente para aquellos usuarios que han demostrado un nivel medio o experto de conocimiento y experiencia en el desarrollo de aplicaciones. Este enfoque graduado permite una recopilación de datos más precisa y orientada a las necesidades y competencias de los encuestados.

5.2 Resultados obtenidos

En primer lugar, comenzamos el cuestionario preguntando detalles sobre nuestros encuestados, como edad, profesión u ocupación, así como sus conocimientos acerca de la programación y desarrollo software, o el interés que este ámbito supone para cada uno de ellos.

La edad del público varía desde los 16, 17 años, edad de un estudiante de instituto, hasta los 80 años que puede tener una persona jubilada. Sin embargo, la mayoría se encuentran entre los 20 y los 30 años, con una alta variedad de ocupaciones (estudiantes, ingenieros, sanitarios, funcionarios, cuerpo militar, trabajadores en ámbitos deportivos, etc.)

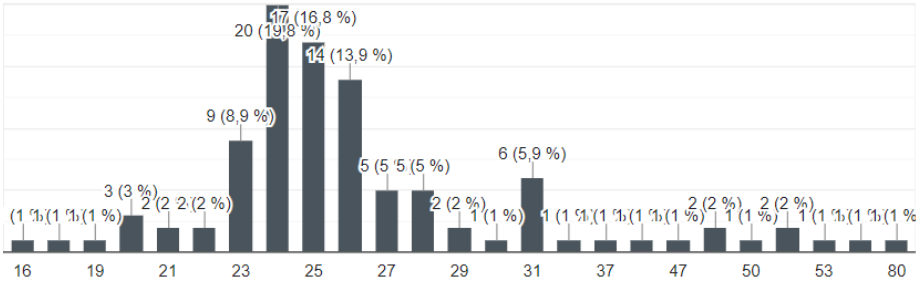


Figura 5.1

Pese a tener una media de edad joven, cerca de los 28 años, la experiencia con las herramientas de programación y desarrollo de software es muy escasa, siendo más del 50% (*figura 5.2*), personas con nivel básico.

Sin embargo, como vemos en la *figura 5.3* la falta de conocimientos y experiencia no supone un gran inconveniente, siendo el interés por la creación de aplicaciones y software personalizado muy alto entre nuestro público.

Figura 5.2

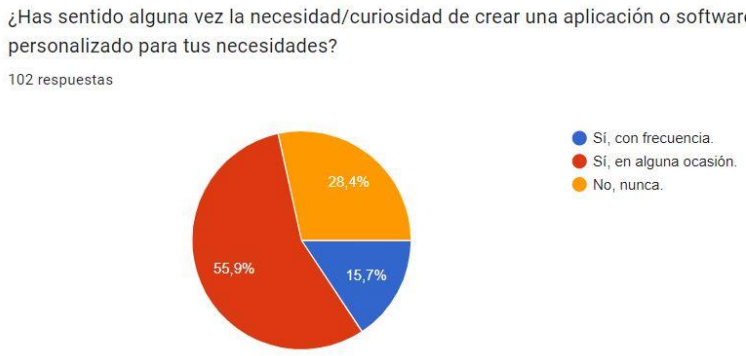


Figura 5.3

Si bien algunas personas están poco conectadas con los avances tecnológicos y las facilidades que la informática nos proporciona, muchas otras sienten una gran curiosidad por conocer nuevos métodos de trabajo y/o maneras que puedan facilitar su vida personal.

No obstante, la mayoría de ambos grupos encuentran múltiples obstáculos a la hora de enfrentarse a diferentes herramientas de desarrollo software: falta de conocimientos técnicos, complejidad del proceso de desarrollo software, falta de asistencia profesional, limitaciones de tiempo, o incluso falta de motivación.

Por ello, en caso de que fuera posible proporcionar unas herramientas software eficientes y fácil de utilizar los encuestados **SÍ** darían mayor importancia a desarrollar aplicaciones propias. Para fortalecer este argumento, observamos como la *figura 5.4* nos muestra una evidencia de la opinión general, siendo más del 90% personas que considerarían óptima la idea de poder crear aplicaciones para usos personales y diarios.

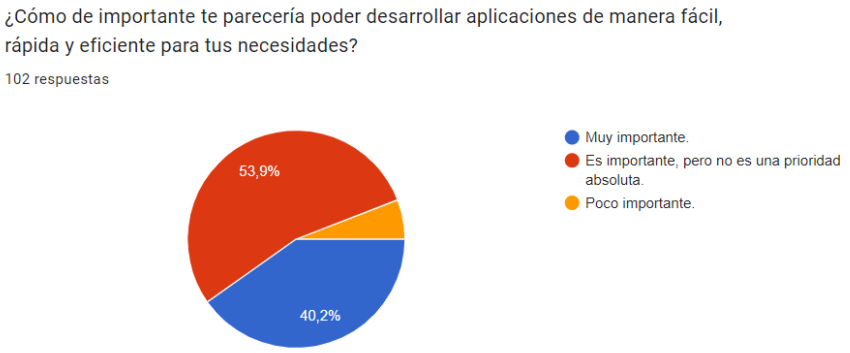


Figura 5.4

Para poder concretizar en más detalle este asunto, la gráfica mostrada en la *figura 5.5* nos permite verificar las posibles facilidades que una plataforma de desarrollo software puede presentar. Más del 50% de los encuestados ven como necesario principalmente que las plataformas sean simples de utilizar incluso para perfiles con conocimientos básicos, y en casos convenientes poder optar a asistencia adecuada, guías paso a paso para el proceso de desarrollo o una amplia documentación para situaciones problemáticas.

Figura 5.5

Otro aspecto importante a mencionar es la eficiencia. Un porcentaje mayor de 45% de los encuestados buscan rapidez en la creación de software de aplicaciones basada en posibles plantillas predefinidas, mientras que otros (42%) consideran muy importante la necesidad de personalización del software, con el objetivo de poder adaptarlo a sus propias necesidades.

En otro orden de ideas, también podemos comentar las aplicaciones más solicitadas y/o necesitadas por los encuestados siendo las más comunes aquellas con fines personales o laborales:

- Aplicaciones que permiten automatizar tareas repetitivas y gestionar óptimamente los proyectos. Son las más solicitadas y demandadas (>52%).
- Aplicaciones centradas en el seguimiento de salud y el estado físico. (>44%)
- Otro tipo de aplicaciones bien valoradas son aquellas cuya finalidad se basa en la gestión de los gastos y en la productividad personal. (>40%)
- Aplicaciones con perfil interactivo, como la comunicación o actividades de ocio, son menos solicitadas (<30%).
- Un pequeño porcentaje de interés es mostrado hacia ideas de aplicaciones aun no diseñadas.

A continuación, optamos por introducir a los encuestados unas diapositivas breves resumiendo el concepto de *Low-code* y las facilidades que esta herramienta podría ofrecer para que el desarrollo software sea accesible para usuarios de todo tipo y nivel.

En base a los conceptos explicados en dichas diapositivas acerca del Low-code, podemos observar como el interés total generado, abarcando diferentes niveles de interés, es superior al 95% en el caso de los encuestados (*figura 5.6*). Esto demuestra que, si se proporcionan soluciones o herramientas menos complejas y tediosas para desarrollo de aplicaciones como puede llegar a ser la programación en muchas ocasiones, los usuarios intentarían en casos oportunos o necesarios utilizarlas.

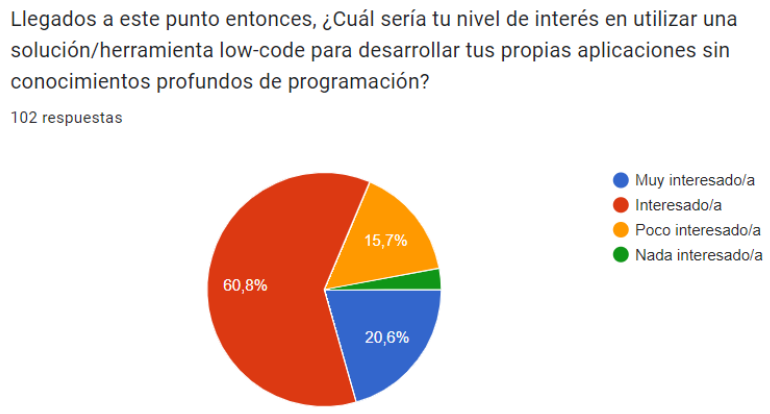


Figura 5.6

Para fortalecer estos argumentos, basándonos en las *figuras 5.7 y 5.8*, vemos que más de un 80% resultaron interesados en invertir tiempo para recibir formación y ayuda sobre las plataformas Low-code. Las principales herramientas de ayuda que podemos destacar son: asistencia personal, documentación detallada, foros y comunidades de usuarios, o lo más solicitado, tutoriales y guías paso a paso.

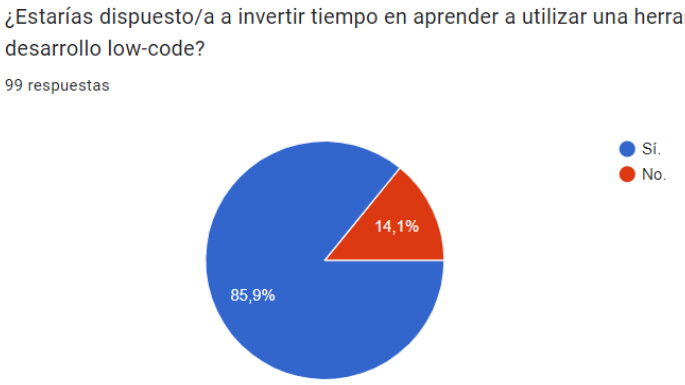


Figura 5.7

Figura 5.8

Sin embargo, si es cierto que un pequeño grupo, cercano al 20% como vemos en la figura 5.8 prefieren invertir tiempo, pero explorando por cuenta propia las diferentes plataformas Low-code, investigando sobre las diferentes herramientas que proporcionan, así como las ventajas y limitaciones que pueden estas plataformas presentar en comparación a la programación convencional. En este caso, se trata principalmente de personas con un nivel intermedio o avanzado de programación.

Análisis basado en experiencia

A partir de este punto, trataremos con encuestados de nivel intermedio o experto, que conforman aproximadamente un 30%. El objetivo de esta parte es poder observar y analizar de forma detallada las ventajas, limitaciones, necesidades e impresiones acerca de las plataformas Low-code, en comparación a las soluciones de desarrollo convencionales.

En base a las respuestas recibidas, podemos afirmar que el principal factor a favor de las soluciones Low-code son la optimización del tiempo de desarrollo de software y la posibilidad de no depender de profesionales o expertos en desarrollo, es decir, desarrollar de forma autónoma sus propias aplicaciones, propios programas, etc.

Otro factor importante y encontrado por los participantes encuestados, son la alta flexibilidad mostrada por estas soluciones Low-code, así como la adaptabilidad a todo tipo de usuarios y sus diferentes necesidades. Funcionalidades como "arrastrar y soltar elementos", la integración con bases de datos y sistemas ya existentes, o la generación automática de código, son funcionalidades o características muy solicitadas y buscadas en el mercado, motivo por el cuál muchos programadores, desarrolladores o simples informáticos optan hoy en día por utilizar soluciones Low-code con el objetivo de reducir los tiempos y la carga de trabajo.

Sin embargo, si se habla de desventajas, las principales mencionadas fueron las limitaciones en la personalización de aplicaciones y los problemas de escalabilidad y rendimiento, indicando que las soluciones/plataformas Low-code todavía tienen un amplio margen de mejora para poder reducir contratiempos.

Otras desventajas señaladas fueron la dependencia de diferentes proveedores, o la necesidad de garantizar la seguridad y la protección de datos; las soluciones Low-code deben evitar brechas de seguridad, proteger datos sensibles y cumplir con regulaciones de privacidad. Garantizar la confidencialidad e integridad de la información es crucial en entornos de desarrollo rápido y fácil, previniendo riesgos y daños reputacionales.

Por último, una vez vistas las consideraciones acerca del Low-code, podemos observar y deducir que, los participantes estarían generalmente dispuestos a sustituir los lenguajes convencionales de programación por herramientas Low-code, dependiendo de los proyectos o aplicaciones específicas, y evaluando las ventajas y desventajas. Asimismo, con el objetivo de mejorar la eficiencia y la productividad del desarrollo de aplicaciones, más de un 90% considerarían oportuno utilizar las plataformas Low-code para compartir y colaborar en proyectos de desarrollo de software.

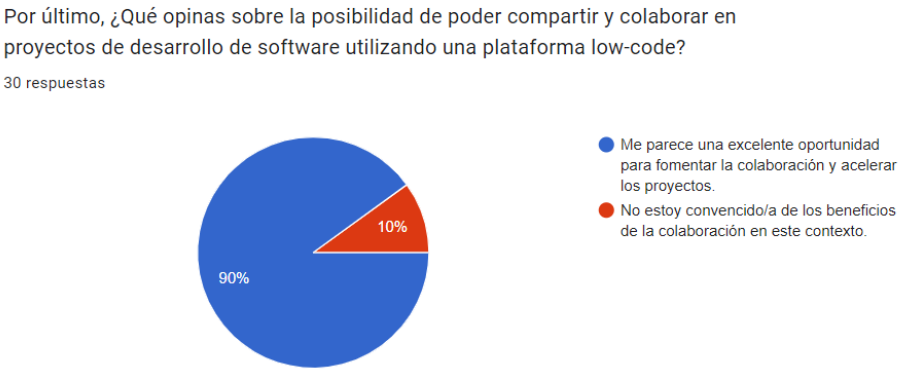


Figura 5.8

En resumen, los datos obtenidos revelan que existe una necesidad y un interés significativo en utilizar herramientas Low-code para el desarrollo de aplicaciones, especialmente entre aquellos sin conocimientos profundos de programación. Los encuestados valoran la facilidad de uso, la posibilidad de crear aplicaciones personalizadas y la disponibilidad de soporte y recursos de aprendizaje. Estos resultados respaldan la relevancia y la oportunidad de explorar y evaluar las soluciones Low-code en el contexto de este estudio.

5.3 Consideraciones éticas

Para preservar el anonimato de los participantes, se ha asignado un identificador numérico a cada encuesta comenzando por el 1 e incrementando de manera sucesiva con cada encuesta realizada, en lugar de utilizar nombres o información personal. De esta manera, la recopilación de datos no posibilita identificar individualmente a los participantes en los resultados y análisis; cualquier dato identificable, como direcciones de correo electrónico o nombres, fue omitida para garantizar la privacidad de los participantes. La información se almacenará en el perfil personal del alumno y solo será accedida por el mismo y su tutor.

Al finalizar la encuesta, se ha agradecido a los participantes su colaboración y tiempo dedicado a contribuir con la investigación. Se reconoce su importancia y el valor que su aportación tiene en el estudio. Si los participantes así lo desean, se les ofrece la opción de recibir un resumen de los resultados finales o ser informados sobre las conclusiones generales del estudio. Además, en caso de que el trabajo sea publicado o presentado en algún evento, se considerará la posibilidad de mencionar la participación y aportación de los voluntarios en los agradecimientos o en cualquier otra forma de reconocimiento adecuada.

5.4 Limitaciones de la investigación

A pesar de los esfuerzos realizados para llevar a cabo una investigación rigurosa y completa sobre las soluciones Low-code, es importante reconocer que este estudio puede estar sujeto a ciertas limitaciones que pueden afectar la interpretación de los resultados y la generalización de las conclusiones.

El tamaño de la muestra de participantes en la encuesta puede influir en la representatividad de los resultados obtenidos. En este caso, hemos decidido lanzar la encuesta y proceder a analizarla después de alcanzar la centena de participantes, buscando un número confiable y óptimo para el estudio realizado.

Para evitar sesgos y ambigüedades en la investigación, se ha puesto especial énfasis en la formulación clara y precisa de las preguntas de la encuesta, asegurando que sean comprensibles para diferentes perfiles de encuestados, incluidos aquellos sin experiencia en programación. Se han utilizado opciones de respuesta amplias y variadas para permitir a los participantes expresar sus percepciones y opiniones de manera adecuada. Además, se ha tenido en cuenta la diversidad de los encuestados al seleccionar una muestra representativa y amplia, con el objetivo de abarcar diferentes perspectivas y contextos. Asimismo, se ha proporcionado información y aclaraciones sobre el concepto Low-code para asegurar una comprensión adecuada y evitar malentendidos. Estas medidas contribuyen a obtener datos más fiables y relevantes, lo que facilita un análisis más preciso y una interpretación más sólida de los resultados obtenidos.

6 Herramientas Low-code y No-Code

Para este apartado se abordará el estudio y análisis de diversas plataformas de desarrollo low-code y no-code que han surgido como alternativas revolucionarias en el ámbito de la creación de aplicaciones.

6.1 Herramientas Low-Code

Entre un abanico de opciones potenciales para nuestro estudio, encontramos varias plataformas Low-code como OutSystems, Mendix, Appian, Zoho Creator, Oracle, Google App Maker, SAP, ServiceNow, AgilePoint, Lego Mindstorms y Microsft Power Apps [4], muchas de ellas presentes en la siguiente ilustración (*figura 6.1*) que nos muestra un cuadrante clasificatorio de las principales tecnologías Low-code presentes en el mercado.

Figura 6.1 Cuadrante clasificatorio de las principales tecnologías Low-Code

Haciendo uso de la versión de prueba de determinadas plataformas pudimos tener una impresión general del conjunto de estas. Decidimos realizar pruebas sin incurrir en gastos, con el propósito de evaluar un amplio abanico de funcionalidades y características.

Importante destacar el gran interés que desprtaron plataformas como Mendix, Appian, Google App Make, Zoho Creator, entre otras. Sin embargo, las que captaron una atención particular y que, por ende, profundizaremos en su análisis fueron OutSystems, Lego Mindstroms y Power Apps.

1. Microsoft Power Apps

Microsoft Power Apps es una plataforma de desarrollo de aplicaciones Low-code que permite a usuarios crear de forma rápida y sencilla aplicaciones empresariales personalizadas sin necesidad de conocimientos profundos de programación. A través de su interfaz visual, sus plantillas y las herramientas de *arrastra y suelta*, los usuarios pueden diseñar sus propias aplicaciones web y móviles conectadas a bases de datos y servicios en la nube. [21] Además, el uso de la plataforma Power Apps no requiere ninguna descarga de las aplicaciones, de manera que se podemos utilizarla en cualquier lugar siempre y cuando se disponga de un ordenador conectado a internet.

En relación al modo de ejecución de las aplicaciones, este aspecto se distingue notablemente de otras herramientas Low-Code. Una característica destacada es la posibilidad de probar el comportamiento de la aplicación directamente desde la interfaz de diseño, sin necesidad de compilar ni realizar pasos previos.

Además, se ofrece una opción de ejecución que implica la descarga de la aplicación en un dispositivo móvil, donde las aplicaciones se almacenan en una lista accesible. Para probarla en el mismo dispositivo, basta con seleccionar la aplicación deseada y se ejecutará sin requerir protocolos adicionales. En la *figura 6.2* podríamos observar una simulación de uso de la aplicación desde un teléfono.

Figura 6.2 Vista previa de la aplicación en formato móvil

Power Apps también presenta una tercera forma de ejecución: la capacidad de iniciar la aplicación con solo presionar un botón (*figura 6.3*), o su equivalente en atajo de teclado, F5. Esta funcionalidad permite que la aplicación se ejecute sin salir del entorno de desarrollo, simplificando el proceso de pruebas y evaluación.

Figura 6.3 Botón de ejecución de aplicación en vista previa (esquina superior derecha)

Para asegurar un correcto aprendizaje de uso de la herramienta, Microsoft Power Apps proporciona a los usuarios nuevos cursos gratis, breves y detallados de diferentes niveles, como observamos en la *figura 6.4.*

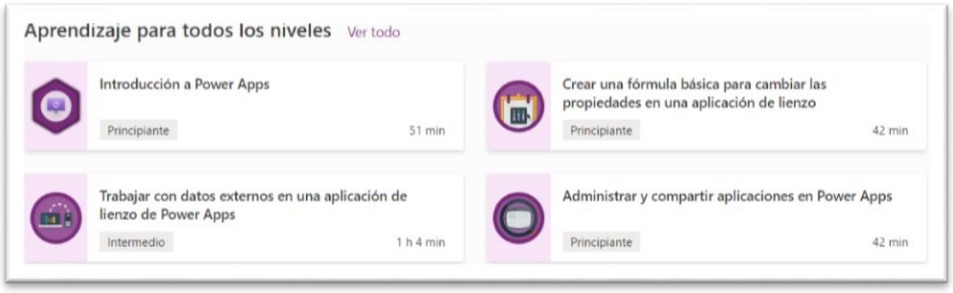


Figura 6.4: Cursos de Power Apps organizados por niveles de dificultad

Dichos cursos abordan conceptos esenciales a la hora de generar un proyecto, desde cómo crear aplicaciones hasta la descripción de las diferentes funciones predefinidas.

Por último, cabe mencionar los planes de pago que Microsoft ofrece. En primer lugar, tendremos un plan de prueba gratuito, que será introductorio y que permitirá a los usuarios autenticados utilizar la aplicación de Power Apps, durante un tiempo limitado de 30 días. Por otro lado, Microsoft ofrece también un plan de pago, **Power Apps Premium,** estimado a un precio de 18,70€ mensuales que permitirá a los usuarios compilar, modernizar e implementar aplicaciones ilimitadas. [21]

2. Lego Mindstorms

Lego Mindstorms es una línea de kits de robótica educativa creada por la compañía danesa Lego. Estos kits están diseñados para permitir a niños, jóvenes y adultos construir y programar robots funcionales de forma interactiva y creativa. Cada kit incluye bloques de Lego especiales, sensores y motores, que pueden ser ensamblados para crear robots con diferentes funcionalidades y características. Además, se proporciona un software de programación gráfica (compatible con Windows, Mac, Chromebook e iOS) que permite a los usuarios programar el comportamiento de los robots de manera intuitiva y sin necesidad de conocimientos avanzados de programación. [23]

Respecto a los modos de ejecución, Lego Mindstorms presenta un enfoque singular en comparación con otras plataformas. Los usuarios tienen la capacidad de controlar y evaluar el comportamiento de los robots directamente desde la interfaz de programación, sin la necesidad de llevar a cabo compilaciones previas ni pasos adicionales. Asimismo, se facilita la opción de cargar programas en los robots y ejecutarlos en modo autónomo, permitiendo ver sus creaciones en acción sin complicaciones.

- **Lego Mindstorms EV3**

Para facilitar la adquisición de habilidades en el uso de la herramienta, Lego Mindstorms proporciona cursos detallados con el conjunto de trabajo Lego Mindstorms EV3.

Figura 6.5 Kit educativo Lego Mindstorms EV3

En el ámbito educativo, Lego Mindstorms EV3 *(figura 6.5)* puede empezarse a utilizar con jóvenes a partir de los 9 años, hasta personas adultas que quieren experimentar con la programación, pero no tienen conocimientos al respecto, o simplemente para crear robots fácilmente. Muchos docentes optan por este pack para enseñar a sus alumnos robótica y STEM (Ciencia, Tecnología, Ingeniería y Matemáticas), con el objetivo de poder desarrollar el pensamiento creativo de estos.

Si los usuarios desean aprender por cuenta propia o realizar cursos online, los cursos introductorios son gratuitos. Sin embargo, si los usuarios desean realizar los cursos avanzados, estos abarcan precios desde 19.99€ hasta los 99.99€. Además, debemos mencionar que el kit de trabajo Lego Mindstorms EV3 que se debe adquirir para trabajar con la plataforma alcanza precios desde 180€ en adelante.

3. OutSystems

Por último, nos centraremos en la plataforma OutSystems, una plataforma de desarrollo de aplicaciones que destaca por su enfoque en la creación ágil y eficiente de software, proporcionando las herramientas y soluciones para poder desarrollar, administrar e implementar aplicaciones de carácter empresarial [22].

En comparación a otras plataformas el entorno de trabajo de la plataforma OutSystems requiere la descarga de una aplicación específica para iniciar la labor. En términos de simulación de ejecución, se sigue un proceso que involucra la verificación de la validez de la aplicación antes de lanzarla en un navegador. A diferencia de otras herramientas, esta metodología evita la necesidad de simuladores o excesivo consumo de recursos. Similar a la tercera opción de ejecución que nos ofrecía Microsoft Power Apps, OutSystems posibilita la prueba fuera del entorno de desarrollo. [24]

La plataforma OutSystems ofrece un marketplace llamado Forge, que provee componentes adicionales para aplicaciones móviles y web, y funciones pre-desarrolladas que permiten a los usuarios agilizar el proceso de desarrollo.

En cuanto al costo, existen notables diferencias; la versión de prueba es gratuita e ilimitada en tiempo, pero con funcionalidades limitadas. Sin embargo, las opciones de pago brindan acceso a características mucho más avanzadas y personalizables, lo que hace de OutSystems una solución versátil para crear aplicaciones empresariales de manera efectiva. Dichas opciones de pago requieren una tarifa mensual que oscila entre 1500 y 8000 euros.

Destacando las razones para elegir esta plataforma frente a las demás, se mencionan aspectos de interés. El entorno es intuitivo y permite la visualización gráfica de componentes y esquemas de base de datos, como se observa en la *figura 6.6*. La colaboración se destaca, con herramientas para gestionar modificaciones simultáneas. Además, el tratamiento de datos es sencillo, permitiendo la importación de bases de datos desde Excel y consulta detallada de su estructura. Generar aplicaciones para IOS y Android sigue un proceso uniforme, reduciendo el tiempo adicional.

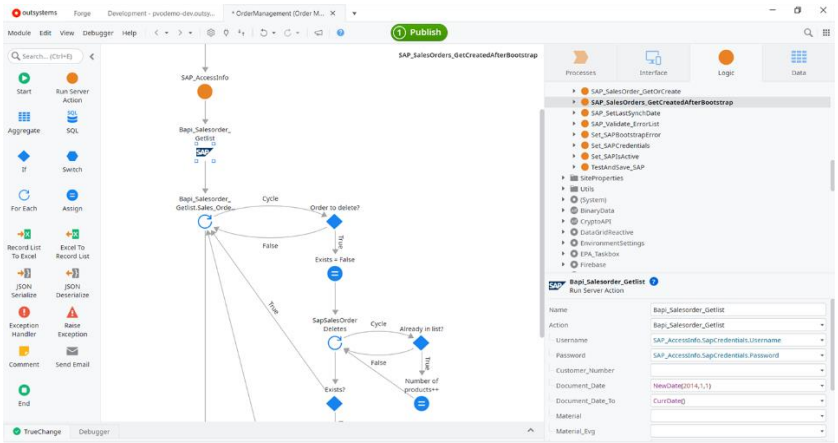


Figura 6.6 Entorno de trabajo de OutSystems (ejemplo)

OutSystems se presenta como una herramienta completa, capaz de desarrollar proyectos potentes con formación adecuada. Aunque ofrece funcionalidades avanzadas, su aprendizaje puede requerir tiempo significativo debido a la complejidad de sus características.

Debido a ello, para maximizar el potencial de OutSystems, la formación a través de cursos está disponible (*figura 6.7*). Estos cursos consisten principalmente en videos formativos seguidos de cuestionarios, y nos ofrecen diferentes opciones: podremos realizar los cursos sólo para practicar, podremos practicar y conseguir certificaciones de los cursos realizados, o diseñar en casos necesarios cursos de entrenamiento para equipos de trabajo con varios miembros. Las opciones de certificación y diseño de cursos son opciones de pago, siendo opción de sólo práctica gratuita, pero hasta un determinado nivel de dificultad.

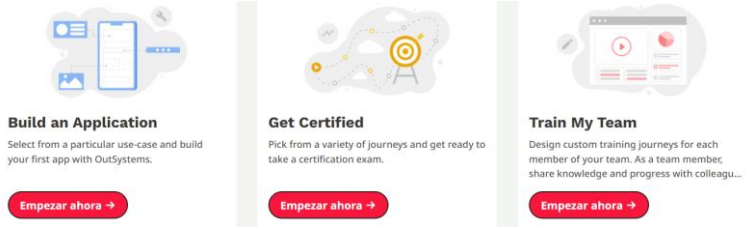


Figura 6.7 Acceso a cursos y certificaciones de OutSystems

Conclusiones

Si comenzamos hablando de Power Apps, es importante destacar su estrategia de producto, su enfoque en la innovación y su habilidad para integrar diversos servicios. Esta plataforma simplifica en gran medida el proceso de diseño, ofreciendo un amplio abanico de APIs y puntos de conexión. En términos de innovación, se sitúa en una posición avanzada al permitir a los desarrolladores aprovechar modelos de IA sin la necesidad de entrenarlos previamente.

No obstante, también presenta desafíos. La capacidad de respuesta del mercado y la estructura de precios son aspectos a considerar. En sus inicios, Microsoft incluyó la capacidad de utilizar Power Apps en las licencias más populares. Sin embargo, esta condición cambió en el tiempo, lo que resultó en interrupciones del servicio para muchos usuarios. Además, ciertos detalles como las limitaciones en el uso de flujos de procesos empresariales deben ser tenidos en cuenta; estos flujos pueden ser utilizados, pero solo para aplicaciones basadas en modelos.

En segundo, si hablamos de OutSystems, sus aspectos sobresalientes radican en la experiencia del usuario y en su arquitectura. Su interfaz de usuario alberga una amplia gama de plantillas y patrones, además de brindar componentes personalizables para la creación de aplicaciones. Además, permite crear y emplear patrones propios en la interfaz.

La arquitectura de OutSystems es otro punto fuerte, ya que permite a las aplicaciones interactuar con diversas fuentes de datos y servicios.

Sin embargo, hay consideraciones menos favorables. La estrategia de producto y la capacidad de desarrollo de usuarios no expertos son los aspectos más débiles. OutSystems tiende a ser más favorable para desarrolladores con experiencia, no siendo tan amigable para aquellos sin conocimientos previos en ingeniería del software.

Para terminar, investigando la plataforma Lego Mindstorms, pudimos observar que en comparación a las demás, esta plataforma Low-code enfocaba más su uso en aspectos educativos, fomentando la creatividad, la colaboración y el aprendizaje a través de la construcción y programación de robots. Su enfoque lúdico resulta muy atractivo para niños, jóvenes y adultos interesados en la robótica. Los kits vienen con piezas modulares y sensores intuitivos, facilitando su montaje sin habilidades avanzadas en ingeniería y la programación se hace mediante software gráfico, accesible para principiantes.

Como consideraciones no favorables, podemos comentar que para proyectos avanzados, la complejidad podría ser limitada debido a las piezas de Lego estándar y la programación gráfica, aunque amigable, puede ser menos versátil en proyectos complejos. Además, los kits pueden ser muy costosos, especialmente al ampliar capacidades con piezas adicionales.

6.2 Herramientas No-Code

Continuando con los enfoques previamente mencionados en la evaluación de las soluciones Low-Code, se presentarán las herramientas No-Code más prominentes en el ámbito, considerando tanto la experiencia personal como un análisis integral de artículos y estudios sobre las soluciones No-Code. Los criterios para seleccionar estas herramientas se mantendrán coherentes con los expuestos en la revisión de soluciones Low-Code.

Es relevante resaltar el considerable interés despertado por plataformas como Bubble, Code2, Webflow, Voiceflow o Appy Pie. No obstante, optamos finalmente por profundizar en la llamativa en términos de facilidad uso y obtención, Canva.

- **Canva**

Canva es una herramienta de diseño gráfico que se caracteriza por proveer cientos de plantillas para que los usuarios puedan crear sus propios formatos sin la necesidad de tener conocimientos en el área. Su interfaz es muy intuitiva y fácil de usar, lo que la convierte en una de las herramientas más usadas en el marketing digital [25].

Además, es la gran aliada de blogueros, Social Media y Community Managers, emprendedores y todo tipo de perfiles. Sus más de 8.000 plantillas permiten crear diseños como flyers, invitaciones, posts para redes sociales, infografías, currículums y todo lo "diseñable" que se nos pase por la mente.

Adentrándonos en la web de la herramienta decidimos crear un proyecto propio (un curriculum), para descubrir su herramientas y posibilidades. El entorno de trabajo es muy sencillo, ofreciendo la posibilidad de comenzar a trabajar sobre un diseño de proyecto predefinido, o en caso contrario, crear nuestro propio diseño del currículum.

Si optamos por la segunda opción, podremos añadir los elementos que se deseen, dibujar y escribir sobre la plantilla creada, o incluso añadir diferentes contenidos de aplicaciones externas como Youtube, Bitmoji, Lector de código QR, Giphy o Google Drive, entre otras. Todas las opciones se encuentran en el lateral derecho de la *figura 6.8.*

Figura 6.8 Entorno de trabajo de Canva

Si mencionamos los costes de la plataforma, podemos mencionar que Canva proporciona una opción de uso gratuita, que garantiza múltiples diseños, y facilidades. Sin embargo, para usos avanzados y expertos de la plataforma ofrece la opción de CanvaPro; los usuarios pueden adquirirla realizando un pago mensual de 12 euros, o anual de 100 euros, además de recibir un mes de prueba gratis. Esta opción ofrece mayor rapidez y número de diseños, y almacenamiento de los diseños ilimitado.

6.3 Herramienta del caso de estudio

En base a todos los estudios y artículos consultados, y a las versiones de prueba utilizadas de algunas plataformas ya mencionadas, optamos finalmente por profundizar en la plataforma de Microsft Power Apps. La elección de esta plataforma para nuestro trabajo de fin de grado, se fundamentó en una combinación de factores clave.

En primer lugar, se consideraron las opiniones y recomendaciones de desarrolladores y profesionales del ámbito del desarrollo de software, quienes destacaron la relevancia y el impacto de esta herramienta en el mercado actual.

Además, se evaluaron las facilidades de obtención y acceso a las plataformas, asegurando que estuviera disponible para el estudio y análisis en el marco de este trabajo de investigación.

Por último, se valoró especialmente la facilidad de uso y la accesibilidad de Power Apps, ya que uno de los objetivos de este estudio es comprender cómo las soluciones low-code pueden ser una opción atractiva y viable para usuarios con diversos niveles de conocimientos y habilidades en el desarrollo de software.

7. Estudio y desarrollo mediante plataformas Low-Code

Como se ha explicado en el apartado anterior, y con el propósito de profundizar en el estudio de este Trabajo de Fin de Grado, tomamos la iniciativa de investigar las capacidades y características de la plataforma Microsoft Power Apps. Analizando de manera detallada sus funcionalidades, facilidades de uso y posibles limitaciones, pudimos entender cómo se desenvuelve y cuáles son sus ventajas y desafíos en distintos contextos de desarrollo.

7.1 Desarrollo mediante Microsoft Power Apps

Con el fin de desarrollar las funcionalidades de diferentes aplicaciones utilizando Microsoft Power Apps, debemos realizar algunas aclaraciones básicas en relación con la herramienta Microsoft Power Apps y su entorno. Todo lo expuesto a continuación, ha sido en base a la versión 3.23075 de la herramienta.

Además, para adquirir habilidades específicas en el manejo de Microsoft Power Apps, hemos optado por participar en el curso titulado "Describe how to build applications with Microsoft Power Apps." Este programa de formación nos ha permitido obtener una comprensión profunda de las capacidades y funcionalidades de la herramienta, capacitándonos para explorar y desarrollar aplicaciones de manera efectiva para nuestro proyecto de TFG.

Página de inicio

En primer lugar, nos centramos en la página principal, donde encontramos un aspecto como el mostrado en la *figura 7.1*.

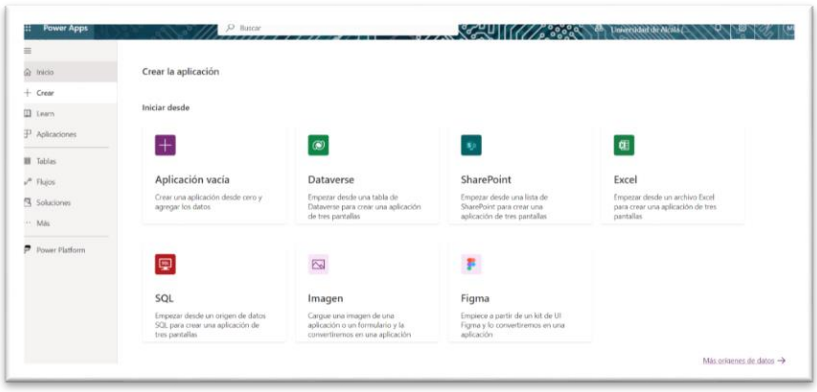


Figura 7.1 Pagina inicial de Power Apps.

Como se puede ver, se trata de un entorno intuitivo, donde podemos comenzar creando una aplicación propia, bien comenzando desde cero, o bien a partir de una herramienta ya existente (listas, tablas, bases de datos, imagen, etc.).

Si nos centramos en crear una aplicación nueva desde cero, podemos optar por utilizar una *aplicación de lienzo en blanco*, opción que será la que nosotros profundizaremos, y en la cual comenzamos con una pantalla en blanco, a la que irán agregándose diferentes componentes, utilizando el método de *"arrastrar y soltar"*.

Otras dos opciones que Microsoft Power Apps nos ofrece son la creación de *aplicaciones en blanco basada en modelos* (o Dataverse) [8] donde la plataforma contiene una pequeña cantidad de código inteligente y escalable, o la opción de crear un *sitio web de Power Pages*, sin necesidad de codificación, y con las características y aspecto personalizados.

Entorno de trabajo

A continuación, procedemos a acceder la herramienta centrándonos en su entorno, entorno cuyo aspecto podemos observar en la *figura 7.2*.

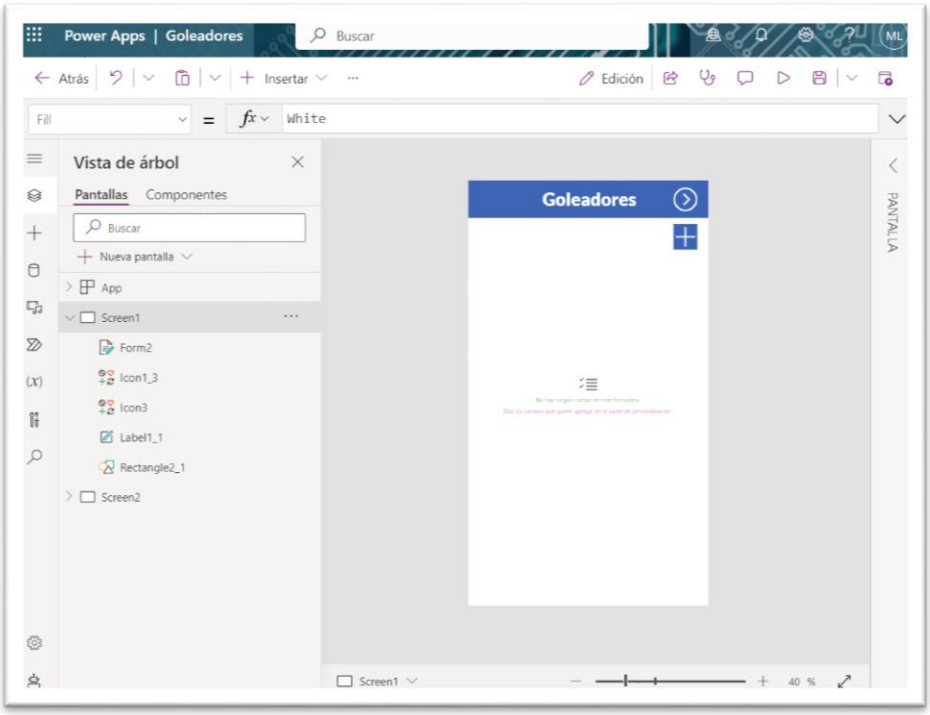


Figura 7.2 Entorno de trabajo de Power Apps

Como podemos ver, se trata de un entorno caracterizado por la clasificación de las diferentes funciones en distintos apartados.

En la *figura 7.3* se muestra el primer apartado denominado **Vista de árbol**; dicho apartado nos mostraría esquemáticamente las diferentes pantallas de las que disponemos. Decidimos cambiar el nombre de las pantallas en función de la tarea que iba a realizar: buscar, mostrar, o editar.

En cada pantalla (BrowseScreen1, DetailScreen1, EditScreen1) se pueden consultar los distintos componentes que se encuentran en su interior. En este apartado, entre otras acciones, podríamos crear o añadir nuevas pantallas, modificar los nombres de tanto pantallas como componentes, u ordenar los diferentes componentes o pantallas en base a cierta jerarquía.

Para aclarar conceptos de cara a los siguientes apartados, es de suma importancia saber que, si estamos hablando de "pantalla", será parecido al término de "pestaña" utilizado por otros programas (abrir nueva pestaña, cerrar pestaña), como pueden ser los navegadores, softwares de programación, etc.

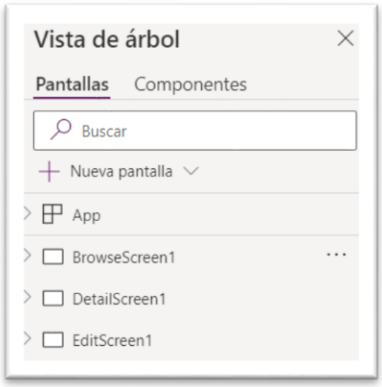
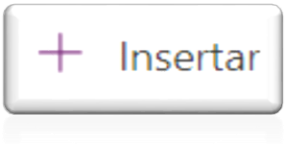


Figura 7.3 Vista de árbol

Como se puede ver en la *figura 7.2*, disponemos de un apartado denominado **Insertar**. En él se dispone de una serie de componentes que se pueden incluir. Desde este mismo apartado se pueden añadir componentes a la pantalla, **arrastrándolos** hacia el lugar donde los queremos incluir y **soltándolos**.

Otro apartado importante es **Datos**, mostrado en la *figura 7.4*. Este apartado nos permite gestionar las diferentes fuentes de datos que poden utilizarse durante el desarrollo de la aplicación. Esas fuentes se pueden obtener mediante las diferentes conexiones que realizamos con los servicios de alojamiento de archivos.



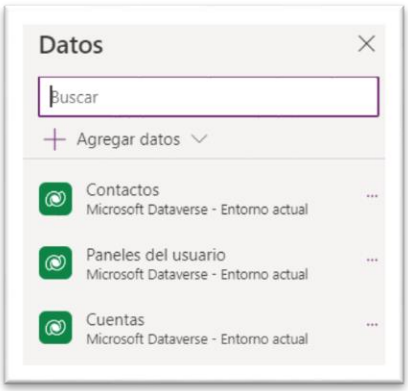


Figura 7.4 Apartado Datos

A continuación, la *figura 7.5* ilustra el apartado **Multimedia**. En él se gestiona el uso de elementos multimedia en nuestra aplicación. Desde esta interfaz se podrían subir los diferentes objetos multimedia a utilizar (imágenes, audio, vídeos, etc.), además de poder ver todos estos elementos mediante una lista, clasificándolos por tipo de elementos.

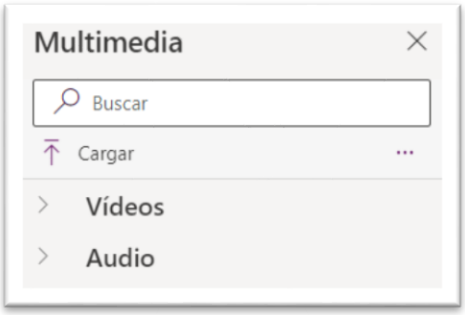


Figura 7.5 Apartado Multimedia

Por último, el apartado **Herramientas avanzadas (***figura 7.6***)**, nos permitirá hacer uso de las funciones más novedosas y vanguardistas de la aplicación. Sin embargo, por este motivo algunas funciones se encuentran todavía en fase experimental. Este apartado no constituye un punto fundamental de la herramienta, se puede desarrollar una aplicación óptima y completa sin necesidad de utilizar ninguna función del mismo.

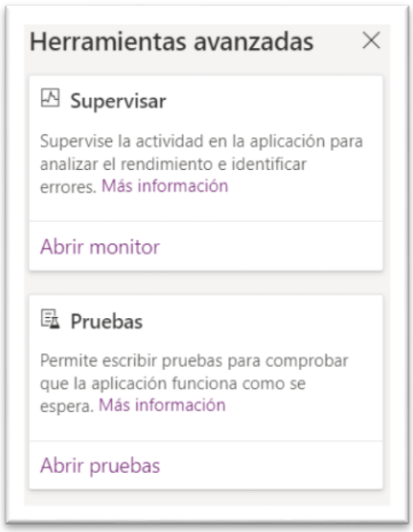


Figura 7.6 Apartado Herramientas avanzadas

Otros dos apartados a mencionar dentro de los servicios que Power Apps ofrece son **Power Automate** y **Variables**. Estos dos apartados son apartados con un nivel mayor de complejidad, y utilizados generalmente en ámbitos empresariales y profesionalaes.

Power Automate (*figura 7.7*) es un servicio complementario a Microsoft Power Apps, y una herramienta conectora de aplicaciones que permite automatizar procesos empresariales, normalmente rutinarios.

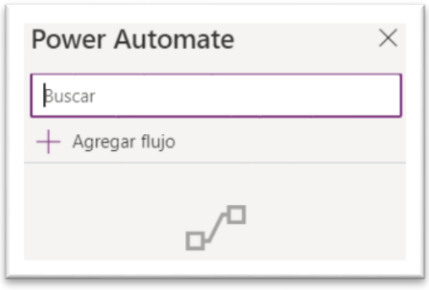


Figura 7.7 Apartado Power Automate

Por otro lado, tenemos el apartado **Variables** (*figura 7.8*) que nos muestra las diferentes variables creadas para nuestra aplicación (globales, locales, contextuales, etc.), así como el valor correspondiente de cada una de ellas.

Figura 7.8 Apartado Variables

Una vez conocidos los diferentes apartados que proporciona el entorno de Microsoft Power Apps, podríamos proceder a insertar los elementos deseados en la pantalla en blanco, para crear y desarrollar nuestra propia aplicación. Estos elementos son variados, incluyendo botones para avanzar o retroceder en la página o aplicación, cuadros de entrada de texto, formularios, galerías con posibilidad de visualizar y/o añadir fotos o vídeos, gráficos, etc.

Después de insertar los elementos a nuestra aplicación, tendríamos la posibilidad de modificar sus propiedades. Este tipo de acciones se pueden realizar desde la interfaz que la plataforma nos proporciona en el momento que un objeto es seleccionado. En la *figura 7.9* se puede ver una interfaz con cada uno de los factores que se pueden modificar de dicho objeto; la personalización de objetos se realiza mediante la parametrización de las variables que te muestran.

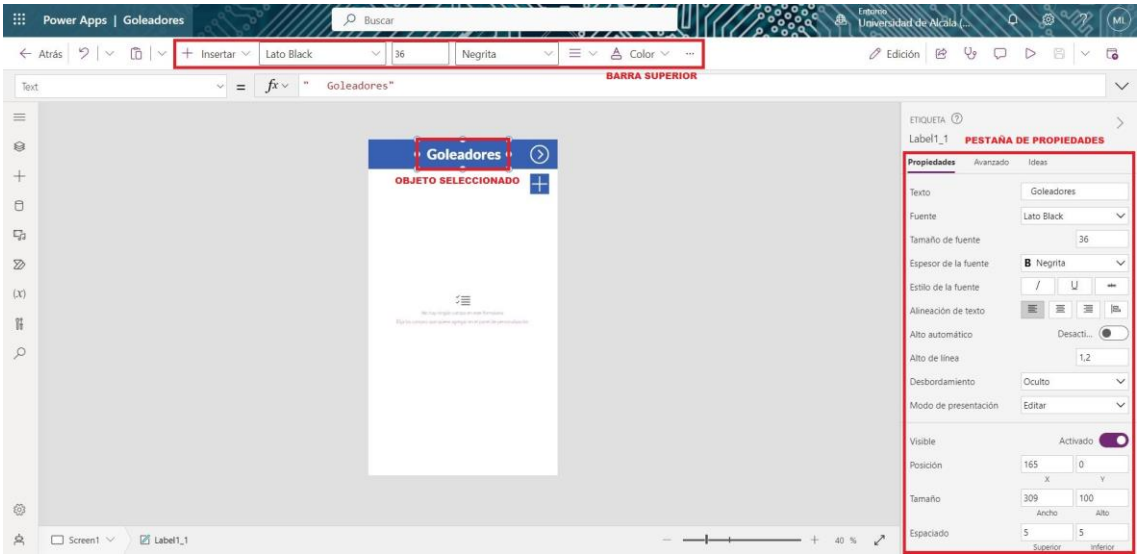


Figura 7.9 Interfaz de trabajo Power Apps (prototipo inicial)

Otras propiedades de los objetos que podemos modificar o añadir, se pueden observar en la barra superior. En ella, disponemos de otros factores que se pueden modificar expuestos en una lista desplegable.

A continuación, si pulsamos uno de los elementos de nuestra aplicación, como en el caso de la *figura 7.10* donde seleccionamos el botón de **Añadir**, podemos observar que se abre y muestra la propiedad de dicho objeto, *"OnSelect"* (en el momento que es pulsado el elemento de la interfaz, se ejecutará aquello que se encuentre en el valor de dicha propiedad).

Esta propiedad puede cambiarse, dependiendo de la acción que deseamos que nuestro elemento realice (en este caso se trata de un botón que al seleccionarse/pulsarse nos permitirá añadir jugadores a la lista). Además, el valor de todas las propiedades puede especificarse en la pestaña de propiedades situada en la derecha de la *figura 5.10.*

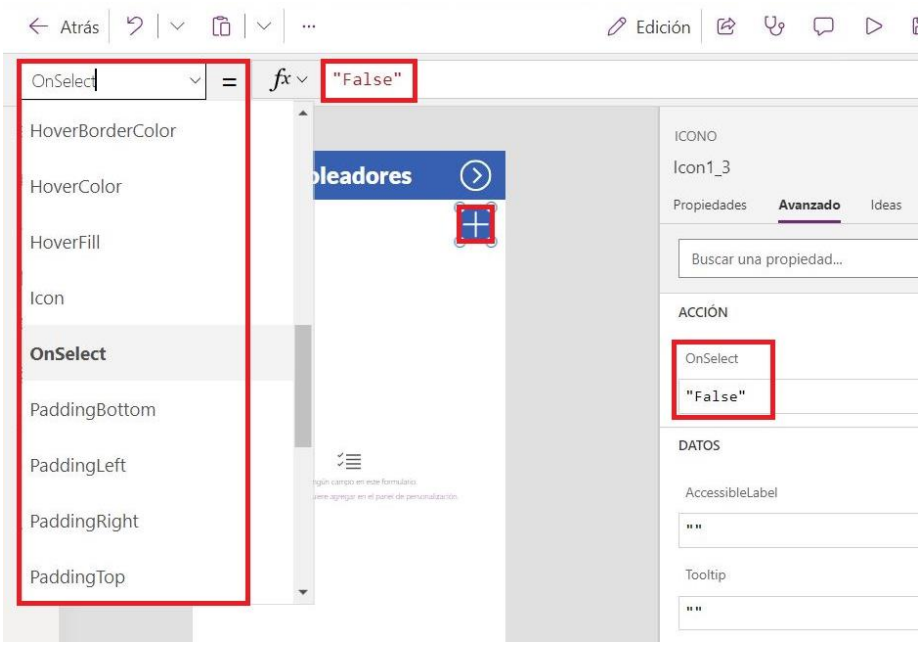


Figura 7.10 Propiedad OnSelect

Actualmente la propiedad "*OnSelect"* se encuentra con valor "false"; esto indicaría que no se debe realizar nada si se pulsa en dicho elemento. De querer incluir alguna funcionalidad, deberíamos añadir una fórmula en dicho campo (recordemos que en nuestro caso se trata de un botón que al ser pulsado abrirá una pestaña nueva que nos permitirá añadir jugadores a la lista).

La fórmula que podemos incluir, la cual se calculará para la propiedad en cuestión, puede estar compuesta por valores, operadores y funciones. Las funciones son muy utilizadas: toman una serie de parámetros, con los cuales realizan una operación y devuelven un valor. Disponemos de una amplia variedad de funciones, todas ellas se pueden consultar desde la documentación oficial de Power Apps.

7.2 Exploración y diseño interactivo

El diseño de interfaces y la navegación entre ellas constituyen uno de los grandes puntos a la hora de desarrollar una aplicación; todo lo que sugiere la estética conlleva un peso especial para el usuario final.

En un mercado saturado de opciones, es fundamental prestar atención a cada detalle, especialmente a aquellos que causan una primera impresión impactante; es decir, lo que percibimos visualmente. La estética juega un papel crucial, especialmente para los usuarios menos inclinados hacia los aspectos técnicos, ya que influye significativamente en su decisión de adoptar un producto.

Esta dimensión distintiva es el motivo detrás de la propuesta de este desafío. Para llevar a cabo una evaluación precisa, se ha optado por enmarcar la discusión en términos de experiencia. A menudo, nos dejamos guiar por las especificaciones técnicas y características de un producto, pero la verdadera diferenciación radica en la vivencia del usuario.

De esta manera, se propone cambiar el prototipo básico inicial de nuestra aplicación, optimizándolo con un menú mucho mejor y más eficiente. Este nuevo diseño incluirá funciones avanzadas, como la capacidad de añadir jugadores y/o modificar la información de cada jugador registrado, así como la opción de eliminar jugadores de la lista, entre otras características mejoradas.

o Menú principal

Para comenzar la optimización de la aplicación, nos centraremos en la primera ventana creada, el menú principal. Creamos un bloque de galería para la primera pantalla de nuestra aplicación, BrowseScreen1, que será el encargado de mostrar los jugadores almacenados en orden alfabético. La información que veremos acerca de cada jugador una vez añadido, será parecida a la observada en la *figura 7.11*: nombre del jugador, equipo en el que habilita, y el número de goles marcados.

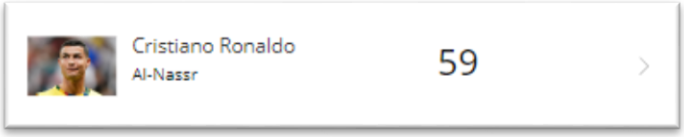


Figura 7.11 Jugador, equipo y nº de goles

A continuación, configuramos el botón “añadir jugador  ” que ya habíamos creado pero sin función alguna hasta el momento, pues la propiedad “OnSelect” de este botón mantiene el valor “False”, establecido por defecto.

Su configuración, como muestra la *figura 7.12*, realizará dos acciones para completar el propósito deseado: si el botón es pulsado, crea una nueva pantalla, cuyo formato será el asociado a la pantalla EditScreen1, que posteriormente configuraremos. En segundo lugar, mediante la función Navigate() el software nos cambiará el entorno de trabajo, realizando la transición de la pantalla inicial, BrowseScreen1, a la pantalla de "Añadir jugador", EditScreen1.

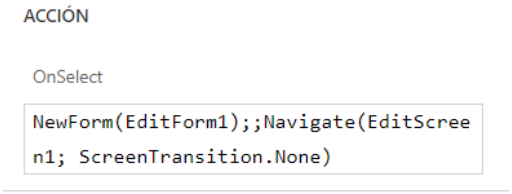


Figura 7.12 Propiedad OnSelect del botón Añadir

Otras 2 funcionalidades fueron añadidas al menú principal: refrescar e invertir orden de elementos. Insertando los 2 botones desde la barra de herramientas, y asignándoles las funciones *Refresh() y UpdateContent(¡SortDescendind),* conseguimos que nuestro menú pudiera refrescar en caso de modificar algún dato de los jugadores, e invertir el orden de los jugadores mostrados respectivamente.

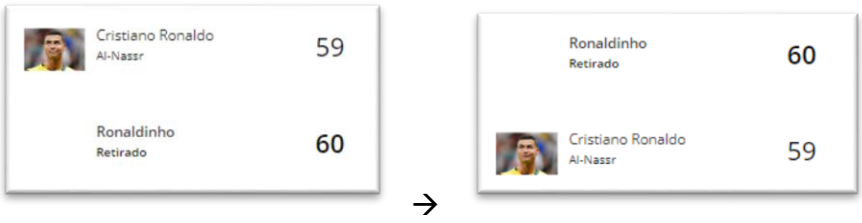
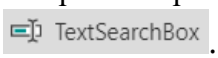


Figura 7.13 Ejemplo de inversión del orden de elementos

Por último, optamos por añadir la funcionalidad de buscar jugador, mediante el componente predeterminado que ofrece la plataforma Power Apps, "TextSearchBox" 

.

o Añadir jugador

Como hemos comentado en el apartado anterior, si pulsamos el botón de "Añadir" se abrirá una nueva pantalla, la pantalla EditScreen1, que será la encargada de mostrar las diferentes casillas que debemos rellenar con los datos correspondientes a los nuevos jugadores que añadimos.

Para crear las casillas de entrada de texto, agregamos en primer lugar un formulario de edición desde la pestaña insertar. A este formulario, añadimos varias herramientas de

entrada de texto, y las asignamos un nombre a cada una: nombre jugador, equipo, dorsal, nº de goles, entre otras.

Una vez creados todos los componentes, se asignará a cada casilla un valor a la propiedad "DisplayName", para poder enlazarlo con las diferentes funciones que han de reflejar los datos, como por ejemplo en la pantalla DetailScreen1 donde deben mostrarse todos los datos registrados. En la siguiente figura mostramos un ejemplo de la asignación del DisplayName para la casilla "Jugador" donde podemos escribir el nombre del jugador a añadir. El resto de casillas seguirán el mismo patrón de creación y asignación.

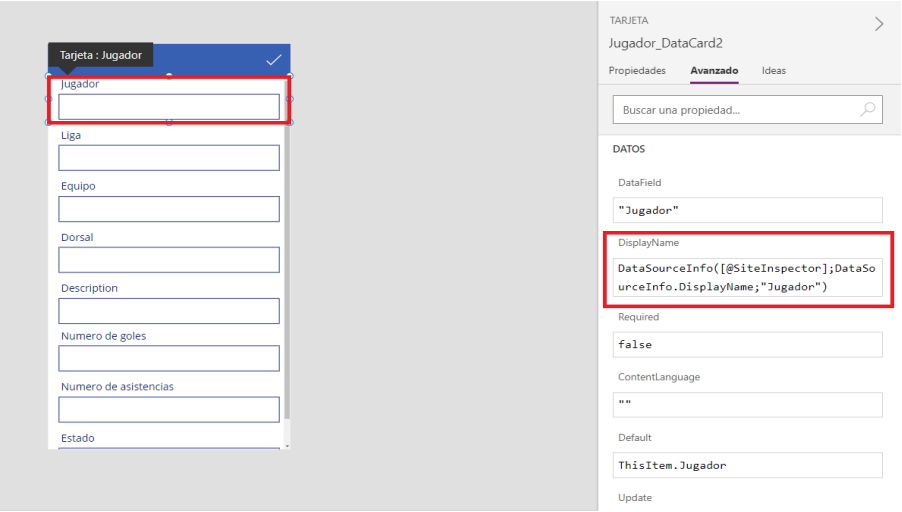


Figura 7.14 Asignación del campo DisplayName

Además, dentro de cada casilla podemos establecer si ha de ser rellenada o no obligatoriamente, poniendo el campo Required a "True".

Después de completar todos los pasos de configuración para el campo "Jugador", procedimos a crear y configurar de la misma manera el resto de campos disponibles en la pantalla "Añadir" (*figura 7.14*).

Por último, nos centramos en los dos botones situados en la parte superior de nuestra aplicación, que son los responsables de "**Confirmar y enviar elemento**" (√) cuando el formulario es rellenado, y/o de "**Cancelar elemento**" (×) en caso oportuno, y retroceder a la pantalla inicial del menú.

En el caso del botón de "Confirmar y enviar elemento" debemos asignarle la función SubmitForm() (*figura 7.15*), de manera que al ser pulsado con un click, y el formulario haya sido rellenado correctamente, debe almacenar la información introducida y añadir el jugador en la lista de jugadores de la pantalla inicial del menú, mostrando correctamente el nombre, el nº de goles y el equipo del jugador.

Figura 7.15 Función confirma elemento

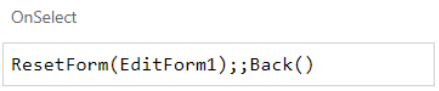


Figura 7.16 Función Cancelar y volver

Por otro lado, como podemos observar en la *figura 7.16*, para configurar la funcionalidad del botón "Cancelar elemento" debemos utilizar la función ResetForm(). Además, para que el botón nos envíe a la pantalla inicial del menú en caso de ser pulsado, debemos añadirle al botón la función Back().

También se puede apreciar en ambas figuras mostradas que, el elemento que debemos asignarle a las funciones debe ser el formulario "EditForm1", ya que se trata del formulario que abarca todos los campos rellenados. De esta manera, reducimos las líneas de código necesarias, y evitamos crear múltiples funciones para pantalla "añadir" cada campo por separado.

○ Mostrar detalles jugador

Una vez añadidos los jugadores deseados y confirmado en la pantalla de "Añadir jugador", volveremos a la pantalla inicial donde nos aparecerá en el menú principal una lista ordenada alfabéticamente con todos los jugadores registrados, como nos mostraba la *figura 7.11*.

Ahora, debemos proceder a configurar la pantalla de visualización, de manera que estando en el menú inicial, si pulsamos con el click encima de cualquier jugador deberá abrir una nueva pantalla con los datos previamente registrados (Jugador, equipo, dorsal, etc). En esta misma pantalla, tendremos en la barra superior 3 botones que nos deberán permitir: volver hacia atrás, editar los datos registrados si lo deseamos, o borrar directamente los datos registrados, y, por ende, el jugador.

En primer lugar, debemos asignar al bloque de Galería del formulario de la pantalla inicial una función específica que realice dicha funcionalidad, es decir, en el momento que un jugador es pulsado deberá abrir una nueva pantalla con todos los detalles sobre el mismo. Para ello, debemos configurar la propiedad OnSelect del bloque, cambiando el "False" por la función Navigate(), que realizará la transición a la pantalla "DetailScreen1". Esta configuración, parecida a la función que aplicamos al botón de añadir jugador (+) para la pantalla EditScreen1, se muestra en la siguiente figura con más detalle.

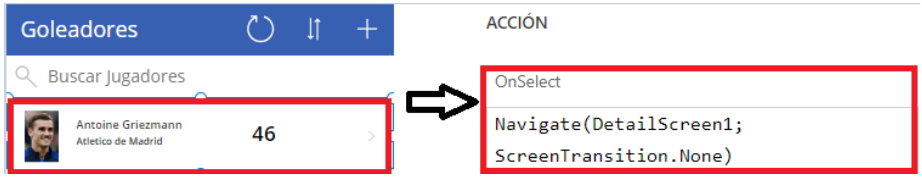


Figura 7.17 Propiedad OnSelect galería pantalla inicial

Después de terminar la configuración anterior, accedemos la pantalla “DetailScreen1” donde la aplicación mostrará la información registrada de cada jugador. Esta pantalla debe mostrar completados de manera correcta todos los campos mencionados y registrados anteriormente.

Considerando este aspecto, debemos insertar los componentes necesarios y realizar las configuraciones oportunas para que los campos mostrados en pantalla sean los mismos que en la pantalla “EditScreen1”, es decir, nombre introducido del jugador, dorsal, liga, etc. En este caso, los componentes que insertaremos serán un formulario de presentación, y varias tarjetas de presentación que mostrarán un campo registrado cada una.

Para configurar cada tarjeta de presentación, debemos cambiar el valor “False” de la propiedad DisplayName igual que hicimos en la pantalla EditScreen1, y como mostramos en la *figura 7.14*. Una vez realizado este paso, deberíamos poder visualizar en pantalla los datos registrados de cada jugador.

Por último, nos centraremos en los 3 botones que mencionamos al principio del apartado y que nos permiten realizar las acciones de volver hacia atrás, editar los datos de jugador, o borrar datos del jugador. En el caso del primero, su configuración es sencilla basándose en la función Navigate() que deberá realizar la transición de escenarios, volviendo a la pantalla “BrowseScreen1”.

OnSelect

```
Navigate(BrowseScreen1; ScreenTransition.None)
```

Figura 7.18 Propiedad OnSelect del botón “Volver atrás”

En segundo caso, la configuración del botón de “Editar datos” también será sencilla, ya que su funcionalidad consiste en realizar la transición de la pantalla “DetailScreen1” a la pantalla “EditScreen1” cada vez que sea pulsado. Para poder realizar esta acción usaremos nuevamente la función Navigate(), además de la función EditForm() para el formulario de la pantalla “EditScreen1”.

Por último, el botón de eliminar jugador será escasamente más complicado de configurar, ya que deberá borrar mediante la función Remove() todos los datos registrados en cada tarjeta de presentación y la foto del jugador almacenada en la galería de la aplicación, y, además, volver mediante la función Back() a la pantalla inicial, BrowseScreen1.

OnSelect

```
Remove([@SiteInspector]; BrowseGallery1.Selected);;
If (IsEmpty(Errors([@SiteInspector];
BrowseGallery1.Selected)); Back())
```

Figura 7.19 Propiedad OnSelect del botón Borrar

- **Resultado final**

En conclusión, presentamos en la figura 7.20 el resultado final obtenido tras crear y configurar nuestra aplicación. Como se puede observar, se muestran las 3 pantallas completas, en formato móvil, con todos los datos mostrados, con los diferentes botones disponibles en cada una de ellas, y los diferentes campos de los formularios que los usuarios podrían pulsar, leer o rellenar.

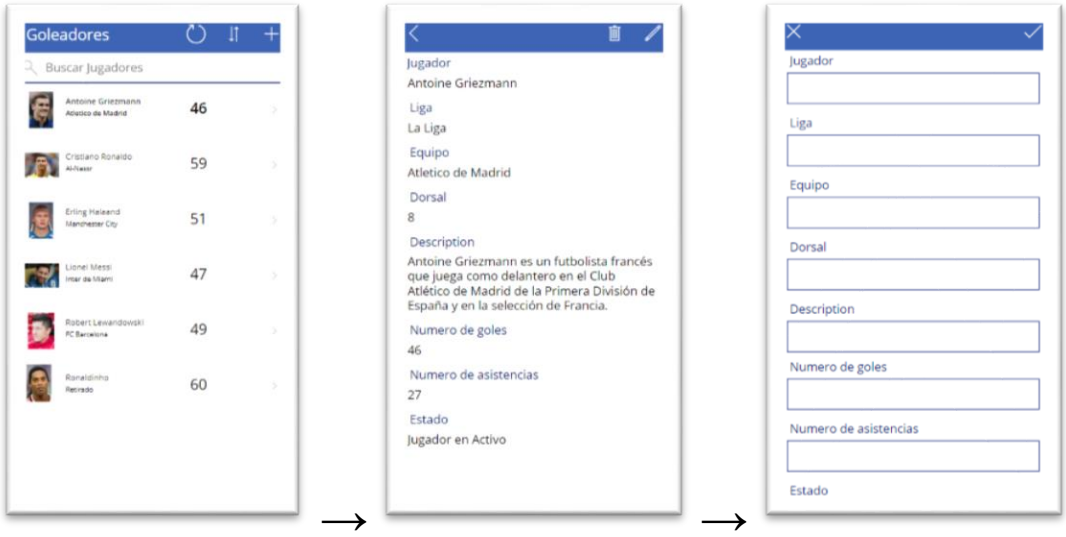


Figura 7.20 simulación de las 3 pantallas de la app

Para terminar, debemos mencionar que, en el caso de todos los componentes de nuestra aplicación, se pueden hacer cambios de aspecto, como: el tamaño de letra, los colores de texto, los colores de las casillas, colores de los bordes o el color de la barra superior de la aplicación; todos estos cambios se realizan desde la pestaña propiedades de cada componente. Para demostrarlo procederemos a cambiar algunos detalles de nuestra aplicación, cambiando el tema y los colores establecidos por defecto.

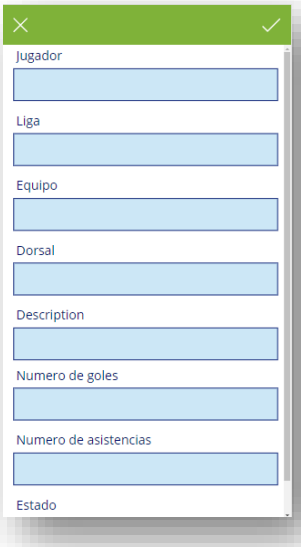


Figura 7.21 Pantalla "Añadir Jugador" con cambios de aspecto

7.3 Integración con Microsoft 365

En el mundo empresarial actual, la capacidad de aprovechar al máximo los recursos y datos disponibles es esencial para lograr una operación eficiente y una toma de decisiones informada. Una de las ventajas destacadas de Microsoft Power Apps es su estrecha integración con diversas herramientas y servicios de la suite Microsoft 365. En particular, la integración con Microsoft Excel se erige como un recurso poderoso que permite a las organizaciones crear aplicaciones personalizadas que se conectan directamente con las hojas de cálculo de Excel. Esta sinergia entre Power Apps y Excel no solo simplifica la experiencia del usuario, sino que también amplifica la capacidad de analizar, manipular y presentar datos en tiempo real.

En este contexto, trataremos de implementar la aplicación a partir de un archivo Excel como ejemplo de prueba, analizando detenidamente cómo esta integración puede mejorar nuestra eficiencia operativa, y explorando cómo transformar los datos almacenados en una hoja de cálculo en una aplicación funcional y altamente personalizada.

En primer lugar, poniendo en práctica lo aprendido en los cursos de inicialización de Microsoft Power Apps. debemos crear un archivo Excel con los datos pertenecientes a cada jugador (nombre, liga, país, dorsal, descripción, etc) separando por columnas cada campo, y después rellenando las columnas con los datos adecuados de cada jugador (*figura 7.22*).

ID	Title	Jugador	Liga	Pais	Descripción2	Numero_goles	Numero_de_asistenci	Partidos_jugado	Equipo	Photo[image]
1	Jugador 1	Cristiano Ronaldo	SPL	Portugal	Cristiano Ronaldo dos Santos Aveiro, conocido como Cristi	59	21	43	Al-Nassr	.\Goleadores\CR7.jpg
2	Jugador 2	Lionel Messi	MLS	Argentina	Lionel Andrés Messi Cuccittini, conocido como Leo Messi,	47	30	43	Inter de Miami	.\Goleadores\Messi.jpg
3	Jugador 3	Erling Halaand	Premier I	Noruega	Erling Braut Haaland es un futbolista noruego que juega c	51	12	50	Manchester City	.\Goleadores\Halaand.jpg
4	Jugador 4	Robert Lewandowski	La Liga	Polonia	Robert Lewandowski es un futbolista polaco que juega cor	49	14	50	FC Barcelona	.\Goleadores\Lewandowski.jpg
5	Jugador 5	Antoine Griezmann	La Liga	Francia	Antoine Griezmann es un futbolista francés que juega con	46	27	48	Atletico de Madrid	.\Goleadores\Griezmann.jpg
6	Jugador 6	Ronaldinho	Retirado	Brasil	Ronaldo de Assis Moreira, conocido como Ronaldinho Ga(	47	29	50	Retirado	.\Goleadores\Ronaldinho.jpg
7	Jugador 7	Maradona	Retirado	Argentina	Diego Armando Maradona fue un futbolista y entrenador	46	21	46	Retirado	.\Goleadores\Maradona.jpg
8	Jugador 8	Sergio Ramos	Ligue 1	España	Sergio Ramos García es un futbolista español que juega c	34	20	48	PSG	.\Goleadores\Ramos.jpg

Figura 7.22 Ejemplo de archivo Excel para integración de la app

En segundo lugar, como explicamos al principio de este capítulo, para iniciar la creación de nuestra aplicación utilizando Microsoft Power Apps podemos utilizar diferentes tipos de archivo como Bases de datos, archivos Excel, listas share point, entre otras. Nosotros optaremos por **empezar a partir de un archivo Excel**.

Después de unos momentos, navegará a la pantalla **Conexiones**, donde debería ver sus conexiones existentes como OneDrive, Dataverse, etc. Si su conexión no aparece inmediatamente, seleccione **Actualizar**, o presione en **Añadir conexión** y seleccione la opción deseada. En nuestro caso, optamos por utilizar OneDrive.

A continuación, entre los archivos mostrados en nuestro OneDrive debemos seleccionar el archivo Excel a partir del cual deseamos crear la aplicación y pulsar el botón **Conectar** en la siguiente ventana que se abra. Como observamos, en nuestro caso el archivo será Goleadores.xlsx (figura 7.23).

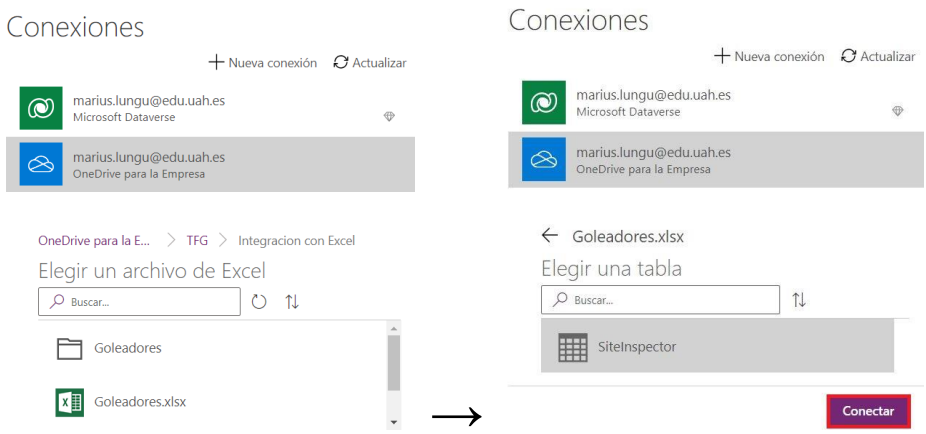


Figura 7.23 Como crear una app a partir de archivo Excel

Por último, después de pulsar en **Conectar,** debemos esperar que la aplicación sea creada e implementada; el tiempo de espera variará en función del número de datos que nuestro archivo Excel tenga. El resultado en este caso será parecido a la aplicación mostrada anteriormente, variando algunos detalles como pueden ser: la colocación de los nombres, número de goles, el tamaño de los caracteres, o el número de jugadores mostrados en la pantalla de inicio, ya que en este caso optamos por añadir dos jugadores más a partir del archivo Excel. Sin embargo, los componentes y el funcionamiento de la aplicación serán los mismos que el prototipo creado anteriormente.

7.4 Conclusión del caso de estudio

A continuación, se presentan las conclusiones extraídas de cada uno de los desafíos enfrentados a lo largo del caso de estudio:

Aspecto	Conclusiones
1. Facilidad de Uso	Microsoft Power Apps ofrece una interfaz intuitiva y amigable que permite a usuarios sin experiencia en programación crear aplicaciones personalizadas de manera sencilla. Su enfoque basado en arrastrar y soltar facilita la creación de flujos de trabajo y formularios.
2. Variedad de Plantillas	La amplia gama de plantillas predefinidas disponibles en Power Apps permite a los usuarios comenzar rápidamente con proyectos específicos. Esto es especialmente beneficioso para aquellos que desean una solución rápida sin la necesidad de construir una aplicación desde cero.
3. Integración con Microsoft 365	La integración profunda con la suite de Microsoft 365 brinda a los usuarios la capacidad de aprovechar datos de aplicaciones como en nuestro caso Excel, o SharePoint y otros servicios mencionados, lo que aumenta la eficiencia y la utilización de recursos ya existentes.
4. Automatización de Procesos	Power Apps permite la automatización de procesos mediante flujos de trabajo y reglas de lógica, lo que simplifica tareas repetitivas y optimiza la productividad. La interconexión con Power Automate amplía estas capacidades, brindando un enfoque integral.
5. Personalización Avanzada	Para usuarios con conocimientos técnicos más profundos, Power Apps ofrece la flexibilidad de personalizar aplicaciones con lenguajes como Power Query y Power BI, lo que permite abordar requisitos específicos y crear soluciones altamente adaptadas.
6. Acceso Móvil y Multiplataforma	La capacidad de implementar aplicaciones en dispositivos móviles y en múltiples plataformas garantiza que los usuarios puedan acceder y colaborar en sus aplicaciones en cualquier momento y lugar, lo que contribuye a la flexibilidad y la accesibilidad.

7. Colaboración en Tiempo Real	Power Apps fomenta la colaboración en tiempo real a través de la coautoría, lo que permite que varios usuarios trabajen juntos en una aplicación simultáneamente. Esto mejora la eficiencia y promueve la creatividad en equipos.
8. Actualizaciones y Mantenimiento Simplificados	La infraestructura en la nube de Power Apps elimina la necesidad de realizar actualizaciones manuales, lo que facilita el mantenimiento y garantiza que las aplicaciones estén siempre actualizadas con las últimas características y mejoras.

En conclusión, Microsoft Power Apps se destaca por su enfoque en la creación de aplicaciones personalizadas de manera intuitiva y efectiva. Su amplia gama de herramientas y facilidades, desde la facilidad de uso hasta la integración con servicios de Microsoft y la capacidad de personalización avanzada, hacen que sea una plataforma poderosa para abordar una variedad de necesidades de negocio y mejorar la productividad en diversos escenarios. Ya sea para automatizar procesos, agilizar flujos de trabajo o crear soluciones específicas, Power Apps ofrece un conjunto completo de capacidades que benefician tanto a usuarios sin experiencia técnica como a profesionales más avanzados.

8. Conclusión final

El logro de los objetivos establecidos al inicio de este trabajo es evidente. Se han investigado y experimentado en primera persona las ventajas y desventajas derivadas de la incorporación de una tecnología Low-code en el ámbito del desarrollo de aplicaciones.

Hablando de la parte teórica, resultó muy enriquecedor y útil el estudio de los artículos y libros encontrados. La investigación y la lectura especializada nos permitió conocer y adentrarnos en el mundo de las soluciones Low-code, siendo este un tema de estudio en pleno desarrollo.

Podemos evidenciar que las soluciones Low-code representan una revolución en el desarrollo de aplicaciones, ofreciendo un enfoque más rápido y eficiente para el desarrollo de aplicaciones, así como la reducción de costos, entre otras. Sin embargo, la personalización limitada, la complejidad de abordar aplicaciones altamente especializadas o la necesidad de mantenimiento constante, son desafíos que hacen de las soluciones Low-code, unas herramientas aún optimizables.

Por su parte, la oportunidad de conocer y conversar con un desarrollador profesional, ayudó a conocer de manera más detallada la visión y perspectiva sobre el Low-code en entornos empresariales. Además, las recomendaciones recibidas facilitaron el estudio de las aplicaciones empleadas durante el trabajo de fin de grado, y la elección final del caso de estudio.

Otro aspecto importante en el estudio fue la realización de la encuesta a múltiples usuarios con diferentes niveles de habilidad y experiencias. Estas opiniones proporcionaron una visión más completa y matizada de cómo las herramientas Low-code se perciben y utilizan en el mundo real. Observar estas perspectivas desde distintos ángulos nos ayudó a comprender de manera más profunda las implicaciones y oportunidades que estas soluciones ofrecen. En este punto es importante entender que, tratándose de nuevas tecnologías, se necesita cierto tiempo para atraer la atención de múltiples tipos de usuarios y convencerlos, pues el temor que muchos contemplan al usarlas sólo puede vencerse si estas herramientas demuestran su facilidad y eficiencia.

Centrándonos en la parte práctica, Microsoft Power Apps, en particular, ha demostrado ser una herramienta de utilidad al reducir los costes temporales del proceso de desarrollo y al proporcionar apoyo para la realización de ideas que, sin su apoyo, habrían quedado inalcanzables. La herramienta también se destaca en la simplificación de tareas, como el diseño de la aplicación. Además, su sólida integración con datos externos y la posibilidad de incluir APIs contribuyen a su atractivo como opción para proyectos futuros.

Dos claros ejemplos son las funcionalidades **Vista de árbol** y **Multimedia**. La primera nos ha permitido visualizar y explorar la estructura jerárquica de todos los elementos presentes en las diferentes pantallas de la aplicación y mantener un óptimo orden de todos sus elementos. Por otro lado, la opción **Multimedia** es particularmente valiosa para la comunicación y el compromiso del usuario, ya que permite crear aplicaciones más ricas

y atractivas. Por ejemplo, cuando creamos el menú de nuestra aplicación, pudimos incluir imágenes de todos los jugadores registrados para facilitar su reconocimiento.

Otro factor favorable fueron los cursos previos que ofrece para aprender su utilización. Estos cursos brindan una curva de aprendizaje suave y eficaz, permitiendo a los usuarios adquirir habilidades en el desarrollo de aplicaciones sin experiencia previa. Esta característica facilita la creación de aplicaciones personalizadas y agiliza la adopción de la plataforma, lo que resulta en una mayor productividad y capacidad para aprovechar al máximo sus capacidades.

Sin embargo, la adopción de estas plataformas Low-code también ha revelado ciertas desventajas. La orientación hacia la experiencia del usuario puede generar desafíos de control, ya que aspectos técnicos se vuelven menos manejables, pues la falta de control sobre el mantenimiento, la seguridad y la eficiencia de la solución puede generar problemas y preocupación para muchas empresas y sus desarrolladores.

Otra desventaja que pudimos apreciar durante el estudio son la rigidez de las herramientas Low-code y su limitada compatibilidad con otras herramientas. La falta de flexibilidad puede ser un obstáculo para lograr hitos más complejos, y la posibilidad de innovación se ve restringida por la limitación de funciones personalizadas. Por su parte, la escasa compatibilidad con herramientas tradicionales también es una limitación que podría abordarse mediante una mejor integración entre ambos enfoques.

Por último, terminando de comentar las principales desventajas que pudimos identificar a la hora de emplear las herramientas Low-code, podríamos incluir la dependencia de las actualizaciones y cambios en la plataforma. En algunos casos, si una plataforma cambia sus características o políticas, los proyectos existentes podrían requerir modificaciones significativas para adaptarse, lo que podría llevar a interrupciones y requerir tiempo adicional para mantener la funcionalidad.

Sin embargo, es muy importante mencionar que, como comentamos en el apartado de **conclusiones del caso de estudio**, Power Apps es un ejemplo de aplicación que evita este problema eliminando toda necesidad de realizar actualizaciones manuales, y facilitando el mantenimiento de las aplicaciones.

A continuación, dado este análisis, surgen varias preguntas, ¿dónde sería adecuado emplear tecnologías de este tipo? y ¿podrían reemplazar a las soluciones tradicionales?

En mi opinión, aunque es un desafío sustituir completamente las herramientas tradicionales, las tecnologías Low-code pueden ofrecer complementariedad, es decir, podrían ofrecer la posibilidad de combinar estas tecnologías con enfoques tradicionales para agilizar el proceso de desarrollo en ciertas fases y reducir la carga de trabajo para el desarrollador.

A modo de ejemplo, se podrían utilizar plataformas Low-code para crear prototipos rápidos de aplicaciones o características. Esto permitiría una iteración muy ágil y la validación temprana de conceptos. Una vez que el prototipo sea validado, se podría profundizar utilizando métodos de programación tradicional para optimizar el rendimiento y la personalización.

También se podría emplear en casos de personalización altamente avanzada; si una aplicación necesita características complejas que van más allá de lo que una plataforma low-code permite, la programación tradicional posibilitará el desarrollo de soluciones a medida. Un ejemplo sería optimizar la aplicación desarrollada durante este trabajo mediante métodos de programación orientada a objetos estudiados durante la carrera.

Otro escenario donde la complementación entre la programación tradicional y las herramientas Low-code podría facilitar nuestro trabajo sería el desarrollo de interfaces de usuario. Generalmente, las soluciones Low-code son muy eficientes para crear interfaces de usuario simples y funcionales. Sin embargo, si nuestra aplicación requiere interfaces más complejas o personalizadas que van más allá de las capacidades visuales de la plataforma, se podría complementar con programación tradicional.

Para concluir, personalmente siento un alto nivel de satisfacción respecto a la realización de este Trabajo de Fin de Grado, por haberme brindado la oportunidad de explorar y adquirir conocimientos en relación con las soluciones Low-code.

Además, siendo un tema de estudio poco exprimido todavía, tras experimentar con diferentes herramientas quedé profundamente sorprendido y cautivado por la amplia gama de posibilidades que estas ofrecen. La capacidad interactiva de crear aplicaciones sin requerir habilidades de programación y las facilidades ofrecidas para reducir los tiempos de trabajo me han dejado impresionado.

Por este motivo, considero que esta experiencia podría influir en mi enfoque hacia futuros proyectos, considerando en caso necesario la inclusión de estas herramientas como alternativa de trabajo.

Anexos

Anexo I. Entrevista al desarrollador profesional de Webem

1) ¿Qué opinión tienes sobre las soluciones Low-code en comparación con los métodos de programación convencionales?

Las soluciones Low-code son una solución muy óptima para aprender los conceptos básicos de programación. Por ejemplo, su uso resulta muy óptimo si vas a programar pocas veces en la vida (programar la web de tu empresa), o no vas a programar funcionalidades muy complejas, ya que estas soluciones Low-code no te permiten una amplia personalización que sí permiten las soluciones de programación clásicas.

2) ¿Qué beneficios percibes al utilizar soluciones Low-code en comparación con enfoques tradicionales de desarrollo de software? (agilidad, aceleración de tiempo, eficiencia, menos carga de trabajo)

Creo que utilizar Low-code es un gran avance para pequeñas empresas y autónomos, ya que estas soluciones les permiten hacer sus propios programas a su gusto, ahorrando dinero al evitar contratar personas externas. Además, el nivel necesario para utilizar estas plataformas suele ser nulo o bajo; no tienes que tener conocimientos previos de programación. Solo necesitas saber utilizar la solución elegida, y tener claro lo que quieres hacer.

3) ¿Has notado alguna diferencia en la calidad o robustez del software desarrollado con herramientas Low-code en comparación con métodos convencionales?

En cuanto a la calidad y robustez de estas herramientas, creo que pueden ser más seguras y robustas que métodos tradicionales, ya que los bloques de código que se pueden utilizar para crear programas son limitados y están bien testeados.

4) ¿Qué limitaciones o desafíos identificados en el uso de soluciones Low-code? ¿Has enfrentado alguno de estos desafíos tú mismo?

Los mayores desafíos a los que nos enfrentamos en el uso de estas soluciones son, en un primer momento, aprender cómo funcionan, y una vez aprendido, ver las posibilidades que tiene, ya que no todas tienen gran capacidad de personalización. Por ejemplo, creando una página web con una solución Low-code, pude descubrir que la personalización más allá de las plantillas y los bloques que tenían definido, era muy escasa, al no proporcionar todas las posibilidades o componentes que yo quería utilizar.

5) ¿En qué áreas o casos de uso has encontrado más éxito al utilizar soluciones low-code?

Donde más considero que podríamos sacar partido a las soluciones Low-code, es para enseñar conocimientos básicos de programación en los colegios, o pequeñas empresas y autónomos que necesiten crear algún programa y no tengan la capacidad de contratar desarrolladores externos para que le hagan el programa. Por ejemplo, en bastantes colegios, se está utilizando Lego Technic para enseñar conocimientos básicos de programación y robótica a los niños de primaria.

6) ¿Cómo ves el futuro de las soluciones Low-code en la industria del desarrollo de software? ¿Consideras que las soluciones Low-code pueden reemplazar por completo los métodos de programación convencionales en ciertos casos?

Respondiendo a tu primera pregunta, cada vez se están desarrollando más herramientas Low-code, más potentes y más completas. Por este motivo, en un futuro, cabe la posibilidad de que la mayoría de las empresas utilicen estas soluciones.

Respecto a la segunda pregunta, no creo que puedan reemplazar por completo los lenguajes convencionales. En primer lugar, debido a la falta de flexibilidad en proyectos complejos, y las necesidades de rendimiento avanzadas. Además, otro argumento importante es, que de alguna manera se tienen que programar estas soluciones Low-code, y sin programación convencional sería imposible.

Anexo II. Preguntas encuesta (capítulo 5)

Primera sección: general

1. ¿Tienes experiencia en el desarrollo de software o programación?
   a) Si.
   b) Tengo algo de experiencia, pero no considero que sea extensa o avanzada.
   c) No.

2. ¿Qué nivel de conocimientos consideras que tienes acerca del mundo de la informática y la programación de aplicaciones?
   a) Nivel avanzado (experto con habilidades en áreas de desarrollo de software, ciberseguridad, inteligencia artificial o administración de sistemas)
   b) Nivel intermedio (habilidades en programación muy básica, configuración de redes y manejo básico de sistemas operativos)
   c) Nivel básico (correo electrónico, navegación web, juegos, etc.)

3. ¿Has sentido alguna vez la necesidad/curiosidad de crear una aplicación o software personalizado para tus necesidades?
   a) Sí, con frecuencia.
   b) Sí, en alguna ocasión.
   c) No, nunca.

4. ¿Te resulta intimidante o complicado aprender a programar para crear tus propias aplicaciones?
   a) Sí.
   b) Algunas veces encuentro cierta dificultad, pero estoy dispuesto/a a aprender y superar los desafíos.
   c) No.

5. ¿Cuáles son los principales desafíos que enfrentas al intentar desarrollar aplicaciones sin conocimientos profundos de programación?
   a) Falta de conocimientos técnicos
   b) Complejidad del proceso de desarrollo
   c) Limitaciones de tiempo
   d) Otro (escribir)

6. ¿Qué tipo de aplicaciones te gustaría poder crear para mejorar tu trabajo o vida personal?
   a) Aplicaciones para el seguimiento y control de gastos.
   b) Aplicaciones de organización, recordatorios y productividad personal.
   c) Aplicaciones para el seguimiento de salud y estado físico.
   d) Aplicaciones para la planificación y reserva de actividades de ocio.
   e) Aplicaciones para la automatización de tareas repetitivas y proyectos.
   f) Otro.

7. ¿Cómo de importante te parecería poder desarrollar aplicaciones de manera fácil, rápida y eficiente para tus necesidades?
   a) Muy importante.
   b) Es importante, pero no es una prioridad absoluta.
   c) Poco importante.

8. En cuanto a la facilidad de uso de una plataforma de desarrollo de software, ¿Cuáles serían tus expectativas?
   a) Me gustaría que fuera muy fácil de usar, incluso para principiantes.
   b) Me gustaría que la plataforma ofrezca asistentes o guías paso a paso para ayudarme en el proceso de desarrollo.

c) Valoraría que la plataforma tenga una amplia documentación y recursos de apoyo para resolver cualquier duda o problema.
d) Me gustaría tener acceso a plantillas y componentes predefinidos para acelerar el proceso de desarrollo de aplicaciones.
e) Espero que la plataforma proporcione opciones de personalización para adaptar las aplicaciones a mis necesidades específicas.
f) Otro.

En este punto de la encuesta optamos por introducir a los encuestados unas diapositivas breves resumiendo el concepto de Low-code y las facilidades que esta herramienta podría ofrecer.

9. Llegados a este punto entonces, ¿Cuál sería tu nivel de interés en utilizar una solución/herramienta low-code para desarrollar tus propias aplicaciones sin conocimientos profundos de programación?
   a) Muy interesado/a
   b) Interesado/a
   c) Poco interesado/a
   d) Nada interesado/a (Termina la encuesta)

Segunda sección: sólo siguen los interesados sobre el Low-Code.

10. ¿Estarías dispuesto/a a invertir tiempo en aprender a utilizar una herramienta de desarrollo low-code?
    a) Sí.
    b) No.

11. ¿Te gustaría recibir capacitación o recursos para aprender a utilizar una plataforma de desarrollo low-code?
    a) Sí, me gustaría tener acceso a cursos o tutoriales para aprender a utilizarla.
    b) No, preferiría explorarla por mí mismo/a sin ningún recurso adicional.

12. ¿Qué tipo de soporte o asistencia te gustaría recibir al utilizar una plataforma low-code? Se contesta en la casilla.

13. En caso de ser un usuario con nivel de conocimientos avanzado o intermedio:
    a) Sí.
    b) No. (Termina la encuesta)

Tercera sección: en esta sección, sólo nos centramos en los usuarios que tienen cierto nivel de conocimientos sobre programación e informática, es decir, los que marcaran en la pregunta 13 la casilla **Sí**.

14. ¿Cuáles crees que podrían ser las ventajas de utilizar soluciones low-code en comparación con otros métodos de desarrollo de software?
    a) Mayor rapidez en el desarrollo de aplicaciones.
    b) Menor dependencia de programadores expertos.
    c) Mayor flexibilidad y adaptabilidad a diferentes tipos de usuarios.
    d) Otro.

15. ¿Qué te preocupa más al considerar el uso de soluciones low-code?
    a) Facilidad de arrastrar y soltar elementos para crear interfaces.
    b) Capacidades de integración con sistemas y bases de datos existentes.
    c) Generación automática de código para tareas comunes.
    d) Soporte para colaboración en equipo.

16. ¿Qué te preocupa más al considerar el uso de soluciones low-code?
    a) Limitaciones en la personalización y adaptabilidad de las aplicaciones.
    b) Posibles problemas de escalabilidad y rendimiento.
    c) Dependencia de un proveedor específico o plataforma.
    d) Seguridad y protección de datos.
    e) Otro.

17. ¿Consideras que las soluciones low-code podrían mejorar tu eficiencia y productividad en el desarrollo de aplicaciones?
    a) Sí.
    b) No estoy seguro/a, preferiría explorar más sobre el tema.

18. ¿Estarías dispuesto a sustituir los lenguajes de programación convencionales por las herramientas low-code?
    a) Sí.
    b) Depende del proyecto o aplicación específica, evaluaría las ventajas y desventajas antes de tomar una decisión.
    c) No, prefiero seguir utilizando los lenguajes de programación convencionales.

19. Por último, ¿Qué opinas sobre la posibilidad de poder compartir y colaborar en proyectos de desarrollo de software utilizando una plataforma low-code?
    a) Me parece una excelente oportunidad para fomentar la colaboración y acelerar los proyectos.
    b) No estoy convencido/a de los beneficios de la colaboración en este contexto.

Bibliografía

[1] https://en.wikipedia.org/wiki/Low-code_development_platform

[2] Bucaioni, A., Cicchetti, A. & Ciccozzi, F. Modelling in low-code development: a multi-vocal systematic review. Softw Syst Model 21, 1959–1981 (2022). https://doi.org/10.1007/s10270-021-00964-0

[3] Engineering (Beijing, China), 2020, Vol.6 (9), p.960-961: https://www.sciencedirect.com/science/article/pii/S2095809920301843

[4] I.G (2022, 29 septiembre) Las mejores plataformas low code para crear aplicaciones https://www.iebschool.com/blog/las-mejores-plataformas-low-code-para-crear-aplicaciones-tecnologia/ Consultado 14 de Julio de 2023

[5] Meijers, J. (2018, 2 mayo). The difference between low-code and no-code platforms. Triggre. https://triggre.com/the-difference-between-low-code-and-no-codeplatforms/ Consultado el 10 de febrero de 2021.

[6] Alores. (2021, 16 febrero). ¿Por qué el movimiento no-code / low-code beneficia a los programadores profesionales? Velneo. https://velneo.es/por-que-el-movimientono-code-low-code-beneficia-a-los-programadores-profesionales/ Consultado el 21 de febrero de 2021.

[7] El movimiento no-code. La democratización del software. (2020, 19 junio). ABANCA innova. http://abancainnova.com/es/opinion/el-movimiento-no-code-lademocratizacion-del-software/ Consultado el 10 de febrero de 2021.

[8] Pons, C., Giandini, R. S., & Pérez, G. (2010). Desarrollo de software dirigido por modelos. Editorial de la Universidad de la Plata. Consultado el 25 de marzo de 2021.

[9] Martin, James (1990). MacMillan Publishing Co., ed. Rapid Application Development.[1] Archivado el 22 de abril de 2013 en Wayback Machine.

[10] Maurer and S. Martel. (2002). "Extreme Programming: Rapid Development for Web-Based Applications". IEEE Internet Computing, 6(1) pp 86-91 Enero/Febrero 2002.

[11] Hildenbrand, T., & Seyff, N. (2019). Low-Code Development Platforms: A Systematic Literature Review. In Proceedings of the 2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice Track (pp. 150-159).

[12] Bellmann, L., & Matthes, F. (2020). Low-Code Platforms: A Systematic Mapping Study. In Proceedings of the 2020 IEEE/ACM 42nd International Conference on Software Engineering: Software Engineering in Practice Track (pp. 367-376).

[13] Mendonca, N., Filho, F. S., & Neto, M. R. (2019). Low-Code Development Platforms: A Systematic Mapping Study. In Proceedings of the 2019 ACM/IEEE

International Symposium on Empirical Software Engineering and Measurement (pp. 1-8).

[14] Launonen, P., & Itkonen, J. (2018). Exploring Low-Code Platforms: A Systematic Mapping Study. In Proceedings of the 2018 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (pp. 29:1-29:6).

[15] Forrester Research. (2020). The Forrester Wave™: Low-Code Development Platforms For AD&D Professionals, Q1 2021. Retrieved from https://www.forrester.com/report/The+Forrester+Wave+LowCode+Development+Platforms+For+ADD+Professionals+Q1+2021/-/E-RES143687

[11] Hildenbrand, T., & Seyff, N. (2019). Low-Code Development Platforms: A Systematic Literature Review. In Proceedings of the 2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice Track (pp. 150-159).

[16] Da Silva, F. S., Schwertner Charão, A., & Alves, C. (2021). Low-Code Development: An Integrative Literature Review. IEEE Access, 9, 13153-13167.

[17] Ernst, N. A., & Schneider, K. (2018). Low-Code Development: A State-of-the-Art Survey. In Proceedings of the 2018 IEEE International Conference on Software Architecture Companion (pp. 205-209).

[18] Klaebe, P., Heß, S., & Satzger, G. (2021). Low-Code Platforms: A Systematic Literature Review. In Proceedings of the 2021 IEEE 16th International Conference on Global Software Engineering (ICGSE) (pp. 73-82).

[19] DeLoach, J., & Magennis, D. (2018). The impact of low-code development on the software development lifecycle. Cutter Business Technology Journal, 31(11), 43-48.

[20] Sutherland, J., & Reinertsen, D. G. (2019). Escaping the planning fallacy: How to build predictable delivery into your software development project. Harvard Business Review, 97(3), 88-97.

[21] K. (2021, 25 mayo). ¿Qué es Power Apps? - Power Apps. Microsoft Docs. https://docs.microsoft.com/es-es/powerapps/powerapps-overview Consultado el 18 de agosto de 2023

[22] Peña, A. (2021, 8 abril). Qué es Outsystems y para qué se utiliza. Incentro. https://www.incentro.com/es-es/blog/stories/que-es-outsystems-para-que-sirve/ Consultado el 14 de agosto de 2023.

[23] M. (2020, 13 de marzo). ¿Qué es Lego Mindstorms?. https://www.thegreenmonkey.es/sarria/que-es-lego-mindstorms/ Consultado el 22 de agosto de 2023

[24] J.M (January 19th 2022) Rapid Application Development with OutSystems https://learning.oreilly.com/library/view/rapid-application-development/9781800208759/B16071_TOC_Final_SK_ePub.xhtml Consultado 18 agosto de 2023

[25] (2022, 12 de agosto) ¿Que es Canva? | Diseños profesionales en minutos https://www.thepowermba.com/es/blog/que-es-canva-y-como-usarlo-para-crear-disenos-profesionales Consultado el 24 de agosto de 2023

[26] A. (2023, 18 de mayo) Power Apps, la revolución low-code en el mundo empresarial https://blog.aitana.es/2023/05/18/power-apps-la-revolucion-low-code-en-el-mundo-empresarial/ Consultado el 18 de agosto de 2023

[27] A. (2021, 6 de septiembre) Una breve historia del desarrollo low-code https://www.velneo.com/blog/una-breve-historia-del-desarrollo-low-code Consultado el 2 de junio de 2023

Universidad de Alcalá
Escuela Politécnica Superior

Universidad
de Alcalá