

TRABAJO FIN DE GRADO

**Desarrollo de aplicaciones multiplataforma con
Outsystems**

AUTOR: VÍCTOR ALDAMA MESADO
TUTOR: ALBERTO ALBIOL COLOMER

Trabajo Fin de Grado presentado en la Escuela Técnica Superior de Ingenieros de Telecomunicación de la Universidad Politécnica de Valencia, para la obtención del Título de Graduado en Ingeniería de Tecnologías y Servicios de Telecomunicación

Curso 2018-19

Valencia, 10 de septiembre de 2019

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universidad Politécnica de Valencia
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

**VLC/
CAMPUS**
VALENCIA, INTERNATIONAL
CAMPUS OF EXCELLENCE



Agradecimientos

Desde aquí dar las gracias a mi tutor por sus consejos, recomendaciones y la amabilidad mostrada durante el desarrollo de este proyecto.

También agradecer en especial a mi familia y seres queridos por el apoyo en los momentos duros y de baja motivación tanto durante el desarrollo de este proyecto como durante los años del grado.

Resumen

Este trabajo comprende el aprendizaje y estudio de una de las nuevas apuestas en el mundo del software, el Low-Code. En concreto, se estudiará la solución aportada por OutSystems[®], empleando su herramienta de trabajo, Service Studio para el desarrollo acelerado de aplicaciones y su herramienta de extensiones Integration Studio, empleada para extender las funcionalidades de Service Studio con funciones personalizadas y bases de datos externas. Para mostrar la potencia de la herramienta y su funcionamiento se desarrollará una aplicación web con una reducida versión móvil para así a su vez mostrar las facilidades que proporciona OutSystems[®] para el desarrollo multiplataforma y la facilidad de escalabilidad que nos ofrece.

Este proceso ha implicado el aprendizaje desde cero de lenguajes como HTML, CSS o C# y ha requerido de conocimientos de SQL y manejo de bases de datos, conceptos de UX/UI e indirectamente el uso de Python. Se trata de una herramienta donde el desarrollador desempeña una función full-stack, es decir, se encarga del desarrollo del back-end y el front-end por igual.

Resum

Aquest treball compren l'aprenentatge i l'estudi de una de les noves apostes en el món del software, el Low-Code. En concret, s'estudiarà la solució que ens ofereix OutSystems[®], empleant les seues ferramentes de treball, com Service Studio, per a desenvolupar aplicacions de forma accelerada, o Integration Studio, per a la construcció de extensions a les funcionalitats de la primera. Per poder mostrar el potencial de aquesta tecnologia i el seu funcionament, es desenvoluparà un aplicatiu web amb una xicoteta versió mòbil, per poder mostrar també les capacitats de desenvolupament multiplataforma i la facilitat de escalabilitat oferides per OutSystems[®].

Este projecte, ha implicat l'aprenentatge de dede zero de llenguatges com HTML, CSS o C# i ha necessitat de coneixements de SQL i bases de dades, conceptes de UX/UI, inclús indirectament de l'us de Python. Es tracta de una ferramenta on el desenvolupador exerceix una funció full-stack, es a dir, s'encarrega tant del front-end com del back-end.

Abstract

This work includes learning and studying one of the new bets in the world of software, the Low-Code. Specifically, we will study the solution provided by OutSystems[®], using its work tool, Service Studio, for accelerated application development and its Integration Studio extension tool, used to extend the functions of Service Studio with custom functions and external databases. To show the power of the tool and its operation, a web application with a reduced mobile version will be developed to show the facilities provided by OutSystems[®] for cross-platform development and the ease of scalability it offers us.

This process has involved learning from scratch languages such as HTML, CSS or C# and has required knowledge of SQL and database management, UX / UI concepts and indirectly the use of Python. It is a tool where the developer plays a full-stack function, that is, he is responsible for the development of back-end and front-end alike.

Índice general

0.1. Introducción	1
0.2. Objetivos	2
1. El Low-Code	3
1.1. El Low-Code	3
1.1.1. Crecimiento e implicación en la transformación digital	3
1.1.2. Principales Puntos Fuertes	4
1.1.3. Ejemplos de Herramientas Low-Code	5
1.2. Low-Code VS No-Code	6
2. OutSystems®	7
2.1. Historia y Contexto	7
2.2. Por qué OutSystems® ?	10
3. Las Herramientas de OutSystems®	11
3.1. Service Studio	14
3.1.1. Galería de Aplicaciones	14
3.1.2. La Distribución del “WorkSpace”	15
3.1.3. “The Forge”	17
3.2. Integration Studio	18
3.3. Service Center	19
3.4. Life Time	20
4. La Aplicación “Easy Sports”	21

4.1. Idea y Objetivo	21
4.2. Modelo de Datos	22
4.3. Aplicación Web	27
4.3.1. Alcance	27
4.3.2. La Web en detalle	28
4.4. Aplicación Móvil	48
4.4.1. Alcance	48
4.4.2. La App Movil en detalle	49
5. Conclusiones y líneas futuras	54
5.1. Conclusiones	54
5.2. Líneas futuras	55
A.	56
A.1. Algoritmo de Cruces	56

Índice de figuras

1.1. Gartner 2019 Magic Quadrant for Enterprise Low Code Application Platforms . . .	6
2.1. “Forrester Wave: Low-Code Development Platforms For application development and delivery Pros, Q4 2017”	8
2.2. “Jul 2018 Gartner Magic Quadrant for Mobile App Development Platforms” . . .	9
2.3. “2018 Gartner Magic Quadrant for Enterprise High-Productivity Application Platform as a Service”	9
3.1. Estructura Básica del Servidor de Outsystems	11
3.2. Estructura ejemplo de entorno con licencia	12
3.3. Proceso de desarrollo tipo con OutSystems®	13
3.4. Galería de Aplicaciones Service Studio	14
3.5. Modulos de las Aplicaciones en Service Studio	15
3.6. Generacion Nativa de Aplicaciones en Service Studio	15
3.7. Layout del editor de Service Studio	16
3.8. Detalles del Area de Edición Service Studio	16
3.9. “The Forge” en Service Studio	17
3.10. Layout de Integration Studio	18
3.11. Layout de Service Center	19
3.12. Layout de Life Time	20
4.1. Modelo de datos de “Easy Sports”	22
4.2. Pantalla Principal	28
4.3. Pantalla Login	29

4.4. Pantalla Selección de Rol	29
4.5. Opciones de manager	30
4.6. Paso 1 Formulario Creación	31
4.7. Paso 2 Formulario Creación	32
4.8. Paso 3.a Formulario Creación	33
4.9. Paso 3.b Formulario Creación	33
4.10. Paso 3.c Formulario Creación	34
4.11. Paso 3.d Formulario Creación	34
4.12. Paso 4 Formulario Creación	35
4.13. Paso 5.a Formulario Creación	35
4.14. Paso 5.b Formulario Creación	36
4.15. Paso 5.c Formulario Creación	36
4.16. Paso 6 Formulario Creación	37
4.17. Parte de la Logica de Validación	38
4.18. Parte de la Logica de Creación	38
4.19. Pantalla de Administración	39
4.20. Pantalla de Validación	40
4.21. Listado para la validación de equipos	41
4.22. Configuración de los enfrentamientos	41
4.23. Opciones de competición	42
4.24. Clasificación	43
4.25. Timeline	43
4.26. Edición de los resultados	44
4.27. Opciones de Jugador	44
4.28. Pantalla de inscripción	45
4.29. Formulario pantalla de inscripción	45
4.30. Pantalla “Your Competitions”	46

4.31. Competiciones Públicas	47
4.32. Datos de Usuario	47
4.33. Login Movil	49
4.34. Listados Movil	50
4.35. Detalles de Competición	50
4.36. Detalle de Resultado	51
4.37. Pantalla de inscripción Movil	51
4.38. Formulario de inscripción Movil	52
4.39. Menu Lateral	53
A.1. Parte 1Codigo Python	57
A.2. Parte 2Codigo Python	57
A.3. Parte 2Codigo C#	58
A.4. Parte 2Codigo C#	59
A.5. Parte 3Codigo C#	60
A.6. Implementación en Service Studio	61

0.1. Introducción

Vivimos en una sociedad con un ritmo frenético, siempre conectada, donde cada vez recibimos y producimos más información y donde las tendencias varían y se transforman continuamente. Hoy en día todo negocio que pretenda alcanzar o mantener un estado de éxito debe ser capaz de adaptarse a este ritmo de transformación digital.

En este ambiente las aplicaciones quedan desfasadas o pierden el foco e incluso el propósito de su utilidad a velocidades vertiginosas. Por ello, una aplicación que pretende cubrir una necesidad o ofrecer un servicio puede perder su interés o la rentabilidad esperada de ella antes incluso de terminar su desarrollo si este se prolonga demasiado o no es capaz de adaptarse a cambios durante el proceso.

Es aquí, donde las tecnologías Low-Code establecen su punto fuerte, proporcionando herramientas para la aceleración de los desarrollos, escalabilidad, productividad, etc. Permitiendo a las empresas adaptarse a los cambios y necesidades con mayor agilidad

En este proyecto hablaremos de las ventajas de las tecnologías Low-Code, en concreto de OutSystems® y presentaremos un ejemplo de desarrollo en detalle, una aplicación para la gestión de competiciones deportivas. También veremos, cómo esta tecnología se adapta a la necesidad de opciones de movilidad y de las capacidades de personalización y extensibilidad adaptando algoritmos personalizados tanto con código externo, como empleando las funciones propias de Service Studio.

0.2. Objetivos

El objetivo principal de este Trabajo Fin de Grado (TFG) es presentar las tecnologías Low-Code, en concreto OutSystems[®], mostrando sus ventajas y como estas afectan al proceso de desarrollo de aplicativos frente a un proceso de desarrollo tradicional.

Se construirá una aplicación de gestión de competiciones deportivas, orientada a gestionar ligas con diferentes niveles de seriedad de forma sencilla y que facilite el proceso de inscripción y seguimiento.

Tambien se pretende incluir funcionalidades fuera de las posibilidades por defecto de OutSystems[®], con el objetivo de mostrar la capacidad de expansión y personalización que se puede alcanzar al usar Integration Studio.

Capítulo 1

El Low-Code

1.1. El Low-Code

Una plataforma de desarrollo Low-Code (LCDP)¹ se puede definir como un software el cual proporciona un entorno que los programadores pueden usar para crear software de aplicación² a través de interfaces gráficas y configuración en lugar de la programación tradicional.[1]

Estas plataformas pueden centrarse en el diseño y desarrollo de un tipo particular de aplicación o proceso, como la gestión de bases de datos, procesos comerciales o el diseño de interfaces de usuario, o a producir aplicativos completamente operativos, como aplicaciones web y móvil, aunque pueden requerir de codificación adicional para situaciones específicas. Las plataformas de Low-Code reducen la cantidad de codificación manual tradicional, lo que permite la entrega acelerada de aplicaciones comerciales.

Otro beneficio de las LCDPs, es que una gama más amplia de personas puede contribuir al desarrollo de la aplicación, no solo aquellas con habilidades formales de programación, aunque aquellas personas que las posean podrán explotar de forma más efectiva esta tecnología, a lo que acompaña una reducción en el coste inicial de configuración, capacitación e implementación.

1.1.1. Crecimiento e implicación en la transformación digital

Con el proceso de transformación digital que están experimentando las empresas, la demanda de soluciones basadas en plataformas Low-Code no deja de aumentar.

- Según la empresa Forrester³, las plataformas Low-Code representan una industria que generó un mínimo de 1.700 millones de dólares durante el 2015.

¹“Low-Code Development Platform”

²El Software de Aplicación son los programas diseñados para o por los usuarios para facilitar la realización de tareas específicas.

³Forrester Research es una empresa independiente de investigación de mercados que brinda asesoramiento sobre el impacto existente y potencial de la tecnología[2]

- Recientemente, Forrester realizó una encuesta online donde participaron 254 responsables de TI y negocios de UK, EE.UU, Canadá y Australia sobre el uso, experiencias y expectativas acerca de las plataformas Low-Code, extrayendo las siguientes conclusiones[5]:
 - El Low-Code acelera el desarrollo de aplicaciones, demanda cada vez más importante por parte de las empresas. El 84 % de las empresas ya han adoptado una plataforma o herramienta de desarrollo Low-Code. Gracias a las LCDPs, las empresas se están volviendo más ágiles y alcanzando velocidades de comercialización mayores.
 - El Low-Code es capaz de satisfacer altas exigencias y aportar soluciones específicas con las que las empresas pueden acceder al mercado incluso superando requisitos de seguridad elevados.
 - Existe una tendencia en aumento de sustituir el código personalizado para las funciones claves de la lógica de negocio por alternativas Low-Code .
- De acuerdo con “Low-Code Development Platform Market by Component (Solution and Services (Professional and Managed)), Deployment Mode, Organization Size, Vertical (Telecom and IT, BFSI, Government), and Region - Global Forecast to 2022”, publicado por MarketsandMarkets [4] se espera que el tamaño de mercado de estas tecnologías crezca de 4.32 billones de USD⁴ en 2017 a 27.23 billones de USD en 2022.

El Low-Code sigue creciendo y parece que va a tomar gran importancia en el futuro del desarrollo software, sustituyendo a sistemas tradicionales para aportar mayor velocidad, agilidad y flexibilidad a las empresas y desarrolladores.

En mi propia y corta experiencia profesional he podido observar como las secciones dedicadas a estas tecnologías dentro de la empresa crecen de manera exponencial y que conocimientos en herramientas de este tipo son muy demandados en el mercado laboral actual.

1.1.2. Principales Puntos Fuertes

Hasta el momento hemos podido observar varias ventajas de las LCDPs, pero cabría destacar los siguientes puntos:

- **Reducción del tiempo de desarrollo:** Al ser estos sistemas capaces de generar automáticamente el código subyacente de los elementos gráficos que ofrecen, reduciendo en la mayoría de los casos las tareas del desarrollador a la configuración de dichos elementos.
- **Reducción del mantenimiento:** El proceso de actualización del código y funciones para mantener la alineación con el negocio desaparece o se reduce en gran manera al no disponer de ese código que actualizar y debido a la flexibilidad ofrecida por las LCDPs.
- **Reducción del coste:** Al reducir los tiempos obtenemos una reducción directa en el coste, pero además estas tecnologías permiten reducir el número de desarrolladores necesarios respecto al desarrollo tradicional.

⁴Dolares Estadounidenses

- **Mejora la productividad del desarrollador:** Al permitir dejar los detalles más técnicos a cargo de las herramientas Low-Code el desarrollador puede centrarse en obtener la máxima funcionalidad del software que está desarrollando.
- **Permite Alinear IT con los objetivos empresariales:** Al simplificar el proceso de desarrollo, perfiles mas afines al negocio que a los aspectos técnicos pueden implicarse en mayor medida permitiendo que las aplicaciones estén mas alineadas con los objetivos empresariales.
- **Gran afinidad con las metodologías ágiles:** Las LCDPs son extremadamente compatibles con las metodologías ágiles, lo que ayuda en gran medida a producir aplicaciones muy cercanas a las expectativas del cliente o negocio con gran flexibilidad y capacidad de adaptación a cambios durante el desarrollo.

1.1.3. Ejemplos de Herramientas Low-Code

A continuación, se introducirán brevemente algunos ejemplos de plataformas Low-Code:

- **Appian®:** Appian Corporation es una compañía de computación en la nube que ofrece una LCDP para el desarrollo de aplicaciones empresariales que pretende facilitar la combinación y automatización de procesos de negocio, inteligencia artificial, automatización robótica, datos e integraciones⁵.
- **Sales Cloud (Salesforce®):** Sales Cloud es el producto mas conocido de la empresa estadounidense Salesforce® que ofrece un CMR⁶ basado en tecnología Low-Code, esta solución tiene como objetivo facilitar el desarrollo de software para las relaciones con los clientes (gestión de usuarios, seguimiento, oportunidades de negocio).⁷
- **Microsoft PowerApps:** PowerApps es la LCDP de Microsoft® para el desarrollo de aplicaciones empresariales de forma acelerada que ofrece gran cantidad de plantillas y posee integraciones y conectores con otros servicios de Microsoft® como Azure® que la dotan de funcionalidades más avanzadas y capacidad de ampliación e incluso con herramientas CMR como la anteriormente mencionada Sales Cloud®.⁸
- **Mendix®:** Otra de las LCDPs líder para el desarrollo de aplicaciones según el “Gartner⁹ 2019 Magic Quadrant for Enterprise Low Code Application Platforms” como se muestra en la Figura 1.1

⁵<https://www.appian.com/platform/>

⁶Customer Relationship Management

⁷<https://www.salesforce.com/es/products/sales-cloud/overview/>

⁸<https://powerapps.microsoft.com/es-es/>

⁹Empresa consultora y de investigación de las tecnologías de la información con sede en Stamford



Figura 1.1: Gartner 2019 Magic Quadrant for Enterprise Low Code Application Platforms [11]

Existen muchas otras plataformas, hemos comentado algunas de las más importantes a falta de OutSystems[®], que como se ve en la Figura 1.1 se encuentra también liderando el sector de las LCDPs y que veremos con más detalle en el capítulo siguiente.

1.2. Low-Code VS No-Code

Tras una primera vista, podríamos llegar a pensar que Low-Code es prácticamente lo mismo que No-Code, pero esto no es así.

La principal diferencia entre el Low-Code y el No-Code, es la capacidad del primero para dejar un espacio abierto a la codificación tradicional, es decir, permitir la expansión de sus funcionalidades básicas predefinidas y de la personalización de la interfaz mediante el uso directo de un lenguaje de programación no gráfico. Esto requiere que las LCDPs posean herramientas capaces de combinar el código externo, introducido por el programador, con el autogenerado por la plataforma.

Esta diferencia es fundamental, puesto que otorga a las tecnologías Low-Code una gran ventaja en cuanto a la personalización y la reducción en las limitaciones de funcionalidad.

También destacar, que aunque el Low-Code evite en gran medida los lenguajes de programación tradicional, cada LCDP (sobre todo aquella que permite un desarrollo completo de aplicaciones) es en si misma un lenguaje, ya que sus elementos, aunque sean gráficos, han de ser configurados y estructurados como si de código se tratara y encontramos de igual manera sentencias de control como “if, for, switch...”, con el factor añadido de que cada LCDP posee su estructura propia, sus propios elementos y su forma y sintaxis concreta para configurarlos.

Capítulo 2

OutSystems®

OutSystems® es una compañía global de software empresarial cuya sede actual se encuentra en Atlanta, Georgia, en los Estados Unidos, es una LCDP para el desarrollo de aplicaciones web y móviles principalmente de carácter empresarial, que se ejecutan en la nube, en entornos locales o en entornos híbridos dependiendo de las necesidades del cliente. La versión actual de la plataforma es la 11.

En este capítulo trataremos de realizar una introducción a la evolución de la plataforma y los motivos por los cuales OutSystems® es la tecnología escogida para la realización de este trabajo.

2.1. Historia y Contexto

OutSystems®, nace como una StartUp fundada en Lisboa (Portugal) en 2001 por su CEO Paulo Rosado sobre la premisa de intentar lograr un software capaz de realizar cambios en las aplicaciones desarrolladas, de la forma más rápida y eficiente en costes independientemente del tamaño de la aplicación.

En 2002 OutSystems® consigue su primer cliente con la primera versión de la plataforma llamada “Hub Edition”. Por este tiempo OutSystems® tenía como objetivo las compañías de telecomunicaciones y las aplicaciones para el manejo de los envíos de SMS.

Para 2003 la compañía ya dispone de las primeras versiones de las herramientas Service Studio y Service Center ¹ y se concentra en la optimización del código generado y la compatibilidad con los distintos sistemas del momento. Es aquí, cuando también aparece la funcionalidad “1-click Publish” que permite centralizar todos los procesos de compilación y despliegue de la aplicación en una misma acción.

Es en 2004, cuando OutSystems® percibe que muchos de sus clientes están empleando la plataforma para el desarrollo de gran parte de las aplicaciones de gestión interna y no solo para

¹Las versiones actualizadas de las herramientas que dispone OutSystems® se presentarán en el capítulo 3

las aplicaciones de gestión de SMS, y deciden cambiar el rumbo y el foco inicial, puesto sobre las compañías de telecomunicación, para abarcar un ámbito mucho más general.

En los años siguientes la herramienta evoluciona a gran velocidad centrándose en aspectos como las integraciones por “web services” (las cuales suponen una gran expansión de las capacidades funcionales de las aplicaciones) en 2005, un control de versiones avanzado integrado en la misma plataforma, la gestión de Roles o la capacidad de realizar una depuración visual avanzada (esto nos permite movernos sobre nuestros flujos y componentes gráficos de forma visual durante las depuraciones) en 2006, la concienciación en documentar y aportar formación propia sobre la herramienta en 2007, etc.

Llegado 2016, OutSystems® ya dispone de una versión muy avanzada de su herramienta, que permite desarrollos de aplicaciones web y móvil acelerados, muy similar a la versión actual. Es en febrero de este año cuando OutSystems® anuncia una ronda de financiación de 55 millones de USD de la mano de North Bridge Growth Equity ² Tras este impulso, OutSystems® se sitúa ya en 2017 como uno de los líderes del Low-Code, según Forrester (Figura 2.1).



Figura 2.1: “Forrester Wave: Low-Code Development Platforms For application development and delivery Pros, Q4 2017”

Ya en 2018, Outsystems® de nuevo recibe financiación, esta vez 360 millones de USD de KKR and Goldman Sanchs y se proclama plataforma líder para el desarrollo de aplicaciones móviles (Figura 2.2) y también líder en las plataformas como producto para el desarrollo empresarial de alta productividad (Figura 2.3) según Gartner.

²Actualmente esta empresa inversora se llama Guidepost Growth Equity

Figure 1. Magic Quadrant for Mobile App Development Platforms



Source: Gartner (July 2018)

Figura 2.2: “Jul 2018 Gartner Magic Quadrant for Mobile App Development Platforms”



Figura 2.3: “2018 Gartner Magic Quadrant for Enterprise High-Productivity Application Platform as a Service”

Actualmente OutSystems® se mantiene como una de las plataformas líderes en el sector como se reflejaba en la Figura 1.1 y las predicciones para su futuro y el del Low-Code son muy positivas.

2.2. Por qué OutSystems® ?

La elección de OutSystems® como herramienta para la realización de este trabajo tiene dos motivos. El primero de ellos, OutSystems® permite el acceso gratuito a prácticamente la totalidad de sus funciones y permite mantener una visión global del proyecto habilitando desarrollos full stack. En segundo lugar, es la herramienta con la que realicé mis prácticas en empresa y con la que actualmente me encuentro trabajando.

Entrando en mas detalle en cada uno de estos motivos, OutSystems® proporciona a cualquiera que se registre en su plataforma un entorno personal gratuito³ con casi la plenitud de sus funcionalidades, esto ligado a ser capaz de controlar y desarrollar de forma centralizada todos los aspectos de una aplicación (bases de datos, front-end, back-end,), que permiten a una sola persona abordar un desarrollo completo de considerable magnitud, hacen de OutSystems® la herramienta adecuada para mostrar la capacidad y el potencial que poseen las LCDPs.

Por otro lado, en mis practicas en empresa en everis Spain S.L.⁴ entre a formar parte de la nueva apuesta de la compañía en España, OutSystems®.⁵ Actualmente en España no se trabaja mucho con OutSystems®⁶, aunque si se hace muy habitualmente con otras LCDPs como la de Salesforce®.

El empezar esta nueva sección en la empresa me ha permitido formarme en las herramientas de OutSystems® y conocer su funcionamiento y posibilidades. Mediante, autoformación (nadie tenía experiencia previa con esta tecnología en la sede) tanto directamente en OutSystems® como en lenguajes en los que este se apoya (HTML, CSS, C#, JavaScript) he obtenido los conocimientos necesarios para desarrollar aplicaciones empleando sus herramientas. Disponer de estos conocimientos me impulsa a elegir OutSystems® frente a otras opciones.

³En el siguiente capítulo hablaremos de los entornos en OutSystems®

⁴Empresa de consultoría tecnológica, cuya actividad principal reside en el desarrollo de soluciones software empresariales.

⁵OutSystems® ya se empleaba de forma habitual en las sedes de otros países como Portugal.

⁶Una de las principales barreras para expansión resultan ser los idiomas, la documentación está completamente en inglés o portugués

Capítulo 3

Las Herramientas de OutSystems®

A continuación, se mostrarán las diferentes herramientas que proporciona OutSystems® tanto para el desarrollo de aplicaciones como para su gestión, mantenimiento, extensión y monitorización (Gestión de las DevOps¹), que nos permiten obtener los beneficios derivados de emplear Low-Code (Sección 1.1.2).

Antes de centrarnos en las herramientas por separado, introduciremos el contexto para el funcionamiento de OutSystems® y cómo es y trabaja la estructura que sostiene el sistema.

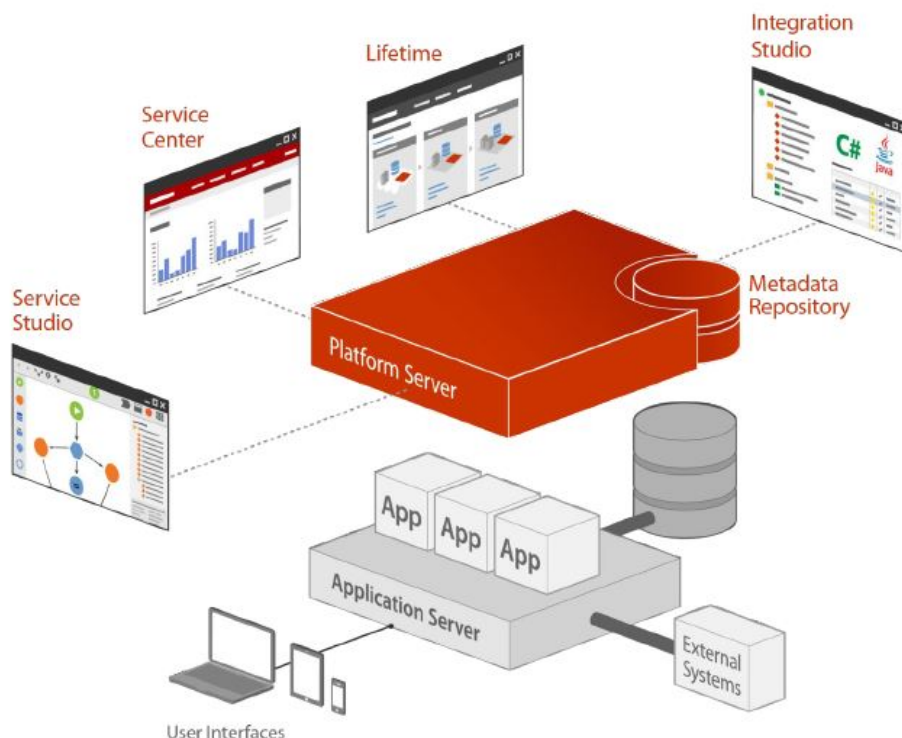


Figura 3.1: Estructura Básica del Servidor de Outsystems

¹Acrónimo inglés de development (desarrollo) y operations (operaciones), que se refiere a una metodología de desarrollo de software que se centra en la comunicación, colaboración e integración entre desarrolladores de software y los profesionales TI[12]

La tecnología de OutSystems® se encuentra alojada en la nube. Los servidores de OutSystems® son los encargados de realizar todos los procesos de compilación, despliegue, ejecución y el control de las aplicaciones. Estos servidores, son accesibles desde las diferentes herramientas, cómo Service Studio (IDE principal de OutSystems®), y es en el repositorio de metadatos de este servidor donde se almacenan las diferentes versiones de las aplicaciones.

Este servidor de la plataforma, tras realizar las tareas de compilación y generación de las aplicaciones, realizará el despliegue de estas en servidores de aplicación estándar que puedan emplear bases de datos y sistemas externos tradicionales para su ejecución. La Figura 3.1 muestra la estructura descrita.

Es importante conocer, para entender como se trabaja en OutSystems®, como este se encuentra organizado en entornos, los cuales podemos dividir en dos grandes grupos:

- **Entornos Personales:** Estos entornos son, como su nombre indica, personales y ligados a la cuenta de cada desarrollador², no están sujetos a licencia y permiten emplear casi la totalidad de las funcionalidades de OutSystems®, a excepción de algunas relacionadas con operaciones, control y integración con sistemas propietarios. También presentan otras limitaciones en cuanto a espacio de almacenamiento (2 GB), capacidad de colaboración y portabilidad de los desarrollos realizados (No es posible compartir proyectos entre entornos personales).

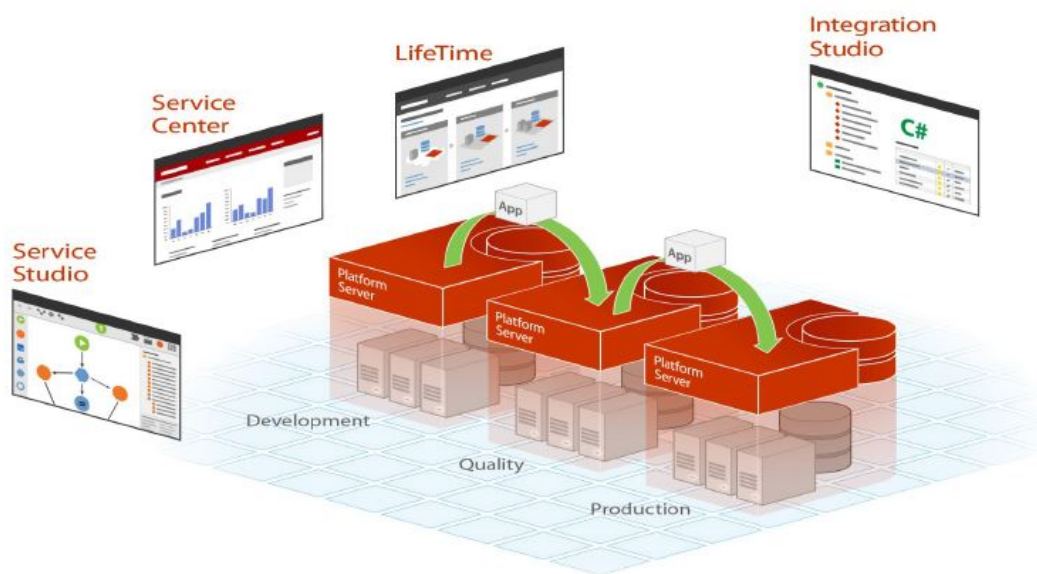


Figura 3.2: Estructura ejemplo de entorno con licencia

- **Entornos Profesionales:** Estos están sujetos a licencia, poseen todas las funcionalidades tanto de operación y control, como de integración³, permite ajustar los recursos (almacenamiento) a las necesidades del desarrollo y permite y facilita el trabajo colaborativo junto a la portabilidad. La funciones de control y operación cobran verdadero

²Si un desarrollador posee varias cuentas puede tener varios entornos personales

³Es posible que algunas opciones dependan del tipo de licencia

sentido en este tipo de entornos ya que podrían ser definidos como entornos compuestos, es decir, se componen de entornos o particiones menores dedicadas a etapas y funciones distintas durante el proceso de desarrollo global de una aplicación (desarrollo, testing, QA, producción). La Figura 3.2 muestra un ejemplo estructural.

Si hablamos ahora del proceso de desarrollo de una aplicación en Outsystems, se puede establecer que este sigue una estructura bastante marcada por los siguientes pasos:

- **Elección de extensiones:** Con una visión global del proyecto, de la arquitectura y de la infraestructura, seleccionaremos e identificaremos las aplicaciones, bases de datos externas o código personalizado necesario, y prepararemos estas extensiones para posteriormente incluirlas a nuestro desarrollo low-code de forma sencilla e intuitiva.
- **Desarrollo y diseño visual:** Con todos los componentes listos, empleando Service Studio desarrollaremos la totalidad de nuestra aplicación, tanto la gestión de datos, los procesos y la interfaz.
- **Generación de la aplicación:** gracias a la funcionalidad de outsystems “One Click Publish con un solo clic podremos iniciar el proceso de generación y compilación del código HTML, JavaScript y .NET optimizado para la implementación de nuestra aplicación móvil o web que nos permiten pasar a la gestión y monitorización de las versiones de nuestra aplicación.
- **Despliegue personalizado:** Podremos desplegar nuestra aplicación sobre la infraestructura que deseemos: PaaS⁴, “On-Premises⁵, o mixtas.
- **Iteración Continua:** Gracias a la velocidad de producción de OutSystems® y sus herramientas de feedback, los tres pasos anteriores pueden iterarse de forma continua para alcanzar los objetivos deseados y mantener un proceso de mejora continuada.

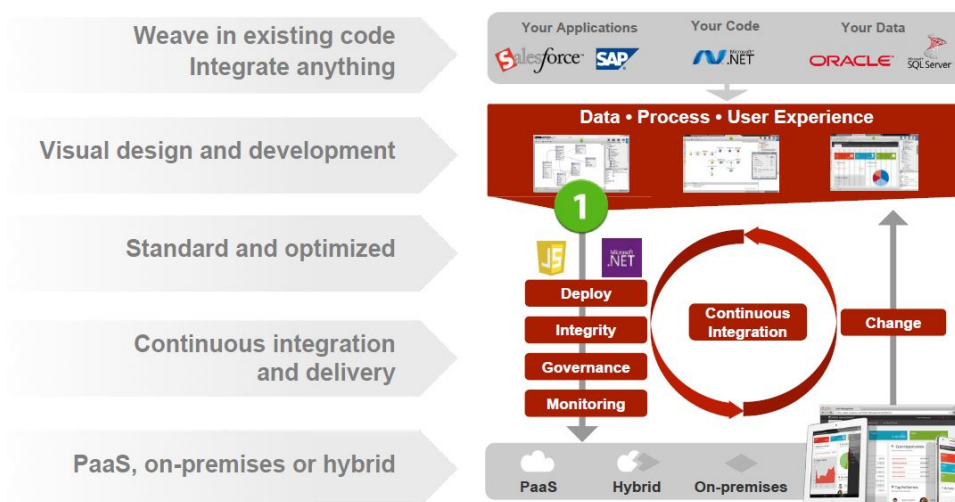


Figura 3.3: Proceso de desarrollo tipo con OutSystems®

⁴Plataforma como servicio (Cloud)

⁵Infraestructura física propia

Ahora si, pasaremos a ver de forma mas detallada cada una de las herramientas que nos proporciona OutSystems®

3.1. Service Studio

Service Studio, es el IDE⁶ principal de OutSystems donde se realiza prácticamente la totalidad del desarrollo de aplicaciones tanto web como móvil.

Es la herramienta que nos permite crear aplicaciones y módulos para desplegarlos en el servidor, nos permite definir las estructuras de datos que empleará nuestra aplicación, es también donde crearemos las interfaces de usuario de nuestras aplicaciones web o móvil y donde podemos modificar el código CSS para personalizar el “look and feel de estas, y nos permitirá definir la lógica de negocio junto a los procesos y temporizadores que esta necesite.

En definitiva, será en Service Studio donde se produzcan todos los procesos de creación, edición, actualización, publicación, previsualización y depuración de nuestras aplicaciones.

Pasaremos a continuación, a ver más en detalle de qué elementos se compone esta herramienta, cómo funciona e introduciremos brevemente algunos de los componentes con el fin de hacer más sencilla la presentación de la aplicación desarrollada.

3.1.1. Galería de Aplicaciones

Esta es la primera interfaz con la que nos encontramos al acceder a Service Studio, encontramos aquí una galería con todas las aplicaciones y “Plug-Ins que hayamos creado o descargado en nuestro entorno, con pequeñas indicaciones de si son móvil o web (Figura 3.4).

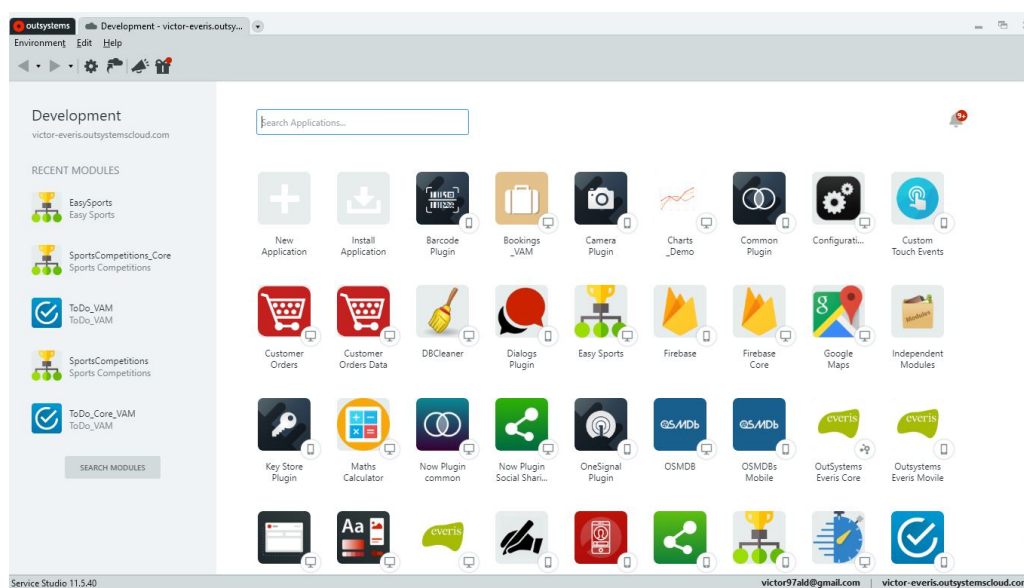


Figura 3.4: Galería de Aplicaciones Service Studio

⁶“Integrated Development Environment”

Desde aquí podemos crear nuevas aplicaciones o instalarlas, y accediendo a una de ellas encontraremos acceso a los módulos (y a la posibilidad de crearlos) que componen la aplicación (Figura 3.5). Será al abrir estos módulos cuando accedamos al “WorkSpace para el desarrollo.

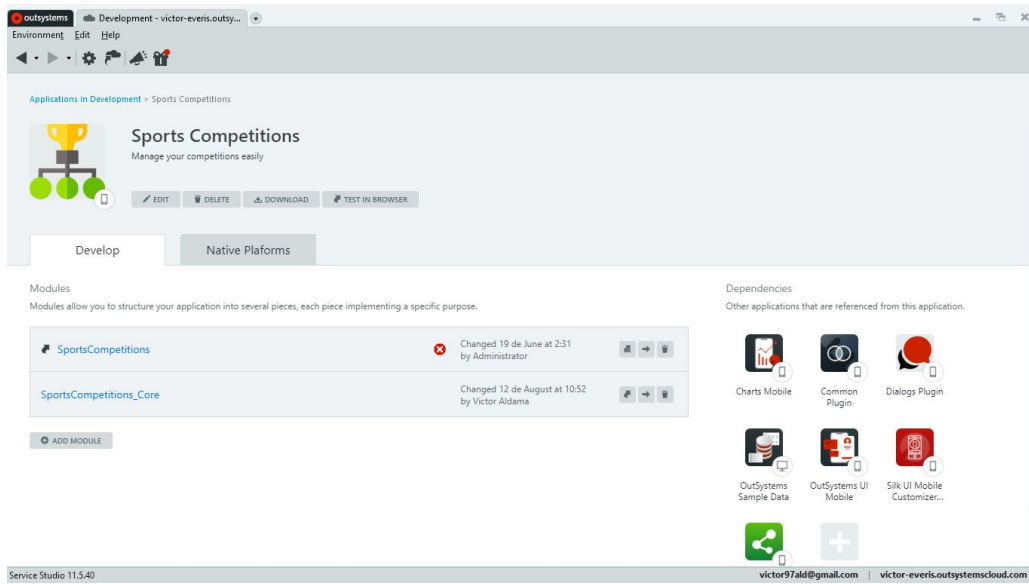


Figura 3.5: Módulos de las Aplicaciones en Service Studio

También en la pantalla que mostraba la Figura 3.5, si se tratara de una aplicación móvil encontramos la posibilidad de generar las aplicaciones nativas para Android o IOS. (Figura 3.6)

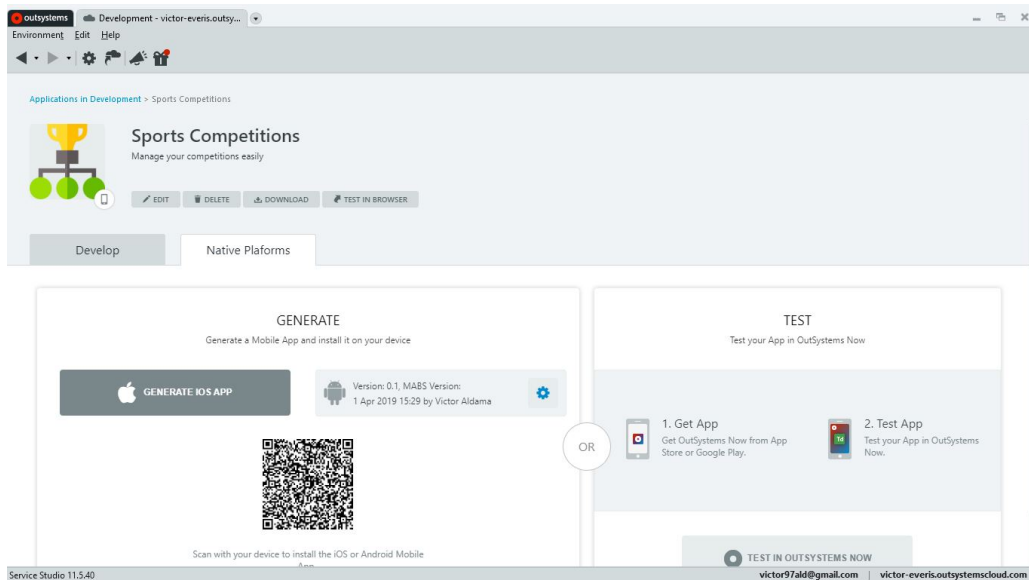


Figura 3.6: Generacion Nativa de Aplicaciones en Service Studio

3.1.2. La Distribución del “WorkSpace”

Como ya sabemos, las soluciones Low-Code apuestan por entornos de desarrollo gráficos que faciliten la tarea al desarrollador, Service Studio mantiene esta premisa y presenta un

entorno de trabajo para la edición de los distintos módulos de las aplicaciones con la estructura que muestra la Figura 3.7

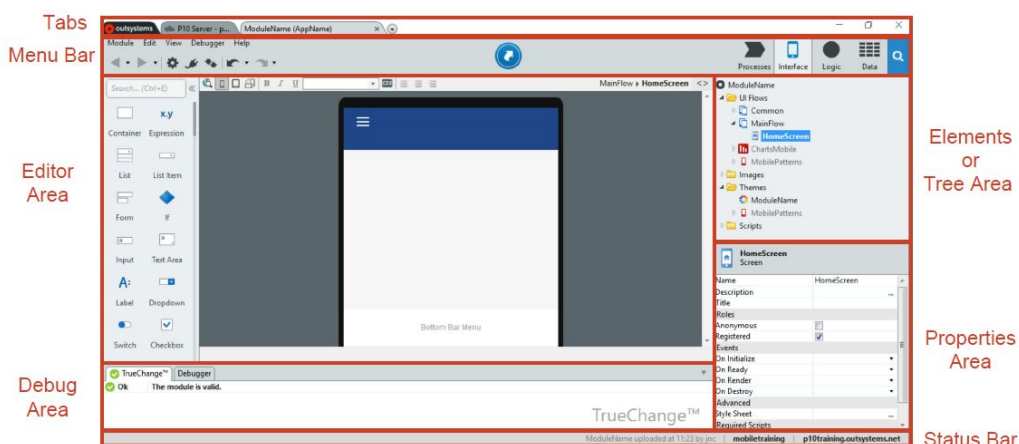


Figura 3.7: Layout del editor de Service Studio

No se encuentra en el alcance de este documento explicar detalladamente el uso y utilidad de cada una de las áreas y barras de esta herramienta, pero si vamos a destacar algunas partes que facilitarían posteriores explicaciones.

Es importante destacar, que las 4 pestañas a la derecha en la barra del menú organizan los elementos de la aplicación según su tipo, procesos, interfaz, lógica y estructura de datos. Movernos entre estas pestañas cambiara el contenido del área dedicada al árbol de elementos que mostrara aquellos que guardan relación con la pestaña seleccionada, de esta forma se obtiene una visión organizada y estructurada durante el desarrollo.

El área de edición se subdivide a su vez en varias partes menores de las que destacaremos el “Canvas y la “ToolBox(Figura 3.8)

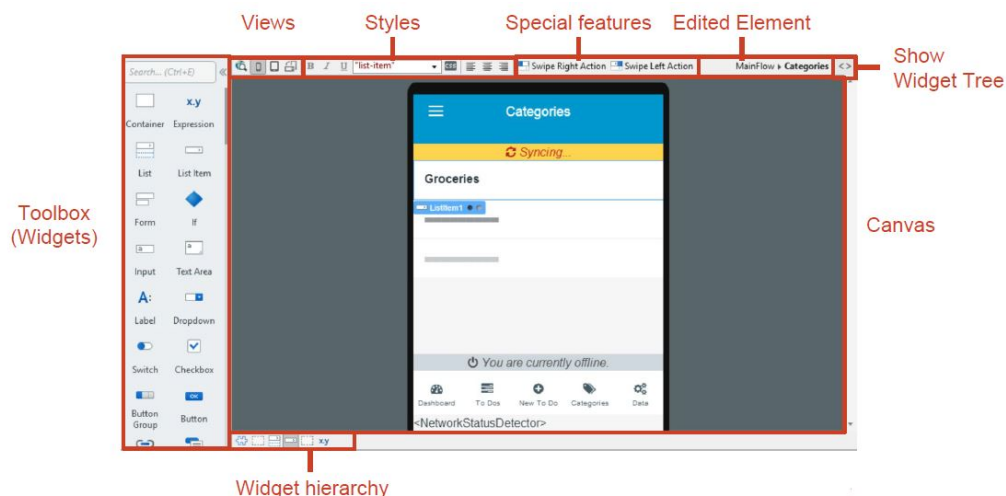


Figura 3.8: Detalles del Area de Edición Service Studio

El “Canvas es del tipo WYSIWYG ⁷ cuando nos encontremos editando la interfaz, aunque esta área cambiará cuando editemos lógica, procesos, estructuras o consultas a bases de datos.

La ”ToolBox” contiene los componentes que emplearemos para el desarrollo de las diferentes partes de la aplicación (formularios, tablas, entradas de datos, etc., cuando diseñemos la interfaz; sentencias de control, if, foreach, etc., cuando diseñemos lógica).

Será en el **área de propiedades** donde se realizara la configuración de todos estos componentes al añadirlos al “canvas para la interfaz o flujos de lógica.

3.1.3. “The Forge”

La fragua, The Forge en inglés, es un repositorio de aplicaciones, módulos, plug-ins, extensiones e integraciones, creadas tanto por la comunidad como por el propio OutSystems®.

Este espacio nos permite extender la funcionalidad de nuestras aplicaciones con componentes más allá de los por defecto de Service Studio y nos permite acceder a funcionalidades de nuestros dispositivos como la cámara.

The Forge posee un espacio reservado dentro de Service Studio con una pantalla dedicada, la cual posee un buscador que nos permitirá encontrar las funcionalidades que necesitemos.(Figura 3.9)

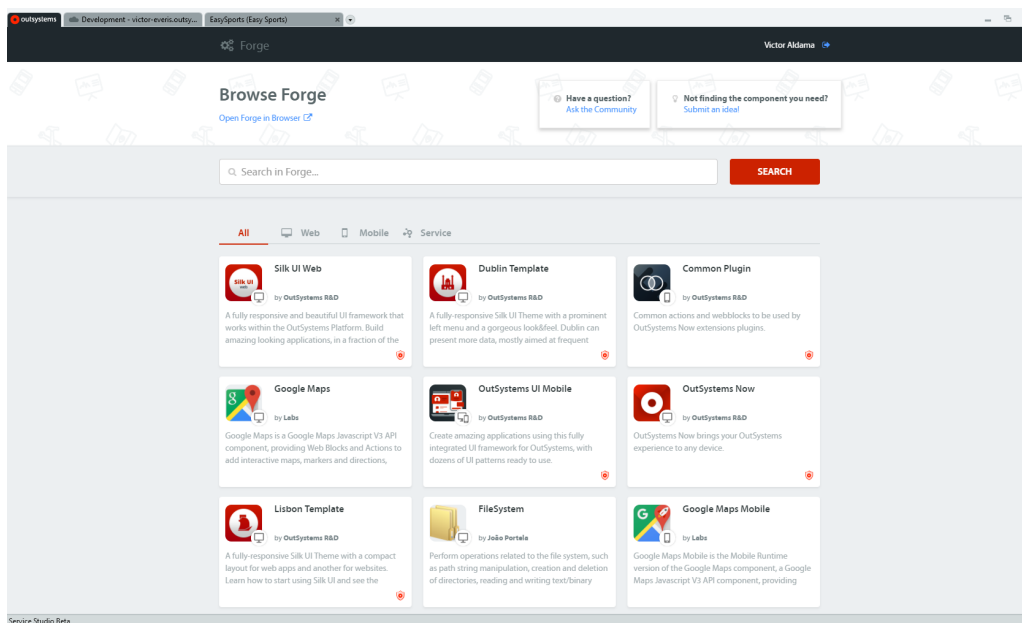


Figura 3.9: “The Forge” en Service Studio

Cabe destacar que una vez abiertas cada una de las secciones anteriores, podremos navegar entre ellas mediante las pestañas que se pueden ver en la esquina superior izquierda de la Figura 3.9

⁷“What You See Is What You Get

3.2. Integration Studio

Integration Studio es la herramienta de OutSystems® que nos permitirá realizar las extensiones con funcionalidades externas a las ofrecidas por defecto en Service Studio y que nos permitirán complementar o personalizar completamente nuestras aplicaciones o integrarlas con otros sistemas ya existentes. Con este propósito hay tres acciones principales que podemos realizar con Integration studio.(En verde en la Figura 3.10)

- **Crear entidades:** Nos permite crear las entidades (tablas) sobre las que se mapearán las tablas de una base de datos externa, es decir, preparar las tablas que podremos usar en Service Studio y que se comportaran como replicas referenciadas de nuestra base de datos original (modificarlas en Service Studio las modificara también en el origen) y realizar cualquier conversión de tipo de datos que sea necesaria para compatibilizar ambos entornos.
- **Crear Acciones:** Podemos crear acciones que aporten lógica o funcionalidades complejas a nuestro desarrollo en Service Studio, para ello deberemos realizar un desarrollo en .NET que contenga la lógica o funcionalidad que deseemos implementar y que gracias a Integration Studio seremos capaces de convertir en un elemento utilizable dentro del desarrollo Low-Code de Service Studio. Con este propósito, Integration Studio permite configurar el acceso directo a un IDE de desarrollo en .NET (C#), como por ejemplo, Visual Studio® al que se tienen acceso directo desde donde se indica en rojo en la Figura 3.10. Sobre el IDE configurado se generará el esqueleto del código que permitirá a Integration Studio cumplir con su cometido.
- **Crear estructuras:** Es un elemento que nos permitirá manejar tipos de datos complejos, listas, etc. Son el sustento para poder realizar cualquiera de las dos operaciones anteriores.

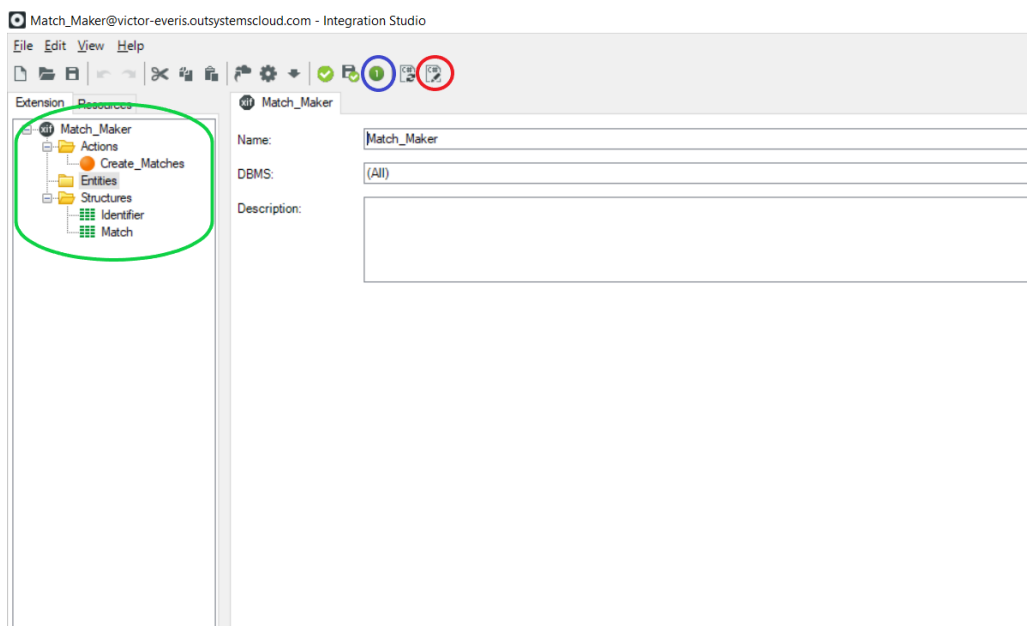


Figura 3.10: Layout de Integration Studio

Integration Studio, posee también su propia versión del “1-click publish que se encargará de realizar toda la labor de compilación y despliegue sobre el entorno con el nos hayamos conectado a la herramienta.(En azul sobre la Figura 3.10)

3.3. Service Center

Service Center es la herramienta de administración principal de los entornos. Mediante ella, OutSystems® nos dota de la capacidad de administrar una gran cantidad de propiedades de nuestros entornos y aplicaciones, como por ejemplo, las versiones, pudiendo borrar versiones antiguas para liberar espacio si fuera necesario o también configurar los accesos a bases de datos externas, puesto que es aquí y no en Integration Studio donde introduciríamos los parámetros y credenciales necesarias para conectar con las bases de datos externas, y muchas otras mas funciones como editar variables o propiedades de carácter global de cada una de las aplicaciones.

Es junto a Life Time, la solución que nos presenta OutSystems® para facilitar y acelerar las DevOps.⁸ En la Figura 3.11 se puede ver el “Layout” de la herramienta con algunas de sus opciones.

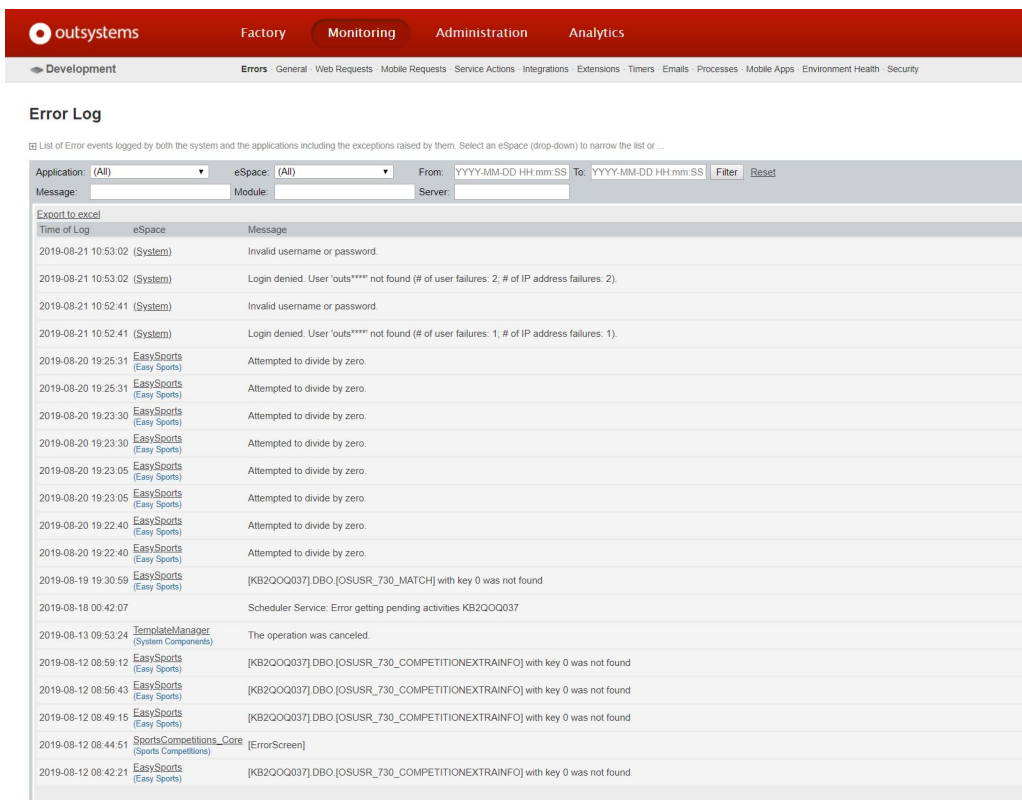


Figura 3.11: Layout de Service Center

⁸ El uso de estas dos herramientas queda fuera del alcance de este documento por lo que no se detallaran tanto como las anteriores

3.4. Life Time

Como se ha indicado anteriormente, Life Time está destinada a facilitar las DevOps, pero a diferencia de Service Center, esta más orientada a gestionar los movimientos entre los diferentes subentornos de un entorno con licencia (desarrollo, testing,) de forma que estos movimientos no generen incompatibilidades o errores durante su transcurso, monitorizando y asegurando el proceso. Life Time, también permite llevar un control sobre la salud de nuestros entornos (espacio restante u otros inconvenientes), como se muestra en la Figura 3.12, y además posee otras funcionalidades como el control de usuarios o el acceso a analíticas del estado, funcionamiento y concurrencia de nuestra aplicación.

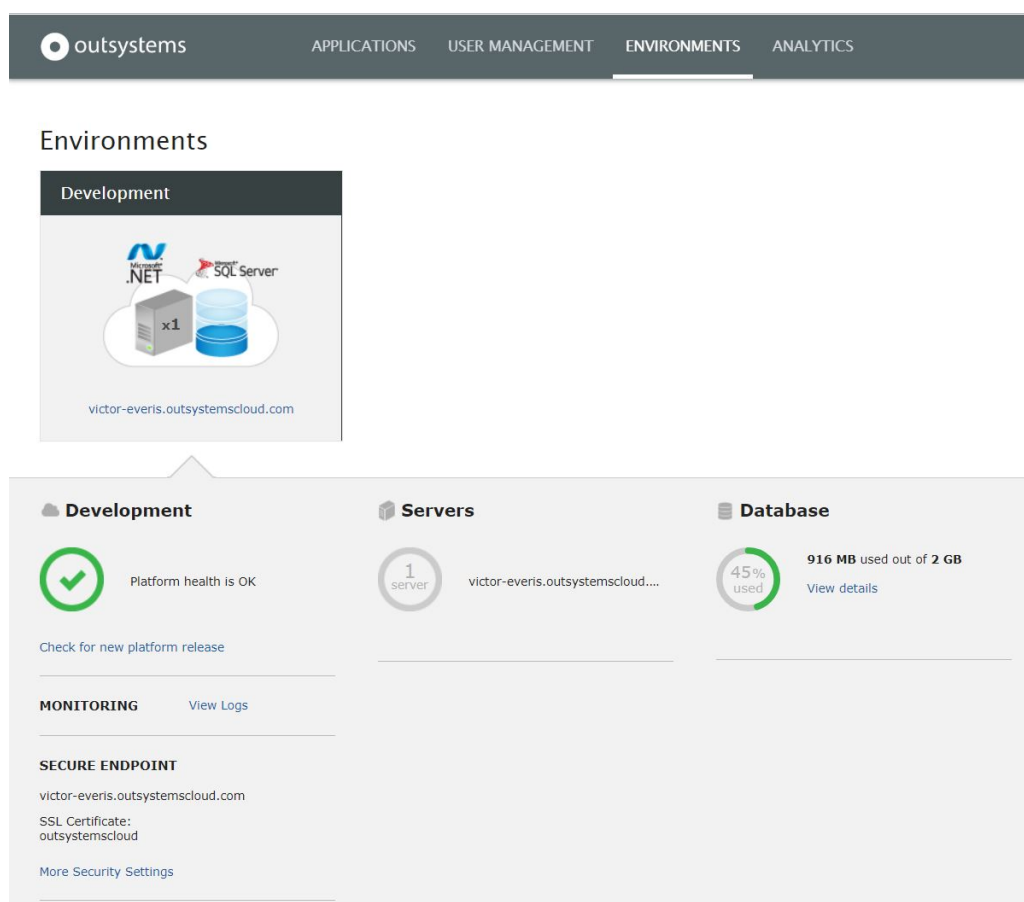


Figura 3.12: Layout de Life Time

Capítulo 4

La Aplicación “Easy Sports”

En este capítulo se estudiará en detalle la aplicación desarrollada “Easy Sports, se detallarán los distintos elementos que la componen, sus funcionalidades y opciones, las extensiones implementadas y se tratará de mostrar algunos ejemplos del “código gráfico realizado en OutSystems® con el objetivo de mostrar más detalles sobre el proceso de desarrollo en esta tecnología.

4.1. Idea y Objetivo

El objetivo de la aplicación es ayudar a simplificar el proceso de crear y gestionar una competición deportiva, desde decidir cómo será la competición y qué información deberán proporcionar los usuarios, a poder seguir la clasificación y los resultados de los partidos, pasando por validar la información de los jugadores. La idea es poder satisfacer tanto las necesidades de una competición informal, como una liga entre equipos de amigos, como las de una competición de carácter más serio, como las competiciones internas de la universidad.

Para entender mejor este objetivo, será de ayuda conocer el origen de la idea que lo impulsa. La idea surge de dos escenarios distintos.

El primero de ellos, es mi propia experiencia como jugador y organizador en eventos deportivos. En la mayoría de las competiciones, sobre todo en las más pequeñas, la organización de estos eventos se encuentra descentralizada, es decir, existe una serie de personas encargadas de la organización que contactan con otras para subrogarles parte de este trabajo, por el camino la información se distribuye por email, WhatsApp o en el peor de los casos en papel o cara a cara.

De esta forma, llegar hasta los jugadores es un proceso con varios pasos y que puede complicarse, la información personal pasa por distintos tipos de mensajería y cualquier contratiempo puede tardar demasiado en encontrar solución. Es por ejemplo el caso del Trofeo UPV, en el cual responsables de la delegación de alumnos deben reclutar a los jugadores, se forman grupos de WhatsApp los cuales se saturan de información, también los calendarios se manejan

con documentos o fotografías que circulan por estos grupos y el proceso en general se vuelve ineficiente (También para los encargados de recopilar o proveer información a los jugadores).

Para evitar esto, surge la idea de una aplicación capaz de reducir la mensajería externa a las credenciales para el acceso a la competición.

Por otra parte, mi tutor de este trabajo, Alberto Albiol Colomer, me comento haber desarrollado el mismo una aplicación para gestionar una competición de pádel con sus amigos, lo que hizo mas evidente la necesidad de una aplicación que gestionara este tipo de eventos.

4.2. Modelo de Datos

Pasaremos ahora a los detalles de la aplicación y empezaremos por el modelo de datos que alimenta toda la lógica de la aplicación web.

Este modelo de datos esta desarrollado en un módulo separado desde el cual alimenta a los dos módulos que contienen la lógica e interfaz de las aplicaciones Web y Móvil, la estructura del modelo se puede observar en la Figura 4.1.

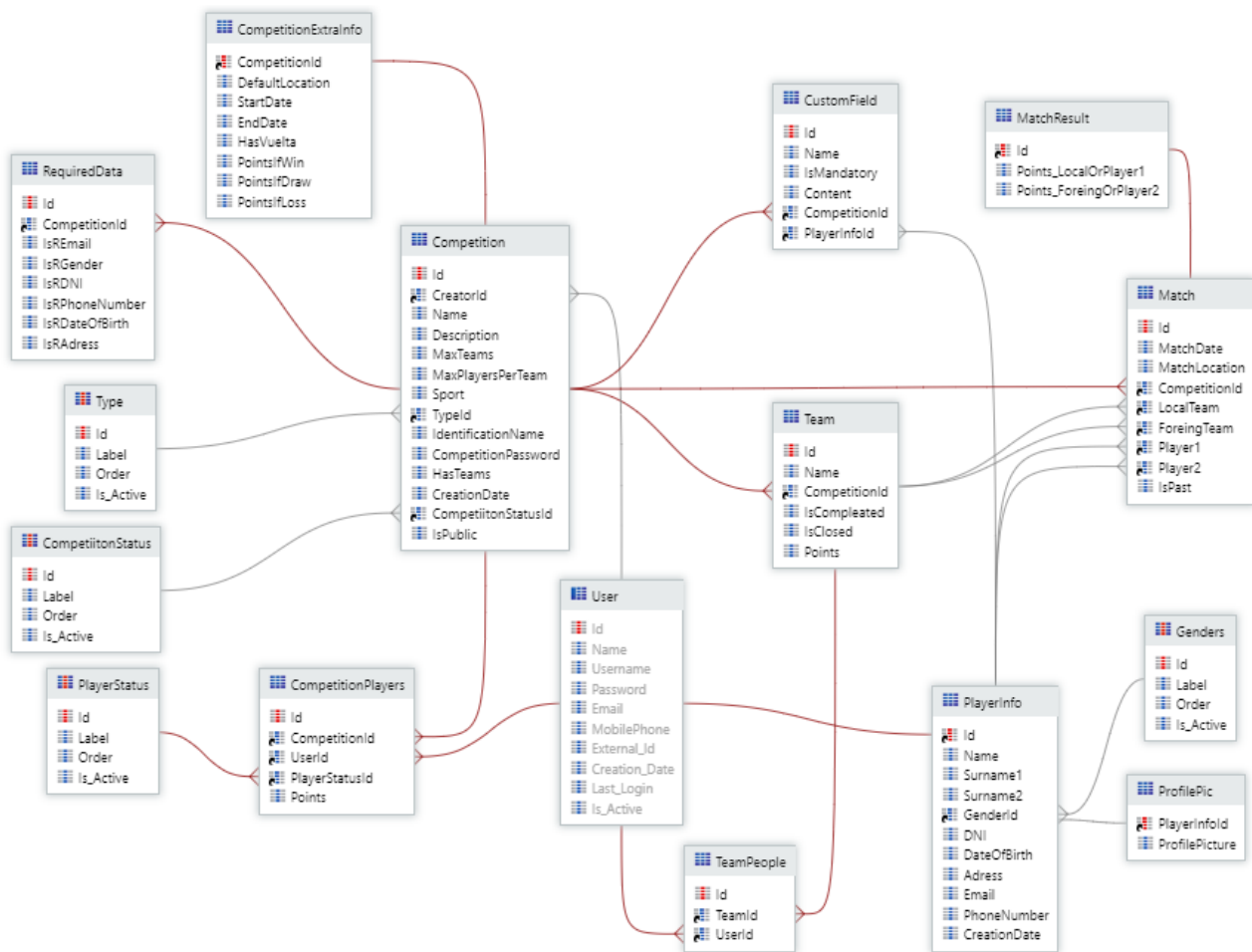


Figura 4.1: Modelo de datos de “Easy Sports”

Podemos ver dos tipos de entidades¹ las que se presentan con las tres columnas azules al lado del nombre de la entidad, como Competitions, y las que se presentan con la columna central en rojo, como Type. Estas segundas se denominan entidades estáticas y se diferencian de las primeras porque sus registros (filas), únicamente pueden definirse o modificarse durante el desarrollo, es decir, contienen valores invariables durante la ejecución.

Todas las entidades poseen un atributo que recibe el nombre de Id y que funciona como “Primary Key, cada vez que nos refiramos al Id de una entidad o que encontremos el nombre de una entidad seguido de Id, es que se está haciendo referencia a este atributo (Son los empleados para establecer las claves foráneas).

Las líneas del modelo indican las relaciones entre tablas, los finales ramificados indican que esta tabla es una parte n de las relaciones (1 a 1, 1 a n, n a n), de no estarlo implicaría que es la parte 1.

El color rojo de las líneas nos indica que al eliminarse un registro de la tabla de la cual se posee una “foreign key o Id, el registro de la tabla que posee este Id será eliminado con él. Este tipo de relación se denomina “Delete”, que permite la eliminación en cascada de registros (util para mantener la integridad de la base de datos). Existen dos tipos de relaciones más, “Ignore” y “Protect”, que respectivamente mantienen el registro intacto o evitan que se pueda eliminar el registro del que poseemos la referencia o Id si existe alguno que le haga referencia en la tabla con la que se mantiene esta relación.

Por ejemplo, al borrar una competición, se eliminarían todos los registros de las entidades que guarden relación con ella si todas las relaciones son “Delete”, si existe alguna con relación “Ignore” no se eliminara el registro relacionado, lo que generará inconsistencias en la base de datos, al no existir la competición a la que están ligados, y si por el contrario existe alguna relación con “Protect”, no podremos eliminar la competición.

El modelo de datos está compuesto de las siguientes tablas o entidades:

- **User:** Esta es una entidad del sistema, es decir, existe por defecto en OutSystems® y recoge la información esencial de todos los usuarios registrados en el entorno.
- **Competition:** Esta es la entidad central del modelo, borrar un registro de esta entidad conllevará la eliminación en cascada de muchos registros de otras entidades para mantener la consistencia del modelo de datos. Los atributos (columnas) de esta entidad son:
 - **Creator Id:** Es la clave foránea que relaciona esta tabla con la entidad de Users de forma que podemos identificar cual fue el usuario que creó la competición.
 - **Name:** El nombre de la competición.
 - **Description:** Una pequeña descripción de la competición que de más información al usuario que va a unirse.
 - **Max Teams:** Entero que determina el máximo de equipos que admite la competición.

¹Como ya se ha comentado anteriormente las tablas se denominan entidades en OutSystems®

- **MaxPlayersPerTeam:** Este atributo será también un entero con una peculiaridad, si la competición es por equipos, indicara el número máximo de jugadores por equipo, si esta fuera individual, este campo hará referencia al número total de jugadores que pueden participar.
 - **Sport:** En este atributo se indicara el deporte de la competición, el cual será un texto de carácter informativo, ya que la generación de la competición es independiente del deporte escogido.
 - **TypeId:** Este atributo es una llave foránea a la entidad estática Type que nos indica el tipo de competición².
 - **IdentificationName:** Este atributo contiene un nombre abreviado (NickName) y auto generado el cual se empleará para compartir y dar acceso a los jugadores a la competición a la que pertenezca.
 - **Password:** Acompañara al NickName para dar acceso a la competición.
 - **HasTeams:** Este atributo será de tipo “Boolean”, y nos indicara si la competición es por equipos o no.
 - **CreationDate:** Es la fecha en la que se creo la competición lo que nos permite conocer su antigüedad para funciones de ordenación o limpieza de en la base de datos.
 - **CompetitionStatusId:** Es la clave foránea a la entidad estática CompetitionStatus, nos proporciona información sobre el estado de la competición para uso interno, es decir, saber en que paso de la creación, administración o progreso se encuentra la competición durante su ciclo de vida.
 - **IsPublic:** Este “Boolean” nos indicara si la competición es de carácter público, es decir, si daremos acceso a ver los resultados y progreso de la misma a usuarios no registrados.
-
- **Type:** Es la entidad estática que contiene los posibles tipos de competición que se pueden crear.
 - **CompetitionStatus:** Es la entidad estática con los estados en los que se puede encontrar una competición.
 - **PlayerInfo:** Esta entidad recoge la información de los jugadores a nivel de la aplicación, mantiene una relación 1-1 con la entidad User, es decir, el Id de esta tabla es a su vez el Id de un registro de la tabla User, de esta forma conociendo el Id del usuario podemos acceder a la información de la plataforma y también a su información en la aplicación. Los atributos que componen la entidad recogen la información que su propio nombre indica, y también contiene un relación con la entidad estatica Genders.
 - **Genders:** Es la entidad estática que contiene las opciones que podremos elegir cuando debamos elegir un género.

²Mas adelante se explicaran los distintos tipos de competiciones que podemos crear.

- **ProfilePic:** Es una entidad con relación 1 a 1 con PlayerInfo, ya que contiene la información binaria de la imagen de perfil asociada a cada registro de esta tabla. Esta separación en dos tablas es por motivos de rendimiento, ya que de esta manera no es necesario mover los datos de esta tabla, los cuales son más voluminosos, cuando necesitemos acceder a la información del jugador (acceso frecuente).
- **Team:** Esta entidad contiene los equipos creados para una competición, y se compone de los siguientes atributos:
 - **Name:** Es el nombre que se le ha dado al equipo.
 - **CompetitionId:** Relaciona cada equipo registrado con una competición, en este modelo se considera que cada equipo esta ligado a una competición, es decir, un equipo no podrá coexistir en dos competiciones al mismo tiempo, sin embargo si puede existir un equipo con el mismo nombre y con los mismos jugadores.

El principal motivo de este sistema, es que será el creador de la competición el encargado de establecer los equipos, y es en esta misma entidad donde se lleva el conteo de los puntos del equipo en la competición.
 - **IsCompleted:** Este booleano nos proporciona información sobre si se ha alcanzado el máximo de jugadores que pueden formar el equipo.
 - **IsClosed:** Este booleano permite saber si el equipo ya ha sido validado y cerrado por el administrador.
 - **Points:** Aquí es donde se lleva la cuenta de los puntos que ha acumulado el equipo en la competición.
- **TeamPeople:** Esta entidad nos permite establecer una relación n a n entre User y Team, de forma que somos capaces de asociar usuarios con su respectiva información a un equipo, y por tanto, a una competición.
- **CompetitionPlayers:** Esta entidad tiene la misma función que la anterior pero contemplando el caso en que la competición no es por equipos y por tanto debemos relacionar directamente a los usuarios con la competición (esta relación se establece igualmente en el caso por equipos, pero será la única en caso de ser individual). Esta entidad contiene también un Id a la entidad estática PlayerStatus mediante la que podemos establecer si este ha sido validado o rechazado en la competición y el entero 'Points' que nos permite llevar la cuenta de los puntos del jugador en una competición individual.
- **PlayerStatus:** Contiene los estados en los que se puede encontrar un jugador dentro de una competición.
- **CompetitionExtraInfo:** Esta entidad expande la información sobre la competición (relación 1 a 1) con fines orientados a la generación automática de los partidos y otros aspectos destinados al transcurso de la competición. Se compone de los siguientes atributos:
 - **CompetitionId:** Establece la relación 1 a 1 entre esta entidad y la entidad Competition.

- **DefaultLocaion:** Contendrá la ubicación por defecto en la que tendrá lugar la competición.
 - **StartDate/EndDate:** Contienen las fechas entre las que debe de estar comprendida la competición.
 - **HasVuelta:** Es un booleano para indicarnos si la competición a generar tendrá un formato de ida o de ida y vuelta, lo que implica el doble de jornadas y partidos.
 - **PointsIf Win/Draw/Loss:** Estos atributos son enteros que nos permitirán saber la cantidad de puntos que recibirá cada equipo en cada una de las situaciones de victoria, empate o derrota.
- **Match:** En esta entidad es la que guarda los datos de los partidos o enfrenamientos, de sus atributos cabe destacar:
- **MatchDate:** Contiene la fecha del enfrentamiento generada dentro del rango de fechas especificado en CompetitionExtraInfo, puede modificarse si necesita.
 - **MatchLocation:** Por defecto toma el valor especificado en CompetitionExtraInfo, pero puede modificarse para cada partido después de la autogeneración.
 - **CompetitionId:** Relaciona los partidos con la competición.
 - **LocalTeam:** Contiene el Id del equipo local para este enfrentamiento, de tratarse de una competición sin equipos su valor sería nulo.
 - **ForeingTeam:** Contiene el Id del equipo visitante, y tiene le mismo comportamiento que el atributo anterior.
 - **Player1:** Contiene el Id del jugador Local o 1, de ser una competición por equipos su valor sería nulo.
 - **Player2:** Contiene el Id del jugador Visitante o 2, y se comporta como el atributo anterior.

Las dos entidades restantes se emplean dependiendo del tipo de competición, para dar opción a elegir o ampliar la información que queremos solicitar a los jugadores, estas son:

- **RequestedData:** Esta entidad nos permitirá registrar mediante sus atributos (Booleanos), cuáles de los campos (cuyo nombre identifican) son información necesaria para ingresar en la competición. Esta tabla podría tener una relación 1 a 1 con Competitions, pero en un origen no se realiza esta relación y ya avanzado el desarrollo es peligroso para la consistencia de la base de datos realizar esta modificación por lo que se mantuvo la relación original.
- **CustomField:** Gracias a esta entidad se ha podido implementar la funcionalidad de realizar una solicitud de campos totalmente personalizados por el creador de la competición. Los atributos que componen esta entidad son:
- **Name:** Es el atributo mediante el cual el creador de la competición podrá indicar al jugador que información es la que está solicitando.

- **IsMandatory:** Este atributo booleano nos permite indicar si completar este campo es obligatorio para poder entrar a la competición o no.
- **Content:** Sera aquí donde se almacene la información proporcionad por el jugador al rellenar el campo en el proceso de inscripción.
- **PlayerInfoId:** Por último, este atributo nos permite relacionar la información con el usuario que la envió par unirse a la competición.

4.3. Aplicación Web

Como ya se ha mencionado anteriormente este trabajo incluye el desarrollo de una aplicación web y una aplicación móvil. Empezaremos viendo en detalle la aplicación web, la más completa de las dos.

Para mostrar correctamente ambas, haremos primero una pequeña introducción sobre el alcance original y el alcance final de la aplicación, para después ver con más detalle las diferentes opciones, flujos y funcionalidades que poseen cada una de ellas.

4.3.1. Alcance

La aplicación web de Easy Sports, permite administrar competiciones en formato liga y comprende la totalidad de las cuatro fases que forman parte del ciclo de vida de una competición, estas son:

- **Creación:** En esta fase el administrador entra en la aplicación para crear la competición, establece los parámetros básicos, el tipo, la cantidad de jugadores, los equipos, la información a aportar, etc... y finalmente crea la competición con un Nick y contraseña que deberá compartir con los jugadores a los que quiera permitir unirse a la competición.
- **Incorporacion y validación:** Es la fase en la que los jugadores podrán unirse a la competición empleando las credenciales proporcionadas por el creador/administrador de la competición y aportar la información requerida, la cual será validada posteriormente por el administrador, el cual podrá aceptar o rechazar jugadores y cerrar los equipos o competiciones (no permitir el acceso de nuevos jugadores).
- **Configuración y generación:** En este punto el administrador debe encargarse de configurar la generación de la liga, autogenerar los enfrentamientos y editar los parámetros de cada uno de ellos si se deseara cambiar los valores establecidos por la autogeneración.
- **Transcurso de la competición:** Una vez generados los enfrentamientos, el administrador podrá iniciar la competición, y encargarse de editar los resultados una vez alcanzada la fecha de los partidos, también podrá finalizarla en el momento que lo desee.

Otras posibilidades dentro del alcance de la aplicación web es la de acceder al listado de las competiciones que han sido marcadas como públicas, para consultar sus enfrentamientos y clasificación, sin la necesidad de estar registrado en la aplicación.

4.3.2. La Web en detalle

En esta sección mostraremos detalladamente la composición de la App Web, es decir, las pantallas y opciones con las que nos encontraremos durante la ejecución entrando en algunos ejemplos del desarrollo para ver mas concretamente como se trabaja en Outsystems®.³

Pantalla Principal

Empezaremos pues por la pantalla principal, la cual se muestra en la Figura 4.2. Esta es la pantalla a la que llegaremos directamente al acceder a la URL de la aplicación. Para acceder a esta pantalla no requerimos estar registrados, lo cual permite acceder a la funcionalidad “acceso público”. Clicando sobre la opción “Public Access”(1 en Figura 4.2) los usuarios no registrados podrán acceder para ver el progreso de las competiciones públicas.

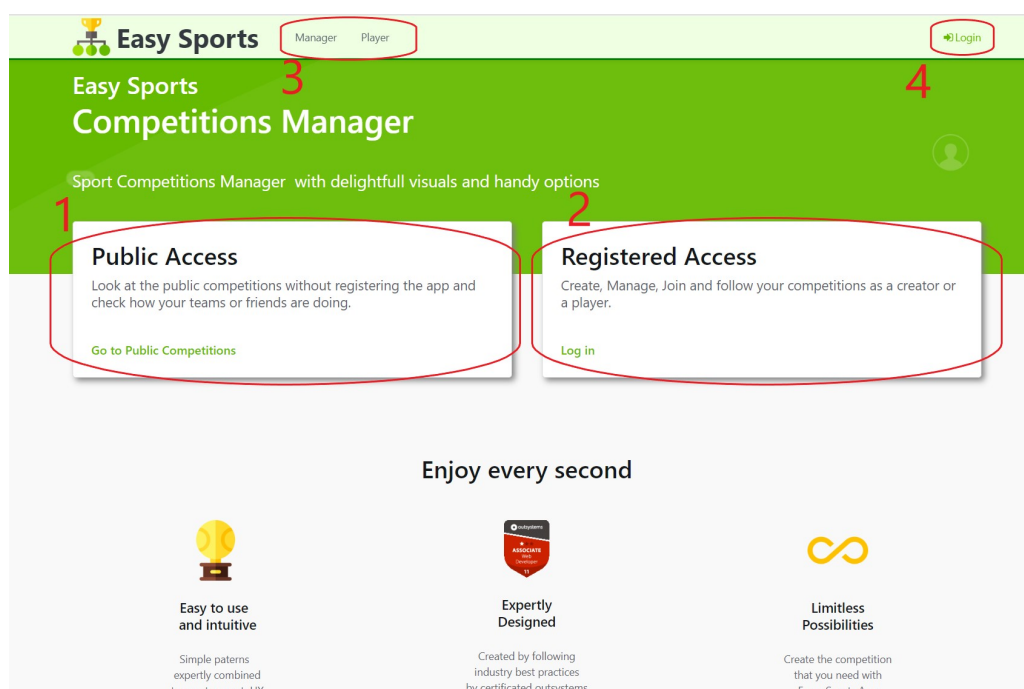


Figura 4.2: Pantalla Principal

Desde la opción “Acceso Registrado” (2 en Figura 4.2) pasaremos a la siguiente pantalla, la selección de Roles (Figura 4.4), para la cual deberemos estar registrados. Si no nos hemos registrado previamente empleando la opción de Login (4 en Figura 4.2) , se nos redigirá automáticamente a la pantalla de Login (Figura 4.3) para que lo hagamos, y así poder continuar.

³Para hacer más sencilla esta tarea se empleara la nomenclatura “x en Figura y” donde x será un entero que señale un elemento sobre la figura con referencia y

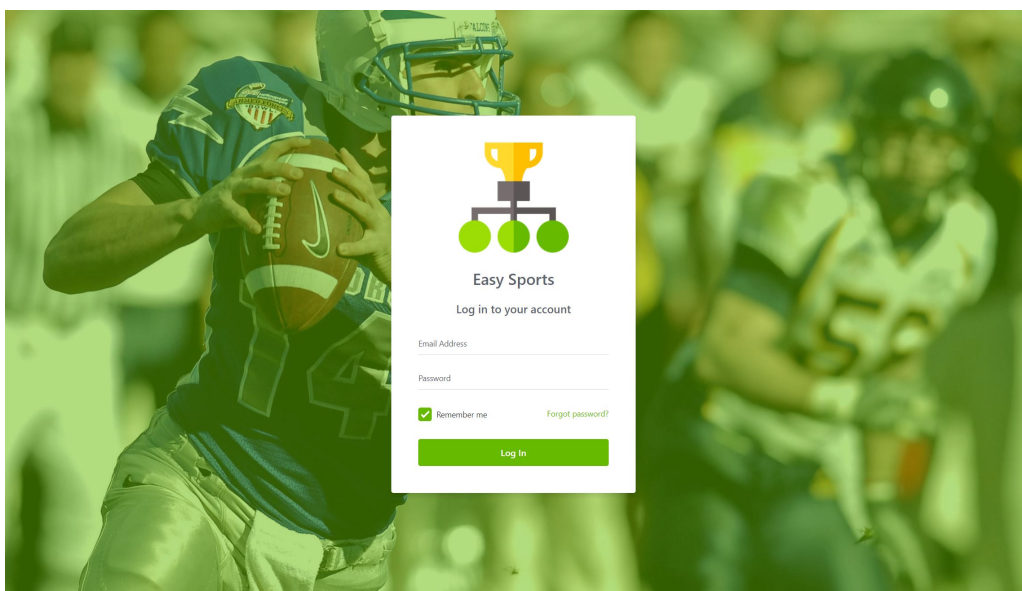


Figura 4.3: Pantalla Login

Por ultimo mencionar que disponemos de dos atajos en el “Top Menu” (3 en Figura 4.2) que nos permiten acceder directamente a las opciones de “Manager y “Player” redirigiéndonos directamente a las pantallas de las Figuras 4.5 y 4.27 respectivamente, pasando por la de Login si fuera necesario. Por último, mencionar que clicar sobre el rotulo Easy Sports en cualquier momento nos traerá directamente a esta pantalla.

Selector de Rol

Desde la opción Registered Access (2 en Figura 4.2) accederemos a la pantalla de selección de Rol, Figura 4.4.

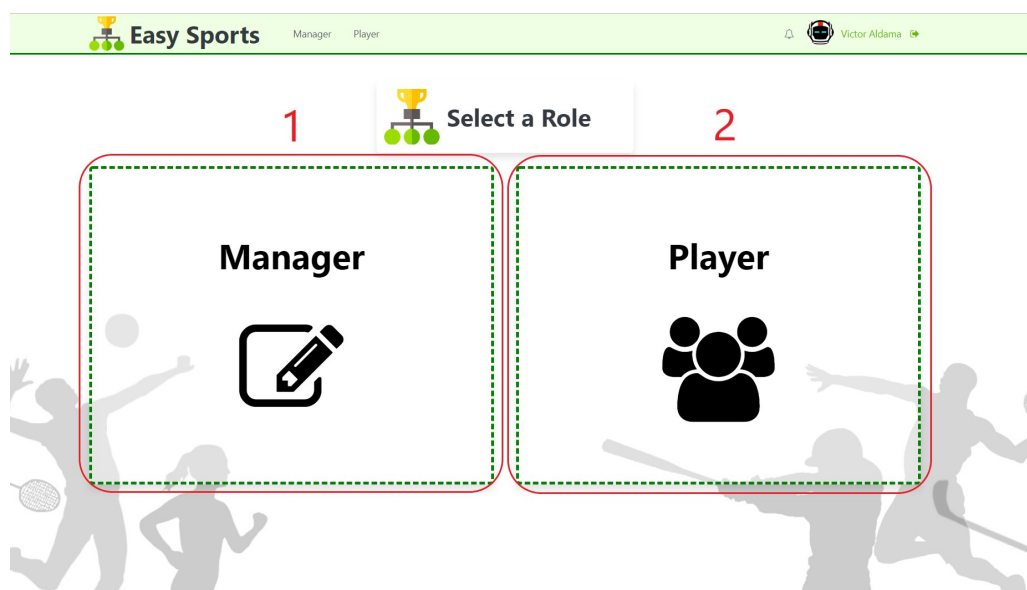


Figura 4.4: Pantalla Selección de Rol

Desde aquí elegiremos si las acciones que vamos a realizar a continuación irán relacionadas con la creación o gestión de las competiciones, accediendo a las opciones de manager (clicando sobre **1 en Figura 4.4**), o si por el contrario nuestra intención es unirnos o revisar nuestro progreso en una competición (clicando sobre **2 en Figura 4.4**).

Opciones de Manager

Si en la pantalla anterior hemos elegido la opción de Manager, seremos redirigidos a la pantalla de opciones para Manager (Figura 4.5), en la cual se nos ofrecen dos opciones principales.

- **Crear Competición:** Si accedemos a esta opción, clicando sobre **1 en Figura 4.5**, accederemos al formulario de creación de competiciones.
- **Gestionar Competiciones:** Si vamos a esta otra opción, clicando sobre **2 en Figura 4.5**, accederemos a la pantalla donde tendremos acceso a la gestión de todas las competiciones creadas por nosotros durante las diferentes etapas de estas.

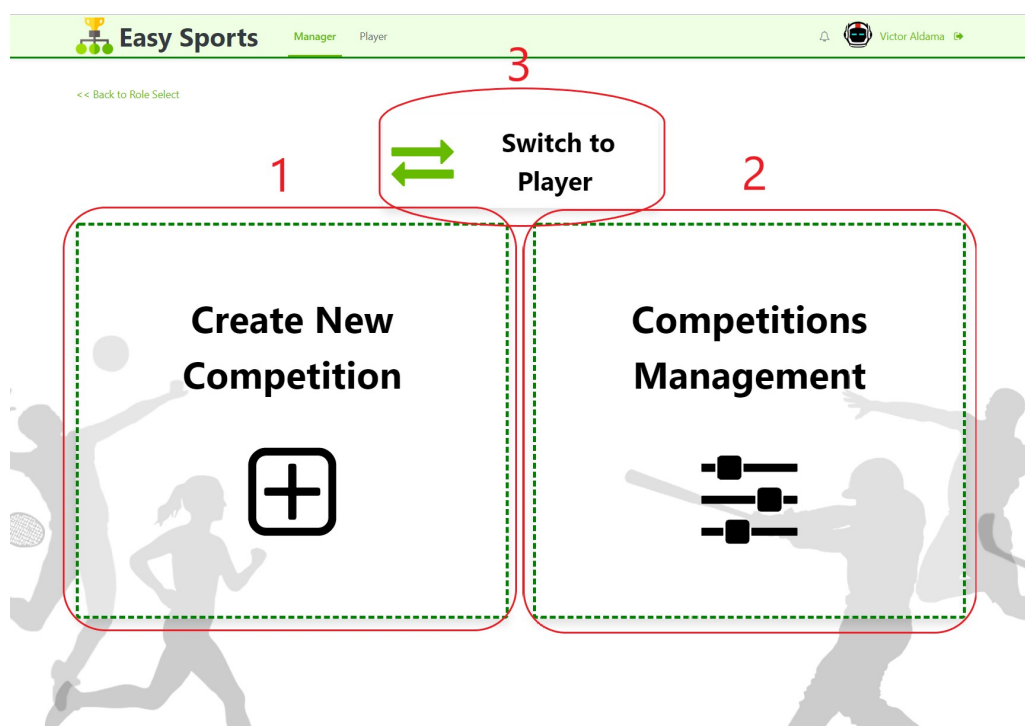


Figura 4.5: Opciones de manager

Si elegimos “Switch to Player” (clicando sobre **3 en Figura 4.5**) podremos acceder directamente a las opciones de jugador sin necesidad de retroceder al selector de roles.

Formulario de Creación

Esta pantalla es la que contiene el formulario para crear nuevas competiciones, aunque se trata del formulario más extenso de la aplicación, se ha tratado de facilitar la experiencia al usuario mediante distintos elementos gráficos, como una barra de navegación (o “wizard”), dividiendo el formulario en pasos más sencillos y adaptados al tipo de competición.

También a nivel lógico, esta pantalla a requerido gran cantidad de validaciones y controles para poder asegurar el correcto envío de la información y ofrecer las diferentes funcionalidades como añadir equipos o realizar peticiones de información personalizadas, por lo que mostraremos algunos ejemplos de los flujos de lógica que la sustentan.

Cuando accedemos a esta pantalla nos encontramos con los elementos visibles en la Figura 4.6, como se puede ver por defecto el wizard (**1 en Figura 4.6**) en su **Paso 1** posee cuatro pasos, los cuales podrán cambiar según el tipo de competición elegida en el paso siguiente (Figura 4.7). En este primer paso común para todas las competiciones, se nos solicita la información más básica, como puede verse en **2 de la Figura 4.6**.

The screenshot shows a web interface for creating a new competition. At the top, there is a breadcrumb trail: 'Manager Options > Creation Form'. Below it, the title 'New Competition' is displayed in green. A progress bar at the top indicates four steps: 1. General Settings (active), 2. Select Type, 3. Teams & Players, and 4. Sharing Settings. The main form area contains three input fields: 'Name *' (a text box), 'Sport *' (a dropdown menu), and 'Description' (a larger text area). A green 'Next' button is positioned at the bottom right of the form. The entire form is highlighted with a red oval labeled '2'. The progress bar is labeled '1' and the 'Next' button is labeled '3'. The background features silhouettes of athletes in various sports, including tennis, basketball, and baseball.

Figura 4.6: Paso 1 Formulario Creación

Pasamos al **Paso 2** (Figura 4.7), en este punto deberemos elegir que tipo de competición queremos (**2 en Figura 4.7**). Hay tres tipos:

- **Básica:** Es el más sencillo no incluye los pasos de “Required Data” ni de “Custom Fields”, por lo que no podremos solicitar más información a los jugadores que su nombre completo.
- **Seria:** Ahora ya disponemos del paso “Required Data” por lo que podemos seleccionar entre una lista de campos la información que queremos solicitar a los jugadores.

- **Oficial:** Con todos los pasos, este tipo dispone de la opción “Custom Fields” que permite solicitar información totalmente personalizada a los jugadores.

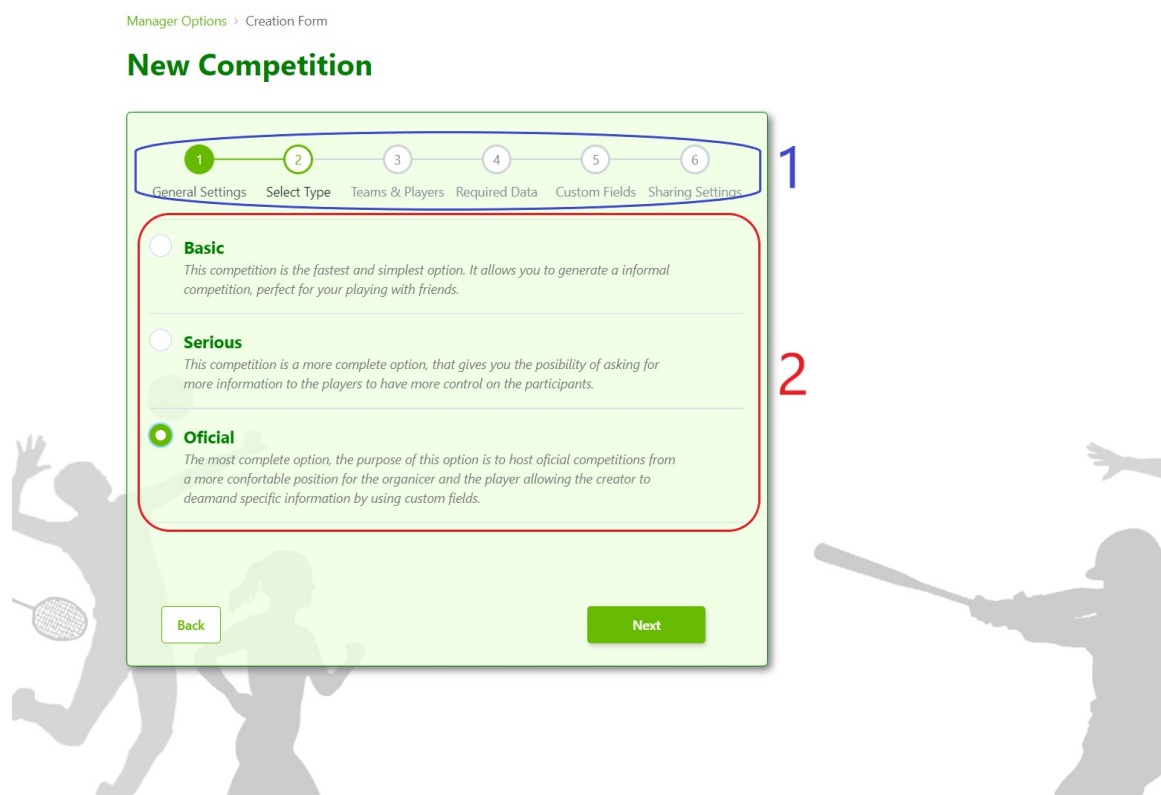


Figura 4.7: Paso 2 Formulario Creación

En este documento, seguiremos el flujo del tipo Oficial, que es el que posee todos los pasos (**1 en Figura 4.7**), para poder mostrar todas las funcionalidades de configuración que posee el formulario. Mientras no seleccionemos ningún tipo, el botón de “Next” permanecerá deshabilitado. Podemos usar el botón “Back” para regresar al paso anterior.

Si pasamos al **Paso 3**, deberemos seleccionar aquí si nuestra competición será por equipos o no empleando el “Switch” de **1 en Figura 4.8**.

Si nuestra competición no va a ser por equipos, deberemos indicar el máximo de jugadores (**2 en Figura 4.8**), y proseguir pulsando “Next”, que validará que exista un mínimo de 2 jugadores, y pasará al siguiente paso.

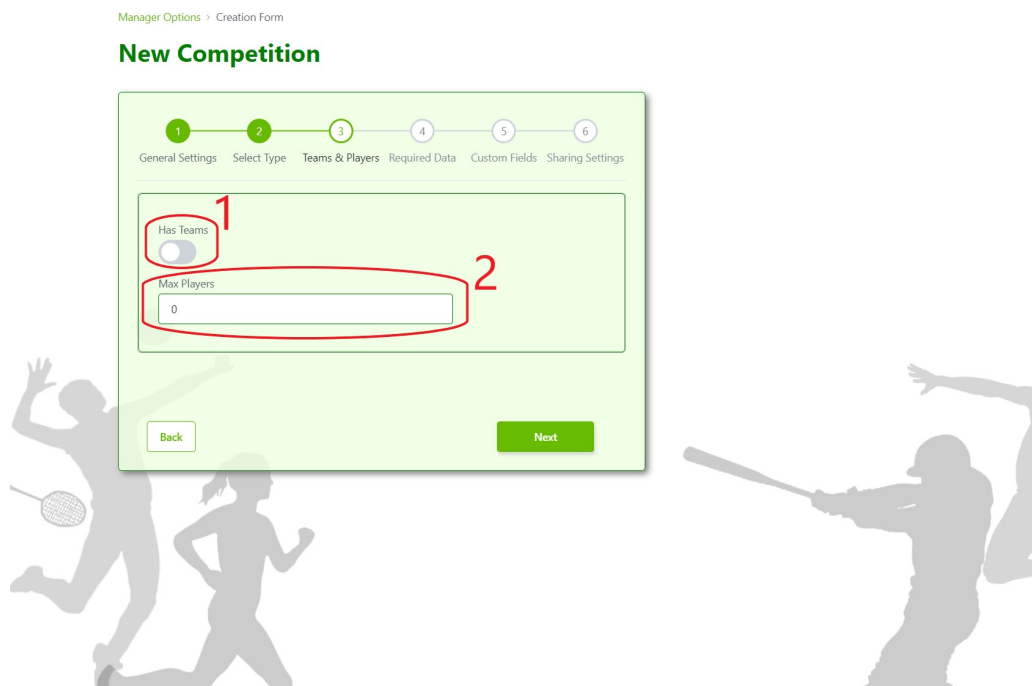


Figura 4.8: Paso 3.a Formulario Creación

Si en cambio, la competición es por equipos al indicarlo con el “Switch” de **1 en Figura 4.8**, el formulario cambiará y tendremos que indicar en **1 en Figura 4.9** el máximo de equipos y de jugadores por equipos.

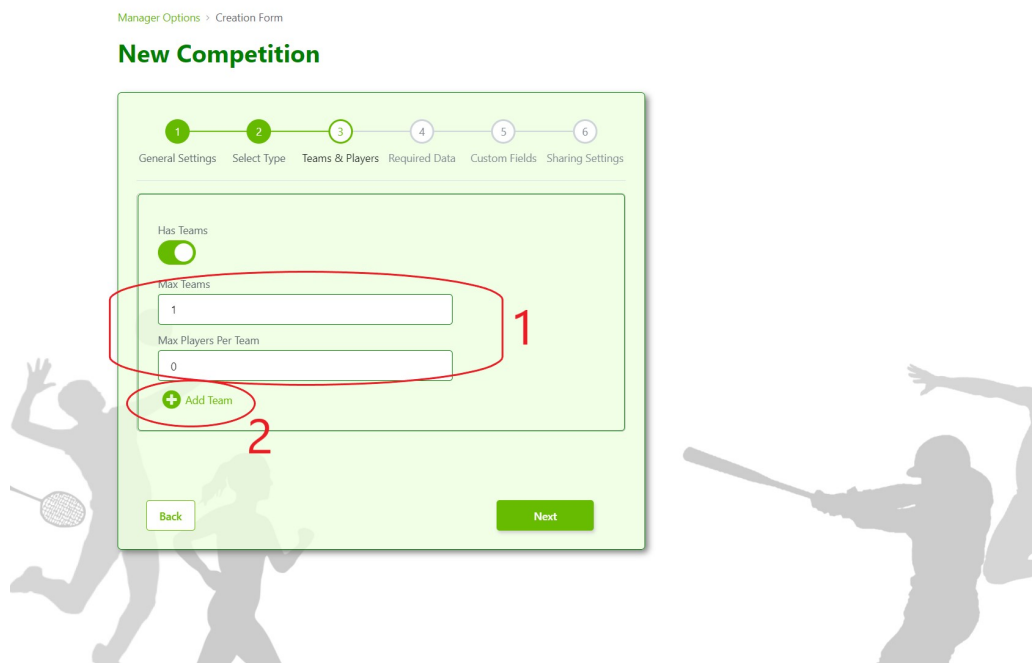


Figura 4.9: Paso 3.b Formulario Creación

Además, si clicamos sobre “Add Team” (**2 en Figura 4.9**) se abrirá un desplegable como muestra la Figura 4.10 que nos permitirá añadir los equipos indicando su nombre, hasta llegar al máximo de los admitidos en la competición.

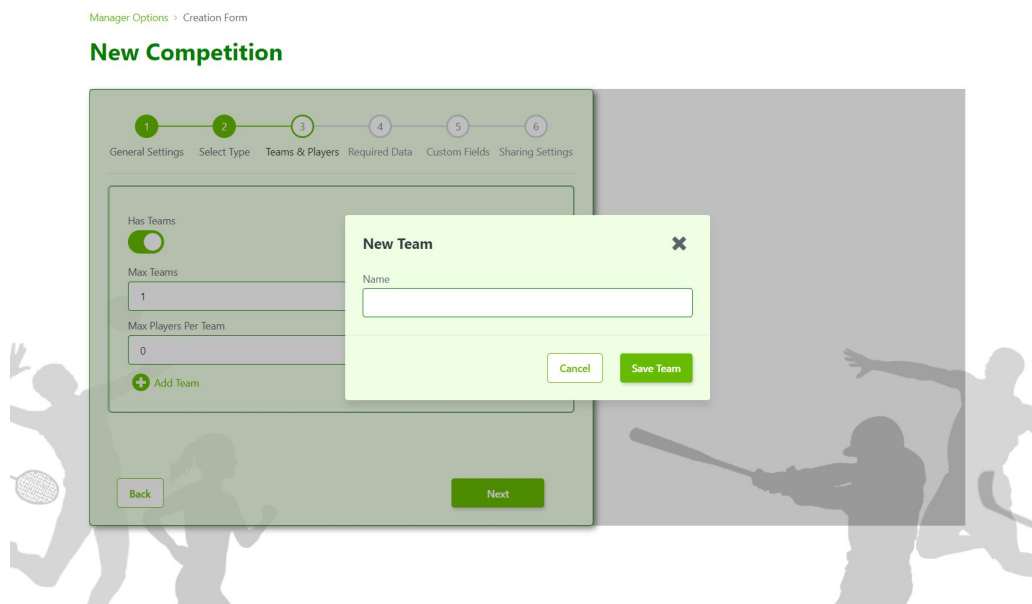


Figura 4.10: Paso 3.c Formulario Creación

Tras añadirlos se mostrarán como en **1 en Figura 4.11** donde podremos eliminarlos si deseamos.

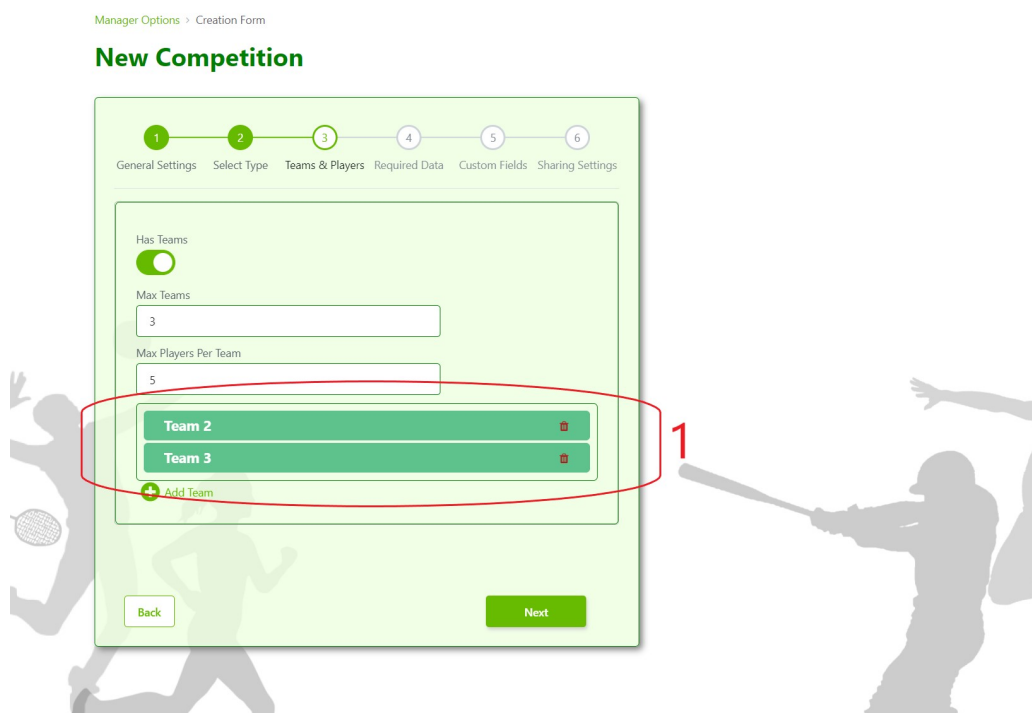


Figura 4.11: Paso 3.d Formulario Creación

Para pasar al siguiente paso deberemos seleccionar “Next”, que se encargará de realizar las validaciones necesarias y de almacenar la información de los equipos para crearlos cuando finalmente creemos la competición.

Llegamos al **Paso 4**, este paso está disponible únicamente en competiciones serias u

oficiales, es un paso sencillo en el que deberemos marcar los “CheckBox” con la información que deseemos solicitar a los jugadores como muestra la Figura 4.12.

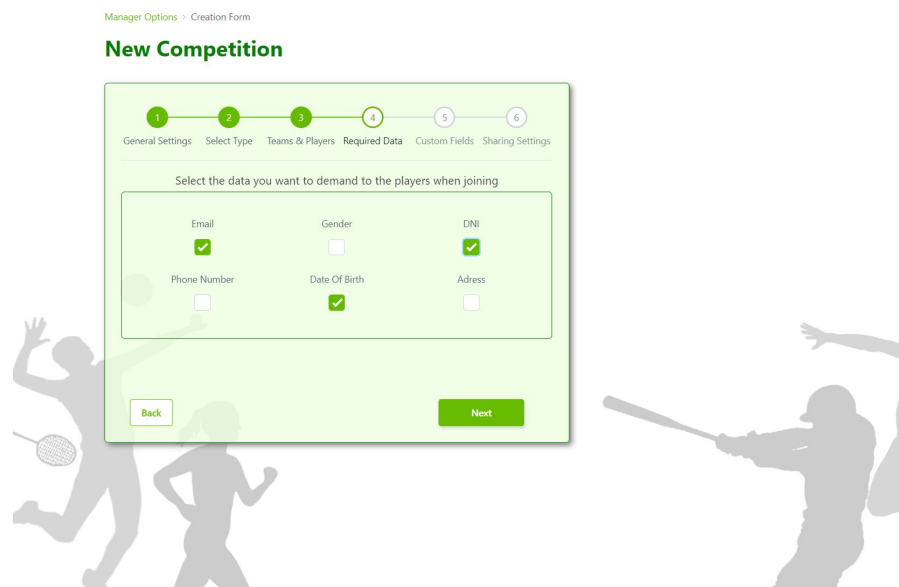


Figura 4.12: Paso 4 Formulario Creación

El **Paso 5**, es exclusivo de las competiciones de tipo oficial. Como se ve en la Figura 4.13, al acceder a este paso nos encontramos con una lista vacía de campos personalizados.

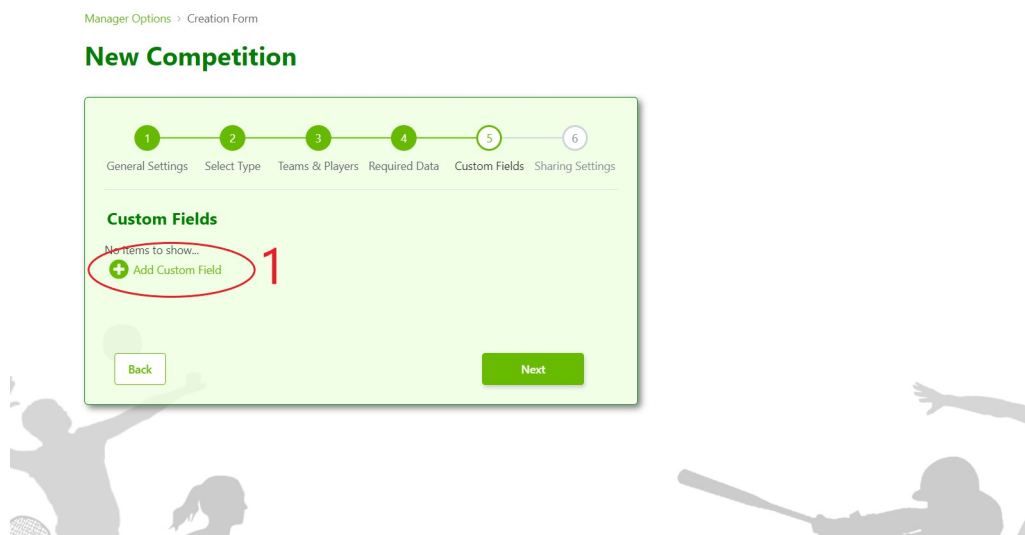


Figura 4.13: Paso 5.a Formulario Creación

Disponemos de la acción de añadir campos personalizados “Add Custom Field”(1 en **Figura 4.13**), que al abre un desplegable (Figura 4.14), donde podemos indicar el nombre y la obligatoriedad del campo.

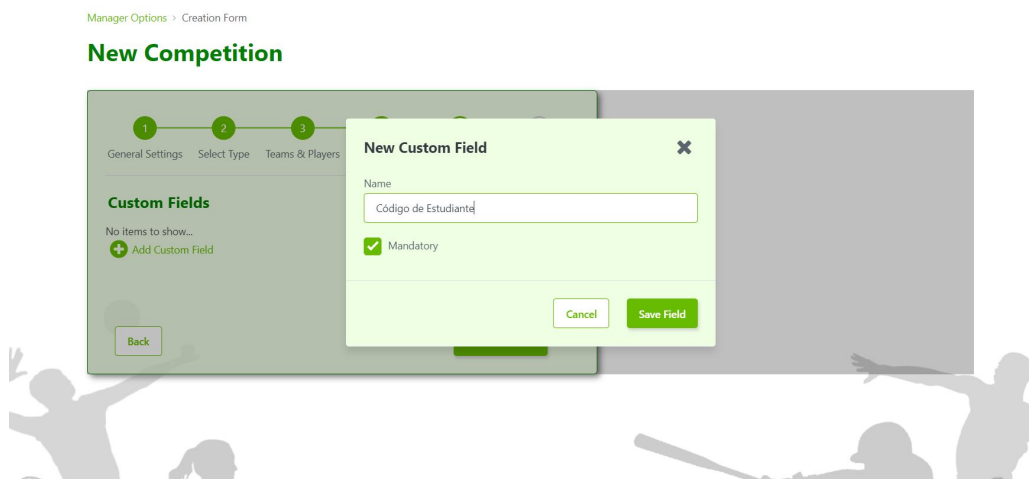


Figura 4.14: Paso 5.b Formulario Creación

De nuevo como en el Paso 3 al guardar el un campo se añade a una lista desde donde podemos revisar si es obligatorio o no y borrarlo si fuera necesario (Figura 4.15).

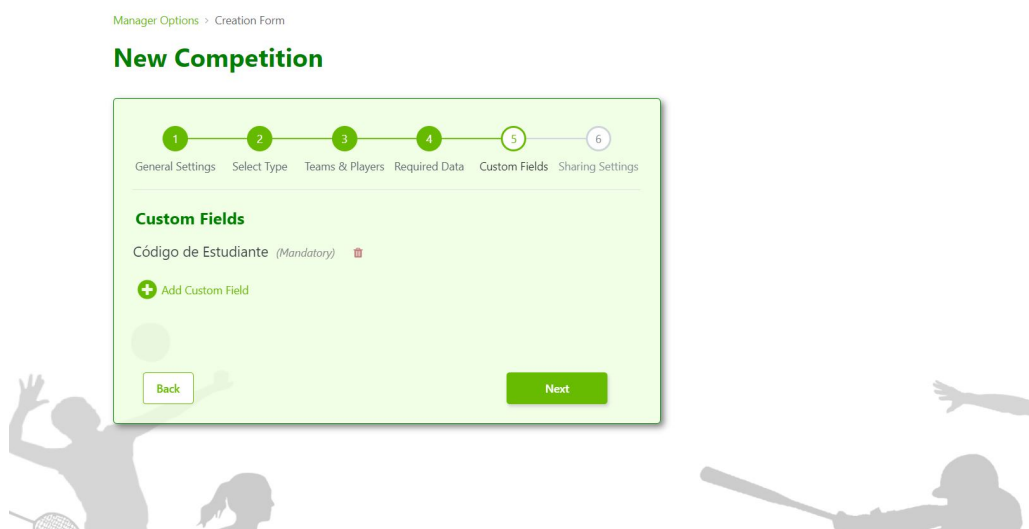


Figura 4.15: Paso 5.c Formulario Creación

Finalmente llegamos al **Paso 6**, en el cual obtenemos un Nick autogenerado, que nos permitirá dar acceso a la competición a los jugadores (**1 en Figura 4.16**), también disponemos de un “CheckBox” (**2 en Figura 4.16**) para establecer si la competición aparecerá más adelante entre las listadas en “Public Access” (Figura 4.31) y un sistema par establecer una contraseña para la competición (**3 en Figura 4.16**).

Manager Options > Creation Form

New Competition

1 2 3 4 5 6

General Settings Select Type Teams & Players Required Data Custom Fields Sharing Settings

Competition Nickname
TornBa1 1

Public 2

Password 3

Confirm Password

Back Save Competition 4

Figura 4.16: Paso 6 Formulario Creación

Para generar la competición debemos clicar “Save Competition” (4 en **Figura 4.16**). lo que iniciara el proceso de validación y posterior creación de la competición y de todos los equipos y demás parámetros establecidos en el formulario.

Deberemos compartir tanto el Nick como la contraseña con aquellos jugadores que deseemos que se unan a la competición.

Las figuras 4.17 y 4.18 presentan respectivamente partes de los flujos correspondiente a la validación de los pasos y la creación de la competición para ejemplificar como se trabaja la lógica dentro de OutSystems®.

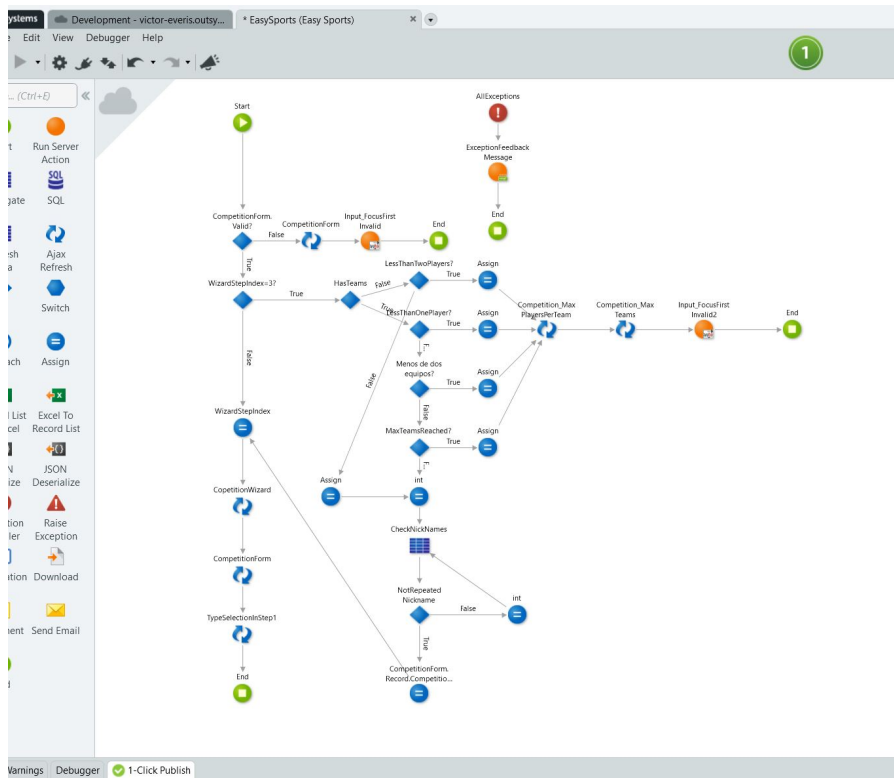


Figura 4.17: Parte de la Logica de Validación

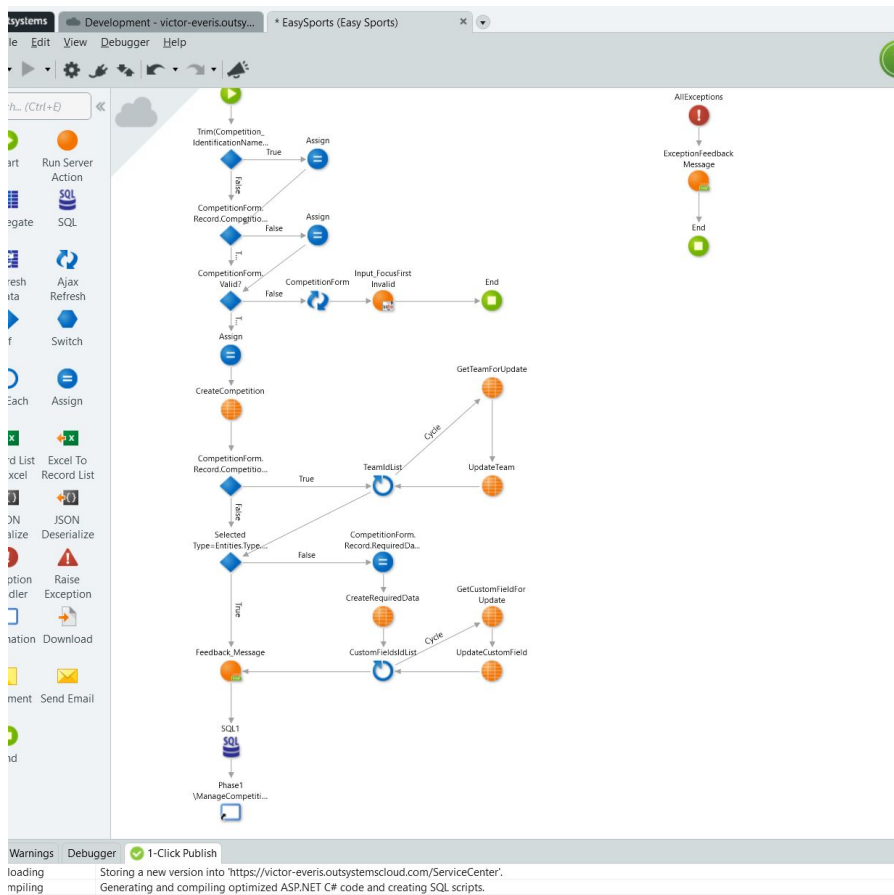


Figura 4.18: Parte de la Logica de Creación

Pantalla de Administración

Si elegimos **2 en Figura 4.5**, en las opciones de manager accedemos a la pantalla de administración (Figura 4.19). En esta pantalla disponemos de listados con las competiciones que hemos creado divididas en grupos dependiendo de su estado.

The screenshot shows the 'Manage Competitions' interface. At the top, there's a navigation bar with 'Easy Sports', 'Manager', 'Player', and a user profile 'Victor Aldama'. Below is a breadcrumb 'Home > Manage Competitions' and a title 'Manage Competitions'. The main content is divided into three sections:

- Waiting Players:** Contains a search bar and a table with columns: Name, Nick, Sport, Players. One record is shown: 'Senior Masculino Zonal F.V.B' with Nick 'SeniBa1', Sport 'Baloncesto', and 1/36 players. A trash icon is visible next to the record.
- Configuration pending:** Contains a search bar and a table with columns: Name, Sport, Players. One record is shown: 'Liga Teleco de Fútbol Mixto' with Sport 'Fútbol' and 4/40 players. A refresh icon is visible next to the record.
- In Progress:** Contains a search bar and a table with columns: Name, Sport, Progress. Two records are shown: 'Liga Cervecera' (Baloncesto, 67% progress) and 'Liga Padel Algiros' (Padel, 33% progress). Progress indicators are shown as circular gauges.

Figura 4.19: Pantalla de Administración

- Esperando jugadores:** En esta lista se encuentran las competiciones que están abiertas para que se unan nuevos jugadores. Se presenta el nombre, nick, deporte y número de jugadores unidos respecto del máximo permitido. Si clicamos sobre el nombre, (**1 en Figura 4.19**) accederemos a la pantalla de validación de jugadores/equipos (Figura 4.20), si lo hacemos sobre **2 en Figura 4.19**, tendremos la opción de borrar por completo la competición.
- Pendiente de Configuración:** Las competiciones que se listan en esta sección son aquellas cuyos jugadores o equipos ya se han unido y han sido validados, y se encuentran a la espera de que se configuren los enfrentamientos. Clicando sobre el Nombre, (**3 en Figura 4.19**) accederemos a la pantalla de configuración de enfrentamientos (Figura 4.22) y lo hacemos sobre **4 en Figura 4.19**, podremos reabrir la competición de nuevo para que se unan más jugadores, lo que devolverá la competición al listado anterior.
- En Progreso:** Las competiciones ya configuradas y marcadas como empezadas, se mostrarán aquí. Desde **5 en Figura 4.19**, podremos acceder a los detalles de la competición (Figura 4.24), y en **6 sobre Figura 4.19** tenemos un pequeño círculo de progreso que nos indica lo avanzada que se encuentra la competición.

Pantallas de Validación

Tras clicar sobre el nombre en **1 en Figura 4.19**, accedemos a la pantalla de validaciones, donde el manager de la competición deberá comprobar la información proporcionada por los jugadores y validarlos o rechazarlos dependiendo de si es correcta o no. Existen dos escenarios distintos, cuando la competición es por equipos y cuando no lo es.



Figura 4.20: Pantalla de Validación

En caso de que no lo fuera se accederá directamente a la pantalla de la Figura 4.20, donde al clicar sobre “Validate Data” (**1 en Figura 4.20**) se desplegará la sección **2 en Figura 4.20**, la cual puede esconderse si se desea con **3 en Figura 4.20**, en la que podremos ver la información proporcionada por el jugador y validarla o rechazarla en función de su veracidad u otros criterios.

En caso de que el jugador fuera validado se mostrará un “tick” verde al lado de su nombre (**4 en Figura 4.20**) y “Validate Data” cambiará por “Review Data” (**5 en Figura 4.20**) que realizará la misma acción que **2 en Figura 4.20** pero sin esta vez el desplegable no tendrá los botones que permiten validar o rechazar, de querer recuperar estas opciones deberemos clicar en desbloquear (el candado junto a “Review Data”), lo que devolverá al jugador a un estado pendiente de validación.

Las competiciones individuales al terminar con las validaciones habilitan el botón de **6 en Figura 4.20** con la opción de directamente cerrar la competición para pasar a la configuración de los enfrentamientos, sin embargo, como en este ejemplo las competiciones por equipo solo te

permiten cerrar el equipo, es cuando todos los equipos están cerrados cuando tenemos la opción de cerrar la competición para configurarla. El acceso a los distintos equipos se gestiona desde una pantalla intermedia entre la actual y la mostrada en la Figura 4.19, la pantalla del listado de equipos (Figura 4.21). Es desde 1 en esta pantalla desde donde accedemos a la pantalla de validación mostrada anteriormente.

Name	Status	Joined Players
C.B. Amics	Validation Pending	0 / 12
C.B.C	Validation Pending	0 / 12
EBV	Validation Pending	2 / 12

Figura 4.21: Listado para la validación de equipos

Configuración de enfrentamientos

Desde 3 en Figura 4.19 podemos acceder a la pantalla de configuración de las competiciones que ya han pasado el proceso de validación, para poder configurar los enfrentamientos en la pantalla que muestra la Figura 4.22 .

Competition Configuration

1 Match Generation

[Auto Generate Matches](#) [Configure Competition Details](#)

2 Competition Details

Name: Liga Teleco de Fútbol Mixto **Has Teams:** ✓
Creation Date: 2019-09-02 **Created By:** Victor Aldama
Description: Liga de fútbol para estudiantes de Telecomunicaciones.
 AVISO: No se admitirán jugadores desligados de la ETSIT UPV.

3

Round: 1				
Equipo Imagen y Sonido	VS	Equipo Telemática	Pabellón UPV	11-09-2019
Equipo Sistemas	VS	Equipo Electrónica	Pabellón UPV	09-09-2019

Round: 2				
Equipo Electrónica	VS	Equipo Imagen y Sonido	Pabellón UPV	15-09-2019
Equipo Telemática	VS	Equipo Sistemas	Pabellón UPV	13-09-2019

Round: 3				
Equipo Sistemas	VS	Equipo Imagen y Sonido	Pabellón UPV	19-09-2019

Figura 4.22: Configuración de los enfrentamientos

En esta pantalla podemos, clicando **1 en Figura 4.22**, generar automáticamente los enfrentamientos. Esta autogeneración, puede configurarse desde un formulario al que se accede desde **2 en Figura 4.22**, donde estableceremos los rangos de fechas, si la competición tiene ida y vuelta, cual es el emplazamiento por defecto, etc. En este formulario también podemos establecer cuál era el sistema de puntuación de la competición (Figura 4.23).

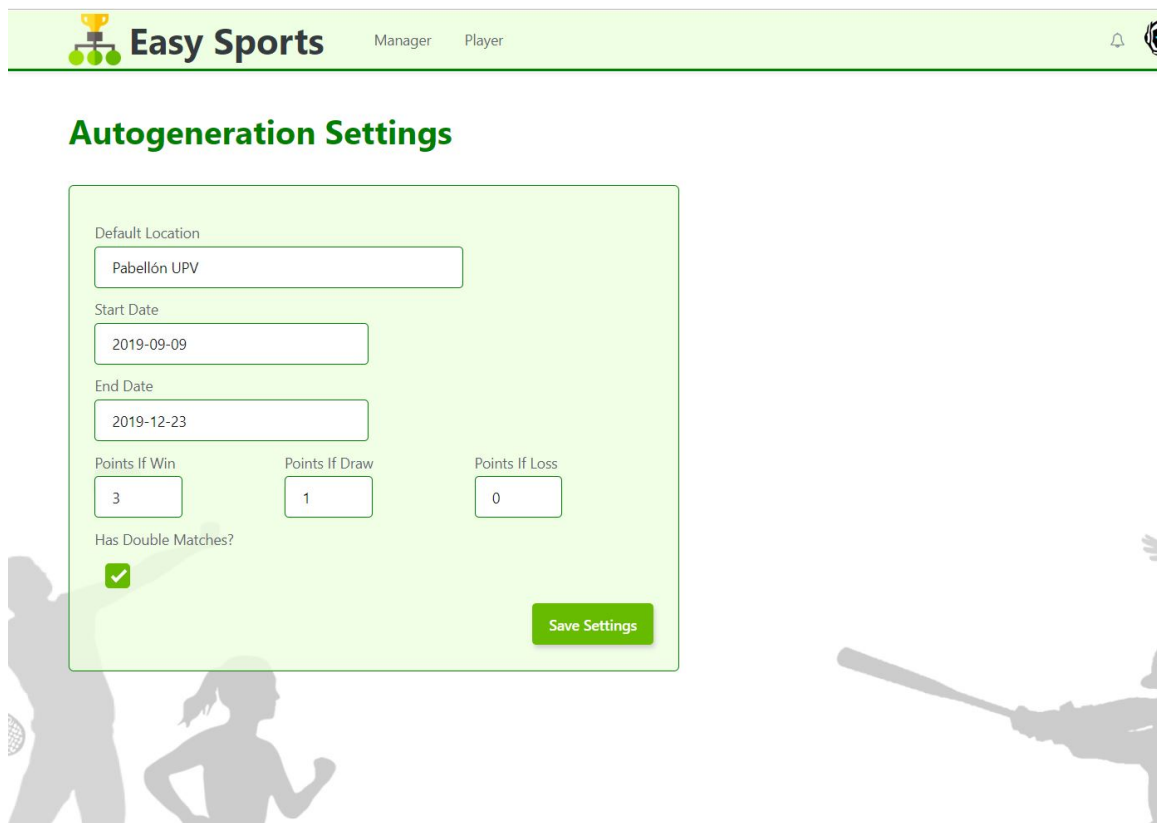


Figura 4.23: Opciones de competición

El proceso y algoritmo detrás de la autogeneración de los cruces han requerido de un desarrollo externo en C# que se detalla en el Anexo de este documento.

Desde **3 en Figura 4.22** podemos editar el emplazamiento y la fecha del encuentro si lo deseáramos.

Una vez generados los partidos podemos guardarlos y dar por comenzada la competición (estas opciones no se muestran en la Figura 4.22 pero existen al final de la página).

Competiciones en Progreso

Como última opción desde la pantalla de administración seleccionando **3 en Figura 4.19**, accederemos a la pantalla con los detalles del progreso de la competición. Esta pantalla incluye dos pestañas, una con la clasificación (Figura 4.24) y otra con una “timeline” de los partidos (Figura 4.25).

Player	Points	Played Matches	Played As Local	Played As Visitor
1.- Mary Jane	3	1/4	1	0
2.- Victor Aldama	3	2/4	0	2
3.- Will Smith	0	1/4	1	0

End Competition

Figura 4.24: Clasificación

Next Matches

- 1 Will Smith vs Mary Jane
31 Aug 2019
C/ Serpis 43
- Victor Aldama vs Mary Jane
02 Sep 2019
C/ Serpis 43
- Mary Jane vs Will Smith
02 Sep 2019
C/ Serpis 43
- Victor Aldama vs Will Smith
02 Sep 2019
C/ Serpis 43

Figura 4.25: Timeline

En esta pantalla el administrador puede editar los resultados de los partidos cuya fecha sea la del día actual o esté en el pasado clicando sobre **1 en Figura4.25** y abriendo el desplegable de la Figura 4.26.

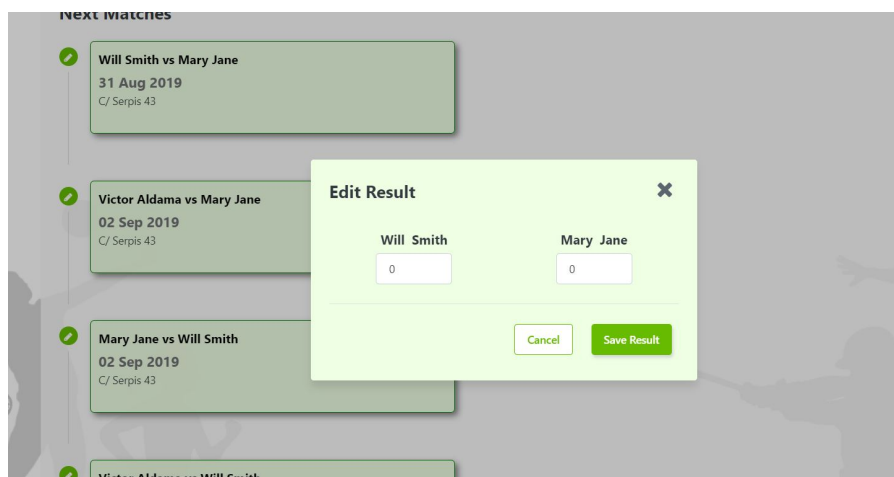


Figura 4.26: Edición de los resultados

Hasta ahora hemos recorrido las opciones como manager, ahora retornaremos a la selección de rol (Figura 4.4) para tomar el camino con las opciones del jugador (2 en Figura 4.4).

Opciones de Jugador

Nos encontramos ahora frente a una pantalla muy similar a la que contiene las opciones de mánager, tenemos de nuevo la funcionalidad “Switch” que en este caso nos lleva directo a las opciones de mánager.

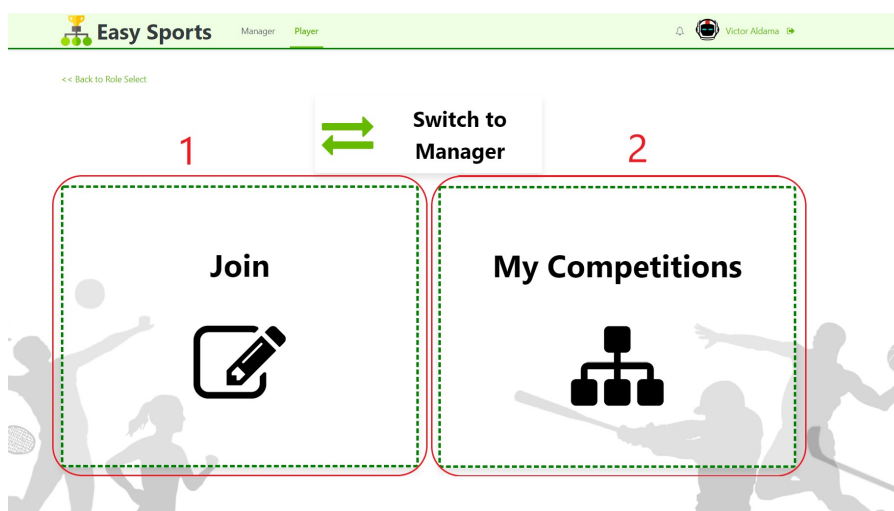


Figura 4.27: Opciones de Jugador

En esta pantalla, clicando sobre **1 en Figura 4.27** accederemos a la pantalla de inscripción (Figura 4.28) donde si disponemos de las credenciales de una competición podremos unirmos a esta, por otro lado, clicando sobre **2 en Figura 4.27**, accederemos a una pantalla muy similar a la de administración donde se listan las competiciones en las que estamos inscritos.

Pantalla de inscripción

Al acceder a esta pantalla nos encontramos un pequeño formulario que nos solicita las credenciales de la competición, como se muestra en la Figura 4.28 .

Figura 4.28: Pantalla de inscripción

Si las introducimos correctamente nos aparecerá una ventana con un formulario más detallado (Figura 4.29) requiriendo los datos necesarios para inscribirse, que dependiendo del tipo de competición se compondrá de más o menos pasos.

Figura 4.29: Formulario pantalla de inscripción

Una vez enviada la información ya estaremos inscritos en la competición, a falta de ser validados, y seremos redirigidos a la pantalla de “Tus Competiciones”.

Pantalla “Tus competiciones”

Esta pantalla contiene un listado muy similar al que mostraba la pantalla de administración (Figura 4.19), pero esta vez solo encontramos dos grupos de competiciones, las que están esperando la inscripción de jugadores o que se configuren los cruces y las que ya han empezado.

Easy Sports Manager Player Victor Aldama

Home > Your Competitions

Your Competitions

Waiting To Start

Search by Name...

Name	Sport	Max Teams	Joined Players	Status	Your Status
Liga Teleco de Fútbol Mixto	Fútbol	4	4/40	Configuration Pending	Accepted
Senior Masculino Zonal F.V.B	Baloncesto	3	2/36	Waiting Players	Validation Pending

2 records

In Progress

Search by Name...

Name	Sport	Progress
Liga Cervecera	Baloncesto	67 %
Liga Padel Aljirós	Padel	33 %

2 records

Figura 4.30: Pantalla “Your Competitions”

Como se puede ver en la Figura 4.30, las listas nos proporcionan información sobre la competición o nuestro estado en ella. En el caso de las que están a la espera de jugadores o de ser configuradas, se nos muestra en que paso se encuentra la competición y cuál es nuestro estado dentro de ella, en el caso de que este empezada podemos ver directamente el progreso de la competición.

Al igual que la pantalla de administración (Figura 4.19), desde la primera lista podemos acceder al listado de equipos o jugadores que se han inscrito hasta el momento, y desde la segunda podemos acceder a las pantallas con los detalles de la competición (Figuras 4.24 y 4.25), pero esta vez no dispondremos de ningunas de las acciones que quedan reservadas para el administrador, como validar o editar resultados.

Competiciones públicas

Por último, tenemos la pantalla de competiciones públicas, a la cual accederemos desde **1 en Figura 4.2** y en la cual tenemos ahora una sola lista con las competiciones en juego desde la que accederemos a los detalles de estas (clasificación y timeline). En esta pantalla (Figura 4.31), solo aparecerán las competiciones que se hayan marcado como públicas en el formulario de creación (Figura 4.16).

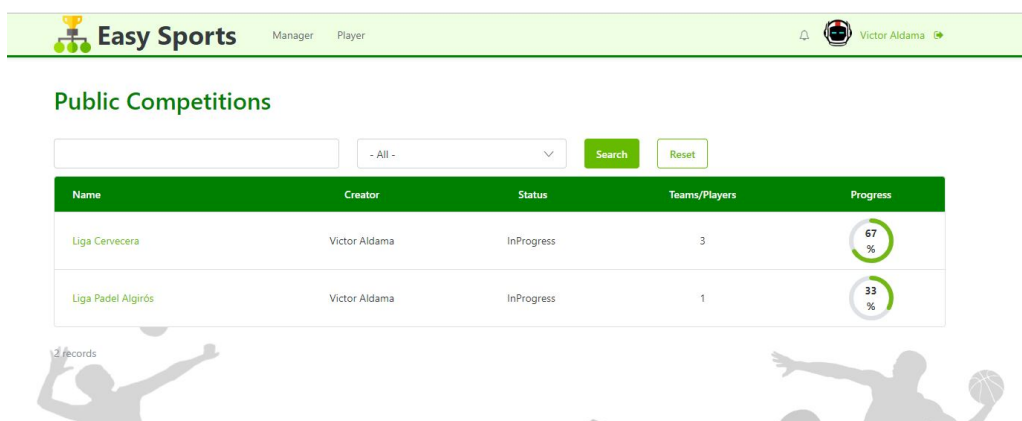


Figura 4.31: Competiciones Públicas

Pantalla de datos de Usuario

También siempre que, desde cualquier interfaz donde la barra de menú superior (“top-bar”) sea accesible, cliquemos sobre nuestro nombre o foto de perfil (parte superior derecha) accederemos a la pantalla de datos de usuario, donde podremos editar o rellenar nuestra información personal y foto de perfil (Figura 4.32).

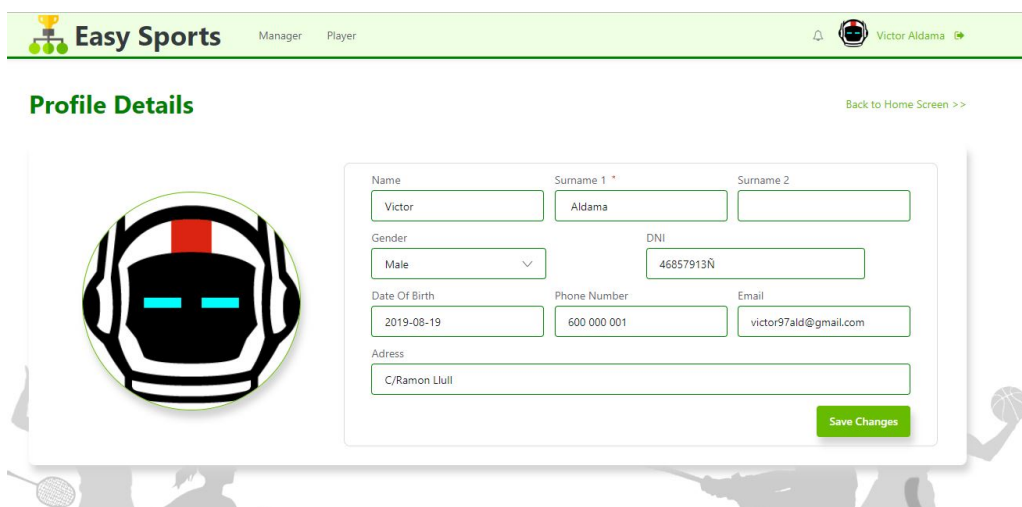


Figura 4.32: Datos de Usuario

Además de las interfaces mostradas, existen una gran cantidad de ventanas emergentes para la confirmación de diferentes acciones, y muchos mensajes de error configurados en caso de no superar las validaciones en los diferentes formularios.

4.4. Aplicación Móvil

La versión Movil de Easy Sports como se ha mencionado con anterioridad se alimenta de la misma base de datos de la que lo hace la aplicación web.

Aunque seria posible realizar un desarrollo con OutSystems[®], en el cual contempláramos un escenario offline, por simplicidad se ha optado por un desarrollo que necesite de conexión a internet para funcionar, ya que esto nos permitirá estar siempre actualizados con los cambios más recientes y evitará que nuestro teléfono de almacene información de forma local.

4.4.1. Alcance

El alcance de la funcionalidad de la aplicación móvil, se reduce considerablemente respecto a la de la aplicación web, en este caso desde el móvil seremos capaces de:

- **Unirnos a la competición:** Vamos a poder unirnos a la competición y aportar la información requerida. No será posible validar a los jugadores desde esta aplicación.
- **Ver el progreso de la competición:** Vamos a ser capaces de ver la clasificación y los resultados de los partidos de las competiciones a las que pertenecemos, en qué estado están aquellas que aun no han comenzado o revisar aquellas que ya han terminado. Sin embargo, no seremos capaces de modificar ningún aspecto de la competición o de los resultados de los enfrentamientos.

Por tanto, como hemos podido observar, el principal afectado por esta reducción es el administrador, ya que ninguna de sus funciones se encuentra presente en la aplicación móvil, esto se debe principalmente a dos motivos.

- **Tamaño de la pantalla:** Al disponer de un tamaño de pantalla mucho mas reducido en los dispositivos móviles, la creación de formularios que requieran muchos datos o de ventanas tipo “Pop-up para requerir información o solicitar aprobación resulta mucho más problemática o desagradable teniendo en mente al usuario final, quien percibirá un “display sobrecargado que producirá una experiencia de usuario negativa desde un primer momento al usar la aplicación. De esta forma reducimos los grandes formularios al ámbito web donde sin mucho más cómodos de gestionar.
- **Navegabilidad:** También pensando en la experiencia del usuario, en los dispositivos móviles la cantidad de información que podemos presentar por pantalla es reducida, esto implica que si incluimos muchas opciones vamos a requerir de muchas pantallas o despleables para contenerlas. Incrementar el numero de pantallas disponibles en una aplicación puede resultar excesivamente confuso y pude conducirnos a sentirnos “perdidos dentro de la aplicación. Por ello, es preferible reducir las funcionalidades a las principales o más necesarias para proporcionar una experiencia mas fluida y controlada a los usuarios.

4.4.2. La App Movil en detalle

Como se ha indicado anteriormente la funcionalidad de la aplicación móvil es reducida respecto a la funcionalidad de la web, por ello el número de pantallas de las que disponemos es considerablemente menor.

Pantallas de Login y Listado

En la aplicación móvil será necesario registrarse para poder acceder, por lo que la primera pantalla con la que nos encontraremos será la de “Login” (Figura 4.33).

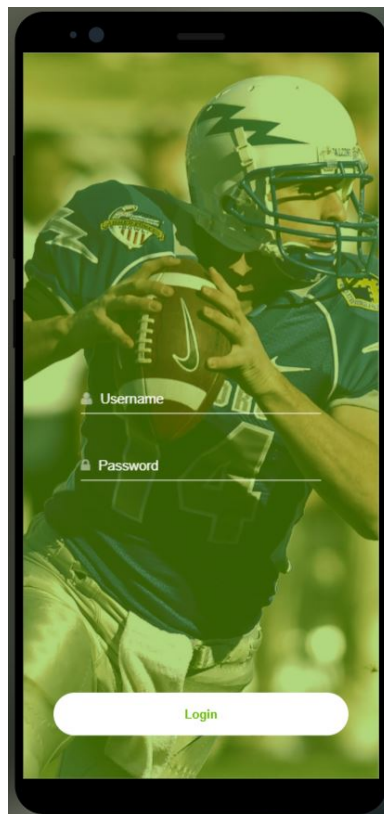


Figura 4.33: Login Movil

Una vez registrados seremos redirigidos al listado de competiciones, donde encontramos tres pestañas que organizan las competiciones de forma similar a como se hacía en la Web. (Empezadas, esperando jugadores y finalizadas) y cada pestaña se organiza individualmente por secciones en orden alfabético.

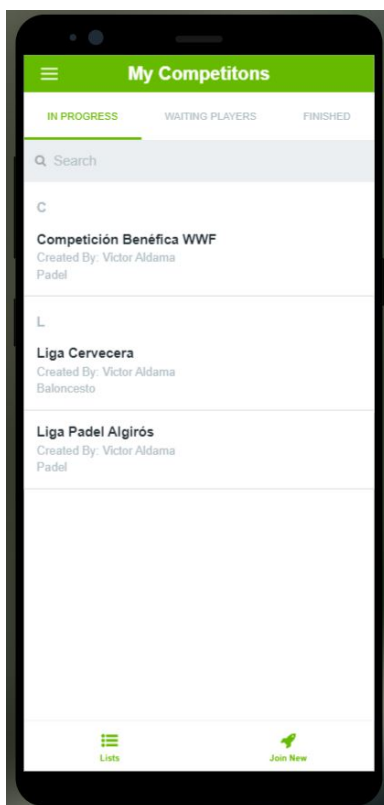


Figura 4.34: Listados Movil

En esta pantalla (Figura 4.34), solo las competiciones en la pestaña de “In progress” tienen funcionalidad. Al clicar sobre una de ellas accederemos a los detalles de dicha competición que de nuevo son la clasificación y un “timeline” como se muestra en la Figura 4.35.

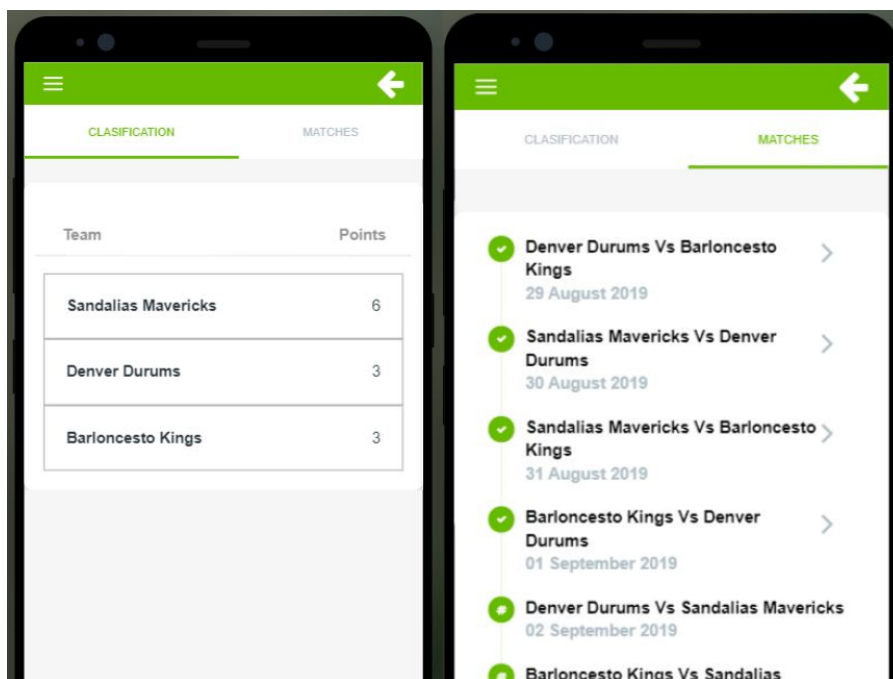


Figura 4.35: Detalles de Competición

Los ítems en la “timeline” que están marcados con un “tick” y que tienen una flecha en la

parte derecha, son aquellos enfrentamientos que ya se han jugado y de los que ya disponemos el resultado, el cual podemos consultar si clicamos sobre ellos accediendo a la pantalla de la Figura 4.36.

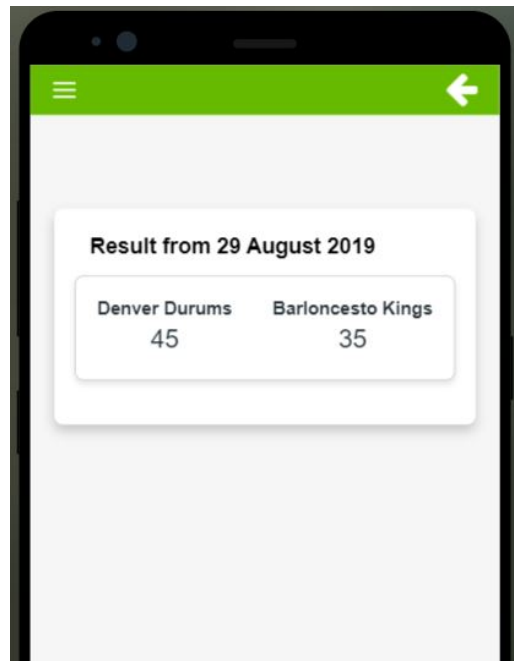


Figura 4.36: Detalle de Resultado

Pantalla y formulario de inscripción

Al igual que en la web, al acceder a esta pantalla nos encontramos un pequeño formulario que nos solicita las credenciales de la competición, como se muestra en la Figura 4.37.

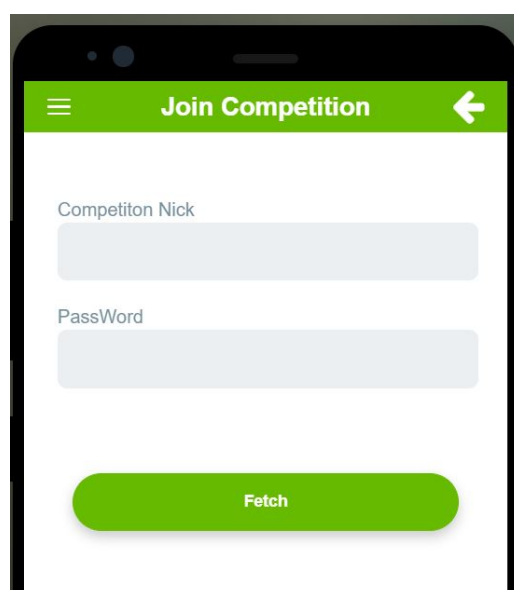


Figura 4.37: Pantalla de inscripción Movil

Si poseemos las credenciales correctas de una competición, al introducirlas navegaremos a una nueva pantalla, con un formulario más detallado (Figura 4.38) requiriendo los datos necesarios para inscribirse y que dependiendo del tipo de competición se compondrá de más o menos campos.

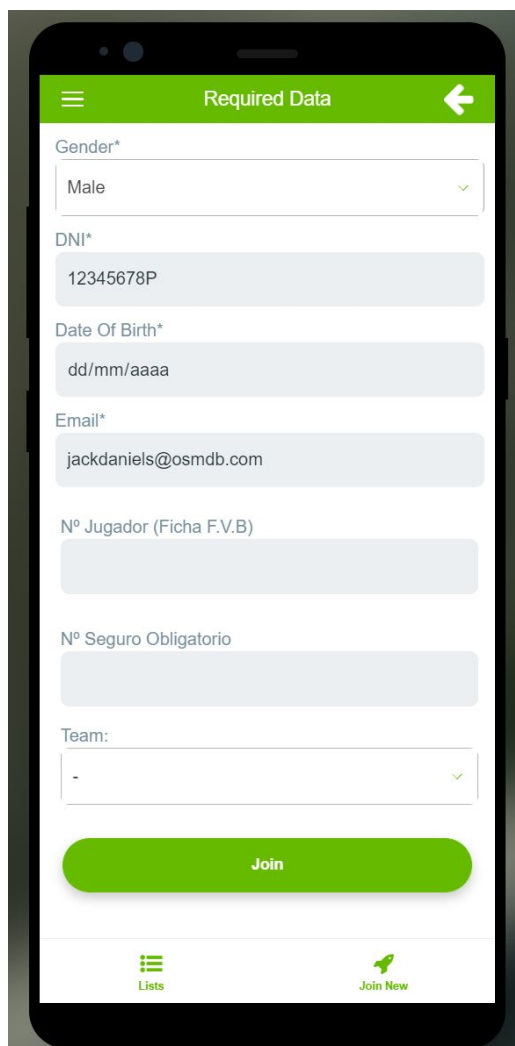
The image shows a mobile application screen titled "Required Data". At the top, there is a green header bar with a hamburger menu icon on the left, the text "Required Data" in the center, and a back arrow icon on the right. Below the header, the form contains several input fields: "Gender*" with a dropdown menu showing "Male"; "DNI*" with a text input field containing "12345678P"; "Date Of Birth*" with a text input field containing "dd/mm/aaaa"; "Email*" with a text input field containing "jackdaniels@osmdb.com"; "Nº Jugador (Ficha F.V.B)" with an empty text input field; "Nº Seguro Obligatorio" with an empty text input field; and "Team:" with a dropdown menu showing "-". At the bottom of the form is a large green rounded button labeled "Join". Below the form, there is a white bar with two icons: a hamburger menu icon labeled "Lists" and a green location pin icon labeled "Join New".

Figura 4.38: Formulario de inscripción Movil

Además de las pantallas mostradas, como se puede ver por ejemplo en la Figura 4.38, existen en la parte inferior (en la “Bottom-Bar”) dos accesos directos que permiten la navegación directa a los listados y a la pantalla de inscripción. También existe un menú desplegable desde la esquina superior izquierda o deslizando hacia la derecha que se muestra en la Figura 4.39, que posee los mismos accesos directos, información personal y la opción de Logout.

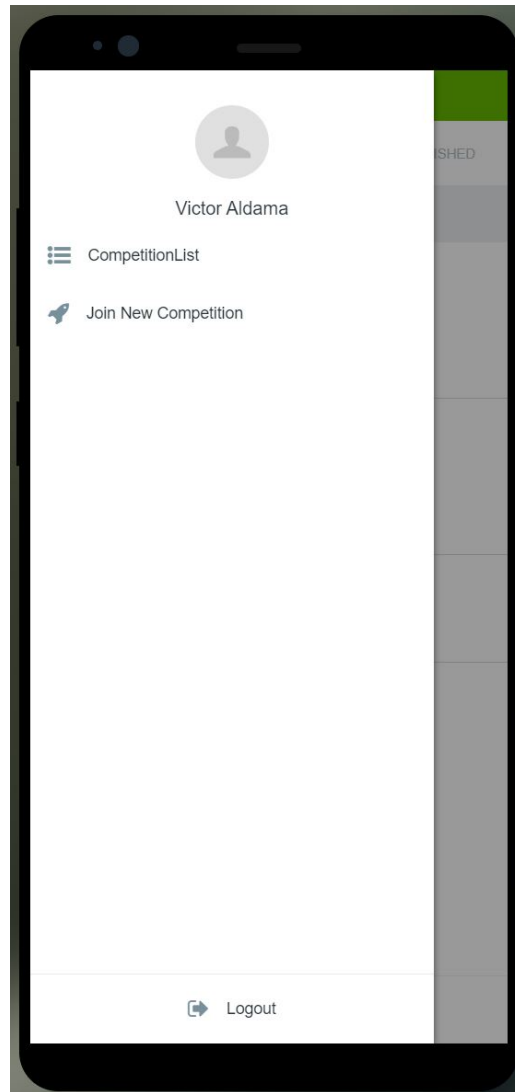


Figura 4.39: Menu Lateral

Capítulo 5

Conclusiones y líneas futuras

5.1. Conclusiones

Tras ver la potencia que posee una LCDP como OutSystems® para el desarrollo acelerado de aplicaciones y analizando mi experiencia durante el proceso, es posible extraer las siguientes conclusiones:

- Las plataformas Low-Code, realmente son una ventaja para la aceleración del proceso de desarrollo de aplicaciones, ya que permiten a una sola persona aumentar en gran medida su productividad, alcanzando a realizar en pocos días lo que un equipo de forma tradicional podría tardar semanas y supone la posibilidad para perfiles con mas visión de conjunto de relizar un desarrollo “full-stack”.
- A pesar de simplificar desarrollo, plataformas como OutSystem® siguen requiriendo y permiten codificación para algunos escenarios, esto como hemos visto puede ser una ventaja para añadir funcionalidad, pero al mantenernos generalmente abstraídos del código generado, resulta un gran salto en el nivel de técnica necesaria para realizar estas extensiones de funcinalidad.
- Las LCDPs poseen su propio lenguaje, aunque este sea mayormente gráfico, sigue, como todo lenguaje de programación, una serie de normas sobre aquello que puedes o no puedes hacer, posee sus tipos de datos, sus métodos y su sintaxis de configuración propia para los componentes y sentencias condicionales.
- No son necesarios conocimientos técnicos previos, pero se extrae mayor potencial en menor tiempo si dispones de ellos, por ejemplo en OutSystems®, si quieres sacarle todo el partido a las herramientas que ofrece, será de gran ayuda un fuerte conocimiento en CSS, javaScript, JSON, SQL, C# y tener experiencia con sentencias y flujos de control condicionales, etc...
- Outsystem® es una gran herramienta para el desarrollo móvil. La aceleración en el desarrollo móvil es realmente el punto mas fuerte de Outsytms®, siendo capaz de generar

las aplicaciones tanto para Android como para IOS de forma automatizada con una sola acción y facilitando un desarrollo muy intuitivo con muchos componentes que simplifican el trabajo del desarrollador permitiendo también un despliegue e introducción de cambios a gran velocidad y directamente en los dispositivos, lo que reduce la necesidad de actualizaciones y permite el testeado inmediato en cualquier fase del desarrollo.

- Las LCDPs son el futuro de los desarrollos de aplicaciones, su imbatible velocidad las convierte en herramientas ágiles y capaces de adaptarse a cambios durante el desarrollo o mantenimiento de una aplicación, lo que las convierte en fuertes aliadas para la transformación digital.

5.2. Líneas futuras

En cuanto a las líneas futuras, me centrare en la aplicación desarrollada, puesto que ya conocemos las previsiones de crecimiento de las LCDPs.

Respecto a Easy Sports, hay multitud de funcionalidades contempladas en un primer momento que no han llegado para la versión presentada, algunas de ellas son:

- Gestión de otras estructuras de competición como torneos o “play-offs, como se ha visto anteriormente, la versión actual solo soporta el formato liga, y sería necesario realizar desarrollos con código externo similares para incluir los distintos modelos dentro de OutSystems.
- Ampliación de la información sobre la competición, mediante la inclusión de estadísticas detalladas que pudieran bajar incluso al desempeño de los jugadores dentro de un equipo.
- Inclusión de la figura del árbitro, dando la oportunidad a este de subir actas de partidos o editar los resultados liberando al mánager de tal responsabilidad.
- Añadir servicios de ubicación via Google Maps o similares para facilitar la localización o el establecimiento de rutas a los emplazamientos donde se realizan los eventos o enfrentamientos.
- Mejora de la asignación de fechas, incluyendo la asignación de horas. El sistema actual es bastante rudimentario y no contempla una asignación automática de horas. La idea sería consultar al creador tanto rangos de fechas como de horas y dar la opción de evitar o forzar que se empleen ciertos días y horas.
- Ampliar la funcionalidad móvil manteniendo un interfaz limpio y que proporcione una experiencia de usuario positiva.

Hay infinitud de mejoras y ampliaciones más que pueden aplicarse, y gracias a la agilidad de las LCDPs como OustSystems®, es posible continuar iterando sobre las versiones del desarrollo de forma rápida y eficaz, sin necesidad de actualizaciones que puedan dejar a los usuarios sin poder acceder al servicio actual.

Apéndice A

A.1. Algoritmo de Cruces

La autogeneración de los enfrentamientos en las competiciones, se sustenta sobre un algoritmo de emparejamiento que ha requerido de codificación adicional en C# para poder incorporar esta funcionalidad a OutSystems[®].

Aunque en apariencia pueda sonar trivial, hay una serie de requisitos que el sistema de emparejamiento debe cumplir que hacen que resulte una tarea mucho más elaborada:

- Los partidos deben repartirse en jornadas con el mismo número de encuentros.
- En cada jornada, ninguno de los jugadores o equipos participantes pueden participar en más de un partido.
- Los partidos deben de ser únicos en la jornada en la que se encuentran y en las demás jornadas (exceptuando el caso de que haya Ida y Vuelta).
- Si la competición es de ida y vuelta, los partidos de vuelta deben aparecer en el mismo orden que los de ida, pero invirtiendo el orden de los equipos (Lo que implica un cambio entre Local y Visitante).

Para satisfacer estas necesidades de forma automática, requerimos de un algoritmo en base sencillo sobre papel, pero relativamente complejo de implementar programáticamente.

El proceso para llegar a la versión final pasó por implementar una primera versión en Python, que resulta un lenguaje mucho más ligero e intuitivo que C# para realizar una prueba de concepto y comprobar que el algoritmo programado funcionaba. Las Figuras A.1 y A.2 recogen esta versión del algoritmo.

```

algorithmTFM.py
import math
import copy

impar = False
vuelta = True

def compute_journey(v1, v2):
    journey = []
    for i in range(len(v1)):
        partido = []
        partido.append(v1[i])
        partido.append(v2[i])
        journey.append(partido)
    return journey

size = int(input("Enter the number of teams: ")) #
v0 = list(range(1, size + 1)) # Estas instrucciones se sustituiran por una lista dada de Ids

if size % 2 != 0: # Caso en el que el número de equipos es impar
    v0.append(0)
    size += 1
    impar = True

print(v0, end="\n-----\n")

v1 = []
v2 = []

for i in v0:
    if i % 2 == 0:
        v2.append(i)
    else:
        v1.append(i)

print("v1: ", v1)
print("v2: ", v2)
journeys = []
total_matches = (math.factorial(size) / (2 * math.factorial(size - 2)))
numOfjourneys = int(total_matches / (size / 2))

# -----Codigo para permutaciones y desplazamientos (generacion de jornada por ciclo)-----

```

Figura A.1: Parte 1 Codigo Python

```

41
42 # -----Codigo para permutaciones y desplazamientos (generacion de jornada por ciclo)-----
43
44 for i in range(numOfjourneys):
45     journey = compute_journey(v1, v2)
46     # print("Jornada", (i+1),": ", journey, end="\n#####\n")
47     journeys.append(journey)
48     if i % 2 == 0:
49         aux = v1
50         v1 = v2[1:] + [v2[0]]
51         v2 = aux
52     elif i % 2 != 0:
53         aux = v1
54         v1 = [v1[0]] + v2[:-1]
55         v2 = [aux[-1]] + aux[1:-1] + [v2[-1]]
56
57 # -----Limpieza de los partidos de descanso -----
58
59 if impar:
60     for j in journeys:
61         for p in j:
62             for e in p:
63                 if e == 0:
64                     j.remove(p)
65                     break
66
67 # -----Generación de la vuelta -----
68
69 if vuelta:
70     journeysVuelta = copy.deepcopy(journeys)
71
72     for j in journeysVuelta:
73         for p in j:
74             p.reverse()
75     for e in journeysVuelta:
76         journeys.append(e)
77
78 for item in journeys:
79     print(item)
80

```

Figura A.2: Parte 2 Codigo Python

Una vez implementado en Python y probado, se procedió a implementar el mismo código en C#, esta traducción supuso un incremento considerable tanto en complejidad como en extensión del código. Se programó una aplicación de consola para probar el código al momento y una vez funcionando se pasó a adaptarlo para la encapsulación en OutSystems®.

Este proceso no resulta trivial si, como es el caso, vas a emplear estructuras o listas de tipos de datos compuestos para proporcionar y extraer los datos que emplea y genera el algoritmo. Esto se debe a que, para permitir la encapsulación de código y el uso de estas estructuras, Integration Studio genera automáticamente el código optimizado que envolverá a la función que deseamos añadir, lo que incluye una serie de clases y métodos para tratar con las estructuras y sus datos, por lo que adaptar el algoritmo resultó un proceso delicado.

Las siguientes figuras muestran el código del algoritmo en C#¹ y como finalmente esto se reduce a un componente más en Service Studio (Rojo Figura A.6).

```

1  using System;
2  using System.Collections;
3  using System.Collections.Generic;
4  using System.Data;
5  using System.Linq;
6  using OutSystems.RuntimePublic.Platform;
7  using OutSystems.RuntimePublic.Db;
8
9  namespace OutSystems.NssMatch_Maker {
10
11     public class C#Match_Maker: I#Match_Maker {
12
13         /// <summary>
14         /// recives
15         /// </summary>
16         /// <param name="ssLocation"></param>
17         /// <param name="ssTeamsIds"></param>
18         /// <param name="ssCompetition"></param>
19         /// <param name="ssVuelta"></param>
20
21         public void M#ssCreate_Matches(string ssLocation, R#IdentifierRecordList ssTeamsIds, out R#MatchRecordList ssCompetition, bool ssVuelta)
22         {
23             ssCompetition = new R#MatchRecordList();
24             List<int> v# = new List<int>();
25             foreach(R#IdentifierRecord i in ssTeamsIds)
26             {
27                 v#.Add((int)i.ssIdentifier.ssIdentifier);
28             }
29             bool hasVuelta = ssVuelta; //INPUT PARAMETER
30             int size = v#.Count; //INPUT PARAMETER
31             if (size % 2 != 0)
32             {
33                 v#.Add(0);
34                 size += 1;
35             }
36
37             ///int w = 3;
38             ///int n = (int)(45 / 3 - w);
39             ///Console.WriteLine("{0}", string.Join(", ", v#));
40
41             List<int> v1 = new List<int>();
42             List<int> v2 = new List<int>();
43
44             for (int i =0;i<v#.Count;i++)
45             {
46                 if (i % 2 == 0)
47                 {
48                     v2.Add(v#[i]);
49                 }
50                 else

```

Figura A.3: Parte 2 Codigo C#

¹No se muestran las estructuras ni otras clases autogeneradas por OutSystems® .

```

43
44     for (int i = 0; i < v0.Count; i++)
45     {
46         if (i % 2 == 0)
47         {
48             v2.Add(v0[i]);
49         }
50         else
51         {
52             v1.Add(v0[i]);
53         }
54     }
55
56
57     List<List<List<int>>> competition = new List<List<List<int>>>();
58     int totalMatches = (Factorial(size)) / (2 * Factorial(size - 2));
59     int numOfJourneys = (int)(totalMatches / (size / 2));
60     for (int i = 0; i < numOfJourneys; i++)
61     {
62         List<List<int>> journey = new List<List<int>>();
63         journey = ComputeJourney(v1, v2);
64         competition.Add(journey);
65         if (i % 2 == 0)
66         {
67             List<int> aux = new List<int>(v1);
68             v1 = v2.GetRange(1, v2.Count - 1);
69             v1.Add(v2[0]);
70             v2 = aux;
71         }
72     }
73     else if (i % 2 != 0)
74     {
75         List<int> aux1 = new List<int>(v1);
76         List<int> aux2 = new List<int>(v2);
77         v1.Clear();
78         v1.Add(aux1[0]);
79         v1 = v1.Concat(aux2.GetRange(0, aux2.Count - 1)).ToList();
80         v2.Clear();
81         v2.Add(aux1[aux1.Count - 1]);
82         v2 = v2.Concat(aux1.GetRange(1, aux1.Count - 2)).ToList();
83         v2.Add(aux2[aux2.Count - 1]);
84     }
85 }
86
87 if (hasVuelta)
88 {
89     List<List<List<int>>> competition2 = new List<List<List<int>>>();
90
91     for (int j = 0; j < competition.Count; j++)
92     {
93         List<List<int>> auxjourney = new List<List<int>>();
94         for (int p = 0; p < competition[j].Count; p++)
95         {
96             int value1 = competition[j][p][0];
97             int value2 = competition[j][p][1];
98             List<int> auxpartido = new List<int>();
99
100             auxpartido.Add(value2);
101             auxpartido.Add(value1);
102             auxjourney.Add(auxpartido);
103         }
104         competition2.Add(auxjourney);
105     }
106
107     competition.AddRange(competition2);
108 }
109
110
111
112     for (int j = 0; j < competition.Count; j++)
113     {
114         for (int p = 0; p < competition[j].Count; p++)
115         {
116
117             var match = new RCMatchRecord();
118             match.ssSTMatch.ssP1 = competition[j][p][0];
119             match.ssSTMatch.ssP2 = competition[j][p][1];
120             match.ssSTMatch.ssJourney = j+1;
121             match.ssSTMatch.ssLocation = ssLocation;
122             ssCompetition.Add(match);
123         }
124     }
125 }
126
127

```

Figura A.4: Parte 2 Código C#

```
127
128     public static int Factorial(int number)
129     {
130         int result = 1;
131         while (number != 1)
132         {
133             result *= number;
134             number -= 1;
135         }
136         if (result == 0)
137         {
138             result = 1;
139         }
140         return result;
141     }
142
143     public static List<List<int>> ComputeJourney(List<int> v1, List<int> v2)
144     {
145         List<List<int>> journey = new List<List<int>>();
146         int i = 0;
147         foreach (int t in v1)
148         {
149             if (v1[i] == 0 || v2[i] == 0)
150             {
151                 i++;
152             }
153             else
154             {
155                 List<int> m = new List<int> { v1[i], v2[i] };
156                 journey.Add(m);
157                 i++;
158             }
159         }
160         return journey;
161     }
162
163     // MssCreate_Matches
164
165     // CssMatch_Maker
166 } // OutSystems.NssMatch_Maker
167
168
169
170
171
```

Figura A.5: Parte 3 Código C#

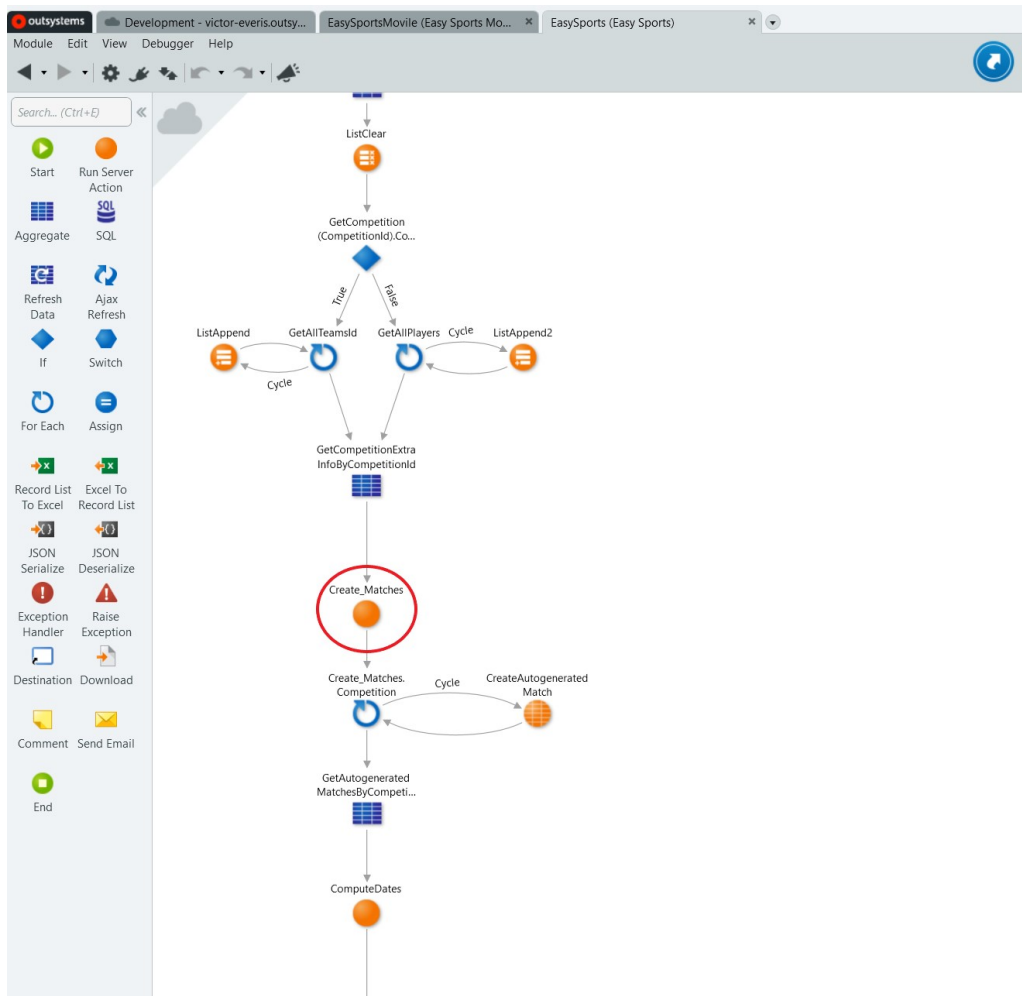


Figura A.6: Implementación en Service Studio

Bibliografía

- [1] «Low-Code Development Platform». En Wikipedia, 18 de julio de 2019. https://en.wikipedia.org/w/index.php?title=Low-code_development_platform&oldid=906801739.
- [2] «Forrester Research». En Wikipedia, la enciclopedia libre, 7 de febrero de 2017. https://es.wikipedia.org/w/index.php?title=Forrester_Research&oldid=96747759.
- [3] «Qué es una plataforma Low-Code y cómo puede ayudar a tu empresa?». Accedido 15 de agosto de 2019. <https://www.cibernos.com/blog/desarrollo-de-software/una-plataforma-low-code-puede-ayudar-empresa>.
- [4] MarketsandMarkets.«Low-Code Development Platform Market Worth 27.23 Billion USD by 2022». Accedido 15 de agosto de 2019. <https://www.prnewswire.com/news-releases/low-code-development-platform-market-worth-2723-billion-usd-by-2022-668601263.html>.
- [5] pi.com, Redacción Byte TI byteti@mkm-. «Las grandes empresas apuestan por el low-code». Revista Byte TI, 2 de abril de 2019. <https://www.revistabyte.es/actualidad-byte/las-grandes-empresas-low-code/>.
- [6] «Low-Code Enterprise Application Platform». Appian (blog). <https://www.appian.com/platform/>.
- [7] «Salesforce.com». En Wikipedia, la enciclopedia libre, 18 de julio de 2019. <https://es.wikipedia.org/w/index.php?title=Salesforce.com&oldid=117504803>.
- [8] «Sales Cloud: herramientas de automatización de la fuerza de ventas». Salesforce.com. Accedido 16 de Agosto de 2019. <https://www.salesforce.com/es/products/sales-cloud/overview/>.
- [9] «Crear aplicaciones de lienzo o controladas por modelos | Microsoft PowerApps». <https://powerapps.microsoft.com/es-es/build-powerapps/>.
- [10] «Mendix - Go Make It». Mendix, 15 de mayo de 2012. <https://www.mendix.com/>.
- [11] «Magic Quadrant for Enterprise Low-Code Application Platforms». Gartner. Accedido 15 de agosto de 2019. <https://www.gartner.com/en/documents/3956079/magic-quadrant-for-enterprise-low-code-application-platf>.

- [12] «DevOps». En Wikipedia, la enciclopedia libre, 31 de julio de 2019. <https://es.wikipedia.org/w/index.php?title=DevOps&oldid=117879130>.
- [13] «OutSystems Training: Learn to Develop Mobile and Web Apps». Accedido 4 de septiembre de 2019. <https://www.outsystems.com/learn/>.
- [14] «Modelos de servicio de cloud IaaS PaaS SaaS», 20 de agosto de 2019. <https://www.ibm.com/es-es/cloud/learn/iaas-paas-saas>.