



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

– **TELECOM** ESCUELA
TÉCNICA **VLC** SUPERIOR
DE INGENIERÍA DE
TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería de
Telecomunicación

Modernización del sistema de control de vacaciones de
una empresa tecnológica mediante el desarrollo de una
aplicación utilizando Power Platform

Trabajo Fin de Grado

Grado en Tecnología Digital y Multimedia

AUTOR/A: Yebra Castillo, Mario

Tutor/a: Guerola Navarro, Vicente

CURSO ACADÉMICO: 2023/2024

Resumen:

El propósito de este Trabajo Final de Grado se basa en la modernización y mejora del sistema de gestión de vacaciones dentro de una gran empresa del sector de la consultoría tecnológica.

El principal objetivo será el diseño e implementación de una nueva solución técnica que pueda cubrir las necesidades detectadas. Para ello se partirá de un análisis de requisitos que permita detectar las vulnerabilidades del sistema actual, derivando en el posterior diseño de la arquitectura de la solución, integraciones con sistemas externos e implementación.

Para llevar a cabo dicha modernización, se desarrollará desde cero una solución técnica completa mediante la tecnología Low Code de Microsoft Power Platform, empleando principalmente Power Apps, para la creación y diseño de la aplicación de gestión de vacaciones, y Power Automate, para la automatización de flujos de trabajo, notificaciones y procesos de aprobación. La elección de esta tecnología situada dentro del ámbito de desarrollo de low code permitirá un elevado grado de personalización con un time-to-market reducido, adaptándose a las necesidades cambiantes del mercado.

Dada la criticidad del dato al tratarse de un sistema de Recursos Humanos, se prestará especial atención a la seguridad e integridad de la información, empleando Microsoft Dataverse para realizar el diseño de la base de datos que nutra la solución.

Este sistema permitirá así la contabilización de los días de vacaciones, el envío y aprobación de peticiones, un calendario laboral personal de cada trabajador y la auditoría de todos los cambios que se realicen, entre otras funcionalidades. Obteniendo así una gran evolución y mejora en el formato de gestión.

Palabras clave:

Gestor de vacaciones; Power Platform; Desarrollo de aplicación; Modernización; Empresa tecnológica; Low code; Calendario Laboral; Automatización.

Resum:

El propòsit d'aquest Treball Final de Grau es basa en la modernització i millora del sistema de gestió de vacances dins d'una gran empresa del sector de la consultoria tecnològica.

El principal objectiu serà el disseny i implementació d'una nova solució tècnica que puga cobrir les necessitats detectades. Per a això es partirà d'una anàlisi de requisits que permetia detectar les vulnerabilitats del sistema actual, derivant en el posterior disseny de l'arquitectura de la solució, integracions amb sistemes externs i implementació.

Per a dur a terme aquesta modernització, es desenvoluparà des de zero una solució tècnica completa mitjançant la tecnologia Low Code de Microsoft Power Platform, emprant principalment Power Apps, per a la creació i disseny de l'aplicació de gestió de vacances, i Power Automate, per a l'automatització de fluxos de treball, notificacions i processos d'aprovació. L'elecció d'aquesta tecnologia situada dins de l'àmbit de desenvolupament de low code permetrà un elevat grau de personalització amb un time-to-market reduït, adaptant-se a les necessitats canviants del mercat.

Donada la criticitat de la dada en tractar-se d'un sistema de Recursos Humans, es prestarà especial atenció a la seguretat i integritat de la informació, emprant Microsoft Dataverse per a realitzar el disseny de la base de dades que nodrisca la solució.

Aquest sistema permetrà així la comptabilització dels dies de vacances, l'enviament i aprovació de peticions, un calendari laboral personal de cada treballador i l'auditoria de tots els canvis que es realitzen, entre altres funcionalitats. Obtenint així una gran evolució i millora en el format de gestió.

Paraules clau:

Gestor de vacances; Power Platform; Desenvolupament d'aplicació; Modernització; Empresa tecnològica; Low code; Calendari Laboral; Automatització.

Summary:

The purpose of this Final Degree Project is based on the modernization and improvement of the vacation management system within a large company in the technology consulting sector.

The main objective will be the design and implementation of a new technical solution that can cover the identified needs. To achieve this, we will start with a requirements analysis to detect the vulnerabilities of the current system, leading to the subsequent design of the solution architecture, integrations with external systems, and implementation.

To carry out this modernization of the vacation management system, a complete technical solution will be developed from scratch using Microsoft Power Platform Low Code technology, mainly using Power Apps, for the creation and design of the vacation management application, and Power Automate, for the automation of workflows, notifications and approval processes. The choice of this technology, located within the low-code development environment, will allow a high degree of customization with a reduced time-to-market, adapting to the changing market needs.

Given the criticality of the data as it is a Human Resources system, special attention will be paid to the security and integrity of the information, using Microsoft Dataverse to design the database that feeds the solution.

This system will thus allow the accounting of vacation days, the submission and approval of requests, a personal work calendar for each employee, and auditing of all changes made, among other functionalities. This will result in a significant evolution and improvement in the management format.

Keywords:

Vacation manager; Power Platform; Application development; Modernization; Technology company; Low code; Work calendar; Automation.

RESUMEN EJECUTIVO

La memoria del TFG del Grado en Tecnología Multimedia debe desarrollar en el texto los siguientes conceptos, debidamente justificados y discutidos, centrados en el ámbito de las tecnologías digitales y multimedia

CONCEPT (ABET)	CONCEPTO (traducción)	¿Cumple? (S/N)	¿Dónde? (páginas)
1. IDENTIFY:	1. IDENTIFICAR:		
1.1. Problem statement and opportunity	1.1. Planteamiento del problema y oportunidad	S	10
1.2. Constraints (standards, codes, needs, requirements & specifications)	1.2. Toma en consideración de los condicionantes (normas técnicas y regulación, necesidades, requisitos y especificaciones)	S	13-20
1.3. Setting of goals	1.3. Establecimiento de objetivos	S	11
2. FORMULATE:	2. FORMULAR:		
2.1. Creative solution generation (analysis)	2.1. Generación de soluciones creativas (análisis)	S	20-23, 31, 35-36, 42, 49-50, 53-54, 56-57, 61-62, 65-66, 71-72
2.2. Evaluation of multiple solutions and decision-making (synthesis)	2.2. Evaluación de múltiples soluciones y toma de decisiones(síntesis)	S	23-24
3. SOLVE:	3. RESOLVER:		
3.1. Fulfilment of goals	3.1. Evaluación del cumplimiento de objetivos	S	81-82
3.2. Overall impact and significance (contributions and practical recommendations)	3.2. Evaluación del impacto global y alcance (contribuciones y recomendaciones prácticas)	S	83-84

Agradecimientos

Quiero agradecer a mis docentes, en especial a mi tutor, Vicente, por su dedicación, implicación y empatía con la que me ha guiado en esta etapa final de la universidad.

A mi tutor y compañeros de la empresa, quienes me han ayudado y brindado la oportunidad de aplicar mis conocimientos en un entorno real.

También a mi familia y amigos, por sus ánimos, apoyo y cariño que me han dado siempre y me seguirán dando haga lo que haga.

Y sobre todo a mi pareja, Marta, por su apoyo incondicional día tras día y por acompañarme en este viaje tan especial.

Por último, quiero dedicar este logro a mi madre, quien siempre ha sido y seguirá siendo mi mayor inspiración. Estoy seguro de lo orgullosa que estarías de mí.

Tabla de contenido

1	Introducción -----	10
1.1	Justificación y Motivación personal -----	10
1.2	Objetivos-----	11
1.3	Relación con el Grado en Tecnología Digital y Multimedia:-----	12
2	Marco conceptual-----	13
2.1	Herramientas tecnológicas -----	13
2.1.1	Low code -----	13
2.1.2	Power Platform-----	14
2.2	Acerca de la empresa -----	18
2.3	Vacaciones en una empresa-----	18
3	Análisis de requisitos-----	20
3.1	Explicación del sistema actual -----	20
3.2	Identificación de vulnerabilidades del sistema actual y necesidades de mejora 23	
3.3	Posibles soluciones -----	23
4	Diseño de la arquitectura-----	24
4.1	Definición de los componentes técnicos -----	24
4.2	Definición del modelo de datos -----	25
4.3	Definición de roles de seguridad-----	27
5	Desarrollo de la solución -----	28
5.1	Desarrollo de la base de datos -----	28
5.2	Desarrollo de la aplicación-----	30
5.2.1	Pantalla 0 - Bienvenida -----	31
5.2.2	Pantalla 1 - Inicio-----	35
5.2.3	Pantalla 2 – Solicitar días-----	42
5.2.4	Pantalla 3 – Mis solicitudes -----	49
5.2.5	Pantalla 4 – Consultar calendario -----	53
5.2.6	Pantalla 5 – Modo mánager / administrador-----	56
5.2.7	Pantalla 6 – Solicitudes-----	61
5.2.8	Pantalla 7 – Gestión de las solicitudes -----	65
5.2.9	Pantalla 8 – Modificar festivos-----	71

5.3	Desarrollo de flujos de trabajo automatizados-----	78
5.3.1	Flujo - Actualizar días a final de año-----	79
6	Resultados de la solución -----	81
6.1	Resultados finales -----	81
6.2	Funcionamiento tras uso -----	81
6.3	Posibles mejoras-----	82
7	Conclusiones -----	83
8	Bibliografía -----	84

Tabla de figuras

Figura 1: Elementos que componen Power Platform. Fuente: Microsoft -----	15
Figura 2: Captura Excel, Hoja "Cargabilidad 2024". Fuente: Elaboración propia -----	21
Figura 3: Captura Excel, Leyenda. Fuente: Elaboración propia -----	21
Figura 4: Captura Excel, Hoja "Vacaciones". Fuente: Elaboración propia -----	22
Figura 5: Diagrama de la arquitectura de la solución. Fuente: Elaboración propia -----	24
Figura 6: Diagrama de la base de datos. Fuente: Elaboración propia -----	26
Figura 7: Captura creación de la solución. Fuente: Elaboración propia -----	28
Figura 8: Captura creación de una nueva tabla. Fuente: Elaboración propia -----	29
Figura 9: Captura creación de una nueva columna. Fuente: Elaboración propia -----	29
Figura 10: Captura resultado tabla Empleados. Fuente: Elaboración propia -----	29
Figura 11: Captura resultado de todas las tablas creadas. Fuente: Elaboración propia --	30
Figura 12: Diagrama de pantallas de la aplicación. Fuente: Elaboración propia -----	30
Figura 13: Boceto previo pantalla 0. Fuente: Elaboración propia -----	31
Figura 14: Editor de Power Apps nada más iniciar. Fuente: Elaboración propia -----	32
Figura 15: Resultado 1 de la pantalla 0. Fuente: Elaboración propia -----	34
Figura 16: Resultado 2 de la pantalla 0. Fuente: Elaboración propia -----	35
Figura 17: Boceto previo de la pantalla 1. Fuente: Elaboración propia -----	36
Figura 18: Vista de árbol del Container1. Fuente: Elaboración propia -----	40
Figura 19: Resultado 1 de la pantalla 1. Fuente: Elaboración propia -----	40
Figura 20: Resultado 2 de la pantalla 1. Fuente: Elaboración propia -----	41
Figura 21: Resultado 3 de la pantalla 1. Fuente: Elaboración propia -----	41
Figura 22: Boceto previo de la pantalla 2. Fuente: Elaboración propia -----	42
Figura 23: Vista de árbol de la galería "MonthDayGallery2". Fuente: Elaboración propia -	44
Figura 24: Resultado 1 de la pantalla 2. Fuente: Elaboración propia -----	49
Figura 25: Resultado 2 de la pantalla 2. Fuente: Elaboración propia -----	49
Figura 26: Boceto previo de la pantalla 3. Fuente: Elaboración propia -----	50
Figura 27: Resultado 1 de la pantalla 3. Fuente: Elaboración propia -----	52
Figura 28: Resultado 2 de la pantalla 3. Fuente: Elaboración propia -----	53
Figura 29: Boceto previo de la pantalla 4. Fuente: Elaboración propia -----	54
Figura 30: Resultado de la pantalla 4. Fuente: Elaboración propia -----	56
Figura 31: Boceto previo de la pantalla 5. Fuente: Elaboración propia -----	57
Figura 32: Resultado 1 de la pantalla 5. Fuente: Elaboración propia -----	60
Figura 33: Resultado 2 de la pantalla 5. Fuente: Elaboración propia -----	60
Figura 34: Resultado 3 de la pantalla 5. Fuente: Elaboración propia -----	61
Figura 35: Boceto previo de la pantalla 6. Fuente: Elaboración propia -----	62
Figura 36: Resultado 1 de la pantalla 6. Fuente: Elaboración propia -----	64
Figura 37: Resultado 2 de la pantalla 6. Fuente: Elaboración propia -----	64
Figura 38: Resultado 3 de la pantalla 6. Fuente: Elaboración propia -----	65
Figura 39: Boceto previo de la pantalla 7. Fuente: Elaboración propia -----	66
Figura 40: Resultado 1 de la pantalla 7. Fuente: Elaboración propia -----	70
Figura 41: Resultado 2 de la pantalla 7. Fuente: Elaboración propia -----	71
Figura 42: Boceto previo de la pantalla 8. Fuente: Elaboración propia -----	72
Figura 43: Resultado 1 de la pantalla 8. Fuente: Elaboración propia -----	76
Figura 44: Resultado 2 de la pantalla 8. Fuente: Elaboración propia -----	77
Figura 45: Resultado 3 de la pantalla 8. Fuente: Elaboración propia -----	77

Figura 46: Resultado 4 de la pantalla 8. Fuente: Elaboración propia-----	78
Figura 47: Captura de todos los pasos que componen el flujo. Fuente: Elaboración propia -----	79
Figura 48: Captura del primer paso del flujo en detalle. Fuente: Elaboración propia -----	79
Figura 49: Captura del segundo paso del flujo en detalle. Fuente: Elaboración propia----	79
Figura 50: Captura del tercer paso del flujo en detalle. Fuente: Elaboración propia -----	80
Figura 51: Captura del último paso del flujo en detalle. Fuente: Elaboración propia -----	80
Figura 52: Diagrama final de pantallas y navegación. Fuente: Elaboración propia -----	81

1 Introducción

1.1 Justificación y Motivación personal

El presente proyecto final de grado surge como resultado de una experiencia enriquecedora en la que actualmente me encuentro inmerso: un período de prácticas en una destacada consultoría tecnológica. Durante este periodo estoy teniendo la oportunidad de aprender y trabajar con una variedad de herramientas, entre las cuales destaca Power Platform. Mi función principal en la empresa consiste en el desarrollo de aplicaciones para clientes externos utilizando esta plataforma, cuyo potencial ha despertado en mí tal interés que Power Platform constituirá la herramienta principal en el desarrollo de la aplicación que será el foco central de este trabajo fin de grado (TFG).

La idea concreta de la aplicación a realizar nace de la identificación de necesidades y problemas en la actual forma de gestión de las vacaciones en la empresa en la que me encuentro realizando las prácticas. Considerando que sería una gran opción aprovechar los conocimientos aprendidos en el desarrollo de aplicaciones mediante Power Platform durante este periodo para diseñar y desarrollar un nuevo sistema de gestión que facilite y automatice todas las funcionalidades a los trabajadores, acabando así con las vulnerabilidades y problemas actuales.

En este sentido, la realización de este TFG no solo representa un desafío académico, sino también una oportunidad para aplicar los conocimientos adquiridos tanto durante mi formación universitaria, como en este propio periodo laboral, enfocándolos hacia un contexto práctico y relevante. Asimismo, este es un proyecto muy motivador para mí ya que el proceso de desarrollo de este contribuirá en gran medida a mi crecimiento profesional y personal. En primer lugar, permitiéndome explorar al máximo las posibilidades que ofrece Power Platform y otras herramientas tecnológicas complementarias. Por otro lado, desarrollando mi capacidad para la identificación y resolución de problemas reales dentro del entorno empresarial. Desarrollando también mi capacidad creativa a la hora de diseñar soluciones eficientes y adaptadas a cada problema. Y, además, mejorando mis habilidades de pensamiento crítico y de análisis gracias al proceso de investigación que es necesario realizar.

Todo esto, sentará las bases para mi desarrollo profesional continuo, equipándome con las habilidades y la confianza necesarias para enfrentar con éxito los desafíos futuros en mi carrera.

Y por supuesto, se espera que los resultados obtenidos aporten valor a la empresa. Con la intención principal de que dicha aplicación desarrollada sea integrada y utilizada de manera interna por los trabajadores como nuevo método de gestión de las vacaciones, lo cual también es una gran motivación extra.

Siguiendo así con los objetivos generales de: compromiso por la innovación y resolución eficaz de problemáticas empresariales, que caracterizan a la consultoría tecnológica en la que me encuentro.

1.2 Objetivos

Además de los objetivos de desarrollo personal y profesional mencionados anteriormente, el objetivo principal de este Proyecto Final de Grado es el diseño y desarrollo de una Aplicación que ayude a mi empresa a modernizar y mejorar el sistema actual de gestión de días de vacaciones. Actualmente esta gestión se realiza mediante una pesada hoja de cálculo en Excel donde aparecen los datos y el calendario laboral de todos los trabajadores de la empresa, y donde se van filtrando y modificando manualmente estos datos según se desee.

Por tanto, los objetivos que se pretenden y se deben alcanzar con el desarrollo de este nuevo sistema son:

1. **Acabar con las vulnerabilidades del sistema actual:** Se pretende cubrir todas las necesidades detectadas y acabar con los problemas y puntos débiles que posee el actual método de gestión.
2. **Automatización y centralización de la gestión:** La aplicación desarrollada permitirá la centralización de la información y de todas las acciones realizables en un único sistema, tales como el envío de peticiones de días de vacaciones, la aprobación o rechazo de esas mismas peticiones o la consulta de todo tipo de información.
3. **Optimización del tiempo y recursos:** Con este nuevo sistema se persigue optimizar el tiempo y los recursos dedicados a la gestión de días de vacaciones.
4. **Reducción de errores:** Al ser un sistema más automatizado se conseguiría minimizar la cantidad de errores manuales que se puedan producir.
5. **Incremento de la Satisfacción del Personal:** Al agilizar y simplificar el proceso de consulta y gestión de las vacaciones, se espera aumentar la satisfacción del personal, brindando una experiencia más cómoda y rápida en la solicitud y seguimiento de sus días de descanso.
6. **Seguridad de la información:** Se pretende implementar unas robustas medidas de seguridad, protegiendo al máximo la integridad y confidencialidad de los datos de los trabajadores.
7. **Usabilidad y experiencia del usuario:** Se persigue, además, priorizar la experiencia del usuario en el diseño de la aplicación, asegurando que esta sea intuitiva y fácil de usar para todo el personal.
8. **Cumplimiento Normativo y Legal:** Por último, se debe asegurar que la aplicación cumpla con las normativas y regulaciones legales pertinentes, tanto en materia de protección de datos y privacidad, como en materia de legislación de vacaciones laborales.

1.3 Relación con el Grado en Tecnología Digital y Multimedia:

En el grado que actualmente estoy cursando, Tecnología Digital y Multimedia, se abordan todas las etapas de vida de un contenido digital, desde la creación (Diseño gráfico, narrativa, interacción...) hasta la explotación (gestión de proyectos, metodologías ágiles...) pasando, como no, por la distribución (redes de acceso, seguridad, transporte...). Combinando así la formación en una gran cantidad de tecnologías con el fomento de habilidades creativas y de organización y planificación.

Por tanto, se pueden encontrar una importante cantidad de similitudes y conexiones entre el Grado y este TFG. Proporcionando así, el grado, una base sólida en diversas áreas y una variedad de aprendizajes que son completamente necesarios para poder llevar a cabo la realización de este TFG.

En primer lugar, y probablemente la conexión más importante entre el Grado y este TFG, la programación. Durante los cuatro años del Grado se han ido abordando y aprendiendo diferentes lenguajes de programación, los cuales estaban presentes, enfocados de una manera u otra, en prácticamente todas las asignaturas de la carrera. Y a pesar de que Power Platform sea una tecnología Low code, la programación sigue siendo en ella el eje principal a la hora de desarrollar una aplicación como la presente. En el caso de Power Platform el lenguaje utilizado es PowerFX, el cuál se puede asemejar en cierta manera a JavaScript, pero realmente, para aprenderlo y utilizarlo ha sido de fundamental ayuda tener una base sólida de otros muchos lenguajes de programación que se han visto en el grado, tales como Python, Java, C, HTML o el propio JavaScript. Por eso, asignaturas como *Programación*, *Tecnologías Web* o *Aplicaciones y Usabilidad*, entre otras, han sido claves a la hora de brindar el aprendizaje necesario en programación para poder desarrollar este proyecto.

Otra conexión importante es la del diseño digital, ya que la aplicación a desarrollar requerirá un diseño de interfaz de usuario intuitivo y atractivo. Por tanto, aquí es clave el aprendizaje y sabiduría desarrollado en algunas asignaturas del Grado, tales como *Diseño Gráfico*, para el apartado visual de la aplicación, o *Tecnologías Web* para la parte de diseño funcional.

Por otro lado, también encontramos relación si hablamos de la seguridad y protección de los datos, el cual es un tema transversal en el Grado y que se ha ido enseñando y concienciando a lo largo de diferentes asignaturas. Lo que ha servido para aprender de la importancia de prestar especial atención a la seguridad e integración de la información. Este tema se ha tratado especialmente en asignaturas como *Seguridad y Gestión de Derechos Digitales* o incluso en algunos talleres y seminarios como el de SQL, a la hora de la gestión de bases de datos.

Por último, existe una relación general en el desarrollo de habilidades de organización y planificación, las cuáles son fundamentales a la hora de realizar cualquier tipo de proyecto, y por supuesto, en este Trabajo Final de Grado también. Las asignaturas del grado que han brindado este aprendizaje son *Dirección y Gestión de Proyectos*, *Organización y Transformación Digital* e incluso, *Modelos de Negocio*. Ayudando a la gestión y

planificación de tareas y procesos que se han de realizar para el correcto desarrollo y realización de este mismo Trabajo Final de Grado.

2 Marco conceptual

2.1 Herramientas tecnológicas

2.1.1 Low code

Como ya se ha nombrado antes, Power Platform, el conjunto de herramientas de Microsoft que se va a emplear para desarrollar el sistema de gestión de las vacaciones, utiliza la tecnología low code.

Pero, **¿Qué significa low code?**

El término *low code* podría traducirse de manera general como código bajo o *poco código*, con la intención de evitar así, en la medida de lo posible, la programación manual convencional.

Este concepto de *low code* fue introducido por la firma de análisis de mercado Forrester Research alrededor del año 2014 en un reporte que exploraba nuevas plataformas de desarrollo de aplicaciones centradas en el usuario. Lo cual habla claramente de la novedad e innovación que caracteriza a este tipo de desarrollo.

Las plataformas que utilizan este enfoque de low code se basan principalmente en interfaces gráficas intuitivas y en la utilización de elementos visuales predefinidos. Facilitando de manera importante el proceso de desarrollo de software, minimizando la codificación y utilizando un enfoque modular y visual que permite a los usuarios diseñar la interfaz de usuario, definir la lógica empresarial y conectar con sistemas externos sin necesidad de tener un elevado conocimiento en programación.

No obstante, estas plataformas low code no prescinden totalmente de la programación manual, como sí que lo hacen las llamadas plataformas no-code, las cuales permiten ahorrar aún más tiempo y recursos, pero poseen la desventaja lógica de que no son tan flexibles y potentes como las low code. Por esto mismo, en plataformas low code como Power Platform, la programación sigue siendo un elemento crucial a la hora de desarrollar complejos procesos o aplicaciones.

¿Cuáles son las ventajas del low code?

1. **Velocidad de desarrollo:** Al simplificar el proceso de desarrollo y requerir de menos programación, se reduce considerablemente el time-to-market, es decir, las aplicaciones completas se desarrollan más rápidamente permitiendo a las empresas crearlas y lanzarlas al mercado en un tiempo récord. Lo que se puede llegar a traducir en una considerable ventaja competitiva.

2. **Eficiencia:** No solo se reduce el tiempo de desarrollo, si no que se aumenta la eficiencia y satisfacción de los propios programadores, que pueden centrar la atención en el desarrollo principal y las funciones importantes en lugar de malgastar una gran cantidad de tiempo buscando fallos en el código.
3. **Reducción de costos:** Al acelerar el proceso de desarrollo y aumentar la eficiencia de los recursos internos, se puede obtener una reducción de los costos asociados con el desarrollo de software. Además de que la inversión inicial es menor debido a que los trabajadores no necesitan costosos cursos de aprendizaje y capacitación.
4. **Accesibilidad:** Las tecnologías low code permiten que el desarrollo de aplicaciones sea accesible para un espectro más amplio de personas, eliminando la barrera de entrada de la codificación y apostando por la sencillez en el desarrollo de aplicaciones básicas.
5. **Colaboración:** La accesibilidad de las plataformas low code para el personal de cualquier ámbito de una empresa facilita la colaboración entre equipos multidisciplinarios y la participación de todos ellos en el diseño, desarrollo y creación de una aplicación que cubra las necesidades de todos y cada uno de los departamentos.

En definitiva, las tecnologías low code están revolucionando el método de desarrollo de las aplicaciones, ofreciendo una alternativa más ágil, rápida, accesible y rentable, y que además mantiene todas las opciones de personalización y flexibilidad del desarrollo tradición de software.

2.1.2 Power Platform

Como ya se ha mencionado anteriormente, la plataforma a través de la cual se va a desarrollar todo el sistema de gestión de las vacaciones es Power Platform.

Pero, ¿Qué es Microsoft Power Platform?

Power Platform es un conjunto de herramientas de Microsoft integradas en Office 365, que están preparadas para crear soluciones personalizadas, automatizar procesos, analizar datos y actuar sobre ellos. Todas estas herramientas de productividad incluidas en Power Platform son denominadas “low code”, tal y como ha sido explicado anteriormente, y se integran tanto entre sí como con otras soluciones de Microsoft.

Esta plataforma SaaS de aplicaciones empresariales ha ido adquiriendo popularidad en los últimos años debido a su capacidad para ayudar a las empresas a optimizar sus procesos, sistemas y flujos de trabajo, ofreciendo una lógica empresarial personalizable, automatización con poco código y aplicaciones funcionales basadas en datos.

El objetivo de Microsoft con esta plataforma se centra en establecer un marco conocido como el "Ciclo de triple A". Un sistema de ciclo cerrado que capacita a los usuarios para extraer conocimientos de los datos (Analizar), los cuales se emplean para impulsar la

creación de aplicaciones (Actuar) y la automatización de procesos empresariales (Automatizar), fomentando así la inteligencia en los negocios.

Esta plataforma se compone de 5 aplicaciones principales:

- **Power BI:** Herramienta por excelencia de análisis empresarial que permite realizar informes dinámicos, visualizar datos y compartirlos de manera intuitiva y rápida.
- **Power Apps:** Herramienta de desarrollo de aplicaciones que permite crear y publicar aplicaciones empresariales personalizadas y conectadas con la fuente de datos y las herramientas del día a día que se deseen.
- **Power Automate:** Potente motor de flujos que permite crear flujos de trabajo automatizados entre sus aplicaciones, simplificado y automatizando así procesos y tareas del día a día.
- **Power Virtual Agents:** Herramienta que permite crear chatbots inteligentes de manera ágil e intuitiva, capaces de responder consultas de clientes y automatizar interacciones con los usuarios.
- **Power Pages:** Herramienta que permite crear, alojar y administrar sitios web corporativos de manera rápida y optimizada con alto rendimiento.

Y de otros elementos claves para el desarrollo y la conectividad de las soluciones como:

- **Data connectors:** Se tratan de un conjunto de conectores que facilitan la comunicación con una amplia gama de fuentes de datos, permitiendo acceder a cientos de ellas.
- **AI Builder:** Permite utilizar e incluso crear modelos de inteligencia artificial para optimizar procesos y obtener información de los datos en Power Apps y en Power Automate.
- **Dataverse:** Plataforma que permite almacenar y gestionar los datos utilizados por las aplicaciones empresariales de manera segura y eficiente en la nube. Actuará como base de datos del sistema que se va a desarrollar en este proyecto.



The low code platform that spans Microsoft 365, Azure, Dynamics 365, and standalone apps.

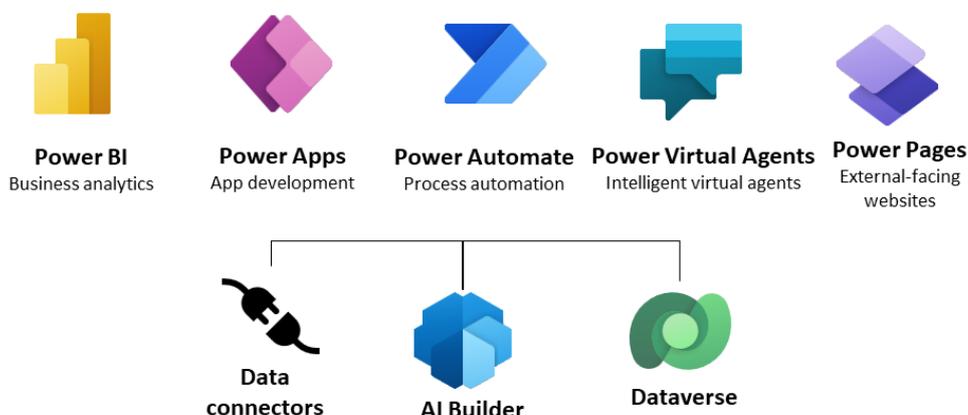


Figura 1: Elementos que componen Power Platform. Fuente: Microsoft

De las 5 aplicaciones mencionadas que forman Microsoft Power Platform, las dos que serán utilizadas para el desarrollo del nuevo sistema de gestión de vacaciones serán principalmente Power Apps y Power Automate:

2.1.2.1 Power Apps

Tal y como se ha mencionado, Power Apps es la plataforma de desarrollo de aplicaciones móviles, web o de escritorio que ofrece Microsoft. Esta herramienta permite a los usuarios crear rápidamente aplicaciones personalizadas para su negocio e integrarlas con los datos y sistemas existentes. Todo ello se puede realizar de manera ágil, sencilla y sin requerir de profundos conocimientos de programación, debido a utilizar el tipo de desarrollo low code.

Sin embargo, Power Apps ofrece también una plataforma extensible que capacita a los desarrolladores expertos para interactuar con datos y metadatos mediante programación más compleja, implementar lógica empresarial, desarrollar conectores personalizados y establecer integraciones con fuentes de datos externas.

En Power Apps encontramos dos posibles tipos de Aplicaciones a utilizar:

- Canvas o tipo Lienzo: Como su nombre indica, se parte de un lienzo en blanco y por tanto poseemos el control absoluto del diseño, permitiendo así crear una interfaz totalmente personalizada. Además, ofrece conectividad con más de 200 orígenes de datos diferentes. Este tipo de Power Apps será el utilizado para el desarrollo de la aplicación.
- Model Driven: En este tipo de aplicaciones partimos del modelo de datos que decidamos y de una interfaz prediseñada, por lo que contamos con menos capacidad de personalización. Sin embargo, gracias a esto contamos con un diseño completamente adaptativo, que ofrece respuesta en todos los dispositivos.

En cualquiera de los dos casos, se trata de una herramienta intuitiva que ofrece un entorno de diseño visual que permite a los usuarios diseñar la aplicación combinando un sistema de *Drag and Drop*, es decir, arrastrar y soltar elementos de la interfaz como botones, formularios y galerías, con el código de programación necesario en cada elemento para obtener el diseño y la funcionalidad deseada en cada situación.

2.1.2.2 Power Automate

Microsoft Power Platform, antes conocido como Microsoft Flow, es un sistema basado en la nube que permite a los usuarios crear flujos de trabajo automatizados. Simplificando y optimizando así los procesos comerciales.

El objetivo principal de esta plataforma es brindar la oportunidad de aumentar la productividad y la eficiencia empresarial mediante la automatización de cualquier tarea

repetitiva, de bajo esfuerzo o que consuma mucho tiempo. Facilitando también el intercambio de datos entre diferentes servicios y aplicaciones.

Para utilizar esta herramienta deberemos ir añadiendo elementos de forma encadenada mediante la intuitiva interfaz y combinándolo con cierto código de programación, para así dar forma a los flujos de trabajo que queremos crear. Estos elementos que hay encadenar pueden ser de tres tipos:

- **Desencadenadores o triggers:** Como su nombre indica, son los elementos que inician el flujo de trabajo. Cada flujo debe de tener al menos uno o más desencadenadores. (Puede ser cuando te llegue un correo, cuando se guarde un nuevo registro en la base de datos, etc.)
- **Acciones:** las acciones que se han de realizar una vez ya se haya activado el flujo. Aquí podemos aplicar también elementos de control, como condiciones o bucles, entre otros. (Puede ser guardar un archivo, enviar un correo, modificar algún dato en la base de datos...)
- **Conectores:** Estos son los elementos necesarios para conectar 2 o más aplicaciones. Sin olvidar que la herramienta nos solicitará las credenciales de las dos aplicaciones

Por otro lado, según la forma de activar y llamar al flujo, encontramos tres tipos de flujos:

- **Flujos automatizados:** El flujo se desencadena automáticamente por la sucesión de un evento.
- **Flujos instantáneos:** El elemento desencadenador en este caso es el click de un botón, es decir, el flujo se desencadena manualmente desde el dispositivo que deseas.
- **Flujos Programados:** En ellos se programa previamente cuándo y con qué frecuencia queremos que se desencadene el flujo. (Por ejemplo, todos los días a las 20:30, o cada 5 minutos, o solo un día concreto a una hora concreta)

En definitiva, Power Automate ofrece una gran variedad de tipos de flujo y opciones a implementar, con la intención de que las organizaciones puedan automatizar una amplia gama de procesos empresariales, ya sea tareas simples pero repetitivas o procesos complejos que involucran multitud de aplicaciones y sistemas.

2.1.2.3 Power FX

Tal y como hemos dicho antes, en Power Apps y Power Automate, a pesar de ser tecnologías low code, la programación sigue siendo una parte más que fundamental, sobre todo si deseamos desarrollar aplicaciones o flujos algo complejos. Pues el lenguaje de programación utilizado en estas herramientas se denomina Power FX.

Power FX es un lenguaje de programación funcional, declarativo, de tipo fuerte, de código bajo y de propósito general. Este nuevo lenguaje desarrollado por Microsoft está actualmente integrado únicamente con los productos de Power Platform, pero pronto estará disponible como software de código abierto, según anuncia la propia compañía.

La estructura de Power FX combina objetos con fórmulas declarativas que se asemejan a las hojas de cálculo. La lógica de estas fórmulas recalcula el valor automáticamente, es

decir, las fórmulas son evaluadas en tiempo real, permitiendo una interacción constante y dinámica con los datos y las acciones que realiza el usuario.

Sin embargo, Power Fx ofrece también lógica imperativa, lo cual es totalmente necesario a la hora de implementar funcionalidad en la aplicación. Y tanto para lógica declarativa como para la imperativa se utiliza el mismo lenguaje de expresión. Todo esto lo iremos viendo ejemplificado durante el desarrollo de la aplicación en todos y cada uno de los elementos.

2.2 Acerca de la empresa

Antes de empezar con el desarrollo de la aplicación, es necesario brindar cierto contexto también sobre la empresa para la cual va a ir destinada. Por supuesto, sin proporcionar el nombre de la empresa ni información adicional indebida para conservar así la protección de datos.

Esta empresa, en la que actualmente estoy realizando las prácticas, es una multinacional especializada en servicios de consultoría tecnológica. La cual ha ido creciendo de manera exponencial en los últimos años, alcanzando la cifra de casi 200.000 trabajadores en todo el mundo y contando, por tanto, con infinidad de proyectos activos y clientes satisfechos de todo el mundo.

El propósito principal de la empresa es la innovación, apostando sobre todo por la transformación digital con la intención de construir un futuro más próspero y contribuir así a mejorar la vida de las personas.

Por ello, y gracias a la gran cantidad de departamentos y secciones diferentes con los que cuenta la empresa, ofrecen a sus clientes todo tipo de servicios, tales como consultoría estratégica, infraestructuras, aplicaciones, servicios de BPO, etc. Operando además en un amplio espectro de sectores, desde servicios financieros y salud hasta automoción y sector público. Todo esto, con la intención de ayudar a los clientes a adaptarse a un entorno empresarial cada vez más competitivo y digitalizado.

Por otro lado, algunos de los valores que caracterizan a la empresa son el “Trabajo en equipo”, ya que trabajando con un objetivo en común y compartiendo sabiduría se obtienen resultados mejores, “El cliente es lo primero”, priorizándolo siempre para resolver sus inquietudes y garantizar su éxito, y “Previsión”, anticipándose siempre a al futuro y a los nuevos retos.

Sin dejar de lado otros valores como sostenibilidad, diversidad, equidad e inclusión. Lo cuales son indispensables en cualquier organización y en cualquier ámbito de la vida.

2.3 Vacaciones en una empresa

Por último, la temática y razón de la aplicación a desarrollar es la gestión de las vacaciones de los trabajadores en la empresa, por lo que es necesario conocer primeramente todo el funcionamiento y la legislación de estas vacaciones.

Las vacaciones en una empresa constituyen un derecho laboral fundamental que tiene como finalidad garantizar el descanso y el bienestar de los trabajadores. Este derecho se

encuentra recogido en el Estatuto de los Trabajadores, regulando así las condiciones y los requisitos mínimos de las vacaciones anuales.

Sin embargo, es necesario aclarar que estas condiciones de las vacaciones pueden variar según diferentes factores, tales como el país, los convenios colectivos que haya establecidos o el tipo de contrato que se haya firmado. Por lo que es fundamental que tanto los empleadores como los trabajadores estén siempre informados adecuadamente de estos factores con el fin de cumplir con todas las regulaciones correspondientes.

En cualquier caso, encontramos varios puntos clave que suelen ser comunes respecto al funcionamiento y legislación de las vacaciones laborables, y que, en este caso, van a ser fundamentales para poder realizar un correcto diseño y desarrollo de la aplicación deseada.

En primer lugar, uno de los puntos clave es la cantidad de días de vacaciones que corresponden anualmente. Según el artículo 38 del Estatuto de los trabajadores deberán concederse al menos 30 días naturales de vacaciones a los trabajadores por año natural. O lo que es lo mismo, 23 días laborables en el caso de que en tu convenio se contabilice en días laborables en vez de en días naturales. Es decir, cada trabajador genera 2,5 días naturales de vacaciones por mes trabajado, que luego podrá ir disfrutando a lo largo del año.

La diferencia entre los dos tipos de contabilización es simplemente que, si se hace en días naturales, a la hora de realizar el cómputo de días de vacaciones disfrutados deberemos contar también como disfrutados los días festivos y de descanso, cosa que en el caso de realizar el cómputo en días laborables no será necesario y solo contarán como disfrutados los propios días laborables.

Otro punto clave es cuándo se pueden disfrutar estos días de vacaciones acumulados y quién lo determina. De nuevo, aquí entra en juego el Estatuto de los Trabajadores, el cual en el artículo 38.2 determina que estos periodos de disfrute deben ser pactados entre el empresario y el trabajador. Sin embargo, en ciertos convenios colectivos es posible que se limite la disponibilidad de elección y se establezcan ciertos periodos obligatorios de disfrute de las vacaciones.

Para poder pactar estos periodos de disfrute, normalmente es el trabajador el que realiza solicitudes con antelación al empresario, siguiendo por supuesto con los procedimientos establecidos en la empresa. Y, por tanto, es el jefe o empresario el que tras valorar diferentes factores como disponibilidad del personal, carga de trabajo o temporada del año, responde al trabajador con la aprobación o el rechazo de los días de vacaciones que se habían solicitado. Por esto mismo, en la aplicación que se va a desarrollar se incluirá precisamente un sistema que agilice y simplifique todo el proceso de realizar las solicitudes, valorarlas y aprobarlas o rechazarlas.

Por otro lado, otro punto importante es que sucede si no se disfrutaron las vacaciones, ¿Se compensan de alguna manera los días no disfrutados a final de año? Pues la respuesta por norma general es no, las vacaciones no son sustituibles por compensación económica. Pero, evidentemente, existen ciertas excepciones, como en caso de despido, donde la empresa deberá abonar en el finiquito la cantidad equivalente a los días de vacaciones generados que no se ha disfrutado, también en caso de finalización de contrato por

cualquiera de las causas, o en caso de que el trabajador no haya podido disfrutar de sus vacaciones por una incapacidad temporal.

Además, en principio, tampoco se pueden acumular estas vacaciones no disfrutadas para otros años, aun que, de nuevo, dependiendo del convenio o empresa, puede haber situaciones en las que el trabajador disponga de un periodo extra del siguiente año para disfrutar de las vacaciones del año anterior. O en el caso de haber sufrido una incapacidad temporal y no haberlas disfrutado, dispones de 18 meses desde el fin de la incapacidad para hacerlo.

Por último, es importante saber que existen las llamadas ausencias laborales justificadas o ausencias legales. Estas se refieren a periodos en los que el empleado tiene derecho a ausentarse del trabajo de manera justificada y siendo igualmente remunerado, de acuerdo con la legislación laboral. Existen numerosos motivos por los que se pueden justificar estas ausencias, y por cada uno de ellos corresponden una diferente cantidad de días. Estos son algunos ejemplos de las ausencias legales más comunes:

Motivo	Días correspondientes
Matrimonio	15
Accidente o enfermedad de familiar cercano	5
Fallecimiento de familiar cercano	2-4
Mudanza	1
Deber público	Mínimo indispensable
Examen o cita médica	Mínimo indispensable
Nacimiento de hijos o lactancia	1 hora

Sabiendo todo esto, ya poseemos la contextualización necesaria en los diferentes ámbitos para proceder con el análisis de requisitos y el posterior desarrollo de la aplicación.

3 Análisis de requisitos

3.1 Explicación del sistema actual

Para poder realizar el desarrollo de una aplicación óptima que ofrezca todas las opciones y posibilidades que necesitamos, es fundamental realizar en primer lugar un análisis del sistema actual. Con la intención de identificar correctamente los problemas y debilidades que este sistema posee y poder así formular de manera creativa las posibles soluciones a estos problemas.

Como ya se ha mencionado anteriormente, el sistema actual se basa en una hoja de cálculo en Excel que se encuentra almacenada en la nube y a la cual tienen acceso los trabajadores de la empresa para consultar o modificar los datos.

Este Excel consta de dos hojas diferentes que actualmente tienen el nombre de “Cargabilidad 2024” y “Vacaciones”. La primera de ellas esta:

pestaña “Cargabilidad 2024” se van actualizando automáticamente los campos según se modifique el calendario laboral:

Las vacaciones no disfrutadas del año anterior se caducan en Marzo del año en curso					
USUARIO	AÑO VIGENTE	AJUSTE AÑO ANTERIOR	PLANIFICADO	CONFIRMADAS	PENDIENTE
Juan Enrique Lucas	23		0	1	22,00
Juan Luis Cuevas	23		6	0	17,00
Leopoldo Campo	23		15	7	1,00
Luciano Casanova					0,00
Manuel Jose Vallejo	23	3	0	3	23,00
Marco Sobrino	23		0	0	23,00
Marcos Moro	23	4	0	4	23,00
Mano Merchan	23		0	0	23,00
Matias Espinoza	23	4	4	1	22,00
Mikel Vergara	23		0	21	2,00
Narciso Reig	23	4	11	2	14,00
Nil Iniesta	23	6	12	5	12,00
Norberto Galera	23		0	5	18,00
Oier Amado	23	0	0	23	0,00
Oriando Mayor	21		21	0	0,00
Pascual Amaya	23	11	1	11	22,00
Pedro Jesus Murcia	23		0	0	23,00
Pere Alarcon	23	7	0	0	30,00
Pere Duarte	23	3	12	0	14,00
Placido Real	23		0	0	23,00
Rayan Flores	23	5	0	7	21,00
Rayan Prados	23		0	9	14,00
Robert Acosta	23		0	3	20,00
Rogelio Ferrero	23		20	0	3,00
Ruben Burgos	23		6	0	17,00
Rufino Abellan	23		0	2	21,00
Samuel Parra	23		0	0	23,00
Santiago Olmo	23		0	0	23,00
Saturmino Villaverde	23		0	0	23,00
Sebastian Velez	23	6	12	11	6,00
Unai Gomez	23		0	15	8,00
Victor Jose Casas	23	3	0	5	21,00
Yago Saura	23	4	17	4	6,00
Youssef Juarez	23	11	0	11	23,00

Figura 4: Captura Excel, Hoja "Vacaciones". Fuente: Elaboración propia

En esta tabla podemos observar los siguientes campos:

- Año vigente: Días de vacaciones que corresponden por el año actual. Si el trabajador va a permanecer en la empresa todo el año le corresponderán los 23 días de vacaciones, y si va a estar únicamente una cantidad de meses le corresponderán los días respectivos al tiempo que va a trabajar.
- Ajuste año anterior: Son los días de vacaciones que no se disfrutaron el año anterior. Tal y como se indica en la parte superior, estos días se podrán disfrutar únicamente hasta marzo del siguiente año, al superar esta fecha los días de vacaciones caducarán y se perderá el derecho a su disfrute.
- Planificado: Es la cantidad de días que el trabajador ha solicitado para disfrutar pero que aún no le han confirmado. Estos días de vacaciones aparecen en el calendario laboral como: **HP**
- Confirmadas: Es la cantidad de días de vacaciones que el trabajador tiene asignado de manera confirmada para el año vigente. Estos días aparecen en el calendario como: **H**
- Pendiente: Es la resta entre los días totales que dispones para disfrutar menos los días que ya tienes planificados o confirmados. Resultando en los días restantes que todavía no has utilizado y que, por tanto, tienes disponibles para solicitar.

Como hemos dicho antes, son tanto los empleadores como los trabajadores quienes, tras consensuarlo con sus superiores, modifican el calendario de la primera página a su elección. Dejando así constancia de las horas y días trabajadas y de todo tipo de ausencias o vacaciones que se produzcan. Este sistema posee ciertas ventajas y facilidades, pero a su vez, consta de ciertos problemas y puntos débiles que han de ser subsanados.

3.2 Identificación de vulnerabilidades del sistema actual y necesidades de mejora

Realizando un análisis tras el uso de esta hoja de cálculo durante varios meses, hemos podido observar que se generan los siguientes problemas:

1. Uno de los problemas principales que encontramos es que el sistema actual es un Excel compartido con todos los trabajadores que aparecen, es decir, los trabajadores del departamento en el que se utiliza. Esto implica que todos estos empleados pueden ver toda la información de los demás trabajadores. Tanto información personal (Número de empleado, equipo, nivel de inglés...) como información privada relacionada con los días de vacaciones disfrutados, las ausencias que van a suceder y los motivos de las mismas. Cuando toda esta información debería de ser privada excepto para los empleados de Recursos Humanos y los superiores de cada proyecto.
2. Además, no solo cualquier trabajador puede visualizar la información de los demás, si no que puede modificarla. Es obvio que en principio esto no va a suceder con malas intenciones, pero es posible que de manera involuntaria se modifique sin darse cuenta la información o el calendario de otro trabajador, lo que constituiría un gran problema.
3. Por otro lado, el hecho de que al abrir la hoja de cálculo aparezca el calendario con todos los meses y de todos los trabajadores a la vez puede resultar algo incómodo e incluso algo dificultoso a la hora de interpretar la información. Sobre todo, para una persona que únicamente desea consultar un mes concreto de su propio calendario laboral.
4. Ligado con el anterior punto, también encontramos problemas a la hora de interpretar el calendario debido a la leyenda. La cual se encuentra en la parte final de la hoja y, por tanto, si el usuario del cual se está consultando el calendario se encuentra en las primeras filas, puede resultar realmente incomodo tener que ir bajando y subiendo constantemente para poder descifrar las siglas y colores que corresponden a cada uno de los tipos de ausencias.
5. También se podría contar como un punto débil el hecho de que este sistema no cuenta con ninguna auditoría en la que se registren los cambios que se realizan en la información. Ni, por supuesto, la fecha y el autor de estos mismos cambios.
6. Por último y en general, es un sistema no suficientemente automatizado, ya que debemos ir introduciendo y modificando la información de forma manual en la mayoría de los casos. Esto puede conllevar que se produzcan más errores en la información que se maneja o que simplemente esta información no esté completamente actualizada.

3.3 Posibles soluciones

En primer lugar, para poder acabar tanto con el primero como con el segundo problema, sería necesario crear un sistema de roles de seguridad en el que cada usuario tuviera ciertos permisos. Mediante estos roles, que dependerían del puesto en la empresa, cada usuario podría visualizar o modificar una información u otra. Habría que valorar la cantidad

de roles diferentes que se necesitan y las capacidades y permisos que va a tener cada uno de ellos.

En segundo lugar, sería interesante crea una pantalla en la aplicación donde el trabajador pueda consultar únicamente su calendario laboral de una manera más clara y mejor diseñada, con la posibilidad de ir avanzando o retrocediendo entre meses y pudiendo visualizar así únicamente la información relevante para el trabajador. Con esta posible solución solventaríamos el tercer problema mencionado en la sección anterior.

Además, en esa misma pantalla podría ser buena opción dejar suficiente espacio para incluir la leyenda del calendario. Así se podría subsanar el cuarto problema ya que se podría ir consultando la leyenda y el calendario al mismo tiempo, facilitando su interpretación y mejorando la satisfacción del trabajador al utilizar la aplicación.

Por otro lado, con el fin de automatizar más aún el sistema se deberían implementar flujos de trabajo en Power Automate. Los cuales estarían conectados tanto con la aplicación como con la base de datos e irían actualizando automáticamente la información sin necesidad de intervención humana y, por tanto, evitando ciertos errores que esta intervención puede producir.

Además, al utilizar Microsoft Dataverse como base de datos, en las tablas se van a ir guardando los datos de las fechas y los autores de cada registro nuevo que se cree o incluso de las modificaciones que se hagan en ellos. Con esto acabaríamos con el problema número cinco mencionado anteriormente, ya que en todo momento se podría saber cuándo y quien ha aceptado las solicitudes, las ha creado o simplemente ha modificado algo de información.

4 Diseño de la arquitectura

4.1 Definición de los componentes técnicos

Para poder definir la arquitectura de la aplicación a desarrollar, es necesario establecer sus diferentes componentes en un diagrama como el realizado a continuación:

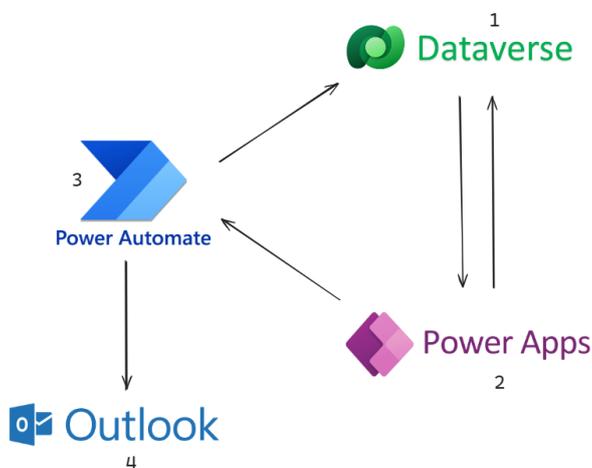


Figura 5: Diagrama de la arquitectura de la solución. Fuente: Elaboración propia

1. En primer lugar, encontramos la base de datos, la cual se va a establecer en Microsoft Dataverse. Esta plataforma, como ya se ha mencionado anteriormente, forma parte del conjunto de herramientas de Power Platform. Esta base de datos estará formada por un conjunto de Tablas y campos relacionados entre sí, que irán proporcionando información y datos al segundo componente, la propia aplicación.
2. La aplicación, desarrollada y establecida en Power Apps, irá utilizando la información proporcionada por la base de datos con el fin de mostrarla a lo largo de las diferentes pantallas. Además, en ciertas acciones, la aplicación actuará de vuelta sobre el componente 1, la base de datos, modificando la información existente o creando nuevos registros.
Por otro lado, encontraremos situaciones en las que desde la aplicación se llamen y se lancen diferentes flujos de trabajo, que constituyen el tercer componente.
3. Es a través de Power Automate donde se desarrollarán y posteriormente se ejecutarán los flujos de trabajo. Estos tendrán diferentes funciones, algunos de ellos se encargarán de modificar información en la base de datos cuando sucedan determinadas acciones en la aplicación, otros estarán programados para también actuar en la base de datos pero en un momento concreto deseado, y otros se encargarán de enviar notificaciones/avisos cuando se cumplan ciertas condiciones a través del cuarto componente.
4. Este componente (Outlook), ya más externo que los tres anteriores, será el utilizado por los flujos para mandar correos electrónicos como avisos de situaciones que se han cumplido. Un ejemplo de esto serán los correos que se mandarán a los responsables cuando se creen nuevas solicitudes de días de vacaciones.

4.2 Definición del modelo de datos

Como hemos comentado, la base de datos se va a almacenar y manejar en Microsoft Dataverse. Y tras valorar y analizar las diferentes opciones respecto a que tablas y campos iban a ser necesarios para la aplicación, se va a establecer el siguiente modelo de datos:

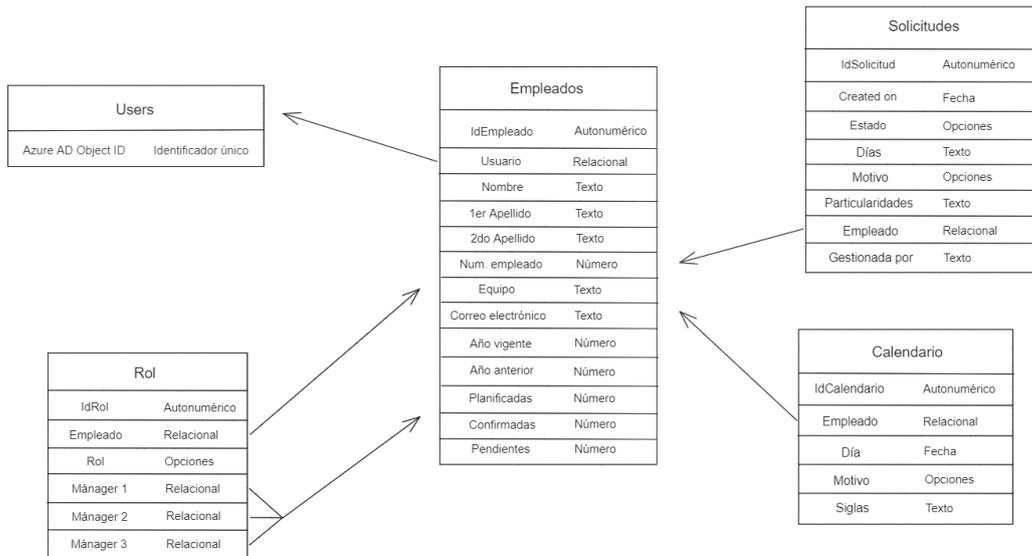


Figura 6: Diagrama de la base de datos. Fuente: Elaboración propia

En el anterior diagrama podemos observar todas las tablas que se van a emplear, los campos de cada una (columna de la izquierda) y los tipos de datos de cada uno de los campos (columna de la derecha).

En primer lugar encontramos la tabla “Empleados”, que se considera la tabla principal del modelo de datos. Esta contiene, por un lado, campos de información básica de los empleados, tales como el nombre, los apellidos, el número de empleado, el correo electrónico o el equipo al que pertenece. Por otro lado, también contiene los campos sobre la situación de días de vacaciones en la que está el empleado. Estos campos de tipo numérico son la cantidad de días de vacaciones correspondientes a este año vigente, los correspondientes al año anterior, los planificados, los ya confirmados, y los pendientes a utilizar.

Además, en esta tabla encontramos el campo Usuario, el cual está precisamente relacionado con la tabla “Users”. La cual no es una tabla creada por nosotros, si no que es una tabla ya existente que contiene los datos e identificadores de los usuarios de la plataforma, relacionando así las cuentas de los usuarios con cada uno de los empleados.

Por otro lado, observamos la tabla “Rol” que está conectada con la tabla “Empleados” mediante el propio campo llamado Empleado. Esta tabla define principalmente el rol que tiene cada trabajador (se verán los posibles roles en el siguiente apartado) y define también cuales son los mánagers del empleado, como se puede observar cada trabajador puede llegar a tener hasta 3 mánagers diferentes.

Otra de las tablas que vamos a necesitar es la tabla “Solicitudes”, donde se almacenarán los registros de cada solicitud de vacaciones que realicen los trabajadores. Los campos que encontramos en esta tabla son los datos de cada solicitud, tales como el estado de la solicitud, el motivo, la fecha en que ha sido creada, los días que se han solicitado, el nombre de la persona que la ha gestionado, en caso de que así haya sido, o algunas otras

particularidades de la solicitud. Por supuesto cada solicitud está relacionada con el trabajador a la que pertenece mediante el campo Empleado.

Y finalmente, encontramos la tabla “Calendario” que va a ser la utilizada para registrar y almacenar individualmente cada día festivo de cada trabajador, tanto si son festivos nacionales, como días de vacaciones pendientes/confirmados, como días de libre disposición, o cualquier otro tipo de ausencia. Por eso mismo, los campos principales que encontramos en esta tabla son: el día, el motivo de la ausencia, las siglas con las que se representa en el calendario y, por supuesto, el empleado al que corresponde la ausencia, que se relaciona precisamente a la tabla “Empleados”.

4.3 Definición de roles de seguridad

Tal y como se ha detectado en el análisis de requisitos, el problema principal del antiguo sistema era que cualquier persona podía tanto visualizar como modificar la información de cualquier otro trabajador. Constituyendo así una gran falta de seguridad y privacidad que necesitaba ser subsanada con el nuevo sistema.

Por eso mismo, y como se ha mencionado anteriormente, mediante la tabla “Rol” se va a establecer el rol de cada trabajador. Estipulando así las opciones y permisos que cada empleado va a tener disponibles en la aplicación. Y tras analizar y valorar las necesidades de diferenciación que deben tener los usuarios, se van a establecer los tres tipos siguientes de roles:

- **Empleado:** Es el rol más básico, únicamente podrá visualizar en la aplicación su información, no la de sus compañeros ni sus managers. Podrá observar tanto su información básica como su calendario laboral detallado. Además, podrá realizar solicitudes de días de vacaciones y observar sus solicitudes pasadas.
- **Mánager:** Como un trabajador más, podrá realizar todas las funciones que posee el rol de Empleado. Sin embargo, las personas con este rol podrán observar la información y calendario laboral de todos los empleados que están en su equipo/proyecto, es decir, de todos los trabajadores que le tengan a él como manager. Además, será el encargado de gestionar las solicitudes de estos mismos empleados, teniendo la capacidad de aceptar/rechazar las peticiones de vacaciones.
- **Administrador:** Es el rol con más permisos y posibilidades. En la aplicación es capaz de ver toda la información y calendarios de todos los trabajadores, no solo los de sus proyectos. Por supuesto también puede visualizar todas las solicitudes y gestionarlas. Y, además, también tiene la posibilidad de eliminar y añadir festivos directamente a cualquier usuario. Incluso, puede actuar directamente sobre la base de datos para modificar cualquier campo o registro.

La idea es que justo nada más ejecutar la aplicación se evalúe el rol que tiene el usuario, mostrado así directamente la aplicación personalizada con las opciones que ese usuario tiene disponibles para ver o realizar.

5 Desarrollo de la solución

Una vez definida la arquitectura con todos los componentes, se puede proceder con el desarrollo de la aplicación. Para ello, vamos a crear en nuestro entorno de Power Platform una nueva solución, la cual se va a llamar “Gestor de Vacaciones” y es donde se van a albergar todos los componentes del sistema, desde las tablas de la base de datos, hasta los flujos de trabajo o la propia aplicación.

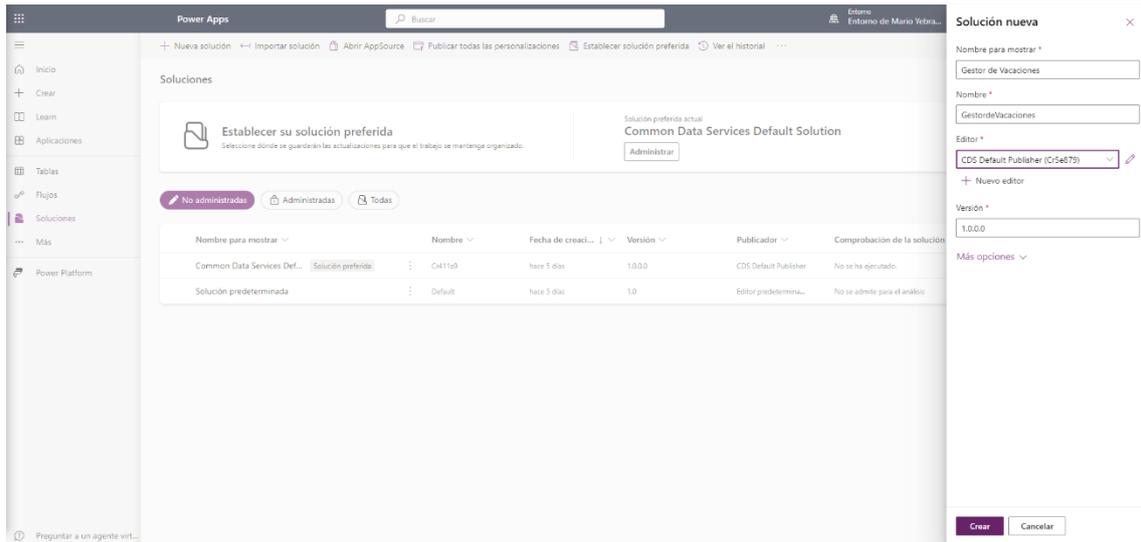


Figura 7: Captura creación de la solución. Fuente: Elaboración propia

Gracias a que todo va a estar almacenado en una misma solución, luego va a ser posible mover todo el sistema de manera compacta al entorno que deseemos.

Una vez ya creada la solución que va a almacenar todo el sistema, siguiendo con el orden de los componentes técnicos, es necesario crear primeramente la base de datos en Microsoft Dataverse.

5.1 Desarrollo de la base de datos

Para ir formando la base de datos de nuestro sistema, dentro de la solución vamos a ir creando tablas y conectándolas entre si según el modelo de datos que hemos diseñado. Empezamos por la tabla “Empleados” que, como hemos mencionado, será la tabla principal del modelo de datos.

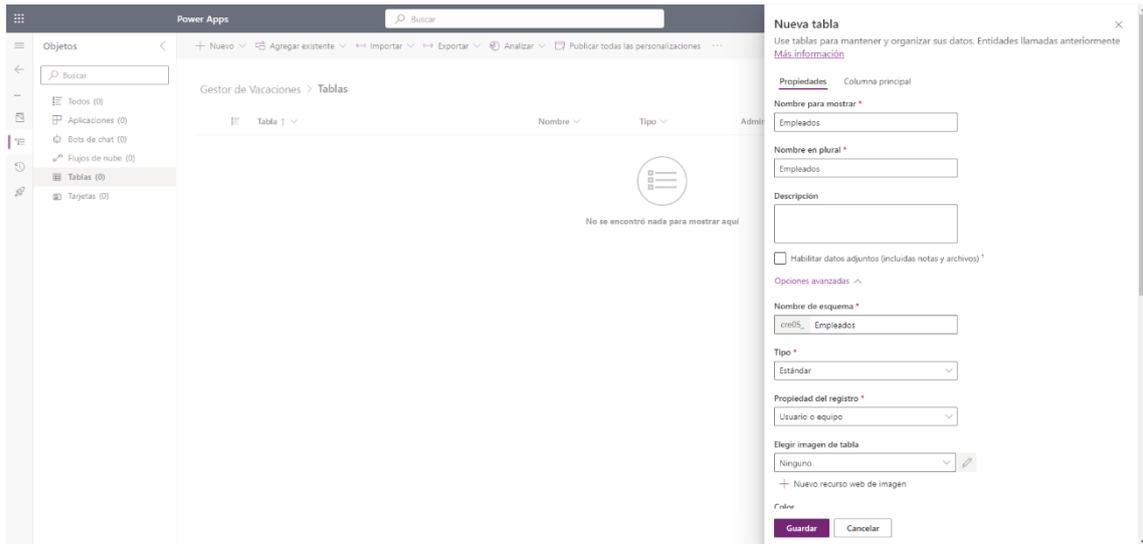


Figura 8: Captura creación de una nueva tabla. Fuente: Elaboración propia

Dentro de la propia tabla creada, vamos creando las diferentes columnas o campos indicando el tipo de datos de cada uno de ellos. Estos nuevos campos se van uniendo a los que se crean por defecto.

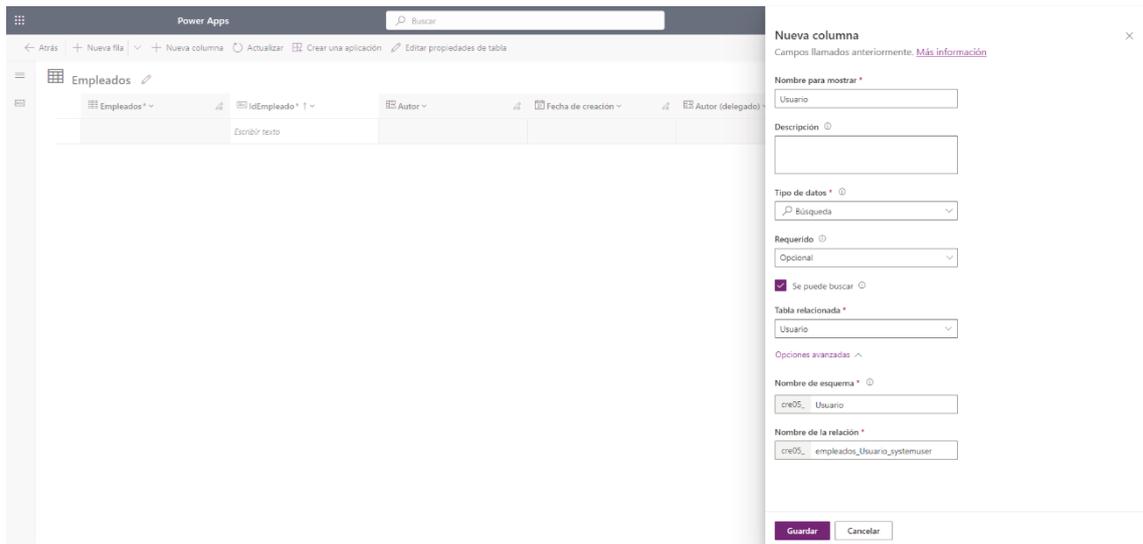


Figura 9: Captura creación de una nueva columna. Fuente: Elaboración propia

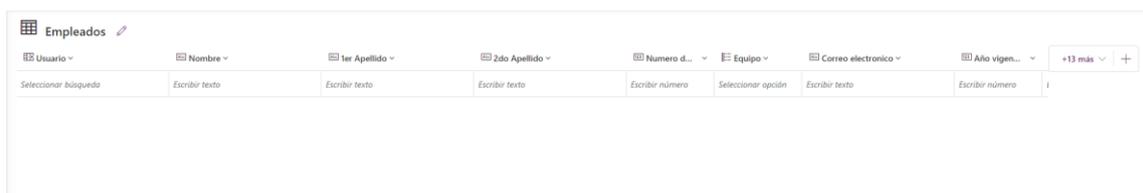


Figura 10: Captura resultado tabla Empleados. Fuente: Elaboración propia

Y tras realizar este mismo proceso con las cinco tablas que componen el modelo de datos, tenemos ya establecida lo que será la estructura de la base de datos que utilizará la aplicación:

Tabla 1	Nombre	Tipo	Administrada	Personalizado	Personalizable	Etiquetas
Calendario	cre05_calendario	Standard	No	No	Si	Personalizado
Empleados	cre05_empleados	Standard	No	No	Si	Personalizado
Rol	cre05_rol	Standard	No	No	Si	Personalizado
Solicitud	cre05_solicitud	Standard	No	No	Si	Personalizado
Usuario	systemuser	Standard	Si	No	Si	Estándar

Figura 11: Captura resultado de todas las tablas creadas. Fuente: Elaboración propia

5.2 Desarrollo de la aplicación

Tras todo el análisis de requisitos y el diseño de la arquitectura, para poder comenzar con el desarrollo puro de la aplicación se ha realizado un primer boceto del diseño y funcionalidad de cada una de las pantallas que se van a necesitar. Por supuesto también, confeccionando cuál sería la conexión entre cada una de ellas y marcando así las interacciones que debe de realizar el usuario para navegar entre las mismas. Para todo ello, se ha tenido en cuenta todas las funcionalidades del sistema antiguo que deben permanecer y se han evaluado las posibles soluciones y mejoras mencionadas en el apartado 3.3

El resultado ha sido el siguiente diagrama:

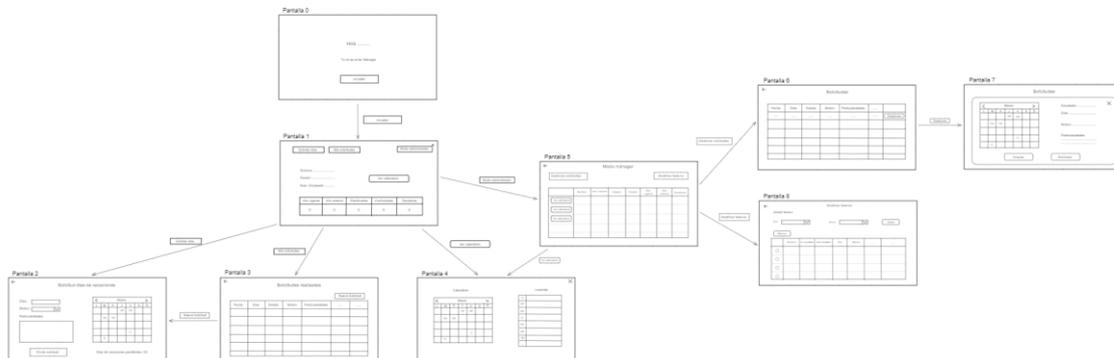


Figura 12: Diagrama de pantallas de la aplicación. Fuente: Elaboración propia

Dicho esto, se puede proceder con el detallamiento y el desarrollo individual de cada una de las diferentes pantallas.

5.2.1 Pantalla 0 - Bienvenida

5.2.1.1 Análisis funcional

Esta pantalla, que va a ser una pantalla previa a acceder a la aplicación, debe contener las siguientes funciones:

- Mostrará un mensaje de bienvenida.
- Evaluará el rol del usuario que acaba de acceder, lo mostrará por pantalla y lo guardará en una variable, la cual se utilizará en las siguientes pantallas para adaptar el sistema al rol que posee el usuario.
- Contendrá un botón que nos permita acceder a la aplicación al pulsarlo.
- En el caso de que el usuario que haya accedido no tenga acceso, se mostrará un mensaje indicándolo y no estará disponible el botón de acceder.

Por ello, este sería un primer boceto de la pantalla, a partir del cual poder basarnos para desarrollarla:

Pantalla 0



Figura 13: Boceto previo pantalla 0. Fuente: Elaboración propia

Respecto a la gama de colores que se va a utilizar en el diseño de la aplicación, se ha valorado si utilizar tonos azules o verdes. Concluyendo en la opción de utilizar colores azules y por supuesto, blancos, para el diseño general, ya que el color azul transmite tranquilidad, confianza y seguridad, valores que deseamos que estén presentes en este sistema de gestión.

5.2.1.2 Desarrollo

Para dar un poco de contexto inicial sobre el funcionamiento del desarrollo en Power Apps, cuando seleccionamos crear una nueva aplicación y le damos a editar nos encontramos con el siguiente editor:

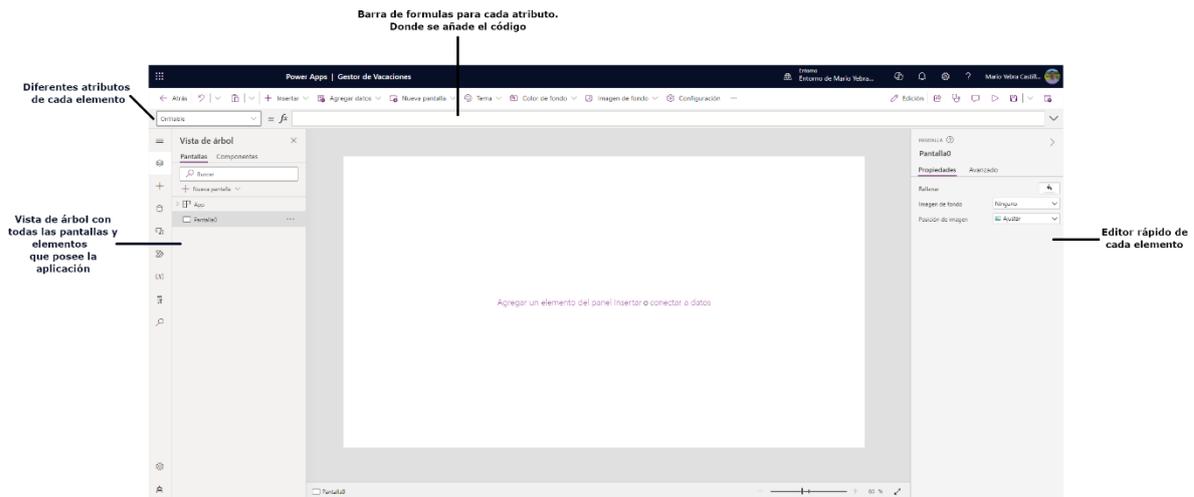


Figura 14: Editor de Power Apps nada más iniciar. Fuente: Elaboración propia

En el cuál, iremos creando pantallas nuevas y añadiéndoles elementos. Estos elementos están formados y definidos por atributos tales como: Color, Height, Width, Text, OnSelect, Visible, DisplayMode, Padding, X, Y, etc. Y en cada atributo de estos iremos añadiendo código a través de la barra de fórmulas para conseguir la personalización y la funcionalidad deseada de cada elemento existente.

Por ejemplo, en el desarrollo de esta 'Pantalla 0' encontraremos los siguientes elementos con sus atributos más relevantes:

Al arrancar la app:

Elemento	Atributos	Código
App	OnStart	<pre>If(!IsEmpty(Filter(Empleados; Usuario.'Id. de objeto de Azure AD' = User().EntraObjectId)); Set(GUIDEmpleado; Lookup(Empleados; Usuario.'Id. de objeto de Azure AD' = User().EntraObjectId; Empleados)));; Set(RolEmpleado; Lookup(Roles; Empleado.Empleados = GUIDEmpleado).'Rol (cre05_rol)')</pre>

Con esto conseguimos que nada más el empleado ejecute la app, se guarde en una variable llamada "GUIDEmpleado" el identificador del empleado que ha accedido (Siempre y cuando ese empleado este registrado en la base de datos y tenga un rol asignado), esto se utilizará tanto para esta pantalla 0, como a lo largo de la aplicación.

Además, establecemos también en la variable RolEmpleado el rol que tiene dicho empleado para posteriormente utilizarla de manera sencilla y sin tener que realizar el LookUp a la base de datos en cada ocasión.

Primera etiqueta de texto

Elemento	Atributos	Código
Label0_1	Text	<code>Concatenate("Hola "; LookUp(Usuarios; 'Id. de objeto de Azure AD' = User().EntraObjectId; 'Nombre de pila'); ", bienvenido al Gestor de Vacaciones")</code>
	X	<code>Parent.Width/2 - Self.Width/2</code>

En el atributo Text, a través del LookUp a la tabla “Usuarios” obtenemos y añadimos a la cadena de texto el nombre de la persona que ha accedido, ya sea si es un usuario con o sin rol para acceder a la app.

Y en el atributo X utilizamos esa fórmula para que independientemente de la longitud del texto siempre este centrado en la Pantalla.

Segunda etiqueta de texto

Elemento	Atributos	Código
Label0_2	Text	<code>If(!IsBlank(GUIEmpleado); Concatenate("Tu rol es: "; LookUp(Roles; Empleado.Empleados = GUIEmpleado; 'Rol (cre05_rol)')); "No tienes permiso para acceder al gestor")</code>
	X	<code>Parent.Width/2 - Self.Width/2</code>

Mediante al atributo Text, en este label se mostrará el Rol que posee el empleado que ha ejecutado la app, y si este no posee Rol, se mostrará “No tienes permiso para acceder al gestor”. Esta condición se evalúa gracias a si la variable GUIEmpleado creada en el OnStart de la App está vacía o contiene datos.

Botón de acceder

Elemento	Atributos	Código
Button0_Acceder	OnSelect	<code>Navigate(Pantalla1; ScreenTransition.Fade)</code>
	DisplayMode	<code>If(!IsBlank(GUIEmpleado); DisplayMode.Edit; DisplayMode.Disabled)</code>
	X	<code>Parent.Width/2 - Self.Width/2</code>

	Text	"Acceder"
--	------	-----------

Con el Onselect establecemos que al pulsarlo se nos dirija a la Pantalla 1 usando una transición tipo Fade. Y con el DisplayMode indicamos que, si el usuario no tiene el rol para acceder, el botón aparezca desactivado y no se pueda pulsar.

Además en la parte derecha del editor, se han establecido los colores del botón y el borde del mismo, tanto su color estándar, como el color para cuando pasas por encima, como para cuando los pulsas.

5.2.1.3 Resultados

Con todos los elementos añadidos y ciertos detalles referidos al diseño, la pantalla 0 de la aplicación resultaría así. Este es el ejemplo de una persona llamada Mario y que tiene el Rol de Administrador:



Figura 15: Resultado 1 de la pantalla 0. Fuente: Elaboración propia

En el caso de una persona que no tuviera ningún rol asignado en la base de datos y, por tanto, no tuviera acceso a la app quedaría así:



Figura 16: Resultado 2 de la pantalla 0. Fuente: Elaboración propia

Y como ya hemos mencionado, al apretar el botón de acceder (si es posible) nos llevaría a la pantalla 1.

5.2.2 Pantalla 1 - Inicio

5.2.2.1 Análisis funcional

Esta será la pantalla inicial y probablemente principal de la aplicación. Ya que es la pantalla que nos va a dar acceso para la mayoría de las siguientes pantallas, y por tanto, será la pantalla a la que volveremos recurrentemente entre las diferentes tareas que realicemos en la app.

Las funciones y elementos que ha de tener esta pantalla 1 son los siguientes:

- Mostrará información básica e invariable del propio empleado que ha accedido, como el nombre, número de empleado, equipo...
- Mostrará también información variable y relevante como los días de vacaciones que les corresponden, los que le sobraron del año pasado, los que ya tiene confirmados, etc.
- Deberá haber un botón que permita visualizar nuestro calendario laboral con todas los festivos y ausencias indicadas.
- Deberá tener dos botones que nos den acceso a la página para solicitar vacaciones y a la página para visualizar nuestras solicitudes realizadas respectivamente.
- También deberá haber un botón extra que en función del rol del usuario esté visible o no para permitir (o no) el acceso al modo administrador.

Sabiendo todo esto, se ha diseñado este primer boceto de la pantalla:

Pantalla 1

Año vigente	Año anterior	Planificadas	Confirmadas	Pendiente
X	X	X	X	X

Figura 17: Boceto previo de la pantalla 1. Fuente: Elaboración propia

El cual, como todos, está sujeto a cambios por posibles problemas que podamos encontrar posteriormente o por simples mejoras logísticas.

5.2.2.2 Desarrollo

Estos son algunos de los elementos más importantes con sus atributos más relevantes de esta pantalla:

Al visualizar la pantalla

Elemento	Atributos	Código
Pantalla1	OnVisible	<code>Set(_Empleado; LookUp(Empleados; Empleados = GUIDEmpleado))</code>

Con esto, nada más visualizar esta pantalla, establecemos en la variable `_Empleado` todo el registro de datos del empleado que ha accedido, facilitando así que luego podamos acceder a esos datos en otros elementos de manera más rápida.

Etiquetas con los títulos

Elemento	Atributos	Código
Label1_Equipo	Text	<code>"Equipo: "</code>
	X	<code>Label1_Nombre.X</code>
	Y	<code>Label1_Nombre.Y + Label1_Nombre.Height +5</code>

Este es el caso concreto de la etiqueta de “Equipo”, pero todas estas etiquetas de con los títulos son similares. Tanto la X como en la Y están relacionadas con las demás etiquetas para conseguir una mejor alineación de las etiquetas, incluso para tener la posibilidad de moverlas conjuntamente durante el desarrollo.

Etiquetas con los datos

Elemento	Atributos	Código
Label1_Equipo2	Text	<code>_Empleado.Equipo</code>
	X	<code>Label1_Nombre2.X</code>
	Y	<code>Label1_Equipo.Y</code>

Misma situación con las etiquetas que contienen los datos. En el atributo Text, a través de utilizar “_Empleado.xxxx” accedemos al dato deseado del empleado.

Botón “Solicitar días”

Elemento	Atributos	Código
Button1_SolicitarDias	OnSelect	<code>Navigate(Pantalla2; ScreenTransition.Fade)</code>
	Text	<code>"Solicitar días"</code>
	Color	<code>RGBA(255; 255; 255; 1)</code>
	Fill	<code>RGBA(137; 196; 244; 1)</code>
	BorderColor	<code>ColorFade(Self.Fill; -15%)</code>
	HoverFill	<code>RGBA(0; 134; 208; 1)</code>
	HoverBorderColor	<code>Self.HoverFill</code>
	PressedColor	<code>Self.Fill</code>
	PressedFill	<code>Self.Color</code>

De nuevo con el OnSelect, establecemos que al presionar el botón se nos lleve a la pantalla 2. Y con los atributos de Color, Fill, BorderColor, HoverColor, HoverFill, HoverBorderColor, PressedColor y alguno más, conseguimos establecer la combinación de colores deseada para cada situación del botón. Tanto sus colores en estado de reposo, como sus colores al situar el ratón encima, como sus colores al pulsarlo. Cuando por ejemplo un atributo de este tipo tiene como código: `Self.HoverFill`, significa que el contenido de ese atributo es el mismo que el contenido que hay en el atributo HoverFill de ese mismo elemento.

Botón “Mis Solicitudes”

Elemento	Atributos	Código
Button1_MisSolici	OnSelect	<code>Navigate(Pantalla3; ScreenTransition.Fade)</code>
	Text	<code>"Mis solicitudes"</code>
	X	<code>Button1_SolicitarDias.X</code>
	Y	<code>Button1_SolicitarDias.Y + Button1_SolicitarDias.Height +35</code>

Respecto a los colores de este botón, se mantiene exactamente la misma combinación que en el botón de “Solicitar días”, con la intención así de guardar una armonía y semejanza de colores. Y tanto en la X como en la Y referenciamos la posición con respecto al primer botón, para conseguir que estén alineados correctamente.

Botón “Calendario”

Elemento	Atributos	Código
Button1_Calendario	OnSelect	<code>Navigate(Pantalla4; ScreenTransition.Fade)</code>
	Text	<code>"Ver calendario"</code>
	X	<code>Button1_SolicitarDias.X</code>
	Y	<code>Button1_MisSolici.Y + Button1_MisSolici.Height +35</code>

De nuevo en este botón se mantienen los mismos colores en cada respectiva situación y, además, el botón se encuentra también alineado con los anteriores gracias a los atributos X e Y. Simplemente que, en este caso, al pulsarlo se nos llevará a la pantalla 4, en la cual el usuario podrá observar su calendario laboral.

Botón “Modo administrador”

Elemento	Atributos	Código
Button1_ModoAdmin	OnSelect	<code>Navigate(Pantalla5; ScreenTransition.Fade)</code>

	Text	<code>If(RolEmpleado = 'Roles Empleados'.Administrador; "Modo administrador"; "Modo Mánager")</code>
	Visible	<code>If(RolEmpleado = 'Roles Empleados'.Basico; false; true)</code>
	Color	<code>RGBA(255; 255; 255; 1)</code>
	Fill	<code>RGBA(0; 134; 208; 1)</code>
	BorderColor	<code>ColorFade(Self.Fill; -15%)</code>
	HoverFill	<code>ColorFade(Self.Fill; -15%)</code>
	HoverBorderColor	<code>Self.HoverFill</code>
	PressedColor	<code>Self.Fill</code>
	PressedFill	<code>Self.Color</code>

En el caso de este botón, si lo presionamos nos llevaría a la Pantalla 5 y, sin embargo, en este botón se utiliza otra combinación de colores azules. Ya que este botón se encuentra sobre la cabecera, la cual es del mismo color que los botones anteriores, necesitando así diferenciar el fondo del botón.

Por otro lado, a través del atributo Text establecemos que dependiendo de si el rol del empleado es Administrador o Mánager aparezca como texto “Modo administrador” o “Modo mánager” respectivamente. Y, además, en el Visible indicamos que si el empleado posee el rol Básico directamente el Botón no sea visible, y por tanto, no sea necesario un texto específico para ese rol.

Por último, para mostrar los datos variables de las cantidades de días de vacaciones, se ha creado un contenedor en el que se han ido añadiendo rectángulos para formar las tablas, y etiquetas de texto para completar los datos.

Estas etiquetas que se usan en la cabecera de la tabla (Label1_xxxx) son muy similares a las “Etiquetas con los títulos” explicadas anteriormente, y las etiquetas que rellenan la tabla (Label1_xxxx2) son muy similares a las “Etiquetas con los datos” que también han sido explicadas con anterioridad.

Esta es la vista de árbol de este contenedor explicado:

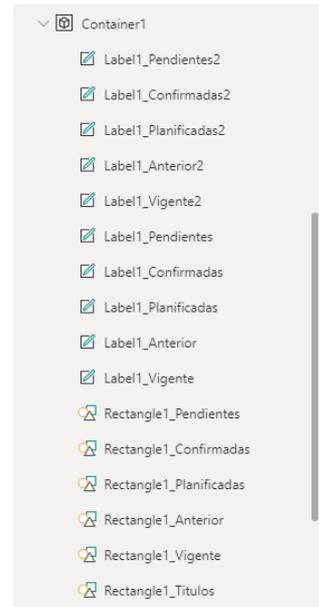


Figura 18: Vista de árbol del Container1. Fuente: Elaboración propia

5.2.2.3 Resultados

Una vez desarrollada toda la pantalla, este sería el resultado visual de la misma si se trata de un usuario con el rol más básico:



Figura 19: Resultado 1 de la pantalla 1. Fuente: Elaboración propia

Sin embargo, si el usuario posee el rol de Mánager o de Administrador, la pantalla aparece con un botón más que además refleja el respectivo rol. Quedando así en cada situación:

Gestor de vacaciones
Modo Mánager

Nombre:	Mario Yebra Castillo	Solicitar días Mis solicitudes Ver calendario
Equipo:	Valencia	
Núm. de empleado:	275639	
Correo electrónico:	yebracastillo@gmail.com	
Proyecto:	Trabajo fin de grado	
CAR: Sí	Activo: Sí	

Año vigente	Año anterior	Planificadas	Confirmadas	Pendientes
23 días	2 días	3 días	5 días	17 días

Figura 20: Resultado 2 de la pantalla 1. Fuente: Elaboración propia

Gestor de vacaciones
Modo administrador

Nombre:	Mario Yebra Castillo	Solicitar días Mis solicitudes Ver calendario
Equipo:	Valencia	
Núm. de empleado:	275639	
Correo electrónico:	yebracastillo@gmail.com	
Proyecto:	Trabajo fin de grado	
CAR: Sí	Activo: Sí	

Año vigente	Año anterior	Planificadas	Confirmadas	Pendientes
23 días	2 días	3 días	5 días	17 días

Figura 21: Resultado 3 de la pantalla 1. Fuente: Elaboración propia

Como se puede apreciar, la distribución ha cambiado ligeramente respecto al boceto inicial de la pantalla con la intención de mejorar el diseño, haciéndolo más claro y ordenado.

Por otro lado, los botones funcionan correctamente, dirigiendo a la pantalla deseada en cada caso. La información mostrada es dinámica y específica para el usuario que ha iniciado sesión, ajustándose en tiempo real a los datos almacenados en la base de datos.

5.2.3 Pantalla 2 – Solicitar días

5.2.3.1 Análisis funcional

Esta va a ser la pantalla a la que nos dirigamos al pulsar el botón “Solicitar días” en la pantalla 1. Y precisamente consistirá en una pestaña donde podamos enviar solicitudes de días de vacaciones. Por tanto, deberá contener las siguientes opciones y funcionalidades:

- Principalmente la pantalla debe ser un formulario, el cual el usuario deberá rellenar y enviar para realizar la solicitud.
- Este formulario deberá contener campos como: Los días de vacaciones que queremos seleccionar, el motivo de la ausencia de estos días y un campo opcional de particularidades que se podrá rellenar si queremos detallar ciertas cosas de la solicitud.
- Además, tiene que haber un botón que permita enviar y, por tanto, registrar la solicitud.
- Es necesario también que se indique la cantidad de días
- Y, por último, el usuario también debería poder observar su calendario laboral a la hora de solicitar los días con el fin de adecuar su selección a los festivos o ausencias que ya tenga marcadas. Para este caso encontramos dos opciones:
 - La primera que este calendario sirva únicamente para visualizarlo, teniendo que seleccionar los días escribiéndolos en una entrada de texto o algo similar.
 - Y la segunda que este calendario sirva, además de para poder visualizarlo, para seleccionar pulsando sobre él los días que queremos solicitar.

Valorando las dos posibilidades consideramos que es más útil y práctica la segunda de ellas, así que, siempre y cuando sea factible de implementar, será la opción elegida para el desarrollo.

Sabiendo todo lo que ha de tener esta pantalla, se ha diseñado el siguiente boceto de esta:

Pantalla 2

←
Solicitud días de vacaciones

Días:

Motivo:

Particularidades:

Marzo						
<	L	M	X	J	V	>
				HP	HP	
		CE	CE			
					H	
	C					

Días de vacaciones pendientes: XX

Figura 22: Boceto previo de la pantalla 2. Fuente: Elaboración propia

5.2.3.2 Desarrollo

Para facilitar el desarrollo del calendario interactivo que encontramos tanto en esta pantalla como en algunas de las siguientes, vamos a establecer primeramente los colores de cada tipo de ausencia en el atributo Formulas del elemento App:

Elemento	Atributos	Código
App	Formulas	<pre> fxColor = { // Named Color Values ColorFD: RGBA(48;84;150;1); ColorF: RGBA(150;15;104;1); ColorUS: RGBA(226;254;106;1); ColorCM: RGBA(255;102;0;1); ColorCE: RGBA(154;174;4;1); ColorOP: RGBA(51;204;204;1); ColorBH: RGBA(255;0;0;1); ColorH: RGBA(51;102;255;1); ColorC: RGBA(33;89;103;1); ColorB: RGBA(255;0;255;1); ColorMI: RGBA(204;153;255;1); ColorP: RGBA(247;150;70;1); ColorLA: RGBA(153;51;0;1); ColorD: RGBA(0;255;0;1); ColorPC: RGBA(166;166;166;1); ColorHP: RGBA(255;204;0;1) }; </pre>

Dejando así claras, mediante RGBA, las tonalidades exactas de los colores de cada tipo de festivo.

Respecto al calendario, el cual también va a ser utilizado en siguientes pantallas, está formado por diversos elementos y componentes. Estos son algunos de los más relevantes:

Calendario

Primeramente, se inicializan las siguientes variables nada más sea visible la pantalla 2

Elemento	Atributos	Código
Pantalla2	OnVisible	<pre> Set(_firstDayOfMonth; DateAdd(Today(); 1 - Day(Today()); TimeUnit.Days)); Set(_firstDayInView; DateAdd(_firstDayOfMonth; -(Weekday(_firstDayOfMonth; StartOfWeek.Monday) - 2 + 1); TimeUnit.Days)); Set(_lastDayOfMonth; DateAdd(DateAdd(_firstDayOfMonth; 1; TimeUnit.Months); -1; TimeUnit.Days)); Clear(DiasSeleccionados); Set(Diastexto;""); Reset(Dropdown2_Motivo); Reset(TextInput2_Particularidades) </pre>

De esta manera, se establecen las variables que se utilizan en los componentes del calendario como: El primer día del mes en el que nos situamos (el actual de forma predeterminada), el primer día que se muestra y el último día del mes mostrado. Además se reinician varias variables y elementos para que no aparezca lo seleccionado anteriormente.

Por otro lado, el elemento principal de este calendario es una galería de ítems (MonthDayGallery2). Cada ítem de la galería consta de un contenedor (Container3) que a su vez contiene dos etiquetas (Title2_Dia y Label2_Calend). Cada uno de estos ítems corresponderá a un día del mes, mostrando en la primera de las etiquetas el número del día y en la segunda los colores/siglas de ese día.

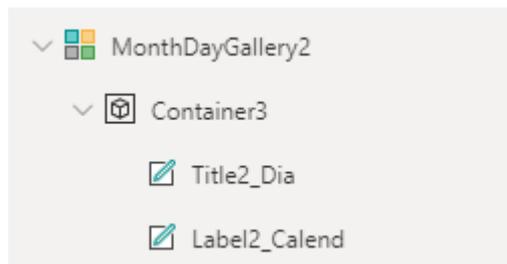


Figura 23: Vista de árbol de la galería "MonthDayGallery2". Fuente: Elaboración propia

Elemento	Atributos	Código
MonthDayGallery2	Items	[0;1;2;3;4;5;6;7;8;9;10;11;12;13;14;15;16;17;18;19;20;21;22;23;24;25;26;27;28;29;30;31;32;33;34;35;36;37;38;39;40;41]
	Border Color	RGBA(137; 196; 244; 1)

Se establecen 42 ítems ya que es el máximo de días que se van a llegar a mostrar.

Elemento	Atributos	Código
Container3	DropShadow	DropShadow.Light
	Visible	!(DateAdd(_firstDayInView; ThisItem.Value; TimeUnit.Days) - Weekday(DateAdd(_firstDayInView; ThisItem.Value; TimeUnit.Days); StartOfWeek.Monday) + 1 > _lastDayOfMonth)
	BorderThickness	If(CountRows(Filter(DiasSeleccionados; Value = Text(DateAdd(_firstDayInView; ThisItem.Value; TimeUnit.Days); "dd/mm/yyyy")))>0;3; 0)

Con el atributo DropShadow se implementa una ligera sombra que separe los ítems visualmente. En el visible se establece que solo sea visible el componente y lo que contiene si se trata de un día comprendido en las semanas que se muestran. Y con el atributo BorderThickness se consigue que si ese ítem (el día) esta seleccionado, el contenedor tenga un borde indicando que esta seleccionado, y si no es así, que no tenga borde.

Elemento	Atributos	Código
Title2_Dia	Text	Day(DateAdd(_firstDayInView; ThisItem.Value; TimeUnit.Days))
	OnSelect	Select(Label2_Calend)

Se muestra en número el día del mes en cada ítem. "ThisItem.Value" corresponde al número del ítem, que se va sumando al primer día que se muestra (_firstDayInView), formando así todo el mes.

Elemento	Atributos	Código
Label2_Calend	OnSelect	<pre>If(CountRows(Filter(DiasSeleccionados; Value = Text(DateAdd(_firstDayInView; ThisItem.Value; TimeUnit.Days); "dd/mm/yyyy"))))>0; Remove(DiasSeleccionados; LookUp(DiasSeleccionados; Value = Text(DateAdd(_firstDayInView; ThisItem.Value; TimeUnit.Days); "dd/mm/yyyy"))); Collect(DiasSeleccionados; Text(DateAdd(_firstDayInView; ThisItem.Value; TimeUnit.Days); "dd/mm/yyyy")));; Set(Diastexto; Concat(Sort(DiasSeleccionados; Value); Value; ", "))</pre>
	Text	<pre>LookUp(Calendario; GUIDEmpleado = Empleado.Empleados And Day(DateAdd(_firstDayInView;ThisItem.Value; TimeUnit.Days)) = Day(Dia) And Month(DateAdd(_firstDayInView;ThisItem.Value; TimeUnit.Days)) = Month(Dia) And Year(DateAdd(_firstDayInView; ThisItem.Value; TimeUnit.Days)) = Year(Dia); Siglas)</pre>
	Fill	<pre>If(Abs(Title2_Dia.Text - ThisItem.Value) > 10; Switch(Self.Text; "FD"; ColorFade(fxColor.ColorFD; 0,5); "F"; ColorFade(fxColor.ColorF; 0,5); "US"; ColorFade(fxColor.ColorUS; 0,5); "CM"; ColorFade(fxColor.ColorCM; 0,5); "CE"; ColorFade(fxColor.ColorCE; 0,5); "OP"; ColorFade(fxColor.ColorOP; 0,5); "BH"; ColorFade(fxColor.ColorBH; 0,5); "H"; ColorFade(fxColor.ColorH; 0,5); "C"; ColorFade(fxColor.ColorC; 0,5); "B"; ColorFade(fxColor.ColorB; 0,5); "MI" ;ColorFade(fxColor.ColorMI; 0,5); "P" ;ColorFade(fxColor.ColorP; 0,5); "LA" ;ColorFade(fxColor.ColorLA; 0,5); "D" ;ColorFade(fxColor.ColorD; 0,5); "PC" ;ColorFade(fxColor.ColorPC; 0,5); "HP"; ColorFade(fxColor.ColorHP; 0,5); RGBA(200; 200; 200; 0,3)); Switch(Self.Text; "FD"; fxColor.ColorFD; "F"; fxColor.ColorF; "US"; fxColor.ColorUS; "CM"; fxColor.ColorCM; "CE"; fxColor.ColorCE; "OP"; fxColor.ColorOP; "BH"; fxColor.ColorBH; "H"; fxColor.ColorH; "C"; fxColor.ColorC; "B"; fxColor.ColorB; "MI"; fxColor.ColorMI; "P"; fxColor.ColorP; "LA"; fxColor.ColorLA; "D"; fxColor.ColorD; "PC"; fxColor.ColorPC; "HP"; fxColor.ColorHP; RGBA(0; 0; 0; 0)))</pre>
	Color	<pre>If(Abs(Title2_Dia.Text - ThisItem.Value) > 10; ColorFade(RGBA(0; 0; 0; 1); 0,3); RGBA(0; 0; 0; 1))</pre>

Mediante el Onselect conseguimos que, al pulsar un día, se añade ese día a una colección si no estaba previamente, y si estaba anteriormente se elimina. Además creamos una variable (Diastexto) en la que concatenamos los días seleccionados a modo de string, para utilizarla posteriormente. Con el atributo Text extraemos y mostramos las siglas que corresponden a cada día del usuario de la aplicación. Y dependiendo de estas siglas, a través del Fill y el Color, pintamos el día de su respectivo color.

Encima de esta galería situamos otra galería del mismo ancho que simplemente servirá para mostrar las etiquetas de los días de la semana:

Elemento	Atributos	Código
WeekdayGallery2	Items	["Lu"; "Ma"; "Mi"; "Ju"; "Vi"; "Sá"; "Do"]
	BorderColor	RGBA(137; 196; 244; 1)

Y encima de estas dos galerías se sitúa una etiqueta de texto que nos dice que mes se está mostrando:

Elemento	Atributos	Código
LabelMesSelec2	Text	Text(_firstDayOfMonth; "mmm yyyy")
	BorderColor	RGBA(137; 196; 244; 1)

Se aprovecha la variable “_firstDayOfMonth” para mostrar el mes y el año de ese primer día del mes seleccionado.

Y por último en el calendario, encontramos dos flechas con las que podemos ir cambiando el mes que se muestra. Se muestra únicamente el contenido de una de los dos iconos ya que la otra flecha es muy similar pero con efecto contrario:

Elemento	Atributos	Código
IconNextMonth2	Icon	Icon.ChevronRight
	OnSelect	<pre>Set(_firstDayOfMonth; DateAdd(_firstDayOfMonth; 1; TimeUnit.Months));; Set(_firstDayInView; DateAdd(_firstDayOfMonth; - (Weekday(_firstDayOfMonth; StartOfWeek.Monday) - 2 + 1); TimeUnit.Days));; Set(_lastDayOfMonth; DateAdd(DateAdd(_firstDayOfMonth; 1; TimeUnit.Months); -1; TimeUnit.Days));;</pre>
	Tooltip	"Ver el mes siguiente"

Al pulsarla se modifican las variables de _firstDayOfMonth, _firstDayInView y _lastDayOfMonth, consiguiendo cambiar el mes mostrado. Y con Tooltip se establece el mensaje que se mostrará si mantenemos el ratón encima del icono.

Formulario

Para el formulario encontramos, en primer lugar, una etiqueta que muestra los días seleccionados para los que se va a realizar la solicitud:

Elemento	Atributos	Código
Label2_Dias2	Text	<code>If(IsEmpty(DiasSeleccionados); "Selecciona los días en el calendario"; Diastexto)</code>
	Color	<code>If(IsEmpty(DiasSeleccionados); RGBA(202; 202; 202; 1); RGBA(0; 0; 0; 1))</code>
	BorderColor	<code>RGBA(137; 196; 244; 1)</code>

Con esto se consigue que, si hay días seleccionados en el calendario, aparezcan en la etiqueta encadenados en forma de texto gracias a la variable “Diastexto”. Y sin embargo, si no se ha seleccionado ningún día aun, que aparezca en la etiqueta el mensaje “Selecciona los días en el calendario” en gris suave.

Luego encontramos un componente que es una lista desplegable donde se seleccionará el tipo de días festivos que se quiere solicitar:

Elemento	Atributos	Código
Dropdown2_Motivo	Items	<code>Choices('Motivos Solicitud')</code>
	ChevronBackground	<code>RGBA(137; 196; 244; 1)</code>
	ChevronHoverBackground	<code>ColorFade(RGBA(137; 196; 244; 1); -20%)</code>
	HoverFill	<code>RGBA(186; 202; 226; 1)</code>
	PressedFill	<code>RGBA(0; 134; 208; 1)</code>
	SelectionFill	<code>RGBA(137; 196; 244; 1)</code>

Mediante el atributo Items hace referencia a una lista de opciones llamada “Motivos Solicitud”, y es esta lista la que establece las posibilidades que aparecen para seleccionar.

También encontramos en el formulario una entrada de texto multilínea que permite escribir particularidades de la solicitud, pudiendo explicar así lo que se crea conveniente.

Finalizando con el formulario, encontramos el botón para enviar la solicitud:

Elemento	Atributos	Código
Button1_SolicitarDias2	OnSelect	<code>Patch(Solicitudes; Defaults(Solicitudes); {Dias:Diastexto; Empleado:LookUp(Empleados; GUIDEmpleado=Empleados); 'Estado (cre05_estado)': 'Estado de solicitud'.Pendiente; Motivo: Dropdown2_Motivo.Selected.Value; Particularidades:TextInput2_Particularidades.Text});; ForAll(DiasSeleccionados As DIA;</code>

		<pre>If(Dropdown2_Motivo.Selected.Value = 'Motivos Solicitud'.Vacaciones; Patch(Calendario; Defaults(Calendario); {Dia: DateValue(DIA.Value); Motivo:'Tipos de dias festivos'.'Vacaciones NO Confirmadas'; Siglas: "HP" ; Empleado:LookUp(Empleados; GUIDEmpleado=Empleados)})); Patch(Empleados; LookUp(Empleados; GUIDEmpleado=Empleados); {Pendientes: LookUp(Empleados; GUIDEmpleado=Empleados; Pendientes)-1; Planificadas: LookUp(Empleados; GUIDEmpleado=Empleados; Planificadas)+1 })); Back(ScreenTransition.Fade)</pre>
	DisplayMode	<pre>If(!IsEmpty(DiasSeleccionados); DisplayMode.Edit; DisplayMode.Disabled)</pre>
	Text	<pre>"Enviar solicitud"</pre>

En este botón se ha implementado que al presionarlo se haga un Patch a la tabla Solicitudes que cree un nuevo registro con todos los datos que se han indicado de la solicitud y algunos más como el estado de la misma o el empleado que la ha realizado. Además, si el motivo de la solicitud es “Vacaciones”, cada día que se haya seleccionado se establece en la tabla Calendario como un día de vacaciones no confirmadas, que posteriormente aparecerá marcado en el calendario. En estos casos también se modifica el número total de días pendientes de vacaciones y de días planeados, aumentándolo o disminuyéndolo en función de la cantidad de días seleccionados. Tras realizar todo lo anterior, se transiciona y se vuelve a la pantalla anterior, dando por finalizada la solicitud.

Es importante mencionar también el atributo DisplayMode, que indica que, si no se ha seleccionado ningún día, este botón aparezca desactivado y no se pueda presionar, con el fin de evitar que se registren solicitudes incompletas.

Por último en esta pantalla, encontramos el icono de volver atrás, que también estará en muchas otras de las páginas siguientes:

Elemento	Atributos	Código
Icon2_Atrás	Icon	Icon.ArrowLeft
	OnSelect	Back(ScreenTransition.Fade)

El cual, al pulsarlo, nos permite volver a la pantalla anterior en la que estábamos.

5.2.3.3 Resultados

Los resultados tanto visuales como funcionales de esta Pantalla 2 son los siguientes. Nada mas acceder a esta pantalla, sin haber seleccionado ningún dato se verá así:

Solicitud de días de vacaciones

Días:

Motivo:

Particularidades:

Lu	Ma	Mi	Ju	Vi	Sá	Do
26	27	28 MI	29	30	31	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16 H	17 H	18	19	20 BH	21	22
23	24	25	26	27	28	29
30 HP	1	2	3 HP	4 HP	5	6

Días de vacaciones pendientes: 15

Figura 24: Resultado 1 de la pantalla 2. Fuente: Elaboración propia

Y, sin embargo, una vez indicados los datos de la solicitud aparecerá tal que así:

Solicitud de días de vacaciones

Días:

Motivo:

Particularidades:

Lu	Ma	Mi	Ju	Vi	Sá	Do
26	27	28 MI	29	30	31	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16 H	17 H	18	19	20 BH	21	22
23	24	25	26	27	28	29
30 HP	1	2	3 HP	4 HP	5	6

Días de vacaciones pendientes: 15

Figura 25: Resultado 2 de la pantalla 2. Fuente: Elaboración propia

Aquí como podemos observar, los días seleccionados aparecen tanto en la etiqueta de texto como en el calendario marcados. Además, el botón ya está disponible para presionar, ya que el formulario está debidamente complementado.

5.2.4 Pantalla 3 – Mis solicitudes

5.2.4.1 Análisis funcional

Esta es la pantalla a la que se va a acceder al pulsar el botón “Mis solicitudes” de la Pantalla 1. Es una pantalla sencilla que va debería contener las siguientes funciones:

- El usuario debe poder visualizar todas las solicitudes que ha realizado anteriormente, pudiendo observar todos los datos las mismas. Con la intención principal de saber cual es el estado de estas solicitudes, es decir, si han sido aceptadas/rechazadas o aún están pendientes.
- Además sería conveniente poder filtrar estos registros, por ejemplo con un buscador para cada campo que permita mostrar únicamente los resultados que coinciden con la búsqueda.
- Se ha puede considerar buena opción que en esta misma pestaña exista la posibilidad de acceder a la Pantalla 2 para crear una nueva solicitud. Así que debería haber un botón que te permita esto.
- Como en tantas otras pantallas, debe haber también la posibilidad de volver a la pantalla anterior

Considerando todos estos requisitos se ha diseñado el siguiente boceto de la pantalla:

Pantalla 3



← Solicitudes realizadas

Nueva Solicitud

Fecha	Días	Estado	Motivo	Particularidades

Figura 26: Boceto previo de la pantalla 3. Fuente: Elaboración propia

Donde para mostrar las solicitudes se podría utilizar una galería que permitiera desplazamiento vertical y poder mostrar así todas las solicitudes existentes del usuario indicado.

5.2.4.2 Desarrollo

Esta pantalla consiste principalmente en un contenedor grande que posee dos elementos. El primero de ellos es un contenedor en la parte superior donde encontramos toda la cabecera de la tabla, tanto las etiquetas de los nombres de las columnas como las entradas de texto que funcionan como buscadores.

Estos componentes son simples, pero han de estar alineados y referenciados entre si correctamente para obtener un formato y diseño adecuado en la tabla.

Entradas de texto

Elemento	Atributos	Código
TextInput3_Dias	X	<code>TextInput3_Estado.X + TextInput3_Estado.Width + 10</code>
	Y	<code>TextInput3_Estado.Y</code>
	BorderColor	<code>RGBA(137; 196; 244; 1)</code>

Etiquetas de los nombres de las columnas

Elemento	Atributos	Código
Label3_Dias	Text	<code>"Días"</code>
	Width	<code>TextInput3_Dias.Width</code>
	X	<code>TextInput3_Dias.X</code>
	Y	<code>Label3_Estado.Y</code>

Estos son los ejemplos de etiqueta y entrada de texto del campo Días, pero en la tabla encontramos otras cinco columnas que son: Estado, Motivo, Particularidades, Fecha envío y Gestionada por.

Por otro lado, el segundo elemento que posee el contenedor principal es una galería horizontal. En la cual vamos a encontrar todos los registros de solicitudes que ha realizado el usuario de la aplicación.

Galería

Elemento	Atributos	Código
Gallery3_Solicitudes	Items	<pre>Filter(Solicitudes; Empleado.Empleados = GUIDEmpleado And (IsBlank(Trim(TextInput3_Estado.Text)) Or Trim(TextInput3_Estado.Text) in 'Estado (cre05_estado)') And (IsBlank(Trim(TextInput3_Dias.Text)) Or Trim(TextInput3_Dias.Text) in Text(Dias)) And (IsBlank(Trim(TextInput3_Motivo.Text)) Or Trim(TextInput3_Motivo.Text) in Motivo) And (IsBlank(Trim(TextInput3_Particularidades.Text)) Or Trim(TextInput3_Particularidades.Text) in Particularidades) And (IsBlank(Trim(TextInput3_Fecha.Text)) Or Trim(TextInput3_Fecha.Text) in 'Fecha de creación') And (IsBlank(Trim(TextInput3_GestionadaPor.Text)) Or Trim(TextInput3_GestionadaPor.Text) in 'Gestionada por'))</pre>

En ella, a través del atributo ítems, se establecen los filtros que deciden que registros se muestran. En este caso se filtran las solicitudes que hayan sido enviadas por el usuario que está utilizando la aplicación, y además se filtran las solicitudes por los textos que hayan sido introducidos en los buscadores (en caso de que se haya introducido algo).

Posteriormente, dentro de la galería y para cada uno de los registros que se muestren, encontramos 6 etiquetas de texto que corresponden con las 6 columnas de la tabla. De nuevo, este es el ejemplo de la etiqueta de texto de la columna Días:

Elemento	Atributos	Código
Label3_Dias2	Text	<code>ThisItem.Dias</code>
	Width	<code>TextInput3_Dias.Width</code>
	X	<code>TextInput3_Dias.X</code>
	Overflow	<code>Overflow.Scroll</code>

Por último, además del botón de “Atrás” que ya se detallado en el desarrollo de otras Pantallas, encontramos el botón de Nueva Solicitud:

Elemento	Atributos	Código
Button3_NuevaSolicitud	OnSelect	<code>Navigate(Pantalla2; ScreenTransition.Fade)</code>
	Text	<code>"Nueva solicitud"</code>

El cual mantiene la combinación de colores de la mayoría de los botones de la aplicación y en este caso nos vuelve a dirigir a la Pantalla 2, permitiéndonos crear una nueva solicitud.

5.2.4.3 Resultados

Con todo este desarrollo implementado, la pantalla 3 ha resultado así:



Figura 27: Resultado 1 de la pantalla 3. Fuente: Elaboración propia

Donde, como ya se ha mencionado, aparecen solo las solicitudes que ha realizado el usuario de la aplicación. Y al utilizar uno, o más, de los buscadores la aplicación filtra los registros que contengan el texto introducido en el campo deseado:



The screenshot shows a mobile application interface titled "Mis Solicitudes". At the top right, there is a button labeled "Nueva solicitud". Below this is a table with the following columns: Estado, Días, Motivo, Particularidades, Fecha envío, and Gestionada por. The "Motivo" column is currently filtered to show only "Vacaciones".

Estado	Días	Motivo	Particularidades	Fecha envío	Gestionada por
Pendiente	12/06/2024, 13/06/2024	Vacaciones	Considero que son buenos días para un	10/06/2024 14:36	
Aceptada	06/08/2024, 07/08/2024, 08/08/2024	Vacaciones		13/06/2024 11:01	Alberto García Lopez
Pendiente	08/07/2024, 09/07/2024, 10/07/2024	Vacaciones		14/06/2024 11:03	

Figura 28: Resultado 2 de la pantalla 3. Fuente: Elaboración propia

Como en este ejemplo, que se pretende que solo aparezcan las solicitudes que tengan “Vacaciones” como motivo.

5.2.5 Pantalla 4 – Consultar calendario

5.2.5.1 Análisis funcional

A esta pantalla se va a poder acceder tanto desde la pantalla 1, al pulsar el botón “Ver calendario”, como desde la pantalla 5. Y está destinada a que el usuario pueda consultar su calendario laboral, con todas las indicaciones de todos los tipos de días y ausencias que tiene planeadas. Sin embargo, es una pantalla con muy poca interacción, únicamente la de poder navegar por los meses del calendario, ya que esta pantalla pretende simplemente mostrar el calendario del usuario de manera clara y visual. Por tanto, debería incluir las siguientes funciones:

- El elemento principal debe ser el calendario, se pueden utilizar los mismos componentes que se utilizaron para desarrollar el calendario en la pantalla 2, pero en este caso no es necesario que se puedan seleccionar días, simplemente consultarlo.
- Acompañando a este calendario es completamente necesario que haya una leyenda que indique que significan los colores y las siglas que encontramos en el calendario. Se puede utilizar la misma leyenda que en el sistema antiguo, con el fin de evitar posibles confusiones de los trabajadores que ya estaban acostumbrados a las anteriores siglas y colores.

	Height	<i>Rectangle4_Grande.Height/16</i>
	Width	<i>Rectangle4_Grande.Width</i>

Y en el interior de estos rectángulos encontramos, por un lado, las etiquetas que contienen siglas y color de cada tipo de día:

Elemento	Atributos	Código
Label3_1	Text	"F"
	Fill	<i>fxColor.ColorF</i>
	X	<i>Rectangle4_Grande.X</i>
	Y	<i>Rectangle4_2.Y</i>
	Height	<i>Rectangle4_Grande.Height/16</i>
	Width	<i>Rectangle4_Vertical.Width</i>

Y por otro, las etiquetas que contienen la descripción de ese tipo de día:

Elemento	Atributos	Código
Label4_1	Text	"Formación"
	X	<i>Rectangle4_Grande.X + Rectangle4_Vertical.Width</i>
	Y	<i>Rectangle4_2.Y</i>
	Height	<i>Rectangle4_Grande.Height/16</i>
	Width	<i>Rectangle4_Grande.Width - Rectangle4_Vertical.Width</i>

Finalmente, encontramos el icono para cerrar esta pantalla. El cual tiene la misma funcionalidad que los iconos de "Atrás" en las anteriores pantallas, pero en este caso, el icono es el de una equis.

5.2.5.3 Resultados

Una vez implementado todo la anterior, la pantalla resulta de la siguiente manera:

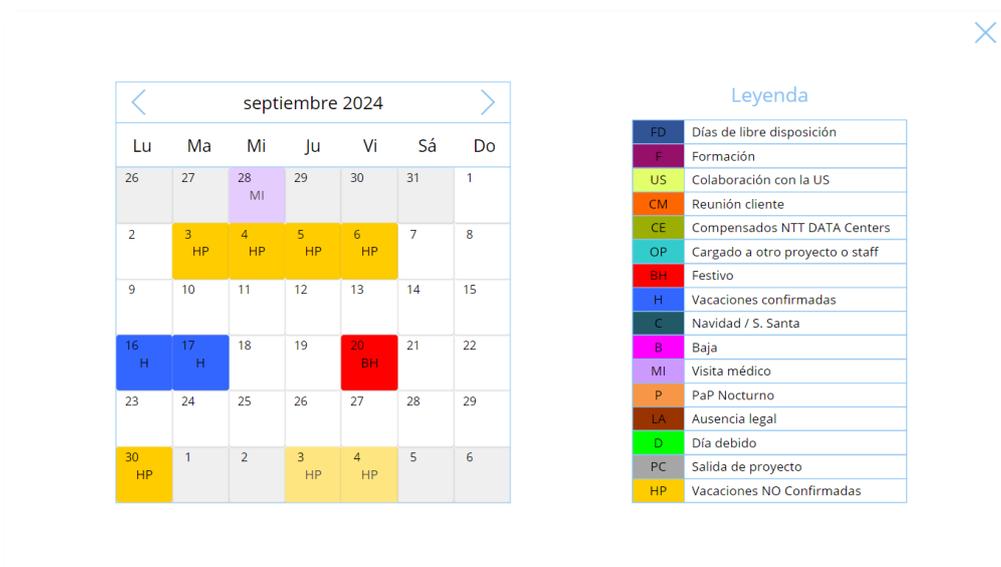


Figura 30: Resultado de la pantalla 4. Fuente: Elaboración propia

Evidentemente para esta muestra de resultados se utiliza un calendario de ejemplo, pero este se mostraría completamente personalizado y adecuado al calendario laboral de cada trabajador.

5.2.6 Pantalla 5 – Modo mánager / administrador

5.2.6.1 Análisis funcional

Esta pantalla 5 es a la que accederemos al pulsar el botón “Modo Mánager/Administrador” que encontramos en la pantalla. Por eso mismo a esta pantalla solo se va a poder acceder si el rol del usuario es Mánager o Administrador.

Las principales intenciones y funcionalidades de esta pantalla son:

- Si el usuario tiene el rol de administrador debe poder observar los datos de todos los trabajadores, tanto los datos fijos como el nombre, el número de empleado o el correo, como los datos variables como los días de vacaciones pendientes, planificados o confirmados.
- Sería conveniente que se pudiera visualizar el calendario laboral de cada uno de estos trabajadores
- Además, debido a la gran cantidad de datos que se van a poder visualizar es necesario permitir filtrar los registros según los campos que se deseen.
- En el caso de que el usuario posea el rol Mánager, este debería poder visualizar toda esa misma información mencionada anteriormente, pero en este caso solo podrá hacerlo de los trabajadores que se encuentren en el proyecto en el que el usuario sea mánager, es decir, solo los trabajadores que le tengan como mánager.
- También debe incluir un botón que le permita al mánager gestionar las solicitudes que sus trabajadores han enviado, es decir, poder aceptarlas o rechazarlas.
- Finalmente, esta pantalla también debe incluir otro botón, disponible solo para administradores, que permita acceder a una pantalla donde se podrán modificar

los festivos manualmente, tanto de una persona individual como de un grupo de trabajadores.

Por todo ello, este sería un posible boceto de la pantalla que se va a desarrollar:

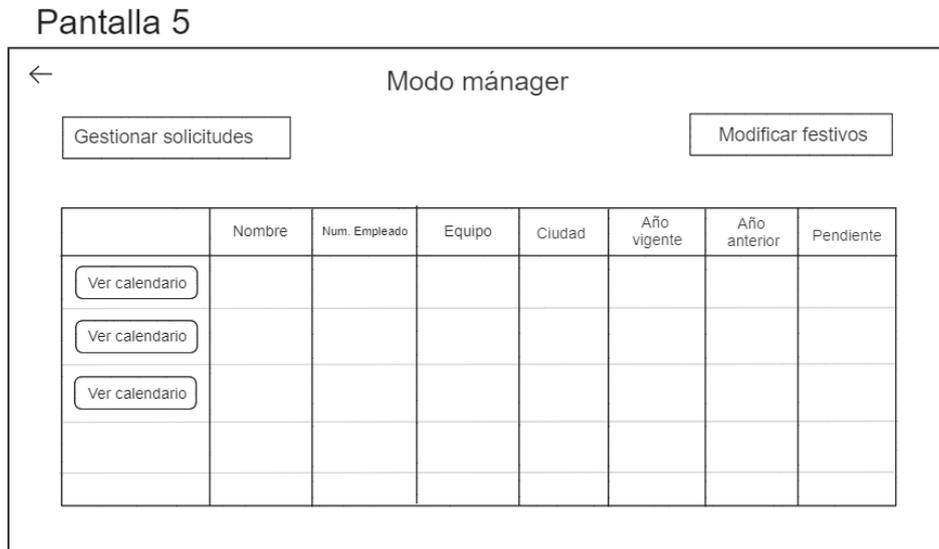


Figura 31: Boceto previo de la pantalla 5. Fuente: Elaboración propia

5.2.6.2 Desarrollo

En primer lugar, el título de esta pantalla variará dependiendo del rol que posea el usuario. Si el rol es el de Mánager el título de la pantalla será “Modo mánager” y si el rol que posee el usuario es Administrador, el título será Modo administrador:

Elemento	Atributos	Código
Label5_Titulo	Text	<code>Concatenate("Modo "; Lower(LookUp(Roles; Empleado.'Empleados (cre05_empleadosid)' = GUIDEmpleado; 'Rol (cre05_rol)'))</code>
	X	<code>Parent.Width/2 - Self.Width/2</code>

En segundo lugar, para poder mostrar todos los registros de los empleados se va a utilizar una galería muy similar a la utilizada en la pantalla 3. En la cual encontraremos también las etiquetas de texto de los nombres de las columnas, las entradas de texto que servirán como buscadores y los ítems de texto de la galería que será los encargados mostrar los datos de cada columna.

Sin embargo, en este caso la galería mostrará los registros de la tabla “Empleados” en vez de la tabla “Solicitudes” ya que lo que se desea es visualizar la información de los empleados con sus respectivos campos de datos.

Elemento	Atributos	Código
Gallery5_Empleados	Items	<pre> Filter(Empleados; If(RolEmpleado = 'Roles Empleados'.Manager; 'Manager 1'. 'Empleados (cre05_empleadosid)' = GUIDEmpleado Or 'Manager 2'. 'Empleados (cre05_empleadosid)' = GUIDEmpleado Or 'Manager 3'. 'Empleados (cre05_empleadosid)' = GUIDEmpleado; !IsBlank(Empleados)) And (IsBlank(Trim(TextInput5_NumEmple.Text)) Or Trim(TextInput5_NumEmple.Text) in 'Numero de empleado') And (IsBlank(Trim(TextInput5_Nombre.Text)) Or Trim(TextInput5_Nombre.Text) in Concatenate(Nombre; " "; '1er Apellido'; " "; '2do Apellido')) And (IsBlank(Trim(TextInput5_Correo.Text)) Or Trim(TextInput5_Correo.Text) in 'Correo electronico') And (IsBlank(Trim(TextInput5_Equipo.Text)) Or Trim(TextInput5_Equipo.Text) in Equipo) And (IsBlank(Trim(TextInput5_Proyecto.Text)) Or Trim(TextInput5_Proyecto.Text) in Proyecto) And (IsBlank(Trim(TextInput5_Vigente.Text)) Or Trim(TextInput5_Vigente.Text) in 'Año vigente') And (IsBlank(Trim(TextInput5_Anterior.Text)) Or Trim(TextInput5_Anterior.Text) in 'Año anterior') And (IsBlank(Trim(TextInput5_Planificadas.Text)) Or Trim(TextInput5_Planificadas.Text) in Planificadas) And (IsBlank(Trim(TextInput5_Confirmadas.Text)) Or Trim(TextInput5_Confirmadas.Text) in Confirmadas) And (IsBlank(Trim(TextInput5_Pendientes.Text)) Or Trim(TextInput5_Pendientes.Text) in Pendientes) And (IsBlank(Trim(TextInput5_Activo.Text)) Or Trim(TextInput5_Activo.Text) in Activo) And (IsBlank(Trim(TextInput5_CAR.Text)) Or Trim(TextInput5_CAR.Text) in CAR) </pre>

En ella, a través del atributo Items, se establece que, si el usuario tiene el rol de Mánager, se filtren y se muestren solo los empleados que tienen al usuario como mánager. Y si el usuario no tiene el rol mánager, es decir, tiene el rol de administrador, se muestren todos los registros de la tabla empleados. Además, se establecen también todos los filtros de los buscadores de cada campo. Y en el caso de esta galería, debido a la gran cantidad de columnas que deseamos que tenga, es necesario activar el desbordamiento horizontal en el contenedor que la engloba.

Elemento	Atributos	Código
Container5_1	LayoutOverflowX	LayoutOverflow.Scroll

Por otro lado, y como un campo más en la galería, se ha implementado un botón que aparecerá en cada registro y que permitirá visualizar el calendario laboral del empleado de ese registro:

Elemento	Atributos	Código
Button5_Calendario	OnSelect	<code>Navigate(Pantalla4; ScreenTransition.Fade; {GUIDEmpleadoCalendario: ThisItem.'Empleados (cre05_empleadosid)'})</code>
	Text	<code>"Ver calendario"</code>

Al pulsarlo se nos dirige a la pantalla 4, que muestra el calendario con la leyenda de manera clara. Pero en este caso se establece en la variable "GUIDEmpleadoCalendario" el ID del registro concreto en el que se ha pulsado el botón. Esta variable es la que se utiliza posteriormente en la pantalla 4 para saber de qué empleado mostrar la información en el calendario. Consiguiendo así utilizar la pantalla 4 para cumplir con esta funcionalidad de la pantalla 5.

Y por último en esta pantalla, además del icono para volver atrás, encontramos dos botones. El primero de ellos lo encontramos en la parte superior izquierda, encima de la galería, y es el que le permitirá al usuario acceder a la pantalla donde se gestionan las solicitudes de los trabajadores:

Elemento	Atributos	Código
Button5_GestSolic	OnSelect	<code>Navigate(Pantalla6; ScreenTransition.Fade)</code>
	Text	<code>"Gestionar solicitudes"</code>

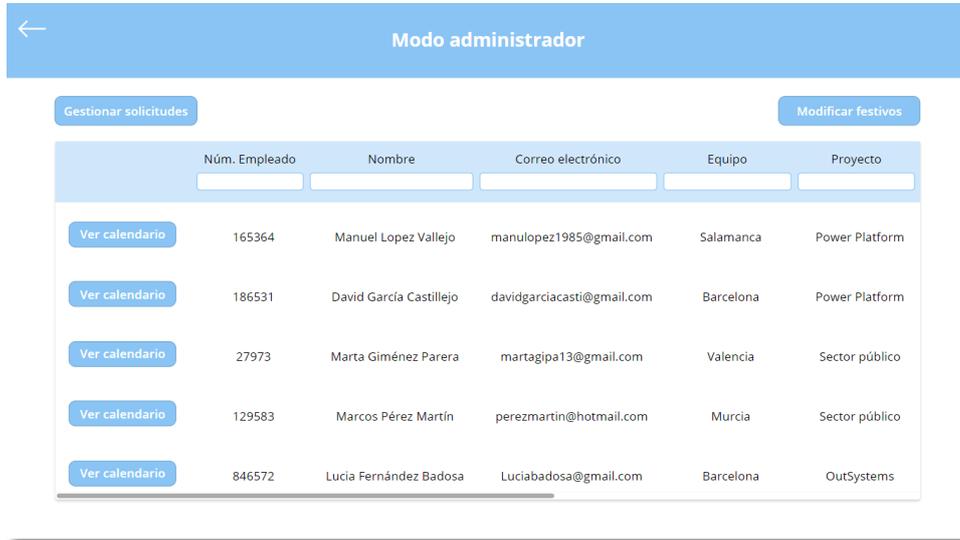
Y el segundo de ellos, que lo encontramos en el lado opuesto, es el que le permitirá al usuario acceder a la pantalla donde se pueden añadir y modificar festivos:

Elemento	Atributos	Código
Button5_ModifFest	OnSelect	<code>Navigate(Pantalla8; ScreenTransition.Fade)</code>
	Text	<code>"Modificar festivos"</code>
	Visible	<code>If(RolEmpleado = 'Roles Empleados'.Administrador ; true; false)</code>

Pero, como se puede observar en el atributo Visible, este botón tiene la particularidad de que solo va a ser visible, y por tanto pulsable, si el rol que posee el usuario es el de administrador. Ya que la pantalla a la que se accede permite funcionalidades específicas del rol administrador.

5.2.6.3 Resultados

Una vez implementado todo lo anterior, la pantalla 5 se vería así en el caso de que el usuario tenga el rol de administrador:



Modo administrador					
Gestionar solicitudes		Modificar festivos			
	Núm. Empleado	Nombre	Correo electrónico	Equipo	Proyecto
Ver calendario	165364	Manuel Lopez Vallejo	manulopez1985@gmail.com	Salamanca	Power Platform
Ver calendario	186531	David García Castillejo	davidgarciacasti@gmail.com	Barcelona	Power Platform
Ver calendario	27973	Marta Giménez Parera	martagipa13@gmail.com	Valencia	Sector público
Ver calendario	129583	Marcos Pérez Martín	perezmartin@hotmail.com	Murcia	Sector público
Ver calendario	846572	Lucía Fernández Badosa	Luciabadosa@gmail.com	Barcelona	OutSystems

Figura 32: Resultado 1 de la pantalla 5. Fuente: Elaboración propia

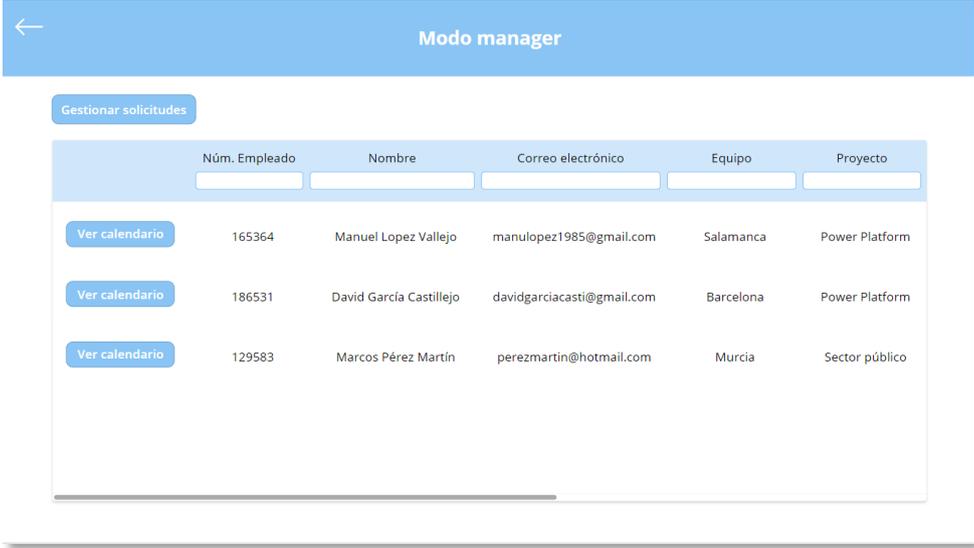


Modo administrador								
Gestionar solicitudes		Modificar festivos						
Equipo	Proyecto	Año vigente	Año anterior	Planificadas	Confirmadas	Pendientes	Activo	CAR
Salamanca	Power Platform	23 días	5 días	3 días	7 días	18 días	Sí	No
Barcelona	Power Platform	18 días	0 días	2 días	4 días	12 días	Sí	Sí
Valencia	Sector público	23 días	10 días	4 días	8 días	21 días	Sí	Sí
Murcia	Sector público	23 días	0 días	0 días	11 días	12 días	Sí	No
Barcelona	OutSystems	23 días	2 días	8 días	0 días	17 días	Sí	No

Figura 33: Resultado 2 de la pantalla 5. Fuente: Elaboración propia

Como se puede observar, aparece una gran cantidad de registros y en la galería se puede desplazar tanto verticalmente, como horizontalmente para poder visualizar todas las columnas existentes.

Sin embargo, si el usuario posee el rol de mánager la pantalla se vería así:



← **Modo manager**

Gestionar solicitudes

	Núm. Empleado	Nombre	Correo electrónico	Equipo	Proyecto
Ver calendario	165364	Manuel Lopez Vallejo	manulopez1985@gmail.com	Salamanca	Power Platform
Ver calendario	186531	David García Castillejo	davidgarciacasti@gmail.com	Barcelona	Power Platform
Ver calendario	129583	Marcos Pérez Martín	perezmartin@hotmail.com	Murcia	Sector público

Figura 34: Resultado 3 de la pantalla 5. Fuente: Elaboración propia

Donde podemos percibir que el título de la pantalla ha cambiado, que el botón de “Modificar festivos” no aparece, y que aparecen menos registros, ya que solo aparecen los de los trabajadores de los que el usuario es mánager.

Una vez más los datos que se muestran son datos de ejemplo, y no corresponden con la información real de ningún trabajador.

5.2.7 Pantalla 6 – Solicitudes

5.2.7.1 Análisis funcional

Como ya sabemos, esta es la pantalla a la que se va a acceder en caso de que el rol del usuario sea mánager o administrador y de que pulse el botón “Gestionar solicitudes” de la pantalla 5.

- En esta pantalla el usuario deberá poder visualizar las solicitudes que sus trabajadores hayan realizado, es decir, solo podrá ver las solicitudes de los empleados de los cuales él es el mánager.
- Puede ser conveniente que solo se visualicen las peticiones que están pendientes aún, para evitar que se acumulen las solicitudes con el tiempo y se muestren muchas que ya han sido aceptadas/rechazadas.
- De estas solicitudes se deberán mostrar todos los datos relacionados, tales como los días que se han seleccionado, el motivo, las particularidades que se han añadido, la fecha de envío de la solicitud, el empleado al que pertenece la solicitud, etc.
- Además, el usuario debe tener la posibilidad de poder seleccionar una solicitud concreta para posteriormente poder elegir si aceptarla o no.

Teniendo en cuenta todo lo anterior, este sería el boceto de la pantalla:

Pantalla 6

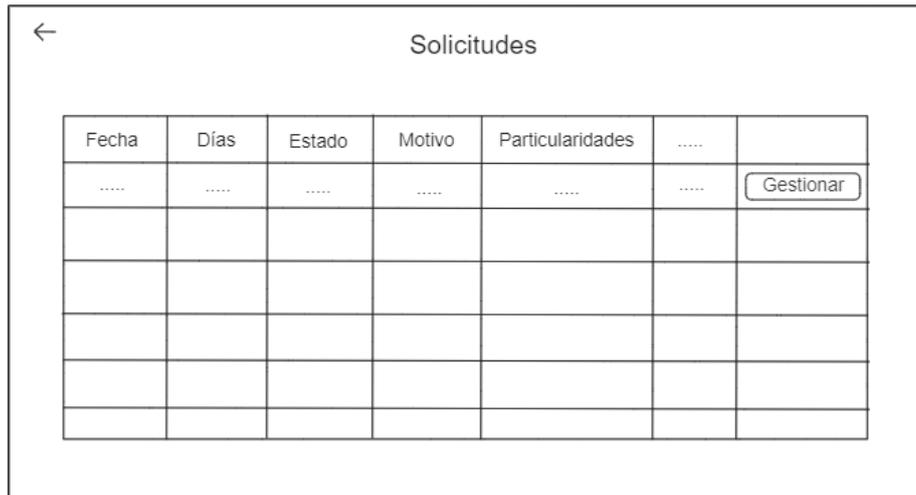


Figura 35: Boceto previo de la pantalla 6. Fuente: Elaboración propia

En principio esta puede ser una pantalla sencilla ya que se ha considerado que el proceso de aceptación o rechazo de las solicitudes se efectúe en una nueva pantalla que aparecerá al presionar el botón gestionar de una de las solicitudes.

5.2.7.2 Desarrollo

De nuevo para mostrar la información en esta pantalla se utilizará una galería. Y en este caso más similar aún a la de la pantalla 3 ya que esta galería también va a estar conectada con la tabla “Solicitudes” de la base de datos.

Por tanto, utilizamos los componentes de las anteriores galerías para desarrollar esta también, manteniendo las entradas de texto que funcionan como buscadores para que se puedan filtrar los registros según se desean, lo cual siempre es algo conveniente para la satisfacción del usuario.

Elemento	Atributos	Código
Gallery5_Empleados	Items	<pre> Filter(Solicitudes; (Empleado.'Manager 1'. 'Empleados (cre05_empleadosid)' = GUIDEmpleado Or Empleado.'Manager 2'. 'Empleados (cre05_empleadosid)' = GUIDEmpleado Or Empleado.'Manager 3'. 'Empleados (cre05_empleadosid)' = GUIDEmpleado) And ('Estado (cre05_estado)' = 'Estado de solicitud'.Pendiente) And (IsBlank(Trim(TextInput6_Estado.Text)) Or Trim(TextInput6_Estado.Text) in 'Estado (cre05_estado)') And (IsBlank(Trim(TextInput6_Empleado.Text)) Or Trim(TextInput6_Empleado.Text) in Concatenate(Empleado.Nombre; " "; Empleado.'1er Apellido'; " "; Empleado.'2do Apellido')) And </pre>

		<pre>(IsBlank(Trim(TextInput6_Dias.Text)) Or Trim(TextInput6_Dias.Text) in Dias) And (IsBlank(Trim(TextInput6_Motivo.Text)) Or Trim(TextInput6_Motivo.Text) in Motivo) And (IsBlank(Trim(TextInput6_Particularidades.Text)) Or Trim(TextInput6_Particularidades.Text) in Particularidades) And (IsBlank(Trim(TextInput6_FechaEnvio.Text)) Or Trim(TextInput6_FechaEnvio.Text) in 'Fecha de creación') And (IsBlank(Trim(TextInput6_Equipo.Text)) Or Trim(TextInput6_Equipo.Text) in Empleado.Equipo) And (IsBlank(Trim(TextInput6_Proyecto.Text)) Or Trim(TextInput6_Proyecto.Text) in Empleado.Proyecto))</pre>
--	--	--

Sin embargo, en esta galería no se diferencia entre si el rol del usuario es mánager o administrador, ya que en los dos casos se filtran y se muestran solo las solicitudes de los trabajadores que tengan al propio usuario como mánager.

Además, a través del atributo ítems de la galería también se filtran para que se muestren solo las solicitudes que estén en estado = pendiente. Por supuesto, también se realizan los filtros correspondientes a los buscadores de cada uno de los campos.

En el caso de esta galería, el primer elemento que parecerá en todos los registros será el botón de “Gestionar”:

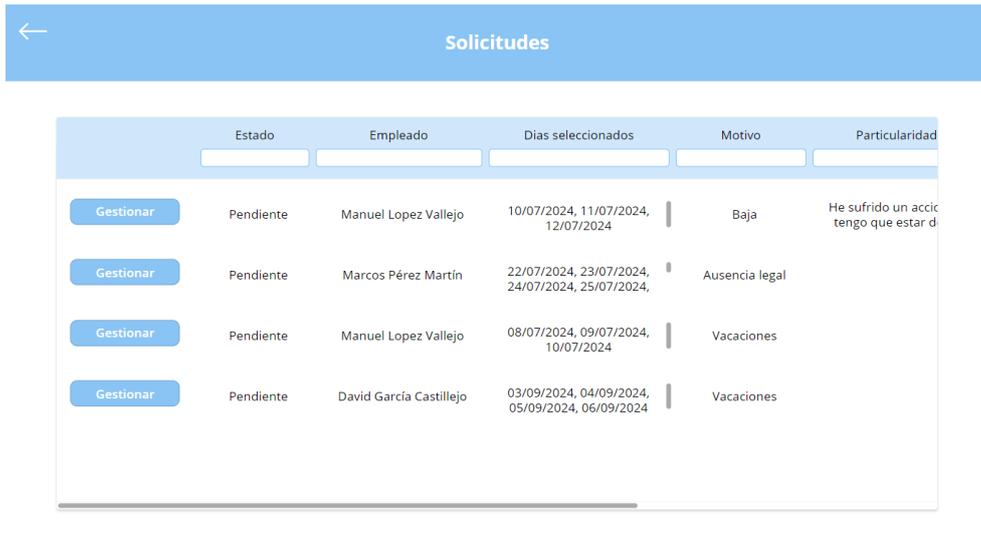
Elemento	Atributos	Código
Button6_Gestionar	OnSelect	<pre>Set(Pantalla7_visible; true);; Set(SolicitudSeleccionada; ThisItem)</pre>
	Text	<pre>"Gestionar"</pre>

Con este botón accederemos a la pantalla 7 donde el usuario podrá consultar y aceptar/rechazar una solicitud concreta. Pero en este caso no utilizamos el *Navigate* que se ha utilizado anteriormente ya que la pantalla 7 va a ser algo diferente a las anteriores. En este caso se establece en true la variable Pantalla7_Visible y en la variable SolicitudSeleccionada se guarda el registro que se ha seleccionado con el botón.

Por otro lado, los campos que se muestran en esta galería son: Estado de la solicitud, empleado que la ha realizado, los días que se han seleccionado, el motivo, las particularidades, la fecha en la que se envió y el equipo y el proyecto al que pertenece el empleado.

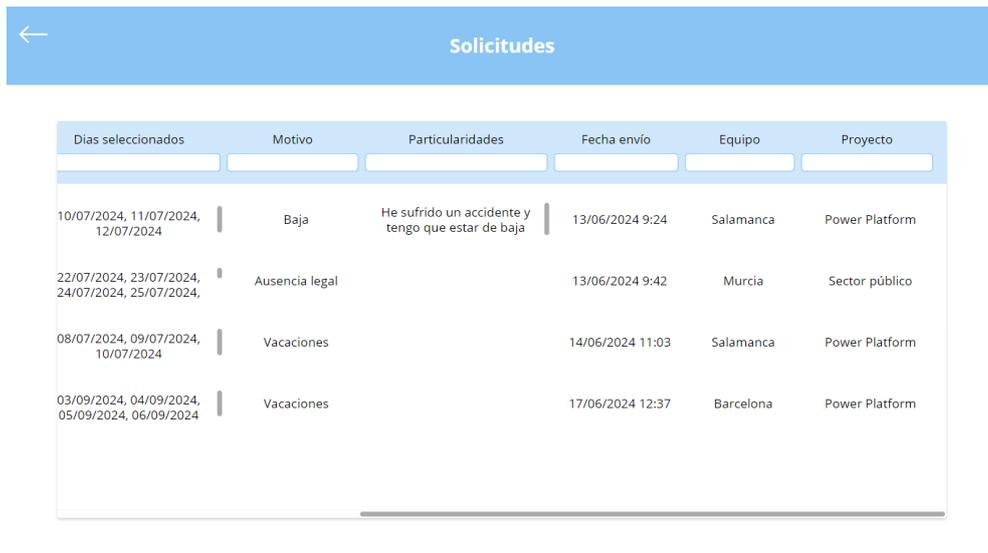
5.2.7.3 Resultados

Esta pantalla 6 resultaría así tras todo lo anterior implementado:



	Estado	Empleado	Dias seleccionados	Motivo	Particularidad
Gestionar	Pendiente	Manuel Lopez Vallejo	10/07/2024, 11/07/2024, 12/07/2024	Baja	He sufrido un accidente y tengo que estar d
Gestionar	Pendiente	Marcos Pérez Martín	22/07/2024, 23/07/2024, 24/07/2024, 25/07/2024,	Ausencia legal	
Gestionar	Pendiente	Manuel Lopez Vallejo	08/07/2024, 09/07/2024, 10/07/2024	Vacaciones	
Gestionar	Pendiente	David García Castillejo	03/09/2024, 04/09/2024, 05/09/2024, 06/09/2024	Vacaciones	

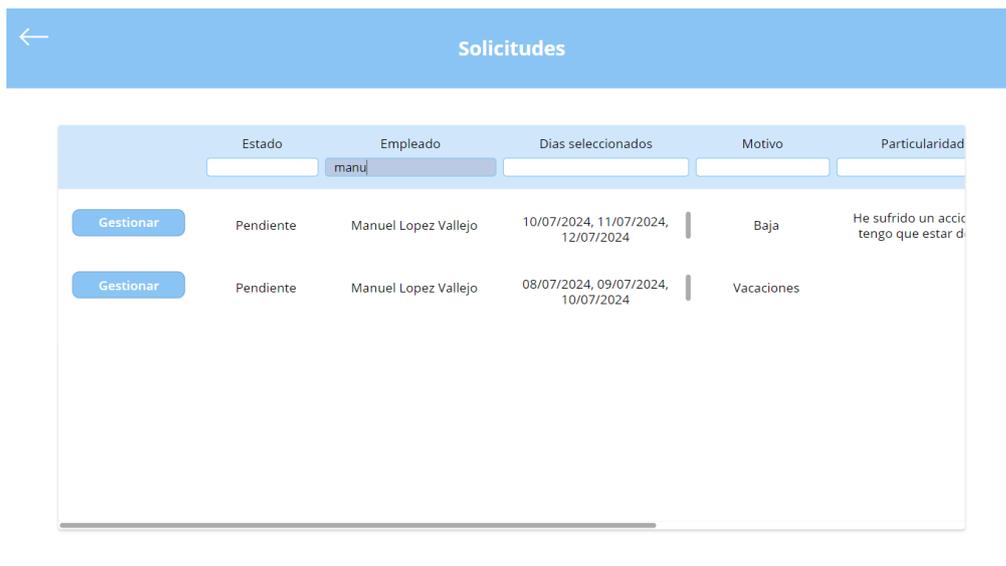
Figura 36: Resultado 1 de la pantalla 6. Fuente: Elaboración propia



Dias seleccionados	Motivo	Particularidades	Fecha envío	Equipo	Proyecto
10/07/2024, 11/07/2024, 12/07/2024	Baja	He sufrido un accidente y tengo que estar de baja	13/06/2024 9:24	Salamanca	Power Platform
22/07/2024, 23/07/2024, 24/07/2024, 25/07/2024,	Ausencia legal		13/06/2024 9:42	Murcia	Sector público
08/07/2024, 09/07/2024, 10/07/2024	Vacaciones		14/06/2024 11:03	Salamanca	Power Platform
03/09/2024, 04/09/2024, 05/09/2024, 06/09/2024	Vacaciones		17/06/2024 12:37	Barcelona	Power Platform

Figura 37: Resultado 2 de la pantalla 6. Fuente: Elaboración propia

Donde, efectivamente, esas solicitudes pertenecen a empleados que tienen como Mánager al usuario que está utilizando de la aplicación. Y donde, una vez más, funciona a la perfección el filtro dinámico que ofrecen los buscadores:



	Estado	Empleado	Dias seleccionados	Motivo	Particularidad
Gestionar	Pendiente	Manuel Lopez Vallejo	10/07/2024, 11/07/2024, 12/07/2024	Baja	He sufrido un accidente tengo que estar d
Gestionar	Pendiente	Manuel Lopez Vallejo	08/07/2024, 09/07/2024, 10/07/2024	Vacaciones	

Figura 38: Resultado 3 de la pantalla 6. Fuente: Elaboración propia

5.2.8 Pantalla 7 – Gestión de las solicitudes

5.2.8.1 Análisis funcional

El comportamiento de esta pantalla va a ser algo diferente al de las anteriores. En este caso, la pantalla 7 no será una nueva pantalla aparte, funcionará como una pestaña que aparecerá a modo de ‘pop-up’ encima de la pantalla 6 al pulsar el botón “Gestionar” de una de las solicitudes. La idea es que esta pestaña no ocupe toda la pantalla 6 y que además se oscurezca el fondo cuando la pantalla 7 aparece, brindando así mucho más sentido visual y funcional al hecho de abrir una solicitud para aceptarla o rechazarla. Los elementos que debería contener esta nueva pestaña son los siguientes:

- Se debe mostrar toda la información de la solicitud que se ha seleccionado. Al menos, los campos de: Empleado que ha realizado la solicitud, días seleccionados, motivo de la solicitud y particularidades.
- El mánager que va a gestionar la solicitud debe poder consultar también en esta pantalla el calendario laboral de la persona que ha realizado la solicitud, para poder evaluar mejor la decisión final sobre esa solicitud.
- Si es posible, sería conveniente que en el calendario apareciera marcados también los días que se han seleccionado en la solicitud, facilitando visualmente a la interpretación de la misma.
- Por otro lado, en la pestaña debemos encontrar dos botones, el de aceptar y el de rechazar. Lo cuales van a permitir al mánager cambiar el estado de la solicitud y añadirán (o no) los días seleccionados al calendario del empleado con el motivo propuesto.
- Evidentemente la pestaña también deberá incluir un icono de equis que permita cerrar la pestaña sin necesidad de aceptar o rechazar la solicitud.

Sabiendo todo esto, este sería el posible boceto de la pantalla 7 que, como se ha mencionado, pretende ser una ventana emergente que se muestra sobre la pantalla 6.

Pantalla 7

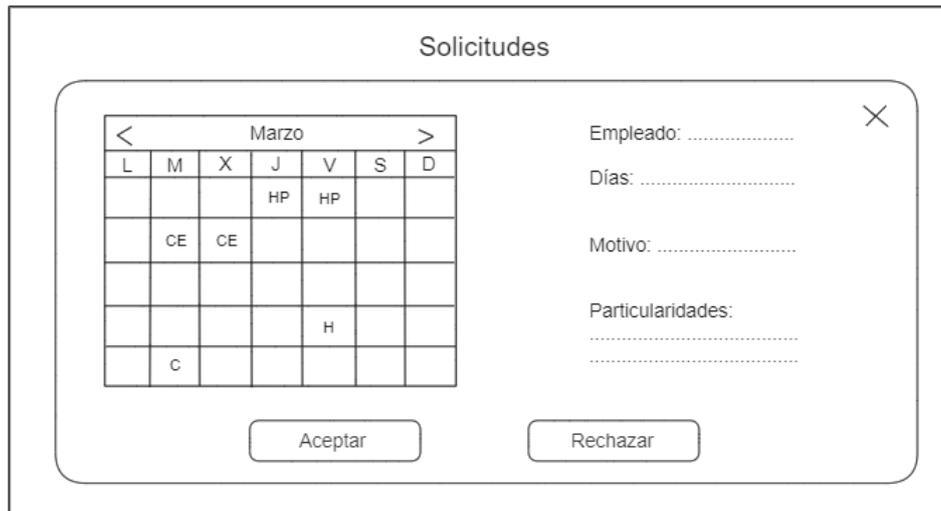


Figura 39: Boceto previo de la pantalla 7. Fuente: Elaboración propia

5.2.8.2 Desarrollo

En la pantalla anterior ya se implementó que, al pulsar el botón de “Gestionar” de alguna solicitud, la variable “Pantalla7_Visible” se estableciera en True. Esto permite que toda esta pantalla 7 se encuentre dentro de un contenedor, que va a ser visible únicamente cuando la variable “Pantalla7_Visible” sea True:

Elemento	Atributos	Código
Container_Pantalla7	Visible	<code>Pantalla7_Visible</code>
	DropShadow	<code>DropShadow.Bold</code>

Esto hace que el atributo Visible del contenedor sea exactamente lo que contiene la variable Pantalla7_Visible, es decir, si la variable es True se mostrara el contenedor y si la variable está en False no se mostrará.

Además, junto a este contenedor, se hará visible también un rectángulo que se encuentra en una capa inferior al Container_Pantalla7 y que cubrirá toda la pantalla 6 y la oscurecerá, permitiendo así distinguirla mejor de la ventana emergente que será la pantalla 7:

Elemento	Atributos	Código
Rectangle_Pantalla7	Visible	<code>Pantalla7_Visible</code>
	Fill	<code>RGBA(237; 237; 237; 0,5)</code>

Una vez ya dentro del contenedor encontramos, por un lado, las etiquetas de texto que muestran la información de la solicitud, gracias a la variable “SolicitudSeleccionada” que se ha establecido previamente con el registro de la solicitud en la cual se ha presionado el botón de Gestionar. Este es por ejemplo el caso de los días seleccionados:

Elemento	Atributos	Código
Label7_Dias2	Texto	<code>SolicitudSeleccionada.Dias</code>

Y por otro lado encontramos el calendario laboral. Una vez más utilizamos los mismos componentes que ya se han desarrollado y explicado para pantallas anteriores, pero en este caso no se debe mostrar el calendario del usuario de la aplicación si no el calendario del empleado que ha realizado la solicitud. Por tanto, la etiqueta de texto de lo que se muestra cada día del calendario quedaría así:

Elemento	Atributos	Código
Label7_Calend	Texto	<pre> LookUp(Calendario; SolicitudSeleccionada.Empleado.'Empleados (cre05_empleadosid)' = Empleado.'Empleados (cre05_empleadosid)' And Day(DateAdd(_firstDayInView;ThisItem.Value; TimeUnit.Days)) = Day(Dia) And Month(DateAdd(_firstDayInView;ThisItem.Value; TimeUnit.Days)) = Month(Dia) And Year(DateAdd(_firstDayInView;ThisItem.Value; TimeUnit.Days)) = Year(Dia); Siglas) </pre>

Y, además, para que en el calendario aparezcan marcados los días que se han seleccionado en la solicitud, hay que meter estos días en la colección DiasSeleccionados, que es la que ya se utilizaba previamente para para añadir los días que se seleccionaban y poder mostrarlos así. Esta funcionalidad se añade en el atributo Onselect del botón “Gestionar” de la pantalla 6:

Elemento	Atributos	Código
Button6_Gestionar	OnSelect	<pre> ForAll(Split(SolicitudSeleccionada.Dias; ", ") As DIA; Collect(DiasSeleccionados; DIA));; Set(_firstDayOfMonth; DateAdd(First(DiasSeleccionados).Value; 1 - Day(First(DiasSeleccionados).Value); TimeUnit.Days));; Set(_firstDayInView; DateAdd(_firstDayOfMonth; -(Weekday(_firstDayOfMonth; StartOfWeek.Monday) - 2 + 1); TimeUnit.Days));; </pre>

		<pre>Set(_lastDayOfMonth; DateAdd(DateAdd(_firstDayOfMonth; 1; TimeUnit.Months); -1; TimeUnit.Days));;</pre>
--	--	--

Así al seleccionar la solicitud deseada ya se introducen los días en la colección DíasSeleccionados y se pueden mostrar nada más visualizarse la pantalla 7. También se establecen las variables de _firstDayOfMonth, _firstDayInView y _lastDayOfMonth para que en el calendario se muestre directamente el mes donde se encuentran los días seleccionados y no que se muestre el mes actual como pasaba en los calendarios de las anteriores pantallas.

Posteriormente, en la parte inferior de la pantalla encontramos los botones de Rechazar y Aceptar. Los dos comparten la misma combinación de colores que los botones de las pantallas anteriores, y tienen la siguiente funcionalidad:

Elemento	Atributos	Código
Button7_Rechazar	OnSelect	<pre>Patch(Solicitudes; SolicitudSeleccionada; {'Estado (cre05_estado)':'Estado de solicitud'.Rechazada});; If(SolicitudSeleccionada.Motivo = 'Motivos Solicitud'.Vacaciones; Set(DiasAModificar; Filter(Calendario; Solicitud.IdSolicitud = SolicitudSeleccionada.IdSolicitud));; ForAll(DiasAModificar As DIA; Remove(Calendario; DIA));; Set(Pantalla7_Visible; false)</pre>

En el de rechazar se realiza, en primer lugar, un patch a la tabla Solicitudes a través del cual se modifica el estado de la solicitud seleccionada a Rechazada. En segundo lugar, si el motivo de la solicitud ha sido Vacaciones, se utiliza el bucle *ForAll* para eliminar todos los días que se habían creado al realizar la solicitud y que se encontraban en la situación de “Vacaciones no confirmadas”. Por último, se establece la variable Pantalla7_Visible para que se deje de mostrar la Pantalla 7 y se vuelva a la 6.

En el caso del botón de Aceptar, encontramos el siguiente código:

Elemento	Atributos	Código
Button7_Aceptar	OnSelect	<pre>Patch(Solicitudes; SolicitudSeleccionada; {'Estado (cre05_estado)':'Estado de solicitud'.Aceptada});; Switch(SolicitudSeleccionada.Motivo; 'Motivos Solicitud'.Vacaciones; Set(DiasAModificar; Filter(Calendario; Solicitud.IdSolicitud = SolicitudSeleccionada.IdSolicitud));; ForAll(DiasAModificar As DIA; Patch(Calendario; DIA; {Motivo: 'Tipos de dias festivos'.Vacaciones Confirmadas'; Siglas: "H"}));;</pre>

		<pre> 'Motivos Solicitud'. 'Ausencia legal'; ForAll(Split(SolicitudSeleccionada.Dias; ", ") As DIA; Patch(Calendario; Defaults(Calendario); { Dia: DateValue(DIA.Value); Motivo: 'Tipos de dias festivos'. 'Ausencia Legal'; Siglas: "LA"; Empleado:SolicitudSeleccionada.Empleado; Solicitud: SolicitudSeleccionada })); 'Motivos Solicitud'. Baja; ForAll(Split(SolicitudSeleccionada.Dias; ", ") As DIA; Patch(Calendario; Defaults(Calendario); { Dia: DateValue(DIA.Value); Motivo: 'Tipos de dias festivos'. Baja; Siglas: "B"; Empleado:SolicitudSeleccionada.Empleado; Solicitud: SolicitudSeleccionada })); 'Motivos Solicitud'. 'Días de libre disposición'; ForAll(Split(SolicitudSeleccionada.Dias; ", ") As DIA; Patch(Calendario; Defaults(Calendario); { Dia: DateValue(DIA.Value); Motivo: 'Tipos de dias festivos'. 'Días de libre disposición'; Siglas: "FD"; Empleado:SolicitudSeleccionada.Empleado; Solicitud: SolicitudSeleccionada })); 'Motivos Solicitud'. 'Visita médico'; ForAll(Split(SolicitudSeleccionada.Dias; ", ") As DIA; Patch(Calendario; Defaults(Calendario); { Dia: DateValue(DIA.Value); Motivo: 'Tipos de dias festivos'. 'Visita médico'; Siglas: "MI"; Empleado:SolicitudSeleccionada.Empleado; Solicitud: SolicitudSeleccionada }));); Set(Pantalla7_Visible; false) </pre>
--	--	---

En este botón también realizamos el mismo patch en primer lugar, pero en este caso el estado de la solicitud se modifica a Aceptada. En segundo lugar, se comprueba mediante un Switch si el motivo de la solicitud es vacaciones, y en caso de que resulte afirmativa esta condición, se irán cambiando uno a uno, con el bucle ForAll, todos los días que se han creado en la tabla Calendario al enviar la solicitud. Ya que tienen el tipo de día en Vacaciones No confirmadas y se ha de pasar a poner en Vacaciones Confirmadas debido a que se ha aceptado la solicitud. Sin embargo, si el motivo de la solicitud no es Vacaciones, no se han creado registros con cada uno de los días. Y por tanto, en el mismo Switch que se evaluaba la condición anterior, se va comprobando cual es el motivo de la solicitud y creando los días seleccionados en la tabla Calendario con el motivo y las siglas correspondientes y con el empleado que había enviado la solicitud. Finalmente, se vuelve establecer la variable Pantalla7_Visible en false para cerrar la pestaña.

Por último en esta pantalla, encontramos el icono de cerrar en la parte superior derecha de la ventana emergente:

Elemento	Atributos	Código
Icon7_Cerrar	OnSelect	<code>Set(Pantalla7_Visible; false);;</code> <code>Clear(DiasSeleccionados)</code>
	Icon	<code>Icon.Cancel</code>
	Color	<code>RGBA(137; 196; 244; 1)</code>

En este se establece que, al pulsarlo, la variable Pantalla7_Visible se ponga en False y, por tanto, se deje de mostrar la pantalla 7. Y además, se limpia la variable DíasSeleccionados para que cuando se seleccione otra solicitud no sigan guardados y aparezcan en el calendario los días de solicitudes antiguas.

5.2.8.3 Resultados

Tras implementar todo lo anterior, si seleccionamos una de las solicitudes en la pantalla 6, aparecería una ventana emergente que se vería así:

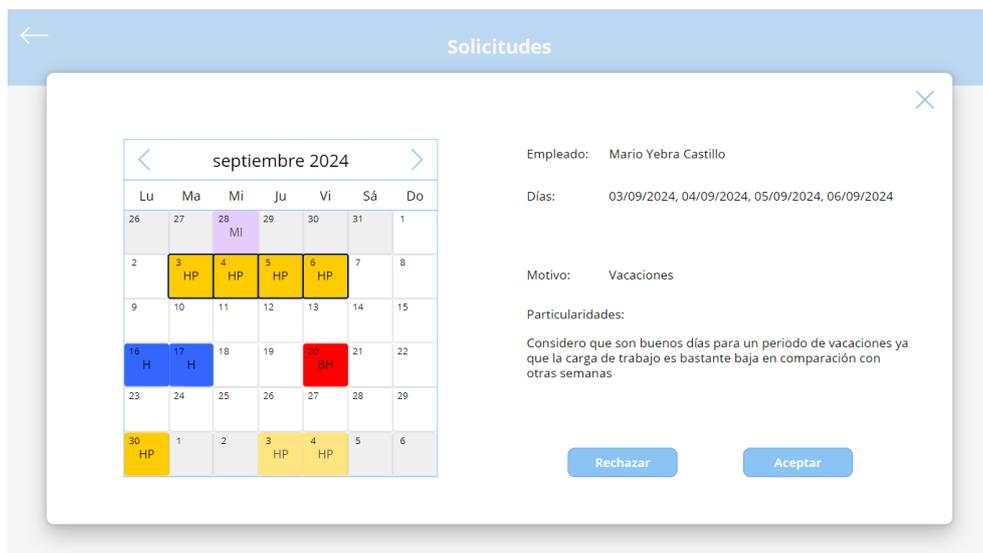


Figura 40: Resultado 1 de la pantalla 7. Fuente: Elaboración propia

En este caso, en el calendario se muestra directamente el mes de septiembre (ya que es el mes que contiene los días seleccionados) y, al ser una solicitud de vacaciones, aparecen ya los días en amarillo con las siglas HP (Vacaciones no confirmadas). Pero si el motivo fuera otro, aparecerían simplemente los bordes azules en las casillas indicando que eso son los seleccionados:

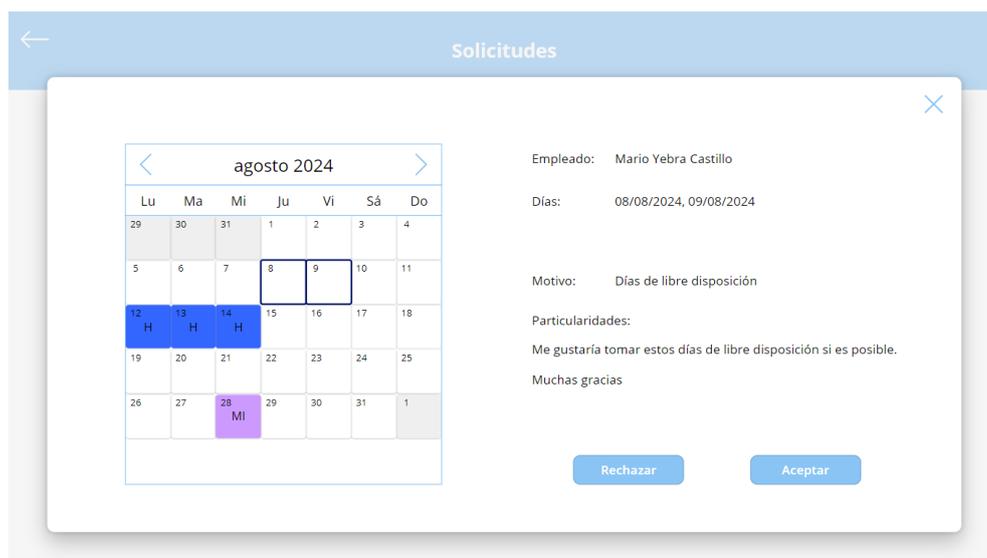


Figura 41: Resultado 2 de la pantalla 7. Fuente: Elaboración propia

Y, efectivamente, en el caso de aceptar las solicitudes, aparecen directamente en el calendario del empleado estos días que se habían seleccionado con el motivo acorde a la solicitud.

5.2.9 Pantalla 8 – Modificar festivos

5.2.9.1 Análisis funcional

Esta última pantalla será a la que accederemos al pulsar el botón “Modificar festivos” de la pantalla 5. La idea principal de esta nueva pantalla es que el administrador pueda desde aquí eliminar días festivos de cualquier trabajador y añadirlos también según desee. Por ello, se deberían encontrar los siguientes elementos:

- En primer lugar, la pantalla debería contener un formulario con diversas listas desplegables en el que el usuario pueda seleccionar que tipo de día quiere añadir, que día del calendario y a que trabajador se lo quiere añadir.
- En el caso de que el tipo de día sea un festivo nacional o de la comunidad autónoma, debería aparecer otra lista desplegable donde seleccionar la zona de ese día festivo.
- Por supuesto, debe haber un botón para pulsarlo y que así se cree en la base de datos un registro con el día, el motivo y el trabajador seleccionado.
- Por otro lado, una buena opción sería que apareciera una galería con todos los días y ausencias de todos los trabajadores donde se pueda filtrar por la columna que se desee.
- En esa galería se debería poder seleccionar uno o más registros y debería existir un botón que al pulsarlo eliminara de la base de datos todos los registros que se hayan seleccionado.
- Como siempre, también debería incluir un icono de flecha hacia atrás que permita al usuario volver a la pantalla anterior.

Y sabiendo todo esto, este sería el posible boceto de esta última pantalla de la aplicación:

Pantalla 8

Figura 42: Boceto previo de la pantalla 8. Fuente: Elaboración propia

5.2.9.2 Desarrollo

Para la primera parte de esta pantalla se va a implementar un formulario, el cual va a contener los siguientes elementos:

El primero de ellos va a ser un selector de día que ofrece Power Apps de manera predeterminada. El cual lo personalizamos ligeramente mediante los siguientes atributos:

Elemento	Atributos	Código
DatePicker8	DefaultDate	Blank()
	StartOfWeek	StartOfWeek.Monday
	BorderColor	RGBA(137; 196; 244; 1)
	IconBackground	RGBA(137; 196; 244; 1)

En segundo lugar, encontramos una lista desplegable en la que el usuario deberá elegir el tipo de ausencia que desea añadir. Es decir, si es va a ser un Festivo, si va a ser un día de Vacaciones, una visita al médico, una baja, etc.

Elemento	Atributos	Código
Dropdown8_TipoDia	Items	Choices('Tipos de dias festivos')
	Default	"Festivo"

A través del atributo Items se establece que las opciones que van a aparecer en la lista desplegable sean las mismas que las de la lista ya existente llamada ‘Tipos de días festivos’, la cual incluye ya todas las opciones posibles.

Después encontramos otras dos listas desplegables que se encuentran en la misma posición y que van a ser visibles o invisibles dependiendo de lo que se haya seleccionado en la lista “Dropdown8_TipoDia”.

En el caso de que este seleccionado “Festivo” como tipo de ausencia, se mostrará la siguiente:

Elemento	Atributos	Código
Dropdown8_Ciudades	Items	<code>Choices('Ciudades Power Platform')</code>
	Default	<code>"Nacional"</code>
	Visible	<code>If(Dropdown8_TipoDia.Selected.Value = 'Tipos de dias festivos'.Festivo; true; false)</code>

En la cual se deberá seleccionar en que zona va a ser ese festivo para que se aplique a todos los trabajadores de esa zona. Se puede seleccionar tanto “Nacional” para que se aplique a todos los trabajadores de España, como una ciudad concreta para que solo se aplique a los trabajadores de esa ciudad.

Sin embargo, si en el elemento “Dropdown8_TipoDia” se selecciona cualquier otro tipo de ausencia que no sea “Festivo”, se ocultará la lista “Dropdown8_Ciudades” y se hará visible la siguiente:

Elemento	Atributos	Código
ComboBox8_Empleado	Items	<code>Empleados</code>
	DisplayFields	<code>["cre05_nombrecompleto"]</code>
	Visible	<code>If(Dropdown8_TipoDia.Selected.Value = 'Tipos de dias festivos'.Festivo; false; true)</code>
	IsSearchable	<code>true</code>

En esta se deberá seleccionar el trabajador concreto al cual se le añadirá el día al calendario. Por ello, a través del Items se establece que se muestren todos los registros existentes en la tabla Empleados de la base de datos y con el DisplayFields se establece que el campo que se muestre en al desplegar sea el campo nombre completo. Además, estableciendo en true el atributo IsSearchable permitimos que se pueda buscar para facilitar la selección del empleado.

Finalmente, como último elemento del formulario encontramos el botón “Añadir día” que precisamente permite al usuario crear los nuevos registros en la base de datos y añadir así a los calendarios de los empleados todo lo seleccionado.

Elemento	Atributos	Código
Button8_	Text	<code>"Añadir día"</code>

AñadirDia	Onselect	<pre> If(!(Dropdown8_TipoDia.Selected.Value = 'Tipos de dias festivos'.Festivo); Patch(Calendario; Defaults(Calendario); {Empleado: ComboBox8_Empleado.Selected; Dia:DatePicker8.SelectedDate; Motivo:Dropdown8_TipoDia.Selected.Value; Siglas: Switch(Dropdown8_TipoDia.Selected.Value; 'Tipos de dias festivos'. 'Días de libre disposición'; "FD"; 'Tipos de dias festivos'.Formación; "F"; 'Tipos de dias festivos'. 'Colaboración con la US'; "US"; 'Tipos de dias festivos'. 'Reunión cliente'; "CM"; 'Tipos de dias festivos'. 'Compensados NTT DATA Centers'; "CE"; 'Tipos de dias festivos'. 'Cargado a otro proyecto o staff'; "OP"; 'Tipos de dias festivos'. 'Vacaciones Confirmadas'; "H"; 'Tipos de dias festivos'. Baja; "B"; 'Tipos de dias festivos'. 'Visita médico'; "MI"; 'Tipos de dias festivos'. 'PaP Nocturno'; "P"; 'Tipos de dias festivos'. 'Ausencia Legal'; "LA"; 'Tipos de dias festivos'. 'Día Debido'; "D"; 'Tipos de dias festivos'. 'Salida de Proyecto'; "PC"; 'Tipos de dias festivos'. 'Vacaciones NO Confirmadas'}});; Notify("El día seleccionado se ha añadido correctamente"; NotificationType.Success); If(!(Dropdown8_Ciudades.Selected.Value = 'Ciudades Power Platform'. Nacional); ForAll(Filter(Empleados; Equipo = Dropdown8_Ciudades.Selected.Value) As Persona; Patch(Calendario; Defaults(Calendario); {Empleado: Persona; Dia:DatePicker8.SelectedDate; Motivo:'Tipos de dias festivos'.Festivo; Siglas: "BH"}));; Notify("El festivo seleccionado se ha añadido correctamente"; NotificationType.Success); ForAll(Filter(Empleados; Equipo <> 'Ciudades Power Platform'. Lisboa) As Persona; Patch(Calendario; Defaults(Calendario); {Empleado: Persona; Dia:DatePicker8.SelectedDate; Motivo:'Tipos de dias festivos'.Festivo; Siglas: "BH"}));; Notify("El festivo seleccionado se ha añadido correctamente"; NotificationType.Success));; Reset(DatePicker8);; Reset(Dropdown8_TipoDia);; Reset(ComboBox8_Empleado) </pre>
	DisplayMode	<pre> If(Dropdown8_TipoDia.Selected.Value = 'Tipos de dias festivos'.Festivo; If(!IsBlank(DatePicker8.SelectedDate); DisplayMode.Edit; DisplayMode.Disabled); If(!IsBlank(DatePicker8.SelectedDate) And !IsBlank(ComboBox8_Empleado.SelectedItems); DisplayMode.Edit; DisplayMode.Disabled)) </pre>

A través del Onselect se implementa que, si se ha seleccionado Festivo en el Tipo de ausencia, se guarde ese día en el calendario de todos los trabajadores que corresponda. Y si se ha seleccionado otro tipo de ausencia se guarde en el calendario del empleado concreto seleccionado con las siglas correspondientes al tipo de ausencia seleccionado. Con el atributo DisplayMode se establece que el botón se encuentre desactivado hasta que todos los campos estén cumplimentados, y por tanto se active y ya se pueda pulsar.

Por otro lado, la segunda parte de la pantalla va a consistir en brindar la posibilidad de eliminar los días que se deseen. Para ello, se implementa una galería similar a la de las pantallas anteriores, pero que en este caso muestra todos los registros de la tabla Calendario, es decir, muestra individualmente todas las ausencias que todos los empleados tienen en su calendario. Por supuesto y como en las anteriores, se implementa la posibilidad de filtrar los registros mediante entradas de texto para todos los campos.

Elemento	Atributos	Código
Gallery8_Calendarario	Items	<pre>Filter(Calendarario; (IsBlank(Trim(TextInput8_Empleado.Text)) Or Trim(TextInput8_Empleado.Text) in Concatenate(Empleado.Nombre; " "; Empleado.'1er Apellido'; " "; Empleado.'2do Apellido')) And (IsBlank(Trim(TextInput8_Dia.Text)) Or Trim(TextInput8_Dia.Text) in Dia) And (IsBlank(Trim(TextInput8_Motivo.Text)) Or Trim(TextInput8_Motivo.Text) in Motivo) And (IsBlank(Trim(TextInput8_Siglas.Text)) Or Trim(TextInput8_Siglas.Text) in Siglas) And (IsBlank(Trim(TextInput8_Correo.Text)) Or Trim(TextInput8_Correo.Text) in Empleado.'Correo electronico')))</pre>

En este caso los campos que se muestran en la galería son: Nombre completo del empleado, día, motivo de la ausencia, siglas que le corresponden a ese día y correo electrónico del empleado, que puede ser útil para filtrar los registros.

Pero además en cada registro que aparezca en la galería se mostrara una casilla marcable (Checkbox) para que se puedan seleccionar los registros que se desean eliminar:

Elemento	Atributos	Código
Button6_Gestionar	OnCheck	<pre>Collect(EliminarDias; ThisItem.IdCalendarario)</pre>
	OnUncheck	<pre>Remove(EliminarDias; Lookup(EliminarDias; Value = ThisItem.IdCalendarario))</pre>
	CheckmarkFill	<pre>RGBA(137; 196; 244; 1)</pre>
	CheckmarkFill	<pre>RGBA(0; 134; 208; 1)</pre>

Con esto se consigue que al pulsar la casilla de uno de los registros se guarde el id de ese registro en la colección "EliminarDias". Y al desmarcar la casilla se borre ese registro de la colección.

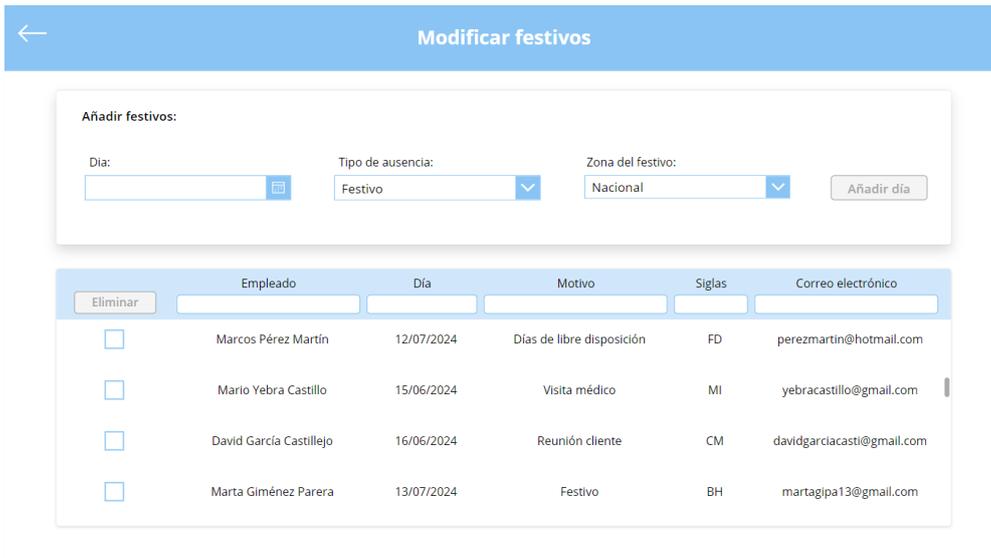
Y finalmente, encontramos un botón que servirá para eliminar de la base de datos todos los registros de que se encuentren marcados:

Elemento	Atributos	Código
Button6_Gestionar	OnSelect	<code>ForAll(EliminarDias As DiaSelecc; Remove(Calendario; LookUp(Calendario; IdCalendario = DiaSelecc.Value)));; Clear(EliminarDias)</code>
	DisplayMode	<code>If(!IsEmpty(EliminarDias); DisplayMode.Edit; DisplayMode.Disabled)</code>

Con esto se utiliza la colección “EliminarDias” para ir eliminando de la tabla Calendario con un bucle todos los registros que se habían seleccionado, y posteriormente se limpia la colección para que pueda volver a ser utilizada. Además, de nuevo este botón solo estará activo cuando se haya seleccionado alguna casilla, en caso de que no sea así no se podrá pulsar.

5.2.9.3 Resultados

Con todo lo anterior implementado y desarrollado, así es como nos encontramos la pantalla 8 nada más acceder:



← **Modificar festivos**

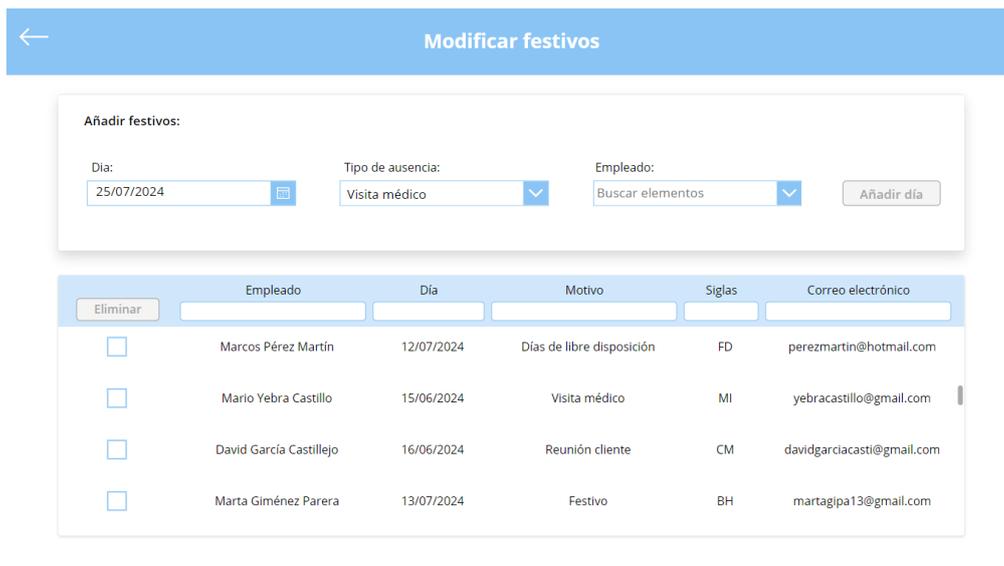
Añadir festivos:

Día: Tipo de ausencia: Zona del festivo:

Eliminar	Empleado	Día	Motivo	Siglas	Correo electrónico
<input type="checkbox"/>	Marcos Pérez Martín	12/07/2024	Días de libre disposición	FD	perezmartin@hotmail.com
<input type="checkbox"/>	Mario Yebra Castillo	15/06/2024	Visita médico	MI	yebrecastillo@gmail.com
<input type="checkbox"/>	David García Castillejo	16/06/2024	Reunión cliente	CM	davidgarciacasti@gmail.com
<input type="checkbox"/>	Marta Gilménez Parera	13/07/2024	Festivo	BH	martagipa13@gmail.com

Figura 43: Resultado 1 de la pantalla 8. Fuente: Elaboración propia

Si se selecciona el día en el formulario y cambiamos el tipo de ausencia a algo que no sea Festivo, la pantalla pasaría a estar así:



Modificar festivos

Añadir festivos:

Día:

Tipo de ausencia:

Empleado:

Eliminar	Empleado	Día	Motivo	Siglas	Correo electrónico
<input type="checkbox"/>	Marcos Pérez Martín	12/07/2024	Días de libre disposición	FD	perezmartin@hotmail.com
<input type="checkbox"/>	Mario Yebra Castillo	15/06/2024	Visita médico	MI	yebrecastillo@gmail.com
<input type="checkbox"/>	David García Castillejo	16/06/2024	Reunión cliente	CM	davidgarciacasti@gmail.com
<input type="checkbox"/>	Marta Giménez Parera	13/07/2024	Festivo	BH	martagipa13@gmail.com

Figura 44: Resultado 2 de la pantalla 8. Fuente: Elaboración propia

Y si finalmente seleccionamos el empleado al que se lo queremos añadir, se activaría el botón:



Modificar festivos

Añadir festivos:

Día:

Tipo de ausencia:

Empleado:

Eliminar	Empleado	Día	Motivo	Siglas	Correo electrónico
<input type="checkbox"/>	Marcos Pérez Martín	12/07/2024	Días de libre disposición	FD	perezmartin@hotmail.com
<input type="checkbox"/>	Mario Yebra Castillo	15/06/2024	Visita médico	MI	yebrecastillo@gmail.com
<input type="checkbox"/>	David García Castillejo	16/06/2024	Reunión cliente	CM	davidgarciacasti@gmail.com
<input type="checkbox"/>	Marta Giménez Parera	13/07/2024	Festivo	BH	martagipa13@gmail.com

Figura 45: Resultado 3 de la pantalla 8. Fuente: Elaboración propia

Al presionar el botón “Añadir día” se reinician los campos y aparece una notificación confirmando que se ha guardado correctamente todo lo seleccionado.

Por otro lado, si deseamos eliminar días de algunos empleados se han de pulsar las casillas de los registros correspondientes:

←
Modificar festivos

Añadir festivos:

Día:
 Tipo de ausencia:
 Zona del festivo:

	Empleado	Día	Motivo	Siglas	Correo electrónico
<input type="checkbox"/>	Marcos Pérez Martín	12/07/2024	Días de libre disposición	FD	perezmartin@hotmail.com
<input checked="" type="checkbox"/>	Mario Yebra Castillo	15/06/2024	Visita médico	MI	yebracastillo@gmail.com
<input checked="" type="checkbox"/>	David García Castillejo	16/06/2024	Reunión cliente	CM	davidgarciacasti@gmail.com
<input type="checkbox"/>	Marta Giménez Parera	13/07/2024	Festivo	BH	martagipa13@gmail.com

Figura 46: Resultado 4 de la pantalla 8. Fuente: Elaboración propia

Al hacerlo se activa botón de “Eliminar”, y si se pulsa, efectivamente se eliminan esos días de la base de datos y desaparecen de manera instantánea de la galería.

5.3 Desarrollo de flujos de trabajo automatizados

Además del desarrollo de las diferentes pantallas de la aplicación, una parte importante también para conseguir un desarrollo óptimo de una solución son los flujos automatizados. Los cuales precisamente, y como se ha mencionado anteriormente, se desarrollan mediante Power Automate.

5.3.1 Flujo - Actualizar días a final de año

Este flujo de trabajo se va a encargar de que, al acabar el año, los días de vacaciones que le hayan sobrado a cada trabajador pasen a la columna de 'Año Anterior' y que los días de la columna 'Pendientes', es decir, los disponibles del año actual se reinicien a 23, que es el número de días que corresponden a un año completo.

Para ello se ha creado un flujo programado que contiene los siguientes pasos:

En primer lugar, se establece que el flujo se vaya a ejecutar el día 1 de enero de 2025 y que se repita la ejecución cada 12 meses, es decir, el 1 de enero de cada año



Figura 48: Captura del primer paso del flujo en detalle. Fuente: Elaboración propia



Figura 47: Captura de todos los pasos que componen el flujo. Fuente: Elaboración propia

En el segundo paso se obtienen todos los registros que se van a modificar, que son todos los registros de la tabla Empleados. Extrayendo concretamente el id de cada registro y la cantidad de días pendientes, que son los que han sobrado:



Figura 49: Captura del segundo paso del flujo en detalle. Fuente: Elaboración propia

Posteriormente se implemente un bucle que se va a repetir tantas veces como registros se hayan obtenido. Para ello se le pasa al bucle el Output de el paso 2.

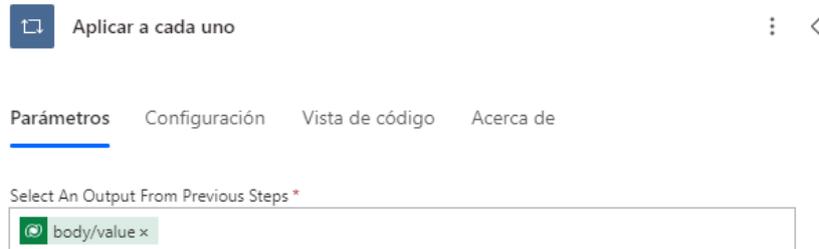


Figura 50: Captura del tercer paso del flujo en detalle. Fuente: Elaboración propia

Y finalmente, dentro de ese bucle se va actualizando uno a uno todos los registros de la tabla Empleados. Indicando que registro con los Id que se han obtenido y actualizando los campos de “Año anterior” con el número de días pendientes que se había obtenido, y el de “Pendientes” estableciéndolo a 23.

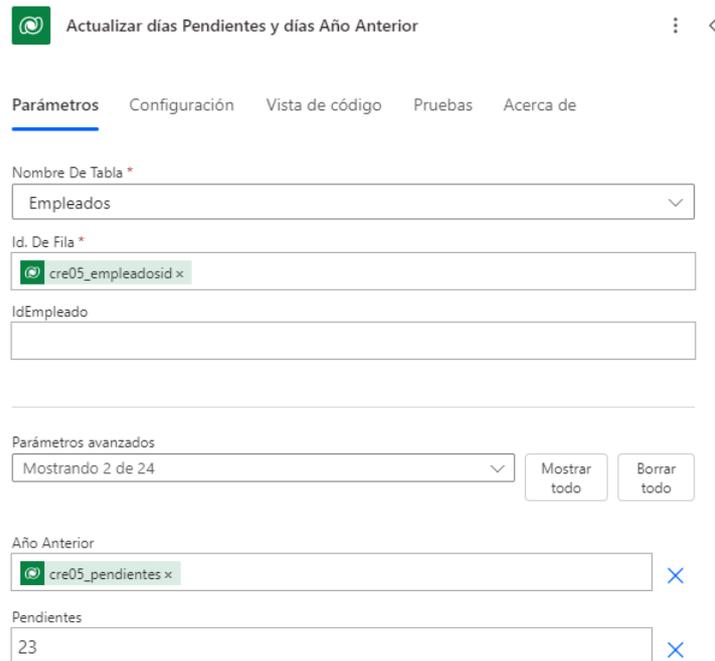


Figura 51: Captura del último paso del flujo en detalle. Fuente: Elaboración propia

Y tras testear el flujo se ha comprobado que funciona correctamente, actualizando de la manera deseada los campos de Año Anterior y Pendientes.

6 Resultados de la solución

6.1 Resultados finales

Tras el exhaustivo proceso de desarrollo, en el que se han implementado todas las pantallas, elementos, flujos y funcionalidades, se ha logrado una solución integral y perfectamente interconectada, la cual se encuentra completamente lista y preparada para ser utilizada por los usuarios.

Finalmente, este es el diagrama de las 9 pantallas que contiene la aplicación, el cual muestra todas las posibilidades de navegación y conexión entre ellas:

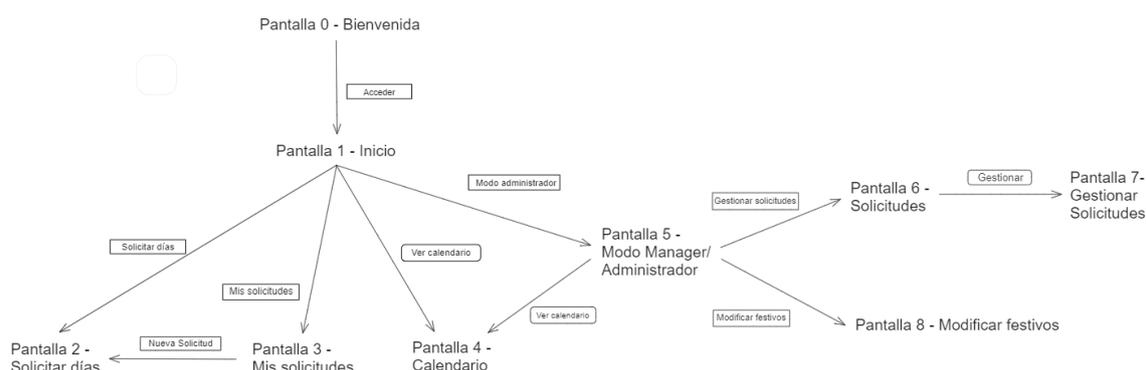


Figura 52: Diagrama final de pantallas y navegación. Fuente: Elaboración propia

Dejando de lado la posibilidad que hay en cada pantalla, evidentemente, de volver a la anterior pantalla.

Esto nos da como resultado una interfaz completamente intuitiva y amigable para el usuario, que permite la navegación y la utilización de la aplicación de una manera fácil, clara y entendible. Además, esta interfaz no solo integra eficazmente todas las funcionalidades requeridas, sino que además brinda un diseño limpio y bonito que mejora con creces el diseño del anterior sistema de gestión de las vacaciones.

Y principalmente, se pueden valorar estos resultados de manera positiva ya que se han conseguido subsanar todos los problemas y satisfacer todas las necesidades que existían con el anterior método de gestión. Todos estos problemas que se han solucionado son los mencionados en el apartado “3.2 Identificación de vulnerabilidades del sistema actual y necesidades de mejora”.

6.2 Funcionamiento tras uso

Como en cualquier tipo de desarrollo de aplicaciones, una parte fundamental de ese proceso es el testing, es decir, realizar miles de pruebas en la aplicación tratando de abarcar todas las posibles combinaciones y funcionalidades que existen. Evidentemente esto se ha ido haciendo a la vez que se iban implementando nuevos elementos o funciones, pero una vez terminada la aplicación, se ha de seguir haciendo con mayor énfasis si cabe para poder acabar con todos los errores o fallos que contenga la aplicación antes de que sea utilizada por los usuarios finales.

En este caso, el periodo de testing ha sido un completo éxito. Se ha evaluado y probado rigurosamente la aplicación tanto por mí, como por otros posibles usuarios finales de la misma. Lo cual además ha permitido detectar y solucionar ligeros fallos o imperfecciones que han sido encontrados.

Pero principalmente los resultados han mostrado la alta estabilidad y eficiencia que posee la solución, confirmando que la aplicación está bien equipada para satisfacer todas las necesidades de los usuarios en un entorno real. Además, gracias a las pruebas realizadas por usuarios externos se ha podido confirmar un aumento en la satisfacción por parte de los usuarios respecto al anterior modelo de gestión de vacaciones, garantizando así una futura experiencia óptima.

Sumado a esto, este proceso de pruebas ha servido para poder identificar también ciertos aspectos o ámbitos con los que se podría mejorar aún más la aplicación. Lo que permitiría para un futuro poder seguir trabajando en la aplicación, optimizando su rendimiento y funcionalidad, y por supuesto la satisfacción de los usuarios.

6.3 Posibles mejoras

Como en cualquier ámbito de la vida, es importante ser exigente y ambicioso. Y hay que tratar de seguir mejorando y actualizándose. Por tanto, no iba a ser menos en el caso de esta aplicación, para la cual se han identificado las siguientes posibles mejoras de cara a ir desarrollándolas en un futuro.

- Una posible mejora sería que, actualmente, los datos de los trabajadores como nombre, correo, numero de empleado, rol... se introducen inicialmente de forma manual en la base de datos. Por tanto, podría ser útil una nueva pantalla (similar a la de modificar festivos, con formularios) a través de la cual los administradores puedan crear y modificar esos datos de los empleados, y que no hiciera falta modificarlo directamente en la base de datos
- Similar a lo anterior, actualmente se guardan automáticamente las fechas y horas en las que se crean o se modifican todos los registros, y en muy pocos casos esos datos se pueden visualizar desde la aplicación, ya que en la mayoría de los casos no es una información relevante. Pero podría ser interesante plantear una pantalla de auditoría de cambios en la que se puedan visualizar todos estos cambios, quien los ha realizado y en qué fecha, de manera ágil y sin tener que acceder a la base de datos.
- Por otro lado, podría ser bastante útil implementar, a través de un flujo automatizado, que cada vez que un empleado envíe una nueva solicitud, sus managers, es decir, los que se la pueden aceptar, reciban una notificación por correo electrónico de que tal trabajador ha creado una nueva solicitud.
- También se podría hacer a la inversa, es decir, implementar que cuando el mánager acepte o rechace una solicitud, le llegue un correo al empleado avisándole de eso mismo

7 Conclusiones

La implementación del gestor de vacaciones ha representado un avance significativo para la empresa, no solo en términos de eficiencia operativa, sino también en cuanto a la satisfacción y bienestar de los empleados. Desde su puesta en marcha, la aplicación ha comenzado a ser utilizada activamente por el personal, lo que ha generado un impacto positivo palpable en la dinámica de gestión de las vacaciones y días festivos. La modernización del sistema, que hasta ahora se manejaba de manera manual actualizando un Excel, ha permitido que los trabajadores puedan solicitar y consultar sus días de descanso de manera más ágil y transparente, lo que ha redundado en un notable aumento de la satisfacción general.

Uno de los beneficios más destacables ha sido el ahorro de tiempo tanto para los empleados como para los responsables de recursos humanos. La automatización de los procesos de solicitud y aprobación de vacaciones ha reducido considerablemente el tiempo que anteriormente se destinaba a estas tareas, permitiendo que los recursos humanos se enfoquen en actividades de mayor valor estratégico para la empresa. Además, la aplicación ha contribuido a una reducción significativa de errores en la gestión de los datos relacionados con las vacaciones, lo que anteriormente podía generar confusiones y malentendidos que afectaban tanto a la planificación interna como a la experiencia del empleado.

Para la empresa, esta modernización ha significado un paso crucial hacia la digitalización de sus procesos internos. Al adoptar una solución tecnológica avanzada, la empresa no solo se alinea con las mejores prácticas del sector, sino que también mejora su capacidad para adaptarse a las necesidades cambiantes de sus trabajadores y a las demandas de un entorno laboral cada vez más digitalizado. Esta transformación no solo optimiza el flujo de trabajo, sino que también refuerza el compromiso de la empresa con la innovación y la mejora continua, asegurando así una mayor competitividad en el mercado.

En conclusión, el desarrollo e implementación del gestor de vacaciones ha superado las expectativas iniciales, ofreciendo a la empresa un sistema moderno y eficiente que ha mejorado notablemente la experiencia del empleado, la precisión en la gestión de datos y la productividad general. Este avance tecnológico no solo ha beneficiado a los trabajadores, quienes ahora disfrutan de un proceso más claro y cómodo para gestionar sus vacaciones, sino que también ha permitido a la empresa optimizar sus recursos y avanzar en su proceso de modernización, preparándose para los desafíos futuros con una base tecnológica más sólida.

Por otro lado, la realización de este Trabajo de Fin de Grado (TFG) ha sido para mí una experiencia profundamente enriquecedora tanto a nivel académico como personal. A lo largo de este proceso, he tenido la oportunidad de aplicar e incrementar bastantes de los conocimientos adquiridos durante el Grado en Tecnología Digital y Multimedia, teniendo la oportunidad de enfrentarme a desafíos reales y desarrollando habilidades que serán fundamentales para mi futuro profesional.

Otro de los aspectos más gratificantes de esta experiencia ha sido la posibilidad de descubrir y aprender en profundidad sobre este nuevo estilo de desarrollo low code.

Permitiéndome adquirir un conocimiento más amplio y detallado sobre la herramienta de Power Platform, cuyo potencial y uso está creciendo de manera exponencial. Y que, por tanto, este conocimiento va a ser clave en mi futuro laboral cercano, sabiendo además que mi continuidad en la empresa tras el periodo de prácticas está confirmada.

Además, la realización de este proyecto ha contribuido de manera significativa al desarrollo de muchas más habilidades personales como la capacidad crítica, que ha sido clave a la hora de analizar e identificar problemas para proponer soluciones, y la capacidad de investigación, importantísima en cualquier tipo de programación a la hora de buscar en internet y descubrir nuevas fórmulas y métodos que se pueden utilizar para un desarrollo óptimo.

La planificación y gestión del tiempo también han sido elementos cruciales en la realización de este TFG. He aprendido a establecer objetivos claros y realistas, a priorizar tareas y a mantener un calendario de trabajo organizado. Este aprendizaje no solo ha sido útil para la ejecución de este proyecto, sino que también me ha dotado de herramientas valiosas para gestionar futuros proyectos en mi vida profesional.

A pesar de ser un trabajo individual, el trabajo en equipo y la comunicación también han sido aspectos destacados. Ya que la colaboración con mis tutores, tanto el de la universidad como el de la empresa, y la ayuda y opinión de mis compañeros de trabajo han sido esenciales para realización de este proyecto. Dejando claro una vez más la importancia de escuchar y valorar las opiniones de los demás, así como de ser flexible y adaptable ante las sugerencias y críticas.

A nivel personal, este TFG ha sido un viaje de autodescubrimiento y crecimiento. He aprendido a confiar más en mis capacidades y a superar mis limitaciones. La perseverancia y la dedicación han sido clave para enfrentar los momentos de duda e incertidumbre. Este proyecto me ha demostrado que, con esfuerzo y determinación, es posible alcanzar las metas propuestas.

En conclusión, la realización de este Trabajo de Fin de Grado ha sido una experiencia formativa integral que me ha preparado para los retos del mundo profesional. He desarrollado competencias técnicas, habilidades de gestión y capacidades interpersonales que serán fundamentales en mi futuro. Estoy profundamente agradecido por esta oportunidad y por todo el apoyo recibido a lo largo de este camino.

8 Bibliografía

- Aglaia. (2024, 2 julio). *Power Automate: Todo lo que necesitas saber* | Aglaia Consulting. Obtenido de: <https://aglaia.es/power-automate/>

- Algoritmia. (2022, 28 septiembre). *¿Qué es Microsoft Power Automate y cómo comenzar a usarlo?* Obtenido de: <https://algoritmia8.com/2019/12/10/que-es-microsoft-power-automate-y-como-comenzar-a-usarlo/>

- BOE-A-2015-11430 Real Decreto Legislativo 2/2015, de 23 de octubre, por el que se aprueba el texto refundido de la Ley del Estatuto de los Trabajadores. (2015, 24 octubre). Obtenido de: <https://www.boe.es/buscar/act.php?id=BOE-A-2015-11430#a38>
- BUFETE DE PRADA, ABOGADOS. (2019, 22 abril). *¿Se pueden compensar económicamente las vacaciones?* Obtenido de: <https://www.bufetedeprada.com/compensacion-economica-vacaciones/>
- Caltico. (2022, 26 julio). *¿Qué es Power Apps y para qué sirve?* Obtenido de: <https://www.caltico.es/que-es-power-apps-y-para-que-sirve/>
- Canorea, E. (2021, 16 marzo). *¿Qué es Microsoft Power Automate? ¿Y Microsoft Flow?* Plain Concepts. Obtenido de: <https://www.plainconcepts.com/es/que-es-microsoft-power-automate/>
- Darias Pérez, S. (2023, 9 enero). *Power Platform: ¿Qué debes saber y cuáles son sus beneficios?* Intelequia. Obtenido de: <https://intelequia.com/es/blog/post/power-platform-qu%C3%A9-debes-saber-y-cu%C3%A1les-son-sus-beneficios>
- Equipo editorial de IONOS. (2020, 1 octubre). *Low code: definición, características y ámbitos de aplicación.* IONOS Digital Guide. Obtenido de: <https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/que-es-el-low-code/>
- Factorial. (2024, 18 junio). *Políticas de vacaciones: disfrutar vacaciones en días sueltos.* Obtenido de: <https://factorialhr.es/blog/politicas-de-vacaciones-todo-lo-que-debes-saber/>
- Ferri, M. (2019, 13 noviembre). *¿Qué es Power Platform?* Aglaia. Obtenido de: <https://aglaia.es/blog/actualidad-microsoft/que-es-power-platform/>
- Grupo Active. (2024). *¿Qué es Power Platform?* Obtenido de: <https://grupoactive.es/es/software/microsoft-power-platform/que-es-power-platform>
- Iberdrola. (2024). *Qué es el low code o programación sin código.* Obtenido de: <https://www.iberdrola.com/innovacion/low-code>
- Iglesias Castro, B. (2021, 30 agosto). *Power FX, el nuevo lenguaje de programación para usuarios de Excel.* Formadores IT. Obtenido de: <https://formadoresit.es/power-fx-el-nuevo-lenguaje-de-programacion-para-usuarios-de-excel/>
- Kaushik, R. (2021, 7 enero). *Microsoft Power Apps: Canvas apps vs Model-driven apps.* Cynoteck. Obtenido de: <https://cynoteck.com/es/blog-post/microsoft-power-apps-canvas-apps-vs-model-driven-app/>
- Lindhorst, G., Maniar, T. y Deore, A. (2024, 22 marzo). *Información general sobre Microsoft Power Fx - Power Platform.* Microsoft Learn. Obtenido de: <https://learn.microsoft.com/es-es/power-platform/power-fx/overview>
- Maniar, T., O'Rourke, M. y Brady, G. (2024, 16 agosto). *¿Qué es Power Apps? - Power Apps.* Microsoft Learn. Obtenido de: <https://learn.microsoft.com/es-es/power-apps/powerapps-overview>

- Microsoft. (2024) *Herramientas de bajo código con tecnología de IA | Microsoft Power Platform*. Obtenido de: <https://www.microsoft.com/es-es/power-platform#products>
- Ouakli Boudarene, M. (2023). *Caso de estudio del uso de las Power Platform en un entorno empresarial: GSK Academy*. [Trabajo Fin de Grado de Ingeniería Informática, Escuela de Ingeniería Informática de Segovia]. Obtenido de: [https://uvadoc.uva.es/bitstream/handle/10324/59825/TFG-B.%201914.pdf?sequence=](https://uvadoc.uva.es/bitstream/handle/10324/59825/TFG-B.%201914.pdf?sequence=1)
- Palma Carpio, L. (2024, 12 abril). *Las vacaciones anuales y su regulación en el Estatuto de los Trabajadores*. Civic Abogados. Obtenido de: <https://civicabogados.com/vacaciones-anuales/>
- Peart, M., O'Rourke, M. y Brady, G. (2024, 13 junio). *¿Qué es Microsoft Dataverse? Power Apps*. Microsoft Learn. Obtenido de: <https://learn.microsoft.com/es-es/power-apps/maker/data-platform/data-platform-intro>
- Personio. (2023, 6 abril). *Vacaciones de 22 o 23 días laborables: ¿cuántos días corresponden a cada trabajador?* Obtenido de: <https://www.personio.es/glosario/vacaciones-22-o-23-dias-laborables/>
- Piñuel, M. (2023, 6 febrero). *Microsoft Power Platform: todo lo que necesitas saber*. Plain Concepts. Obtenido de: <https://www.plainconcepts.com/es/microsoft-power-platform/>
- Ríos, K. (2024, 19 febrero). *Las vacaciones laborales, todo lo que necesitas saber*. Payfit. Obtenido de: <https://payfit.com/es/contenido-practico/vacaciones-laborales/>
- Roig Hervás, D. (2021). *Tecnologías Low-Code y No-Code: Un caso práctico para estudiar su potencial y limitaciones*. [Trabajo Fin de Grado de Ingeniería Informática, Universitat Politècnica de València]. Obtenido de: [https://riunet.upv.es/bitstream/handle/10251/174725/Roig%20-%20Tecnologias%20Low-Code%20y%20No-Code%20Un%20caso%20practico%20para%20estudiar%20su%20potencial%20y%20lim](https://riunet.upv.es/bitstream/handle/10251/174725/Roig%20-%20Tecnologias%20Low-Code%20y%20No-Code%20Un%20caso%20practico%20para%20estudiar%20su%20potencial%20y%20limitaciones.pdf?sequence=1)
[itaciones.pdf?sequence=1](https://riunet.upv.es/bitstream/handle/10251/174725/Roig%20-%20Tecnologias%20Low-Code%20y%20No-Code%20Un%20caso%20practico%20para%20estudiar%20su%20potencial%20y%20limitaciones.pdf?sequence=1)