



Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Departamento de Estadística, Informática y Matemáticas

Trabajo de fin de máster

Ecosistema de Aprendizaje: Innovación en la formación corporativa

Autor:

Yasin Ez-ziani Chakir

Supervisor:

Ugaitz Amozarrain Perez

Pamplona, 2022

Tu dedicatoria, opcional

Índice general

Índice de figuras	III
Índice de tablas	V
Listings	VI
1. Introducción	1
2. Análisis de requisitos	3
2.1. Requisitos funcionales	3
2.2. Requisitos no funcionales	4
2.3. Roles	4
3. Tecnologías y herramientas utilizadas	7
3.1. Lenguaje de programación y ecosistema .NET	7
3.2. Interfaz de usuario	8
3.3. Gestión de datos: Entity Framework y SQL Server	9
3.4. Sistema de notificaciones y comunicaciones	10
4. Estructura del proyecto	11
4.1. Presentación	12
4.2. Aplicación	14
4.3. Infraestructura	15
4.4. Dominio	15
5. Desarrollo del Ecosistema de Aprendizaje	17
5.1. Acceso a plataforma de aprendizaje	18
5.2. Acceso directo a la plataforma de ciberseguridad	19

5.3.	Solicitar nueva acción formativa	19
5.4.	Solicitar incluirse en una acción formativa ya creada	28
5.5.	Clonación y valoraciones	29
5.5.1.	Información para responsables y directores	29
5.6.	Planes de formación y propuestas	31
6.	Funcionamiento interno	37
6.1.	Arquitectura técnica y organización del proyecto	37
6.2.	Implementación de consultas (Query Handlers)	38
6.3.	Implementación de comandos (Command Handlers)	40
6.4.	Acceso a datos e infraestructura	43
6.5.	Sistema de permisos y filtros de autorización	46
6.6.	Sistema de notificaciones y enlaces únicos	48
6.7.	Procesamiento de encuestas y valoraciones	49
6.8.	Conclusión técnica	49
	Bibliografía	51
	Índice alfabético	53
	Apéndice	53

Índice de figuras

4.1. Diagrama de una arquitectura por capas.	11
5.1. Pantalla home con los accesos directos.	18
5.2. Pantalla plataforma de aprendizaje.	18
5.3. Plataforma de ciberseguridad.	19
5.4. Cuestionario INTAKE para solicitar formación.	20
5.5. Formulario de datos generales a rellenar.	20
5.6. Ventana para seleccionar propuesta a asignar a la formación.	22
5.7. Pestañas que cargan después de crear una formación.	22
5.8. Pestaña de los docentes de la formación.	23
5.9. Seleccionar asistentes para enviarles envío de convocatoria.	23
5.10. Página para rellenar encuesta de eficacia formativa.	24
5.11. Pestaña de valoraciones de una formación.	24
5.12. Resultado de una encuesta realizada a una formación.	25
5.13. Pestaña de calendario de una formación.	25
5.14. Pestaña con el histórico de cambios en una formación.	26
5.15. Pestaña de gestión de una formación.	26
5.16. Listado de pedidos para asignar a una formación.	27
5.17. Pestaña de gestión de una formación con pedidos asignados.	27
5.18. Pestaña de solicitudes de inscripción pendientes de aprobar.	28
5.19. Pantalla de HOME con las formaciones ya tramitadas sin comenzar.	28
5.20. Botones de clonación de la formación.	29
5.21. Flujo de aprobación y notificaciones de las acciones formativas, incluyendo rutas de rechazo.	31
5.22. Pantalla con el listado de planes de formación anual.	32
5.23. Pantalla con el formulario para crear un plan de formación anual.	32
5.24. Pantalla de edición del plan de formación anual con todas las pestañas.	33

5.25. Pantalla de departamentos del plan de formación anual.	33
5.26. Usuarios con permiso de un departamento del plan de formación anual. . . .	33
5.27. Pantalla con el listado de las propuestas de plan de formación anual.	34
5.28. Pantalla con el formulario para crear una propuesta del plan de formaciones.	34
5.29. Pantalla con el listado de las propuestas de plan de formación anual con una fila seleccionada.	35
5.30. Pantalla con el listado de las propuestas de plan de formación anual con dos filas seleccionadas.	35
5.31. Pestaña de asistentes dentro de la propuesta del plan de formaciones.	36
5.32. Pestaña de propuestas unificadas dentro de la propuesta del plan de forma- ciones.	36
5.33. Pestaña de formaciones creadas que tienen asignada la propuesta del plan de formaciones.	36
Apéndice 1. Listado de mantenimiento de modalidad.	53
Apéndice 2. Listado de mantenimiento de tipos de contenido.	53
Apéndice 3. Listado de mantenimiento de medios pedagógicos.	54
Apéndice 4. Pestaña datos generales de configuración.	54
Apéndice 5. Pestaña de configuración de correo.	54
Apéndice 6. Pestaña con las plantillas de los correos.	55
Apéndice 7. Ejemplo con una plantilla de correo.	55
Apéndice 8. Pestaña sharepoint de configuración.	55
Apéndice 9. Pestaña administrador de configuración.	56

Índice de tablas

Listings

4.1. Controlador del proyecto WebUI con métodos de selección de formaciones almacenados en base de datos.	12
4.2. Interfaz de formaciones con métodos para llevar a cabo las operaciones sobre la base de datos.	14
6.1. Clase Handler que realiza una consulta para obtener formaciones.	38
6.2. Clase DTO para filtrar las formaciones.	40
6.3. Clase Handler para crear una formación.	40
6.4. Clase Validator para crear una formación.	43
6.5. Mapeo de tablas mediante Fluent API.	44
6.6. Mapeo de tablas mediante Fluent API.	44
6.7. Clase que valida el permiso del usuario para poder consultar vistas o realizar acciones.	46

En la actualidad, las organizaciones se encuentran inmersas en un entorno empresarial en constante transformación, donde la gestión del conocimiento y el desarrollo de competencias profesionales se han convertido en factores estratégicos para garantizar la competitividad y la sostenibilidad a largo plazo. La capacidad de formar a los equipos de manera ágil, estructurada y alineada con los objetivos corporativos es un desafío clave para cualquier empresa que aspire a mantener una posición destacada en su sector.

En este contexto, surge el presente Trabajo de Fin de Máster, que tiene como propósito el diseño, desarrollo e implementación de la aplicación web Ecosistema de Aprendizaje en colaboración con Tracasa Global [1] y Tracasa Instrumental [2]. Esta solución tecnológica se concibe como una plataforma integral destinada a optimizar los procesos de gestión del aprendizaje, permitiendo a los trabajadores acceder a un entorno digital centralizado desde el cual pueden solicitar nuevas acciones formativas, inscribirse en programas ya existentes y acceder a diferentes recursos de capacitación externos.

La aplicación no solo buscará facilitar el acceso a la formación, sino también incorporar un sistema de validación y aprobación por parte de responsables y directores, garantizando así la coherencia entre las necesidades individuales de los empleados y las prioridades estratégicas de la empresa. Además, se prevé que incluya funcionalidades avanzadas como la planificación de actividades, el seguimiento del progreso y la valoración de la eficacia formativa, con el fin de contribuir a una mayor trazabilidad y transparencia en los procesos de desarrollo profesional.

El desarrollo del proyecto se estructurará en distintas fases claramente definidas. En primer lugar, se llevará a cabo una fase de análisis de requisitos, en la que se recopilarán las necesidades funcionales y técnicas de la empresa, identificando los procesos clave a digitalizar y los distintos perfiles de usuario que interactuarán con la aplicación (empleados, responsables, directores y administradores del sistema). A partir de esta información se definirán los requisitos funcionales principales, como la creación y gestión de acciones formativas, la validación de solicitudes, la integración con plataformas externas de aprendizaje, la generación de informes de seguimiento y la automatización de notificaciones por correo.

Posteriormente, se realizará la fase de diseño, en la que se definirá la arquitectura de la solución siguiendo un modelo por capas, garantizando la separación de responsabilidades y la escalabilidad del sistema. En esta etapa se elaborarán los diagramas de flujo, los modelos de datos y la estructura de componentes que servirán como base para la fase de implementación.

Durante la fase de desarrollo, se procederá a construir los distintos módulos de la aplicación, que se verá en la sección de *Desarrollo del Ecosistema de Aprendizaje*, utilizando tecnologías modernas del ecosistema *.NET* [3] y *C#* [4], junto con *TypeScript* [5] y *HTML*[6] / *CSS*[7] para la capa de presentación. Se aplicará el patrón arquitectónico *Command Query Responsibility Segregation (CQRS)* [8] para separar las operaciones de lectura y escritura, asegurando así un código más limpio y mantenible.

Una vez implementada la base del sistema, se llevará a cabo la fase de pruebas, en la que se realizarán tanto pruebas unitarias como de integración [9], además de validaciones funcionales con usuarios de la empresa colaboradora. Este proceso permitirá detectar posibles errores, ajustar la experiencia de usuario y garantizar la fiabilidad del sistema antes de su despliegue.

Finalmente, la fase de despliegue y entrega contemplará la puesta en marcha de la aplicación en el entorno corporativo, la configuración de las bases de datos y servidores, así como la formación de los usuarios en el uso de la herramienta.

Se estima que el desarrollo completo del proyecto se extenderá a lo largo de aproximadamente 1 año, combinando sesiones de planificación y seguimiento con el equipo técnico de Tracasa y el área de gestión de proyectos tecnológicos (PyT). Para ello, se adoptará una metodología ágil de trabajo, que permitirá iterar de manera continua sobre los avances, incorporar mejoras basadas en el feedback de los usuarios y priorizar las funcionalidades más relevantes para cada perfil.

En definitiva, este proyecto representa una apuesta firme por la innovación y la transformación digital en el ámbito de la formación corporativa. La implementación del Ecosistema de Aprendizaje permitirá a Tracasa Global y Tracasa Instrumental fortalecer su modelo de gestión del talento, fomentar el aprendizaje continuo y consolidar una cultura organizativa orientada al crecimiento y a la excelencia.

Análisis de requisitos

Antes de abordar el desarrollo y la estructura técnica de la aplicación, resulta fundamental realizar un análisis de los requisitos funcionales y no funcionales que definirán el alcance del Ecosistema de Aprendizaje. Este análisis tiene como objetivo identificar las necesidades específicas de la organización, los usuarios involucrados y las funcionalidades esenciales que el sistema deberá cubrir para cumplir con los objetivos planteados.

2.1. Requisitos funcionales

Los requisitos funcionales describen las acciones y procesos que la aplicación deberá permitir realizar a sus usuarios. A partir de las reuniones iniciales con Tracasa Global y Tracasa Instrumental, y del estudio de los flujos actuales de gestión de la formación, se han definido las siguientes funcionalidades principales:

Gestión de acciones formativas: permitir la creación, edición, eliminación y clonación de acciones formativas, distinguiendo entre formaciones planificadas y extraordinarias.

Gestión de solicitudes: permitir que los empleados soliciten nuevas formaciones o su participación en acciones ya existentes, con validación por parte de responsables, directores y el área de PyT (Proyectos y Tecnología).

Flujo de aprobación y notificaciones: automatizar el envío de correos electrónicos para informar a los distintos roles del estado de las solicitudes, aprobaciones y valoraciones pendientes.

Gestión de docentes y asistentes: permitir asociar docentes internos o externos a una formación, así como gestionar los asistentes y sus convocatorias.

Encuestas y valoraciones: generar encuestas de eficacia formativa y registrar valoraciones asociadas a cada acción formativa.

Planes de formación anual: posibilitar la creación, edición y seguimiento de planes de formación, con la asignación de propuestas por departamento y la gestión de permisos para cada usuario.

Integración con plataformas externas: permitir el acceso directo a la plataforma de aprendizaje GoodHabit y a la plataforma de ciberseguridad corporativa desde la propia aplicación.

Histórico y trazabilidad: registrar los cambios y actualizaciones realizados sobre las acciones formativas para mantener un seguimiento completo de la evolución de cada proceso.

2.2. Requisitos no funcionales

Además de las funcionalidades descritas, la aplicación deberá cumplir con una serie de condiciones relacionadas con su rendimiento, seguridad y usabilidad:

Usabilidad: la interfaz debe ser intuitiva, clara y accesible para usuarios con distintos niveles de experiencia técnica.

Seguridad: el acceso a la aplicación deberá estar autenticado y controlado mediante un sistema de roles (solicitante, responsable, director, PyT o administrador), garantizando la protección de los datos personales y formativos.

Rendimiento: el sistema debe ser capaz de gestionar múltiples solicitudes y consultas simultáneas sin afectar la velocidad de respuesta.

Escalabilidad: la arquitectura debe permitir la incorporación de nuevas funcionalidades o módulos sin afectar el funcionamiento existente.

Mantenibilidad: el código debe estar bien estructurado y documentado, siguiendo principios de modularidad y reutilización.

Compatibilidad: la aplicación debe ser accesible desde los navegadores corporativos más utilizados y adaptarse correctamente a diferentes resoluciones de pantalla.

2.3. Roles

Durante la fase de análisis se identificaron los diferentes tipos de usuarios que interactuarán con el sistema, cada uno con permisos y responsabilidades específicas:

Solicitante: puede solicitar nuevas acciones formativas o inscribirse en formaciones ya creadas.

Responsable de área: recibe y valida las solicitudes de los empleados de su equipo.

Director de departamento: revisa y aprueba las solicitudes validadas por los responsables de las áreas que forman parte de su departamento.

PyT/PyV (Proyectos y Tecnología): gestiona las solicitudes aprobadas, tramita las formaciones y supervisa el cumplimiento de los procesos.

Administrador del sistema: mantiene la aplicación, gestiona usuarios y configura los parámetros globales de funcionamiento.

Tecnologías y herramientas utilizadas

La elección de las tecnologías empleadas en el desarrollo del Ecosistema de Aprendizaje responde a la necesidad de construir una solución robusta, segura y con capacidad de crecer a largo plazo dentro del entorno corporativo de Tracasa Global y Tracasa Instrumental. En este sentido, se van a seleccionar herramientas consolidadas en el sector y ampliamente utilizadas en el desarrollo de aplicaciones empresariales, garantizando así un mantenimiento eficiente, una curva de aprendizaje asumible y una larga vida útil del sistema.

3.1. Lenguaje de programación y ecosistema .NET

Antes de profundizar en las tecnologías utilizadas en el desarrollo del backend, resulta pertinente introducir brevemente el concepto de **framework**, ya que constituye la base sobre la que se construye gran parte de la solución implementada.

Un *framework* [10] puede entenderse como un conjunto de herramientas, componentes y estructuras previamente definidas que facilitan la creación de aplicaciones de manera más ágil y eficiente. Su propósito es evitar que los desarrolladores deban partir desde cero en cada proyecto, proporcionando elementos reutilizables y mecanismos estandarizados para resolver problemas comunes. Gracias a ello, se reduce el tiempo de desarrollo, se mejora la calidad del software y se garantiza una mayor consistencia en la implementación.

En este contexto, la plataforma **.NET** constituye uno de los frameworks más completos y ampliamente adoptados en el ámbito empresarial. Se trata de una plataforma de código abierto y multiplataforma diseñada para compilar y ejecutar aplicaciones de todo tipo, desde soluciones web hasta servicios distribuidos o herramientas de escritorio. .NET destaca por su alto rendimiento, su sistema avanzado de ejecución y su amplio conjunto de bibliotecas optimizadas, que permiten trabajar de forma segura, productiva y escalable. Entre sus características clave se encuentran la administración automática de memoria, la seguridad de tipos, la protección de la memoria y la existencia de un extenso ecosistema de herramientas y marcos de trabajo especializados.

Sobre esta plataforma se apoya el lenguaje **C#**, utilizado como lenguaje principal en el desarrollo del Ecosistema de Aprendizaje. C# es un lenguaje moderno, fuertemente

tipado y orientado a objetos, diseñado para ofrecer productividad y claridad en el código, además de funciones avanzadas como la concurrencia integrada, el manejo automático de recursos y una sintaxis elegante fácil de mantener a largo plazo. Su interoperabilidad con otros lenguajes del ecosistema .NET y su alta fiabilidad lo han convertido en la opción predilecta dentro del entorno profesional cuando se requieren aplicaciones robustas, seguras y de gran rendimiento.

La elección conjunta de C# y .NET proporciona una base tecnológica sólida para el proyecto, especialmente en un entorno corporativo donde la seguridad, la escalabilidad y la mantenibilidad son requisitos fundamentales.

Entre los motivos que justifican el uso de esta tecnología destacan los siguientes:

- **Madurez y estabilidad:** .NET es una plataforma ampliamente consolidada y en constante evolución, lo que garantiza fiabilidad incluso en proyectos de larga duración.
- **Rendimiento elevado:** el entorno de ejecución de .NET ofrece tiempos de respuesta competitivos, ideales para sistemas corporativos que gestionan un gran volumen de solicitudes simultáneas.
- **Seguridad integrada:** la plataforma incorpora mecanismos nativos de protección, validación y control de acceso, esenciales en aplicaciones que manejan datos sensibles como los relativos a formación y gestión interna.
- **Productividad y facilidad de mantenimiento:** gracias al uso de C# y al acceso a bibliotecas optimizadas, el desarrollo es más ágil y el código más fácil de mantener.
- **Interoperabilidad y ecosistema amplio:** .NET permite integrar la solución con otros servicios y tecnologías corporativas, reduciendo riesgos y facilitando futuras ampliaciones.

En conjunto, el uso del ecosistema .NET y del lenguaje C# constituye una opción altamente adecuada para el desarrollo del Ecosistema de Aprendizaje, proporcionando un equilibrio óptimo entre rendimiento, seguridad, mantenibilidad y capacidad de evolución tecnológica.

3.2. Interfaz de usuario

La capa de presentación del Ecosistema de Aprendizaje se ha desarrollado combinando **TypeScript**, el motor de componentes **Blazor** y estilos definidos mediante **CSS global**. Este conjunto tecnológico permite construir interfaces modernas, dinámicas y fáciles de mantener, aprovechando tanto el ecosistema web tradicional como las capacidades integradas de .NET para el desarrollo de aplicaciones interactivas.

- **TypeScript** aporta tipado estático, una organización más clara del código y la detección temprana de errores durante el desarrollo. Esto reduce fallos en ejecución y mejora significativamente la mantenibilidad de la interfaz.

- **Blazor** facilita la creación de componentes interactivos empleando código familiar para el equipo de desarrollo, permitiendo integrar lógica de presentación de forma estructurada sin necesidad de depender exclusivamente de JavaScript. Su capacidad para trabajar con datos y eventos de manera fluida mejora la consistencia de la experiencia de usuario.
- **CSS** se utiliza para definir estilos unificados en toda la aplicación, garantizando coherencia visual, facilidad de personalización y una apariencia profesional adaptada al entorno corporativo.

Este conjunto de tecnologías permite construir una interfaz intuitiva, ligera y adaptada a los estándares actuales de usabilidad, asegurando que la experiencia de usuario resulte coherente para todos los perfiles que interactúan con la aplicación.

3.3. Gestión de datos: Entity Framework y SQL Server

El acceso a datos se ha implementado mediante **Entity Framework**, un *ORM* que simplifica la comunicación con la base de datos. Sus principales ventajas incluyen:

- **Reducción del código repetitivo:** permite abstraer consultas comunes sin necesidad de escribir SQL manual, disminuyendo la probabilidad de errores.
- **Sincronización continua del modelo de datos:** gracias al sistema de migraciones, es posible actualizar la base de datos de forma segura a medida que evoluciona el proyecto.
- **Facilidad de mantenimiento:** su integración natural con C# permite trabajar de forma coherente con las entidades del sistema.

Como motor de almacenamiento se ha utilizado **SQL Server** [11], una tecnología ampliamente extendida en entornos corporativos y que presenta ventajas significativas:

- **Potentes capacidades de administración:** SQL Server facilita la gestión de volúmenes grandes de información, el control de transacciones y la seguridad.
- **Escalabilidad:** permite afrontar un crecimiento progresivo del sistema sin necesidad de realizar cambios disruptivos en la infraestructura.
- **Integración con herramientas empresariales existentes:** lo que agiliza los procesos de despliegue y mantenimiento.

Estas características hacen que la combinación de Entity Framework con SQL Server sea particularmente adecuada para aplicaciones internas de gestión, donde la consistencia, seguridad y trazabilidad de los datos son elementos clave.

3.4. Sistema de notificaciones y comunicaciones

La plataforma incorpora un sistema automatizado de **envío de correos electrónicos**, fundamental para coordinar las aprobaciones, convocatorias, valoraciones y avisos entre usuarios. Su importancia radica en que:

- Reduce los tiempos de gestión.
- Evita pérdidas de información entre departamentos.
- Asegura una comunicación constante y centralizada.

La integración del sistema de notificaciones dentro de la lógica de la aplicación permite que el usuario reciba información en tiempo real, lo que incrementa la eficiencia de los flujos de trabajo relacionados con la formación.

En conjunto, todas estas tecnologías permiten construir una aplicación robusta, segura, fácil de mantener y preparada para evolucionar en el futuro. Su uso está alineado con las tendencias actuales en desarrollo de software corporativo, ofreciendo garantías de fiabilidad y sostenibilidad a largo plazo dentro del entorno empresarial de Tracasa.

Estructura del proyecto

El proyecto se estructura siguiendo una arquitectura por capas¹, lo que permite organizar el código y separar las responsabilidades de forma clara y eficiente. Cada capa cumple un papel distinto. La presentación se encarga de la interfaz y la interacción con el usuario, la aplicación contiene la lógica de negocio, la infraestructura gestiona el acceso a recursos externos como la base de datos y la capa de dominio define las reglas y entidades principales del modelo. La elección de esta arquitectura se debe a que proporciona una base sólida y escalable para el desarrollo, favorece la mantenibilidad del sistema y facilita la incorporación de nuevas funcionalidades sin afectar al resto de componentes. Además, permite una mayor reutilización del código y mejora la calidad del software al fomentar un diseño más limpio, modular y fácilmente testeable.

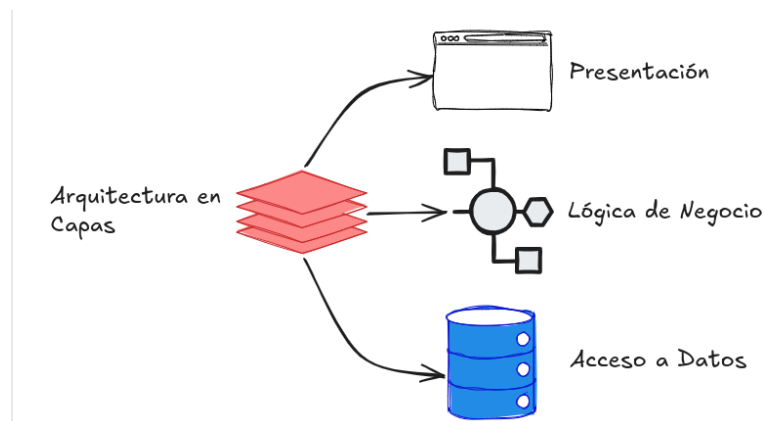


Figura 4.1: Diagrama de una arquitectura por capas.

¹<https://reactiveprogramming.io/blog/es/estilos-arquitectonicos/capas>

4.1. Presentación

La capa de presentación constituye el nivel superior de la arquitectura y se encarga de gestionar la interacción directa con el usuario. En esta capa se encuentra el proyecto Interfaz de Usuario Web (*WebUI*), que corresponde a la aplicación web visible para los usuarios finales. Dentro de *WebUI* se definen las vistas, es decir, los elementos visuales que conforman la interfaz, junto con los estilos *CSS* que determinan su apariencia.

Asimismo, en esta capa se desarrollan las clases *TypeScript*, responsables de manejar los distintos eventos del usuario y de definir las acciones que se ejecutan como respuesta a dichos eventos. Los controladores también forman parte de esta capa y su función principal es invocar a las clases de niveles inferiores para realizar operaciones sobre la base de datos. Estas clases, situadas en capas más profundas, se explicarán más adelante. Se incluyen también los *ViewModels*, que son modelos diseñados específicamente para las vistas. Estos modelos facilitan la organización y presentación de los datos de forma estructurada y coherente, permitiendo que la información se muestre correctamente en la interfaz web. En esencia, actúan como una plantilla que define tanto la apariencia como el comportamiento de los datos dentro de la interfaz de usuario.

Un ejemplo ilustrativo puede observarse en el Código 4.1, donde se presentan varios métodos pertenecientes al controlador de la clase **Formación**. El primer método, que se invoca mediante una petición *POST*, devuelve un conjunto de formaciones almacenadas en la base de datos, permitiendo además aplicar filtros de búsqueda a través del parámetro **filtros**, en caso de que ocurra algún error y no se devuelven las formaciones se muestra un mensaje al usuario en pantalla. En cambio, el segundo método se ejecuta mediante una petición *GET*, que solicita un recurso específico del servidor y recupera una formación concreta identificada por su ID pasado como parámetro en la línea 27 (**FormacionID**), aquí también se hace una validación para comprobar el tipo de respuesta ya sea un error para informar al usuario, un aviso de que esa persona no está autorizada a ver los detalles de esa formación ó hay éxito que en ese caso se genera el *ViewModel* correspondiente y se le pasa los datos de la formación para poder mostrarlos por pantalla al usuario.

Además de los métodos de consulta, el controlador incluye también operaciones para crear, modificar y eliminar registros de formaciones. Otra nota importante aquí es cuando ponemos a los métodos [**TienePermisosFilter(EAutorizacionPermisos.SOLICITANTE)**] , esto nos valida el rol del usuario el cual llama al método. En este caso definimos que solamente aquellos que mínimo el rol de *Solicitante* podrán acceder a esté método, aunque más tarde explicaremos con más detalle el tema de los roles y como funciona la validación de permisos.

```
1 [HttpPost("[action]")]
2 [TienePermisosFilter(EAutorizacionPermisos.SOLICITANTE)]
3 public async Task<IActionResult>
4     GetListadoFormaciones([DataSourceRequest]
5         DataSourceRequest request, FormacionFiltrosDTO filtros)
```



```
        filtrado");
6
7    var formacionesRespuesta = await Mediator.Send(new
        GetAllFormacionesQuery(filtros));
8
9    if (formacionesRespuesta.Tipo == TipoRespuesta.Error)
10    {
11        ModelState.AddModelError("ListadoGridError", "Ha
            ocurrido un error al obtener el listado de
            formaciones.");
12    }
13
14    return Json(formacionesRespuesta.Data.ToDataSourceResult
15        (request, ModelState));
16 }
17
18 [HttpGet("[action]/{FormacionID}")]
19 [TienePermisosFilter(EAutorizacionPermisos.SOLICITANTE)]
20 public async Task<IActionResult> Detalle(int formacionID)
21 {
22     logger.LogInformation($"Pantalla de detalle de la
        formacion: {formacionID}");
23
24     var formacion = await Mediator.Send(new
        GetSingleFormacionQuery(formacionID));
25
26     if (formacion.Tipo == TipoRespuesta.Error)
27     {
28         TempData["NotificacionMensaje"] =
29             NotificacionMensaje.FormacionObtenerError
30                 .ToString();
31         return RedirectToAction("Index");
32     }
33     else if (formacion.Tipo == TipoRespuesta.NoAutorizado)
34     {
35         return RedirectToAction("HttpStatusCodeHandler",
36             "Error", new { statusCode = 403 });
37     }
38
39     var viewModel = new FormacionViewModel();
40     viewModel.FormacionGeneral =
        viewModel.FormacionGeneral.GetViewModel
```

```
40         (formacion.Data);  
41  
42         return View(viewModel);  
43     }
```

Listing 4.1: Controlador del proyecto WebUI con métodos de selección de formaciones almacenados en base de datos.

4.2. Aplicación

La capa de aplicación representa el nivel en el que se concentra la lógica de negocio del sistema. En este nivel se implementa el principio de *Command Query Responsibility Segregation* (CQRS), mediante el cual cada proceso o acción empresarial se modela como un comando o una consulta independiente. Este patrón arquitectónico permite diferenciar las operaciones que modifican el estado del sistema (comandos) de aquellas que únicamente recuperan información (consultas), promoviendo así una estructura más ordenada, clara y escalable.

Dentro de esta capa también se definen las clases conocidas como *Data Transfer Objects* (DTO). Estas entidades sirven como vehículos de intercambio de información entre las diferentes capas, evitando que los modelos internos se expongan directamente. Su propósito es agrupar y transportar los datos necesarios para ejecutar una acción concreta, garantizando una comunicación sencilla, coherente y desacoplada entre componentes de la aplicación.

Finalmente, las operaciones relacionadas con los comandos y las consultas, como se muestra en el Código 4.2, se declaran mediante interfaces localizadas en esta misma capa, mientras que su implementación específica se desarrolla en la capa inferior.

```
1 public interface IFormacionesDataService  
2 {  
3     Task<List<FormacionListDTO>>  
4         SelectAll(FormacionFiltrosDTO filtros,  
5             CancellationToken cancellationToken);  
6     Task<List<int>> SelectAreas(List<string> areas,  
7         CancellationToken cancellationToken);  
8     Task<bool> PerteneceFormacionArea(int formacionId,  
9         List<string> areas, CancellationToken  
10         cancellationToken);  
11     Task<FormacionDTO> SelectSingle(int formacionID,  
12         CancellationToken cancellationToken);  
13     Task<bool> ExisteCodigo(string codigo, CancellationToken  
14         cancellationToken);  
15     Task<bool> ComprobarValoracionCompletada(int formacionID);  
16     Task<FormacionDTO> Insert(FormacionDTO formacionDTO,  
17         CancellationToken cancellationToken);
```

```
10     Task<FormacionDTO> Update(FormacionDTO formacionDTO ,  
    CancellationTokens cancellationTokens);  
11     Task<bool> Delete(int formacionID, CancellationTokens  
    cancellationTokens);  
12 }
```

Listing 4.2: Interfaz de formaciones con métodos para llevar a cabo las operaciones sobre la base de datos.

4.3. Infraestructura

La capa de infraestructura constituye el nivel responsable de la interacción con los recursos externos y de proporcionar los servicios necesarios para que las demás capas puedan operar correctamente. En esta capa se ubican las clases encargadas de acceder a sistemas externos, tales como bases de datos, sistemas de archivos, servicios web o cualquier otro recurso externo que el sistema requiera.

Las clases que pertenecen a este nivel implementan las interfaces definidas en la capa de aplicación, cumpliendo así el principio de inversión de dependencias y garantizando una arquitectura más modular y desacoplada. Gracias a este enfoque, el resto del sistema no necesita conocer los detalles técnicos de cómo se realizan las operaciones externas, sino únicamente la interfaz que las define, lo que facilita la sustitución o actualización de componentes sin afectar al funcionamiento general.

Otro elemento importante presente en esta capa son los procesos de migración, que hacen referencia al conjunto de procedimientos destinados a trasladar o actualizar la estructura y los datos de una base de datos. Las migraciones permiten mantener la coherencia entre el modelo de datos y el código fuente del proyecto, especialmente durante las fases de desarrollo o actualización del sistema. Mediante estas operaciones se pueden aplicar cambios de esquema, incorporar nuevas tablas o modificar relaciones existentes, asegurando la integridad de la información y la compatibilidad entre versiones.

4.4. Dominio

La capa de dominio constituye el núcleo esencial del sistema, ya que en ella se define la lógica de negocio que modela los procesos y reglas fundamentales de la aplicación. En este nivel se ubican las entidades, que representan los elementos principales del modelo de negocio y encapsulan tanto sus atributos como el comportamiento asociado.

El propósito de esta capa es reflejar con fidelidad la realidad del problema que la aplicación pretende resolver, garantizando la coherencia interna y la integridad de los datos mediante las operaciones que se realizan sobre dichas entidades.

Una característica clave del dominio es su independencia respecto a la infraestructura y a los elementos externos, como bases de datos o servicios. Esta separación asegura que las reglas de negocio permanezcan estables y no se vean afectadas por los cambios tecnológicos, facilitando así la evolución y el mantenimiento del sistema a largo plazo.

Desarrollo del *Ecosistema de Aprendizaje*

La aplicación *Ecosistema de Aprendizaje* es un entorno digital que nos va a permitir, desde la organización, gestionar el aprendizaje y el desarrollo profesional de una manera más ágil y dinámica. A través de esta aplicación, se busca dar respuesta a la necesidad de disponer de un entorno digital unificado que permita a Tracasa Global y Tracasa Instrumental gestionar de forma eficiente los procesos de formación y desarrollo profesional de sus empleados. Este proyecto se enmarca en una estrategia de transformación digital más amplia, orientada a optimizar la gestión del talento y fomentar el aprendizaje continuo dentro de la organización. Desde el punto de vista funcional, la aplicación ofrecerá una interfaz intuitiva y accesible, compuesta por diferentes pantallas y formularios que permitirán a los usuarios interactuar con el sistema de manera sencilla. La herramienta persigue no solo digitalizar los procesos de solicitud y seguimiento de la formación, sino también mejorar la trazabilidad y la comunicación entre empleados, *managers* y responsables del área de proyectos tecnológicos. De este modo, se potencia una gestión más ágil, transparente y alineada con los objetivos estratégicos de la empresa.

La aplicación nos ofrece varias funcionalidades:

1. Posibilidad de cada persona pueda solicitar formación, que será validada por el *manager* y *PyT*.
2. Posibilidad de conocer qué acciones formativas se van a hacer en la empresa y solicitar incluirse en ella. También será validado por el *manager* y *PyT*.
3. Acceder a una plataforma de aprendizaje donde cada persona podrá desarrollarse en aquellas materias que puedan resultar de interés. Aquellas acciones que sean por interés personal y no estén relacionadas con el desempeño de su puesto, se realizarán fuera de jornada laboral.

Cuando accedemos a la aplicación, visualizamos la pantalla *HOME* que sale en la Figura 5.1, donde podremos elegir la acción que vayamos a realizar. En la Figura 5.1 aparecen marcadas con círculo rojo las acciones que hay.

1. Acceso a la plataforma de aprendizaje *GoodHabit*.
2. Acceso a la plataforma de ciberseguridad.
3. Solicitar una nueva acción formativa.
4. Visualizar las acciones previo a su comienzo y solicitar unirse a ella.

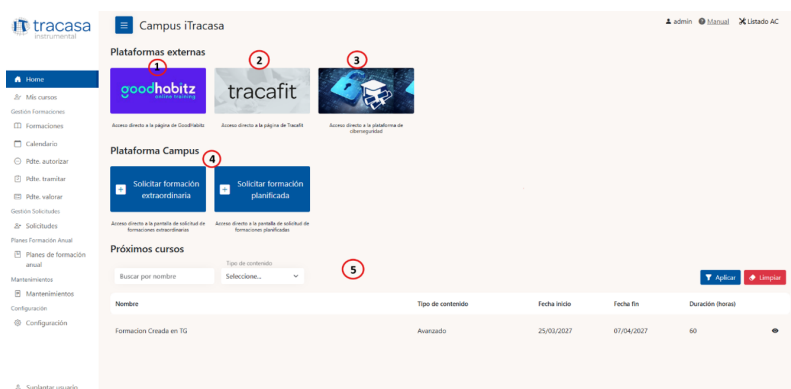


Figura 5.1: Pantalla home con los accesos directos.

5.1. Acceso a plataforma de aprendizaje

Para acceder a la plataforma de aprendizaje, que vemos en la Figura 5.2, debemos posicionarnos en la pantalla Home y seleccionar en GoodHabit. Al acceder a la plataforma, por defecto, y sólo la primera vez, aparecerá un mensaje para aceptar la política de privacidad y acuerdo de usuario.

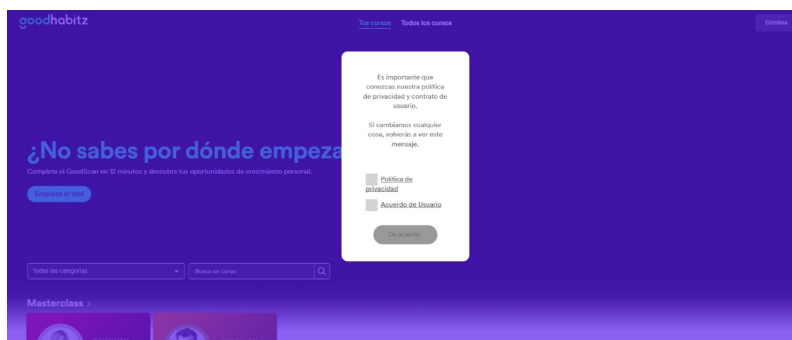


Figura 5.2: Pantalla plataforma de aprendizaje.

A continuación, encontraremos, en la parte superior de la pantalla, dos opciones a las que podremos acceder:

- Tus cursos: donde podremos encontrar aquellas acciones formativas que cada usuario haya realizado, esté realizando en ese momento o, también podemos encontrar aquellas formaciones que la empresa haya sugerido realizar
- Todos los cursos: podremos ver el catálogo de acciones formativas a las que podemos acceder y cursar.

5.2. Acceso directo a la plataforma de ciberseguridad

Acceso directo a contenidos para mejorar las capacidades en ciberseguridad tal y como se observa en la Figura 5.3

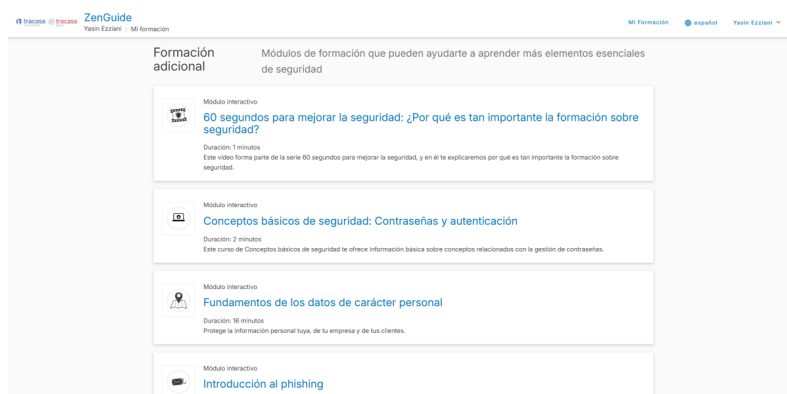


Figura 5.3: Plataforma de ciberseguridad.

5.3. Solicitar nueva acción formativa

Se proporcionan dos accesos para poder solicitar una formación. Podemos crear una formación extraordinaria o planificada. La diferencia que hay es que en el caso de la planificada debemos seleccionar una propuesta de plan de formación (que explicaremos más tarde) que asignaríamos a la formación que vamos a crear y para el caso de extraordinaria no se le asigna.

Al acceder a la opción de *Solicitar formación extraordinaria* encontraremos la pantalla mostrada en la Figura 5.4:

Figura 5.4: Cuestionario INTAKE para solicitar formación.

Un cuestionario de 10 preguntas para recoger información sobre la necesidad formativa u oportunidad de mejora. Este cuestionario solo aparece para el caso de formación extraordinaria.

Nos encontramos con otra pestaña *Datos generales* que sí aparece para ambos tipos de formación, que nos muestra la pantalla de la Figura 5.5:

Figura 5.5: Formulario de datos generales a rellenar.

Cumplimentar los datos de la segunda pantalla:

Nombre: Nombre de la acción formativa.

Área: Área a la que pertenece el creador de la solicitud.

Aprobador/a: Responsable directo del/los asistentes. En caso de asistir personal de varias áreas, identificar como aprobador al responsable del área de la que asistan más personas.

Validador/a: Director de departamento al que pertenecen los asistentes. En caso de asistir

personal de varios departamentos, identificar como aprobador al director del departamento del que asistan más personas.

Valorador/a: Encargado/a de valorar la formación. Puede haber más de un/a valorador/a para cada formación. Por defecto se asigna como valorador/a el aprobador/a.

Planificación: Ubicarla en el trimestre en el que se imparte la acción.

Modalidad de la formación: Modalidad en la que se imparte (Online, Streaming, presencial, mixta...).

Tipo de formación: Si la formación es interna (el docente es de plantilla), externa (el docente es una empresa/persona ajena a la empresa), mixta (se combinan ambas modalidades).

Medio pedagógico: Comunidades en prácticas (aprendizaje en comunidad o social), aprendizaje experiencial (Learning by doing), aprendizaje tradicional (formación tradicional).

Tipo de contenido: Si se trata de conocimientos técnicos, Soft Skills, Idiomas, Agile...

Lugar: Dónde tiene lugar la acción formativa.

Tipo de coste: Coste (si la acción implica un coste en la matriculación) o gratuita (no supone coste en la inscripción).

Código: Aparece cumplimentada por defecto. No rellenable.

Unidades: Número de personas para asistir.

Importe: En caso de que suponga coste, indicar cuantía por persona sin IVA.

Total: Cuantía total sin IVA. No rellenable.

Fecha Inicio: Fecha en la que tiene lugar el inicio de la acción formativa.

Fecha Fin: Fecha en la que finaliza la acción formativa.

Horas de formación: Número de horas que dura la acción formativa.

Sala: La sala donde se impartirá la acción formativa.

Días aviso valoración: Aparece cumplimentada por defecto. No rellenable.

Tipo de plan de formación: Planificado o Extraordinario. Según qué tipo de solicitud hemos solicitado tendrá un valor u otro, con la posibilidad de poder cambiarlo.

Plan de formación anual: Plan para formar en el año actual. En el caso de solicitud extraordinaria este campo estará deshabilitado y, para el caso de planificada, tendrá como valor la propuesta del plan de formación seleccionada, también con la opción de poder cambiarla.

Mostrar en la home: Opción para mostrar o no en la pestaña home.

Es bonificable: Opción para indicar si es bonificable o no la acción formativa.

Contenido y/o programa: Espacio destinado a alojar el link del programa de contenidos, en caso de disponer de él, o adjuntarlo en caso de tenerlo como documento.

Temario: Espacio destinado a adjuntar el temario en caso de tenerlo como documento.

Observaciones: Observaciones opcionales para indicar en la acción formativa.

Si hemos seleccionado la opción de *Solicitar formación planificada* nos aparece lo que vemos en la Figura 5.6

Código	Objetivo	Descripción	Tipo de contenido	Modo pedagógico	Planificación	Fecha estimada de inicio	Duración
PF27-002	sdf	sdf	Avanzado	Aprendizaje experiencial	T2	24/09/2023	
PF30-001	sdf	sdf	Avanzado	Aprendizaje experiencial	T2	24/09/2023	

Figura 5.6: Ventana para seleccionar propuesta a asignar a la formación.

Es un listado de propuestas aceptadas de los planes de formación que estén aprobados y activos. Se selecciona una propuesta y haciendo clic en el botón *Aceptar* nos carga solo la pestaña de *Datos generales* con los campos *Tipo de plan de formación* y *Plan de formación* automáticamente rellenos.

Una vez cumplimentada la pantalla de Datos generales, debemos guardar y continuar la creación de la solicitud y a continuación aparecerá más pestañas en la parte superior, tal y como se muestra en la Figura 5.7



Figura 5.7: Pestañas que cargan después de crear una formación.

La pestaña *Docencia* es el espacio para crear el/los docentes e introducir toda la información relativa a la docencia como vemos en la Figura 5.8:

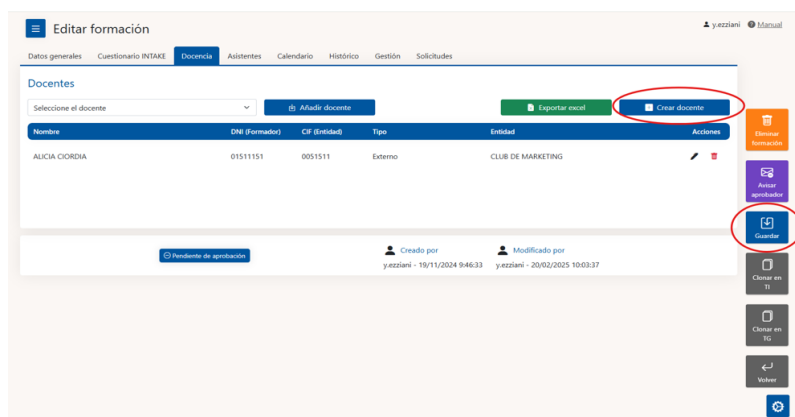


Figura 5.8: Pestaña de los docentes de la formación.

Pulsamos Crear docente y aparece una línea para introducir los datos correspondientes (Nombre, DNI, CIF, Tipo de docente, si es ajeno a la empresa o interno, entidad a la que pertenece. . .).

Cuando la acción formativa está pendiente de tramitación se permite seleccionar asistentes y nos aparece el botón de enviar convocatoria y lo podemos ver en la Figura 5.9:



Figura 5.9: Seleccionar asistentes para enviarles envío de convocatoria.

Al hacer clic se envía un correo a los asistentes seleccionados. En este correo se encuentra un archivo adjunto que contiene los eventos creados en la pestaña Calendario (que veremos más adelante) para agregarlos automáticamente en el Outlook personal de cada asistente.

Por otro lado, en el cuerpo del correo aparece un enlace el cual nos dirige a una página donde se muestra el nombre de la acción formativa y podemos aceptar o rechazar la asistencia con la posibilidad de agregar comentario opcional. Se puede aceptar o rechazar solo una vez, por lo que si se intenta acceder de nuevo después de contestar salta acceso denegado.

Cuando la formación pasa a ser tramitada, automáticamente a todos los asistentes que están pendientes de responder o que no se les mandó correo antes se les envía el correo. Hay dos casos a considerar al enviar un correo a uno o varios asistentes:

1. Si el correo se envía el mismo día de la fecha de finalización de la acción formativa, la asistencia se rechaza automáticamente y se notifica a los asistentes por correo electrónico.

2. Si quedan menos de 10 días para la fecha de finalización de la acción formativa, la asistencia se acepta automáticamente y no se envía ningún correo a los asistentes.

Cuando la formación pasa a ser pendiente de valoración, al seleccionar los asistentes aparece un nuevo botón para enviar correo para encuesta de eficacia formativa y en el correo se manda una url a la pantalla que se ve en la Figura 5.10 para rellenar la encuesta.

Encuesta de eficacia formativa de la formación: ciberseguridad

1. Se ha alcanzado el objetivo de la formación. *

2. El contenido de la formación se ajusta a nuestras necesidades. *

3. Los asistentes han desarrollado nuevas habilidades. *

4. Se han puesto en práctica los conocimientos adquiridos en el curso. *

5. ¿Le gustaría hacer alguna observación? *

Scala
Totalmente en desacuerdo Algo en desacuerdo Algo de acuerdo Totalmente de acuerdo

1 2 3 4 5

Guardar

Figura 5.10: Página para rellenar encuesta de eficacia formativa.

No se puede rellenar más de una vez excepto si se vuelve a enviar al asistente de nuevo el correo como comentamos en convocatoria. En este estado se agrega una pestaña más que es Valoraciones como vemos en la Figura 5.11:

Editar formación

Datos generales **Valoraciones** Cuestionario INTAKE Docencia Asistentes Calendario Histórico Gestión Solicitudes

Valoración de la formación

Valoración del usuario - Yasin Ezziari - 03/04/2025 09:33

Valoración media total: 3.2

Encuestas de eficacia formativas

Encuesta 1

Encuesta 2

Valoración media total: 2.9

Scala
Totalmente en desacuerdo Algo en desacuerdo Algo de acuerdo Totalmente de acuerdo

1 2 3 4 5

Finalizada

Creado por: yezziari - 03/03/2025 9:32:47 Modificado por: yezziari - 04/03/2025 9:33:12

Eliminar formación Guardar Clonar en TI Clonar en TIC Volver

Figura 5.11: Pestaña de valoraciones de una formación.

Nos indica tanto las valoraciones de la formación, que explicaremos más tarde, como las encuestas realizadas, como vemos se muestra marcado en rojo la valoración media total de todas las encuestas. Si desplegamos alguna encuesta podemos ver la respuesta del asistente y la valoración total de dicha encuesta, tal y como se ve en la Figura 5.12.

Encuesta 1

1. Se ha alcanzado el objetivo de la formación. *
★★★★ 4 / 4

2. El contenido de la formación se ajusta a nuestras necesidades. *
★★★☆☆ 3 / 4

3. Los asistentes han desarrollado nuevas habilidades. *
★★★★ 4 / 4

4. Se han puesto en práctica los conocimientos adquiridos en el curso. *
★★★★ 4 / 4

Valoración total: 3,8

5. ¿Te gustaría hacer alguna observación? *
es bastante útil

Figura 5.12: Resultado de una encuesta realizada a una formación.

En la pestaña Calendario, que vemos en la Figura 5.13, se incluirán los datos relativos a días y horas que abarca la formación.

Editar formación

Datos generales | Cuestionario INTAKE | Docencia | Asistentes | **Calendario** | Histórico | Gestión | Solicitudes

Calendario - (21/01/2025 - 28/04/2025)

Hoy • lunes, 20 de enero de 2025 - domingo, 26 de enero de 2025

	20/01	21/01	22/01	23/01	24/01	25/01	26/01
7:00							
8:00							
9:00							
10:00				Formación			
11:00							
12:00							
13:00							
14:00							
15:00							
16:00							

● Pendiente de aprobación

Creado por: y.ezziani - 19/11/2024 9:46:33 | Modificado por: y.ezziani - 20/02/2025 10:03:37

Eliminar formación | Añadir aprobador | Guardar | Clonar en TI | Clonar en TG | Volver

Figura 5.13: Pestaña de calendario de una formación.

Es un sistema flexible que permite personalizar las jornadas y días. El funcionamiento es similar al de la herramienta *Outlook*.

En la pestaña Histórico, que vemos en la Figura 5.14, se guarda un historial de cambios de los estados de la acción formativa, cambios de valores de los campos de los datos generales (importe, horas, fecha inicio, fecha fin), cuando se añade/elimina un docente y cuando se añade/elimina un asistente.

Estado anterior	Estado posterior	Usuario	Fecha	Observaciones
Pendiente de validación	Pendiente de transición	duffado-est	20/01/2025	El validador ha validado la formación
Pendiente de aprobación	Pendiente de validación	duffado-est	20/01/2025	El aprobador ha aprobado la formación

Campo	Valor anterior	Valor posterior	Usuario	Fecha
Eliminar asistente	Eva Cargio Sordo	duffado-est	15/01/2025	
Eliminar asistente	Blanca Guillen Ardanaz	duffado-est	15/01/2025	
Eliminar asistente	Argyle Ordoño Uribe	duffado-est	15/01/2025	
Eliminar asistente	Amabel Barandalla Maedon	duffado-est	15/01/2025	
Eliminar asistente	Amaya 1088 Ameta	duffado-est	15/01/2025	
Eliminar asistente	Mikel Azagra Iruan	duffado-est	15/01/2025	
Añadir asistente	Montse Huarte	mhuarte	15/01/2025	
Fecha fin	30/01/2028 23:59:59	05/02/2028 23:59:59	mhuarte	15/01/2025
Fecha inicio	25/11/2024 00:00:00	27/01/2025 00:00:00	mhuarte	15/01/2025

Figura 5.14: Pestaña con el histórico de cambios en una formación.

En la pestaña Gestión, que vemos en la Figura 5.15, se pueden enviar avisos por correo, rellenar opcionalmente el campo de cantidad bonificada y/o asignar pedidos, creando uno nuevo o asignando uno existente.

Enviar aviso a administración, comité o empresa externa

Enviar administración: mhuarte@itracas.es

Enviar comité: mhuarte@itracas.es

Enviar empresa externa: mhuarte@itracas.es

Avisos de valoración

Bonificación

Cantidad bonificada: 2.50 €

Aprovisionamiento

Crear nuevo pedido | Asignar pedido

Resumen de la formación

Estado: Pendiente de aprobación

Creado por: admin - 02/09/2025 13:57:44

Modificado por: admin - 01/10/2025 9:44:54

Figura 5.15: Pestaña de gestión de una formación.

Al querer asignar pedido nos muestra el listado de pedidos para seleccionar, tal y como nos aparece en la Figura 5.16:

Pedido	Nombre	Ref. Interna	Ref. Compra	Proveedor	Accs	Fecha
T1-2022-12-010	Servicios generales - Formación Aleroz - febrero 2023	Servicios generales - Formación Aleroz - febrero 2023	AD-DA10-79-2022	0027 - CLUB MARKETING	1014	2024-12-20
T1-EX9/2022-12-002	PLATAFORMA DE CONTENIDOS APRENDIZAJE	20221216		0122 - GOODHABITZ SPAIN, S.L.	1013	2024-12-16
T1-2022-12-009	Jurídico - curso inglés jurídico- Tales	Jurídico - curso inglés jurídico- Tales	AD-DA10-78-2022	0028 - EDITORIAL ARANZADI S.L.	1014	2024-12-12
T1-2022-12-006	Paula Moreno Comple v2	20221212	AD-255-2022	0001 - ACREEDORES VARIOS	1013	2024-12-12
T1-2022-11-027	Formación AZURE AZ-900	SIT 2022 AZURE	AD-DA10-75-2022	0001 - ACREEDORES VARIOS	5050	2024-11-30
	PROGRAMA ACTIVIDAD FISICA	20231123		0198 - WEEKUP BIENESTAR CORPORATIVO	1013	2024-11-23
T1-2022-12-001	gerencia - inglés - vaughan - enero - abril 2023	gerencia - inglés - vaughan - enero - abril 2023	AD-DA10-74-2022	0088 - VAUGHAN SYSTEMS, S.L.L.	1011	2024-11-21
T1-2021-11-015	formación power bi	AIN_2021	AD-DA10-01-2021	0035 - AIN ASOC. INDUSTRIA NARANJA	5050	2024-11-15

Figura 5.16: Listado de pedidos para asignar a una formación.

Se pueden filtrar y al asignar un pedido nos muestra la referencia interna y el nº del pedido vinculado con la posibilidad de tanto ver el pedido como desvincularlo.

Numero factura	Fecha factura	Fecha entrada	Importe	Proveedor	Ref. compra	Estado	Acción
2022-13182	30/11/2022	14/12/2022	810.00 €	0071 - CHAI	PPC-GRUPO-99-2022	Pagada	
2022-13146	31/10/2022	16/11/2022	407.30 €	0071 - CHAI	PPC-GRUPO-99-2022	Pendiente	Marcar Pagada

Figura 5.17: Pestaña de gestión de una formación con pedidos asignados.

Vemos en la Figura 5.17 que aparece también el listado de facturas del pedido vinculado. A parte de los datos de las facturas se nos proporciona el estado de cada factura (pagada o pendiente). Si una factura pendiente se ha cambiado de estado a pagada podemos actualizar las facturas de estado usando el botón *Marcar Pagada* que aparece en la columna de Acción para las facturas pendientes señalado con un cuadrado rojo.

En la pestaña Solicitudes, que vemos en la Figura 5.18, podemos ver todas las solicitudes de inscripción pendientes de aprobar

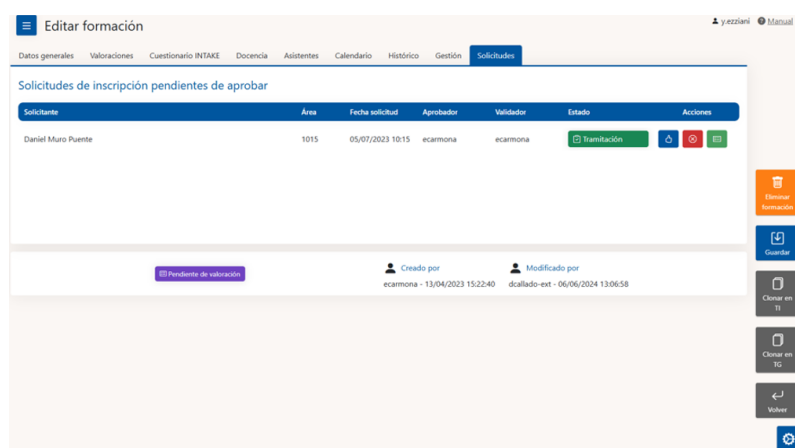


Figura 5.18: Pestaña de solicitudes de inscripción pendientes de aprobar.

Finalizamos la solicitud de acción formativa pulsando *Avisar al aprobador* y automáticamente llega un aviso a los responsables y *PyT* para validar y tramitar la acción, si es el caso. En caso de que, una vez solicitada una acción formativa, surja cualquier tipo de modificación, de procederá a hacer los cambios oportunos en la aplicación y, tras guardar esos cambios, debemos volver a *Avisar al aprobador*.

5.4. Solicitar incluirse en una acción formativa ya creada

En la pantalla Home, tal y como podemos observar en la Figura 5.19, podremos visualizar las acciones formativas que han sido solicitadas, aprobadas y tramitadas pero que aún no han dado comienzo, dando la posibilidad de que otras personas, que no han sido convocadas y que pueden estar interesadas en asistir, puedan solicitar su incorporación.

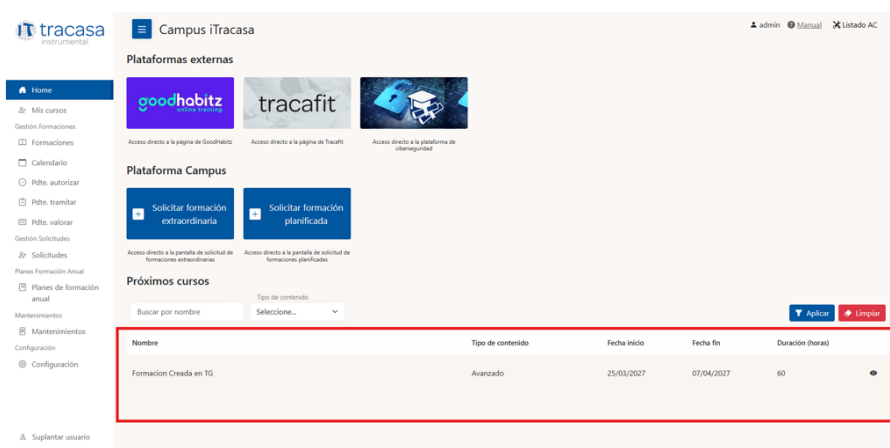


Figura 5.19: Pantalla de HOME con las formaciones ya tramitadas sin comenzar.

Pulsamos en cualquier parte de la línea de la formación en la que estamos interesados. Mostrará la información de la acción y para finalizar la solicitud de incorporación a la acción, debemos pulsar Solicitar inscripción y llegará a responsables y *PyT* para validar y tramitar la solicitud, si es el caso.

5.5. Clonación y valoraciones

Otra funcionalidad que tenemos es la posibilidad de clonar una acción formativa como se ve en la Figura 5.20.

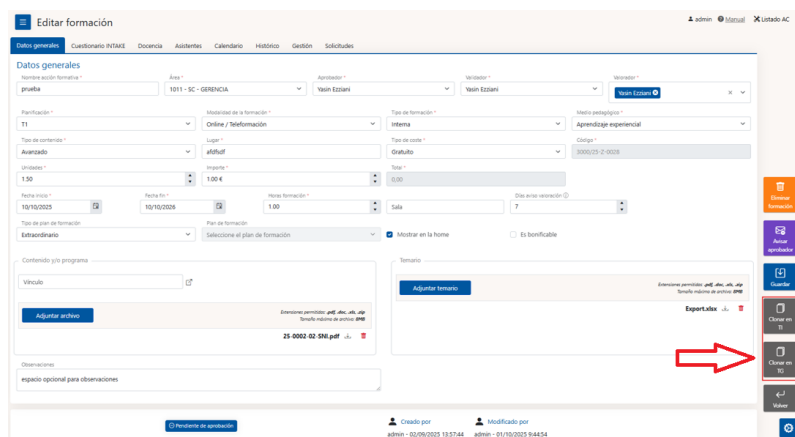


Figura 5.20: Botones de clonación de la formación.

Podemos clonar tanto en instrumental como en global. Al hacer clic se realiza un proceso similar a cuando solicitamos formación con algunos cambios:

1. La pestaña Cuestionario INTAKE se rellena por defecto con la información que había en la acción formativa que se desea clonar.
2. La pestaña Datos generales se rellena por defecto con la información que había en la acción formativa que desea clonar, excepto los campos Fecha inicio y Fecha fin. En el caso de clonar en la empresa contraria los campos que son selectores se cargan con los datos de dicha empresa.
3. Se muestra la pestaña Docencia con los docentes que tenía la acción formativa que se va a clonar sin la posibilidad de agregar o crear nuevos docentes.

5.5.1. Información para responsables y directores

En el momento en que un trabajador crea una acción formativa, la solicitud generada se envía automáticamente al **responsable de área** correspondiente para su revisión inicial. Este será el encargado de analizar la propuesta y valorar si la acción solicitada se ajusta a las necesidades del departamento o si, por el contrario, no resulta procedente.

- En caso de ser rechazada, el sistema genera de forma automática una notificación al creador de la solicitud, informándole tanto del estado de la misma como del motivo del rechazo, garantizando así una comunicación directa y transparente.
- Si la solicitud es aprobada, esta se remite al **director del departamento** (cuando corresponda) para que realice un segundo nivel de validación y análisis antes de su tramitación definitiva.

El proceso se repite en este punto, siguiendo la misma dinámica de aprobación y notificación entre los distintos niveles jerárquicos.

- En caso de que el director rechace la solicitud, el sistema volverá a generar automáticamente un aviso dirigido tanto al responsable de área como al creador de la acción formativa, informando del estado actualizado y del motivo de la decisión.
- Si, por el contrario, la solicitud es aprobada, se enviará un correo electrónico al área de *PyT*, que será la encargada de revisar, aprobar y tramitar la formación de manera definitiva.

De esta forma, el flujo de aprobación garantiza que todas las solicitudes sean revisadas por los distintos niveles de responsabilidad antes de ser ejecutadas, manteniendo un control exhaustivo sobre el proceso y asegurando la coherencia entre las necesidades individuales de los empleados y los objetivos estratégicos de la empresa. Podemos ver este flujo en el diagrama que aparece en la Figura 5.21

Además, los **responsables de área** y los **directores de departamento** recibirán de forma continua notificaciones sobre las solicitudes pendientes de revisión o cualquier otra tarea relacionada con su gestión. En caso de que estas notificaciones no se localicen en el buzón de correo, los usuarios podrán consultar directamente las tareas pendientes a través del menú lateral de la aplicación, donde se ha habilitado un acceso específico para este fin.

Al acceder al sistema, dispondrán de dos nuevas secciones que no están visibles para el resto de trabajadores:

- La opción *Pdte. Autorizar*, que muestra el listado de solicitudes que se encuentran pendientes de aprobación. Desde este apartado los responsables y directores podrán acceder al detalle de cada acción formativa, revisar su información completa y, en función de su criterio, aprobar o rechazar la solicitud directamente desde la interfaz.
- La opción *Pdte. Valorar*, destinada a la gestión de las valoraciones de la eficacia de la formación. En este apartado se mostrarán las formaciones finalizadas que están pendientes de valoración, y al acceder a una de ellas aparecerá una nueva pestaña en la parte superior con el formulario correspondiente para registrar la evaluación.

Gracias a este sistema de comunicación y control, la aplicación centraliza todos los procesos relacionados con la validación, aprobación y valoración de las acciones formativas, reduciendo los tiempos de gestión y evitando pérdidas de información entre los distintos niveles organizativos.

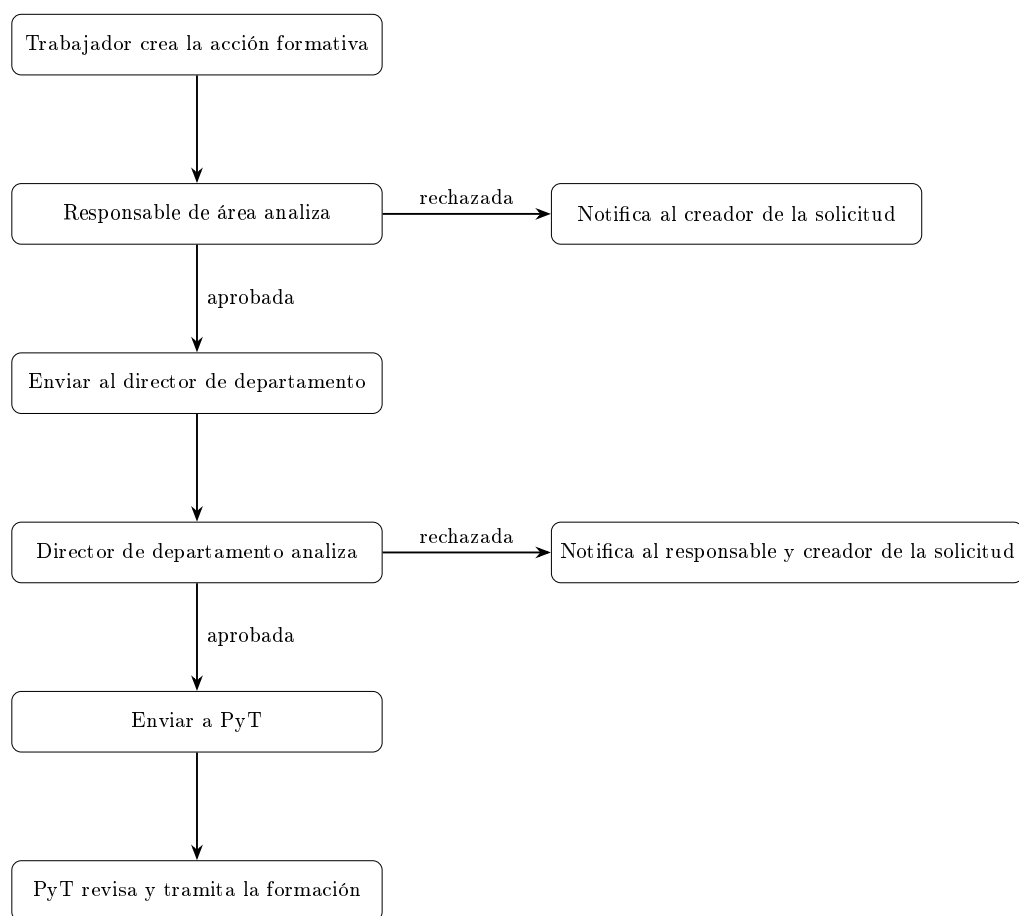


Figura 5.21: Flujo de aprobación y notificaciones de las acciones formativas, incluyendo rutas de rechazo.

5.6. Planes de formación y propuestas

En el menú lateral tenemos una sección de planes de formación anula, si hacemos clic nos aparece la siguiente Figura 5.22:

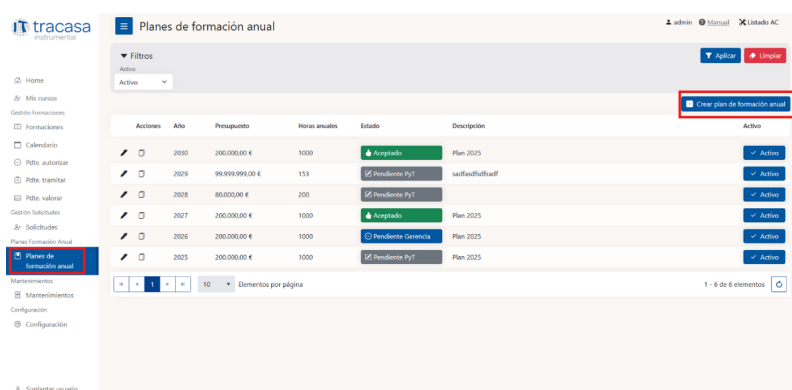


Figura 5.22: Pantalla con el listado de planes de formación anual.

Un listado de planes con la opción de poder filtrar los activos o inactivos. Para crear un plan de formación hacemos clic en el botón *Crear plan de formación anual* que nos muestra la página de la Figura 5.23:

Figura 5.23: Pantalla con el formulario para crear un plan de formación anual.

Al crear el plan aparecen más pestañas, tal y como podemos observar y la Figura 5.24 y en la parte inferior, se nos muestra el estado del plan junto al año del plan.

Edición del plan de formación anual

admin Manual Listado AC

Datos generales Departamentos Propuestas Log Administrador

Datos generales

Año 2032 Presupuesto 1,00 €

Presupuesto consumido 222,00 € Crédito FUNDAR 245.789,00 €

Crédito consumido 0,00 € Crédito pendiente 245.789,00 €

Horas anuales 2 Activo

Descripción

Estado del plan: ☒ Pendiente PyT - Año del plan: 2032

Figura 5.24: Pantalla de edición del plan de formación anual con todas las pestañas.

En la pestaña Departamentos vemos la siguiente Figura 5.25

Datos generales Departamentos Propuestas Log Administrador

Nombre	Total usuarios	Acciones
SIC	2	AS
Usuarios con permisos		
(admin)	<input checked="" type="checkbox"/>	
Yasin Ezziari (yezziari)	<input checked="" type="checkbox"/>	
SIC	1	AS
GTSI	0	AS
SIAESS	1	AS
SIT	0	AS
SIAD	0	AS
SIAA	0	AS
GEDISA	0	AS

Estado del plan: ☒ Pendiente PyT - Año del plan: 2032

Figura 5.25: Pantalla de departamentos del plan de formación anual.

Aquí, por cada departamento se podrá indicar que usuarios tendrán permisos para subir propuestas de formación para el departamento indicado. Para cada usuario hay una columna Enviar plan a *PyT* para poder indicar a que usuario dar los permisos, tal y como vemos en la Figura 5.26

Usuarios con permisos	Enviar plan a PyT
Yasin Ezziari (yezziari)	<input checked="" type="checkbox"/>
Xabier Dendarieta Sarries (xdendarieta)	<input type="checkbox"/>

Figura 5.26: Usuarios con permiso de un departamento del plan de formación anual.

En la pestaña de propuestas se encuentra el listado de las propuestas para el plan de formación como podemos ver en la Figura 5.27

Edición del plan de formación anual

admin

Mensual

Listado AC

Datos generales

Departamentos

Propuestas

Log

Filtros

Área

Seleccione...

Tipo contenido

Seleccione...

Medio Pedagógico

Seleccione...

Estado

Seleccione...

Combinar propuestas

Rechazar

Aprobar

Cargar excel

Exportar excel

Crear propuesta de formación

	Acciones	Código	Objetivo	Descripción	Tipo de contenido	Medio pedagógico
•		PF25-001	a	sadas	IDIOMAS	Aprendizaje tradicional
•		PF25-002	asdas	diadasdas	COMPETENCIAS TÉCNICAS	Aprendizaje tradicional
•		PF25-003	asd	sadas	IDIOMAS	Aprendizaje tradicional
•		PF25-004	asdasd	sadisa	PREU SALUD LABORAL	Aprendizaje tradicional
•		PF25-005	safsf	sadas	IDIOMAS	Aprendizaje tradicional
•		PF26-006	ggg	sadas	IDIOMAS	Aprendizaje tradicional
•		PF26-007	hhh	sadas	IDIOMAS	Aprendizaje tradicional

1

10

1

2

3

4

5

1 - 7 de 7 propuestas

Borrador

Pendiente PyT

Verificado

Aceptado

Rechazado

Estado del plan: Pendiente PyT

Alto del plan: 2025

Figura 5.27: Pantalla con el listado de las propuestas de plan de formación anual.

Debajo del listado nos aparecen los estados que podría tener una propuesta y cada estado con un color diferente. En el listado al inicio de cada propuesta aparece el color que representa su estado. En el caso de la Figura 5.27, hay 2 propuestas unificadas, 4 aceptadas y 1 rechazada.

Si nos fijamos en la parte superior hay una serie de botones:

Crear propuesta de formación: haciendo clic nos dirige a la pantalla, que vemos en la Figura 5.28, donde podemos crear una nueva propuesta rellenando el formulario y haciendo clic en *Guardar y continuar*.

Exportar excel: nos permite descargar un excel con las propuestas que hay en el listado.

Cargar excel: nos permite cargar un excel que tengamos con propuestas y agregarlos al plan automáticamente sin tener que crearlas manualmente.

Creación de propuesta de formación

Departamento *	Áreas *	Tipo de contenido *	Presupuesto previsto *
Seleccione los departamentos	Seleccione las áreas	Seleccione el tipo de contenido	
Medio pedagógico *	Planificación *	Fecha estimada inicio *	Duración (horas)
Seleccione el tipo de contenido	Seleccione la planificación	30/09/2025	
Horario *	Tipo de formación *	Modalidad *	Lugar de impartición *
Seleccione el horario	Seleccione el tipo de formación	Seleccione la modalidad	Seleccione el lugar de impartición
Criterio de selección de asistentes *	Ten *	Equipo *	
Seleccione el criterio de selección de asistentes	Seleccione el ten	Seleccione el equipo	Código
Asistentes *			
Seleccione los asistentes			

Objetivos *	Descripción *

Observaciones

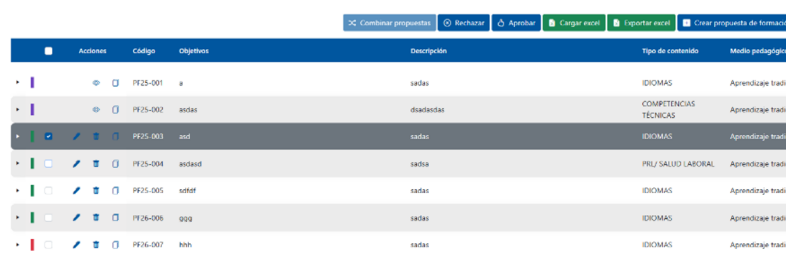
Figura 5.28: Pantalla con el formulario para crear una propuesta del plan de formaciones.

El resto de los botones aparecen deshabilitados por defecto.

Aprobar: nos permite aprobar propuestas. Para ello se debe seleccionar una o varias propuestas y hacer clic al botón para aprobarlas como vemos en la Figura 5.29.

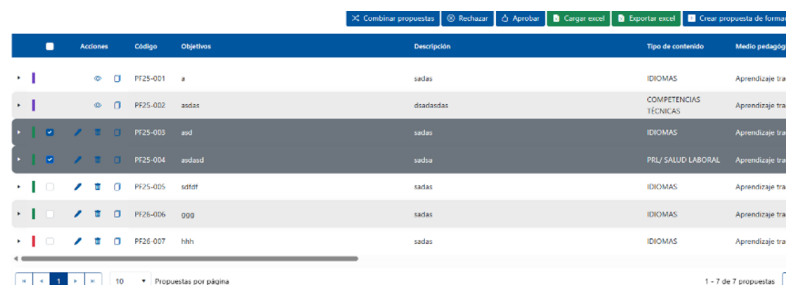
Rechazar: nos permite rechazar propuestas. Para ello se debe seleccionar una o varias propuestas y hacer clic al botón para rechazarlas como vemos en la Figura 5.29.

Combinar propuestas: A parte de poder crear una propuesta rellenando el formulario, tenemos la opción de generar una nueva propuesta combinando dos o más propuestas. Para este caso se deben seleccionar mínimo dos propuestas como vemos en la Figura 5.30



Acciones	Código	Objetivos	Descripción	Tipo de contenido	Medio pedagógico
	PF25-001	a	sadas	IDIOMAS	Aprendizaje tradicional
	PF25-002	sadas	diadestas	COMPETENCIAS TÉCNICAS	Aprendizaje tradicional
	PF25-003	asd	sadas	IDIOMAS	Aprendizaje tradicional
	PF25-004	asdasd	sadas	PRL/ SALUD LABORAL	Aprendizaje tradicional
	PF25-005	sadef	sadas	IDIOMAS	Aprendizaje tradicional
	PF26-006	qoo	sadas	IDIOMAS	Aprendizaje tradicional
	PF26-007	hnh	sadas	IDIOMAS	Aprendizaje tradicional

Figura 5.29: Pantalla con el listado de las propuestas de plan de formación anual con una fila seleccionada.



Acciones	Código	Objetivos	Descripción	Tipo de contenido	Medio pedagógico
	PF25-001	a	sadas	IDIOMAS	Aprendizaje tradicional
	PF25-002	sadas	diadestas	COMPETENCIAS TÉCNICAS	Aprendizaje tradicional
	PF25-003	asd	sadas	IDIOMAS	Aprendizaje tradicional
	PF25-004	asdasd	sadas	PRL/ SALUD LABORAL	Aprendizaje tradicional
	PF25-005	sadef	sadas	IDIOMAS	Aprendizaje tradicional
	PF25-006	qoo	sadas	IDIOMAS	Aprendizaje tradicional
	PF26-007	hnh	sadas	IDIOMAS	Aprendizaje tradicional

Propuestas por página: 10 1 - 7 de 7 propuestas

Figura 5.30: Pantalla con el listado de las propuestas de plan de formación anual con dos filas seleccionadas.

Podemos desplegar las propuestas y nos aparecerán 3 pestañas:

Asistentes: podemos ver el listado de asistentes de la propuesta, lo podemos ver en la Figura 5.31.

Propuestas unificadas: solo si la propuesta ha sido creada combinando otras, nos aparecerá el código de esas propuestas que hemos combinado, lo podemos ver en la Figura 5.32.

Formaciones creadas: nos aparecen las formaciones que tienen asignada esa propuesta, lo podemos ver en la Figura 5.33.

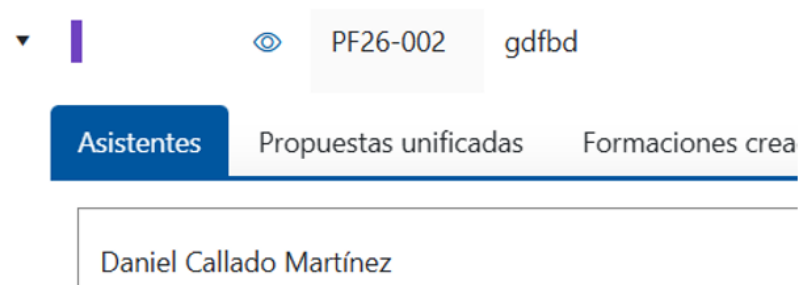


Figura 5.31: Pestaña de asistentes dentro de la propuesta del plan de formaciones.

Acciones	Código	Objetivos	Descripción	Tipo de contenido	Medio pedagógico
•	PF25-003	asd	sadas	IDIOMAS	Aprendizaje tradicional
•	PF25-004	asdasd	sadsa	PRU/ SALUD LABORAL	Aprendizaje tradicional
•	PF26-008	asd	sadas	IDIOMAS	Aprendizaje tradicional

Asistentes Propuestas unificadas Formaciones creadas

PF25-003
PF25-004

Figura 5.32: Pestaña de propuestas unificadas dentro de la propuesta del plan de formaciones.



Figura 5.33: Pestaña de formaciones creadas que tienen asignada la propuesta del plan de formaciones.

En el capítulo 5 se ha descrito el funcionamiento del Ecosistema de Aprendizaje desde el punto de vista del usuario, detallando las pantallas, los flujos y las interacciones. En esta sección se aborda el sistema desde la perspectiva del desarrollador, explicando la arquitectura interna, los patrones aplicados, el funcionamiento del backend y diversos fragmentos de código relevantes para comprender cómo se ha construido la solución.

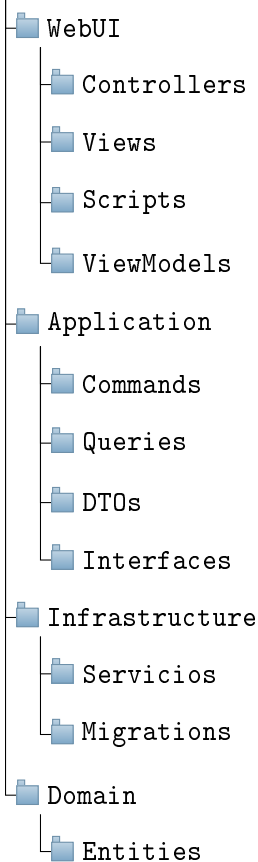
El objetivo de esta sección es aportar una visión técnica del proyecto, profundizando en la estructura del código, la implementación de la lógica de negocio, la comunicación entre las capas y las decisiones técnicas que han guiado el desarrollo.

6.1. Arquitectura técnica y organización del proyecto

El sistema se ha implementado siguiendo una arquitectura por capas combinada con el patrón *Command Query Responsibility Segregation* (CQRS). Esta combinación proporciona una estructura clara, mantenible y fácilmente escalable. La separación entre comandos y consultas permite dividir de forma explícita las operaciones que modifican estado de aquellas que únicamente lo leen.

La estructura general del proyecto puede verse en el siguiente esquema:

Proyecto



Cada una de estas áreas cumple un rol específico:

WebUI: interacción con el usuario, controladores, vistas y modelos de presentación.

Application: lógica de negocio, implementación de CQRS, validaciones y DTOs.

Infrastructure: acceso a datos, repositorios, migraciones y servicios externos.

Domain: entidades principales y reglas del dominio de negocio.

6.2. Implementación de consultas (Query Handlers)

El controlador mostrado en el capítulo 4 de formaciones llama internamente a consultas modeladas como *Handlers*. Estas consultas se encuentran en la capa *Application* y permiten recuperar datos de manera estructurada.

A continuación se muestra un ejemplo simplificado del *Handler* correspondiente a la obtención del listado de formaciones:

```
1 public class GetAllFormacionesQueryHandler :
```

```
2 IRequestHandler<GetAllFormacionesQuery ,
  ApiRespuesta<List<FormacionListDTO>>>
3 {
4     private readonly IFormacionesDataService
        _formacionesDataService;
5
6     public
        GetAllFormacionesQueryHandler(IFormacionesDataService
            formacionesDataService)
7     {
8         _formacionesDataService = formacionesDataService;
9     }
10
11     public async Task<ApiRespuesta<List<FormacionListDTO>>>
        Handle(GetAllFormacionesQuery request,
12             CancellationToken cancellationToken)
13     {
14         var apiRespuesta = new
            ApiRespuesta<List<FormacionListDTO>> {
15             Data = new List<FormacionListDTO>(), Tipo =
                TipoRespuesta.Error };
16
17         List<FormacionListDTO> formaciones = await
            Task.Run(() => _formacionesDataService
18                 .SelectAll(request.Filtros, cancellationToken));
19         if (formaciones != null)
20         {
21             apiRespuesta.Tipo = TipoRespuesta.Success;
22             apiRespuesta.Data = formaciones;
23         }
24
25         return apiRespuesta;
26     }
27 }
```

Listing 6.1: Clase Handler que realiza una consulta para obtener formaciones.

Este fragmento de código define una clase llamada *GetAllFormacionesQueryHandler*. Esta clase se encarga de manejar una solicitud para obtener todas las formaciones en el sistema, basándose en los filtros proporcionados en la consulta. En el constructor, se inyecta una dependencia del servicio *IFormacionesDataService*, que se utiliza para acceder a los datos relacionados con las formaciones. El método *Handle* es el encargado de ejecutar la consulta. Recibe como parámetros el objeto *request*, que contiene los filtros para la consulta, y un *cancellation token* para permitir la cancelación de la operación si es nece-

sario. Dentro del método, se inicializa una *apiRespuesta* con un estado de error. Luego, se ejecuta de forma asíncrona el método *SelectAll* del servicio *formacionesDataService*, que realiza la consulta a la base de datos utilizando los filtros y el token de cancelación. Si la consulta devuelve una lista de formaciones, el tipo de respuesta se actualiza a *Success*, y la lista obtenida se asigna al campo *Data* de la respuesta. Finalmente, el método retorna la respuesta que puede contener tanto los datos de las formaciones como un posible error si no se encuentran datos o si ocurre alguna otra falla. La clase *FormacionListDTO* que en este caso representa una formación contiene:

```
1 public class FormacionFiltrosDTO
2 {
3     public string FiltroTexto { get; set; }
4     public string FiltroArea { get; set; }
5     public string FiltroTipo { get; set; }
6     public string FiltroModalidad { get; set; }
7     public string FiltroTipoConocimiento { get; set; }
8     public string FiltroAsistente { get; set; }
9     public int? FiltroFormacionEstado { get; set; }
10    public int? FiltroPedido { get; set; }
11    public DateTime? FiltroFechaInicioDesde { get; set; }
12    public DateTime? FiltroFechaInicioHasta { get; set; }
13    public bool? FiltroBonificable { get; set; }
14    public bool? FiltroAccionCorrectiva { get; set; }
15    public bool? FiltroPendienteFactura { get; set; }
16 }
```

Listing 6.2: Clase DTO para filtrar las formaciones.

En cada *Query* tenemos un parámetro de filtrado.

6.3. Implementación de comandos (Command Handlers)

Las operaciones que modifican el estado de la aplicación, como crear o actualizar una formación, se implementan mediante *Command Handlers*. Su diseño es similar al de las consultas, pero incluye validaciones y lógica de negocio adicional.

```
1 public class CreateFormacionCommandHandler :
2 IRequestHandler<CreateFormacionCommand,
3     ApiResponse<FormacionDTO>>
4 {
5     private readonly IFormacionesDataService
6         _formacionesDataService;
7     public
8         CreateFormacionCommandHandler(IFormacionesDataService
```

```
        formacionesDataService)
    {
        _formacionesDataService = formacionesDataService;
    }

    public async Task<ApiRespuesta<FormacionDTO>>
        Handle(CreateFormacionCommand request,
        CancellationToken cancellationToken)
    {
        var formacionDTO = request.FormacionDTO;
        var apiRespuesta = new ApiRespuesta<FormacionDTO>() {
            Tipo = TipoRespuesta.Error
            , Data = formacionDTO };

        if (formacionDTO.ListaErrores.Count == 0) {

            InicializarCampos(formacionDTO);

            formacionDTO = await Task.Run(() =>
                _formacionesDataService
                .Insert(formacionDTO, cancellationToken));

            if (formacionDTO != null)
            {
                apiRespuesta.Tipo = TipoRespuesta.Success;
                apiRespuesta.Data = formacionDTO;
            }
        }

        return apiRespuesta;
    }

    private void InicializarCampos(FormacionDTO formacionDTO)
    {
        formacionDTO.FormacionID = 0;
        formacionDTO.FormacionEstadoID =
            (int)EFormacionEstados.PendienteAprobacion;
        formacionDTO.CreadoPor =
            Utility.UsuarioActual.NombreSinDominio;
        formacionDTO.CreadoEn = DateTime.Now;
        formacionDTO.ModificadoPor = formacionDTO.CreadoPor;
        formacionDTO.ModificadoEn = formacionDTO.CreadoEn;
    }
}
```

```
43         formacionDTO.CuestionarioDTO.CuestionarioInicialID =
44             0;
45         formacionDTO.CuestionarioDTO.CreadoPor =
46             formacionDTO.CreadoPor;
47         formacionDTO.CuestionarioDTO.CreadoEn =
48             formacionDTO.CreadoEn;
49         formacionDTO.CuestionarioDTO.ModificadoPor =
50             formacionDTO.CreadoPor;
51         formacionDTO.CuestionarioDTO.ModificadoEn =
52             formacionDTO.CreadoEn;
53
54         if (formacionDTO.AdjuntoFormFile != null
55             && formacionDTO.AdjuntoFormFile.Length > 0)
56         {
57             using (var memoryStream = new MemoryStream())
58             {
59                 formacionDTO.AdjuntoFormFile
60                     .CopyTo(memoryStream);
61                 byte[] archivoBytes = memoryStream.ToArray();
62
63                 formacionDTO.Adjunto = archivoBytes;
64                 formacionDTO.AdjuntoNombre =
65                     formacionDTO.AdjuntoFormFile.FileName;
66             }
67         }
68     }
```

Listing 6.3: Clase Handler para crear una formación.

Este fragmento de código define una clase llamada *CreateFormacionCommandHandler*. Su propósito es manejar la creación de una nueva formación. En el constructor, se inyecta el servicio *IFormacionesDataService*, utilizado para insertar la nueva formación en la base de datos.

El método *Handle* recibe como parámetros el objeto *request*, que contiene los datos de la formación a crear, y un *cancellationToken*. Inicialmente, se crea una respuesta *apiRespuesta* con un estado de error y los datos de la formación. Si la lista de errores de la formación está vacía, se procede a inicializar varios campos del objeto *formacionDTO* mediante el método *InicializarCampos*, que establece valores predeterminados como el estado de la formación y los datos de creación. Luego, se ejecuta la inserción de la formación de manera asíncrona utilizando el método *Insert* del servicio *formacionesDataService*. Si la inserción es exitosa, se actualiza la respuesta a *Success* y se asignan los datos de la formación insertada. Finalmente, el método *Handle* retorna una respuesta que puede contener la

formación recién creada o un error si ocurre algún problema en el proceso.

Este tipo de comandos garantiza que todas las modificaciones sigan las reglas de dominio establecidas en la capa *Domain*. Además, para los comandos, tenemos clases validator por cada comando donde validamos los parámetros de los comandos. Es decir, validar que no sean nulos cuando son parámetros obligatorios, validar longitud de cadenas si tienen un máximo para guardar en base de datos....

```
1      public class CreateFormacionCommandValidator :
2          AbstractValidator<CreateFormacionCommand>
3      {
4          public CreateFormacionCommandValidator()
5          {
6              RuleFor(v => v.FormacionDTO.Nombre)
7                  .NotEmpty()
8                  .NotNull()
9                  .MaxLength(256);
10
11             RuleFor(v => v.FormacionDTO.Lugar)
12                 .NotEmpty()
13                 .NotNull()
14                 .MaxLength(256);
15
16             RuleFor(v => v.FormacionDTO.FechaInicio)
17                 .LessThanOrEqualTo(l => l.FormacionDTO.FechaFin);
18         }
19     }
```

Listing 6.4: Clase Validator para crear una formación.

En este caso, validamos que:

- El valor de *Nombre* de la formación no sea nulo y que no tenga más de 256 caracteres.
- El valor de *Lugar* de la formación no sea nulo y que no tenga más de 256 caracteres.
- El valor de *FechaInicio* de la formación no sea posterior al valor de *FechaFin* de la formación.

Si alguna regla no se cumple se guarda el error y se le muestra al usuario para que rellene el formulario correctamente antes de poder crear la formación.

6.4. Acceso a datos e infraestructura

La capa *Infrastructure* implementa las interfaces definidas en *Application* mediante Entity Framework. Las entidades del dominio se mapean a tablas mediante Fluent API:

```
1 protected override void OnModelCreating(ModelBuilder
  modelBuilder)
2 {
3     modelBuilder.Entity<Formacion>(entity =>
4     {
5         entity.ToTable("Formaciones");
6         entity.Property(f => f.Nombre)
7             .HasMaxLength(200)
8             .IsRequired();
9     });
10 }
```

Listing 6.5: Mapeo de tablas mediante Fluent API.

Este fragmento de código define una sobrecarga del método *OnModelCreating* en una clase que hereda de *DbContext*. En este método, se configura el modelo de la entidad *Formacion* utilizando el objeto *modelBuilder*. La entidad *Formacion* se mapea a la tabla *Formaciones* en la base de datos mediante *entity.ToTable("Formaciones")*. Luego, se configura la propiedad *Nombre* de la entidad para que sea obligatoria (*IsRequired()*) y tenga una longitud máxima de 200 caracteres (*HasMaxLength(200)*). Esta configuración garantiza que la propiedad *Nombre* sea requerida y tenga un límite de caracteres en la base de datos.

Asimismo, cada cambio en el modelo genera una migración que se aplica sobre SQL Server, manteniendo sincronizada la estructura de la base de datos con el código fuente. Los métodos que implementan las interfaces tienen de lógica solamente el mapeo y la interacción con las tablas de base de datos.

```
1 public async Task<FormacionDTO> Insert(FormacionDTO
  formacionDTO,
2 CancellationToken cancellationToken)
3 {
4     using (var transaction =
5         _context.Database.BeginTransaction())
6     {
7         try
8         {
9             CuestionarioInicial cuestionario =
10                 _mapper.Map<CuestionarioInicial>
11                 (formacionDTO.CuestionarioDTO);
12
13                 _context.CuestionarioInicial.Add(cuestionario);
14                 await
15                     _context.SaveChangesAsync(cancellationToken);
16
17                 formacionDTO.CuestionarioInicialID =
```



```
15         cuestionario.CuestionarioInicialID;  
16         Formacion formacion =  
17             _mapper.Map<Formacion>(formacionDTO);  
18  
19         _context.Formacion.Add(formacion);  
20         await  
21             _context.SaveChangesAsync(cancellationToken);  
22  
23         formacionDTO.FormacionID = formacion.FormacionID;  
24         formacionDTO.ValoracionUsuario.ForEach(valorador  
25             =>  
26             {  
27                 Valoracion valoracionEntity = new Valoracion  
28                 {  
29                     FormacionID = formacion.FormacionID,  
30                     ValoracionUsuario = valorador  
31                 };  
32                 _context.Valoracion.Add(valoracionEntity);  
33             });  
34         await  
35             _context.SaveChangesAsync(cancellationToken);  
36         transaction.Commit();  
37         _logger.LogInformation("Exito al crear la  
38             formación");  
39     }  
40     catch (Exception ex)  
41     {  
42         transaction.Dispose();  
43         _logger.LogError(ex, "Error al crear la  
44             formación");  
45         formacionDTO = null;  
46     }  
47  
48     return formacionDTO;  
49 }
```

Listing 6.6: Mapeo de tablas mediante Fluent API.

Este fragmento de código define un método asíncrono *Insert* que inserta una nueva formación en la base de datos. Primero, se inicia una transacción de base de datos. Luego, se mapea el objeto *formacionDTO.CuestionarioDTO* a una entidad *CuestionarioInicial* y se guarda en la base de datos. Después, se mapea *formacionDTO* a una entidad *Formacion*

y se agrega a la base de datos. A continuación, se insertan las valoraciones de los usuarios asociadas a la formación. Si todas las operaciones son exitosas, se confirma la transacción. En caso de error, se captura la excepción, se deshace la transacción y se registra el error. Finalmente, el método retorna el objeto *formacionDTO*.

6.5. Sistema de permisos y filtros de autorización

El control de acceso a los distintos métodos del controlador se realiza mediante un filtro personalizado que valida los roles del usuario autenticado. Esto permite proteger acciones como ver detalles de una formación o tramitar solicitudes.

```
1 public class TienePermisosFilterAttribute :
   AuthorizeAttribute, IAuthorizationFilter
2 {
3
4     private EAutorizacionPermisos AutorizacionPermisos { get;
       set; }
5     public TienePermisosFilterAttribute(EAutorizacionPermisos
       autorizacionPermisos)
6     {
7         AutorizacionPermisos = autorizacionPermisos;
8     }
9
10    public void OnAuthorization(AuthorizationFilterContext
       context)
11    {
12
13        var usuario = Utility.UsuarioActual;
14
15        if (context.HttpContext.User
16            .HasClaim("NotAuthorized", "NotAuthorized")
17            || usuario == null)
18        {
19            context.Result = new RedirectToActionResult(
20                "HttpStatusCodeHandler",
21                "Error",
22                new { statusCode = 403 });
23        }
24        else
25        {
26            switch (AutorizacionPermisos) {
27                case EAutorizacionPermisos.NINGUNO:
28                    if ((usuario.AutorizacionPermisos &
```

```
29         EAutorizacionPermisos.NINGUNO)
30     != EAutorizacionPermisos.NINGUNO)
31     context.Result = new RedirectToActionResult(
32         "HttpStatusCodeHandler",
33         "Error",
34         new { statusCode = 403 });
35     break;
36
37     case EAutorizacionPermisos.SOLICITANTE:
38     if ((usuario.AutorizacionPermisos &
39         EAutorizacionPermisos.SOLICITANTE)
40     != EAutorizacionPermisos.SOLICITANTE)
41     context.Result = new RedirectToActionResult(
42         "HttpStatusCodeHandler",
43         "Error",
44         new { statusCode = 403 });
45     break;
46
47     case EAutorizacionPermisos.PYV:
48     if ((usuario.AutorizacionPermisos &
49         EAutorizacionPermisos.PYV)
50     != EAutorizacionPermisos.PYV)
51     context.Result = new RedirectToActionResult(
52         "HttpStatusCodeHandler",
53         "Error",
54         new { statusCode = 403 });
55     break;
56
57     case EAutorizacionPermisos.ADMINISTRADOR:
58     if (((usuario.AutorizacionPermisos &
59         EAutorizacionPermisos.ADMINISTRADOR)
60     != EAutorizacionPermisos.ADMINISTRADOR) &&
61         !usuario.EsUsuarioSuplantado)
62     context.Result = new RedirectToActionResult(
63         "HttpStatusCodeHandler",
64         "Error",
65         new { statusCode = 403 });
66     break;
67
68     case EAutorizacionPermisos.RESPONSABLENEGOCIO:
69     if (((usuario.AutorizacionPermisos &
70         EAutorizacionPermisos.RESPONSABLENEGOCIO)
71     != EAutorizacionPermisos.RESPONSABLENEGOCIO)
```

```
66         && !usuario.EsUsuarioSuplantado)
67         context.Result = new RedirectToActionResult(
68             "HttpStatusCodeHandler",
69             "Error",
70             new { statusCode = 403 });
71         break;
72
73     default:
74         context.Result = new RedirectToActionResult(
75             "HttpStatusCodeHandler",
76             "Error",
77             new { statusCode = 403 });
78         break;
79     }
80 }
81 }
82 }
```

Listing 6.7: Clase que valida el permiso del usuario para poder consultar vistas o realizar acciones.

Este fragmento de código define una clase *TienePermisosFilterAttribute* que hereda de *AuthorizeAttribute* e implementa la interfaz *IAuthorizationFilter*. Su función es verificar si el usuario tiene los permisos necesarios para acceder a un recurso. En el constructor, se recibe un parámetro *AutorizacionPermisos*, que indica el nivel de permiso requerido. El método *OnAuthorization* comprueba si el usuario está autorizado para realizar la acción solicitada, evaluando su rol mediante un *switch* según el tipo de permiso. Si el usuario no tiene el permiso adecuado o está marcado como *NotAuthorized*, se redirige a una página de error con el código de estado 403.

El uso de este filtro en los controladores permite centralizar la gestión de permisos y evitar duplicación de lógica.

6.6. Sistema de notificaciones y enlaces únicos

El envío automático de correos electrónicos es una de las piezas clave del flujo interno. Tenemos un comando encargado de envío de correos después de que se aplique algún trámite, como por ejemplo al agregar un asistente a una formación le llega un correo para poder aceptar o rechazar la convocatoria. También tenemos otro controlador que es un recordatorio que envía diferentes correos según las fechas. Por ejemplo, cuando un asistente es agregado a una formación y queda una semana para que la formación comience, si ese asistente aún no ha aceptado ni rechazado la convocatoria automáticamente le llega un correo de recordatorio ya que cuando quede un solo día para que comience la formación, si no confirmó entonces se rechaza automáticamente.

6.7. Procesamiento de encuestas y valoraciones

Al finalizar una acción formativa, el sistema envía automáticamente una encuesta de eficacia. Cada respuesta queda asociada a un asistente y a una formación, y el promedio se calcula dinámicamente. Este proceso se refleja posteriormente en la pestaña *Valoraciones*, visible para los roles autorizados.


6.8. Conclusión técnica

El diseño interno del Ecosistema de Aprendizaje combina una arquitectura moderna con patrones avanzados y una estricta separación por capas. Esto ha permitido desarrollar un sistema robusto, mantenible y fácilmente ampliable. La modularidad del código y el uso de Entity Framework, servicios desacoplados y filtros de permisos garantizan que la aplicación pueda evolucionar en el futuro incorporando nuevas funcionalidades sin comprometer su estabilidad.

Bibliografía

- [1] (2025) Página de tracasa global. [Online]. Available: <https://tracasa.es>
- [2] (2025) Página de tracasa instrumental. [Online]. Available: <https://itracasa.es>
- [3] (2025) ¿qué es .net? [Online]. Available: <https://dotnet.microsoft.com/es-es/learn/dotnet/what-is-dotnet>
- [4] (2025) ¿qué es c#? [Online]. Available: <https://dotnet.microsoft.com/es-es/languages/csharp>
- [5] (2025) ¿qué es typescript? [Online]. Available: <https://codigofacilito.com/articulos/typescript>
- [6] (2025) ¿qué es html? [Online]. Available: <https://developer.mozilla.org/es/docs/Web/HTML>
- [7] (2025) ¿qué es css? [Online]. Available: <https://lenguajecss.com/css/introduccion/que-es-css/>
- [8] (2025) ¿qué es cqrs? [Online]. Available: <https://learn.microsoft.com/es-es/azure/architecture/patterns/cqrs>
- [9] (2025) ¿qué son las pruebas de integración? [Online]. Available: <https://www.ibm.com/es-es/think/topics/integration-testing>
- [10] (2025) ¿qué es un framework? [Online]. Available: <https://www.ibm.com/es-es/think/topics/integration-testing>
- [11] (2025) ¿qué es sql server? [Online]. Available: <https://www.microsoft.com/es-es/sql-server>

Apéndice



The screenshot shows a web application interface for managing modalities. At the top, there's a header with a menu icon, the title 'Modalidades', and user information 'yasin', 'Manual', and 'Listado AC'. Below the header is a filter section with a dropdown for 'Filtros' and a search bar for 'Nombre'. The main content area is a table with columns 'Nombre', 'Activo', and 'Acciones'. The table lists four modalities: 'Aula Virtual (Streaming)', 'Mista', 'Presencial', and 'Teleformación (Online)'. Each row has a blue 'Activo' button and a red 'Eliminar' button. At the bottom, there's a pagination bar showing '1 - 4 de 4 elementos'.

Nombre	Activo	Acciones
Aula Virtual (Streaming)	✓ Activo	
Mista	✓ Activo	
Presencial	✓ Activo	
Teleformación (Online)	✓ Activo	

Figura Apéndice.1: Listado de mantenimiento de modalidad.



The screenshot shows a web application interface for managing content types. At the top, there's a header with a menu icon, the title 'Tipos de contenido', and user information 'yasin', 'Manual', and 'Listado AC'. Below the header is a filter section with a dropdown for 'Filtros' and a search bar for 'Nombre'. The main content area is a table with columns 'Nombre', 'Activo', and 'Acciones'. The table lists eight content types: 'COMPETENCIAS TECNICAS', 'IDIOMAS', 'IGUALDAD, INCLUSION Y DIVERSIDAD', 'METODOLOGIAS AGILE', 'PRL/ SALUD LABORAL', 'SEGURIDAD DE LA INFORMACION', 'SISTEMAS DE GESTION', and 'SOFT SKILLS'. Each row has a blue 'Activo' button and a red 'Eliminar' button. At the bottom, there's a pagination bar showing '1 - 8 de 8 elementos'.

Nombre	Activo	Acciones
COMPETENCIAS TECNICAS	✓ Activo	
IDIOMAS	✓ Activo	
IGUALDAD, INCLUSION Y DIVERSIDAD	✓ Activo	
METODOLOGIAS AGILE	✓ Activo	
PRL/ SALUD LABORAL	✓ Activo	
SEGURIDAD DE LA INFORMACION	✓ Activo	
SISTEMAS DE GESTION	✓ Activo	
SOFT SKILLS	✓ Activo	

Figura Apéndice.2: Listado de tipos de contenido.

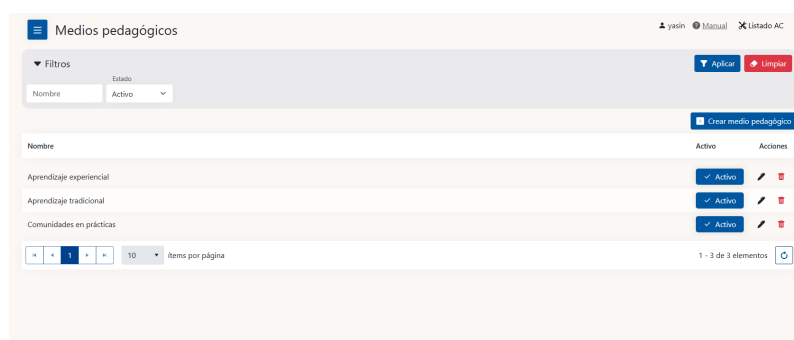


Figura Apéndice.3: Listado de mantenimiento de medios pedagógicos.

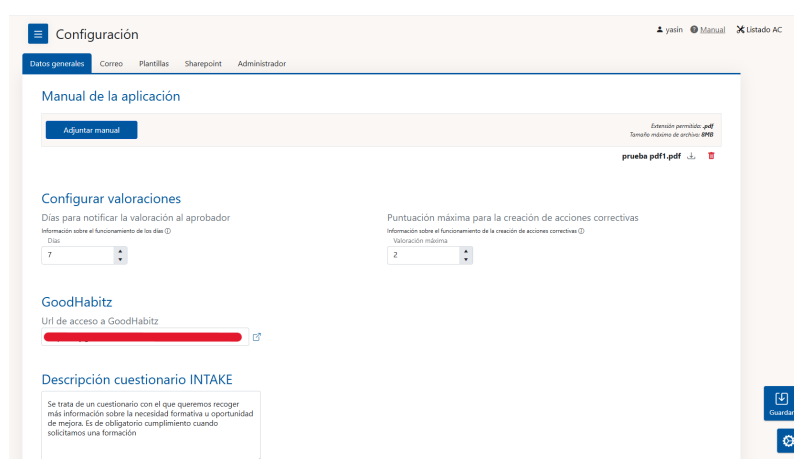


Figura Apéndice.4: Pestaña datos generales de configuración.

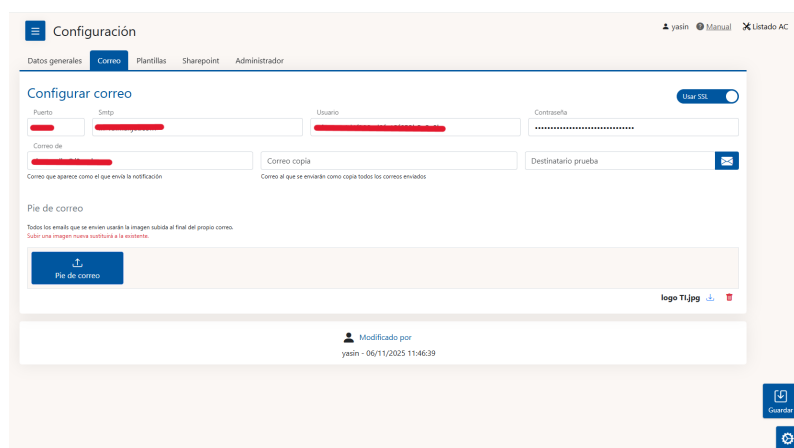


Figura Apéndice.5: Pestaña de configuración de correo.

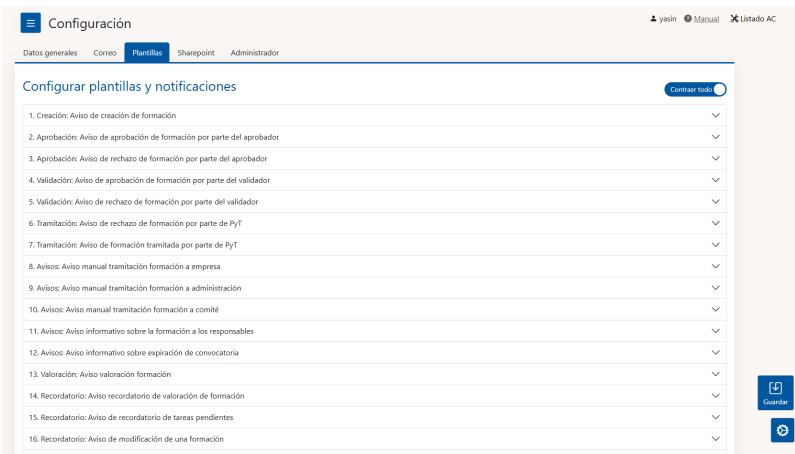


Figura Apéndice.6: Pestaña con las plantillas de los correos.

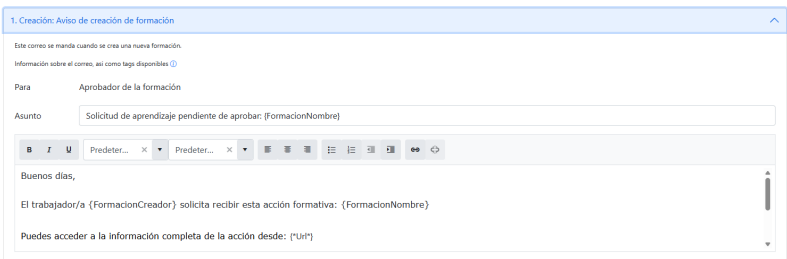


Figura Apéndice.7: Ejemplo con una plantilla de correo.

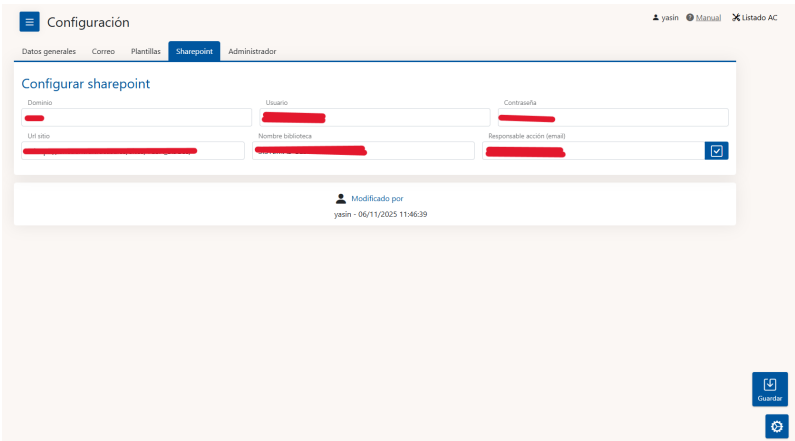


Figura Apéndice.8: Pestaña sharepoint de configuración.

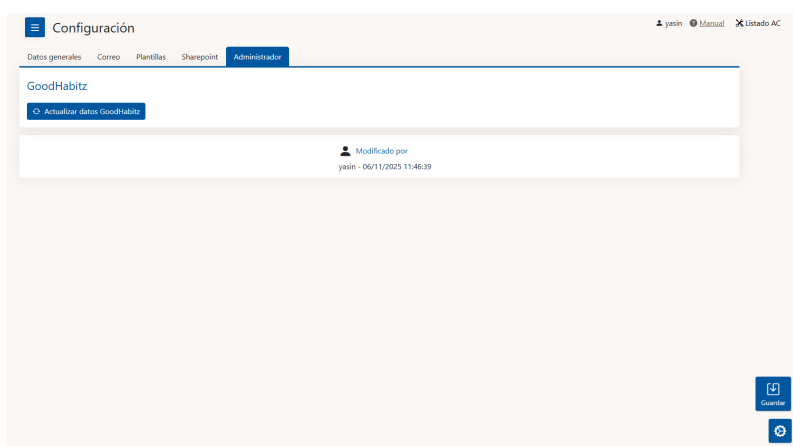


Figura Apéndice.9: Pestaña administrador de configuración.