



UNIVERSIDAD DE BURGOS  
ESCUELA POLITÉCNICA SUPERIOR  
Grado en Ingeniería Informática



## **TFG del Grado en Ingeniería Informática**

### **Gestión de Formación**

Presentado por Víctor Manuel Vaquero Mesa  
en Universidad de Burgos — dd de enero de 2025  
Tutores: D. José Ignacio Santos Martín





UNIVERSIDAD DE BURGOS  
ESCUELA POLITÉCNICA SUPERIOR  
Grado en Ingeniería Informática



D. José Ignacio Santos Martín, profesor del departamento de Ingeniería de Organización.

Exponen:

Que el alumno D. Víctor Manuel Vaquero Mesa, con DNI 52669729M, ha realizado el Trabajo final de Grado en Ingeniería Informática titulado Gestión de Formación.

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, dd de enero de 2025

Vº. Bº. del Tutor:

D. José Ignacio Santos Martín



## **Resumen**

La formación continua de los trabajadores de una empresa es de vital importancia para el desarrollo profesional, la mejora de competencias y la adaptación a los diferentes puestos de trabajo de la organización. Sin embargo, la gestión de las actividades formativas puede resultar compleja debido a la cantidad de datos involucrados, la necesidad de coordinar distintos perfiles de actuación y la supervisión del proceso en todas sus etapas.

Este trabajo tiene como objetivo desarrollar una aplicación web que integre y organice todo el proceso de formación continua en una empresa, permitiendo la gestión eficiente de cursos, inscripciones, participantes y formadores.

## **Descriptores**

Gestión de formación, Portal del participante, portal del gestor, portal del formador, asistencias, calificaciones, certificados, cursos, ediciones, symfony, php.

## **Abstract**

The continuous training of a company's employees is of vital importance for professional development, skills improvement and adaptation to the different jobs in the organisation. However, the management of training activities can be complex due to the amount of data involved, the need to coordinate different profiles and the supervision of the process in all its stages.

The aim of this work is to develop a web application that integrates and organises the whole process of continuous training in a company, allowing the efficient management of courses, registrations, participants and trainers.

## **Keywords**

Training management, Participant portal, Manager portal, Trainer portal, Attendance, Qualifications, Certificates, Courses, Editions, Symfony, php.

---

# Índice general

---

## Tabla de contenido

<i>Índice general</i> .....	<i>V</i>
<i>Índice de figuras</i> .....	<i>VII</i>
<b>1. Introducción</b> .....	<b>VIII</b>
1.1. Estructura de la memoria .....	1
1.2. Materiales adjuntos .....	1
<b>2. Objetivos del proyecto</b> .....	<b>3</b>
2.1. Objetivos generales .....	3
2.2. Objetivos técnicos .....	3
2.3. Objetivos personales .....	4
<b>3. Conceptos teóricos</b> .....	<b>6</b>
3.1. Gestión de usuarios: roles y funcionalidades .....	6
3.2. Gestión de cursos y ediciones .....	6
3.3. Funcionalidades del formador .....	8
3.4. Reglas para los participantes .....	8
3.5. Funcionalidades del rol admin .....	9
<b>4. Técnicas y herramientas</b> .....	<b>10</b>
4.1. Metodologías de desarrollo .....	10
4.2. Herramientas de desarrollo .....	11
4.3. Gestión de dependencias .....	13
4.4. Control de versiones .....	13
4.5. Seguridad .....	13
4.6. Otras herramientas .....	14
<b>5. Aspectos relevantes del desarrollo del proyecto</b> .....	<b>15</b>
5.1. Inicio del proyecto .....	15
5.2. Metodologías empleadas .....	16
5.3. Formación .....	19

5.4.	Desarrollo de la lógica de negocio. ....	21
5.5.	Desarrollo de la aplicación.....	23
5.6.	Pruebas sobre el código. ....	31
6.	<i>Trabajos relacionados</i> .....	33
7.	<i>Conclusiones y Líneas de trabajo futuras</i> .....	34
7.1.	Conclusiones .....	34
7.2.	Líneas de trabajo futuras.....	34
	<i>Bibliografía</i> .....	35



---

## Índice de figuras

---

Figura 5.1: Portal del Gestor .....	16
Figura 5.2: Épicas en zube.io .....	17
Figura 5.3: Resumen elementos metodología Scrum adoptada .....	18
Figura 5.4: lógica de negocio del código de curso.....	22
Figura 5.5: lógica de negocio del código de edición.....	22

---

# 1. Introducción

---

La formación continua es un elemento clave para el desarrollo profesional de los trabajadores y la mejora de la competitividad de las empresas en un mercado laboral cada vez más dinámico y exigente.

La capacidad de adaptación y actualización de los empleados resulta fundamental para cubrir las necesidades de los diferentes puestos de trabajo de una organización. No obstante, la gestión de las actividades formativas supone un reto importante debido a la cantidad de información involucrada y a la necesidad de coordinar distintos perfiles de actuación en todas las etapas del proceso: planificación, ejecución y evaluación.

Actualmente, en muchas organizaciones, la gestión de los cursos de formación se realiza mediante herramientas no especializadas. Esto puede derivar en problemas como duplicidad de datos, dificultades en la supervisión del avance de los cursos, pérdida de información relevante o un uso ineficiente de los recursos. La falta de una herramienta específica que permita organizar y automatizar estas tareas genera una sobrecarga de trabajo y ralentiza la toma de decisiones por parte de los responsables de formación.

Con este planteamiento, surge la necesidad de desarrollar una solución tecnológica que permita simplificar y optimizar la gestión de las actividades formativas.

En este trabajo se propone el desarrollo de una aplicación web que integre y organice todo el proceso de formación continua en una empresa, permitiendo la gestión eficiente de cursos, inscripciones, participantes y formadores.

La aplicación está estructurada en tres portales diferenciados que responden a las necesidades de los distintos perfiles involucrados en el proceso:

- **Gestores:** responsables de la planificación y administración de cursos y ediciones, gestión de inscripciones, supervisión del avance de la formación, gestión de la documentación de los formadores y sus retribuciones, así como de las certificaciones de los cursos impartidos.

- **Formadores:** encargados de registrar la asistencia, calificaciones y detalles relacionados con las sesiones de las ediciones asignadas.
- **Participantes:** trabajadores que pueden consultar el estado de su ficha formativa, inscribirse en cursos disponibles, visualizar su progreso y consultar los próximos cursos por comenzar en los que están inscritos.

La principal ventaja de esta herramienta es la automatización de tareas repetitivas y la centralización de la información, lo que permite un importante ahorro de tiempo y recursos, facilitando la supervisión del proceso formativo en tiempo real y ofreciendo funcionalidades adaptadas a cada rol, mejorando en ello la experiencia de uso y la eficacia del proceso de formación.

Con esta herramienta, se pretende resolver las dificultades actuales en la gestión de formación continua, ofreciendo una solución accesible, eficiente y escalable para organizaciones de cualquier tamaño.

## **1.1. Estructura de la memoria**

La memoria sigue la siguiente estructura:

- **Introducción:** breve descripción del proyecto, estructura de la memoria y listado de materiales adjuntos.
- **Objetivos del proyecto:** descripción de los objetivos que persigue el proyecto.
- **Conceptos teóricos:** breve explicación de los conceptos necesarios para la realización del proyecto.
- **Técnicas y herramientas:** descripción de las metodologías y herramientas que han sido utilizadas para llevar a cabo el proyecto.
- **Aspectos relevantes del desarrollo:** aspectos a destacar a lo largo de la realización del proyecto.
- **Trabajos relacionados:** resumen de trabajos y proyectos ya realizados en el campo del proyecto en curso.
- **Conclusiones y líneas de trabajo futuras:** conclusiones obtenidas al finalizar el proyecto y posibles ideas de continuidad.

## **1.2. Materiales adjuntos**

Los materiales adjuntos a la memoria son:

- Dirección del repositorio del proyecto.
- Dirección del portal de gestión ágil usado en el desarrollo.
- Dirección web del despliegue del proyecto.
- Dirección del video con demostración funcional de la aplicación.
- Dirección del video con descripción del proyecto.

---

## 2. Objetivos del proyecto

---

En este apartado abarcaremos los objetivos generales, técnicos y personales del proyecto.

### 2.1. Objetivos generales

- a) Desarrollar una aplicación web que permita gestionar cursos de formación, facilitando la administración de ediciones, inscripciones, y recursos asociados.
- b) Implementar un solo portal con funcionalidades específicas para tres roles: gestor, participante y formador.
- c) Facilitar la inscripción de participantes en las ediciones de cursos y el seguimiento de su progreso académico.
- d) Proporcionar a los formadores una herramienta para gestionar sesiones de formación, asistencia y calificaciones.
- e) Garantizar un acceso seguro a la aplicación mediante autenticación y control de roles.
- f) Almacenar y organizar toda la información de los cursos, ediciones, participantes y formadores en una base de datos relacional.

### 2.2. Objetivos técnicos

- a) Desarrollar la aplicación utilizando el framework Symfony 7, que sigue el patrón arquitectónico Modelo-Vista-Controlador (MVC) para garantizar una separación clara de responsabilidades.
- b) Utilizar PHP 8.2 como lenguaje principal y MySQL 8 como sistema de gestión de bases de datos relacionales.
- c) Implementar la autenticación de usuarios utilizando un sistema de registro con envío de correos de alta, de seguridad con almacenamiento hash seguro de contraseñas y control de roles de usuario.

- d) Desarrollar formularios con validaciones y mensajes de confirmación para la gestión de participantes, formadores y ediciones de cursos.
- e) Asegurar la calidad del código utilizando herramientas como PHPUnit para la realización de pruebas unitarias y PHPStan para el análisis estático del código.
- f) Integrar un sistema de control de versiones mediante GitHub, aplicando buenas prácticas de desarrollo con ramas diferenciadas por issues, integración en rama developer y pull requests para la integración de producto final en rama main.
- g) Gestionar y automatizar procesos de despliegue mediante pipelines de CI/CD configurados en GitHub Actions.
- h) Utilización de Zube.io como herramienta de gestión ágil para planificar, organizar y dar seguimiento a las tareas del proyecto.
- i) Implementación de gestión de dependencias para la gestión de recursos frontend mediante AssetMapper.
- j) Implementar la base de datos relacional con relaciones como ManyToMany y OneToMany entre entidades clave.
- k) Despliegue de la aplicación en un entorno de pruebas para validar su funcionamiento.

## **2.3. Objetivos personales**

- a) Consolidar los conocimientos adquiridos en el desarrollo de aplicaciones web con Symfony y en el uso del modelo MVC.
- b) Mejorar las habilidades en el diseño e implementación de bases de datos relacionales complejas con MySQL.
- c) Realización de pruebas unitarias y de integración al finalizar cada sprint, asegurando la funcionalidad del código implementado y mejorando su calidad mediante herramientas como PHPUnit y PHPStan.
- d) Familiarización con herramientas de gestión de proyectos como Zube.io y flujos de trabajo en GitHub.

## *2- Objetivos del proyecto*

- e) Desarrollo de un sistema web robusto y funcional que resuelva necesidades reales en la gestión de cursos de formación.
- f) Desplegar una aplicación funcional que pueda servir como base para futuros desarrollos.

---

## 3. Conceptos teóricos

---

En este apartado se exponen los conceptos teóricos del problema que han servido como base para el desarrollo del proyecto. Nos centraremos en la gestión de cursos de formación y sus ediciones, explicando los roles involucrados y las reglas funcionales que rigen la aplicación.

### 3.1. Gestión de usuarios: roles y funcionalidades

La aplicación distingue tres roles principales: **gestor**, **participante** y **formador**. Cada uno con funciones específicas y un portal propio que dan sentido a esta separación de roles entre los usuarios de la aplicación.

**Participantes y formadores:** la aplicación ofrece un **formulario de registro**, siendo esta la única forma de dar de alta a este tipo de usuarios con su rol específico, no siendo posible que el gestor dé de alta directamente a participantes ni formadores.

**Activación de roles adicionales:** en el caso de un usuario existente (participante o formador), el gestor tiene la capacidad de **activar el rol complementario** (es decir, convertir a un participante en formador o viceversa) para dotar al usuario de funcionalidades completas, pero no puede crearlo si no existe un alta previa.

### 3.2. Gestión de cursos y ediciones

Los **cursos** y sus **ediciones** son los elementos centrales en la gestión de la formación.

**Definición de un curso.**

Un curso es asimilable a la guía docente de una asignatura, compuesta por información general como:



- **Código del curso:** cada curso cuenta con un código único de **5 dígitos**, donde:
  - Los **dos primeros dígitos** corresponden al **año del curso**.
  - Los **tres últimos dígitos** son generados de manera **incremental** (por ejemplo, 24001 para el primer curso del año 2024).
- Nombre del curso.
- Objetivos y justificación de la actividad formativa.
- Contenidos del curso.
- Duración y número de participantes.

Los cursos tienen 2 campos de gran importancia que definen su formato de impartición:

- Horas: Son las horas totales de duración del curso.
- Horas virtuales: este campo es el que marca el **modelo de impartición** del curso.
  - Si el curso no tiene horas virtuales estamos declarando un curso **presencial**.
  - Si el número de horas virtuales es inferior a las horas del curso estamos hablando de un curso **semipresencial**.
  - Si las horas virtuales coinciden con las horas del curso estamos ante un curso **virtual**.

En el contexto de este proyecto, el formato de impartición influye significativamente en la naturaleza del curso. Por defecto, todos los cursos que incluyen **horas virtuales** serán **evaluables**, a diferencia de los cursos presenciales, que no lo son necesariamente. La activación de este campo, aunque **puede desactivarse posteriormente**, implica que en estos cursos se medirán los conocimientos adquiridos por los participantes mediante pruebas o actividades específicas.

### Definición de una edición.

Las ediciones representan las distintas instancias o ejecuciones de un mismo curso. Todas las ediciones **comparten propiedades generales**, como el número de horas, las horas virtuales y su carácter evaluable. Además, cada edición incluye datos específicos, como:

- **Código de la edición:** son generados de forma automática y correlativa, y está formado por un código único de **8 caracteres**, donde:
  - Los 5 primeros caracteres corresponden con el **código del curso**.
  - Le sigue una **barra inclinada** que separa el código del curso de la edición.
  - Los dos últimos caracteres son un **número incremental** que identifican la edición (24001/01, 24001/02, etc.).

- Fechas de inicio y fin.
- Lugar de impartición.
- Calendario y horario
- Máximo de participantes, que este si será un dato propio de cada edición.

Para cada curso existe una edición especial identificada con el código codigoCurso/00. Esta edición no permite establecer valores a sus campos ni la asignación de formadores, ya que su propósito es servir como base para implementar futuras funcionalidades, como la gestión de listas de espera o la inscripción a cursos de oficio. De momento, no se utiliza activamente en la planificación regular de los cursos, pero se reserva para posibles ampliaciones del proyecto en el futuro.

### 3.3. Funcionalidades del formador

Los formadores desempeñan un papel clave en la gestión de las ediciones al tener el encargo de realizar las siguientes tareas:

- **Crear y gestionar las sesiones** (días de impartición de una edición) de las ediciones asignadas, utilizando como base el calendario y horario definidos previamente.
- **Registrar asistencia y calificaciones:** los formadores deben garantizar la correcta introducción de los datos de sesiones, asistencias y calificaciones (en caso de cursos evaluables).
- **Remitir datos:** una vez completada la información necesaria, el formador puede remitir los datos para cerrar la edición.

Cuando una edición ha sido cerrada por el formador, el gestor tiene la posibilidad de:

- **Certificar la edición**, validando los datos introducidos.
- **Emitir los certificados** correspondientes para los participantes que cumplan los criterios establecidos (porcentaje de asistencia y calificaciones si las hubiera).

### 3.4. Reglas para los participantes

Los participantes tienen ciertas **restricciones y condiciones** en la gestión de sus inscripciones:

- Solo pueden realizar una inscripción en aquellas **ediciones que aún no hayan comenzado**.
- Si está inscrito en una **edición que ya ha iniciado**, un participante no puede anular su inscripción. En su lugar, debe solicitar una **baja justificada** a través de los mecanismos definidos en la aplicación.
- Un participante no puede inscribirse en dos **ediciones del mismo curso** simultáneamente.
- La posibilidad de realizar inscripciones en las ediciones depende de que el campo "**unidad**" del participante esté relleno. Si este campo está **vacío**, el **usuario** se considera **inactivo** y no puede realizar inscripciones.

### 3.5. Funcionalidades del rol admin

El rol **admin**, desempeñado por el personal del departamento de formación, es el núcleo operativo de la aplicación.

Desde el **Portal del Gestor**, los administradores tienen acceso a las funcionalidades clave que permiten gestionar todos los aspectos relacionados con los cursos, ediciones, inscripciones, formadores y participantes. Su papel es esencial para garantizar el correcto funcionamiento del sistema y el cumplimiento de los objetivos de la aplicación. Por ejemplo:

- Los administradores son responsables de la creación y configuración de los cursos en el sistema, indicando si el curso será visible a los usuarios o no, si será calificable, su duración y modalidad.
- Deberán crear las distintas ediciones y asignarlas a los cursos a los que pertenecen. Indicar el número de participantes que se pueden inscribir, su calendario, horario y lugar entre otros datos.
- Gestionar las inscripciones de los participantes a las distintas ediciones, sus bajas justificadas y emitir las certificaciones que correspondan.
- Asignar los formadores a las distintas ediciones de los cursos creados, según su experiencia y disponibilidad. Registrando la recepción de documentación necesaria del formador, como datos bancarios o justificantes, las horas impartidas de clases y su retribución, si corresponde.
- Facilitando la actualización y el mantenimiento de los datos de los formadores y participantes.

---

## 4. Técnicas y herramientas

---

En esta sección se describen las principales metodologías y herramientas empleadas durante el desarrollo del proyecto, abarcando tanto aspectos técnicos como organizativos. Cada herramienta o técnica se detalla en función de su propósito y del uso específico dentro del proyecto.

### 4.1. Metodologías de desarrollo

- **Gestión ágil con Scrum:**

Se adoptó la metodología ágil Scrum para estructurar el trabajo del proyecto en ciclos iterativos llamados sprints, con una duración de 14 días cada uno. Al inicio de cada sprint, se definieron las historias de usuario a implementar, permitiendo establecer los objetivos considerados más importantes para cada iteración.

Esta metodología facilitó la adaptación a los cambios en los requisitos y la priorización de tareas según su importancia y transcendencia que tenían en el resto de las tareas que se debían implementar posteriormente.

La planificación y el seguimiento de las tareas se realizaron mediante la herramienta Zube.io, que incluye un tablero Kanban y se integra con el repositorio de GitHub, permitiendo un control eficiente del progreso.

Para estructurar y organizar las tareas dentro de cada sprint, se crearon épicas en Zube.io, agrupando las historias de usuario relacionadas. Esto permitió una visión clara de los objetivos de cada sprint y facilitó la priorización de tareas. Además, en GitHub se configuraron milestones para tener una representación visual y centralizada del progreso en cada sprint, alineando la gestión de código con la planificación ágil.

Al finalizar cada sprint, se llevaron a cabo las implementaciones de los tests unitarios y el análisis estático del código para evaluar la calidad del código implementado, así como una revisión del sprint para evaluar el avance de las historias de usuario completadas y reajustar las prioridades para los siguientes ciclos.

- **Integración y Despliegue Continuo (CI/CD):**

Se configuraron pipelines en GitHub Actions para garantizar la calidad del código mediante la ejecución automática de pruebas unitarias (PHPUnit). Estas herramientas aseguraron que cada cambio en el código fuera probado antes de ser integrado en la rama principal.

## 4.2. Herramientas de desarrollo

- **Symfony 7:**

El framework Symfony es el núcleo del desarrollo, utilizado para implementar las funcionalidades del backend y elegido por su robustez, escalabilidad y adherencia al patrón Modelo-Vista-Controlador (MVC). Simplificó la creación y validación de formularios, así como la gestión de entidades mediante Doctrine ORM. La autenticación de usuarios y asignación de roles se implementó de forma sencilla y segura utilizando sus herramientas nativas.

Symfony 7 adopta el patrón Modelo-Vista-Controlador (MVC), que permite separar las responsabilidades en la lógica de negocio (Modelo), la gestión de solicitudes (Controlador) y la presentación de la información (Vista). Este enfoque estructurado facilitó el mantenimiento del código y la implementación de nuevas funcionalidades

- **PHP 8.2:**

El proyecto utilizó PHP 8.2, aprovechando sus características modernas, como:

- Tipos estrictos: asegurando la coherencia de los datos en funciones y métodos.
- Expresiones match: simplificando decisiones condicionales.

PHP 8.2 contribuyó a mejorar el rendimiento y la legibilidad del código.

- **MySQL 8:**

MySQL se empleó como sistema de gestión de bases de datos relacional, ideal para manejar las entidades principales del proyecto (cursos, ediciones, participantes, formadores, entre otros).

- Esquema relacional: se diseñaron relaciones OneToMany y ManyToMany para modelar las interacciones entre los datos.
- Optimización: la base de datos fue configurada para soportar consultas eficientes, clave en la gestión de grandes volúmenes de datos.

- **Doctrine ORM:**

Se empleó Doctrine para el mapeo objeto-relacional, facilitando la interacción con la base de datos, la definición de relaciones entre entidades y reduciendo la necesidad de escribir consultas SQL manuales.

Mediante las migraciones, Doctrine permitió gestionar cambios en el esquema de la base de datos de manera controlada.

- **PHPUnit:**

Se implementaron pruebas unitarias con PHPUnit para validar la funcionalidad de controladores, formularios y entidades, garantizando la estabilidad y fiabilidad del sistema.

PHPUnit fue una herramienta clave en el proyecto, integrándose con GitHub Actions para ejecutar automáticamente las pruebas en cada commit y en cada Pull Request hacia la rama principal.

- **PHPStan:**

El análisis estático de código con PHPStan fue clave para detectar errores e inconsistencias, contribuyendo a mejorar la calidad del código antes de su despliegue.

Se configuró un nivel alto de análisis, lo que permitió identificar problemas de tipo, métodos no utilizados y errores comunes. El feedback proporcionado por PHPStan ayudó a mantener un código limpio y robusto.

- **Sequel Ace:**

Sequel Ace se utilizó para administrar visualmente la base de datos.

- Pruebas manuales: facilitó la validación y corrección de datos durante el desarrollo.
- Gestión de esquemas: simplificó la creación y edición de tablas.

## 4.3. Gestión de dependencias

- **Composer:**

Se utilizó Composer para la gestión de dependencias del proyecto, asegurando que todas las librerías estuvieran actualizadas y compatibles con Symfony.

- **AssetMapper:**

En lugar de herramientas tradicionales como npm o yarn, se utilizó AssetMapper para la gestión de recursos frontend, integrando estilos y scripts de forma eficiente en el entorno de Symfony.

## 4.4. Control de versiones

- **Git y GitHub:**

El proyecto fue gestionado mediante Git, con un flujo de trabajo basado en ramas (developer para desarrollo y main para producción).

Cada historia de usuario se implementó en una rama propia, derivada de la rama developer. Una vez completado el desarrollo, los cambios realizados en la rama fueron integrados en developer.

Se realizaron pull requests para integrar cambios y asegurar revisiones de calidad antes de la fusión con la rama principal.

## 4.5. Seguridad

- **Autenticación y roles:**

El sistema de seguridad de Symfony se utilizó para implementar autenticación segura y asignación de roles (ROLE\_USER , ROLE\_ADMIN, ROLE\_TEACHER).

Las contraseñas de los usuarios se almacenan de forma segura mediante hashing con algoritmos modernos como bcrypt.

## 4.6. Otras herramientas

- **PHPStorm:**

PHPStorm fue el IDE principal, elegido por sus características avanzadas:

- Soporte para Symfony: atajos y herramientas específicas para trabajar con el framework.
- Depuración: integración con Xdebug para identificar errores en tiempo de ejecución.

- **Xdebug:**

Xdebug se utilizó para depurar y generar reportes de cobertura de código en PHPUnit.

- Depuración en profundidad: ayudó a identificar problemas específicos en la lógica de negocio.
- Cobertura de pruebas: permitió visualizar qué partes del código estaban siendo probadas.



---

## 5. Aspectos relevantes del desarrollo del proyecto

---

Este apartado aborda los puntos más destacados y significativos del desarrollo de la aplicación, desde la concepción de la idea hasta su puesta en marcha. Para facilitar su lectura, se presenta dividida en apartados que cubren tanto cuestiones metodológicas como técnicas, de formación y de organización del trabajo.

### 5.1. Inicio del proyecto

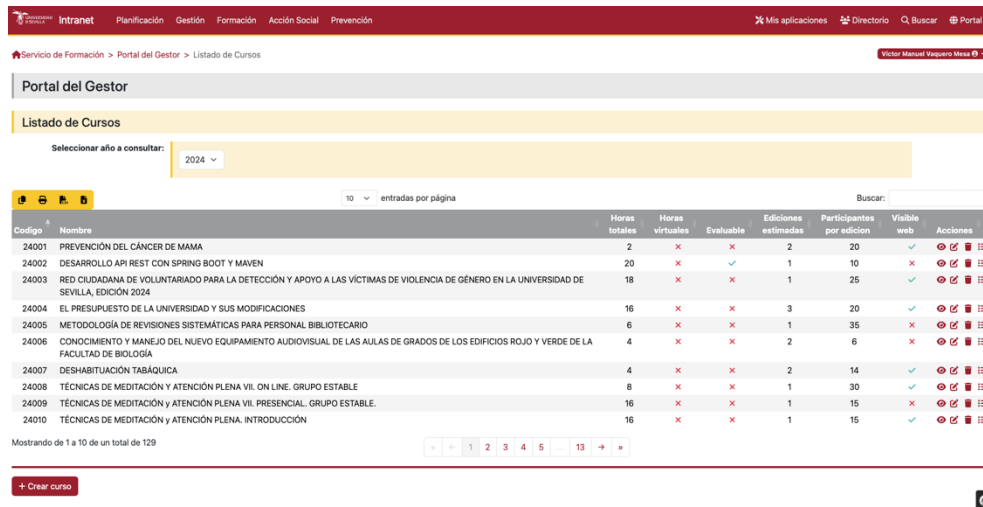
El punto de partida para el desarrollo del proyecto se remonta al verano anterior a la realización de este Trabajo de Fin de Grado, cuando surgió la oportunidad de proponer un sistema de gestión de formación continua que satisficiera ciertas necesidades detectadas en el ámbito laboral y, potencialmente, de la Universidad de Sevilla.

- **Origen y motivación.**

Durante el verano previo a la realización de este TFG, enfrenté la decisión de escoger entre abordar un desafío propuesto en la lista de ofertas del TFG o desarrollar un proyecto propio que diera sentido a los conocimientos adquiridos a lo largo de la carrera.

Tras reflexionar y recibir sugerencias en el ámbito laboral, decidí emprender un proyecto propio que me permitiera aprender un framework nuevo, ya en uso por algunos compañeros, y que al mismo tiempo resultara útil para la Universidad de Sevilla.

Así surgió la idea de desarrollar un sistema para gestionar la formación del Personal Técnico, de Gestión y de Administración y Servicios (PTGAS) de la USE.



Portal del Gestor

Listado de Cursos

Seleccionar año a consultar: 2024

10 entradas por página

Buscar:

Código	Nombre	Horas totales	Horas virtuales	Evaluable	Ediciones estimadas	Participantes por edición	Visible web	Acciones
24001	PREVENCIÓN DEL CÁNCER DE MAMA	2	X	X	2	20	✓	🔍 📄 📅
24002	DESARROLLO API REST CON SPRING BOOT Y MAVEN	20	X	✓	1	10	X	🔍 📄 📅
24003	RED CIUDADANA DE VOLUNTARIADO PARA LA DETECCIÓN Y APOYO A LAS VÍCTIMAS DE VIOLENCIA DE GÉNERO EN LA UNIVERSIDAD DE SEVILLA, EDICIÓN 2024	18	X	X	1	25	✓	🔍 📄 📅
24004	EL PRESUPUESTO DE LA UNIVERSIDAD Y SUS MODIFICACIONES	16	X	X	3	20	✓	🔍 📄 📅
24005	METODOLOGÍA DE REVISIONES SISTEMÁTICAS PARA PERSONAL BIBLIOTECARIO	6	X	X	1	35	X	🔍 📄 📅
24006	CONOCIMIENTO Y MANEJO DEL NUEVO EQUIPAMIENTO AUDIOVISUAL DE LAS AULAS DE GRADOS DE LOS EDIFICIOS ROJO Y VERDE DE LA FACULTAD DE BIOLOGÍA	4	X	X	2	6	X	🔍 📄 📅
24007	DESHABITUACIÓN TABÁQUICA	4	X	X	2	14	✓	🔍 📄 📅
24008	TÉCNICAS DE MEDITACIÓN Y ATENCIÓN PLENA VII. ON LINE. GRUPO ESTABLE	8	X	X	1	30	✓	🔍 📄 📅
24009	TÉCNICAS DE MEDITACIÓN Y ATENCIÓN PLENA VII. PRESENCIAL. GRUPO ESTABLE.	16	X	X	1	15	X	🔍 📄 📅
24010	TÉCNICAS DE MEDITACIÓN Y ATENCIÓN PLENA. INTRODUCCIÓN	16	X	X	1	15	✓	🔍 📄 📅

Mostrando de 1 a 10 de un total de 129

+ Crear curso

Figura 5.1: Portal del Gestor

- **Definición inicial del alcance.**

Se delimitó el alcance para crear una aplicación que ofreciera funcionalidades de gestión para administradores (gestores), formadores y participantes.

Se definieron objetivos concretos: permitir la creación de cursos y ediciones, la inscripción de usuarios, la posibilidad de registrar asistencias y calificaciones, y la emisión de certificaciones.

- **Herramientas de planificación.**

Se estableció un primer cronograma estimado, valorando los tiempos de aprendizaje, de diseño de la base de datos y de implementación.

Se optó por usar Zube.io para la gestión de historias de usuario y GitHub como repositorio centralizado.

## 5.2. Metodologías empleadas.

Dado que la gestión de requisitos y el desarrollo iterativo se convirtieron en factores clave para el éxito del proyecto, la metodología ágil, que permite incorporar cambios de manera flexible, se mostró como la opción más adecuada.

- **Scrum.**

Como herramienta orientada a la metodología ágil se decidió usar Zube.io, que destaca por su integración nativa con GitHub y su facilidad de uso en la gestión de proyectos. Dicha plataforma conecta directamente con los repositorios de GitHub, sincronizando issues, milestones, pull requests y etiquetas.

Cualquier cambio realizado en GitHub (por ejemplo, al cerrar un issue o añadir un comentario) se refleja automáticamente en Zube.io y viceversa, evitando la duplicidad de trabajo al no tener que gestionar las tareas en varias herramientas distintas.

Se optó por realizar sprints de 2 semanas de duración, eligiendo los lunes como día de inicio de cada sprint.

Zube.io posibilita la agrupación de historias de usuario relacionadas dentro de épicas, lo que facilita la organización de grandes funcionalidades o módulos de la aplicación y contribuye a alcanzar una visión más estratégica del proyecto, mostrando cómo cada historia de usuario encaja dentro de un objetivo mayor.

Debido a esto, cada sprint se planificó siguiendo este modelo de conceptualización y desarrollo mediante épicas, en las que se abordaba la parte de la aplicación que se estimaba más importante para la entrega del incremento planeado.

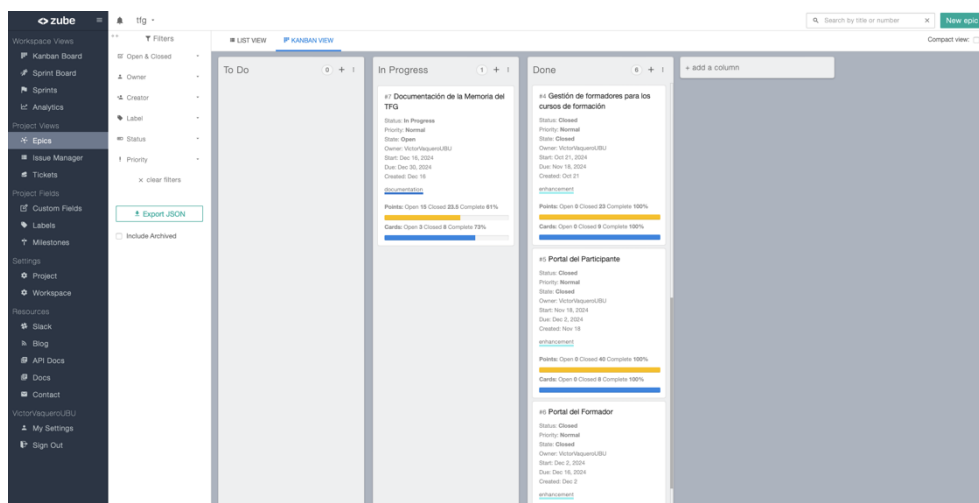


Figura 5.2: Épicas en zube.io

Estas épicas se descomponían en las distintas historias de usuario a desarrollar, otorgando sentido al conjunto.

Cada historia de usuario se implementaba en una rama propia que derivaba de la rama de desarrollo (developer), a la que se funcionaba posteriormente tras la comprobación de su correcto funcionamiento.

Al contrario de lo establecido en el desarrollo guiado por pruebas de software (Test Driven Development), donde primero se escribe una prueba (o test) para posteriormente implementar el código que la hace pasar satisfactoriamente y refactorizar dicho código, nosotros hemos optado por implementar los tests al finalizar cada sprint. De este modo, se detectan posibles errores o áreas de mejora en la revisión del sprint, lo que ayuda en la revisión del trabajo completado e identificar historias pendientes.

- **Kanban.**

Paralelamente, se utilizó un tablero Kanban para controlar el flujo de trabajo con estados en curso, en pruebas, o hecho.

Al compaginar Scrum con Kanban, se logró una visibilidad más granular del estado de cada tarea y se redujo la acumulación de trabajo sin finalizar.

- **Integración y despliegue continuo (CI/CD).**

Cada commit en la rama de desarrollo (developer) desencadenaba un pipeline de GitHub Actions que ejecutaba pruebas unitarias con PHPUnit. Solo cuando las pruebas eran satisfactorias al fusionar los cambios en la rama principal (main).

Elemento	Descripción
Sprint	Intervalo de 14 días para desarrollar y revisar historias de usuario.
Épicas	Conjunto de historias de usuario relacionadas que agrupan módulos o características.
Historias de usuario	Descripciones funcionales concretas de los requisitos, redactadas desde la perspectiva de cada rol (administrador, formador, participante).
Revisiones de sprint	Al final de cada sprint se validan los avances y se redefine la prioridad de las tareas venideras.
Retrospectivas	Espacios para identificar mejoras en el flujo de trabajo y el proceso de desarrollo.

Figura 5.3: Resumen elementos metodología Scrum adoptada

## 5.3. Formación

Durante la fase inicial del proyecto, se identificó la necesidad de adquirir o refrescar conocimientos en distintos aspectos técnicos y metodológicos. Particularmente, fue clave profundizar en la metodología ágil y en la integración de Zube.io con GitHub, así como en los nuevos entornos de desarrollo que se abordarían, entre los que destacan Symfony, Doctrine, Twig y otras herramientas del ecosistema PHP.

A continuación, se detallan los principales ámbitos formativos y los recursos consultados, clasificándolos según el tipo de tecnología o metodología estudiada.

- **Metodologías ágiles y herramientas de apoyo.**

### **Metodología ágil y Scrum.**

- Scrum: The Art of Doing Twice the Work in Half the Time (Jeff Sutherland)
- Official Scrum Guide de Ken Schwaber y Jeff Sutherland.
- Agile Estimating and Planning (Mike Cohn).

Se revisaron estos recursos para asentar las bases de la metodología ágil, entender en profundidad el ciclo de vida de un sprint, la importancia de las reuniones (Daily, Review, Retrospective) y la creación de épicas e historias de usuario.

### **Zube.io.**

- Documentación oficial de Zube.io
- Tutoriales y FAQs ofrecidos en la web de Zube.io, con ejemplos prácticos de integración con GitHub.

Se analizó cómo crear y organizar tableros Kanban, la manera de enlazar issues y pull requests desde GitHub y la metodología para configurar sprints y milestones de manera sincronizada.

### **GitHub y GitHub Actions.**

- Pro Git (Scott Chacon y Ben Straub).
- GitHub Actions Documentation.

Se repasaron conceptos sobre flujo de trabajo con branches, pull requests, etiquetado de versiones (tags), gestión de incidencias y el uso de GitHub Actions para la configuración de pipelines de integración continua.

- **Formación técnica en Symfony y entorno PHP.**

El siguiente bloque formativo se centró en Symfony 7, marco de desarrollo principal, y sus librerías asociadas. La curva de aprendizaje incluyó temas como la arquitectura MVC, el uso de plantillas Twig y la configuración de la seguridad.

**Symfony 7.**

- Symfony Documentation: guía oficial con conceptos como controllers, routing, services, event listeners y configuración de la seguridad (security.yaml).
- Symfony Best Practices: buenas prácticas recomendadas por el equipo de Symfony, con énfasis en la estructura de directorios y la organización del código.

Se puso especial atención en los principios de la arquitectura MVC y en cómo Symfony los implementa, así como en la forma de crear controllers y manejar rutas.

**Doctrine ORM.**

- Doctrine ORM Documentation: abarca la configuración de entidades, las relaciones (OneToMany, ManyToMany), las migraciones y el ciclo de vida de los objetos.
- Tutoriales sobre Doctrine migrations y patrones de repositorio para organizar consultas complejas.

Era fundamental aprender a modelar la base de datos de manera eficiente, comprendiendo el mapeo objeto-relacional y las mejores prácticas para crear y gestionar tablas, índices y consultas optimizadas.

**Twig y AssetMapper.**

- Twig Documentation: explica la sintaxis del motor de plantillas y los filtros más habituales.
- Symfony AssetMapper: recurso oficial para la gestión de dependencias de frontend sin necesidad de npm o yarn.

**Otras librerías y herramientas.**

- Composer para la gestión de dependencias en PHP.
- PHPUnit y PHPStan para la ejecución de pruebas unitarias y análisis estático de código, respectivamente.

- **Formación en otras tecnologías complementaria.**

Aunque el núcleo del proyecto recayó en Symfony, se consideró igualmente necesaria la formación en varias tecnologías complementarias que contribuyeron a mejorar el frontend, la usabilidad y el rendimiento de la aplicación:

**Bootstrap.**

- Bootstrap Documentation.

Ejemplos de plantillas y tutoriales en línea para el uso de componentes de interfaz (botones, menús, modales, alertas, etc.).

**Herramientas de Testing y CI/CD.**

- GitHub Actions: configuración de workflows de integración continua para ejecutar tests y revisiones automáticas.
- Xdebug para depurar y generar reportes de cobertura en PHPUnit.

## 5.4. Desarrollo de la lógica de negocio.

En este apartado se explica la lógica de negocio que respalda las funcionalidades clave de la aplicación. Aunque en un proyecto web con Symfony la mayor parte de la “lógica” está directamente relacionada con la interacción con la base de datos y el frontend, se pueden destacar algunos “algoritmos” o procesos específicos.

- **Cálculo de plazas disponibles.**

Al crear una nueva edición, se define el número máximo de participantes permitidos.

Cada vez que un participante solicita la inscripción, se verifica que no se haya alcanzado el límite y que el participante cumpla las condiciones (unidad no vacía, no estar inscrito ya en el mismo curso, etc.).

- **Generación de códigos de curso y edición.**

**Códigos de curso:** conformados por los 2 dígitos del año + un número incremental de 3 dígitos (por ejemplo, 24001).

```
47  /**
48   * Encuentra el primer código de curso libre en función del año.
49   *
50   * @param int $year Año para el cual se desea obtener el primer código de curso libre (por ejemplo, 2024).
51   * @return string|null Retorna el primer código libre en el formato "aaXXX" o null si no hay códigos disponibles.
52   */
53  1 usage: ± Victor M Vaquero
54  public function findPrimerCodigoCursoLibre(int $year): ?string
55  {
56      $prefix = substr((string)$year, -2);
57
58      $result = $this->createQueryBuilder( 'alias: 'c')
59          ->select( ...select: 'c.codigo_curso')
60          ->where( ...predicates: 'c.codigo_curso LIKE :prefix')
61          ->setParameter( key: 'prefix', value: $prefix . '%' )
62          ->orderBy( sort: 'c.codigo_curso', order: 'ASC')
63          ->getQuery()
64          ->getResult();
65
66      $cursosExistentes = array_map(function($item) use ($prefix) {
67          return (int) substr($item['codigo_curso'], strlen($prefix));
68      }, $result);
69
70      $siguienteNumeroLibre = 1;
71      foreach ($cursosExistentes as $numero) {
72          if ($numero !== $siguienteNumeroLibre) {
73              break;
74          }
75          $siguienteNumeroLibre++;
76      }
77
78      // Formateamos el número encontrado en el formato "aaXXX" y lo retornamos
79      return sprintf( format: '%s%03d', $prefix, $siguienteNumeroLibre);
80  }
```

Figura 5.4: lógica de negocio del código de curso

**Códigos de edición:** construcción de 5 dígitos del curso + “/” + un número incremental de 2 dígitos para enumerar las ediciones (por ejemplo, 24001/01).

```
69  /**
70   * Método para obtener el primer código de edición libre.
71   *
72   * @param string $codigoCurso Código del curso
73   * @return string|null Retorna el primer código de edición libre en el formato "codigoCurso/XX"
74   */
75  1 usage: ± Victor M Vaquero
76  public function findPrimerCodigoEdicionLibre(string $codigoCurso): ?string
77  {
78      $result = $this->createQueryBuilder( 'alias: 'e')
79          ->select( ...select: 'e.codigo_edicion')
80          ->join( join: 'e.curso', alias: 'c')
81          ->where( ...predicates: 'c.codigo_curso = :codigoCurso')
82          ->setParameter( key: 'codigoCurso', $codigoCurso)
83          ->orderBy( sort: 'e.codigo_edicion', order: 'ASC')
84          ->getQuery()
85          ->getResult();
86
87      $edicionesExistentes = array_map(function($item) use ($codigoCurso) {
88          return (int) substr($item['codigo_edicion'], offset: strlen($codigoCurso) + 1);
89      }, $result);
90
91      $siguienteNumeroLibre = 0; // Comenzamos desde 0
92      foreach ($edicionesExistentes as $numero) {
93          if ($numero !== $siguienteNumeroLibre) {
94              break;
95          }
96          $siguienteNumeroLibre++;
97      }
98
99      return sprintf( format: '%s/%02d', $codigoCurso, $siguienteNumeroLibre);
100  }
```

Figura 5.5: lógica de negocio del código de edición

Se implementaron validaciones en el Controller correspondiente para asegurar la unicidad de los códigos.



- **Algoritmo de cierre y certificación.**

El formador marca la edición como finalizada cuando registra la asistencia y calificaciones. El gestor revisa y “certifica” la edición, generando automáticamente los certificados para los participantes que cumplan requisitos de asistencia y/o calificación.

## 5.5. Desarrollo de la aplicación

- **Sprint 0.**

El objetivo del sprint 0 se centró en la formación, requisitos y tener un entorno de desarrollo totalmente configurado y preparado para comenzar con el desarrollo del proyecto web en futuros sprints.

Los objetivos generales del sprint 0 fueron:

- Preparar el entorno de desarrollo.
- Integrar herramientas de gestión y control de versiones.
- Configurar la base del proyecto web de formación en Symfony.

Las herramientas necesarias fueron instaladas y configuradas correctamente, y se estableció un flujo de trabajo con:

- GitHub como plataforma de desarrollo colaborativo.
- Zube.io como plataforma de gestión de proyectos para equipos de desarrollo ágiles.
- PHPUnit como framework de pruebas unitarias para el lenguaje de programación PHP.
- PHPStan como analizador estático de código.

- **Sprint 1.**

El sprint 1 corresponde al primer sprint de desarrollo del proyecto de gestión de formación. A continuación se detallan los objetivos alcanzados, las funcionalidades implementadas, las configuraciones realizadas y los problemas corregidos.

### **Objetivo del Sprint.**

Configurar el entorno de desarrollo e implementación de la estructura básica de la aplicación de gestión de cursos. Durante el sprint, se abordaron las funcionalidades principales de creación, visualización, edición y eliminación de cursos y ediciones de cursos pertenecientes al rol gestor, así como la integración de herramientas de calidad y pruebas.

### **Funcionalidades Implementadas.**

- Gestión de Cursos.

Creación de la entidad Curso con sus campos necesarios (código, nombre, duración, objetivos, contenidos, etc.).

Implementaciones CRUD para cursos, permitiendo ver, crear, editar y eliminar cursos.

Visualización de la lista de cursos disponibles para su gestión.

- Gestión de Ediciones de Cursos.

Creación de la entidad Edición, que representa las ediciones específicas de cada curso.

Implementaciones CRUD para ediciones de cursos, permitiendo gestionar las fechas, modalidad, lugar de impartición y plazas disponibles.

Validación de campos obligatorios (fechas, modalidad, plazas) y configuración del código automático para nuevas ediciones basado en el curso asociado.

### **Mejoras y Correcciones Técnicas.**

- Calidad de Código y Pruebas Automáticas

Configuración de PHPStan y PHPUnit para el análisis de calidad de código y la cobertura de pruebas.

Resolución de todos los errores detectados por PHPStan, incluidos problemas de tipos y consistencia en las entidades Curso y Edicion.

Generación de un reporte de cobertura de código con PHPUnit, garantizando una amplia cobertura y corrección de errores en pruebas unitarias.

- Optimización de Interfaces y Formularios.

Deshabilitación de los campos en formularios de edición de ediciones de cursos cuando el código de edición termina en "00".

Ajuste automático del siguiente número disponible en `codigo_edicion` al crear una nueva edición de curso.

- Corrección de Problemas de Renderizado en Datatables

Solución a problemas de renderizado de botones, buscador y selector en Datatables mediante `window.location.href` para forzar recargas completas de página.

### **Documentación y Procedimientos de Integración Continua**

Creación de un archivo de configuración phpstan.dist.neon para estandarizar el análisis en GitHub Actions.

Integración de GitHub Actions para ejecutar PHPStan en cada push a las ramas developer y main, garantizando una calidad de código consistente.

Revisión de configuración de .env y otros ajustes para asegurar el correcto funcionamiento de las pruebas en el pipeline de CI/CD.

- **Sprint 2.**

El sprint 2 corresponde al segundo sprint de desarrollo del proyecto de gestión de formación. A continuación, se detallan los objetivos alcanzados, las funcionalidades implementadas, las configuraciones realizadas y las pruebas exitosamente ejecutadas.

#### **Objetivo del Sprint**

Consolidar la gestión de cursos y ediciones implementadas en el primer sprint, añadiendo funcionalidades completas para la gestión de participantes y formadores. Durante este sprint, se finalizaron las pruebas automatizadas con PHPUnit y PHPStan, asegurando la calidad del código y el correcto funcionamiento de las nuevas funcionalidades.

#### **Funcionalidades Implementadas.**

- Gestión de Participantes.

Creación de la entidad Participante con sus campos necesarios (nif, nombre, apellidos, unidad, centro, plaza, etc.).

Implementaciones CRUD para participantes, permitiendo consultar, crear, editar y eliminar participantes.

Implementaciones CRUD para inscripciones de los participantes en las distintas ediciones de los cursos, permitiendo consultar, crear, editar y eliminar inscripciones a ediciones no comenzadas, y baja justificada para aquellas ediciones ya comenzadas.

- Gestión de Formadores.

Creación de la entidad Formador con sus campos necesarios (nif, nombre, apellidos, organización, etc.).

Implementaciones CRUD para formadores, permitiendo consultar, crear, editar y eliminar formadores.

Implementaciones CRUD para asignaciones de los formadores en las distintas ediciones de los cursos, permitiendo consultar, crear, editar y eliminar asignaciones a ediciones, indicando las retribuciones, horas impartidas y otros datos clave.

- Mejoras en la Gestión de Ediciones.

Validación avanzada de campos obligatorios y manejo de fechas de impartición de ediciones.

Mejoras en la creación automática del código de los cursos y de las ediciones.

Optimización de interfaces de usuario.

### **Calidad de Código y Pruebas Automáticas.**

- Pruebas con PHPUnit.

Cobertura total para las funcionalidades implementadas en las entidades Participante y Formador.

Nuevas pruebas unitarias y funcionales para:

- a) Inscripción de participantes.
- b) Asignación de formadores a ediciones.
- c) Validación de datos en ediciones y cursos.

- Análisis Estático con PHPStan.

Resolución de todos los errores detectados por PHPStan.

Configuración de GitHub Actions para ejecutar PHPStan automáticamente en cada push a developer y main.

- Integración Continua con GitHub Actions

Pipeline automatizado para pruebas de PHPUnit y análisis de PHPStan.

Badge dinámico en el README.md para reflejar el estado de las pruebas y el análisis estático.

### **Problemas Solucionados.**

Corrección en Datatables mediante la desinstalación de Symfony UX Turbo.

Se solucionaron problemas de renderizado en tablas con datos complejos.

Mejoras en la generación de códigos automáticos:

Ajuste en la lógica de asignación de códigos de edición para evitar conflictos.

Optimización de formularios:

Mejoras en la usabilidad de formularios de inscripción y asignación.

- **Sprint 3.**

El sprint 3 corresponde al tercer sprint de desarrollo del proyecto de gestión de formación. A continuación, se detallan los objetivos alcanzados, las funcionalidades implementadas, las configuraciones realizadas y las pruebas exitosamente ejecutadas.

**Objetivo del Sprint.**

Desarrollar y consolidar la gestión de usuarios, incluyendo la autenticación y asignación de roles, y extender las funcionalidades del sistema para soportar una lógica más robusta en las inscripciones de participantes y asignaciones de formadores. Durante este sprint, se llevaron a cabo pruebas automatizadas y análisis estáticos para garantizar la calidad y el correcto funcionamiento del sistema.

**Funcionalidades Implementadas.**

Gestión de Usuarios y Autenticación.

- a) Creación de la entidad User con los campos email, password (almacenada de manera segura mediante hashing) y roles (almacenados en formato JSON).
- b) Implementación de un formulario de login y acceso de usuarios autenticados. Según sus roles, Administradores (ROLE\_ADMIN), tendrán acceso al Portal del Gestor, y Participantes (ROLE\_USER), tendrán acceso al Portal del Participante.
- c) Formulario de registro para nuevos usuarios con validación de datos y asignación inicial de roles.

**Gestión de Inscripciones.**

Inscripciones a ediciones de cursos para participantes con las siguientes características:

- a) Validación de plazas disponibles.
- b) Cancelaciones de inscripción y cambios de ediciones según fechas de comienzo de estas.
- c) Inscripciones a ediciones según casuística (fecha de comienzo de la edición, estado laboral...)

**Modificación de datos de contacto.**

El participante podrá modificar los datos de contacto según su necesidad.

### **Visualización del estado de sus inscripciones y certificados formativos.**

El participante podrá ver el listado de ediciones en las que ha realizado inscripción y el estado de estas (pendiente de datos, número de certificado del curso, no certificado).

### **Configuración general de la aplicación.**

Configuración de Mailer para enviar correos desde el sistema utilizando un servidor de correos local (Postfix).

Configuración del tiempo de expiración de contraseña (1 año por defecto).

Ajustes en la lógica de generación automática de códigos de ediciones.

Mejoras en la validación de formularios.

### **Calidad de Código y Pruebas Automáticas.**

- Pruebas con PHPUnit

66 pruebas unitarias ejecutadas con éxito, cubriendo funcionalidades de gestión de usuarios, inscripciones y asignaciones de formadores.

Resolución de todos los errores reportados por PHPStan.

Código completamente validado según los estándares de calidad.

Actualización de badges en el README.md para reflejar el estado del pipeline.

### **Problemas Solucionados.**

Corrección de errores en el envío de correos durante el registro de usuarios.

Mejora en la lógica de redirección según roles después de la autenticación.

Ajuste en la generación de códigos de edición para evitar conflictos.

Solución a problemas de validación en formularios complejos.

### **• Sprint 4.**

El sprint 4 corresponde al cuarto y último sprint de desarrollo del proyecto de gestión de formación. Se han implementado funcionalidades clave relacionadas con el Portal del Formador, la gestión de sesiones, el registro de asistencia y mejoras generales en la calidad del código. A continuación, se detallan los objetivos alcanzados, funcionalidades implementadas y pruebas realizadas.

### **Objetivo del Sprint.**

Desarrollar el Portal del Formador, facilitando la gestión de ediciones abiertas y cerradas asignadas a los formadores. Se incorporaron nuevas entidades y formularios para registrar y validar sesiones, asistencias y calificaciones, permitiendo una gestión integral del proceso formativo. Además, se aseguraron los estándares de calidad del código mediante pruebas unitarias y análisis estático.

### **Funcionalidades Implementadas.**

- Portal del Formador.

Mis Datos:

Formulario donde poder modificar los datos de contacto del formador (correo y teléfono)

Mis Cursos:

Visualización de ediciones asignadas al formador, separadas en:

- a) Ediciones Abiertas (estado = 0): permite gestionar sesiones y asistencias.
- b) Ediciones Cerradas: visualización de datos registrados, sin posibilidad de edición.

Gestión de Sesiones:

- a) Creación de sesiones con los campos: fecha, hora de inicio, duración, tipo (presencial/virtual) y observaciones.
- b) Registro de asistencia de participantes a las sesiones.
- c) Introducción de calificaciones, si el curso es calificable
- c) Validación de datos antes de completar la sesión y actualizar el estado de la edición.

- Portal del Gestor.

Ediciones para Certificar.

Listado de ediciones cerradas por su formador pendiente de revisión y emisión de certificados.

### **Calidad de Código y Pruebas Automáticas.**

- Pruebas con PHPUnit.

107 pruebas unitarias ejecutadas con éxito, cubriendo funcionalidades de gestión de formadores, ediciones asignadas y control de sesiones, asistencias y calificaciones.

- Resolución de todos los errores reportados por PHPStan.

### **Problemas Solucionados.**

Validación de horas y sesiones antes de cerrar ediciones.

Corrección de errores en formularios complejos de gestión de sesiones, asistencias y calificaciones.

Mejora en el flujo de datos entre entidades relacionadas (Formador, Sesión, Asistencia).

### **Configuraciones y Mejoras Adicionales.**

Actualización del README.md con las funcionalidades del Portal del Formador y cambios en el sprint actual.

Refuerzo de validaciones en formularios y estados de ediciones.

### **Resumen del Sprint.**

Con este sprint, se finaliza el desarrollo del Portal del Formador y la funcionalidad principal del proyecto. El sistema incluye la gestión completa de cursos, ediciones, inscripciones de participantes y asignación de formadores, con validaciones robustas y alta calidad de código.

### **• Sprint 5 y 6.**

Los sprint 5 y 6 serán 2 sprint destinados a la creación de documentación del proyecto y correcciones de bugs encontrados.

### **Objetivo del Sprint 5.**

A lo largo de este sprint se va a desarrollar la memoria del proyecto y la preparación de un entorno de producción donde albergar el proyecto para su posterior uso acceso por los miembros de tribunal.

### **Problemas Solucionados.**

A lo largo de la creación de la memoria se han detectado 5 bugs en la aplicación que se han corregido a lo largo del sprint 5.

- Se corrige el error en el que no se activaba como calificable un curso con horas virtuales #78.
- Se corrige la situación de una baja justificada durante la introducción de asistencias y calificaciones en el portal del formador #77.



- Se corrige la grabación, en el campo dirección de la tabla participante\_edicion, de la dirección IP desde donde el alumno realiza la inscripción a la edición o, en su defecto, el username con ROLE\_ADMIN del usuario que ha realizado esa inscripción #76.
- Se corrige un error en el campo horas de un curso y en horas\_virtuales para que permita cursos sin horas enteras (i.e. curso de 4,5 horas) #75.

### **Objetivo del Sprint 6.**

A lo largo de este sprint se va a desarrollar los anexos del proyecto y los videos de presentación del proyecto para el tribunal.

## **5.6. Pruebas sobre el código.**

Uno de los pilares fundamentales para garantizar la calidad y fiabilidad del proyecto ha sido la realización de pruebas automatizadas mediante PHPUnit y el análisis estático del código con PHPStan. Estas herramientas han permitido validar tanto la funcionalidad como la calidad del código en los diferentes componentes de la aplicación.

- **Pruebas unitarias con PHPUnit.**

Las pruebas unitarias realizadas con PHPUnit se centraron en validar el comportamiento de los métodos implementados en las diferentes clases del proyecto. Las métricas de cobertura obtenidas son las siguientes:

### **Resumen general.**

Se alcanzaron los siguientes porcentajes de cobertura de código:

Clases: 71.74% (33/46)

Métodos: 91.77% (379/413)

Líneas: 96.13% (2288/2380)

Estos resultados reflejan un nivel alto de cobertura, especialmente en los métodos y líneas de código, garantizando la verificación de los casos más críticos del sistema.

### **Cobertura por módulo.**

A continuación, se detalla la cobertura obtenida en los módulos principales del proyecto:

Controladores con una cobertura del 100% en métodos y líneas para la mayoría de los controladores, destacando CursoController, EdicionController, y ParticipantePortalController.

Entidades: Cobertura variable según la complejidad de las clases. Por ejemplo:

Edicion: 93.94% de métodos y 86.25% de líneas.

FormadorEdicion: 77.14% de métodos y 69.23% de líneas.

Formularios: La mayoría de los formularios alcanzaron una cobertura del 100%, excepto algunos casos puntuales como SesionType (50% de métodos) debido a funciones no utilizadas.

Repositorios: Cobertura del 100% en métodos y líneas en la mayoría de los repositorios, destacando ParticipanteRepository y SesionRepository.

- **Análisis estático con PHPStan.**

El análisis estático del código se realizó con PHPStan, configurado en un nivel alto de análisis (nivel 8). Esta herramienta ayudó a identificar problemas potenciales como:

- Incompatibilidades de tipo.
- Métodos no utilizados.
- Errores comunes en las anotaciones de tipo en las entidades y formularios.

Tras las correcciones, PHPStan no detectó errores significativos, lo que refleja un código robusto y bien estructurado.

- **Conclusiones de las pruebas**

Los resultados obtenidos en las pruebas evidencian un alto nivel de confianza en la funcionalidad y la calidad del código. Algunos puntos que destacar:

Cobertura alta en módulos críticos: los controladores y formularios, que contienen gran parte de la lógica de la aplicación, tienen una cobertura cercana al 100%, lo que asegura que las funcionalidades principales han sido ampliamente probadas.

Cobertura moderada en entidades: La cobertura de métodos y líneas en las entidades podría incrementarse mediante pruebas adicionales que contemplen casos límite o menos comunes.

Ventajas del análisis estático: PHPStan ha sido una herramienta clave para garantizar la calidad del código, reduciendo la posibilidad de errores en producción.

El uso combinado de PHPUnit y PHPStan ha permitido identificar y solucionar errores en fases tempranas del desarrollo, lo que se traduce en una mayor estabilidad del sistema y una experiencia más confiable para los usuarios finales.

---

## **6.Trabajos relacionados**

---

---

## **7. Conclusiones y Líneas de trabajo futuras**

---

Texto general

### **7.1. Conclusiones**

### **7.2. Líneas de trabajo futuras**

---

## **Bibliografía**

---