Trabajo Final de Grado – Gestion de Empresas Colaboradoras

- Autor: Luis Angel
- Tutora: Elena

- Fecha: 14/12/2025
- Repositorio: https://github.com/lmendez861/TFG-gora

1. Portada

Titulo: Gestion de Empresas Colaboradoras para FP Dual

Autor: Luis Angel
Tutora: Elena

Fecha: 14/12/2025

2. Agradecimientos

[Espacio para reconocer a centro, tutora, familia y empresas colaboradoras.]

3. Indice

[Se generara al maquetar el documento final.]

4. Resumen

4.1 Resumen (ES)

Plataforma web para centralizar empresas colaboradoras, convenios, tutores y estudiantes en FP Dual. Se redisenyo el frontend en React/TypeScript (`frontend/app/src/App.tsx`) con modulos de dashboard, CRUD y seguimiento, y se reconfiguro un backend Symfony para el nuevo dominio. Incluye autenticacion basica (admin/admin123 temporal), roles previstos y paneles con metricas de convenios y asignaciones. Se busca trazabilidad, alertas de caducidad, exportes y cumplimiento RGPD.

4.2 Summary (EN)

Web platform to centralize partner companies, agreements, mentors, and students for dual training. The React/TypeScript frontend (`frontend/app/src/App.tsx`) was redesigned with dashboard, CRUD, and follow-up modules, and a Symfony backend was reconfigured for the new domain. It includes basic authentication (temporary admin/admin123), planned roles, and dashboards with agreement and assignment metrics. Aims for traceability, expiry alerts, exports, and GDPR compliance.

5. Antecedentes / Introduccion

- Gestion previa con hojas de calculo y correos, sin trazabilidad ni alertas.
- Proyecto Agora (legacy) archivado; se reutiliza infraestructura y estilos cuando aportan valor.
- Nueva tematica centrada en gestion de empresas colaboradoras y practicas.
- Documentacion de dominio en `docs/domain-model.md` ; plan de refactor en `docs/refactor-plan.md` .
- Ambito: departamentos FCT/DUAL, tutores academicos y profesionales, coordinacion y empresas.

6. Objetivos y alcance

6.1 Objetivo general

Entregar una plataforma web que centralice convenios, asignaciones y seguimiento de estudiantes con control de permisos y trazabilidad.

6.2 Objetivos especificos

1. Modelar y exponer entidades Empresa, Convenio, Tutor (academico/profesional), Estudiante, Asignacion y Seguimiento.
2. Implementar autenticacion y autorizacion por roles (admin, coordinador, tutor centro, tutor empresa).
3. Ofrecer dashboards con KPI y alertas de caducidad; flujo de aprobacion de solicitudes de empresas.
4. Proveer formularios CRUD con validaciones y subida de documentos asociados.
5. Exportes (CSV/PDF) y checklist RGPD (consentimientos, minimizacion, logs de acceso).

6.3 Alcance

- Dentro: CRUD completo, gestion de solicitudes de empresas, registro de visitas/seguimientos, notificaciones internas basicas, auditoria minima.
- Fuera: firma digital avanzada, integracion ERP/SGA, app movil nativa, pasarela de pagos.

7. Definiciones

- Convenio: acuerdo formal centro–empresa para acoger estudiantes.
- Asignacion: vinculo estudiante–convenio–empresa con tutores designados.
- Seguimiento: registro de tutoria (visita, reunion, incidencia).
- Rol: perfil de permisos aplicado a un usuario autenticado.

8. Notaciones y simbolos

- CRUD, RGPD, JWT, FCT/DUAL, KPI.
- Estados UI: convenios (vigente, borrador, renovacion, planificado), empresas (activa, pendiente_revision), asignaciones (en_curso, planificada, finalizada).

9. Desarrollo del trabajo final

9.1 Analisis de requisitos

- Actores: coordinador, tutor academico, tutor profesional, estudiante, empresa.
- Casos de uso (segun `frontend/app/src/App.tsx`): dashboard con KPI; listado/detalle de empresas y documentos; aprobacion/rechazo de solicitudes con mensajeria; CRUD de convenios, estudiantes y asignaciones via modales; gestion de tutores con paginacion/filtros; seccion de documentacion; perfil admin con atajos.
- Requisitos no funcionales: responsive, trazabilidad minima, seguridad por roles, alertas de caducidad, exportes.

9.2 Diseno de la solucion

- Frontend: React + TypeScript + Vite; componentes de tabla, formularios y modales; hooks para estado/carga; rutas protegidas basicas; estilos `App.css` e `index.css`.
- Backend: Symfony (`backend/`), entidades Doctrine del nuevo dominio creadas; pendiente controladores REST, autenticacion JWT y validaciones.
- Datos mock en `App.tsx` para desarrollo offline (empresas, convenios, estudiantes, asignaciones).

- Arquitectura: SPA que consumira API REST; futura integracion JWT/roles.

9.3 Modelo de datos

- Entidades clave (ver `docs/domain-model.md`): EmpresaColaboradora, Contacto/TutorProfesional, Convenio, TutorAcademico, Estudiante, AsignacionPractica, Seguimiento, EvaluacionFinal.
- Relaciones: Empresa 1–N Convenios; Convenio 1–N Asignaciones; Asignacion une estudiante + tutores + convenio; Seguimientos N por Asignacion.

9.4 Seguridad y cumplimiento

- Estado actual: login demo admin/admin123 en frontend.
- Plan: RBAC, expiracion de tokens, logs de acceso y cambios; minimizacion de datos en vistas; registro de consentimientos RGPD.

9.5 Diseno de interfaz

- Dashboard con KPI y tarjetas de riesgo por caducidad.
- Listas con DataTable, chips de estado y filtros; modales de alta/edicion.
- Panel de solicitudes con workflow de aprobacion y mensajes por solicitud.
- Panel de tutores con paginacion independiente; topbar con notificaciones.
- Tema visual oscuro (`app--dark`) y layout con topbar y acciones rapidas.

9.6 Implementacion

- Frontend (`frontend/app/src/App.tsx`): hooks de carga, CRUD via `services/api` , modales `components/*Form.tsx` , toasts y estados de error/carga, paginacion de tutores, notificaciones en topbar.
- Backend: estructura y entidades listas; pendiente controladores REST, seguridad JWT, exportes y almacenamiento de documentos.

9.7 Pruebas y validacion

- Plan: casos UI (creacion/edicion/borrado), validacion de fechas de convenios, permisos por rol, caducidad de tokens, exportes.
- Estado: pruebas manuales sobre datos mock; pendiente automatizar (unitarias backend, E2E frontend).

9.8 Despliegue y operacion

- Entorno previsto: contenedores (Symfony + DB + frontend estatico).
- Configuracion: variables `.env` para DB y claves JWT; backups; logs centralizados.
- Entrega: PDF de memoria y demo navegable.

9.9 Resultados y metricas (14/12/2025)

- Frontend: dashboard, CRUDs, flujo de solicitudes y panel de tutores operativos con datos mock.
- Backend: estructura lista y entidades definidas; controladores y autenticacion en curso.
- Objetivos: 100% convenios con fecha de caducidad; >95% seguimientos registrados; 0 accesos no autorizados en pruebas.

9.10 Limitaciones y riesgos

- Autenticacion real pendiente; riesgos RGPD si se cargan datos reales antes de roles.
- Migracion de datos legacy; dependencias legales de convenios.
- Escalado de listados sin paginacion server-side (optimizar API).

10. Conclusiones y recomendaciones

- Valor: centralizacion de convenios y practicas, alertas tempranas, trazabilidad de asignaciones y comunicaciones.
- Recomendaciones inmediatas: cerrar API REST y JWT, proteger rutas, habilitar subida segura de documentos y exportes; pruebas E2E basicas; acordar con la tutora los estados finales de flujo (aprobacion, caducidad, renovacion).
- Futuro: firma digital, integracion ERP/SGA, notificaciones push/email, checklist RGPD automatizado.

11. Referencias

- `docs/domain-model.md`, `docs/refactor-plan.md`, `docs/TFG_MEMORIA_PLANTILLA.md`.
- Documentacion Symfony, React/TypeScript, RGPD (UE 2016/679).
- Normativa FCT/DUAL del centro/autonomica.

12. Bibliografia

- Recursos sobre gestion de practicas, diseno de APIs REST, seguridad web y accesibilidad.

13. Anexos

- Diagramas ER y de secuencia.
- Capturas de dashboard, listas y modales.
- Casos de prueba ejecutados y resultados.
- Checklist RGPD aplicado.