

GALITROCO - Sistema de Intercambio de Habilidades

Autor: Antonio Campos

Universidad: Universitat Oberta de Catalunya (UOC)

Asignatura: Trabajo Final de Máster

Fecha: Octubre-Noviembre 2025

Versión: 1.0 (PEC2)

DESCRIPCIÓN DEL PROYECTO

GaliTroco es una plataforma web que permite a los usuarios de Galicia intercambiar habilidades y servicios sin necesidad de dinero. Un usuario puede ofrecer sus conocimientos (programación, idiomas, cocina, etc.) a cambio de aprender otras habilidades de la comunidad.

Características principales:

- ☒ Sistema de autenticación y registro de usuarios
 - ☒ Publicación de habilidades (ofertas y demandas)
 - ☒ Propuestas de intercambio entre usuarios
 - ☒ Sistema de valoraciones con estrellas (1-5)
 - ☒ Panel de administración para moderación
 - ☒ Sistema de reportes de contenido inapropiado
 - ☒ Notificaciones en tiempo real
 - ☒ Recuperación de contraseña por email
-

TECNOLOGÍAS UTILIZADAS

Backend

- **Lenguaje:** PHP 8.2
- **Servidor Web:** Apache 2.4
- **Base de Datos:** PostgreSQL 15 (Supabase)
- **Autenticación:** JWT + Sesiones PHP
- **Email:** Resend API
- **Deploy:** Render.com (Docker)

Frontend

- **Framework:** Angular 19.2.0
 - **Lenguaje:** TypeScript 5.7.2
 - **UI:** Angular Material 19.2.19
 - **HTTP Client:** HttpClient con RxJS
 - **Deploy:** Render.com (Static Site)
-

Infraestructura

- **Repositorio:** GitHub (tonikampos/render-test-php)
 - **CI/CD:** Auto-deploy desde GitHub main branch
 - **Base de datos:** Supabase Cloud PostgreSQL
-

URLS DEL PROYECTO DESPLEGADO

Producción (Render.com) ☒ OPERATIVO

- **Backend API:** <https://render-test-php-1.onrender.com/api.php>
- **Frontend Angular:** <https://galitroco-frontend.onrender.com> (*desplegar según instrucciones*)
- **Base de Datos:** Supabase PostgreSQL 15 (cloud)

Estado: Ambos servicios desplegados en Render.com con auto-deploy desde GitHub.

Testing local (Opcional)

- **Backend:** <http://localhost/probatfm/backend/api/>
 - **Frontend:** <http://localhost:4200>
 - **Base de Datos:** PostgreSQL local o Supabase (recomendado)
-

REQUISITOS PREVIOS

Para ejecutar el backend:

- PHP 8.2 o superior
- Apache 2.4+ con mod_rewrite
- PostgreSQL 15+
- Composer (opcional, si se usan dependencias)
- Extensiones PHP: `pdo_pgsql`, `json`, `session`

Para ejecutar el frontend:

- Node.js 18+ (recomendado: 20.x)
- npm 9+ o 10+
- Angular CLI 19 (`npm install -g @angular/cli`)

Para la base de datos:

- PostgreSQL 15+
 - Cliente SQL (psql, pgAdmin, DBeaver, etc.)
-

INSTRUCCIONES DE INSTALACIÓN

1 CLONAR O DESCOMPRIMIR EL PROYECTO

Si tienes acceso al repositorio:

```
git clone https://github.com/tonikampos/render-test-php.git galitroco
cd galitroco
```

Si tienes el ZIP:

```
unzip PEC2_pry_Campos_Antonio.zip
cd PEC2_pry_Campos_Antonio
```

2 CONFIGURAR LA BASE DE DATOS

Opción A: Usar Supabase (Recomendado para PEC2)

La aplicación ya está configurada para usar Supabase. Solo necesitas las credenciales correctas en `backend/config/database.php`.

Opción B: PostgreSQL Local

1. Crear base de datos:

```
CREATE DATABASE galitroco_tfm;
```

2. Ejecutar esquema:

```
psql -U postgres -d galitroco_tfm -f database/schema.sql
```

3. Cargar datos de prueba:


```
psql -U postgres -d galitroco_tfm -f database/seeds.sql
```

4. Configurar credenciales:

Editar `backend/config/database.php`:

```
return [
    'host' => 'localhost',
    'port' => '5432',
    'dbname' => 'galitroco_tfm',
    'user' => 'postgres',
```

```
'password' => 'tu_password'
];
```

 **Más detalles:** Ver [database/README_DATABASE.md](#)

3 CONFIGURAR EL BACKEND

1. Configurar Apache Virtual Host (ejemplo):

```
<VirtualHost *:80>
    ServerName galitroco.local
    DocumentRoot "C:/xampp/htdocs/probatfm"

    <Directory "C:/xampp/htdocs/probatfm">
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>
</VirtualHost>
```

2. Actualizar archivo hosts (Windows: `C:\Windows\System32\drivers\etc\hosts`):

```
127.0.0.1 galitroco.local
```

3. Configurar variables de entorno:

Crear archivo `backend/config/.env` (opcional):

```
DB_HOST=localhost
DB_PORT=5432
DB_NAME=galitroco_tfm
DB_USER=postgres
DB_PASSWORD=tu_password

RESEND_API_KEY=tu_api_key_de_resend
FRONTEND_URL=http://localhost:4200
```

4. Verificar instalación:

Acceder a: <http://galitroco.local/backend/api/test.php>

Debe mostrar:

```
{
  "status": "ok",
```

```
"php_version": "8.2.x",  
"database": "connected",  
"message": "API Backend funcionando correctamente"  
}
```

📖 **Más detalles:** Ver [backend/README_BACKEND.md](#)

4 CONFIGURAR EL FRONTEND

Opción A: Usar frontend desplegado en Render (RECOMENDADO)

El frontend ya está **desplegado y operativo** en Render.com. Solo necesitas acceder a:

```
https://galitroco-frontend.onrender.com
```

Configuración actual:

- ☒ Build automático desde GitHub (branch [main](#))
- ☒ Conectado al backend de producción (<https://render-test-php-1.onrender.com>)
- ☒ CORS configurado correctamente
- ☒ Variables de entorno de producción

Opción B: Ejecutar frontend localmente (OPCIONAL - para desarrollo)

1. Instalar dependencias:

```
cd frontend  
npm install
```

2. Configurar URL del backend:

El frontend local está configurado para usar el backend de **producción** (Render).

Editar [frontend/src/environments/environment.ts](#) si quieres usar backend local:

```
export const environment = {  
  production: false,  
  // Backend en Render (por defecto)  
  apiUrl: 'https://render-test-php-1.onrender.com/api.php'  
  
  // O backend local (si lo tienes corriendo):  
  // apiUrl: 'http://galitroco.local/backend/api/api.php'  
};
```

3. Iniciar servidor de desarrollo:

```
npm start
```

El frontend local estará disponible en: <http://localhost:4200>

 **Más detalles:** Ver [frontend/README_FRONTEND.md](#)

4.1 Desplegar frontend en Render (YA CONFIGURADO)

Si necesitas redespugar o configurar desde cero:

1. Crear nuevo Static Site en Render:

- Build Command: `cd frontend && npm install && npm run build:prod`
- Publish Directory: `frontend/dist/frontend/browser`
- Auto-Deploy: Yes (desde GitHub `main` branch)

2. Variables de entorno en Render:

- No se requieren variables de entorno
- La URL del backend está en `environment.prod.ts`

3. Verificar archivo `environment.prod.ts`:

```
export const environment = {  
  production: true,  
  apiUrl: 'https://render-test-php-1.onrender.com/api.php'  
};
```

Estado: ☒ Frontend desplegado y operativo en Render

TESTING Y VALIDACIÓN

Testing del Backend

Se ha realizado testing exhaustivo de 25 endpoints en producción (Render.com) con 92% de éxito (23/25 tests pasados).

Documento de evidencias: Ver [documentacion_tecnica/TESTING_Y_ENDPOINTS_TFM.md](#)

Credenciales de prueba:

Usuario normal A:

- Email: `test_6937@testmail.com`

- Password: `Pass123456`
- Rol: `usuario`

Usuario normal B:

- Email: `userB_6566@testing.com`
- Password: `Pass123456`
- Rol: `usuario`

Administrador:

- Email: `admin@galitroco.com`
- Password: `Admin123456`
- Rol: `administrador`

Testing del Frontend

Testing en Producción (RECOMENDADO):

El frontend está **desplegado en Render** e integrado con el backend de producción.

1. **Acceder a:** `https://galitroco-frontend.onrender.com`
2. **Probar flujo completo:**
 - Registro de usuario
 - Login
 - Crear habilidad
 - Proponer intercambio
 - Aceptar/Rechazar propuesta
 - Completar intercambio
 - Valorar usuario

Testing Local (OPCIONAL):

Si prefieres probar localmente:

1. Iniciar frontend: `npm start` (en carpeta `/frontend`)
2. Abrir navegador en: `http://localhost:4200`
3. El frontend local se conecta automáticamente al backend de Render

ESTADO DEL PROYECTO (PEC2)

Backend: ☒ **92% OPERATIVO**

- ☒ 25 endpoints implementados y testeados
- ☒ 23/25 tests pasados en producción
- ☒ 2 bugs críticos corregidos (transacciones ACID, router)
- ☒ 0 bugs pendientes críticos
- ☒ Autenticación JWT + Sesiones PHP

- ☒ Sistema de email funcional (Resend)
- ☒ Desplegado en Render.com con auto-deploy
- ☒ Documentación técnica completa

Frontend: ☒ **95% IMPLEMENTADO Y DESPLEGADO**

- ☒ Autenticación completa (login, registro, recuperación password)
- ☒ CRUD de habilidades con filtros y paginación
- ☒ Sistema de intercambios end-to-end
- ☒ Sistema de valoraciones con estrellas
- ☒ Panel de administración (reportes, usuarios)
- ☒ Guards de seguridad (auth, admin)
- ☒ Angular Material Design
- ☒ Integración con backend de producción (Render)
- ☒ **Desplegado en Render.com como Static Site**
- ⚠ Pendiente: Testing exhaustivo manual

Base de Datos: ☒ **100% OPERATIVA**

- ☒ Esquema completo con 10 tablas
- ☒ Relaciones e integridad referencial
- ☒ Índices y constraints
- ☒ Seeds con datos de prueba
- ☒ Migración a Supabase exitosa

📁 ESTRUCTURA DEL PROYECTO

```
galitroco/
├── backend/                                # API REST en PHP
│   ├── api/                               # Endpoints de la API
│   │   ├── index.php                     # Router principal
│   │   ├── auth.php                     # Autenticación
│   │   ├── habilidades.php              # CRUD habilidades
│   │   ├── intercambios.php            # Sistema de intercambios
│   │   ├── valoraciones.php            # Sistema de valoraciones
│   │   ├── reportes.php                 # Reportes y moderación
│   │   └── ...
│   ├── config/                           # Configuración
│   │   ├── database.php                 # Conexión PostgreSQL
│   │   ├── cors.php                     # CORS y cookies
│   │   └── jwt.php                       # JWT secret
│   ├── models/                           # Modelos de datos
│   ├── utils/                             # Utilidades
│   │   ├── Response.php                 # Respuestas JSON
│   │   ├── Auth.php                     # Middleware auth
│   │   └── Email.php                     # Envío de emails
│   └── Dockerfile                         # Contenedor para Render
```



```

├── frontend/                                # Aplicación Angular
│   ├── src/
│   │   ├── app/
│   │   │   ├── core/                      # Servicios y guards
│   │   │   ├── features/                 # Componentes por módulo
│   │   │   ├── shared/                  # Componentes compartidos
│   │   │   └── layout/                   # Header, footer, etc.
│   │   └── environments/                 # Configuración de entornos
│   ├── package.json
│   └── angular.json
├── database/                               # Scripts SQL
│   ├── schema.sql                       # Esquema completo
│   ├── seeds.sql                        # Datos de prueba
│   └── incremental_*.sql                 # Migraciones
├── documentacion_tecnica/                 # Documentación adicional
│   ├── TESTING_Y_ENDPOINTS_TFM.md
│   ├── ARQUITECTURA_DEPLOY.md
│   └── ...
└── README.md                             # Este archivo

```

SEGURIDAD

Medidas implementadas:

- ☒ Contraseñas hasheadas con bcrypt (cost 12)
- ☒ Prepared statements (prevención SQL Injection)
- ☒ Validación de entrada en todos los endpoints
- ☒ CORS configurado específicamente
- ☒ HTTPS en producción
- ☒ JWT para autenticación stateless
- ☒ Sesiones PHP con SameSite=None
- ☒ Guards de autenticación y autorización en frontend
- ☒ Middleware de protección en backend

LICENCIAS Y RECURSOS DE TERCEROS

Backend

- **PHP:** Licencia PHP License 3.01
- **PostgreSQL:** Licencia PostgreSQL License (similar a MIT)
- **Resend:** API comercial (cuenta de prueba gratuita)

Frontend

- **Angular:** Licencia MIT
- **Angular Material:** Licencia MIT
- **RxJS:** Licencia Apache 2.0
- **TypeScript:** Licencia Apache 2.0

Servicios Cloud

- **Render.com:** Servicio comercial (plan gratuito)
- **Supabase:** Open Source (PostgreSQL) + servicios cloud

 **Lista completa:** Ver [LICENCIAS_TERCEROS.md](#)

PROBLEMAS CONOCIDOS Y SOLUCIONES

1. CORS en localhost

Problema: Error "blocked by CORS policy" al conectar frontend local con backend Render.

Solución: Backend ya tiene configurado CORS para <http://localhost:4200> en [backend/config/cors.php](#).

2. Sesiones PHP entre dominios

Problema: Sesiones no persisten entre frontend y backend en dominios distintos.

Solución: Configurado `SameSite=None` y `Secure=true` en cookies. Usa `withCredentials: true` en Angular.

3. Cold start en Render

Problema: Primera petición al backend tarda 30-60 segundos (servidor dormido).

Solución: Esperar a que el servidor despierte. Render free tier tiene cold start inevitable.

SOPORTE Y CONTACTO

Autor: Antonio Campos

Email: *(incluir email institucional UOC)*

GitHub: <https://github.com/tonikampos/render-test-php>

Consultor: *(incluir nombre del consultor)*

GUÍA DE PRUEBAS PARA EVALUADORES

Esta sección proporciona instrucciones paso a paso para probar todas las funcionalidades de GaliTroco.

 Opción 1: Pruebas en Producción (RECOMENDADO - Sin instalación)

Acceso directo:

- **Frontend:** <https://galitroco-frontend.onrender.com>
- **Backend API:** <https://render-test-php-1.onrender.com/api.php>

Ventajas:

- ☒ No requiere instalación local
- ☒ Base de datos con datos de muestra
- ☒ Configuración completa y operativa
- ⌚ Tiempo de prueba: 15-20 minutos

Credenciales de Prueba

Utiliza estos usuarios para probar diferentes escenarios:

Usuario A (para intercambios)

```
Email:    test_6937@testmail.com
Password: Pass123456
Rol:      Usuario normal
```

Usuario B (para intercambios)

```
Email:    userB_6566@testing.com
Password: Pass123456
Rol:      Usuario normal
```

Administrador (panel admin)

```
Email:    admin@galitroco.com
Password: Admin123456
Rol:      Administrador
```

Escenario 1: Registro y Autenticación (3 minutos)

1.1 Probar registro de nuevo usuario

1. Ir a: <https://galitroco-frontend.onrender.com/registro>
2. Completar formulario:
 - Nombre de usuario: **evaluador_test**
 - Email: **evaluador@test.com**
 - Contraseña: **Test123456**

- Ubicación: **A Coruña**
- 3. Click en **"Registrarse"**
- 4. **Resultado esperado:** Redirección automática al login

1.2 Probar login

1. Ir a: **<https://galitroco-frontend.onrender.com/login>**
2. Ingresar credenciales:
 - Email: **test_6937@testmail.com**
 - Password: **[Pass123456](#)**
3. Click en **"Iniciar sesión"**
4. **Resultado esperado:** Redirección al dashboard con mensaje de bienvenida

1.3 Verificar persistencia de sesión

1. Cerrar navegador completamente
2. Abrir navegador y volver a: **<https://galitroco-frontend.onrender.com>**
3. **Resultado esperado:** Usuario sigue autenticado (sesión persiste)

1.4 Probar logout

1. Click en icono de usuario (esquina superior derecha)
2. Click en **"Cerrar sesión"**
3. **Resultado esperado:** Redirección a página de login

Escenario 2: Gestión de Habilidades (5 minutos)

2.1 Crear habilidad de tipo "Oferta"

1. Login como **test_6937@testmail.com**
2. Ir a: **"Mis Habilidades" → "Nueva Habilidad"**
3. Completar formulario:
 - Tipo: **Oferta**
 - Categoría: **Tecnología e Informática**
 - Título: **Clases de Python para principiantes**
 - Descripción: **Enseño programación en Python desde cero**
 - Duración estimada: **60 minutos**
4. Click en **"Publicar"**
5. **Resultado esperado:** Habilidad visible en "Mis Habilidades"

2.2 Crear habilidad de tipo "Demanda"

1. Click en **"Nueva Habilidad"** nuevamente
2. Completar formulario:
 - Tipo: **Demanda**
 - Categoría: **Clases y Formación**

- Título: **Busco clases de inglés conversacional**
 - Descripción: **Necesito mejorar mi inglés hablado**
 - Duración estimada: **90 minutos**
3. Click en **"Publicar"**
 4. **Resultado esperado:** Ambas habilidades visibles en el listado

2.3 Editar habilidad

1. En "Mis Habilidades", click en **icono de editar** (lápiz)
2. Modificar el título o descripción
3. Click en **"Guardar cambios"**
4. **Resultado esperado:** Cambios reflejados inmediatamente

2.4 Pausar/Activar habilidad

1. Click en botón **"Pausar"** de una habilidad
2. **Resultado esperado:** Estado cambia a "Pausada" (no visible en búsquedas públicas)
3. Click en **"Activar"** nuevamente
4. **Resultado esperado:** Estado vuelve a "Activa"

Escenario 3: Sistema de Intercambios Completo (7 minutos)

3.1 Proponer intercambio (Usuario A)

1. Login como **test_6937@testmail.com**
2. Ir a **"Explorar Habilidades"**
3. Buscar habilidades de otros usuarios
4. Click en una habilidad que te interese
5. Click en botón **"Proponer Intercambio"**
6. Seleccionar tu habilidad a ofrecer (de las que creaste)
7. Escribir mensaje: **Hola, me interesa tu habilidad. ¿Podemos intercambiar?**
8. Click en **"Enviar Propuesta"**
9. **Resultado esperado:**
 - Mensaje de confirmación
 - Propuesta visible en "Mis Intercambios" → "Enviados"
 - Notificación enviada al otro usuario

3.2 Aceptar propuesta (Usuario B)

1. **Cerrar sesión** de Usuario A
2. Login como **userB_6566@testing.com**
3. Ir a **"Mis Intercambios"** → **"Recibidos"**
4. Ver la propuesta recibida
5. Click en **"Aceptar"**
6. **Resultado esperado:**
 - Estado cambia a "Aceptado"

- Se crea conversación automática
- Notificación enviada al Usuario A

3.3 Enviar mensajes en conversación

1. Desde "Intercambios Aceptados", click en "**Ver conversación**"
2. Escribir mensaje: **Perfecto, ¿cuándo empezamos?**
3. Enviar mensaje
4. **Resultado esperado:** Mensaje visible en el chat

3.4 Completar intercambio

1. Click en botón "**Marcar como Completado**"
2. Confirmar acción
3. **Resultado esperado:**
 - Estado cambia a "Completado"
 - Aparece opción para valorar

3.5 Valorar usuario

1. Click en "**Valorar Usuario**"
2. Seleccionar puntuación (1-5 estrellas): **5 estrellas**
3. Escribir comentario: **Excelente experiencia, muy profesional**
4. Click en "**Enviar Valoración**"
5. **Resultado esperado:**
 - Valoración registrada
 - Puntuación promedio del usuario actualizada
6. **Cambiar a Usuario A** y repetir valoración (ambos deben valorarse)

Escenario 4: Búsqueda y Filtros (3 minutos)

4.1 Buscar por categoría

1. Ir a "**Explorar Habilidades**"
2. Seleccionar categoría: **Tecnología e Informática**
3. **Resultado esperado:** Solo habilidades de esa categoría

4.2 Buscar por palabra clave

1. En el buscador, escribir: **inglés**
2. **Resultado esperado:** Habilidades que contengan "inglés" en título o descripción

4.3 Filtrar por tipo

1. Seleccionar filtro: **Solo ofertas**
2. **Resultado esperado:** Solo habilidades de tipo "Oferta"

4.4 Filtrar por ubicación

1. Seleccionar: A **Coruña**
 2. **Resultado esperado:** Solo usuarios de A Coruña
-



Escenario 5: Panel de Administración (3 minutos)

5.1 Acceso al panel admin

1. **Cerrar sesión** de usuario normal
2. Login como **admin@galitroco.com** / **Admin123456**
3. Ir a **"Panel de Administración"**
4. **Resultado esperado:** Acceso permitido (solo admins)

5.2 Ver estadísticas

1. En dashboard admin, verificar widgets:
 - Total de usuarios registrados
 - Total de habilidades publicadas
 - Intercambios completados
 - Valoración promedio de la plataforma
2. **Resultado esperado:** Estadísticas en tiempo real

5.3 Gestionar reportes (si hay)

1. Ir a sección **"Reportes"**
2. Ver reportes pendientes
3. Click en un reporte para ver detalles
4. Marcar como **"Revisado"** o **"Resuelto"**
5. **Resultado esperado:** Estado del reporte actualizado

5.4 Gestionar usuarios

1. Ir a sección **"Usuarios"**
 2. Ver listado completo de usuarios
 3. Buscar usuario específico
 4. **Opciones disponibles:**
 - Ver perfil completo
 - Desactivar/Activar cuenta
 - Ver historial de intercambios
 5. **Resultado esperado:** Acciones de moderación funcionan
-



Escenario 6: Recuperación de Contraseña (2 minutos)

6.1 Solicitar recuperación

1. **Cerrar sesión** de cualquier usuario
2. En página de login, click en "**¿Olvidaste tu contraseña?**"
3. Ingresar email: `test_6937@testmail.com`
4. Click en "**Enviar enlace de recuperación**"
5. **Resultado esperado:**
 - Mensaje de confirmación
 - Email enviado con token de recuperación (via Resend API)

6.2 Verificar token (opcional)

1. Revisar consola de Resend (si tienes acceso)
2. **Resultado esperado:** Email entregado exitosamente

☒ Checklist de Verificación Final

Marca cada funcionalidad después de probarla:

Autenticación:

- ☐ Registro de nuevo usuario funciona
- ☐ Login con credenciales correctas
- ☐ Sesión persiste después de cerrar navegador
- ☐ Logout funciona correctamente
- ☐ Recuperación de contraseña envía email

Habilidades:

- ☐ Crear habilidad tipo "Oferta"
- ☐ Crear habilidad tipo "Demanda"
- ☐ Editar habilidad existente
- ☐ Pausar/Activar habilidad
- ☐ Eliminar habilidad
- ☐ Ver habilidades propias
- ☐ Explorar habilidades de otros usuarios

Intercambios:

- ☐ Proponer intercambio funciona
- ☐ Receptor recibe notificación
- ☐ Aceptar propuesta cambia estado
- ☐ Rechazar propuesta funciona
- ☐ Marcar como completado funciona
- ☐ Ver historial de intercambios

Valoraciones:

- ☐ Sistema de estrellas (1-5) funciona
- ☐ Comentario se guarda correctamente

- ☐ Valoración promedio se calcula
- ☐ Valoraciones visibles en perfil

Mensajería:

- ☐ Conversación se crea automáticamente
- ☐ Enviar mensaje funciona
- ☐ Mensajes se muestran en orden cronológico
- ☐ Notificación de nuevo mensaje

Búsqueda:

- ☐ Filtrar por categoría funciona
- ☐ Buscar por palabra clave
- ☐ Filtrar por tipo (oferta/demanda)
- ☐ Filtrar por ubicación

Panel Admin:

- ☐ Solo admins acceden al panel
- ☐ Estadísticas se muestran correctamente
- ☐ Gestión de reportes funciona
- ☐ Gestión de usuarios funciona

Troubleshooting Común

Problema: "No se puede conectar al servidor"

Solución:

- Verificar que la URL sea correcta: <https://galitroco-frontend.onrender.com>
- Render puede tardar 30-60 segundos en "despertar" el servicio si estuvo inactivo

Problema: "Error al crear habilidad"

Solución:

- Verificar que todos los campos obligatorios estén completos
- El título debe tener al menos 10 caracteres
- La descripción debe tener al menos 20 caracteres

Problema: "No puedo proponer intercambio"

Solución:

- Asegúrate de tener al menos 1 habilidad propia publicada
- No puedes proponer intercambio con tus propias habilidades
- La habilidad del otro usuario debe estar "Activa"

Problema: "No recibo email de recuperación"

Solución:

- Revisar carpeta de spam
- El email puede tardar 1-2 minutos en llegar
- Verificar que el email esté registrado en la base de datos



Endpoints API para Testing Avanzado

Si deseas probar la API directamente (con Postman, curl, etc.):

Base URL: <https://render-test-php-1.onrender.com/api.php>

Autenticación

```
POST /api.php?resource=auth&action=login
Body: {"email": "test_6937@testmail.com", "password": "Pass123456"}
```

Habilidades

```
GET /api.php?resource=habilidades
GET /api.php?resource=habilidades&id=1
POST /api.php?resource=habilidades
```

Intercambios

```
GET /api.php?resource=intercambios
POST /api.php?resource=intercambios
PUT /api.php?resource=intercambios&id=1&action=aceptar
```



Documentación completa de API: Ver archivo [TESTING_Y_ENDPOINTS_TFM.md](#)



Tiempo Total de Pruebas

- **Prueba rápida (funcionalidades básicas):** 10-15 minutos
- **Prueba completa (todos los escenarios):** 25-30 minutos
- **Prueba exhaustiva (+ testing API):** 45-60 minutos



REFERENCIAS Y DOCUMENTACIÓN ADICIONAL

Documentación técnica incluida:

- [backend/README_BACKEND.md](#) - Guía detallada del backend
- [frontend/README_FRONTEND.md](#) - Guía detallada del frontend
- [database/README_DATABASE.md](#) - Guía de la base de datos
- [TESTING_Y_ENDPOINTS_TFM.md](#) - Testing exhaustivo de la API
- [ARQUITECTURA_DEPLOY.md](#) - Arquitectura de despliegue
- [GUIA_RECUPERACION_PASSWORD.md](#) - Sistema de recuperación de contraseña

Documentación externa:

- [Angular Docs](#)
- [PHP Manual](#)
- [PostgreSQL Docs](#)
- [Angular Material](#)
- [Render Docs](#)
- [Supabase Docs](#)



HISTORIAL DE VERSIONES

v1.0 - PEC2 (Noviembre 2025)

- ☒ Backend completo con 25 endpoints
- ☒ Frontend Angular con 95% funcionalidades
- ☒ Sistema de intercambios end-to-end
- ☒ Sistema de valoraciones
- ☒ Panel de administración
- ☒ Testing en producción (92% éxito)
- ☒ Documentación técnica completa

v0.5 - PEC1 (Septiembre 2025)

- ☒ Planificación inicial
- ☒ Análisis de requisitos
- ☒ Diseño de base de datos
- ☒ Wireframes de interfaces



LICENCIA DEL PROYECTO

Este proyecto es un Trabajo Final de Máster para la UOC con fines académicos.

Código fuente: Propiedad de Antonio Campos

Uso: Exclusivamente educativo

Redistribución: No permitida sin autorización

Última actualización: 23 de octubre de 2025

Versión del documento: 1.0 (PEC2)

Estado: ☒ Listo para entrega PEC2