**Instituto Puig Castellar**
**Ciclo formativo:** SMX
**Grado:** CFGM
Proyecto intermodular

# TECHLAN

## Plataforma web en el ámbito TIC

**Autores:** Ivan Samper Fernandez, Iker Ortuño Gutierrez.
**Tutor:** David Delgado Santamaria
**Curso:** 2025-2026
**Fecha de entrega:** 17/05/2026

## Resumen del proyecto

El proyecto Techlan consiste en el desarrollo de una plataforma web orientada a la gestión de incidencias y soporte técnico en el ámbito de las tecnologías de la información. Su objetivo principal es centralizar y organizar las solicitudes de los usuarios, facilitando su seguimiento y mejorando la comunicación entre clientes y administradores.

La aplicación se ha desarrollado utilizando una arquitectura cliente-servidor, con un frontend dinámico implementado en React y un backend basado en PHP que actúa como API para gestionar la lógica del sistema y la conexión con la base de datos. A lo largo del desarrollo se han implementado funcionalidades como el registro e inicio de sesión de usuarios, la creación y gestión de tickets, un panel de administración, un sistema de notificaciones y el envío de correos electrónicos.

Finalmente, el proyecto ha sido desplegado en un entorno real mediante un dominio público, lo que ha permitido validar su funcionamiento fuera del entorno de desarrollo. El resultado es una aplicación funcional, accesible y estructurada, que cumple con los objetivos planteados y ofrece una solución práctica a la gestión de incidencias.

## Palabras clave

- Gestión de incidencias.
- Aplicación web.
- Soporte técnico.
- React.
- PHP.
- Sistema de tickets.
- Desarrollo web.

## Abstract

The Techlan project consists of the development of a web platform focused on incident management and technical support within the field of information technologies. Its main objective is to centralize and organize user requests, improving their tracking and enhancing communication between users and administrators.

The application has been developed using a client-server architecture, with a dynamic frontend built in React and a backend implemented in PHP, which acts as an API responsible for handling the system logic and database communication. Throughout the development process, several key features have been implemented, including user registration and login, ticket creation and management, an administrative panel, a notification system, and email sending functionality.

Finally, the project has been deployed in a real environment through a public domain, allowing its functionality to be tested outside the development environment. The result is a fully functional, accessible, and well-structured application that meets the initial objectives and provides a practical solution for incident management.

## Keywords

- Incident management
- Web application
- Technical support
- React
- PHP
- Ticket system
- Web development

# Índice

# 1. Presentación del proyecto

## 1.1 Introducción

El proyecto **Techlan** consiste en el diseño y desarrollo de una plataforma web orientada a la gestión de incidencias y servicios de soporte técnico en el ámbito de las tecnologías de la información y la comunicación (TIC). Su finalidad principal es proporcionar una herramienta digital que permita centralizar, organizar y optimizar el proceso de gestión de problemas técnicos, facilitando la comunicación entre los diferentes actores implicados: usuarios, técnicos y administradores.

La aplicación ha sido concebida como un sistema accesible a través de un navegador web, lo que permite su utilización desde distintos dispositivos sin necesidad de instalar software adicional. Esto supone una ventaja importante frente a sistemas tradicionales, ya que mejora la accesibilidad y la disponibilidad del servicio.

Desde el punto de vista técnico, el proyecto ha evolucionado a lo largo de su desarrollo. Inicialmente, se planteó como una aplicación web sencilla basada en tecnologías básicas, pero posteriormente se transformó en una solución más completa y moderna. En su versión final, el sistema está compuesto por:

- Un frontend dinámico desarrollado con React, que permite una interacción fluida y sin recargas de página.
- Un backend implementado en PHP, que actúa como API encargada de gestionar la lógica del sistema y la comunicación con la base de datos.
- Un sistema de almacenamiento de datos que permite gestionar usuarios, incidencias y notificaciones.

Además, la aplicación ha sido desplegada en un entorno real mediante un dominio público (**techlan.es**), lo que ha permitido validar su funcionamiento fuera del entorno de desarrollo y comprobar su comportamiento en condiciones reales.

En conjunto, el proyecto no solo representa una solución funcional a un problema concreto, sino también una experiencia completa de desarrollo de software, abarcando desde el análisis inicial hasta la implementación y el despliegue final.

## 1.2 Contexto

En la actualidad, el uso de sistemas informáticos es fundamental en prácticamente cualquier entorno, tanto a nivel empresarial como personal. Ordenadores, redes y sistemas digitales forman parte del día a día, lo que implica que la aparición de incidencias técnicas es algo habitual.

Sin embargo, la gestión de estas incidencias no siempre se realiza de manera eficiente. En muchos casos, las solicitudes de soporte técnico se gestionan mediante herramientas poco adecuadas, como correos electrónicos, llamadas telefónicas o incluso mensajes informales. Este tipo de métodos presentan diversas limitaciones:

- Falta de organización en las solicitudes.
- Dificultad para hacer seguimiento de las incidencias.
- Pérdida de información relevante.
- Problemas de comunicación entre usuarios y técnicos.
- Ausencia de un historial estructurado.

Estas limitaciones afectan directamente a la calidad del servicio, aumentando los tiempos de respuesta y reduciendo la eficiencia en la resolución de problemas.

En este contexto, surge la necesidad de desarrollar herramientas específicas que permitan gestionar de forma estructurada las incidencias técnicas. Las plataformas web orientadas a este propósito ofrecen una solución eficaz, ya que permiten centralizar la información, automatizar procesos y mejorar la comunicación entre los diferentes usuarios del sistema.

El proyecto **Techlan** se sitúa dentro de este contexto, proponiendo una solución digital que responde a estas necesidades mediante el uso de tecnologías web modernas. Además, el hecho de que la aplicación esté disponible en internet refuerza su utilidad, permitiendo su acceso desde cualquier lugar.

## 1.3 Justificación

La realización del proyecto **Techlan** está justificada tanto desde un punto de vista práctico como formativo.

Desde el punto de vista práctico, el proyecto responde a un problema real: la gestión ineficiente de incidencias técnicas. Como se ha mencionado anteriormente, los métodos tradicionales no permiten un control adecuado de las solicitudes, lo que puede generar retrasos, errores y falta de coordinación.

La plataforma desarrollada ofrece una solución a este problema mediante:

- La centralización de todas las incidencias en un único sistema.
- La posibilidad de consultar el estado de las solicitudes en tiempo real.
- La mejora de la comunicación entre usuarios y técnicos.
- La existencia de un historial que permite analizar incidencias pasadas.

Todo esto contribuye a mejorar la eficiencia del servicio y a ofrecer una experiencia más organizada y profesional.

Por otro lado, desde el punto de vista formativo, el proyecto tiene un gran valor educativo. Su desarrollo ha permitido aplicar y consolidar conocimientos adquiridos durante el ciclo formativo, especialmente en áreas como:

- Desarrollo de aplicaciones web.
- Programación frontend y backend.
- Gestión de bases de datos.
- Arquitectura de software.
- Despliegue en entornos reales.

Además, el hecho de haber trabajado con tecnologías actuales como React y PHP, así como haber realizado un despliegue en un dominio real, aporta una experiencia muy cercana al entorno profesional.

En este sentido, el proyecto no solo cumple una función práctica, sino que también sirve como preparación para futuros retos en el ámbito laboral.

# 1.4 Objetivo general y objetivos específicos

## Objetivo general

Desarrollar una plataforma web funcional que permita gestionar incidencias de soporte técnico de manera eficiente, organizada y accesible, mejorando la comunicación entre usuarios, técnicos y administradores.

## Objetivos específicos

Para alcanzar el objetivo general, se han definido una serie de objetivos específicos que guían el desarrollo del proyecto:

- Implementar un sistema de registro e inicio de sesión: Permitir que los usuarios puedan crear una cuenta y acceder a la plataforma de forma segura, garantizando el control de acceso al sistema.
- Desarrollar un sistema de gestión de incidencias (tickets): Permitir a los usuarios crear solicitudes de soporte técnico, describiendo sus problemas y facilitando su resolución.
- Permitir el seguimiento del estado de las incidencias: Ofrecer a los usuarios la posibilidad de consultar en todo momento el estado de sus solicitudes, mejorando la transparencia del proceso.
- Implementar un sistema de gestión para administradores: Proporcionar herramientas que permitan a los administradores visualizar, organizar y actualizar las incidencias.
- Desarrollar un sistema de notificaciones: Informar a los usuarios sobre cambios relevantes en sus incidencias, como actualizaciones de estado.
- Garantizar la seguridad y control de acceso: Asegurar que cada usuario solo pueda acceder a las funcionalidades que le corresponden según su rol.
- Desplegar la aplicación en un entorno real: Publicar la plataforma en un dominio accesible desde internet (**techlan**.*es*), validando su funcionamiento fuera del entorno de desarrollo.

# 2. Estrategia y planificación

## 2.1 Estrategia de desarrollo y viabilidad

El desarrollo del proyecto **Techlan** se ha planteado siguiendo una estrategia progresiva, basada en la evolución del sistema desde una idea inicial sencilla hasta una aplicación web completa y funcional.

En una primera fase, el objetivo principal fue comprender el problema a resolver y definir una solución básica. Para ello, se planteó inicialmente una aplicación web sencilla utilizando tecnologías accesibles, con el fin de centrarse en la lógica del sistema y en la gestión de incidencias.

Sin embargo, a medida que el proyecto avanzó, se detectó la necesidad de mejorar la estructura y el funcionamiento de la aplicación, especialmente en aspectos como la experiencia de usuario, la organización del código y la escalabilidad del sistema. Esto llevó a una evolución en la estrategia de desarrollo, adoptando un enfoque más cercano al desarrollo profesional.

Como resultado, la aplicación final se basa en una arquitectura moderna cliente-servidor, en la que se diferencian claramente tres componentes principales:

- Frontend (React): encargado de la interfaz de usuario y de la interacción con el sistema.

- Backend (PHP): responsable de la lógica del sistema, la gestión de datos y la comunicación con la base de datos mediante una API.
- Base de datos: donde se almacenan los datos persistentes del sistema, como usuarios, incidencias y notificaciones.

Esta separación permite una mejor organización del proyecto, facilita su mantenimiento y hace posible futuras ampliaciones.

Además, durante la fase final del proyecto se tomó la decisión de realizar el despliegue en un entorno real, publicando la aplicación en el dominio **techlan**.*es*. Este paso supuso un cambio importante en la estrategia, ya que fue necesario adaptar el sistema a las condiciones del hosting, especialmente en lo referente al uso de PHP como tecnología backend.

## Viabilidad del proyecto

El proyecto ha sido viable desde diferentes puntos de vista:

- Viabilidad técnica: Las tecnologías utilizadas (React, PHP, bases de datos relacionales) son ampliamente conocidas y cuentan con una gran cantidad de documentación y soporte. Esto ha facilitado el desarrollo y la resolución de problemas.
- Viabilidad temporal: El desarrollo se ha realizado a lo largo del curso académico, dividiendo el trabajo en fases y estableciendo objetivos progresivos. Esto ha permitido avanzar de manera ordenada y cumplir con los plazos establecidos.
- Viabilidad económica: El proyecto tiene un coste reducido, ya que se han utilizado herramientas gratuitas para el desarrollo (editores de código, librerías, etc.). Los únicos costes asociados han sido el dominio y el hosting necesarios para el despliegue de la aplicación.

En conjunto, la estrategia adoptada ha permitido desarrollar una solución funcional, adaptable y cercana a un entorno real.

## 2.2 Metodología de trabajo

Para el desarrollo del proyecto se ha utilizado una metodología de trabajo de tipo iterativo, combinando elementos de planificación inicial con un desarrollo progresivo basado en la mejora continua.

En lugar de desarrollar todo el sistema de una sola vez, se ha optado por dividir el proyecto en pequeñas partes o módulos, implementando cada funcionalidad de forma independiente antes de integrarla en el sistema completo.

Este enfoque ha permitido:

- Detectar errores de forma temprana.
- Realizar pruebas continuas.
- Introducir mejoras progresivamente.
- Adaptarse a cambios durante el desarrollo.

### Fases de trabajo dentro de la metodología

El trabajo se ha organizado en ciclos que siguen una estructura similar:

1. **Análisis de la funcionalidad a implementar:** Se define qué se quiere hacer y qué requisitos debe cumplir.
2. **Diseño de la solución:** Se plantea cómo se va a implementar (estructura, lógica, interfaz).
3. **Desarrollo**: Se implementa la funcionalidad en el código.

4. **Pruebas:** Se comprueba que funciona correctamente y se corrigen errores.
5. **Mejora:** Se optimiza el código y se realizan ajustes si es necesario.

Este ciclo se ha repetido para cada una de las funcionalidades principales del sistema, como el login, la gestión de tickets o el panel de administración.

## Uso de herramientas de desarrollo

Durante el proyecto se han utilizado herramientas que han facilitado el trabajo:

- GitHub: Para el control de versiones, permitiendo guardar cambios, recuperar versiones anteriores y organizar el desarrollo.
- Visual Studio Code: Como entorno de desarrollo, facilitando la edición de código y la integración con diferentes tecnologías.
- Navegadores web: Para la prueba continua de la aplicación.

El uso de estas herramientas ha permitido mantener un desarrollo organizado y eficiente.

# 2.3 Planificación

La planificación del proyecto ha sido un elemento clave para su desarrollo, ya que ha permitido organizar el trabajo y distribuir las tareas a lo largo del tiempo.

Aunque no se ha seguido un cronograma rígido, sí se ha trabajado con una planificación por fases, estableciendo objetivos claros en cada etapa del proyecto.

## Fases del proyecto

**Fase 1: Análisis**

En esta fase inicial se definió el problema a resolver y se identificaron las necesidades del sistema.

Se establecieron:

- Los objetivos del proyecto.
- Los tipos de usuarios.
- Las funcionalidades principales.

**Fase 2: Diseño**

Se diseñó la estructura del sistema, definiendo:

- La arquitectura (cliente-servidor).
- El modelo de datos.
- La organización del frontend y backend.
- La interfaz de usuario.

Esta fase permitió tener una base clara antes de comenzar a programar.

**Fase 3: Desarrollo del frontend**

Se implementó la interfaz de usuario utilizando React, desarrollando:

- Componentes reutilizables.
- Formularios (login, registro, incidencias).
- Páginas principales del sistema.

El objetivo fue crear una interfaz dinámica y fácil de usar.

**Fase 4: Desarrollo del backend**

Se desarrolló la lógica del sistema mediante PHP, creando una API que permite:

- Gestionar usuarios.
- Procesar solicitudes.
- Acceder a la base de datos.
- Devolver respuestas al frontend.

**Fase 5: Integración del sistema**

Se conectó el frontend con el backend mediante peticiones HTTP, permitiendo el funcionamiento completo de la aplicación.

En esta fase se verificó que:

- Los datos se envían correctamente.
- El servidor responde adecuadamente.
- La información se muestra en la interfaz.

**Fase 6: Pruebas**

Se realizaron pruebas para comprobar el correcto funcionamiento del sistema:

- Pruebas de login.
- Creación de tickets.
- Gestión de incidencias.
- Navegación por la aplicación.

**Fase 7: Despliegue**

Finalmente, la aplicación fue publicada en un entorno real:

- Configuración del hosting.
- Adaptación del backend a PHP.
- Subida de archivos.
- Configuración del dominio (**techlan***.es*).

Esta fase permitió validar el proyecto en condiciones reales.

# 3. Análisis

## 3.1 Casos de uso

Los casos de uso permiten describir cómo interactúan los distintos usuarios con el sistema en situaciones reales. En el proyecto **Techlan**, se han identificado varios tipos de usuarios y acciones que pueden realizar dentro de la plataforma.

Los principales roles del sistema son:

- Usuario (cliente): persona que crea incidencias.
- Administrador/Técnico: persona que gestiona las incidencias.

### Caso de uso 1: Registro de usuario

Un usuario que no dispone de cuenta accede a la plataforma y completa el formulario de registro.

El sistema recoge los datos introducidos (como correo electrónico y contraseña), los envía al backend y, tras validarlos, crea un nuevo registro en la base de datos.

Este proceso permite que el usuario pueda acceder posteriormente a la plataforma de forma autenticada.

### Caso de uso 2: Inicio de sesión

Un usuario registrado introduce sus credenciales en el formulario de inicio de sesión.

El sistema envía estos datos al servidor, donde se comprueba que coincidan con un usuario existente. Si la validación es correcta, el usuario accede a su panel personal.

Este caso de uso es fundamental, ya que permite controlar el acceso al sistema y proteger la información.

### Caso de uso 3: Creación de incidencia (ticket)

Una vez autenticado, el usuario puede crear una incidencia describiendo el problema técnico que necesita resolver.

El sistema recoge la información introducida en el formulario y la envía al backend, donde se almacena en la base de datos con un estado inicial (por ejemplo, “pendiente”).

Este caso de uso representa la funcionalidad principal del sistema.

### Caso de uso 4: Consulta del estado de incidencias

El usuario puede acceder a su historial de incidencias y consultar el estado de cada una de ellas.

El sistema recupera los datos desde la base de datos y los muestra en la interfaz, permitiendo al usuario ver si su solicitud está pendiente, en proceso o resuelta.

## 3.2 Requisitos funcionales

Los requisitos funcionales describen las funcionalidades que el sistema debe ofrecer para cumplir su objetivo.

En el caso de **Techlan**, los requisitos funcionales se han definido en base a las necesidades detectadas durante el análisis del problema.

### Gestión de usuarios

- El sistema debe permitir el registro de nuevos usuarios.
- Debe permitir el inicio de sesión mediante credenciales.
- Debe diferenciar entre tipos de usuario (usuario y administrador).

### Gestión de incidencias

- El sistema debe permitir la creación de incidencias.
- Debe almacenar la información en la base de datos.
- Debe asociar cada incidencia a un usuario.

### Seguimiento de incidencias

- El sistema debe permitir consultar el estado de las incidencias.
- Debe mostrar un historial de solicitudes.
- Debe actualizar la información en función de los cambios realizados.

## Gestión administrativa

- El sistema debe permitir a los administradores visualizar todas las incidencias.
- Debe permitir modificar el estado de las incidencias.
- Debe ofrecer herramientas de control del sistema.

## Sistema de notificaciones

- El sistema debe generar notificaciones ante eventos relevantes.
- Debe permitir al usuario visualizar dichas notificaciones.
- Debe permitir marcar las notificaciones como leídas.

## Comunicación mediante formularios

- El sistema debe permitir el envío de mensajes mediante formularios.
- Debe procesar estos mensajes en el backend.
- Debe permitir su gestión o almacenamiento.

## 3.3 Requisitos no funcionales

Los requisitos no funcionales definen características del sistema relacionadas con su calidad, rendimiento y funcionamiento general.

### Usabilidad

La aplicación debe ser fácil de usar, con una interfaz clara e intuitiva que permita a cualquier usuario interactuar con el sistema sin necesidad de conocimientos técnicos avanzados.

Esto se ha conseguido mediante el uso de React, que permite una interfaz dinámica y organizada.

### Rendimiento

El sistema debe responder de forma rápida a las acciones del usuario, evitando tiempos de espera excesivos.

El uso de una arquitectura SPA (Single Page Application) contribuye a mejorar el rendimiento, ya que evita recargas completas de página.

### Seguridad

El sistema debe proteger la información de los usuarios, especialmente sus credenciales.

Para ello:

- Se validan los datos en el backend.
- Se controla el acceso mediante autenticación.
- Se restringe el acceso a ciertas rutas según el rol del usuario.

## Compatibilidad

La aplicación debe funcionar correctamente en diferentes navegadores y dispositivos.

Al ser una aplicación web, se puede acceder desde cualquier dispositivo con conexión a internet.

## Escalabilidad

El sistema debe poder ampliarse en el futuro sin necesidad de ser rediseñado completamente.

La separación entre frontend y backend facilita la incorporación de nuevas funcionalidades.

## Disponibilidad

Al estar desplegada en un dominio público (**techlan**.*es*), la aplicación debe estar disponible en todo momento, permitiendo el acceso continuo por parte de los usuarios.

# 3.4 Análisis de alternativas tecnológicas

Durante el desarrollo del proyecto se han considerado diferentes opciones tecnológicas antes de elegir la solución final.

### Frontend

Inicialmente se planteó el uso de HTML tradicional, pero esta opción presentaba limitaciones en cuanto a dinamismo y experiencia de usuario.

Finalmente, se optó por utilizar **React**, ya que permite:

- Crear interfaces dinámicas.
- Reutilizar componentes.
- Mejorar la experiencia del usuario.
- Organizar mejor el código.

### Backend

En fases iniciales se consideró el uso de Python, pero en el momento del despliegue se optó por utilizar **PHP**.

Esta decisión se tomó principalmente por:

- Compatibilidad con el hosting.
- Facilidad de despliegue.
- Integración directa con el servidor.

## Base de datos

Se consideraron opciones como SQLite en fases iniciales, pero finalmente se optó por una base de datos relacional gestionada desde PHP, más adecuada para un entorno real.

## Arquitectura

Se optó por una arquitectura cliente-servidor con separación frontend/backend.

Esta decisión permite:

- Mejor organización del código.
- Mayor escalabilidad.
- Posibilidad de reutilizar la API.

# 4. Diseño

## 4.1 Arquitectura del sistema

La arquitectura del sistema define la forma en la que se organizan y relacionan los distintos componentes de la aplicación. En el caso del proyecto **Techlan**, se ha optado por una arquitectura de tipo **cliente-servidor**, ampliamente utilizada en aplicaciones web modernas.

Este modelo permite separar claramente las responsabilidades del sistema en diferentes capas, facilitando el desarrollo, mantenimiento y escalabilidad de la aplicación.

### Estructura general del sistema

El sistema se divide en tres componentes principales:

- **Frontend (cliente).**
- **Backend (servidor).**
- **Base de datos.**

Cada uno de estos componentes cumple una función específica dentro del sistema.

### Frontend (React)

El frontend es la parte de la aplicación con la que interactúa directamente el usuario. En este proyecto, se ha desarrollado utilizando **React**, una librería de JavaScript que permite crear interfaces dinámicas y reutilizables.

Su función principal es:

- Mostrar la información al usuario.
- Recoger datos mediante formularios.
- Enviar peticiones al backend.
- Actualizar la interfaz sin recargar la página.

Una de las principales ventajas de utilizar React es que permite desarrollar una **Single Page Application (SPA)**, lo que mejora la experiencia de usuario al evitar recargas completas de página.

Además, el frontend está organizado en componentes, lo que facilita la reutilización de código y la organización del proyecto.

## Backend (API en PHP)

El backend es el encargado de gestionar la lógica del sistema. En este proyecto, se ha implementado mediante **PHP**, estructurado como una API que responde a las peticiones realizadas desde el frontend.

Sus funciones principales son:

- Recibir peticiones HTTP desde el frontend.
- Procesar los datos enviados por el usuario.
- Validar la información.
- Ejecutar operaciones sobre la base de datos.
- Devolver respuestas en formato JSON.

El backend actúa como intermediario entre el frontend y la base de datos, asegurando que la información se gestione de forma correcta y segura.

## Base de datos

La base de datos es el componente encargado de almacenar la información de forma persistente.

En el proyecto **Techlan**, la base de datos almacena:

- Usuarios.
- Incidencias (tickets).
- Notificaciones.

El acceso a la base de datos se realiza exclusivamente a través del backend, lo que permite controlar mejor la seguridad y la integridad de los datos.

## Comunicación entre componentes

La comunicación entre el frontend y el backend se realiza mediante **peticiones HTTP**, en las que se intercambian datos en formato JSON.

El flujo de funcionamiento es el siguiente:

1. El usuario realiza una acción en la interfaz (por ejemplo, crear un ticket).
2. El frontend envía una petición HTTP al backend.
3. El backend procesa la petición y accede a la base de datos si es necesario.
4. El backend devuelve una respuesta al frontend.
5. El frontend actualiza la interfaz con la información recibida.

Este modelo permite una comunicación clara y estructurada entre las distintas partes del sistema.

### Ventajas de la arquitectura utilizada

La arquitectura cliente-servidor adoptada presenta varias ventajas:

- Separación de responsabilidades.
- Mayor facilidad de mantenimiento.
- Posibilidad de escalar el sistema.
- Mejora de la seguridad.
- Reutilización de la API.

En conjunto, esta arquitectura permite construir una aplicación robusta y adaptable.

## 4.2 Modelo de datos

El modelo de datos define cómo se organiza la información dentro del sistema y cómo se relacionan los distintos elementos entre sí.

En el proyecto **Techlan**, se han identificado varias entidades principales que representan los elementos clave del sistema.

### Entidad: Usuarios

Esta entidad almacena la información de los usuarios registrados en la plataforma.

Campos principales:

- **id**: identificador único del usuario.
- **email**: correo electrónico.

- **password**: contraseña (almacenada de forma segura).
- **rol**: tipo de usuario (usuario o administrador).

Cada usuario puede crear incidencias y recibir notificaciones.

## Entidad: Tickets (incidencias)

Esta entidad representa las solicitudes de soporte técnico creadas por los usuarios.

Campos principa

- **id**: identificador del ticket.
- **user_id**: referencia al usuario que crea la incidencia.
- **descripción**: detalle del problema.
- **estado**: estado de la incidencia (pendiente, en proceso, resuelto).
- **fecha**: fecha de creación.

Cada ticket está asociado a un usuario y puede cambiar de estado a lo largo del tiempo.

## Entidad: Notificaciones

Esta entidad se utiliza para informar a los usuarios sobre eventos relevantes.

Campos principales:

- **id**: identificador de la notificación.
- **usuario_id**: usuario al que pertenece.
- **mensaje**: contenido de la notificación.
- **leída**: indica si ha sido vista.
- **fecha**: fecha de creación.

Este sistema permite mejorar la comunicación dentro de la plataforma.

### Relaciones entre entidades

- Un usuario puede tener múltiples tickets.
- Un usuario puede tener múltiples notificaciones.
- Cada ticket pertenece a un único usuario.

Estas relaciones permiten estructurar la información de forma lógica y organizada.

## 4.3 Diseño de interfaz

El diseño de la interfaz de usuario es un aspecto fundamental del proyecto, ya que determina cómo interactúan los usuarios con la aplicación.

En **Techlan**, la interfaz se ha desarrollado utilizando React, lo que permite crear una experiencia dinámica, rápida y organizada.

### Principios de diseño

Durante el desarrollo de la interfaz se han tenido en cuenta varios principios:

- **Simplicidad**: evitar elementos innecesarios.
- **Claridad**: mostrar la información de forma comprensible.
- **Facilidad de uso**: permitir que cualquier usuario pueda utilizar la aplicación.
- **Consistencia**: mantener un diseño uniforme en toda la aplicación.

### Estructura de la interfaz

La aplicación está organizada en diferentes páginas y componentes:

**Páginas principales**

- Página de inicio.
- Página de registro.
- Página de inicio de sesión.
- Panel de usuario.
- Panel de administración.
- Historial de incidencias.
- Página de contacto.

**Componentes reutilizables**

El uso de React ha permitido dividir la interfaz en componentes reutilizables, como:

- Formularios.
- Barras de navegación.
- Listas de tickets.
- Notificaciones (campana).
- Indicadores de estado.

Esto mejora la organización del código y facilita futuras modificaciones.

## Interacción con el usuario

El usuario interactúa con la aplicación principalmente mediante:

- Formularios (login, registro, incidencias).
- Botones (crear, actualizar, eliminar).
- Paneles de visualización.

Cada acción del usuario genera una petición al backend, lo que permite actualizar la información en tiempo real.

## Adaptabilidad

Al tratarse de una aplicación web, la interfaz es accesible desde diferentes dispositivos (ordenadores, tablets, móviles), siempre que dispongan de un navegador web.

# 5. Desarrollo

## 5.1 Estructura del proyecto

La organización del proyecto **Techlan** sigue una estructura modular que permite separar claramente las distintas partes del sistema, facilitando su desarrollo, mantenimiento y escalabilidad.

El proyecto se divide principalmente en dos grandes bloques:

- **Frontend (React).**
- **Backend (API en PHP).**

Además, se incluyen otros elementos auxiliares como archivos de configuración, recursos estáticos y librerías externas.

### Estructura general

De forma simplificada, la estructura del proyecto es la siguiente:

```
techlan/
│
|── src/              > Código del frontend (React)
|── api/              > Backend (PHP)
|── public/           > Archivos públicos
|── php/              > Scripts adicionales en PHP
|── index.html         > Punto de entrada
|── package.json       > Configuración del proyecto React
```

## Frontend (carpeta /src)

Esta carpeta contiene todo el código relacionado con la interfaz de usuario. Está organizada en diferentes subcarpetas que permiten estructurar el proyecto de forma clara:

- **components/ :** Contiene componentes reutilizables de la interfaz, como botones, formularios, listas o elementos visuales.
- **pages/ :** Incluye las distintas páginas de la aplicación, como login, registro, panel de usuario o administración.
- **context/ :** Gestiona el estado global de la aplicación, especialmente la autenticación del usuario (por ejemplo, mediante **AuthContext**).

Esta organización permite separar la lógica de presentación y facilita la reutilización del código.

## Backend (carpeta /api)

El backend está desarrollado en PHP y se organiza como una API compuesta por diferentes endpoints.

Cada archivo PHP tiene una función específica, por ejemplo:

- **login.php** > autenticación de usuarios.
- register.php > registro de nuevos usuarios.
- tickets.php > gestión de incidencias.
- update-ticket-status.php > actualización de estado.
- notifications.php > gestión de notificaciones.

Además, existe una carpeta de configuración (config/) que centraliza la conexión con la base de datos.

### Otros elementos

- **PHPMailer:** Librería utilizada para el envío de correos electrónicos desde el sistema.
- **Archivos HTML y públicos:** Permiten cargar la aplicación en el navegador.

### Conclusión sobre la estructura

Esta organización permite:

- Separar responsabilidades.
- Facilitar el mantenimiento.
- Escalar el sistema en el futuro.
- Trabajar de forma más ordenada.

## 5.2 Implementación de funcionalidades

Durante el desarrollo del proyecto se han implementado diversas funcionalidades clave. A continuación, se describen las más importantes, explicando su propósito y funcionamiento.

### Sistema de autenticación (registro e inicio de sesión)

El sistema de autenticación permite a los usuarios crear una cuenta y acceder a la plataforma de forma segura.

El objetivo de esta funcionalidad es controlar el acceso al sistema, asegurando que cada usuario tenga su propio espacio dentro de la aplicación.

El proceso funciona de la siguiente manera:

- El usuario introduce sus datos en un formulario del frontend (**React**).
- Estos datos se envían al backend mediante una petición HTTP.
- El endpoint correspondiente (**register.php o login.php**) procesa la información.
- En el registro, se crea un nuevo usuario en la base de datos.
- En el login, se validan las credenciales.

Además, el sistema permite gestionar la sesión del usuario mediante el uso de un contexto global en React (**AuthContext**), lo que permite mantener al usuario autenticado mientras navega por la aplicación.

## Sistema de gestión de tickets

Esta es la funcionalidad principal del sistema.

El objetivo es permitir a los usuarios registrar incidencias y realizar su seguimiento.

El funcionamiento es el siguiente:

- El usuario accede a un formulario donde describe su problema.
- La información se envía al backend (**tickets.php**).
- El servidor procesa la solicitud y la guarda en la base de datos.
- Se asigna un estado inicial al ticket (por ejemplo, “pendiente”).

Cada ticket queda asociado al usuario que lo ha creado, permitiendo su seguimiento posterior.

## Gestión del estado de los tickets

Una vez creadas las incidencias, es necesario gestionarlas.

El objetivo de esta funcionalidad es permitir actualizar el estado de cada ticket para reflejar su progreso.

Esto se realiza mediante:

- Peticiones al endpoint **update-ticket-status.php.**
- Modificación del estado en la base de datos.
- Actualización de la información en la interfaz.

Esto permite mantener un control organizado del proceso de resolución.

## Panel de administración

El panel de administración permite gestionar el sistema de forma global.

Su objetivo es ofrecer a los administradores una visión completa de todas las incidencias.

Desde este panel se pueden:

- Visualizar todos los tickets.
- Gestionar su estado.
- Supervisar el sistema.

El acceso a este panel está restringido mediante control de roles, asegurando que solo los usuarios autorizados puedan utilizarlo.

## Sistema de notificaciones

El sistema de notificaciones permite informar a los usuarios sobre cambios relevantes.

Su objetivo es mejorar la comunicación dentro del sistema.

El funcionamiento es el siguiente:

- Se genera una notificación cuando ocurre un evento importante (por ejemplo, cambio de estado de un ticket).

- Esta información se gestiona en el backend (**notifications.php**).
- Se muestra en el frontend mediante componentes específicos (como una campana de notificaciones).

Además, el usuario puede marcar las notificaciones como leídas.

## Sistema de envío de correos electrónicos

Se ha implementado un sistema de envío de correos utilizando PHPMailer.

El objetivo es permitir la comunicación externa desde la aplicación.

El proceso consiste en:

- Recoger datos de un formulario.
- Enviar la información al backend (**sendMail.php**).
- Utilizar PHPMailer para enviar el correo.

Esto añade una funcionalidad adicional que mejora la utilidad del sistema.

## Protección de rutas y control de acceso

Para garantizar la seguridad, se han implementado mecanismos de control de acceso en el frontend.

El objetivo es evitar accesos no autorizados.

Esto se logra mediante:

- Componentes como **ProtectedRoute.**
- Verificación del estado del usuario.
- Restricción de acceso según rol.

## 5.3 Pruebas

Las pruebas son una parte fundamental del desarrollo, ya que permiten verificar que el sistema funciona correctamente.

### Pruebas funcionales

Se han realizado pruebas para comprobar que cada funcionalidad cumple su objetivo:

- Registro de usuarios.
- Inicio de sesión.
- Creación de tickets.
- Consulta de incidencias.

### Pruebas de integración

Se ha verificado la correcta comunicación entre:

- Frontend (React).
- Backend (PHP).
- Base de datos.

Esto asegura que el sistema funciona como un todo.

### Pruebas en entorno real

Una vez desplegado el sistema en **techlan**.*es*, se han realizado pruebas en condiciones reales:

- Acceso desde navegador.
- Funcionamiento general.
- Compatibilidad con distintos dispositivos.

## Resultados

Los resultados han sido satisfactorios:

- El sistema funciona correctamente.
- No se han detectado errores críticos.
- La experiencia de usuario es adecuada.
- La aplicación es funcional en entorno real.

# 6. Conclusiones

## 6.1 Conclusiones generales

El desarrollo del proyecto Techlan ha permitido crear una plataforma web completa orientada a la gestión de incidencias de soporte técnico, cumpliendo con los objetivos planteados al inicio del proyecto.

A lo largo del desarrollo, se ha conseguido pasar de una idea inicial relativamente simple a una aplicación funcional, estructurada y desplegada en un entorno real. Este proceso ha implicado no solo la implementación de funcionalidades, sino también la toma de decisiones técnicas importantes, como la elección de tecnologías y la adopción de una arquitectura cliente-servidor.

Uno de los aspectos más relevantes del proyecto ha sido la evolución del sistema. En lugar de mantenerse en un enfoque básico, se ha optado por mejorar progresivamente la aplicación, incorporando tecnologías más avanzadas como React para el frontend y PHP como backend estructurado en forma de API. Esta evolución ha permitido desarrollar una solución más moderna, dinámica y cercana a un entorno profesional.

Además, el hecho de haber desplegado la aplicación en un dominio público aporta un valor añadido significativo, ya que demuestra que el sistema no solo funciona en un entorno de desarrollo, sino también en condiciones reales.

En conjunto, el proyecto ha resultado ser una experiencia completa de desarrollo de software, abarcando todas sus fases: análisis, diseño, implementación, pruebas y despliegue.

## 6.2 Consecución de objetivos

En relación con los objetivos definidos al inicio del proyecto, se puede afirmar que se han cumplido de manera satisfactoria.

El objetivo general, que consistía en desarrollar una plataforma web para la gestión de incidencias, se ha alcanzado mediante la implementación de un sistema funcional que permite a los usuarios crear, consultar y gestionar solicitudes de soporte técnico.

En cuanto a los objetivos específicos:

- Se ha implementado correctamente un sistema de registro e inicio de sesión, permitiendo gestionar usuarios de forma segura.
- Se ha desarrollado un sistema de gestión de incidencias (tickets), que constituye el núcleo de la aplicación.
- Se ha permitido el seguimiento del estado de las incidencias, mejorando la transparencia del sistema.
- Se ha creado un panel de administración que permite gestionar las solicitudes de forma global.
- Se ha implementado un sistema de notificaciones que mejora la comunicación entre el sistema y el usuario.
- Se ha garantizado el control de acceso mediante la diferenciación de roles.

- Se ha realizado el despliegue de la aplicación en un entorno real, cumpliendo uno de los objetivos más importantes del proyecto.

Por tanto, no solo se han alcanzado los objetivos mínimos, sino que en algunos casos se han superado, incorporando funcionalidades adicionales que enriquecen el sistema.

## 6.3 Valoración de la metodología y planificación

La metodología de trabajo utilizada, basada en un desarrollo iterativo, ha resultado adecuada para este tipo de proyecto.

El hecho de dividir el desarrollo en pequeñas fases o funcionalidades ha permitido avanzar de manera progresiva, facilitando la detección de errores y la mejora continua del sistema. En lugar de intentar desarrollar toda la aplicación de una sola vez, se ha optado por construirla paso a paso, lo que ha contribuido a una mejor organización del trabajo.

Además, la planificación por fases ha permitido mantener una estructura clara durante todo el desarrollo, aunque en algunos momentos ha sido necesario adaptarse a cambios, especialmente en la elección de tecnologías y en la fase de despliegue.

Uno de los aspectos más destacables ha sido la capacidad de adaptación del proyecto. La transición desde un enfoque inicial más simple hacia una arquitectura más compleja (React + API en PHP) demuestra una evolución en la forma de abordar el desarrollo, lo cual es un punto positivo desde el punto de vista formativo.

No obstante, también se pueden identificar posibles mejoras, como una planificación más detallada desde el inicio o una documentación más exhaustiva durante el desarrollo, lo que habría facilitado aún más la organización del proyecto.

## 6.4 Visión de futuro

El proyecto Techlan presenta múltiples posibilidades de mejora y ampliación en el futuro.

Una de las posibles líneas de evolución sería la mejora del sistema de seguridad, por ejemplo mediante la implementación de mecanismos más avanzados de autenticación, como el uso de tokens o sistemas de verificación en dos pasos.

También se podría mejorar la escalabilidad del backend, migrando a tecnologías más orientadas a APIs modernas o frameworks avanzados, lo que permitiría soportar un mayor número de usuarios y funcionalidades.

# 7. Glosario

A continuación, se presentan definiciones de los principales términos técnicos utilizados a lo largo del documento, con el objetivo de facilitar la comprensión del proyecto.

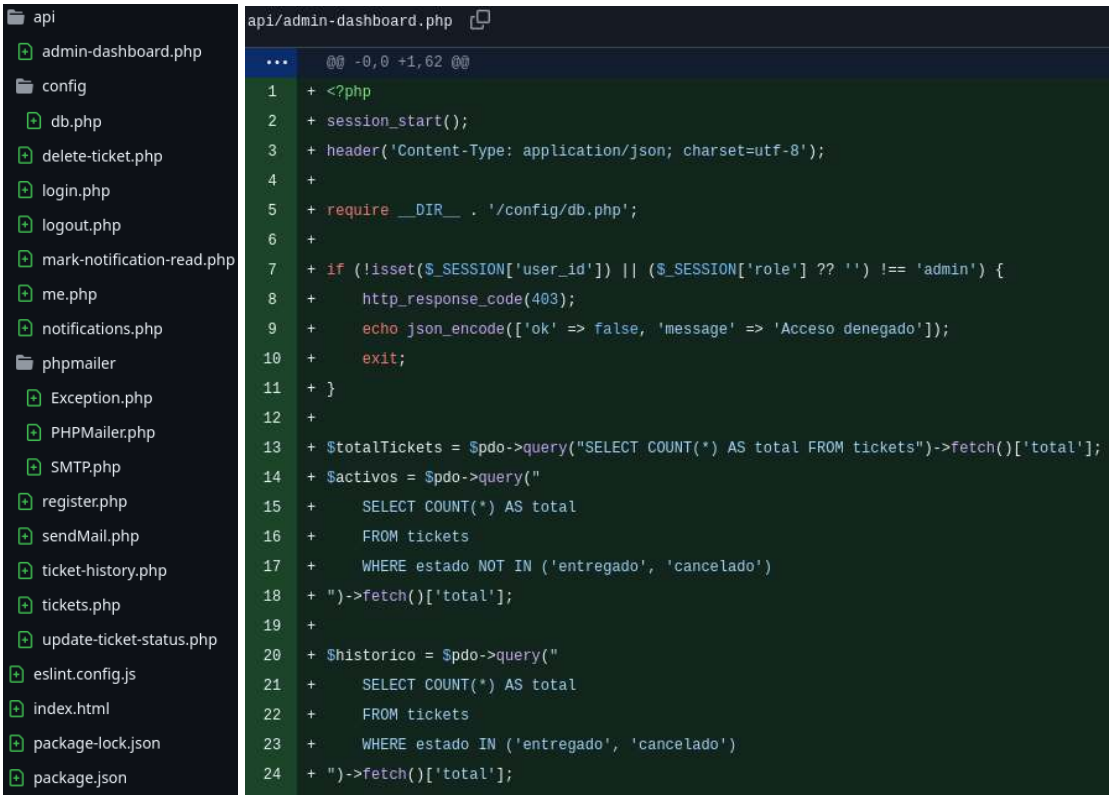
- **API (Application Programming Interface):** Conjunto de reglas y mecanismos que permiten que diferentes aplicaciones se comuniquen entre sí.
- **Backend:** Parte del sistema encargada de gestionar la lógica interna, procesar datos y comunicarse con la base de datos.
- **Base de datos:** Sistema que permite almacenar información de forma organizada y persistente.
- **Componente (React):** Bloque reutilizable de código que forma parte de la interfaz de usuario.
- **Frontend:** Parte de la aplicación con la que interactúa el usuario.
- **Hosting:** Servicio que permite alojar una página web en internet para que sea accesible públicamente.
- **HTTP (HyperText Transfer Protocol):** Protocolo de comunicación utilizado para el intercambio de información en la web.
- **JSON (JavaScript Object Notation):** Formato de intercambio de datos ligero y fácil de leer.
- **PHP (Hypertext Preprocessor):** Lenguaje de programación utilizado para el desarrollo del backend.
- **React:** Librería de JavaScript utilizada para construir interfaces de usuario dinámicas.
- **Servidor:** Sistema que recibe peticiones de los clientes y devuelve respuestas.
- **SPA (Single Page Application):** Tipo de aplicación web que funciona en una única página, sin recargar completamente el contenido.
- **Ticket (incidencia):** Solicitud de soporte técnico creada por un usuario dentro del sistema.
- **Token (autenticación):** Elemento de seguridad utilizado para identificar a un usuario autenticado.

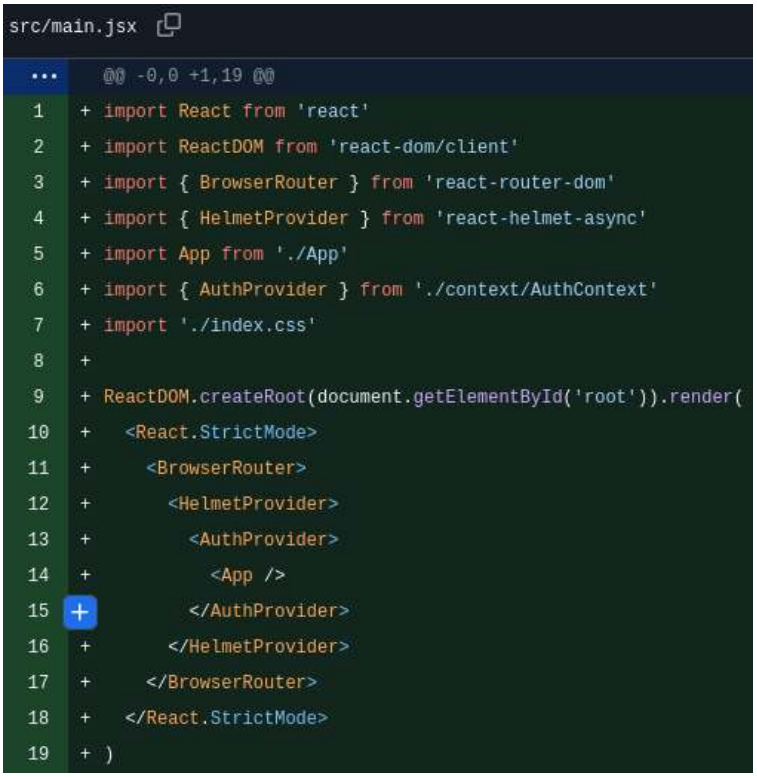
# 8. Bibliografía

Estas fuentes han servido como apoyo para la comprensión de las tecnologías empleadas, la resolución de problemas y la mejora del sistema.

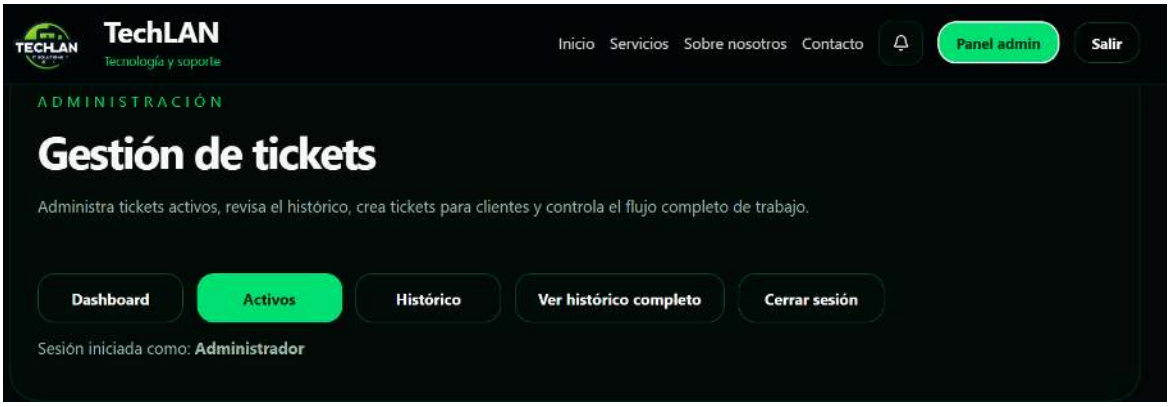
- React Documentation. (s.f.). *React – A JavaScript library for building user interfaces*. https://react.dev/
- PHP Manual. (s.f.). *PHP: Hypertext Preprocessor*. https://www.php.net/manual/es/
- MDN Web Docs. (s.f.). *Mozilla Developer Network*. https://developer.mozilla.org/es/
- W3Schools. (s.f.). *SQL Tutorial*. https://www.w3schools.com/sql/
- Stack Overflow. (s.f.). *Preguntas y respuestas sobre programación*. https://stackoverflow.com/
- W3Schools. (s.f.). *HTML, CSS y JavaScript Tutorial*. https://www.w3schools.com/
- FreeCodeCamp. (s.f.). *Learn to Code — for Free*. https://www.freecodecamp.org/
- PHPMailer GitHub Repository. (s.f.). *PHPMailer*. https://github.com/PHPMailer/PHPMailer
- GitHub. (s.f.). *Plataforma de desarrollo colaborativo*. https://github.com/

# 9. Anexos


api
admin-dashboard.php
config
db.php
delete-ticket.php
login.php
logout.php
mark-notification-read.php
me.php
notifications.php
phpmailer
Exception.php
PHPMailer.php
SMTP.php
register.php
sendMail.php
ticket-history.php
tickets.php
update-ticket-status.php
eslint.config.js
index.html
package-lock.json
package.json
api/admin-dashboard.php
@@ -0,0 +1,62 @@
+ <?php
+ session_start();
+ header('Content-Type: application/json; charset=utf-8');
+
+ require __DIR__ . '/config/db.php';
+
+ if (!isset($_SESSION['user_id']) || ($_SESSION['role'] ?? '') !== 'admin') {
+ http_response_code(403);
+ echo json_encode(['ok' => false, 'message' => 'Acceso denegado']);
+ exit;
+ }
+
+ $totalTickets = $pdo->query("SELECT COUNT(*) AS total FROM tickets")->fetch()['total'];
+ $activos = $pdo->query("
+ SELECT COUNT(*) AS total
+ FROM tickets
+ WHERE estado NOT IN ('entregado', 'cancelado')
+ ")->fetch()['total'];
+
+ $historico = $pdo->query("
+ SELECT COUNT(*) AS total
+ FROM tickets
+ WHERE estado IN ('entregado', 'cancelado')
+ ")->fetch()['total'];


src/main.jsx
@@ -0,0 +1,19 @@
+ import React from 'react'
+ import ReactDOM from 'react-dom/client'
+ import { BrowserRouter } from 'react-router-dom'
+ import { HelmetProvider } from 'react-helmet-async'
+ import App from './App'
+ import { AuthProvider } from './context/AuthContext'
+ import './index.css'
+
+ ReactDOM.createRoot(document.getElementById('root')).render(
+ <React.StrictMode>
+ <BrowserRouter>
+ <HelmetProvider>
+ <AuthProvider>
+ <App />
+ </AuthProvider>
+ </HelmetProvider>
+ </BrowserRouter>
+ </React.StrictMode>
+ )


Sobre nosotros
Una tienda informática nueva, con imagen moderna y vocación de servicio.
QUIÉNES SOMOS
Un proyecto con mentalidad tecnológica y profesional
TechLAN nace con la intención de ofrecer un servicio técnico cercano, moderno y claro. Apostamos por una forma de trabajar más directa, profesional y adaptada a lo que necesita cada cliente.
Queremos ayudarte con reparaciones, montaje de equipos, actualización de componentes y gestión de compras tecnológicas, cuidando tanto la parte técnica como la atención al cliente.
Atención personalizada desde el primer contacto.
Imagen actual, tecnológica y profesional.
Compromiso con un servicio claro y bien explicado.


TECHLAN
TechLAN
Tecnología y soporte
Inicio
Servicios
Sobre nosotros
Contacto
Panel admin
Salir
ADMINISTRACIÓN
Gestión de tickets
Administra tickets activos, revisa el histórico, crea tickets para clientes y controla el flujo completo de trabajo.
Dashboard
Activos
Histórico
Ver histórico completo
Cerrar sesión
Sesión iniciada como: Administrador

## Nota sobre el uso de inteligencia artificial

Durante el desarrollo del proyecto *Techlan* se han utilizado herramientas de inteligencia artificial como apoyo puntual en determinadas tareas, principalmente relacionadas con la redacción, organización de ideas y mejora de la documentación.

El uso de estas herramientas ha estado orientado a facilitar la estructuración de la memoria, mejorar la claridad en la expresión de conceptos técnicos y revisar la coherencia del contenido. En ningún caso se ha utilizado la inteligencia artificial como sustituto del trabajo realizado, sino como una herramienta de apoyo complementaria.

Todo el desarrollo técnico del proyecto, incluyendo la programación del frontend en React, la implementación del backend en PHP, la gestión de la base de datos y el despliegue en un entorno real, ha sido realizado por los autores con apoyo de inteligencia artificial como apoyo. Asimismo, todo el contenido incluido en esta memoria ha sido revisado, comprendido y adaptado por los integrantes del proyecto.

# Licencia