MEMORIA TÉCNICA: VEREDILLAS FM

Desarrollo e Implementación de Plataforma Web para Radiodifusión Educativa

ÍNDICE:

ÍNDICE: 2
RESUMEN EJECUTIVO 3
1. INTRODUCCIÓN Y CONTEXTO 4
1.1 Antecedentes y Motivación 4
1.2 Objetivos del Proyecto 4
2. ANÁLISIS TECNOLÓGICO (STACK) 5
2.1 Frontend y Meta-Framework: Astro 5
2.2 Lenguaje de Programación: TypeScript 5
2.3 Base de Datos: MongoDB (NoSQL) 5
2.4 Automatización e Infraestructura: n8n y Oracle Cloud 6
2.5 Entorno de Desarrollo (IDE): Google Antigravity 6
3. ARQUITECTURA Y DISEÑO 7
3.1 Arquitectura del Software 7
3.2 Diseño de Interfaz (UI/UX) y Estética 7
4. IMPLEMENTACIÓN Y LÓGICA 8
4.1 Metodología de Desarrollo 8
4.2 Funcionalidad Crítica: Sistema de Verificación de Comentarios 8
5. DESAFÍOS TÉCNICOS Y SOLUCIONES DE INFRAESTRUCTURA 9
5.1 El Problema: Bloqueo de Infraestructura Distribuida 9
5.2 La Solución de Ingeniería de Red 9
6. DESGLOSE DETALLADO DE FUNCIONALIDADES (FEATURES) 10
6.1 Experiencia de Usuario (UX) e Interfaz (UI) 10
6.2 Interactividad en directo 11
6.3 Motor de Audio y Podcast 12
6.4 Usuarios, Autenticación y Comunidad 13
6.5 Backend, Datos e Infraestructura 13
6.6 Extras y "Easter Eggs" 15
7. ASEGURAMIENTO DE LA CALIDAD (QA) Y DESPLIEGUE CONTINUO 17
7.1 Testing Automatizado y Pipeline CI/CD 17
7.2 Monitorización y Analítica Post-Despliegue 17
8. CONCLUSIONES Y VISIÓN DE FUTURO 18
8.1 Conclusiones del Proyecto 18
8.2 Hoja de Ruta (Roadmap) 18
9. AGRADECIMIENTOS 19
10. ENLACES DE INTERÉS 19

RESUMEN EJECUTIVO

El presente documento técnico describe de manera exhaustiva el ciclo de vida completo, desde la concepción hasta el despliegue en producción, de **Veredillas FM**, una plataforma web avanzada dedicada a la gestión, difusión y consumo de contenidos en el entorno educativo. Este proyecto nace como respuesta a la necesidad de difundir el contenido más allá de los muros del instituto y la comunidad escolar, representando una evolución tecnológica sustancial respecto a soluciones heredadas y limitadas como Blogger.

La solución final implementada se erige sobre una arquitectura web moderna, escalable y de alto rendimiento, fundamentada en el meta-framework **Astro**, una base de datos documental **MongoDB**, y un sistema de orquestación de procesos de negocio automatizados mediante **n8n**. El sistema no solo cumple con los requisitos funcionales básicos de reproducción de audio, sino que integra funcionalidades complejas de "backend-as-a-service", incluyendo un robusto sistema de verificación de identidad para comentarios, sincronización de contenidos multimedia mediante integración con la API de Spotify, y una interfaz de usuario altamente adaptativa basada en los principios estéticos del *Glass Morphism*. Este documento justifica cada decisión de ingeniería, analizándolas desde perspectivas de rendimiento, seguridad, costo y mantenibilidad.

1. INTRODUCCIÓN Y CONTEXTO

1.1 Antecedentes y Motivación

Originalmente, la radio escolar "Veredillas FM" dependía de una infraestructura basada en Blogger. Si bien esta plataforma permitió los primeros pasos digitales del programa, rápidamente se evidenciaron limitaciones críticas que frenaban el crecimiento del proyecto: imposibilidad de gestionar bases de datos de usuarios, dificultades para integrar reproductores personalizados, falta de identidad de marca cohesiva y una nula capacidad para implementar lógica de negocio del lado del servidor.

La carencia de interactividad real convertía a la web en un tablón de anuncios estático, unidireccional y poco atractivo para una audiencia joven. El objetivo fundamental de este proyecto fue transformar esa presencia digital pasiva en un ecosistema dinámico y propio. Se buscó desarrollar una plataforma que no solo centralizara los episodios, sino que actuara como un hub social para la comunidad educativa (profesores, alumnos e invitados), profesionalizando la imagen del programa y permitiendo la implementación de características avanzadas de software a medida.

1.2 Objetivos del Proyecto

El desarrollo se articuló en torno a cuatro pilares estratégicos:

- **Centralización y Soberanía de Contenidos:** Crear un repositorio unificado que, aunque se alimente de fuentes externas (Spotify), permita un control total sobre cómo se presentan y organizan los datos, eliminando la dependencia exclusiva de las interfaces de terceros.
- **Interacción Segura con la Audiencia:** Implementar canales de comunicación bidireccional (formularios de contacto y sistemas de comentarios) protegidos por capas de seguridad lógica para verificar la identidad de los emisores y prevenir el spam o el abuso.
- **Fidelización y Retención:** Desarrollar mecanismos de *engagement* automatizados, específicamente un sistema de Newsletter que notifique a los suscriptores sobre nuevos lanzamientos sin intervención manual constante.
- **Excelencia en Experiencia de Usuario (UX/UI):** Proveer una interfaz gráfica vanguardista, accesible según estándares web, y totalmente responsiva, asegurando una experiencia fluida tanto en dispositivos móviles como en escritorio.

2. ANÁLISIS TECNOLÓGICO (STACK)

La selección del stack tecnológico no fue arbitraria, sino el resultado de un análisis comparativo basado en criterios de rendimiento (Core Web Vitals), escalabilidad futura y facilidad de mantenimiento.

2.1 Frontend y Meta-Framework: Astro

Se seleccionó **Astro** como el núcleo arquitectónico del proyecto, priorizando sobre otras opciones populares como React o Next.js debido a su innovadora **"Islands Architecture"** (Arquitectura de Islas).

- **Justificación de Rendimiento (Zero JS by Default):** A diferencia de las Single Page Applications (SPA) tradicionales que obligan al navegador a descargar y ejecutar grandes paquetes de JavaScript antes de mostrar contenido, Astro renderiza HTML estático en el servidor por defecto. Solo los componentes estrictamente interactivos (como el reproductor de audio o el formulario de validación) son "hidratados" con JavaScript en el cliente. Esto reduce drásticamente el *Time to Interactive* (TTI) y mejora el SEO, factores críticos para un sitio de contenido público.
- **Estrategia de Renderizado Híbrido:** Se implementó una configuración mixta. Las páginas de contenido estático (landing, información institucional) utilizan **SSG (Static Site Generation)** para máxima velocidad, mientras que las rutas que requieren interacción con la base de datos o APIs externas utilizan **SSR (Server Side Rendering)**, permitiendo dinamismo en tiempo real sin sacrificar el rendimiento global.

2.2 Lenguaje de Programación: TypeScript

La totalidad del código fuente, tanto en los componentes de interfaz como en la lógica de servidor, fue escrita en **TypeScript**.

- **Robustez y Calidad:** El uso de tipado estático fuerte permite definir interfaces claras para los modelos de datos (ej: interface Comment, interface Episode). Esto facilita la detección de errores en tiempo de compilación —antes de que lleguen a producción— y mejora la mantenibilidad del código al servir como documentación autogenerada para futuros desarrolladores del proyecto.

2.3 Base de Datos: MongoDB (NoSQL)

- **Elección del Modelo de Datos:** Se optó por **MongoDB**, una base de datos NoSQL orientada a documentos. Esta elección se justifica por la naturaleza variable y semi-estructurada de los datos gestionados (comentarios con metadatos variables, registros de logs).
- **Integración:** El formato BSON de MongoDB se alinea naturalmente con las estructuras de objetos JSON utilizadas en JavaScript/TypeScript, eliminando la necesidad de complejos ORMs (Object-Relational Mappers) y permitiendo un desarrollo más ágil y directo. Además, su capacidad de escalado horizontal asegura que el sistema pueda manejar picos de tráfico sin degradación del servicio.

2.4 Automatización e Infraestructura: n8n y Oracle Cloud

Para la lógica de negocio desatendida y la integración de servicios dispares, se rechazaron soluciones SaaS costosas (como Zapier) en favor de una instancia *self-hosted* de **n8n**.

- **Soberanía de Infraestructura:** El servicio se aloja en un VPS (Virtual Private Server) en **Oracle Cloud**. Esto proporciona control total sobre los recursos computacionales y la seguridad de los datos.
- **Orquestación de Procesos:** n8n actúa como un *middleware* (puerta de enlace) de automatización, desacoplando tareas pesadas (como el envío masivo de correos o la gestión de listas de suscriptores) del hilo principal de la aplicación web. Esto evita que la experiencia del usuario se vea ralentizada por procesos de fondo y permite modificar los flujos de negocio visualmente sin necesidad de redesplegar el código fuente de la web.

2.5 Entorno de Desarrollo (IDE): Google Antigravity

Como apuesta por la vanguardia en herramientas de ingeniería de software, se utilizó **Google Antigravity** (IDE de Google). Este entorno de desarrollo basado en WindSurf, que a su vez es una versión de VS Code, capaz de integrar asistencia por IA directamente en el flujo de trabajo.

3. ARQUITECTURA Y DISEÑO

3.1 Arquitectura del Software

El sistema sigue un patrón de diseño de **Monolito Modular con Servicios Desacoplados**. Aunque el código base reside en un repositorio unificado en Github (monorepo), las responsabilidades están claramente segregadas y separadas.

- **Capa de Presentación y API (Astro):** Actúa como el servidor principal. Gestiona las rutas de la API (endpoints .ts) que funcionan bajo el patrón *Backend for Frontend* (BFF). Estas rutas protegen las credenciales de la base de datos y actúan como intermediarios seguros, saneando las entradas del usuario antes de interactuar con la persistencia.
- **Capa de Servicios Externos:** La aplicación no es un silo aislado; orquesta peticiones a MongoDB para persistencia, APIs de Google (Calendar para agenda, Analytics para métricas) y el servidor de automatización n8n para mensajería transaccional.

3.2 Diseño de Interfaz (UI/UX) y Estética

El diseño visual no es meramente decorativo, sino funcional. Se fundamenta en el estilo **Glass Morphism**, inspirado en las guías de diseño humano de las versiones recientes de iOS.

- **Principios Visuales:** Se basa en el uso de capas translúcidas, desenfoque de fondo en tiempo real (backdrop-filter: blur()), y bordes sutiles para establecer jerarquía visual y profundidad. Esto crea una sensación de modernidad y ligereza.
- **Accesibilidad y Adaptabilidad:** Se implementó un sistema de variables CSS robusto para soportar el cambio de tema (Claro/Oscuro) sin fricción. Todo el diseño es *mobile-first*, garantizando que los controles interactivos tengan áreas táctiles adecuadas (mínimo 44x44px) en dispositivos táctiles.
- **Herramientas de Aceleración:** Se utilizó **Figma** para el prototipado de alta fidelidad y herramientas de IA generativa (**v0**, **Gemini**) para convertir esos diseños en componentes de código (Tailwind CSS) optimizados, reduciendo significativamente el tiempo de desarrollo del frontend.

4. IMPLEMENTACIÓN Y LÓGICA

4.1 Metodología de Desarrollo

El proyecto no siguió un modelo en cascada rígido, sino una metodología de **Desarrollo Iterativo e Incremental**. Este enfoque ágil permitió evolucionar orgánicamente: comenzando con una prueba de concepto estática, migrando progresivamente funcionalidades al backend, y refinando la UX base a feedback real. El control de versiones mediante **Git** y el alojamiento en **GitHub** facilitaron la gestión de ramas para nuevas características, asegurando que la rama principal (*master*) siempre contuviera código estable.

4.2 Funcionalidad Crítica: Sistema de Verificación de Comentarios

Uno de los desafíos de ingeniería más complejos fue asegurar la calidad y seguridad de los contenidos generados por el usuario (UGC). Para ello, se diseñó un protocolo de verificación de doble factor vía correo electrónico:

1. **Ingesta de Datos:** El usuario envía el comentario y su email a través de un formulario protegido contra CSRF.
2. **Generación de Credenciales:** El backend de Astro genera un token criptográfico de alta entropía (usando librerías estándar de criptografía) y almacena el comentario en MongoDB con un estado inicial pending y una fecha de expiración (TTL).
3. **Disparo de Evento (Webhook):** El backend invoca un webhook seguro en la instancia de n8n, pasando el email y el token cifrado.
4. **Procesamiento Asíncrono:** n8n recibe la señal, maqueta un correo electrónico HTML responsive y utiliza la API transaccional de Mailjet para enviarlo al usuario con un enlace único:

   ´https://veredillasfm.es/verify-comments?email=[email]&token=[token]´

5. **Validación y Publicación:** Al hacer clic, el usuario golpea un endpoint de verificación en Astro. El sistema busca el token en la base de datos, verifica que no haya expirado y, si coincide, realiza una operación atómica de actualización (findOneAndUpdate) cambiando el estado a "*verified*". Solo entonces el comentario es visible públicamente.

5. DESAFÍOS TÉCNICOS Y SOLUCIONES DE INFRAESTRUCTURA

Durante la fase de integración, el obstáculo más significativo estuvo relacionado con la **seguridad perimetral y las políticas de CORS (Cross-Origin Resource Sharing)** entre entornos de nube heterogéneos.

5.1 El Problema: Bloqueo de Infraestructura Distribuida

La comunicación entre el servidor de despliegue (**Vercel**, arquitectura Serverless) y el servidor de automatización (**n8n en Oracle Cloud**, arquitectura IaaS) fallaba de manera intermitente y crítica.

- **Diagnóstico Profundo:** Tras analizar los logs de acceso y trazas de red, se determinó que múltiples capas de seguridad estaban entrando en conflicto. El sistema de seguridad de n8n, las reglas de Firewall (Security Lists) de la VPS en Oracle, y las protecciones anti-bot del WAF de Cloudflare interpretaban las peticiones provenientes de las IPs dinámicas y rotativas de Vercel como tráfico malicioso o ataques DDoS, procediendo a dropear los paquetes.

5.2 La Solución de Ingeniería de Red

Se implementó una solución multicapa a nivel de infraestructura:

1. **Identificación de Origen:** Se realizó un análisis de los rangos CIDR de direcciones IP públicas utilizados por la región de despliegue de Vercel.
2. **Configuración de Listas Blancas (Whitelisting):** Se modificaron las reglas de entrada (Ingress Rules) en la Virtual Cloud Network (VCN) de Oracle para permitir explícitamente el tráfico TCP en el puerto del webhook exclusivamente desde los rangos IP identificados.
3. **Ajuste Fino del WAF:** Se calibraron las reglas heurísticas del Web Application Firewall de Cloudflare, creando excepciones específicas para las firmas de las peticiones API legítimas del sistema, permitiendo el paso de la automatización sin comprometer la seguridad global contra bots reales.

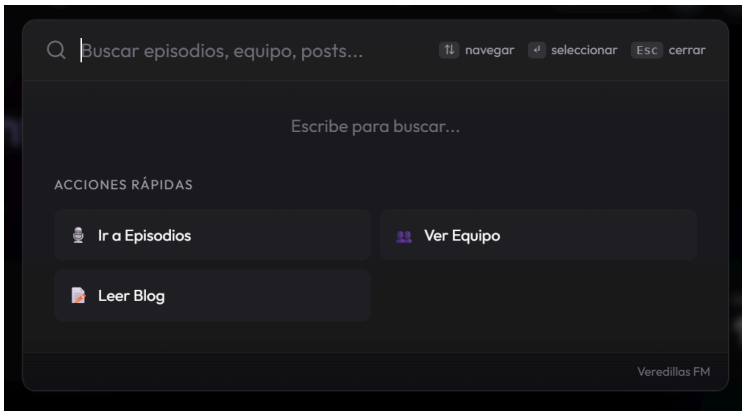
6. DESGLOSE DETALLADO DE FUNCIONALIDADES (FEATURES)

El ecosistema de Veredillas FM integra un conjunto de funcionalidades avanzadas diseñadas para maximizar la retención de usuarios, asegurar la escalabilidad técnica y ofrecer una experiencia de "App Nativa" en la web. A continuación, se detallan las características clave implementadas.

6.1 Experiencia de Usuario (UX) e Interfaz (UI)

La plataforma utiliza tecnologías de vanguardia para garantizar una navegación fluida y una estética moderna.

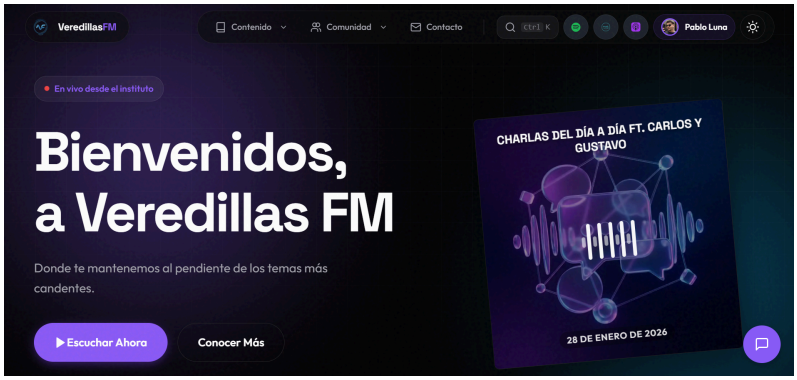
- **Arquitectura de Islas (Astro Islands):** El sitio maximiza el rendimiento cargando HTML estático por defecto. El JavaScript sólo se usa en componentes específicos (widgets de chat, reproductores, contadores), permitiendo una carga ultra-rápida.
- **Navegación Fluida con View Transitions:** Implementación de transiciones de página que eliminan el "parpadeo" blanco al navegar. El contenido se desliza suavemente y el audio no se interrumpe, simulando el comportamiento de una Single Page Application (SPA).
- **Buscador Inteligente (Command Palette):**
    - **Accesibilidad:** Accesible mediante Ctrl + K o icono dedicado (CommandPalette.astro).
    - **Búsqueda Difusa (Fuzzy Search):** Utiliza el algoritmo de Fuse.js para tolerar errores tipográficos y encontrar episodios, posts o miembros del equipo de forma instantánea.



- **Estética "Glassmorphism" y Diseño Dinámico:**
    - **Capas de Vidrio:** Paneles semitransparentes con desenfoque de fondo (backdrop-filter: blur) aplicados en tarjetas y modales.
    - **Fondos Animados "Blob":** Esferas de color difusas que flotan dinámicamente

en el fondo (Layout.astro), creando una atmósfera envolvente.

- **Sistema de Temas (Dark Mode):** Interruptor manual con persistencia en localStorage, adaptando toda la paleta de colores y efectos visuales a la preferencia del usuario.

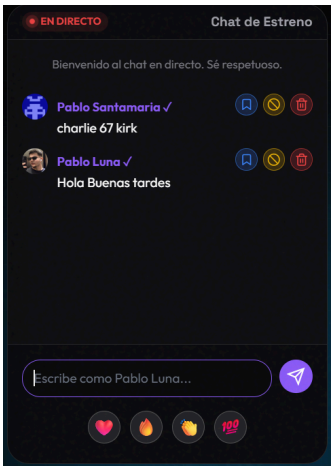


- **Sistema de Notificaciones (Toasts):** Alertas emergentes globales (Toast.astro) para confirmar acciones como "Mensaje enviado" o "Sesión iniciada".

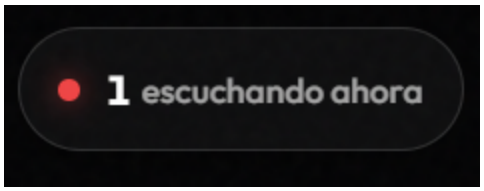
6.2 Interactividad en directo

Veredillas FM utiliza diferentes componentes para mantener el sitio web vivo y donde todos los usuarios tengan la posibilidad de interactuar con el resto sin la necesidad de crearse una cuenta.

- **Chat de Estreno:** Widget de chat sincronizado (LiveChat.astro) con soporte para modo invitado y mensajes fijados (pinned).



- **Reacciones Flotantes (Flying Emojis):** Sistema de partículas animadas (corazones, fuego) que aparecen en la pantalla de todos los espectadores al pulsar los botones de reacción.
- **Contador de Espectadores:** Widget ListenerCount que muestra la audiencia en tiempo real de ese episodio.



- **Optimistic UI:** Los mensajes del chat aparecen instantáneamente para el autor mientras se confirman en el servidor, eliminando la sensación de lag.

6.3 Motor de Audio y Podcast

Optimización de la escucha de contenido bajo demanda (VOD).

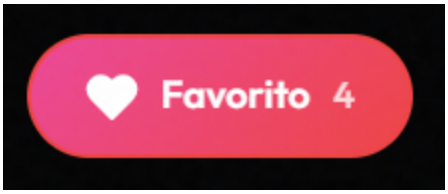
- **MiniPlayer Persistente:** Barra de reproducción fija (MiniPlayer.astro) que permite seguir escuchando un episodio mientras el usuario explora otras secciones del sitio.
- **Gestión de Metadatos y Estado:** Actualización en tiempo real de carátulas y títulos sincronizados con la API de contenidos. Incluye botones de salto rápido (+/- 10 segundos).
- **Integración con Spotify:** Embeds nativos para aquellos episodios que requieren distribución multiplataforma (DEPRECADO).
    - Actualmente se utiliza un sistema de scraping al RSS de Anchor.fm (Spotify Creators) para extraer el enlace de audio (https://anchor.fm/mi-audio.mp3), para poder tener acceso al audio y poder desarrollar herramientas mucho más complejas
- **Accesibilidad y SEO:** Transcripciones desplegables en cada episodio para facilitar la

lectura y mejorar la indexación de palabras clave por los motores de búsqueda.

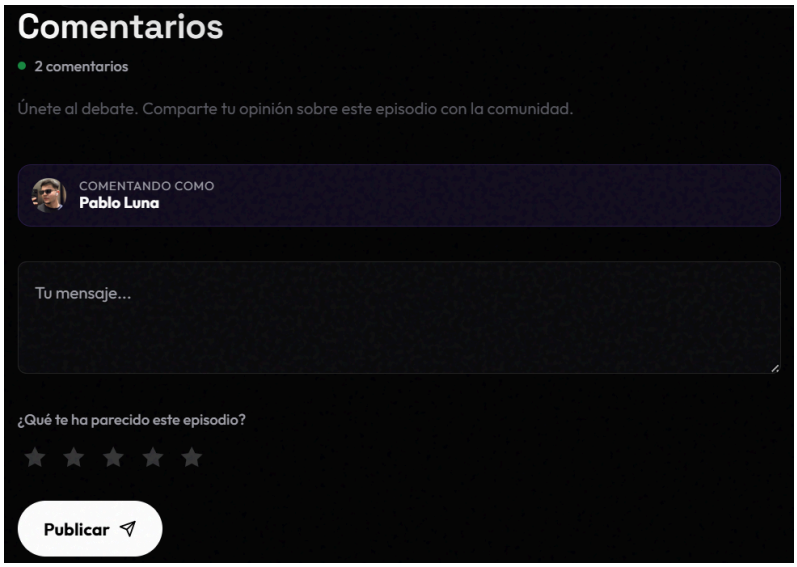
6.4 Usuarios, Autenticación y Comunidad

Sistema robusto de gestión de usuarios para fomentar la fidelización.

- **Autenticación Segura:** Registro y login con hashing de datos sensibles y gestión de sesiones mediante **Cookies HTTP-Only**, protegiendo el token contra ataques XSS.
- **Perfiles Personalizados:**
    - **Avatares Dinámicos:** Generación automática de fotos de perfil basadas en iniciales mediante la API de ui-avatars para usuarios sin imagen.
    - **Favoritos:** Los usuarios pueden guardar episodios en su lista personal (persistencia en MongoDB).



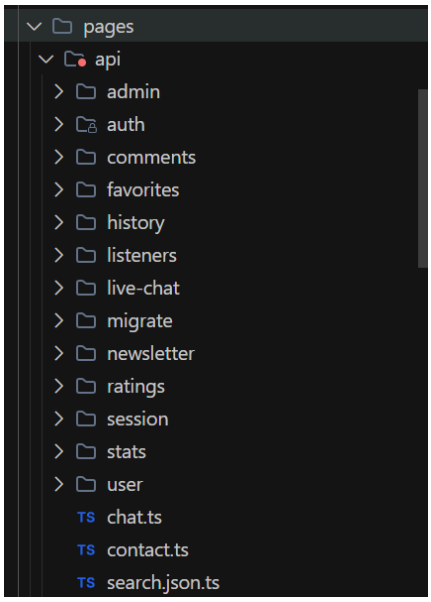
- **Interacción Social:**
    - **Hilos de Comentarios:** Sistema de discusión estructurado en episodios y artículos de blog.
    - **Valoración (Ratings):** Componente EpisodeRating.astro para puntuar contenidos del 1 al 5.



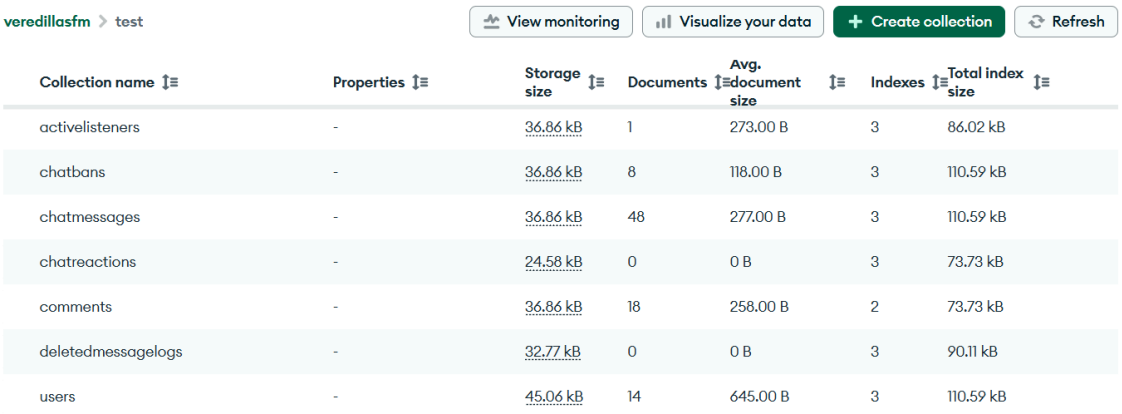
6.5 Backend, Datos e Infraestructura

La arquitectura "invisible" que garantiza la velocidad y seguridad.

- **API REST Interna:** Endpoints modulares en /pages/api/ que gestionan la lógica de negocio (auth, chat, likes) de forma independiente al frontend.



- **Base de Datos Híbrida:**
  - **MongoDB (Mongoose):** Para datos dinámicos como usuarios, comentarios y mensajes de chat.



Collection name	Properties	Storage size	Documents	Avg. document size	Indexes	Total index size
activelisteners	-	36.86 kB	1	273.00 B	3	86.02 kB
chatbans	-	36.86 kB	8	118.00 B	3	110.59 kB
chatmessages	-	36.86 kB	48	277.00 B	3	110.59 kB
chatreactions	-	24.58 kB	0	0 B	3	73.73 kB
comments	-	36.86 kB	18	258.00 B	2	73.73 kB
deletedmessagelogs	-	32.77 kB	0	0 B	3	90.11 kB
users	-	45.06 kB	14	645.00 B	3	110.59 kB

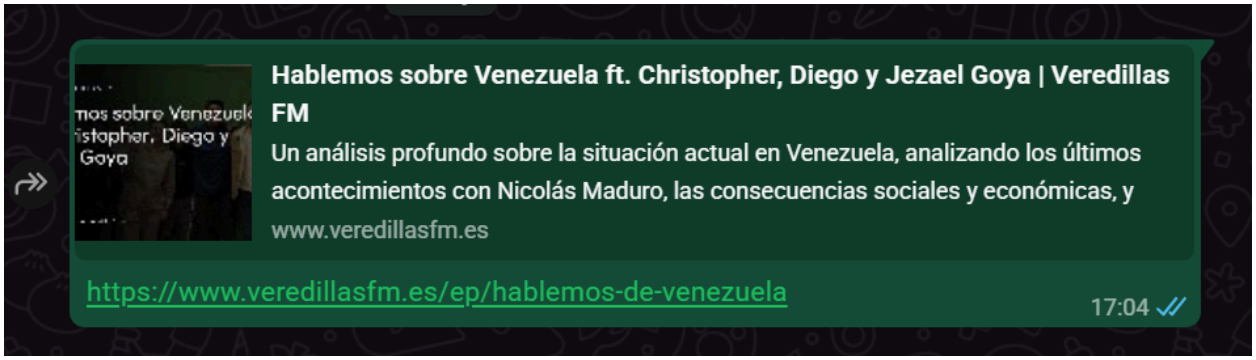
  - **Content Collections (Markdown):** Los episodios y posts se gestionan como archivos .md (MarkDown) compilados, garantizando tiempos de respuesta instantáneos, gracias a su generación al desplegarlos a Producción.

biologo-guia-informatica-alejandro.md

src > content > episodios > biologo-guia-informatica-alejandro.md

```
---
title: "El Biólogo que nos guía en la Informática ft. Alejandro"
description: "Hoy, no vamos a hablar solo de procesadores o código binario, sino de la persona que está detrás del teclado: el Profesor Alejandro, nuestro querido profesor de Informática."
pubDate: 2025-11-25
image: "https://blogger.googleusercontent.com/img/b/R29vZ2xl/AVvXsEjnLK1u1y6yDhK6yumKrztTdimUtq91FmdvK2mHJ9zZyRKmYO8JBZmrAuDorqXAKUVkayN4uV5EcG7sOPOTVZCTqeOISM7jrzSxEac-RItneKwC4Q3NS0S8eL0ioG2v5V8ys6YPOAD9RBjrlkbtWSDp1SmoBxOX46qPTppBbr-sqsBQuLvcCWw-TmEW8DI/w640-h640/Imagen%20de%20WhatsApp%202025-12-10%20a%20las%2012.07.00_ac94ad9a.jpg"
audioUrl: "https://anchor.fm/s/10ca1a038/podcast/play/112421764/https%3A%2F%2Fd3ctxlq1ktw2nl.cloudfront.net%2Fstaging%2F2025-11-10%2F7b84ea23-3111-4257-5de9-75f6df3f7128.mp3"
season: 1
episode: 2
duration: "37 min"
participants: ["Prof. Alejandro"]
tags: ["Profesorado", "Informática", "Ciencia", "Profesorado"]
```

- **SEO Técnico Avanzado:**
  - **OG Image Dinámica (Satori):** Cada episodio genera automáticamente su propia tarjeta social al compartirse en redes (WhatsApp, X, LinkedIn...), integrando título y carátula por código.



  - **RSS Feed & Sitemap:** Generación automatizada de rss.xml.js para apps de podcast (Apple Podcasts) y sitemap-index.xml para Google.
- **Seguridad y Rendimiento:**
  - **Rate Limiting:** Protección contra ataques de spam y DoS en formularios y chat.
  - **Lazy Loading Nativo:** Carga diferida de imágenes para optimizar las métricas de *Core Web Vitals* (LCP).

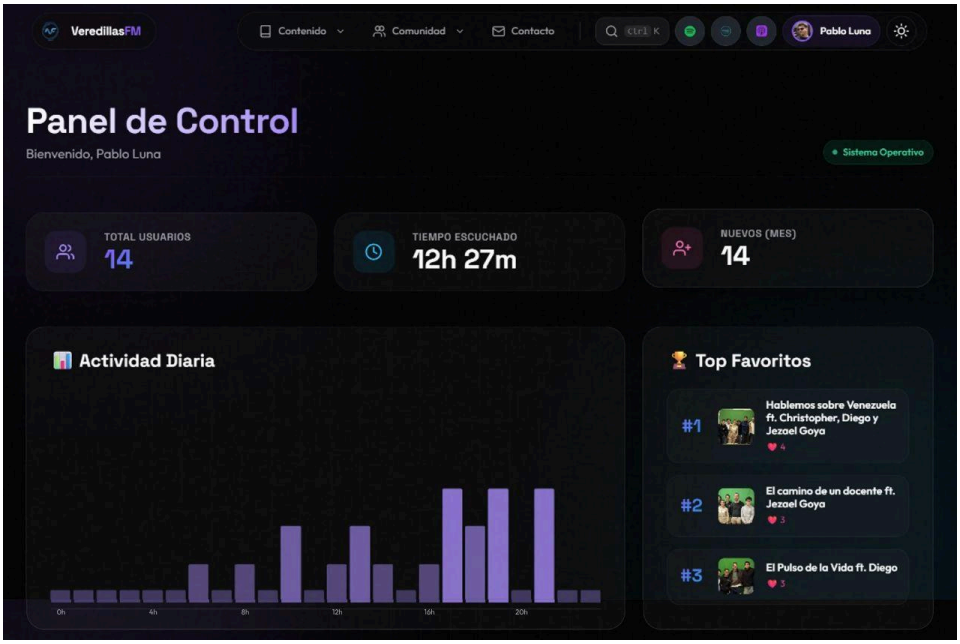
6.6 Extras y "Easter Eggs"

Detalles que refuerzan la identidad digital del proyecto.

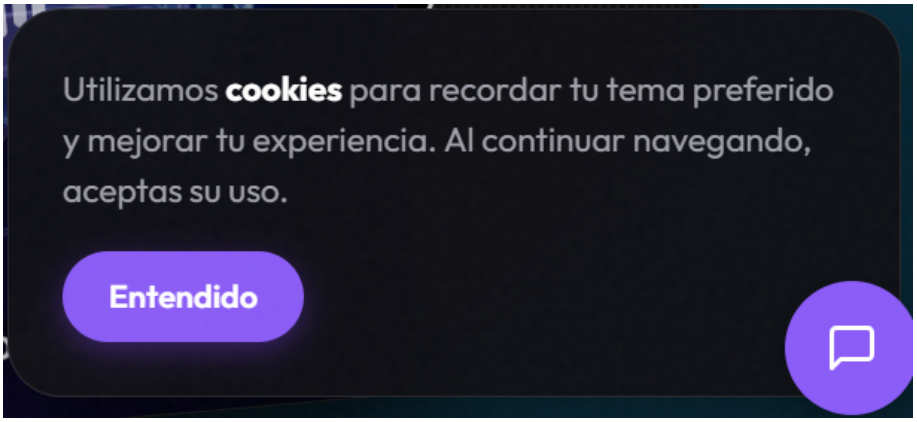
- **Konami Code:** Script oculto (KonamiCode.astro) que desbloquea funciones especiales o efectos visuales tras la secuencia clásica de teclas.
- **Efectos de Celebración:** Uso de canvas-confetti para hitos especiales o

confirmaciones de acciones de usuario.

- **Admin Dashboard:** Panel de administración privado para la gestión de usuarios, moderación de comentarios y visualización de estadísticas de la plataforma.



- **Cumplimiento Legal:** Banner de consentimiento de cookies (CookieConsent.astro) integrado según normativas RGPD.



7. ASEGURAMIENTO DE LA CALIDAD (QA) Y DESPLIEGUE CONTINUO

7.1 Testing Automatizado y Pipeline CI/CD

Para garantizar la estabilidad del producto en un entorno de desarrollo rápido, la calidad se asegura mediante un pipeline de Integración Continua (CI).

- **Framework de Pruebas Unitarias:** Se utiliza **Vitest**, elegido por su extrema velocidad y su integración nativa con Vite (el motor de construcción de Astro).
- **Estrategia de Testing:** Antes de que cualquier cambio se fusione a producción, se ejecuta una batería de tests automáticos que verifican:
  - La integridad estructural de componentes críticos (Header, Footer, Reproductor).
  - La disponibilidad y respuesta correcta (HTTP 200) de los endpoints internos de la API.
  - La latencia y éxito de la conectividad con la base de datos MongoDB.
- **Despliegue Condicional:** La plataforma Vercel actúa como orquestador de CD. Si la batería de tests (Build Step) finaliza con éxito, se procede al despliegue atómico de la nueva versión. Si algún test falla, el proceso se aborta inmediatamente (exit 1), protegiendo el entorno de producción de código defectuoso.

7.2 Monitorización y Analítica Post-Despliegue

Una vez en producción, el ciclo de vida continúa mediante monitorización activa:

- **Analítica de Comportamiento: Google Analytics 4** provee datos sobre la interacción de los usuarios, tiempos de permanencia y episodios más escuchados.
- **Salud del Sitio: Google Search Console** se utiliza para supervisar la indexación, detectar errores 404 y asegurar el cumplimiento de los Core Web Vitals.

8. CONCLUSIONES Y VISIÓN DE FUTURO

8.1 Conclusiones del Proyecto

El desarrollo de Veredillas FM ha trascendido el objetivo académico inicial, resultando en un producto de software de nivel profesional. Ha permitido la consolidación práctica de conocimientos avanzados en arquitectura web *Full Stack*, diseño de esquemas de bases de datos NoSQL, seguridad en aplicaciones distribuidas e integración de sistemas heterogéneos. El proyecto es una demostración empírica de cómo la convergencia de tecnologías modernas (Astro, TypeScript) y herramientas de IA puede resolver problemas reales de comunicación en el entorno educativo con eficiencia y elegancia.

8.2 Hoja de Ruta (Roadmap)

El proyecto mantiene un estatus de desarrollo activo con hitos planificados hasta mayo de 2026. Las próximas líneas de investigación y desarrollo incluyen:

- **Implementación de transmisiones en directo:** Crear un servidor VPS para gestionar las transmisiones en en directo y poder mostrarlas finalmente en el sitio web
- **Interactividad en Tiempo Real:** Expansión del calendario con notificaciones Push y recordatorios.

9. AGRADECIMIENTOS

La realización de este proyecto, tanto en su vertiente técnica como en su contenido, no habría sido posible sin el apoyo y la colaboración de múltiples actores a quienes deseo expresar mi más sincero agradecimiento.

- **Al profesor Alejandro:** Por su mentoría técnica, por facilitarnos el acceso a la infraestructura del aula y al equipo de radio profesional, y fundamentalmente, por transmitirnos la motivación necesaria para transformar una idea de clase en un proyecto de ingeniería real.
- **Al IES Las Veredillas:** Por brindar el marco institucional y los recursos necesarios para fomentar la innovación educativa y tecnológica dentro del centro.
- **Al Equipo de Veredillas FM:** A todos mis compañeros y colaboradores (cuyos roles se detallan en la sección web correspondiente), por su compromiso, su profesionalidad en cada grabación y por haber desempeñado sus funciones específicas con excelencia, permitiendo que la parte técnica brille gracias a un contenido de calidad.
- **A los Invitados Especiales:** A cada profesor, alumno que nos ha visitado, por depositar su confianza en nuestra plataforma y enriquecer el proyecto con su tiempo y sus voces.

Este trabajo es el resultado de un esfuerzo colectivo donde la tecnología y la comunicación se han unido para crear algo único.

10. ENLACES DE INTERÉS

Página Web: https://veredillasfm.es
Repositorio de código: https://github.com/broslunas/veredillas-fm
Instagram: https://www.instagram.com/veredillasfm.es/