



UNL

Universidad
Nacional
de Loja

UNIVERSIDAD NACIONAL DE LOJA

DESARROLLO BASADO EN PLATAFORMAS

Evaluación Sumativa – Unidad 2: Aplicaciones Web Tecnologías de Lado del Cliente

DOCENTE

ING. Edison Coronel

INTEGRANTES

- Alexis Jara
- César López

UpTrack GUI

Este documento describe la arquitectura, estructura, estilos y dependencias del frontend de UpTrack (uptrack-gui).

1. Stack Tecnológico y Dependencias

El proyecto está construido utilizando modernas tecnologías de desarrollo web:

Framework Core: React 19.x

Lenguaje: TypeScript

Build Tool: Vite 7.x

Enrutamiento: React Router DOM 7.x

Estilos: Tailwind CSS 4.x

Visualización de Datos: Recharts (Gráficos y analíticas)

Iconografía: Lucide React

HTML5

2. Estructura del Proyecto

El código fuente se organiza dentro del directorio `src` siguiendo una estructura modular:

```
src/
  └── api/      # Lógica de comunicación con el backend (fetch wrappers)
  └── assets/    # Recursos estáticos (imágenes, fuentes)
  └── components/ # Componentes reutilizables
    └── layout/   # Componentes de estructura (MainLayout, Sidebar, Header,
MobileHeader)
      └── ui/      # Componentes primitivos (Button, Modal,FormField, etc.)
      └── data/     # Datos estáticos o mocks
      └── pages/    # Vistas principales de la aplicación
        ├── Dashboard.tsx # Vista principal con métricas
        ├── TargetDetail.tsx # Detalles específicos de un sistema
        ├── Systems.tsx    # Gestión de sistemas (CRUD)
        ├── Reports.tsx   # Generación de reportes
        ├── Login.tsx     # Autenticación
        ├── Register.tsx  # Registro de usuarios
        └── ...           # Otras páginas (Profile, Settings, etc.)
      └── App.tsx       # Configuración de rutas y layout principal
      └── index.css     # Estilos globales y configuración del tema
    └── main.tsx      # Punto de entrada de la aplicación
```

3. Sistema de Diseño y Estilos

El diseño visual utiliza Tailwind CSS con una configuración de tema personalizada definida en CSS variables (CSS-first configuration de Tailwind v4).

Paleta de Colores (Tema Oscuro)

El esquema de colores es principalmente oscuro, optimizado para dashboards y monitoreo prolongado. Las variables principales son:

Variable CSS Uso Color Hex
:--- :--- :---
`--color-background` Fondo principal `#101622`
`--color-background-surface` Superficies (Navbars) `#111722`
`--color-background-card` Tarjetas y contenedores `#181F2D`
`--color-background-input` Inputs de formularios `#192233`
`--color-text-main` Texto principal `#ffffff`
`--color-text-muted` Texto secundario `#92a4c9`

| `--color-primary` | Color de acento/acciones | `#135bec` |

- **Estados de Sistema**

Se utilizan colores semánticos para indicar el estado de los servicios monitoreados:

- Success (UP): #28a745
- Danger (DOWN): #dc3545
- Warning (DEGRADED): #ffc107

- **Accesibilidad (ARIA)**

La interfaz implementa atributos ARIA para garantizar la accesibilidad:

Roles semánticos (banner, navigation, main, contentinfo).

Etiquetas ARIA (aria-label, aria-labelledby, aria-describedby).

Estados dinámicos (aria-busy, aria-live, aria-expanded).

Navegación por teclado optimizada.

Diseño Responsivo

La interfaz es totalmente responsive ("Mobile First" en Tailwind), adaptándose desde dispositivos móviles hasta pantallas de escritorio grandes:

- Sidebar colapsable/oculto en móviles.
- Tablas que se transforman en tarjetas en vistas compactas.
- Gráficos que ajustan su tamaño al contenedor.

4. Componentes Clave

- **Layout Principal (`MainLayout`)**

Envuelve la aplicación autenticada, gestionando:

- Sidebar de navegación (Desktop).
- Header móvil con menú hamburguesa (Mobile).
- Área de contenido principal (`role="main"`).

- **Dashboard**

Ofrece una vista resumen con KPIs (Sistemas Totales, Online, Alertas) y un listado filtrable de tarjetas de estado de los sistemas.

- **Gráficos (Recharts)**

Implementados principalmente en la vista de reportes y detalles para visualizar:

- Historial de latencia.
- Tiempos de respuesta.
- Uptime.

5. Build y Despliegue

La aplicación se compila utilizando Vite, generando activos estáticos optimizados en la carpeta **dist**.

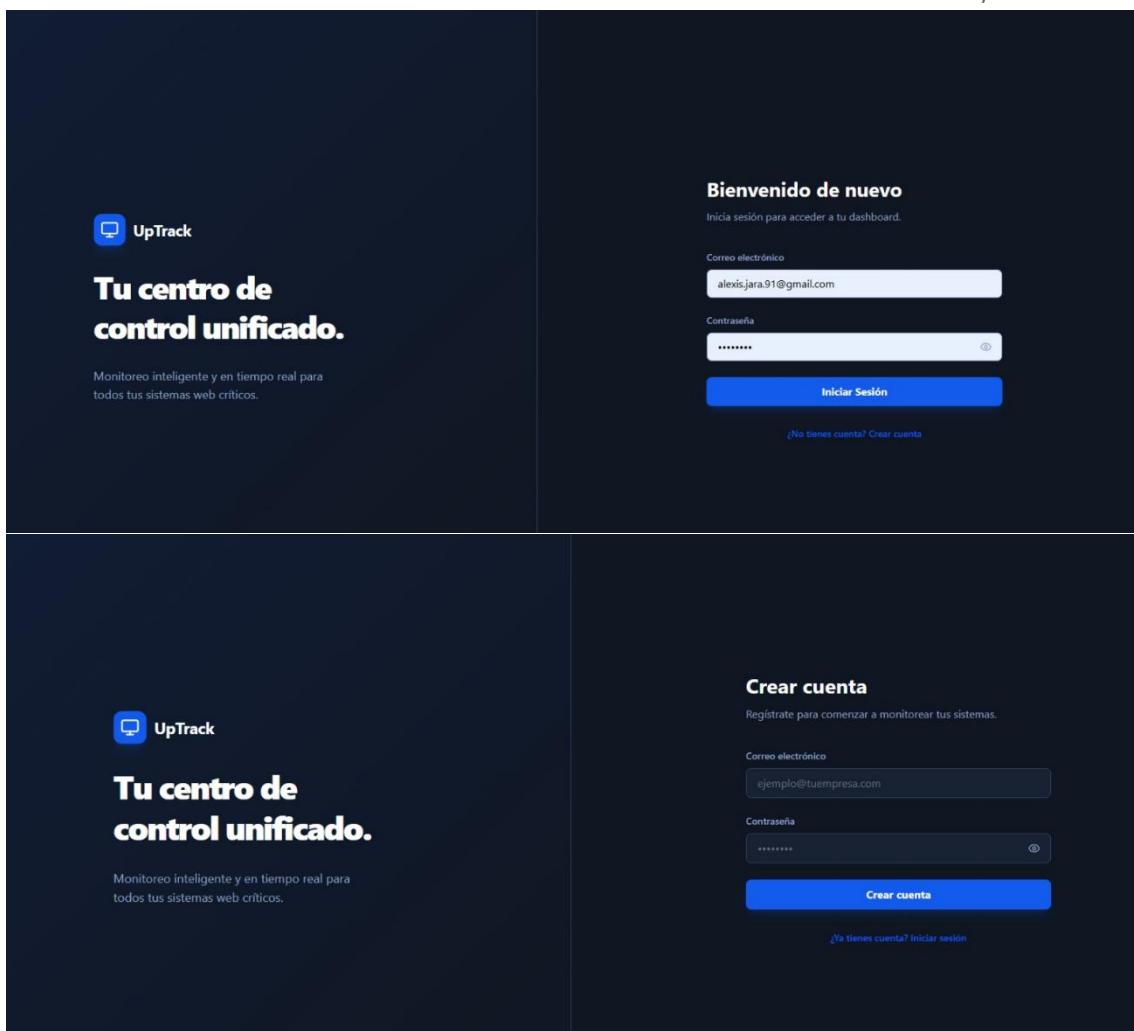
Scripts disponibles:

- pnpm dev: Servidor de desarrollo.
- pnpm build: Compilación para producción (TypeScript + Vite).
- pnpm preview: Vista previa del build de producción.

Manual de Ejecución y Diseño - UpTrack GUI

Este documento proporciona una guía detallada para la ejecución local de la aplicación, conexión con el backend, estándares de diseño y una descripción visual de la interfaz.

5. Capturas de la Aplicación en Funcionamiento



Login / Registro

Login: Diseño dividido (split-screen). A la izquierda, branding con fondo abstracto y logo de UpTrack. A la derecha, formulario de inicio de sesión limpio con campos para email y contraseña.

Registro: Modalidad similar, permitiendo crear nuevas cuentas de usuario.

The screenshot shows the UpTrack dashboard with a sidebar on the left containing links for Dashboard, Systems, Reports, Profile, and Settings. The main area is titled "Dashboard" and displays the following information:

- Total Sistemas:** 49
- Sistemas Online:** 32
- Con Alertas:** 17

Below this, there are six cards showing system status:

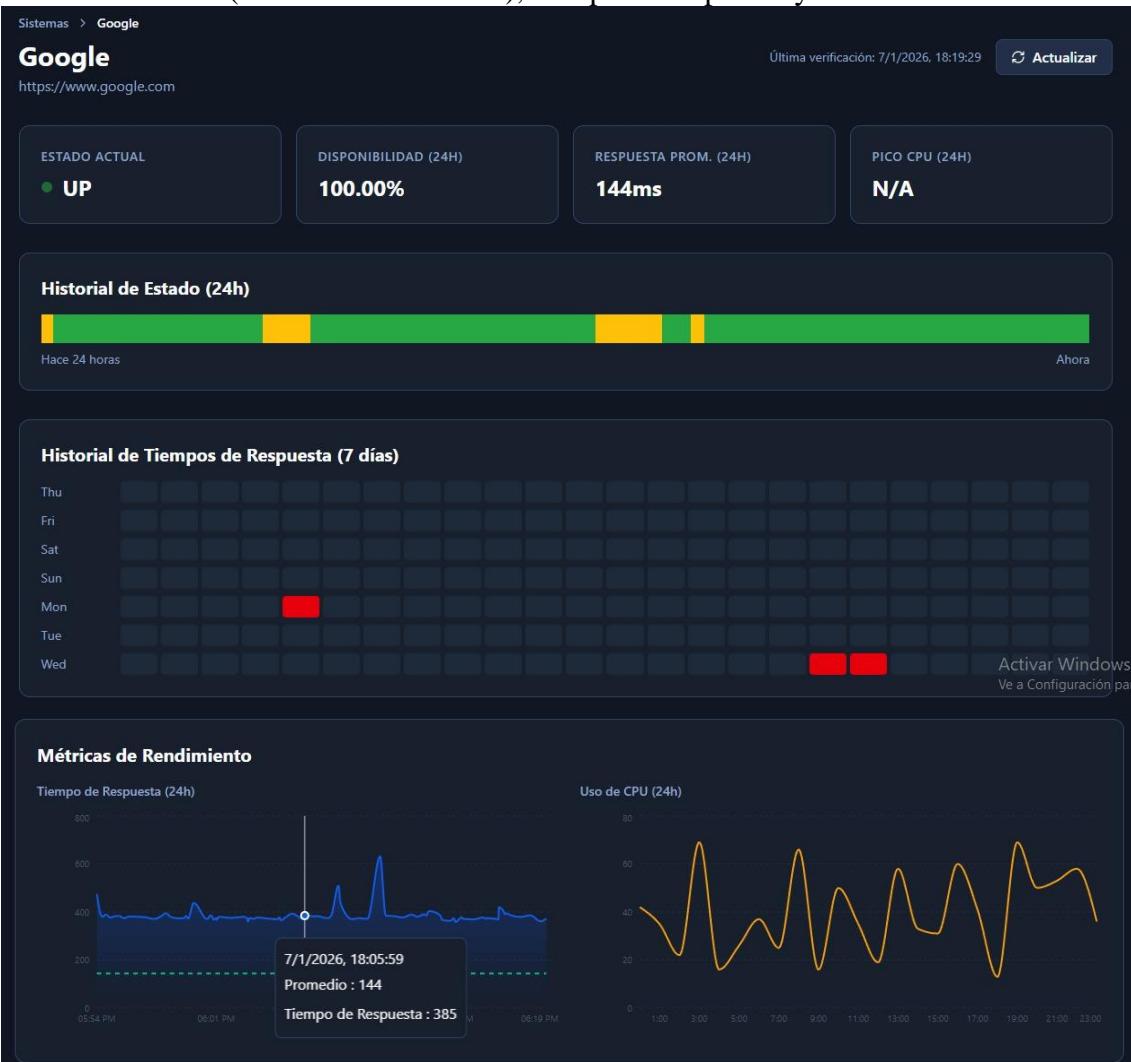
Nombre del Sistema	Estado	Última Verificación
Agify	DEGRADADO	7/1/2026, 18:17:39
Fake Down 1	FUERA DE LÍNEA	N/A
SWAPI	FUERA DE LÍNEA	7/1/2026, 18:17:59
Status 200	FUERA DE LÍNEA	7/1/2026, 18:17:45
HTTP Status 418 Te...	DEGRADADO	7/1/2026, 18:17:45
OpenAI	DEGRADADO	7/1/2026, 18:17:45

At the bottom right of the dashboard, there is a link: "Activar Windows" and "Ve a Configuración para activar Windows".

Dashboard Principal

KPIs Superiores: Tres tarjetas mostrando métricas clave: "Total Sistemas", "Sistemas Online" (verde), "Con Alertas" (rojo).

Lista de Sistemas: Grid de tarjetas para cada sistema monitoreado, mostrando indicadores de estado (color del borde/texto), tiempo de respuesta y última verificación.



Detalles del Sistema (Target Detail)

Barra de Estado Temporal: Visualización tipo "timeline" segmentada por colores (verde/rojo) mostrando el historial de disponibilidad.

Gráficos: Gráficos de línea (Recharts) mostrando la latencia (ms) en las últimas 24 horas.

Heatmap: Mapa de calor semanal (si está disponible) mostrando patrones de disponibilidad.

FACULTAD DE LA ENERGÍA, LAS INDUSTRIAS Y
LOS RECURSOS NATURALES NO RENOVABLES

Carrera de Computación

Select Target Start Date End Date

JSONPlaceholder API (<https://jsonplaceholder.typicode.com>) 05/12/2025 07/01/2026

Generate Report

Save as PDF

SERVICE LEVEL REPORT

UpTrack AI Monitoring System

Generated on: 7/1/2026
Ref: 019AA9D7

1. EXECUTIVE SUMMARY

System Name	JSONPlaceholder API	Target URL	https://jsonplaceholder.typicode.com/posts/1
Reporting Period	4/12/2025 to 6/1/2026		
Overall Uptime	89.89%	Avg Response Time	80ms
Total Checks	100	Total Incidents	18

2. DAILY PERFORMANCE METRICS

Response Time (ms) Uptime %



3. INCIDENT HISTORY

Timestamp	Event Type	Description
7/1/2026, 18:16:09	UP	System recovered and operational.
7/1/2026, 18:13:53	UP	System recovered and operational.
7/1/2026, 18:13:37	DEGRADED	System unreachable or returned error.
7/1/2026, 18:12:36	DEGRADED	System unreachable or returned error.
5/1/2026, 3:39:26	UP	System recovered and operational.
5/1/2026, 3:38:31	DEGRADED	System unreachable or returned error.
16/12/2025, 23:45:53	UP	System recovered and operational.
16/12/2025, 23:38:44	DEGRADED	System unreachable or returned error.
16/12/2025, 23:37:55	UP	System recovered and operational.
16/12/2025, 23:33:40	DEGRADED	System unreachable or returned error.
16/12/2025, 22:56:10	UP	System recovered and operational.
16/12/2025, 22:55:25	DEGRADED	System unreachable or returned error.
16/12/2025, 22:49:24	UP	System recovered and operational.
16/12/2025, 22:47:25	DEGRADED	System unreachable or returned error.
16/12/2025, 22:24:44	UP	System recovered and operational.

* Showing first 15 incidents only.

CONFIDENTIAL - Generated by UpTrack AI

Reportes

Formulario para seleccionar un sistema y un rango de fechas.

Botón "Generate Report" que despliega estadísticas detalladas y una vista imprimible de los datos.

Agregar Nuevo Sistema

Información del Sistema

Nombre del Sistema
Ej. API de Producción

Alias descriptivo para identificación.

URL a Monitorear
https://ejemplo.com

URL completa del endpoint a verificar.

Tipo
WEB API

Configuración de Monitoreo

Intervalo
1 min 3 min 5 min

Red Interna
Habilitar monitoreo interno

Cancelar Agregar Sistema

Agregar Sistema

Formulario para agregar un nuevo sistema

Configuración

Canales de Notificación

Configura los medios por los cuales deseas recibir alertas cuando tus sistemas cambien de estado.

Telegram
Recibe alertas instantáneas a través de nuestro bot

Conectarse

Vincular un medio de comunicación para enviar alertas

6. Instrucciones de Ejecución Local

• Prerrequisitos

Node.js: Versión 18 o superior.

pnpm: Gestor de paquetes recomendado (o npm).

Go: (Para el backend) Versión 1.21+.

Docker: (Opcional, si se usa docker-compose para la base de datos).

• Configuración del Entorno (Frontend)

El archivo .env en la raíz de uptrack-gui debe apuntar a la dirección donde corre tu backend.

VITE_API_BASE_URL=http://localhost:8080

- **Ejecución en Windows**
- **Levantar el Backend:**

Abre una terminal (PowerShell o CMD) en la carpeta uptrack/backend:

- **Instalar dependencias**

go mod download

- **Ejecutar el servidor**

go run main.go

- **El servidor debería iniciar en el puerto 8080**

2. Levantar el Frontend:

Abre una nueva terminal en uptrack/uptrack-gui:

- **Instalar dependencias**

pnpm install

- **Iniciar servidor de desarrollo**

pnpm dev

La aplicación estará disponible en http://localhost:5173.

- **Ejecución en Linux**

- **Levantar el Backend:**

Terminal en uptrack/backend:

Dependencias y ejecución

go mod download

go run main.go

- **Levantar el Frontend:**

Terminal en uptrack/uptrack-gui:

pnpm install

pnpm dev

7. Conexión Backend-Frontend

CORS: El backend (backend/config/server.go) está configurado para permitir peticiones desde cualquier origen (Access-Control-Allow-Origin: *) durante el desarrollo, por lo que no deberías tener problemas de bloqueo CORS.

Endpoint Base: Todas las peticiones del frontend se prefijan con la URL definida en VITE_API_BASE_URL.

Autenticación: El sistema usa JWT. El frontend almacena el token en localStorage tras el login y lo inyecta automáticamente en el header Authorization: Bearer <token> mediante la utilidad fetchWithAuth.

- **Estándares de Diseño Aplicados**

Estilos y CSS

Framework: Tailwind CSS v4.

Metodología: Utility-first CSS. No se usan archivos CSS/SCSS separados por componente, sino clases utilitarias directamente en el JSX.

Tema Personalizado: Se utilizan variables CSS nativas (--color-background, --color-primary) definidas en @theme dentro de src/index.css. Esto facilita el cambio de temas y mantiene la consistencia.

Diseño Oscuro (Dark Mode): La aplicación es nativamente oscura ("Dark Interface") para reducir la fatiga visual en tareas de monitoreo.

Accesibilidad (A11y): Se han implementado atributos ARIA (role, aria-label, aria-live) en componentes interactivos para soporte de lectores de pantalla.

Estructura de Componentes

Atomic Design (Simplificado):

components/ui: Átomos y moléculas básicas (Botones, Inputs, Modales). Son componentes puros, sin lógica de negocio, solo presentación.

components/layout: Organismos estructurales (Sidebar, Header).

pages: Plantillas/Páginas completas que conectan los componentes con la lógica de datos y el estado.

Hooks: La lógica de estado compleja o reutilizable se extrae (aunque actualmente mucha lógica reside en las páginas para simplicidad).

Principio de Responsabilidad Única: Los componentes UI (Button, FormField) son genéricos y reusables. Las páginas (Dashboard, Systems) manejan la llamada a la API y el estado de la vista.

Nomenclatura

Archivos: PascalCase para componentes (TargetDetail.tsx) y camelCase para utilidades (fetch.ts).

Clases CSS: Clases estándar de Tailwind (flex, p-4, text-white).

Variables de Entorno: Prefijo VITE_ obligatorio para exposición al cliente (ej. VITE_API_BASE_URL).

Ramas:

⚡ develop

⚡ main

feat(gui): enhance reports logic/printing and update docs

Conclusiones:

- La estructura del proyecto, que separa componentes de presentación (components/ui) de la lógica de negocio (pages y api), facilita la escalabilidad. Esto permite que el equipo de desarrollo pueda agregar nuevas métricas o vistas sin afectar la estabilidad del código base existente ni generar deuda técnica.
- La decisión de diseño "Dark Mode First", combinada con el uso de colores semánticos (verde/rojo) y gráficos interactivos (Recharts), cumple un objetivo funcional claro: reducir la fatiga visual del operador y permitir la identificación inmediata de fallos en el sistema, lo cual es vital en herramientas de monitoreo continuo.