DevFlow AI

15 herramientas para developers

Trabajo Final de Master — Desarrollo con IA (BIG School)

Alberto Guinda Sevilla · Febrero 2026

El Problema

Los developers usan **10+ herramientas fragmentadas** para tareas cotidianas

Desarrollo con IA

- Prompts sin evaluar calidad
- Costes de API opacos
- Tokenizacion invisible
- Context windows mal gestionados

Tareas diarias

- Formatear JSON, Base64, UUID...
- Ordenar clases Tailwind
- Construir expresiones cron
- Generar commits convencionales

Resultado: Friccion, costes ocultos, perdida de productividad

La Solucion

Una plataforma unificada, gratuita y open-source

Local-first
Todo en el navegador. Sin login. Sin API keys.

IA opcional
4 proveedores con fallback. Pollinations siempre gratis.

Enterprise-grade
CSP, HSTS, SAST, CodeQL, rate limiting, Sentry.

15 herramientas · 1416 tests · Lighthouse 100/100/100/100

Las 15 Herramientas

JSON Formatter — Format, diff, TS interfaces	**Variable Name Wizard** — 8 convenciones	**Regex Humanizer** — Explicacion + tester
Code Review — Smells, complejidad, seguridad	**Cost Calculator** — 10+ modelos, Claude 4.x	**Base64** — URL-safe, data URL, Unicode
UUID Generator — v1, v4, v7, bulk 1000	**DTO-Matic** — JSON → TS + Zod	**Git Commit** — Convencional + emojis
Cron Builder — Visual + calendario	**Tailwind Sorter** — Ordena + diff	**Prompt Analyzer** — Score + inyeccion
Token Visualizer — BPE + costes	**Context Manager** — Chunks + export	**HTTP Status** — 61 codigos + guias

Cada herramienta sigue el patron 5-capas: types → lib/application → hooks → page → tests

Arquitectura: Clean Architecture

Patron 5-capas por herramienta

```
types/<tool>.ts          → Interfaces
lib/application/<tool>.ts → Logica pura
hooks/use-<tool>.ts      → Estado + localStorage
app/.../page.tsx         → UI
tests/.../<tool>.test.ts → Tests
```

Flujo: Presentation → Application → Domain

0 violaciones de capas en las 15 herramientas

AI Backend (opcional)

```
hooks/use-ai-<feature>.ts → SWRMutation
app/api/ai/<endpoint>/    → Route Handler
```

Provider chain:

1. BYOK (tu API key)
2. Gemini 2.0 Flash
3. Groq
4. OpenRouter
5. Pollinations (gratis)

Rate limiter: 10 RPM / 50 RPM BYOK

Stack Tecnologico

Capa	Tecnologia	Capa	Tecnologia
Framework	Next.js 16 (App Router)	Testing	Vitest + Playwright
UI	React 19 (Compiler)	a11y	axe-core WCAG AAA
Lenguaje	TypeScript 5 (max strict)	CI/CD	GitHub Actions (10 jobs)
Estilos	Tailwind CSS v4	Error tracking	Sentry
Componentes	HeroUI v3 beta	Hosting	Vercel Edge
Animaciones	GSAP + Framer Motion	State	Zustand + Context

18 deps produccion · TypeScript maximum strict (15+ flags) · Zero `any`

Server Components & Performance

Homepage (Server Component)

```
page.tsx (async RSC)
├── Hero          → Server (0 JS)
├── Stats         → Server + GSAP island
│   └── GitHub    → Server fetch (ISR 1h)
├── Features      → Client (GSAP stagger)
├── Why DevFlow   → Server (0 JS)
└── Footer        → Server (0 JS)
```

Resultado: Hero y Footer = 0 bytes JS

Lighthouse Desktop

Metrica	Score
Performance	100
Accessibility	100
Best Practices	100
SEO	100

React Compiler + View Transitions API + ISR + optimizePackageImports

Testing: Estrategia 100/80/0

Piramide estrategica

Tier	Target	Path
CORE	80-100%	`lib/application/*.ts`
IMPORTANT	80%	`components/shared/`
INFRA	0%	`types/`, `config/`

Per-file enforcement: cada archivo CORE debe cumplir individualmente

CI bloquea merge si cualquier archivo baja del umbral

Metricas actuales

```
Tests:    1416 passing (45 files)
E2E:      20 Playwright specs
a11y:     19 paginas (axe-core WCAG AA)
Duration: ~50s
```

Top herramientas:

- UUID Generator: 89 tests
- Regex Humanizer: 83 tests
- Cron Builder: 75 tests
- Variable Name Wizard: 74 tests
- Git Commit Generator: 71 tests

Seguridad

HTTP Headers

- **CSP** estricto (sin `unsafe-eval`)
- **HSTS** 2 anos + preload
- **X-Frame-Options** DENY
- **Permissions-Policy** restrictivo
- **Prototype pollution** protegido

Application

- Rate limiting IP-based
- Zod validation en endpoints
- Anti-injection en system prompts
- Zero `eval()` / `innerHTML`

CI/CD Security (10 jobs)

- **npm audit** `--audit-level=high`
- **lockfile-lint** validacion registro
- **CycloneDX SBOM** en cada build
- **CodeQL** SAST JS/TS
- **Semgrep** OWASP Top 10
- **eslint-plugin-security**
- **StepSecurity harden-runner**
- **SHA-pinned** todas las Actions
- **Dependency review** en PRs
- **Renovate** auto-updates

Internacionalizacion

Sistema custom (sin i18next)

- ~1595 claves por idioma (EN/ES)
- Hook `useTranslation()` con interpolacion
- Cambio instantaneo via Zustand
- Paridad perfecta verificada

Cobertura

- 15 tool pages completamente traducidas
- Landing page, settings, docs, history
- ARIA labels, placeholders, toasts
- Error messages, empty states

Ejemplo

```
// locales/en.json
{ "costCalc.cachedPrices": "Using cached prices" }

// locales/es.json
{ "costCalc.cachedPrices": "Usando precios en cache" }

// En el componente
const { t } = useTranslation();
<Chip>{t("costCalc.cachedPrices")}</Chip>
```

0 strings hardcodeadas en produccion

Features Clave

Command Palette (`Cmd+K`)

- Busqueda fuzzy de 15 tools + 5 acciones
- Navegacion por teclado completa
- Zero dependencias extra

Smart Suggestions

- 10 detectores de tipo de dato
- 15 reglas de recomendacion
- Data passing entre herramientas

PWA

- Manifest + Service Worker
- Instalable como app standalone
- Offline-capable

Export/Import Settings

- Backup de todas las preferencias
- Validacion Zod al importar
- Filtro `devflow-*` keys

MagicInput

- Deteccion automatica de tipo
- Redirige a la herramienta correcta
- JSON, Cron, Regex, Base64...

History + Favorites

- `useToolHistory<T>` generico
- Max 50 items por herramienta
- Timestamps relativos (EN/ES)

Resultados

15 Herramientas	1416 Tests passing	35 Rutas
~1595 Claves i18n (x2)	10 Jobs CI/CD	100 Lighthouse (x4)

4 proveedores IA · WCAG AAA · PWA instalable · Zero any

Conclusiones

Logros principales

1. **Producto funcional** — 15 tools en produccion
2. **Clean Architecture** — 5-capas sin excepciones
3. **Performance maxima** — Lighthouse 100x4
4. **Testing robusto** — 1416 + 20 E2E + a11y
5. **Seguridad enterprise** — SAST, CSP, rate limiting
6. **UX avanzada** — PWA, Cmd+K, MagicInput
7. **IA opcional** — 4 proveedores, BYOK, fallback
8. **i18n completo** — ~1595 claves EN/ES

Aprendizajes clave

- **RSC** reduce drasticamente el JS al cliente
- **Clean Architecture** vale la pena para testing
- **TypeScript strict** previene categorias de bugs
- **100/80/0** es mas sostenible que 100% global
- **Claude Code** como pair programmer acelera significativamente
- **Local-first** es ventaja competitiva real

Trabajo futuro

- Cloud sync (Supabase)
- Team collaboration
- Browser extension
- Mobile app (React Native)

Demo en vivo

devflowai.vercel.app

Prompt Analyzer | Code Review | Cost Calculator

Token Visualizer | JSON Formatter | Context Manager

Repo: github.com/albertoguinda/devflow-ai

Preguntas

Recurso	Enlace
Demo	devflowai.vercel.app
Repositorio	github.com/albertoguinda/devflow-ai
TFM Documento	docs/TFM.md