

Contexto y Problema

¿Qué es Help Deploy?

Es una herramienta de software diseñada para cerrar la brecha entre **Desarrollo** y **Operaciones**. Centraliza los comandos de infraestructura en una sola interfaz lógica.

La Problemática Actual

- **Complejidad de Sintaxis:**
Terraform (HCL) requiere memorizar estructuras complejas propensas a errores tipográficos.
- **Errores Manuales:**
La configuración manual de archivos aumenta el riesgo de fallos en el despliegue.
- **Ineficiencia:**
Cambiar constantemente entre documentación y código reduce la productividad del equipo.



Los errores manuales representan el mayor cuello de botella.

Objetivos y Tecnología

Objetivo General

Desarrollar un sistema que automatice tareas repetitivas de infraestructura mediante flujos de trabajo predefinidos.

Stack Tecnológico

```
const stack = {
  core: "TypeScript",
  ui: "VS Code Webviews",
  infra: "Terraform CLI",
  vcs: "Git CLI"
};
```

Arquitectura

El sistema funciona interceptando la intención del usuario y mapeándola a **bloques de código validados**. No "inventa" código, sino que ensambla piezas de infraestructura probadas y seguras.

Resultados e Impacto

Eficiencia Operativa

Comparativa de tiempo para desplegar un entorno base (EC2 + VPC):



Resultado: Reducción del 78% en tiempo de configuración gracias a la automatización de plantillas.

Cronograma 2025

- AGO:** Planificación
- SEP:** Diseño UI/UX
- OCT:** Desarrollo Lógica
- NOV:** Pruebas QA

Solución Integrada

Help Deploy es una extensión para VS Code que automatiza la creación de infraestructura mediante **lógica programática y plantillas estandarizadas**.

⚙️ Funcionalidades

✓ Generación de Scripts:

Uso de plantillas predefinidas para crear archivos `main.tf` sin errores manuales.

✓ Validación Estática:

Análisis de sintaxis en tiempo real para asegurar código limpio.

✓ Asistente Git:

Interfaz guiada para comandos `git init`, `add`, `commit` y `push`.

> Comandos Clave

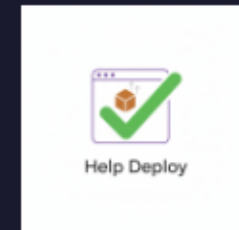
```
> Crear main.tf
> Crear README.md (Plantilla)
> Desplegar proyecto con Terraform
> Clonar repositorio (Git)
> Crear rama (Git)
> Pushear cambios (Git)
> Guardado temporal (stash) (Git)
> Recuperar stash (Git)
> Limpiar historial
```

Conclusiones

"La automatización lógica elimina la fricción técnica, permitiendo que el desarrollador se enfoque en la arquitectura y no en la sintaxis."

Análisis de Factibilidad

- **Técnica:** Desarrollo en TypeScript nativo. Integración directa con APIs de VS Code.
- **Económica:** Cero costo de licencias (Open Source).
- **Operativa:** Curva de aprendizaje mínima al integrarse en el editor diario.



Asistente para Automatización de
Terraform y Git

// v1.0.9 Stable

Curso: Calidad y Pruebas de Software

Docente: Ing. Patrick Jose Cuadros Quiroga

Integrantes:

- Augusto Rivera Muñoz (2022073505)
- Jefferson Rosas Chambilla (2021072618)