1 DESCRICIÓN DO ANTEPROXECTO

(Adaptar a lonxitude do documento segundo sexa preciso)

TÍTULO: DevOps Hub: Plataforma de gestión para equipos de desarrollo

Introducción / Motivación

DevOps Hub es una aplicación web desarrollada para cubrir una necesidad creciente en el contexto actual del desarrollo de software: ofrecer a equipos reducidos y desarrolladores independientes una solución unificada, ligera y gratuita para gestionar todo el ciclo de vida de sus proyectos.

Centraliza funcionalidades clave como la gestión de proyectos, control de versiones, integración con GitHub y medidas básicas de seguridad, todo desde una única interfaz web sencilla y eficaz.

Este proyecto surge como alternativa a herramientas empresariales como Jira o Azure DevOps, que resultan complejas, sobredimensionadas o de pago para quienes trabajan en solitario o en grupos pequeños. DevOps Hub permite a estos profesionales centrarse en lo esencial —el código y la organización técnica del proyecto— sin depender de plataformas costosas o complicadas, aportando así una solución eficiente y accesible adaptada a su realidad.

Propuesta de valor y diferenciación:

- Gratuita y open-source
- Ligera, sin instalación compleja
- Incluye lo esencial para desarrolladores
- Interfaz intuitiva y accesible
- Pensada para freelance, estudiantes y pequeños estudios

Capacidades del promotor:

Dispongo de formación técnica en desarrollo de aplicaciones web, con experiencia en .NET, JavaScript, HTML, CSS, Ext JS...La motivación fue tratar de crear una solución informática encarada a la mejora de la productividad, que pueda facilitar el trabajo de los desarrolladores.

Justificación de la necesidad en entorno productivo

Muchas herramientas como Azure DevOps o Jira resultan excesivas para proyectos individuales o pequeños equipos. DevOps Hub ofrece una alternativa gratuita y liviana, con lo esencial para el seguimiento de proyectos y versiones, permitiendo a desarrolladores centrarse en el código y no en la infraestructura.

Análisis PEST (macroentorno):

- **Político:** Apoyo institucional a la digitalización y el emprendimiento individual.
- **Económico:** Crece el número de desarrolladores freelance o autónomos que necesitan herramientas económicas o gratuitas.
- **Social:** Aumento del trabajo remoto y los equipos distribuidos.
- **Tecnológico:** Ampliación del ecosistema open-source y uso generalizado de APIs REST.

Análisis del microentorno:

- **Competencia:** Existen soluciones como GitHub Projects, Trello, Jira, Azure DevOps… Muchas son complejas o de pago.
- **Clientes potenciales:** Freelancers, estudiantes, pequeños equipos de desarrollo, docentes que gestionan prácticas de programación.

Segmentación de mercado:

- Edad: 20-45 años
- Perfil técnico con experiencia digital básica o media
- Ubicación: principalmente España, pero aplicable a cualquier entorno hispanohablante
- Público: freelancers, microempresas tecnológicas, estudiantes de formación profesional o universitarios

Filosofía empresarial:

- **Visión:** Convertir DevOps Hub en un referente accesible y gratuito para la gestión ágil de proyectos técnicos a pequeña escala.
- **Valores:** Simplicidad, accesibilidad, eficiencia, comunidad.
- **Responsabilidad Social Corporativa (RSC):**

    - Promoción del software libre y del acceso igualitario a herramientas tecnológicas.
    - Uso responsable de recursos: aplicación ligera y eficiente.
    - Apoyo al emprendimiento individual, fomentando la autonomía profesional.

Obxetivos

- Implementar un sistema básico de autenticación (registro e inicio de sesión)
- Desarrollar un panel de control (dashboard) para visualizar y crear proyectos
- Establecer una integración básica con la API de GitHub para visualizar repositorios existentes
- Aplicar medidas de seguridad básicas (validación de formularios y consultas parametrizadas.

Historias de usuario

HU1: Como usuario quiero registrarme e iniciar sesión en la plataforma para acceder a mis proyectos.

- Autenticación básica con sesiones en el backend; formulario de login en el frontend.

HU2: Como usuario quiero crear nuevos proyectos desde la plataforma.

- Modelo de datos para proyectos y API REST para altas.

HU3: Como usuario quiero visualizar la lista de mis proyectos en el dashboard.

- Consulta de la base de datos y renderizado en el frontend.

HU4: Como usuario quiero conectar mi cuenta de GitHub para ver mis repositorios.

- Uso de la API pública de GitHub para listar repositorios del usuario autenticado (modo lectura, sin login OAuth).

HU5: Como usuario quiero que mis datos estén protegidos frente a accesos no autorizados.

- Validación en frontend y backend, consultas SQL parametrizadas.

Descripción técnica

La arquitectura se mantiene en **tres capas** (presentación, lógica de negocio y acceso a datos) para maximizar la claridad del código y facilitar el mantenimiento, pero simplificando las tecnologías utilizadas.

Tecnologías:

- **HTML, CSS, JavaScript y Bootstrap (Capa de presentación):** Estas tecnologías son estándar en el desarrollo web, con amplia documentación y comunidad. Bootstrap acelera la creación de interfaces responsive mediante componentes predefinidos, reduciendo tiempo de maquetación y garantizando compatibilidad móvil.
- **Node.js y Express (Capa de negocio):** Node.js ofrece un entorno asíncrono para gestionar las peticiones del cliente. Express es un framework minimalista pero extensible, ideal para construir APIs REST de forma rápida
- **SQLite (Capa de datos):** Base de datos ligera que no requiere instalación ni configuración de servidores adicionales, facilitando el desarrollo local.
- **Git y GitHub (Control de versiones e integración):** Git para el control de versiones local y GitHub para almacenar el código y para la integración con sus APIs, que permitirá a los usuarios visualizar sus repositorios.
- **Medidas de seguridad:** Implementación básica de validación de formularios en cliente y servidor, y uso de consultas parametrizadas para prevenir inyección SQL

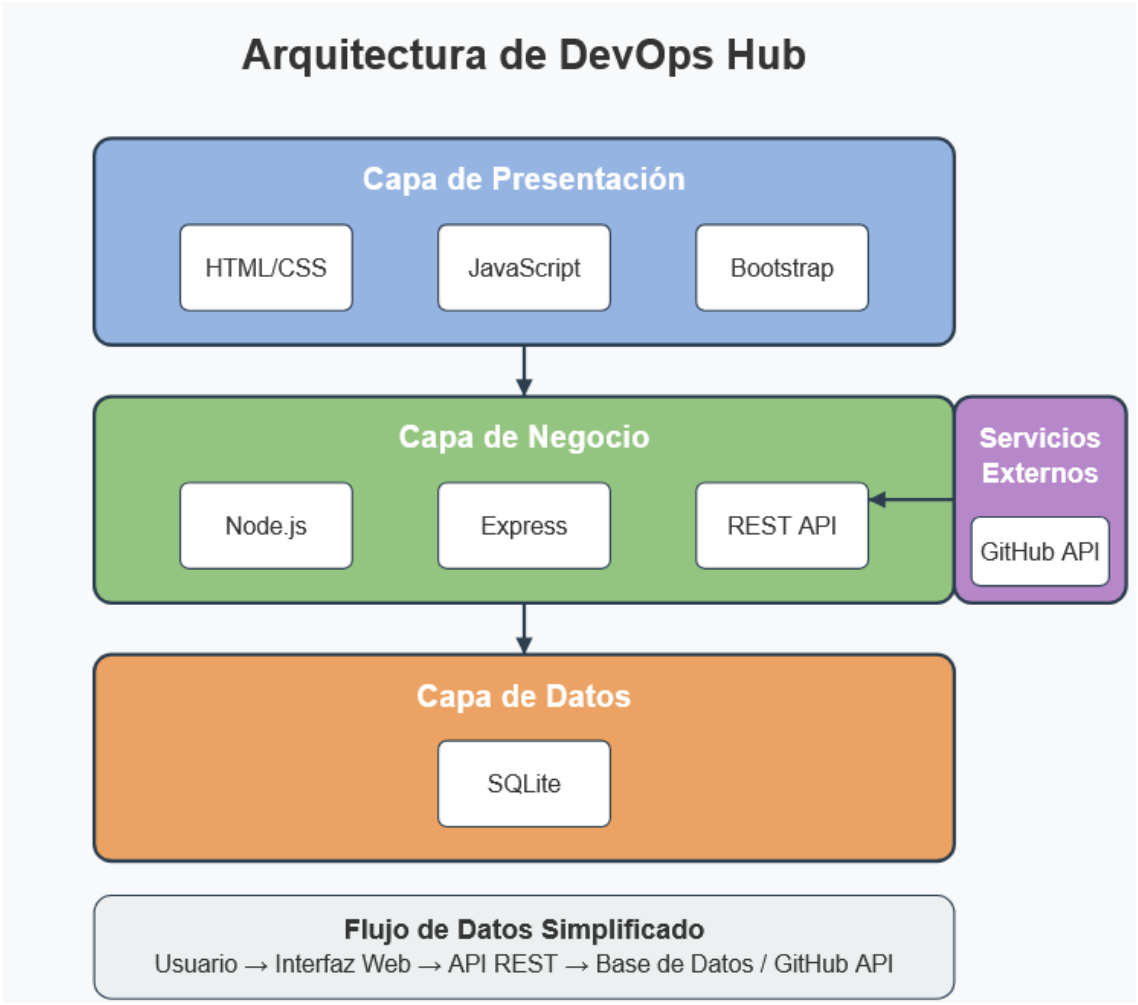


Figura 1: Arquitectura de 3 capas que implementará el sistema.

Proceso de desenvolvemento a empregar no proxecto

El proyecto seguirá el **Proceso Unificado de Desarrollo de Software**, que es iterativo, incremental, dirigido por casos de uso y centrado en la arquitectura. Consta de las siguientes fases:

Fase de Inicio

- Definición del MVP
- Análisis de requisitos
- Identificación de riesgos

Fase de Elaboración

- Diseño de la arquitectura
- Prueba de concepto (autenticación + conexión GitHub)
- Estructura de carpetas y entorno de desarrollo

Fase de Construcción

- Desarrollo iterativo de funcionalidades
- Refactorizaciones parciales
- Validación manual

Fase de Transición

- Pruebas finales completas en entorno local
- Preparación de documentación

Planificación do traballo e estimación temporal

Datos generales

-Dedicación semanal prevista: 30 horas/semana

-Dedicación diaria prevista: 5 horas/día (lunes a sábado)

-Fecha de inicio: 5 mayo 2025

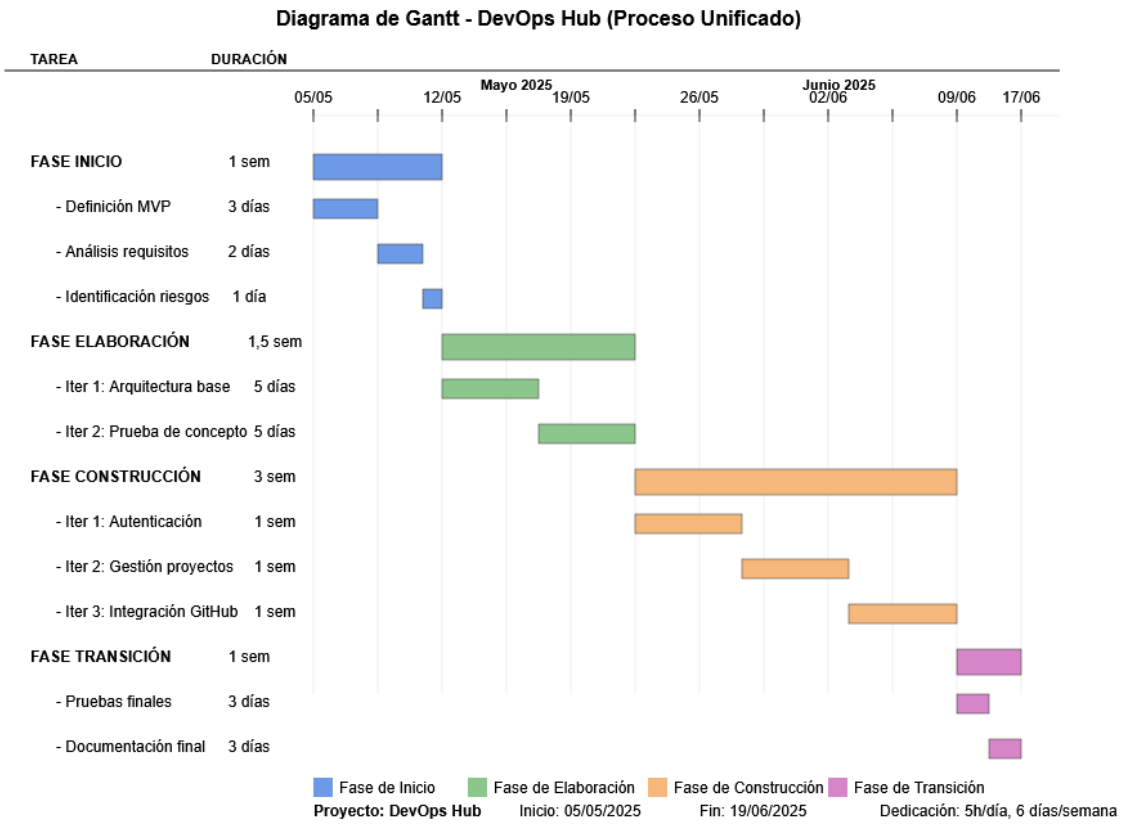
-Fecha de finalización: 19 de Junio 2025

Planificación temporal del proyecto:

Fase	Duración	Fechas	Iteraciones	Horas estimadas
Inicio	**1 semana**	**5 Mayo - 11 Mayo**	**1**	**30 h**
Definición del producto mínimo viable				15h
Análisis inicial de requisitos				10h
Identificación de riesgos				5h
Elaboración	**1,5 semanas**	**12 Mayo - 21 Mayo**	**2**	**45 h**
Iteración 1: Arquitectura base		12 Mayo - 16 Mayo		20h
Iteración 2: Prueba de		17 Mayo - 21 Mayo		25h

concepto				
Construcción	**3 semanas**	**22 Mayo - 11 Junio**	**3**	**90 h**
Iteración 1: Autenticación + Dashboard	1.5 semanas	22 Mayo - 28 Mayo		30h
Iteración 2: Gestión de proyectos	0.5 semanas	29 May - 4 Jun		30h
Iteración 3: Integración con GitHub		5 Jun - 11 Jun		30h
Transición	**1 semana**	12 Jun - 19 Jun		30h
Pruebas finales				15h
Documentación final				15h
Total	**6.5 semanas**	**5 may – 19 jun**		**195 h**

Diagrama de Gantt para mostrar la planificación temporal de forma más visual:



Arquitectura técnica

El sistema se basa en una arquitectura de tres capas:

- **Capa de Presentación**: HTML5, CSS3, JavaScript y Bootstrap
- **Capa de Negocio**: Node.js + Express
- **Capa de Datos**: SQLite

  Servicios externos: API REST de GitHub para visualización de repositorios públicos.

Recursos materiales y humanos

Recursos humanos:

- **Promotor único:** desarrollador full stack autónomo.
- **Forma jurídica:** empresario individual (alta como autónomo).
- **Tarifa plana de Seguridad Social:** 80 €/mes durante el

primer año.

- **Valoración de la hora de trabajo:** 12 €/hora
- **Horas estimadas del proyecto:** 195 horas
- **Coste laboral aproximado:**

    - Remuneración: 195 h × 12 €/h = **2.340 €**
    - Seguridad Social (6,5 semanas): aprox. **80 €**

Recurso	Descripción	Coste estimado
Ordenador	Equipo personal (i5, 8 GB RAM)	0 €
Software	VS Code, Node.js, SQLite, Git (gratis)	0 €
Internet	Consumo doméstico durante el proyecto	30 €
Electricidad	Uso aproximado por jornada laboral	20 €

Orzamento estimado

Costes totales aproximados del proyecto:

- Coste humano: 2.340 €
- Seguridad Social: 80 €
- Gastos de funcionamiento (luz + internet): 50 €
- Total estimado del proyecto: 2.470 €

Marketing y promoción:

- Publicación del repositorio en GitHub
- Promoción en LinkedIn, Twitter y foros técnicos como Dev.to y Reddit
- Página de presentación básica en HTML
- Difusión a través de canales gratuitos (sin inversión inicial)

Previsión de ingresos:
Aunque inicialmente es gratuita, DevOps Hub podría monetizarse en el futuro mediante:

- Funcionalidades premium
- Alojamiento en la nube
- Publicidad no invasiva
- Servicios adicionales como soporte o formación

Financiación:

- Capital propio inicial
- Posibilidad de solicitar subvenciones públicas como el Kit Digital
- Alternativamente, capitalización del paro como pago único para iniciar la actividad como autónomo

Tecnologías

Tecnología	Versión	Coste	Fase del proyecto	Uso previsto
HTML, CSS, JavaScript, Bootstrap	HTML5, CSS3, JS ES6, Bootstrap v5.x	Gratuito	Elaboración y Construcción	Diseño de la interfaz de usuario y desarrollo del dashboard responsivo.
Node.js y Express	Node.js v20.x, Express v3.x	Gratuito	Elaboración y Construcción	Backend y API REST: autenticación, gestión de usuarios y proyectos.
SQLite	SQLite v3.x	Gratuito	Construcción	Base de datos local para usuarios y proyectos, sin necesidad de servidor adicional.
Git y GitHub	Git v2.40+, GitHub (plan gratuito)	Gratuito	Todas las fases	Control de versiones, alojamiento del código, integración con la API.
API de GitHub (REST v3)	v3	Gratuito	Construcción y Transición	Visualización de repositorios del usuario autenticado (modo lectura).
Visual Studio Code	Última versión estable (2025)	Gratuito	Todas las fases	Entorno de desarrollo integrado con soporte para

				JavaScript y Node.js.
DB Browser for SQLite	v3.12.x	Gratuito	Construcción y pruebas	Gestión y visualización de la base de datos SQLite.
Herramientas de documentación	LibreOffice Writer / draw.io	Gratuito	Todas las fases	Documentación técnica y generación de diagramas de arquitectura y modelo ER.

Despliegue

Modalidad: Entorno local (Visual Studio Code + terminal integrada)

Pasos:

1. Clonar el repositorio desde GitHub
2. Ejecutar npm install
3. Iniciar el backend con npm run dev
4. Abrir public/index.html con Live Server o navegador

Requisitos:

- SO: Windows 10, Linux o macOS
- RAM: mínimo 4 GB (recomendado 8 GB)
- Node.js, Git, SQLite
- Internet solo para instalar dependencias e integrar GitHub

Diseño de despliegue en entorno de producción

Solución propuesta: Plataforma PaaS como **Render**

Justificación:

- Soporte nativo para aplicaciones Node.js
- CI/CD automatizado desde GitHub
- Plan gratuito
- Despliegue rápido sin necesidad de configurar servidor
- Ligero y accesible

Pasos de despliegue:

1. Subir el código a repositorio GitHub
2. Crear cuenta en Render
3. Crear nuevo servicio web con Node.js
4. Configuración de:
    a. Build de proyecto: *npm install*
    b. Arranque: *npm run start*
    c. Variables de entorno, si son necesarias.
5. Seleccionar dominio y desplegar
6. App accesible desde https://devops-hub.onrender.com

Consideraciones adicionales

En desarrollo local estoy usando SQLite para entorno funcional, pero en un entorno de producción, usaré PostgreSQL, debido a que permite conexión de múltiples usuarios, tiene mecanismos de seguridad, autenticación, roles y permisos, permite hacer copias de seguridad, y es escalable y seguro.

Descripción de la documentación a entregar

- **Introducción y justificación del problema:** ya reflejada en el anteproyecto, recoge el propósito del proyecto y la necesidad detectada en el entorno productivo.
- **Tecnologías y decisiones de diseño:** descritas en el apartado técnico, donde se detalla el uso de Node.js, Express, SQLite, Bootstrap, etc., con justificación de su idoneidad.
- **Documentación de la API REST**: se documentará al menos un endpoint funcional con método, parámetros y respuesta esperada (formato JSON)
- **Diagrama de arquitectura de tres capas:** representación gráfica de la separación entre frontend, lógica del servidor y base de datos, con tecnologías asociadas.
- **Fragmentos de configuración clave:** ejemplos seleccionados del archivo server.js, definición de rutas y middlewares del backend.
- **Instrucciones de despliegue (local y propuesta de producción):** guía paso a paso para levantar la aplicación en entorno de desarrollo y propuesta de despliegue en Render para producción.

Bibliografía

- Duckett, J. "HTML and CSS: Design and Build Websites"
- Mozilla Developer Network: https://developer.mozilla.org/es/
- Bootstrap Docs: https://getbootstrap.com/docs/
- Node.js: https://nodejs.org/en/docs/
- Express: https://expressjs.com/
- SQLite: https://www.sqlite.org/docs.html
- GitHub REST API: https://docs.github.com/es/rest
- Jacobson, I. et al. "Proceso Unificado"
- Fowler, M. "UML Distilled"
- Pressman, R. "Software Engineering"

Observacións

Este documento ha sido generado combinando contenido redactado por mí, el autor, y usando asistencia de herramientas de IA (ChatGPT 4-o y Claude 3.7) para la tabla de costes, Gantt y primeras versiones. Toda la redacción final ha sido supervisada y adaptada personalmente para cumplir los criterios formales del ciclo de proyecto.

Rúbrica de uso de Inteligencia Artificial

Apartado	Uso de IA	Edición por parte del autor
Tabla de tecnologías y costes	Generación inicial de estructura	Revisada y adaptada a las herramientas reales utilizadas
Diagrama de Gantt	Propuesta de fases y tiempos estimados	Ajustada a la dedicación real del proyecto
Estimación de recursos materiales	Formato propuesto por IA	Cálculos comprobados y redacción final por el autor
Bibliografía	Sugerencias de fuentes	Verificación, adición y ordenación personal