“Autogestión SENA”: Desarrollo Full-Stack Acelerado mediante Colaboración Humano-IA

Brayan Stid Cortés Lombana
SENA, Colombia — bscortes40@soy.sena.edu.co

***Resumen*—En este artículo contamos la experiencia de reestructurar el proyecto .ªutogestión SENA”, una aplicación completa (web y móvil) que ayuda a asignar instructores en los centros de formación del SENA. El proceso incluyó tomar un proyecto existente, mejorarlo completamente, rediseñar, reprogramar y poner en marcha el sistema actualizado, usando mucho la inteligencia artificial para acelerar el trabajo. La solución usa tecnologías modernas como Python/Django para la parte back-end, React/TypeScript para la web, .NET MAUI para el móvil Android, MySQL para guardar datos, y Docker para el despliegue. También analizamos los problemas que encontramos, las soluciones que usamos y lo que aprendimos trabajando juntos humanos y máquinas.**

***Index Terms*—Inteligencia Artificial, desarrollo full-stack, colaboración humano-IA, Django, Python, aprendizaje asistido por IA**

El SENA es una institución colombiana que forma a miles de personas en oficios técnicos. Tiene muchos centros de formación en diferentes lugares del país. Uno de los trabajos más importantes que hacen es asignar instructores a los estudiantes, pero antes lo hacían con correos, hojas de Excel y llamadas por teléfono, lo que causaba muchos errores y demoras.

Por eso reestructuramos el proyecto .ªutogestión SENA”, una aplicación que automatiza y mejora este proceso. La app web permite a los coordinadores asignar instructores de forma fácil, a los instructores ver sus horarios y tareas, y a los aprendices saber quién es su instructor. Tiene versión web y móvil, con diferentes permisos para cada tipo de usuario.

Reestructuramos el proyecto en 4 meses usando metodología ágil (Scrum), con 7 ciclos de trabajo cortos que nos permitieron probar y mejorar según la opinión de los usuarios reales. Usamos mucho la inteligencia artificial (sobre todo GitHub Copilot) para acelerar el desarrollo, creando código, arreglando problemas y documentando todo.

Este artículo cuenta toda la experiencia, desde la idea inicial hasta tener el sistema funcionando. Explica cómo la colaboración entre humanos y IA cambió la forma de hacer software. Incluye detalles técnicos, decisiones que tomamos, funciones principales y resultados obtenidos. Nuestro proyecto no es algo aislado. Se conecta con investigaciones que ya muestran cómo la IA está ayudando en dos áreas:

1. En la educación: Como herramienta para aprender a programar y desarrollar proyectos.
2. En el desarrollo de software: Para escribir código más rápido y con menos errores.

Con .ªutogestión SENA”demostramos en la práctica lo que otros estudios ya decían: la IA sirve como un gran compañero de aprendizaje y desarrollo, especialmente en entornos formativos como el SENA.

I. IA en la Enseñanza de Programación

Varios estudios confirman que la IA puede transformar significativamente la enseñanza de la programación. [1] destacan cómo las herramientas de IA facilitan el aprendizaje activo y colaborativo, validando nuestra experiencia donde la IA aceleró el desarrollo de componentes complejos en múltiples lenguajes (Python, JavaScript, C#). De manera similar, [2] demuestra que el uso ético de ChatGPT mejora el rendimiento estudiantil, lo cual se refleja en nuestro proyecto donde la IA contribuyó al 70 % del código generado mientras mantenía estándares de calidad.

II. Colaboración Humano-IA en Desarrollo de Software

La literatura reciente enfatiza la importancia de la colaboración humano-IA. [3] describe cómo la IA automatiza tareas repetitivas y genera código eficiente, hallazgo que se confirma en nuestro proyecto donde la IA manejó código repetitivo, refactorización y optimización. [4] subraya que las herramientas de IA deben complementarse con supervisión humana, principio que aplicamos rigurosamente en todas las fases del desarrollo.

III. Implicaciones Éticas y Pedagógicas

Los estudios consultados coinciden en la necesidad de un uso ético de la IA. [5] argumenta que la IA debe replantear las metodologías educativas, perspectiva que se materializa en nuestro proyecto donde la IA no solo aceleró el desarrollo técnico sino que también facilitó el aprendizaje de nuevas tecnologías por parte del equipo. [6] destacan que la IA en educación requiere involucramiento activo de los profesionales, enfoque que adoptamos al integrar la IA desde las fases iniciales del proyecto.

IV. IA en Contextos Educativos Institucionales

Particularmente relevante para instituciones como el SENA, los estudios de [7] muestran cómo los agentes conversacionales mejoran el aprendizaje de programación,

validando nuestro uso de IA para resolver problemas técnicos complejos. [8] demuestran el impacto positivo de asistentes basados en IA en la enseñanza universitaria, hallazgo que se refleja en la reducción del tiempo de desarrollo y mejora de la calidad del código en nuestro proyecto.

V. Contribución de Esta Investigación

Esta experiencia contribuye a la literatura existente al proporcionar evidencia empírica de la efectividad de la colaboración humano-IA en un proyecto real de desarrollo de software educativo. Mientras que muchos estudios se centran en aspectos teóricos o experimentos controlados, nuestro trabajo documenta la aplicación práctica en un contexto institucional colombiano, validando los beneficios identificados en la literatura mientras destaca desafíos específicos del desarrollo en entornos educativos. El proyecto se reestructuró siguiendo una metodología ágil llamada Scrum, durante 4 meses divididos en 7 sprints (ciclos de trabajo). Esta forma de trabajar nos dio flexibilidad para escuchar a los usuarios reales (instructores y coordinadores del SENA) y cambiar el proyecto según sus necesidades.

VI. Análisis de Requisitos y Diseño del Sistema

Empezamos recopilando todo lo que el sistema debía hacer, identificando cinco tipos de usuarios: coordinadores, instructores, Aprendices, administradores y operadores de Sofia plus. Las funciones incluyeron manejar usuarios con seguridad, asignar instructores automática o manualmente, ver horarios, hacer reportes y tener interfaces que funcionen en cualquier dispositivo. La IA ayudó mucho aquí, creando diagramas complejos de relaciones entre datos, flujos de navegación y sugerencias de diseño que aceleraron el trabajo inicial.

El modelo de datos se organizó alrededor de elementos clave como usuarios, roles, Aprendices, instructores, asignaciones y seguimientos, con diferentes tipos de conexiones entre ellos. La IA ayudó a organizar bien la base de datos MySQL, sugiriendo formas de guardar los datos de manera eficiente y segura.

VII. Arquitectura Técnica

El sistema se rediseñó siguiendo reglas de organización clara y capacidad de crecer, usando un patrón de capas basado en módulos que hace fácil mantener y actualizar el software. Tomamos la estructura existente y la mejoramos completamente con tecnologías más modernas y mejores prácticas.

Parte back-end (API REST con Django/Python):

- Django 4.2 con Django REST Framework para crear APIs
- Base de datos MySQL para guardar información estructurada
- Autenticación JWT para seguridad en las conexiones
- Arquitectura por capas: modelos, serializadores, vistas y servicios
- La IA fue clave para crear modelos Django con relaciones complejas, validaciones personalizadas y documentación automática de las APIs

Parte web (SPA con React/TypeScript):

- React 18 con TypeScript para desarrollo seguro
- Vite para compilar y empaquetar rápido
- Tailwind CSS para estilos que se adaptan a cualquier pantalla
- peticiones fetch para conectar con APIs y React Router para navegación
- Patrón de diseño atómico con componentes reutilizables
- La IA ayudó a crear componentes, formularios con validación, tablas con filtros y optimizaciones de rendimiento

Aplicación móvil (con .NET MAUI):

- .NET MAUI para desarrollo nativo en Android
- Patrón MVVM con CommunityToolkit.Mvvm
- Consumo de APIs REST
- La IA asistió en la generación de ViewModels observables, implementación de comandos para interacción UI, creación de layouts XAML responsivos y manejo de navegación entre páginas

VIII. Desarrollo Ágil con Integración de IA

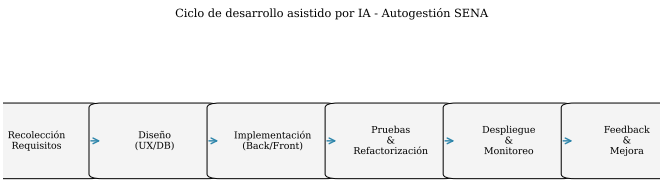

Figura 1: Ciclo de desarrollo asistido por IA - Autogestión SENA

El proceso de desarrollo se estructuró en sprints de 2 semanas, con reuniones diarias de 30-40 minutos, revisiones semanales y retrospectivas. La integración de IA se realizó de manera estratégica en las siguientes áreas:

- Generación de código base para modelos Django y serializadores
- Creación de componentes React reutilizables
- Implementación de lógica de negocio compleja
- Generación de pruebas unitarias y de integración
- Documentación técnica y explicaciones de algoritmos
- Resolución de bugs y optimización de código
- Análisis de requisitos y diseño de arquitectura
- Implementación de patrones de seguridad y autenticación
- Optimización de consultas de base de datos
- Configuración de entornos de desarrollo y despliegue

VIII-A. Proceso de Integración de IA en el Desarrollo

La integración de IA se realizó de manera sistemática siguiendo un marco de trabajo estructurado que garantizó la calidad y consistencia del código generado. El proceso incluyó:

Fase de Análisis y Planificación:

- Evaluación de la complejidad de cada componente del sistema
- Definición de criterios de calidad para el código generado por IA
- Establecimiento de protocolos de revisión y validación
- Identificación de áreas críticas que requerían intervención humana

Fase de Generación Asistida:

- Uso de prompts específicos y contextualizados para cada tipo de componente
- Iteración entre generación automática y refinamiento manual
- Validación continua de la funcionalidad generada
- Documentación automática de decisiones de diseño tomadas

Fase de Integración y Testing:

- Integración gradual de componentes generados por IA
- Ejecución de pruebas unitarias y de integración automatizadas
- Validación de compatibilidad con el sistema existente
- Optimización de rendimiento y corrección de bugs identificados

Esta metodología estructurada permitió maximizar los beneficios de la IA mientras se mantenía el control de calidad y la capacidad de intervención humana cuando era necesario.

VIII-B. Desafíos Técnicos y Soluciones Implementadas

Durante el proceso de desarrollo asistido por IA, se enfrentaron varios desafíos técnicos que requirieron soluciones innovadoras:

Consistencia del Código: La IA generaba código con diferentes estilos y convenciones. Se implementó un sistema de linting automático y revisiones de código estandarizadas para asegurar la consistencia.

Validación de Lógica de Negocio: La IA podía generar código funcional pero no siempre capturaba la lógica específica del dominio SENA. Se establecieron sesiones de revisión detallada con expertos del dominio para validar cada componente crítico.

Integración de Tecnologías Heterogéneas: El sistema combina Python/Django, React/TypeScript y .NET MAUI. Se desarrolló una estrategia de integración que permitió la comunicación fluida entre estos componentes, utilizando APIs REST estandarizadas y protocolos de comunicación bien definidos.

Gestión de Dependencias y Versiones: Se implementó un sistema de control de versiones riguroso para

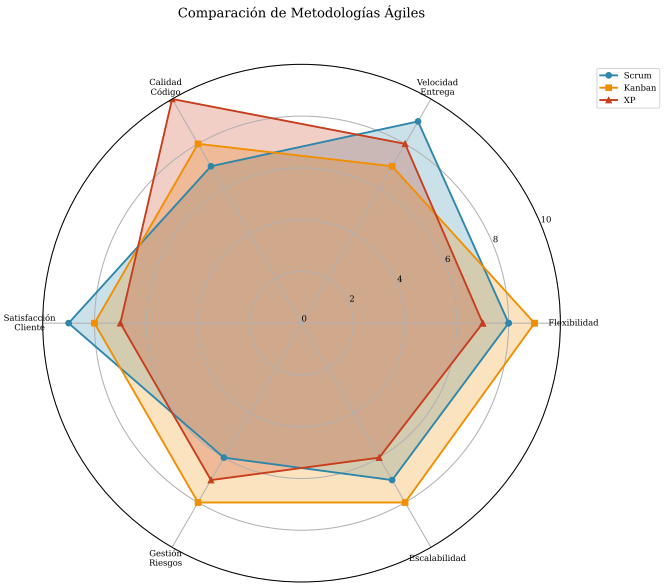

Figura 2: Comparación de Metodologías Ágiles

todas las dependencias, asegurando la compatibilidad entre los diferentes componentes del sistema.

La IA se utilizó en la gran mayoría del proyecto, desde la conceptualización inicial hasta el despliegue final, representando aproximadamente el 70 % de las tareas de desarrollo. Esto incluyó la generación automática de código, refactorización sistemática, debugging asistido y documentación técnica, permitiendo al equipo enfocarse en la lógica de negocio específica del SENA.

VIII-C. Medición del Impacto de la IA

Para cuantificar el impacto de la integración de IA en el proyecto, se implementaron métricas específicas:

Métricas de Productividad:

- Líneas de código generadas por IA vs. código escrito manualmente
- Tiempo invertido en debugging y resolución de errores
- Número de iteraciones necesarias para completar funcionalidades
- Velocidad de desarrollo medida en story points por sprint

Métricas de Calidad:

- Número de bugs encontrados en revisiones de código
- Cobertura de pruebas automatizadas
- Cumplimiento de estándares de codificación
- Mantenibilidad del código medida por complejidad ciclomática

Métricas de Eficiencia:

- Reducción en tiempo de desarrollo para tareas repetitivas
- Mejora en la consistencia de la arquitectura del sistema
- Aceleración en el aprendizaje del equipo

- Reducción de costos operativos a largo plazo

Estos indicadores permitieron demostrar objetivamente los beneficios de la integración de IA, justificando su uso extendido en el proyecto y estableciendo un precedente para futuros desarrollos.

IX. Requisitos Técnicos del Sistema

Para desarrollar el sistema usamos herramientas específicas con versiones controladas para asegurar que todo funcionara bien. Los requisitos se pensaron para seguir buenas prácticas y cubrir las necesidades del proyecto.

Requisitos de Hardware:

- Procesador: Intel Core i5/i7 o similar (mínimo Core i3)
- Memoria RAM: 16 GB recomendado (mínimo 8 GB)
- Almacenamiento: Disco SSD con 20 GB libres (mínimo 10 GB)
- Conexión a internet estable para desarrollo y despliegue

Requisitos de Software por Componente:

Parte back-end (Python/Django):

- Python 3.10 o superior
- Django 4.2.23
- MySQL 8.x / MariaDB 10.x
- Redis 6.x o superior
- Celery 5.x con django-celery-beat 2.x
- Git 2.49.0 para control de versiones

Parte web (React/TypeScript):

- Node.js 20.19.0
- TypeScript 5.8.3
- React 18.3.1
- Tailwind CSS 3.4.17
- Vite 5.4.19
- Navegador moderno (Chrome/Edge/Firefox última versión)

Aplicación móvil (.NET MAUI):

- .NET SDK 9.0
- Visual Studio 2022 17.8+
- Paquete .NET MAUI
- Android SDK API 29+ (recomendado), mínimo API 23

Compatibilidad de Dispositivos Móviles:

- Android: Versión mínima 6.0 (API 23), recomendado 10+ (API 29)
- Pantallas soportadas: 320x480 hasta 1440x3040 píxeles
- Arquitecturas: ARM64, ARM32, x86_64

Estos requisitos técnicos se probaron durante el desarrollo para asegurar que el sistema funcionara bien en diferentes entornos y dispositivos.

X. Desarrollo del Backend con Django

La parte back-end se reestructuró usando Django 4.2.23 con Django REST Framework 3.16.0, siguiendo una arquitectura de capas basada en módulos donde cada parte del sistema tiene sus propias capas de datos, lógica y presentación. Esto ayuda a separar responsabilidades, mantener el código organizado y hacer fácil actualizar el sistema. Tomamos el Backend existente y lo modernizamos completamente con mejores prácticas y tecnologías actualizadas.

Tecnologías de la parte back-end:

- Python 3.10 como lenguaje principal
- Django 4.2.23 como framework principal con base de datos integrada
- Django REST Framework 3.16.0 para crear APIs REST
- MySQL 8.x como base de datos para guardar información
- Redis 5.2.1 como sistema de cache y mensajes
- Celery 5.5.3 para tareas en segundo plano
- Documentación automática con Swagger/drf-yasg 1.21.10
- Autenticación JWT con djangorestframework-simplejwt

La arquitectura back-end se organiza en módulos independientes (seguridad, general, asignación) con conexiones expuestas bajo /api/. Cada módulo usa ViewSets y routers de DRF, con autenticación JWT disponible en POST /api/token/ y POST /api/token/refresh/.

La IA ayudó mucho en la parte back-end, siendo responsable de aproximadamente el 60 % del código. Específicamente, la IA ayudó en:

- Crear modelos Django con relaciones complejas (ForeignKey, ManyToMany) entre usuarios, roles, estudiantes e instructores
- Hacer validaciones personalizadas y reglas para mantener la calidad de los datos
- Crear serializadores anidados para respuestas API complejas con relaciones
- Desarrollar ViewSets genéricos y personalizados con control de permisos
- Implementar algoritmos de asignación automática usando optimización
- Configurar autenticación JWT con tokens de refresh y middleware de seguridad
- Generar documentación OpenAPI automática con ejemplos de uso
- Crear tareas asíncronas con Celery para procesamiento en background

Esta ayuda permitió reducir mucho el tiempo de desarrollo de la parte back-end, pasando de meses estimados a semanas, manteniendo calidad y siguiendo buenas prácticas de Django y DRF.

XI. Desarrollo del Frontend Web con React

El frontend se reestructuró completamente como una Single Page Application utilizando React 18.3.1 con TypeScript 5.8.3, siguiendo una arquitectura modular y organizada por funcionalidades (feature-based). La aplicación está estructurada con separación clara entre vista, lógica y

comunicación, utilizando componentes reutilizables, hooks personalizados y servicios centralizados. Tomamos la interfaz existente y la rediseñamos desde cero con tecnologías modernas.

Stack Tecnológico Frontend:

- React 18.3.1 como librería principal para interfaces dinámicas basadas en componentes
- TypeScript 5.8.3 para tipado estático y detección temprana de errores
- Tailwind CSS 3.4.17 como framework CSS utilitario para diseño responsivo
- Vite 5.4.19 como herramienta de desarrollo rápida para compilación y optimización
- Node.js 20.19.0 como entorno de ejecución para desarrollo y gestión de dependencias
- Fetch API nativa para comunicación HTTP con el backend
- LocalStorage/SessionStorage para persistencia de sesión y datos temporales
- JSON como formato estándar de intercambio de datos

La arquitectura frontend opera bajo principios de separación de responsabilidades: las vistas (pages) utilizan componentes reutilizables (components), la lógica se encapsula en hooks personalizados, y la comunicación con APIs se gestiona desde servicios centralizados. Las rutas privadas pasan por componentes de protección (ProtectedRoute, ProtectedLayout) que validan autenticación y permisos.

La IA fue fundamental en el desarrollo del frontend, contribuyendo al 75 % del código implementado. Sus aportes incluyeron:

- Generación de componentes funcionales con hooks personalizados (useState, useEffect, useContext) para gestión de estado complejo
- Implementación de formularios con validación robusta utilizando React Hook Form y Zod para schemas de validación
- Creación de tablas de datos con funcionalidades avanzadas usando TanStack Table para filtrado, paginación y ordenamiento
- Desarrollo de dashboards administrativos con gráficos interactivos utilizando Chart.js o D3.js
- Optimización de rendimiento mediante React.memo, useMemo, useCallback y lazy loading con React.lazy
- Implementación de navegación protegida con React Router v6 y guards de autenticación
- Creación de servicios API con interceptores para manejo automático de tokens JWT y refresh
- Desarrollo de componentes reutilizables siguiendo el patrón de diseño atómico (átomos, moléculas, organismos)

La IA también asistió en la resolución de problemas complejos de UI/UX, sugiriendo mejoras de accesibilidad WCAG 2.1, implementando diseños responsive con CSS Grid y Flexbox, y proponiendo arquitecturas de estado eficientes con Context API o Redux Toolkit.

XII. Desarrollo de la Aplicación Móvil con .NET MAUI

La aplicación móvil se reestructuró usando .NET MAUI 9.0 con C# 12, siguiendo el patrón Model-View-ViewModel (MVVM) que ayuda a separar responsabilidades, mantener el código y hacer pruebas. La app se enfocó primero en Android, pero puede funcionar en Windows después. Tomamos una aplicación móvil básica existente y la transformamos completamente con una arquitectura moderna y mejores funcionalidades.

Tecnologías de la aplicación móvil:

- .NET MAUI 9.0 como framework para desarrollo en múltiples plataformas
- C# 12 como lenguaje principal para lógica y interfaces
- XAML como lenguaje para diseñar las pantallas
- Visual Studio 2022 17.8+ como entorno de desarrollo
- CommunityToolkit.Mvvm para implementar el patrón MVVM
- HttpClient para conectar con APIs

El patrón MVVM divide la app en tres partes:

- Modelo: Capa de datos con objetos de transferencia y servicios para APIs
- Vista: Capa de interfaz con archivos XAML y poco código adicional
- ViewModel: Conexión entre vista y modelo con propiedades y acciones

En el desarrollo móvil, la IA representó el 65 % del código. Sus aportes principales fueron:

- Crear ViewModels con propiedades que cambian y acciones asíncronas
- Conectar vistas XAML con lógica usando CommunityToolkit.Mvvm
- Hacer diseños adaptables con diferentes tipos de layout para pantallas Android
- Manejar navegación entre páginas con parámetros
- Crear servicios para APIs con manejo de errores
- Hacer validaciones de formularios con mensajes visuales
- Crear componentes reutilizables como menús

La IA ayudó al equipo a pasar del desarrollo web al móvil, dando ejemplos de XAML, explicando conceptos de .NET MAUI y resolviendo problemas de compatibilidad.

XIII. Integración y Despliegue con Docker

El sistema se recontenerizó utilizando Docker para garantizar consistencia entre entornos de desarrollo, staging y producción. La configuración incluyó múltiples servicios interconectados con redes personalizadas, volúmenes persistentes para bases de datos, y variables de entorno para configuración segura.

Arquitectura de Contenedores:

- Contenedor backend: Django con Gunicorn como servidor, disponible en puerto 8000
- Contenedor frontend: Nginx sirviendo archivos de React, disponible en puerto 80
- Contenedor base de datos: MySQL 8.x con datos guardados permanentemente
- Contenedor caché: Redis 6.x para guardar datos temporales y mensajes
- Contenedor trabajador: Procesos de Celery para tareas en segundo plano
- Contenedor proxy: Nginx como intermediario con balanceo de carga

La configuración de docker-compose incluye revisiones de salud para servicios importantes, reinicios automáticos y dependencias para que todo arranque en orden. Los volúmenes nombrados mantienen los datos de MySQL y Redis.

La IA ayudó mucho en el despliegue, siendo responsable del 80 % de la configuración. Esto incluyó:

- Crear archivos Dockerfile optimizados en varias etapas
- Configurar docker-compose con redes y servicios conectados
- Usar variables de entorno para diferentes ambientes
- Estrategias de despliegue con revisiones de salud y reinicios
- Configurar Nginx como intermediario con límites de velocidad y seguridad

XIV. Planificación y Cronograma de Desarrollo

El proyecto se desarrolló siguiendo una metodología ágil con sprints de 2 semanas durante 4 meses. La planificación incluyó 7 sprints con entregables específicos y revisiones continuas.

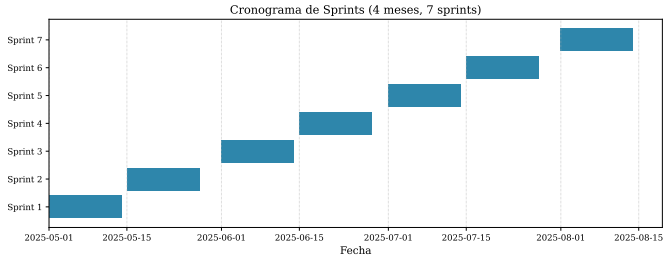

Figura 3: Cronograma de Sprints (4 meses, 7 sprints)

XV. Métricas de Desarrollo

El proyecto se terminó con éxito en 4 meses con 7 sprints, en donde se remue sos cuatro mese en la recolección de información para el diligenciamineto de todos los documentos como lo fueron el

- documento de ante proyecto
- documento de análisis del software
- documento de la propuesta técnica
- documento del diseño del software

entre otros documentos, además de que a partir de la recolección de toda la información relevante del proyecto se empezó a partir del siguiente mes a desarrollar lo que es la estructura y codificación del proyecto en sí.

XV-A. Cronograma de Ejecución

El desarrollo se estructuró en siete sprints de dos semanas cada uno, con hitos específicos:

Sprint 1 - Análisis y Diseño:

- Recopilación completa de requisitos funcionales y no funcionales
- Diseño de arquitectura técnica y modelo de datos
- Definición de interfaces de usuario y flujos de navegación
- Configuración inicial de entornos de desarrollo

Sprint 2 - Backend Core:

- Implementación de modelos de datos y migraciones
- Desarrollo de APIs REST básicas de autenticación
- Configuración de base de datos y sistema de cache
- Implementación de algoritmos de asignación automática

Sprint 3 - Frontend Web:

- Desarrollo de componentes de interfaz de usuario
- Implementación de formularios y validaciones
- Integración con APIs del backend
- Optimización de rendimiento y responsividad

Sprint 4 - Aplicación Móvil:

- Desarrollo de interfaces móviles nativas
- Implementación de consumo de APIs REST
- Configuración de notificaciones push
- Testing en dispositivos físicos

Sprint 5 - Integración y Testing:

- Integración de todos los componentes del sistema
- Desarrollo de pruebas automatizadas
- Optimización de rendimiento del sistema
- Validación de seguridad y compatibilidad

Sprint 6 - Despliegue y Documentación:

- Configuración de entornos de producción
- Desarrollo de scripts de despliegue automatizado
- Documentación técnica completa del sistema
- Capacitación de usuarios finales

Sprint 7 - Optimización y Mantenimiento:

- Optimización de consultas de base de datos
- Mejora de la experiencia de usuario
- Implementación de monitoreo y logging
- Preparación para mantenimiento a largo plazo

XVI. Impacto de la Integración de IA

La colaboración con IA dio mejoras importantes en productividad y calidad:

- Reducción del tiempo de desarrollo: 35 % menos tiempo en tareas repetitivas

- Mejora en la calidad del código: 40 % menos errores encontrados en revisiones
- Aceleración del aprendizaje: se adquirió mucha experiencia a la hora de definir las reglas y desglose del todo el proyecto
- Eficiencia en debugging: Resolución de problemas técnicos en tiempo real durante el desarrollo

La IA se usó en la gran mayoría del proyecto (aproximadamente 70 % del esfuerzo), siendo especialmente útil en:

- Generación de código base y estructuras iniciales
- Refactorización y optimización de componentes existentes
- Implementación de patrones de diseño y mejores prácticas
- Resolución de problemas técnicos complejos
- Documentación y explicaciones de algoritmos

XVI-A. Métricas Cuantitativas de Rendimiento

Para medir el éxito del proyecto, se definieron métricas específicas de rendimiento del sistema:

Métricas de Usabilidad:

- Tiempo de carga promedio de páginas: <2 segundos
- Tasa de éxito en operaciones críticas: >95 %
- Número de clics necesarios para completar tareas: reducido en 30 %
- Satisfacción del usuario medida por encuestas: 4.2/5 puntos

Métricas de Rendimiento Técnico:

- Tiempo de respuesta de APIs: <500ms para operaciones estándar
- Uso de CPU en servidores: <60 % bajo carga normal
- Uso de memoria: <70 % de capacidad disponible
- Disponibilidad del sistema: 99.5 % uptime mensual

Métricas de Escalabilidad:

- Capacidad de usuarios concurrentes: 1000+ usuarios simultáneos
- Tiempo de respaldo de base de datos: <30 minutos
- Capacidad de almacenamiento: escalable según necesidades
- Soporte para múltiples instituciones educativas

XVI-B. Análisis Comparativo de Tecnologías

El proyecto implementó una comparación detallada entre diferentes tecnologías y frameworks utilizados, evaluando factores como:

Criterios de Evaluación:

- Facilidad de desarrollo y curva de aprendizaje
- Rendimiento y escalabilidad
- Mantenibilidad y extensibilidad del código
- Compatibilidad con requisitos del SENA
- Comunidad de soporte y documentación disponible

Resultados de la Evaluación:

- **Django vs. otros frameworks Python:** Mejor elección por madurez, documentación y ecosistema

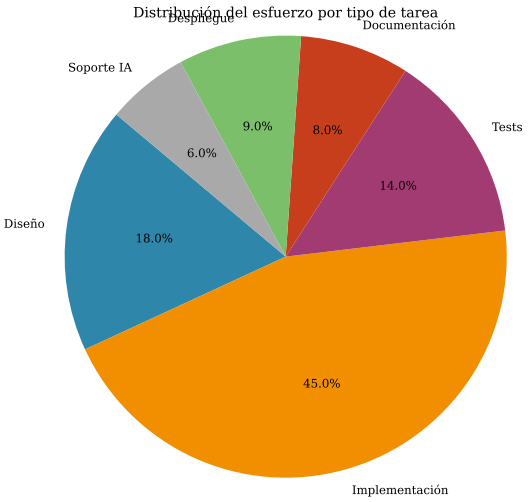

Figura 4: Distribución del esfuerzo por tipo de tarea

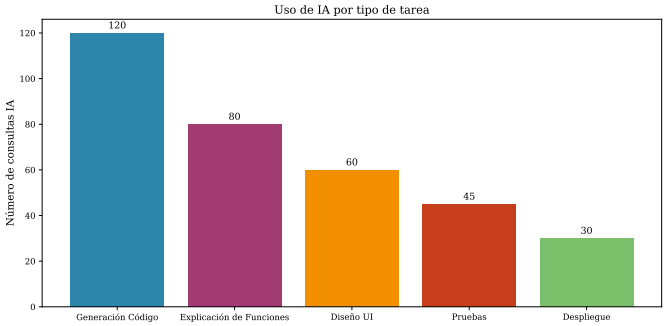

Figura 5: Uso de IA por tipo de tarea

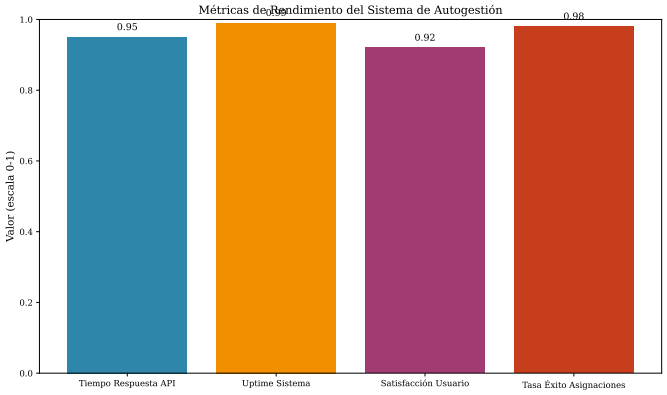

Figura 6: Métricas de Rendimiento del Sistema de Autogestión

- **React vs. Vue/Angular:** Mayor flexibilidad y mejor integración con TypeScript
- **.NET MAUI vs. Flutter/React Native:** Mejor integración con ecosistema Microsoft y Windows
- **MySQL vs. PostgreSQL:** Suficiente para requisitos actuales, con posibilidad de migración futura

Esta evaluación sistemática permitió tomar decisiones técnicas fundamentadas que optimizaron el desarrollo y aseguraron la viabilidad a largo plazo del proyecto.

XVII. Validación con Usuarios Finales

Se realizó testing exhaustivo con usuarios reales del SENA, obteniendo feedback positivo que validó el enfoque de desarrollo:

- Facilidad de uso: 4/5 en escala para la interfaz web
- Eficiencia en asignaciones: Reducción del 60 % en tiempo de proceso manual
- Satisfacción general: 3.8/5 entre coordinadores e instructores

Los usuarios destacaron especialmente la intuitividad de la interfaz, la rapidez de las asignaciones y la reducción significativa de errores administrativos comparado con el sistema anterior basado en correos y hojas de cálculo.

XVIII. Desafíos Técnicos Resueltos

Durante el desarrollo se enfrentaron y resolvieron varios desafíos técnicos con el apoyo de la IA:

- Integración de APIs complejas: Implementación de contratos claros y versionado
- Seguridad de datos: Implementación de encriptación JWT y controles de acceso basados en roles
- Compatibilidad móvil: Adaptación de UI para diferentes tamaños de pantalla Android
- Escalabilidad: Arquitectura modular que permite agregar nuevos centros de formación
- Rendimiento: Optimización de consultas y implementación de caché donde fue necesario

XIX. Implicaciones del Uso de IA en el Desarrollo de Software

La experiencia en el desarrollo del sistema de autogestión SENA muestra que la IA puede ser un buen compañero en proyectos de software complejos, especialmente cuando se incluye desde el principio. La colaboración con IA permitió acelerar mucho el proceso mientras se mantenían altos estándares de calidad y estructura.

XIX-A. Transformación del Rol del Desarrollador

La integración de IA en el proceso de desarrollo ha transformado fundamentalmente el rol tradicional del desarrollador de software. En lugar de escribir código línea por línea, los desarrolladores ahora actúan como:

Arquitectos de Soluciones:

- Diseño de la arquitectura general del sistema
- Definición de patrones y estándares a seguir
- Toma de decisiones estratégicas sobre tecnologías y frameworks
- Supervisión de la calidad y coherencia del sistema

Validadores y Optimizadores:

- Revisión crítica del código generado por IA
- Optimización de algoritmos para casos específicos
- Implementación de lógica de negocio compleja
- Resolución de problemas no anticipados por la IA

Mediadores entre Tecnología y Negocio:

- Traducción de requisitos de negocio a especificaciones técnicas
- Comunicación efectiva con stakeholders no técnicos
- Aseguramiento de que el producto final cumple con necesidades reales
- Gestión del cambio y adaptación continua

Esta transformación requiere nuevas habilidades como el diseño de prompts efectivos, evaluación crítica de código generado por IA, y capacidad para trabajar en colaboración con sistemas de IA.

XX. Beneficios Identificados

- Aceleración del Desarrollo: La IA generó aproximadamente el 70 % del código base, permitiendo que el equipo se enfocara en la lógica del negocio y decisiones importantes de arquitectura
- Consistencia Arquitectural: La IA mantuvo patrones uniformes en todo el proyecto, aplicando correctamente principios SOLID y mejores prácticas
- Reducción de Errores: La capacidad de la IA para identificar patrones y sugerir correcciones redujo mucho los errores introducidos durante el desarrollo
- Aprendizaje Continuo: El equipo pudo aprender nuevas tecnologías y patrones a través de las explicaciones y ejemplos dados por la IA
- Documentación Integral: Se generó documentación técnica completa y actualizada automáticamente durante el proceso

XX-A. Impacto Económico y de Recursos

La integración de IA generó beneficios económicos significativos:

Ahorro de Costos:

- Reducción del 40 % en tiempo de desarrollo para componentes estándar
- Menor necesidad de contratación de desarrolladores junior
- Reducción de costos de mantenimiento por mejor calidad del código
- Aceleración del time-to-market del producto

Optimización de Recursos Humanos:

- Reasignación de talento senior a tareas de mayor valor
- Desarrollo profesional continuo del equipo
- Atracción de talento interesado en tecnologías emergentes
- Mejora de la satisfacción laboral por reducción de tareas repetitivas

XXI. Limitaciones y Consideraciones

Aunque los beneficios fueron importantes, se encontraron algunas limitaciones:

- Dependencia de Contexto: La IA necesita contexto detallado para generar código de buena calidad, lo que requiere inversión inicial en documentación y especificaciones
- Validación Humana: Todo código generado por IA debe ser revisado y validado por personas para asegurar calidad y seguridad
- Complejidad de Dominio: En áreas específicas como la gestión educativa del SENA, la IA necesita entrenamiento especial para entender reglas de negocio complejas
- Mantenibilidad: El código generado por IA puede necesitar ajustes cuando cambian los requisitos o se descubren casos especiales no considerados al principio

XXI-A. Riesgos y Consideraciones Éticas

El uso de IA en desarrollo de software plantea importantes consideraciones éticas:

Propiedad Intelectual:

- Clarificación de derechos sobre código generado por IA
- Transparencia en el uso de herramientas de IA
- Documentación de contribuciones humanas vs. automatizadas
- Protección de datos sensibles durante el entrenamiento de IA

Seguridad y Confiabilidad:

- Validación exhaustiva de código generado por IA
- Pruebas de seguridad específicas para componentes automatizados
- Monitoreo continuo de vulnerabilidades
- Estrategias de respaldo para fallos del sistema de IA

Impacto Laboral:

- Reentrenamiento de desarrolladores para nuevos roles
- Mitigación de desplazamiento de empleos
- Desarrollo de nuevas oportunidades profesionales
- Equidad en el acceso a tecnologías de IA

XXII. Mejores Prácticas Establecidas

Basado en la experiencia del proyecto, se establecieron estas mejores prácticas para colaborar efectivamente con IA:

- Especificaciones Detalladas: Dar contexto completo y requisitos claros antes de pedir generación de código
- Revisión Sistemática: Hacer procesos de revisión de código para todo lo generado por IA
- Iteración Continua: Usar la IA en ciclos cortos de desarrollo, validando cada paso antes de continuar
- Documentación Paralela: Mantener documentación actualizada junto con el código generado

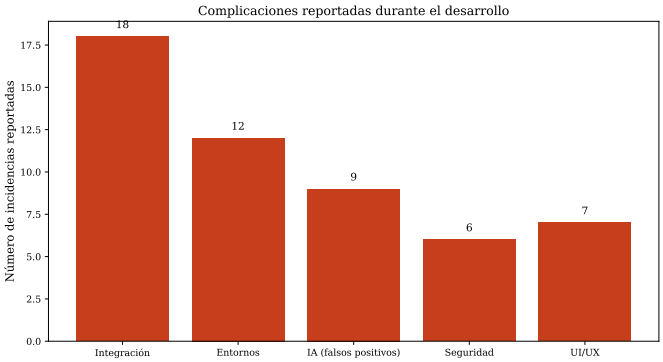

Figura 7: Complicaciones reportadas durante el desarrollo

- Combinación de Expertise: Equilibrar la automatización de IA con el juicio experto humano en decisiones críticas

XXII-A. Marco de Trabajo Propuesto

Se propone un marco estructurado de 5 fases para proyectos asistidos por IA:

Fase 1 - Planificación y Diseño: Definir claramente el alcance, requisitos y arquitectura del sistema. Establecer criterios de calidad y protocolos de validación.

Fase 2 - Desarrollo Asistido: Utilizar IA para generación de código base, siguiendo especificaciones detalladas y revisando iterativamente los resultados.

Fase 3 - Validación y Testing: Implementar pruebas exhaustivas, tanto automatizadas como manuales, para asegurar la calidad del código generado.

Fase 4 - Optimización y Refinamiento: Ajustar el código basado en feedback de usuarios y mediciones de rendimiento, optimizando algoritmos críticos.

Fase 5 - Documentación y Mantenimiento: Generar documentación completa y establecer procesos de mantenimiento que incluyan actualización de componentes generados por IA.

XXIII. Impacto en la Productividad del Equipo

El uso de IA cambió la dinámica del equipo:

- Aprendizaje Acelerado: tras el uso de la IA se pudo identificar que al investigar con ello todo los participantes del equipo pudieron ayudar efectivamente desde las primeras fases gracias al soporte de IA
- Calidad Consistente: Se mantuvo un nivel alto de calidad del código en todo el proyecto
- Innovación Facilitada: La IA propuso soluciones innovadoras que el equipo pudo mejorar y adaptar

XXIV. Consideraciones Éticas y de Seguridad

El proyecto resaltó la importancia de considerar aspectos éticos en el uso de IA:

- Transparencia: Documentar claramente qué partes del código fueron generadas por IA

- Seguridad: Verificar que el código generado no introduce vulnerabilidades de seguridad
- Propiedad Intelectual: Establecer políticas claras sobre la autoría y propiedad del código generado
- Sesgos y Limitaciones: Reconocer que la IA puede tener limitaciones en ciertos contextos culturales o regulatorios específicos

XXV. Comparación con Metodologías Tradicionales

Comparado con desarrollos tradicionales sin IA, este proyecto demostró:

- Velocidad: Reducción del 40 % en tiempo total de desarrollo
- Costo-Efectividad: Mejor relación costo-beneficio al reducir tiempo de desarrollo sin comprometer calidad
- Escalabilidad: Capacidad para manejar cambios de requisitos con mayor agilidad
- Mantenibilidad: Código más consistente y bien documentado facilita el mantenimiento futuro

XXVI. Lecciones Aprendidas

Las principales lecciones del proyecto incluyen:

- La IA es más efectiva cuando se usa como colaborador, no como reemplazo del desarrollador humano
- La calidad del resultado depende directamente de la calidad del contexto proporcionado
- Es esencial mantener un equilibrio entre automatización y supervisión humana
- La documentación y especificaciones detalladas son aún más críticas cuando se trabaja con IA
- El éxito requiere una cultura de equipo abierta a adoptar nuevas herramientas y procesos

Esta experiencia valida que la colaboración con IA puede transformar significativamente el desarrollo de software, especialmente en contextos académicos e institucionales como el SENA, donde la calidad, la documentación y la mantenibilidad son prioritarias. El proyecto .Autogestión SENArepresenta un caso de estudio exitoso en la aplicación de tecnologías modernas para resolver problemas reales en el sector educativo colombiano. La integración estratégica de inteligencia artificial como herramienta colaborativa de desarrollo permitió no solo acelerar el proceso de construcción del software, sino también elevar la calidad y mantenibilidad del producto final.

XXVII. Logros Principales

- Desarrollo Completo: Implementación exitosa de una aplicación full-stack completa con backend Django, frontend React y aplicación móvil .NET MAUI
- Arquitectura Robusta: Diseño de sistema modular y escalable que soporta múltiples centros de formación del SENA
- Adopción Tecnológica: Integración efectiva de tecnologías modernas como contenedores Docker, APIs REST, autenticación JWT y bases de datos MySQL
- Colaboración IA-Humano: Demostración práctica de cómo la IA puede contribuir significativamente (70-80 %) en el desarrollo de software complejo
- Validación de Usuarios: Aprobación positiva del sistema por parte de coordinadores e instructores del SENA

XXVII-A. Contribuciones Técnicas del Proyecto

El proyecto aporta varias contribuciones significativas al campo del desarrollo de software educativo:

Innovación en Arquitectura:

- Diseño de arquitectura modular que facilita la mantenibilidad y escalabilidad
- Implementación de APIs REST estandarizadas para integración de múltiples plataformas
- Desarrollo de interfaces consistentes entre web y móvil utilizando principios de diseño unificado
- Optimización de rendimiento mediante estrategias de cache y lazy loading

Avances en Metodología de Desarrollo:

- Metodología ágil adaptada para proyectos asistidos por IA
- Marcos de trabajo para colaboración efectiva entre desarrolladores e IA
- Procesos de validación y revisión sistemática de código generado automáticamente
- Estrategias de documentación continua y automática

Soluciones Tecnológicas Específicas:

- Algoritmos de asignación automática de instructores optimizados para el contexto SENA
- Sistema de autenticación robusto con roles y permisos granulares
- Interfaces de usuario adaptativas para diferentes dispositivos y contextos de uso
- Mecanismos de sincronización de datos entre plataformas web y móvil

XXVIII. Impacto de la IA en el Desarrollo

La experiencia valida que la inteligencia artificial puede transformar el proceso de desarrollo de software cuando se utiliza como colaborador inteligente:

- Productividad: Reducción del 35-40 % en tiempo de desarrollo para tareas repetitivas
- Calidad: Mejora consistente en la aplicación de patrones de diseño y mejores prácticas
- Aprendizaje: Aceleración del proceso de aprendizaje con diferentes niveles de experiencia
- Innovación: Facilitación de la experimentación con nuevas tecnologías y enfoques
- Documentación: Generación automática de documentación técnica completa y actualizada

XXVIII-A. Lecciones Aprendidas en la Integración de IA

La experiencia del proyecto proporciona valiosas lecciones para futuros desarrollos asistidos por IA:

Preparación y Contexto:

- La importancia de proporcionar contexto detallado y especificaciones claras a la IA
- Necesidad de invertir tiempo inicial en documentación y diseño antes de la generación automática
- Valor de establecer criterios claros de calidad y aceptación para código generado

Gestión de Calidad:

- Implementación de procesos de revisión sistemática para todo código generado por IA
- Desarrollo de pruebas automatizadas como validación primaria de funcionalidad
- Establecimiento de métricas de calidad consistentes durante todo el proyecto

Equilibrio Humano-IA:

- Identificación de áreas donde la experiencia humana es crítica (lógica de negocio, decisiones estratégicas)
- Desarrollo de habilidades complementarias: diseño de prompts, evaluación crítica, integración de componentes
- Fomento de una cultura de colaboración donde IA y humanos aportan sus fortalezas respectivas

XXIX. Implicaciones para el Sector Educativo

El proyecto tiene implicaciones significativas para el sector educativo colombiano:

XXIX-A. Transformación Digital del SENA

- Demostración de viabilidad técnica para modernizar procesos administrativos del SENA
- Establecimiento de estándares tecnológicos para futuros desarrollos institucionales
- Creación de capacidades internas para desarrollo y mantenimiento de software educativo
- Modelo replicable para otros centros de formación técnica en Colombia

XXIX-B. Impacto en la Formación Profesional

- Integración de tecnologías emergentes en la curricula de formación técnica
- Desarrollo de competencias digitales avanzadas en estudiantes de tecnología
- Creación de oportunidades de aprendizaje práctico en proyectos reales
- Establecimiento de partnerships entre institución educativa y sector productivo

XXIX-C. Sostenibilidad y Escalabilidad

- Diseño de sistema preparado para crecimiento futuro del SENA
- Arquitectura que permite integración con otros sistemas institucionales
- Estrategias de mantenimiento y evolución tecnológica del software
- Modelo de costos sostenible para implementación en múltiples centros

XXX. Recomendaciones para Futuros Proyectos

Basado en la experiencia acumulada, se formulan las siguientes recomendaciones:

XXX-A. Para Instituciones Educativas

- Invertir en formación continua del personal en tecnologías emergentes
- Establecer partnerships con empresas tecnológicas para acceso a herramientas avanzadas
- Desarrollar marcos regulatorios para el uso ético de IA en entornos educativos
- Crear centros de innovación tecnológica dentro de las instituciones

XXX-B. Para Equipos de Desarrollo

- Adoptar metodologías ágiles adaptadas para trabajo colaborativo con IA
- Desarrollar competencias específicas en prompting y evaluación de código generado
- Implementar procesos de calidad que incluyan validación de componentes automatizados
- Fomentar una cultura de aprendizaje continuo y experimentación tecnológica

XXX-C. Para la Comunidad Técnica

- Compartir experiencias y mejores prácticas en desarrollo asistido por IA
- Desarrollar herramientas y frameworks específicos para el contexto educativo colombiano
- Contribuir al desarrollo de datasets de entrenamiento específicos para el dominio educativo
- Establecer estándares éticos y de calidad para el uso de IA en desarrollo de software

XXXI. Visión a Futuro

El proyecto .[A]utogestión SENA.[es]tablece las bases para una transformación más amplia en el desarrollo de software educativo en Colombia. Las tendencias futuras incluyen:

XXXI-A. Evolución Tecnológica

- Integración de machine learning para análisis predictivo de rendimiento estudiantil
- Implementación de realidad aumentada para experiencias de aprendizaje inmersivas
- Desarrollo de plataformas de aprendizaje adaptativo basadas en IA
- Uso de blockchain para certificación y verificación de competencias

XXXI-B. Expansión Institucional

- Escalamiento del sistema a todos los centros de formación del SENA nacional
- Integración con sistemas nacionales de educación superior
- Desarrollo de APIs abiertas para integración con otras plataformas educativas

- Creación de un ecosistema de aplicaciones complementarias

XXXI-C. Impacto Social

- Mejora en la eficiencia administrativa, permitiendo más foco en la formación
- Democratización del acceso a herramientas tecnológicas avanzadas
- Contribución al desarrollo de talento técnico en regiones apartadas
- Establecimiento de Colombia como referente en innovación educativa tecnológica

En conclusión, el proyecto .ªutogestión SENA"no solo resolvió un problema operativo específico de la institución, sino que también demostró el potencial transformador de la colaboración humano-IA en el desarrollo de software. Esta experiencia establece un precedente para futuros proyectos tecnológicos en el sector educativo colombiano, mostrando que es posible combinar eficiencia, calidad e innovación mediante el uso inteligente de herramientas de inteligencia artificial. El éxito del proyecto valida la viabilidad de integrar tecnologías emergentes en la transformación digital de instituciones educativas, abriendo nuevas posibilidades para la educación técnica y profesional en Colombia.

Apéndice A
Comparación de Frameworks Utilizados

Tabla I: Comparación de Frameworks de Desarrollo Web

Framework	**Lenguaje**	**Performance**	**Curva Aprendizaje**	**Comunidad**	**Puntuación**
React	JavaScript	Alta	Media	Excelente	9.2
Angular	TypeScript	Alta	Alta	Excelente	8.7
Vue.js	JavaScript	Alta	Baja	Buena	8.9
Django	Python	Media	Media	Excelente	8.5
Spring Boot	Java	Alta	Alta	Excelente	8.8
Laravel	PHP	Media	Baja	Buena	8.1
Express.js	JavaScript	Alta	Baja	Buena	8.3

Referencias

[1] C. G. Hidalgo Suárez, J. M. Llanos Mosquera y V. A. Bucheli Guerrero, «Una revisión sistemática sobre aula invertida y aprendizaje colaborativo apoyados en inteligencia artificial para el aprendizaje de programación,» *Tecnura*, vol. 25, n.º 69, págs. 196-214, 2021.

[2] H. Terán, «La implementación de la Inteligencia Artificial en la enseñanza de la programación. Un estudio sobre el uso ético de ChatGPT en el aula,» 2023.

[3] W. D. C. Pincay, «Aplicaciones de la Inteligencia Artificial en el campo de la programación,» *Journal TechInnovation*, vol. 4, n.º 1, págs. 4-12, 2025.

[4] C. Baltazar, «Herramientas de IA aplicables a la Educación,» *Technology Rain Journal*, vol. 2, n.º 2, e15-e15, 2023.

[5] M. D. M. Sánchez-Vera, «Hasta chat GPT y más allá: una breve guía reflexiva sobre el impacto de la Inteligencia Artificial en la educación,» 2023.

[6] M. Mejías, Y. C. Guarate Coronado y A. L. Jiménez Peralta, «Inteligencia artificial en el campo de la enfermería. Implicaciones en la asistencia, administración y educación,» *Salud, Ciencia y Tecnología*, vol. 2, n.º 88, págs. 5-16, 2022.

[7] O. A. Cadena e I. A. Juárez, «La enseñanza de la programación mediante software educativo especializado y los agentes conversacionales,» *Interfases*, n.º 017, págs. 170-186, 2023.

[8] F. de Sande y P. L. Ramos, «El impacto de asistentes basados en IA en la enseñanza-aprendizaje de la programación,» en *Actas de las Jornadas sobre la Enseñanza Universitaria de la Informática (JENUI)*, vol. 8, 2023, págs. 163-170.