

INFORME FINAL DE PROYECTO DE INVESTIGACION.

I. Identificación del Proyecto

Instituto o Centro: Instituto Tecnológico de Ensenada

Nombre del/de la docente: Luis Armando Cardenas Florido

Título del proyecto: GenLab M5GP: Laboratorio en Línea para Aprendizaje Automático utilizando M5GP

Tipo de investigación: Investigación científica

Duración del proyecto: 1 año

Fecha de inicio del proyecto: 25/agosto/2025

Fecha de término del proyecto: 24/agosto/2026

Periodo de avance reportado: 25/agosto – 24 agosto/2026

II. Resultados

1. Resumen del proyecto

a) Introducción: contiene una descripción general del área de oportunidad o problemática que se abordó con el proyecto de investigación.

En la ciencia, la industria y los negocios se han generado y puesto a disposición datos a gran velocidad, este cambio, dentro del contexto de la ciencia de datos y la inteligencia artificial, se requiere de modelos de aprendizaje automático que no solo sean precisos en sus resultados, sino que también puedan usarse de manera eficiente, mantengan un buen comportamiento cuando crece el tamaño de los problemas y permitan comprender cómo se obtienen sus resultados. En este sentido, la regresión simbólica y la programación genética son una alternativa a los modelos de tipo caja negra. Su mayor ventaja es que pueden generar expresiones matemáticas explícitas, que sirven para estudiar el fenómeno que se está modelando y el conocimiento que se puede obtener a partir de los datos (Cárdenas Florido et al., 2024; Kune et al., 2016; Rudin, 2019).

El problema es que los enfoques tradicionales de programación genética aún tienen limitaciones importantes. Normalmente necesitan mucho esfuerzo de cómputo y se tornan más difíciles de aplicar cuando el volumen de los datos crece. Muchas implementaciones tampoco aprovechan plenamente las posibilidades del cómputo paralelo y de las GPU. Los mecanismos de búsqueda evolutiva también pueden responder de forma limitada a los cambios que ocurren durante el proceso. Estos problemas dificultan su aplicación a los problemas actuales de ciencia de datos y aprendizaje automático, donde el tiempo de entrenamiento, la estabilidad numérica y la generalización son importantes (Cárdenas Florido et al., 2024; Langdon, 2011; Trujillo et al., 2022).

Como propuesta nace la idea del desarrollo del proyecto GenLab M5GP: Laboratorio en Línea para el Aprendizaje Automático con M5GP. El proyecto tiene como objetivo hacer evolucionar y reforzar el algoritmo M5GP mediante mejoras algorítmicas, optimizaciones de cómputo y nuevas funcionalidades. A su vez, plantea la creación de una plataforma en internet para que estudiantes, investigadores y profesionales utilicen el algoritmo, configuren experimentos y analicen sus resultados. El trabajo está orientado a la eficiencia del proceso evolutivo, la adaptabilidad de los operadores genéticos, la estabilidad numérica de los modelos simbólicos y la incorporación de tareas de regresión y clasificación (Cárdenas Florido et al., 2024).

El acceso a esta tecnología también forma parte del problema que atiende el proyecto. Para ejecutar M5GP se requiere contar con al menos de una GPU, además de bibliotecas y configuraciones de software que no siempre están disponibles en instituciones educativas o grupos de investigación con recursos limitados. Con el desarrollo de GenLab M5GP, se propone concentrar estos elementos en un laboratorio en línea, de modo que la comunidad académica pueda emplear el algoritmo en actividades de investigación, docencia y formación sin tener que instalar o administrar directamente toda la infraestructura. Con esta propuesta se busca disminuir las barreras técnicas y económicas que limitan su uso. También se pretende facilitar la

repetición de los experimentos, la revisión de sus configuraciones y la comparación de los resultados obtenidos (Letowski et al., 2020; May et al., 2023; Pineau et al., 2021).

GenLab M5GP no se limita a mejorar el funcionamiento metodológico y computacional del algoritmo, el proyecto también aborda cuestiones de acceso tecnológico, transferencia de conocimiento y formación de capital humano. La plataforma reúne, en un mismo entorno, el desarrollo aplicado del algoritmo, la realización controlada de experimentos y su uso académico entro del campo del aprendizaje automático basado en programación genética.

b) Objetivos: contiene el esclarecimiento de los objetivos que se cumplieron con el desarrollo del proyecto de forma cualitativa.

El proyecto GenLab M5GP: Laboratorio en Línea para Aprendizaje Automático utilizando M5GP se desarrolló como un trabajo de investigación aplicada que combinó actividades de ingeniería de software, integración de componentes y pruebas de funcionamiento de la plataforma. Durante el periodo reportado se trabajó en el análisis del algoritmo M5GP, la definición de la arquitectura tecnológica y la integración de una primera versión funcional de la plataforma. Esta versión reúne el núcleo de cómputo de M5GP, la capa de servicios, las interfaces de programación y la aplicación web desde la cual los usuarios pueden configurar y ejecutar experimentos.

Los objetivos planteados para esta etapa se atendieron mediante el desarrollo e integración de una infraestructura modular que conserva las capacidades científicas y de cómputo de M5GP, al mismo tiempo que amplía sus formas de acceso. El resultado es una base tecnológica funcional, organizada y documentada, sobre la cual pueden incorporarse nuevas capacidades conforme avance el proyecto. A partir de esta versión se continuará con las pruebas experimentales, el fortalecimiento de los mecanismos de seguridad, la evaluación de la experiencia de uso y la preparación de la plataforma para su operación en condiciones regulares de trabajo.

Objetivo general alcanzado

Durante el periodo reportado se desarrolló una versión funcional de GenLab M5GP para la ejecución de experimentos de aprendizaje automático basados en M5GP. La plataforma permite trabajar con problemas de regresión simbólica y clasificación, ya sea desde una aplicación web o mediante servicios API. Además, incorpora funciones para administrar usuarios, conjuntos de datos, experimentos y recursos de cómputo disponibles en GPU.

La arquitectura quedó organizada en varios componentes con funciones claramente diferenciadas. El núcleo algorítmico mantiene la lógica de M5GP; una capa de acceso permite utilizar sus funciones desde otros módulos y conserva la compatibilidad con el esquema de estimadores de scikit-learn; el backend se encarga de la administración de usuarios, experimentos y recursos de ejecución; y el frontend proporciona la interacción gráfica con la plataforma. Esta separación permitió conservar la ejecución local mediante scripts de Python, sin hacer depender el uso científico de M5GP de la aplicación web.

También se mantuvieron las dependencias de cómputo utilizadas por la implementación actual de M5GP, entre ellas GPU NVIDIA y las bibliotecas RAPIDS, cuML, cuDF, CuPy y Numba. Para controlar el uso de estos recursos se definió una ejecución activa por GPU. Cuando el servidor dispone de varias tarjetas gráficas, el sistema puede asignar una unidad libre al experimento que se va a ejecutar. Si no existe una GPU disponible, la ejecución no inicia y el usuario recibe una notificación sobre la falta temporal de recursos. Este control busca evitar interferencias entre procesos y reducir problemas derivados de una asignación excesiva de memoria.

La plataforma incluye además funciones para crear y autenticar usuarios, cargar y seleccionar conjuntos de datos, configurar experimentos, consultar el avance de las ejecuciones y revisar los resultados obtenidos. También permite recuperar información asociada con los modelos generados. Con estas funciones se

dispone de una primera versión operativa de GenLab M5GP que reduce la necesidad de que cada usuario instale y configure por cuenta propia todo el entorno requerido por el algoritmo. Esto abre la posibilidad de utilizar M5GP con mayor facilidad en actividades de docencia, experimentación y trabajos de investigación.

Objetivos específicos cumplidos

1. Analizar la implementación de M5GP e identificar los requerimientos para su integración en una plataforma en línea

Se revisó la implementación existente de M5GP para comprender la organización del código, sus dependencias, las estructuras de datos utilizadas, los operadores disponibles, el proceso evolutivo y los mecanismos de evaluación de los modelos. Esta revisión permitió establecer con mayor precisión cómo se relacionan los distintos módulos y en qué puntos era necesario intervenir para incorporar nuevas formas de acceso al algoritmo.

El análisis también permitió identificar los requerimientos asociados con el uso de GPU y las dependencias especializadas de RAPIDS/cuML, así como los componentes que debían mantenerse sin cambios para preservar el comportamiento del algoritmo. A partir de este trabajo se definieron los ajustes necesarios para integrar M5GP con servicios web sin comprometer su operación científica original.

2. Reorganizar el núcleo de M5GP y preservar su compatibilidad con los esquemas de experimentación existentes

A partir del análisis anterior se reorganizaron componentes relacionados con la configuración del algoritmo, la construcción de expresiones, los operadores, la evaluación de individuos y los procesos de regresión y clasificación. El propósito fue separar con mayor claridad la lógica propia de M5GP de los componentes requeridos para ofrecer acceso mediante servicios externos.

La reorganización se realizó procurando conservar las formas de ejecución utilizadas previamente en el trabajo experimental. M5GP puede continuar ejecutándose localmente mediante scripts Python, siguiendo el esquema empleado en sus archivos de prueba y experimentación, sin depender de la aplicación web ni de los mecanismos de autenticación incorporados a GenLab.

También se mantuvo la compatibilidad necesaria para trabajar con interfaces similares a los estimadores de scikit-learn y con entornos de evaluación utilizados en investigación, como SRBench. De esta manera, el desarrollo de GenLab añade nuevas formas de acceso al algoritmo sin sustituir su uso directo desde código o terminal.

3. Consolidar las capacidades algorítmicas y mejorar la estabilidad del núcleo de M5GP

Se integraron y organizaron dentro del núcleo las extensiones funcionales desarrolladas para M5GP, entre ellas operadores aritméticos, trigonométricos, agregadores y condicionales. De forma paralela, se revisaron los procedimientos relacionados con la construcción y recuperación de expresiones simbólicas, el manejo de operandos y el cálculo de la complejidad de los modelos generados.

También se incorporaron controles para atender valores numéricos no válidos y situaciones que pueden presentarse durante la evaluación de individuos. Estos cambios permitieron disponer de una base más ordenada y estable para ejecutar M5GP desde los distintos mecanismos de acceso considerados en el proyecto.

4. Diseñar e implementar una arquitectura modular para la plataforma GenLab M5GP

Se definió una arquitectura compuesta por el núcleo de cómputo de M5GP, una capa de acceso, el backend, la API y la aplicación frontend. Esta separación permitió asignar responsabilidades específicas a cada componente y reducir la dependencia directa entre la lógica científica del algoritmo y las tecnologías utilizadas para ofrecer los servicios en línea.

El núcleo concentra los procesos propios de M5GP y mantiene sus requerimientos de GPU y del ecosistema RAPIDS/cuML. La capa de acceso funciona como intermediaria entre este núcleo y los demás componentes. Por su parte, el backend administra usuarios, conjuntos de datos, experimentos, resultados y recursos de ejecución; la API expone las operaciones requeridas por otras aplicaciones; y el frontend proporciona la interacción gráfica con los usuarios.

Esta organización permite que los servicios web y la interfaz puedan desarrollarse y mantenerse sin incorporar directamente las dependencias especializadas del núcleo, salvo en el momento en que se solicita la ejecución de un experimento.

5. Incorporar mecanismos de autenticación y control de acceso a la plataforma

Se integró un esquema de autenticación basado en JSON Web Tokens (JWT) para proteger los servicios de la API y restringir las operaciones que requieren una sesión válida. En la aplicación web se desarrollaron las funciones de inicio de sesión, conservación de la sesión y acceso controlado a las secciones privadas.

Este mecanismo permite relacionar los conjuntos de datos, experimentos y resultados con el usuario que los genera. Con ello se estableció una base para mantener la separación de la información, dar seguimiento a las actividades realizadas dentro de la plataforma y, posteriormente, incorporar distintos niveles de permisos o funciones administrativas.

6. Desarrollar el flujo para administrar conjuntos de datos, experimentos y resultados

Se desarrollaron los servicios necesarios para cargar conjuntos de datos, almacenarlos dentro del espacio correspondiente al usuario, consultar los archivos disponibles y seleccionar la información que será utilizada durante un experimento. También se incorporaron los elementos requeridos para establecer el tipo de tarea y los parámetros de configuración de M5GP antes de iniciar una ejecución.

La información de cada experimento puede conservarse junto con su configuración, estado de ejecución, métricas y archivos resultantes. Esto permite consultar posteriormente las condiciones en que se realizó una prueba y facilita el seguimiento de los experimentos efectuados mediante la plataforma.

Dentro del mismo flujo se contempló la recuperación de las métricas obtenidas, la representación del mejor modelo encontrado y la información necesaria para generar gráficas y otros elementos de análisis. Los resultados pueden conservarse en estructuras serializadas, entre ellas archivos JSON, lo que facilita su consulta posterior y su posible utilización desde otras herramientas de experimentación.

7. Incorporar mecanismos para controlar el uso de los recursos GPU

Debido a que la ejecución de M5GP depende de recursos GPU, se desarrolló un mecanismo para comprobar su disponibilidad antes de iniciar un experimento. En esta etapa del proyecto se adoptó como criterio de operación permitir únicamente un proceso activo de M5GP por GPU.

Cuando una unidad se encuentra ocupada, no se inicia otro proceso sobre el mismo recurso. Esta decisión reduce la posibilidad de saturación de memoria y evita que distintas ejecuciones interfieran entre sí. En equipos que dispongan de varias GPU, el mecanismo permite identificar unidades disponibles y asignarlas a diferentes ejecuciones, manteniendo el criterio de un proceso activo por GPU.

Para esta primera versión no se incorporó una cola de trabajos. La plataforma informa sobre la disponibilidad del recurso y solamente inicia el experimento cuando existe una GPU que pueda ser utilizada.

8. Desarrollar una interfaz web para el acceso y operación de GenLab M5GP

Se construyó una aplicación web utilizando React, organizada mediante páginas, componentes reutilizables, servicios y elementos de navegación. La interfaz permite realizar las principales operaciones de la plataforma: autenticarse, consultar el tablero principal, administrar conjuntos de datos, establecer los parámetros de un experimento, iniciar una ejecución, dar seguimiento a su estado y consultar los resultados obtenidos.

La estructura adoptada permite incorporar nuevas pantallas y ampliar las funciones existentes sin tener que modificar de manera extensa el resto de la aplicación. La comunicación con el backend se concentra en servicios encargados de realizar las solicitudes a la API, lo que mantiene separada la lógica de presentación de los mecanismos utilizados para intercambiar información con el servidor.

Con esta interfaz se incorporó una alternativa de acceso dirigida principalmente a usuarios que no requieren interactuar directamente con el código de M5GP, como estudiantes, docentes y usuarios que desean configurar y revisar experimentos desde un entorno gráfico.

9. Elaborar documentación técnica para facilitar el mantenimiento y la continuidad del proyecto

Como parte del desarrollo se documentaron la estructura general del sistema, la organización de módulos y archivos, las responsabilidades de los principales componentes, las dependencias de software y los procedimientos requeridos para preparar el entorno de ejecución.

La documentación también describe la comunicación entre el frontend, la API, el backend, la capa de acceso y el núcleo de M5GP, así como procedimientos para modificar componentes existentes o incorporar nuevas funciones y pantallas. Este material constituye una referencia para el mantenimiento de GenLab M5GP, la incorporación de nuevos colaboradores y la continuidad de las siguientes etapas de desarrollo.

El cumplimiento de estos objetivos permitió ampliar la forma en que M5GP puede ser utilizado, pasando de un esquema centrado principalmente en la ejecución local a una plataforma abierta que conserva dicha modalidad y, al mismo tiempo, incorpora acceso mediante API y una interfaz web.

El trabajo realizado no modificó el propósito científico del algoritmo ni sustituyó sus mecanismos tradicionales de experimentación. En cambio, se desarrolló una capa tecnológica adicional que facilita su utilización desde distintos entornos y por diferentes perfiles de usuario, manteniendo separados el núcleo de cómputo, los servicios de la plataforma y la interfaz web.

Los resultados obtenidos constituyen una versión funcional de GenLab M5GP sobre la cual pueden continuar las actividades de validación experimental, pruebas con usuarios, evaluación del desempeño en condiciones de operación y preparación para escenarios de despliegue más amplios. En este sentido, los avances alcanzados corresponden con los objetivos tecnológicos establecidos para la etapa reportada del proyecto.

Avance alcanzado (12 meses)

AÑO SABÁTICO		MESES											
1er Periodo:	ACTIVIDAD	1	2	3	4	5	6	7	8	9	10	11	12
25-08-2025 al 24-02-2026	Análisis y revisión del código actual de MSG	X	X										
	Reestructuración y optimización del algoritmo		X	X	X								
	Definición de módulos y diseño de arquitectura			X	X	X							
	Desarrollo del backend e interfaces de acceso a funciones (API's)				X	X	X	X	X				
Contará con 30 días naturales para la entrega del reporte intermedio (50%) ante el TecNM a partir de la fecha:						25 de febrero de 2026							
2do Periodo:	ACTIVIDAD	1	2	3	4	5	6	7	8	9	10	11	12
25-02-2026 al 24-08-2026	Diseño y desarrollo del frontend de la plataforma						X	X	X	X			
	Análisis de implementación en la nube o local									X	X		
	Validación y pruebas de la Plataforma										X	X	
	Registro de Propiedad Intelectual y Publicación Científica											X	X
	Implementación y puesta en marcha												X
Contará con 30 días naturales para la entrega del reporte final (100%) ante el TecNM a partir de la fecha:						25 de agosto de 2026							

- c) **Metas:** contiene los resultados concretos obtenidos en forma cuantitativa respecto a: tesis desarrolladas, publicaciones, trabajos de residencia profesional, patentes en trámite, participación en eventos, etc.

Como resultado del desarrollo del proyecto GenLab MSGP se obtuvieron productos vinculados con la formación de recursos humanos, la generación de conocimiento, la difusión académica y el desarrollo tecnológico. A continuación, se presentan las principales metas alcanzadas durante el periodo reportado.

Formación de recursos humanos.

Se contó con la participación de tres estudiantes de licenciatura vinculados directamente con las actividades del proyecto:

- Dos estudiantes de servicio social, quienes colaboraron en actividades de revisión documental, análisis del código fuente, documentación técnica, pruebas iniciales, diseño de extensiones y apoyo en la difusión del proyecto.
- Un estudiante de residencia profesional, involucrado en el análisis, reestructuración e integración del código de MSGP, así como en el desarrollo de componentes relacionados con la arquitectura tecnológica de GenLab MSGP.

Los estudiantes participantes documentados son:

- Eric Cabrera Rodríguez, servicio social.
- Ernesto Escalante Vertti, servicio social.
- Pablo Isai Cota López, residencia profesional.

En consecuencia, se reporta la incorporación de dos estudiantes de servicio social y un estudiante residente, lo que representa un resultado concreto en la formación de recursos humanos en áreas de aprendizaje automático, programación genética, desarrollo web, servicios API y computación acelerada por GPU.

Trabajos de residencia profesional.

Se desarrolló un trabajo de residencia profesional relacionado directamente con el proyecto. Las actividades incluyeron la revisión de la implementación original de M5GP, la reorganización de módulos, la documentación de componentes y el apoyo en la construcción de la plataforma GenLab M5GP.

El trabajo de residencia se realizó durante el periodo comprendido entre el 16 de octubre de 2025 y el 16 de enero de 2026, con la participación del estudiante Pablo Isai Cota López.

Producción científica:

Como producto científico directamente relacionado con el desarrollo y evolución del algoritmo utilizado en GenLab M5GP, se obtuvo la publicación de un artículo científico en una revista internacional arbitrada e indexada:

"M5GP 2.0: Extensions and enhancements to a feature construction system using genetic programming", publicado en la revista Expert Systems, con DOI 10.1111/exsy.70356.

Esta publicación presenta extensiones algorítmicas, mejoras computacionales y resultados experimentales asociados con M5GP 2.0, por lo que constituye un antecedente científico y tecnológico fundamental para la plataforma GenLab M5GP.

Adicionalmente, se continuó trabajando en la preparación de documentos técnicos y productos académicos derivados de la arquitectura, implementación y evaluación de la plataforma. Sin embargo, estos materiales no se contabilizan todavía como publicaciones adicionales, debido a que no se encuentran formalmente publicados o aceptados.

Participación en eventos académicos y actividades de difusión.

Durante el periodo reportado se realizaron cuatro actividades de difusión académica y científica, distribuidas de la siguiente manera:

- 3 conferencias de divulgación científica, dirigidas principalmente a estudiantes, docentes y público académico.
- 1 ponencia presentada en un congreso organizado por IEEE en la ciudad de Tijuana.

En estas actividades se difundieron los fundamentos de M5GP, sus aplicaciones en aprendizaje automático interpretable, las mejoras incorporadas al algoritmo y el planteamiento de GenLab M5GP como laboratorio tecnológico en línea.

La participación en el congreso IEEE permitió presentar una aplicación de M5GP en un contexto de investigación educativa, además de establecer contacto con investigadores, docentes y profesionales relacionados con inteligencia artificial, ciencia de datos y tecnologías computacionales.

Desarrollo tecnológico.

Como resultado del proyecto se desarrolló una plataforma tecnológica, integrada por los siguientes productos principales:

- Un núcleo computacional M5GP reorganizado para su integración con otros servicios.

- Una capa de acceso al algoritmo, compatible con ejecuciones locales mediante scripts Python y con interfaces de tipo scikit-learn.
- Un backend con servicios API REST para la administración de usuarios, datasets, experimentos y resultados.
- Un mecanismo de autenticación basado en JWT.
- Un sistema de administración y asignación de recursos GPU.
- Una aplicación frontend desarrollada con React.
- Un módulo de carga y administración de datasets por usuario.
- Un módulo para la configuración y ejecución de experimentos.
- Un módulo para el seguimiento del estado de las ejecuciones.
- Un módulo para la consulta, visualización y descarga de resultados.
- Documentación técnica de instalación, arquitectura, estructura de archivos y mantenimiento de la plataforma.

Estos componentes conforman una versión funcional de GenLab M5GP y representan los principales resultados tecnológicos alcanzados durante el desarrollo del proyecto. El código fuente del proyecto puede ser accedido y descargado desde la plataforma GitHub a través de la siguiente liga: <https://github.com/armandocardenasf/GenLab-m5gp>.

d) Desarrollo y resultados del proyecto:

El proyecto se desarrolló mediante un proceso incremental de investigación aplicada y desarrollo de software. Cada etapa se orientó al cumplimiento de objetivos concretos: preservar las capacidades científicas de M5GP, reorganizar su código para facilitar el mantenimiento, incorporar nuevas formas de acceso y establecer condiciones adecuadas para la ejecución y reproducción de experimentos. El trabajo no se concentró únicamente en el desarrollo de una interfaz web, sino que incluyó la revisión del núcleo algorítmico, la identificación de sus dependencias, la definición de mecanismos de comunicación entre componentes y la implementación de los servicios necesarios para integrar el flujo completo de trabajo, desde la carga de los datos hasta la obtención y consulta de los resultados.

Como punto de partida se analizó la versión base de M5GP 2.0. La revisión incluyó la representación de los individuos, los operadores evolutivos, el cálculo de aptitud, los mecanismos de selección, la mutación UMAP, la supervivencia y el reemplazo, así como los procedimientos de predicción y construcción de expresiones simbólicas. También se identificaron las dependencias técnicas necesarias para su ejecución, principalmente el uso de GPU NVIDIA, CUDA y herramientas del ecosistema RAPIDS/cuML.

Este análisis permitió determinar qué partes del sistema debían conservarse para mantener el comportamiento científico de M5GP y cuáles podían reorganizarse o complementarse para facilitar su integración con la nueva plataforma. Como parte de esta decisión se mantuvo la ejecución directa mediante scripts Python, de acuerdo con el esquema utilizado previamente en los experimentos y archivos de prueba. Por esta razón, no se incorporó un comando CLI adicional que sustituyera esta forma de trabajo.

A partir de esta delimitación se reorganizó el proyecto como un repositorio modular. El núcleo se concentra en una sola ubicación, `core/src/m5gp`, con el propósito de evitar copias divergentes entre la ejecución local, el backend y la integración con SRBench. La herramienta `tools/install_core_from_github.py` obtiene la versión indicada del repositorio independiente de M5GP 2.0 y coloca los módulos en esa fuente única, sin alterar los kernels CUDA ni los procedimientos internos del algoritmo. Los scripts de prueba se mantienen en `core/tests`, de modo que el motor pueda ejecutarse y verificarse de manera independiente al laboratorio web. La integración con SRBench se resolvió mediante un adaptador e instalador específico en `integrations/srbench`, sin duplicar permanentemente el motor dentro de GenLab M5GP.

En la siguiente etapa se implementó la capa de servicios con FastAPI. El backend se estructuró en componentes de configuración, persistencia, modelos de datos, seguridad, dependencias, esquemas, rutas, servicios y worker de ejecución. Se habilitaron operaciones para comprobar la salud del servicio, registrar usuarios, iniciar y renovar sesiones, identificar al usuario autenticado, administrar datasets, crear experimentos, consultar recursos GPU, iniciar o repetir ejecuciones y descargar artefactos. La autenticación mediante JWT y el control de propiedad de los recursos se incorporaron para que cada usuario acceda únicamente a sus datasets, experimentos y archivos. La especificación OpenAPI/Swagger se utilizó como mecanismo de revisión del contrato de la API y como apoyo para realizar pruebas directas de los endpoints.

La gestión de datos se desarrolló como un flujo trazable por usuario. La plataforma admite archivos CSV y TSV, valida su extensión, determina el separador correspondiente, registra metadatos y permite obtener una vista previa antes de crear un experimento. Los archivos se almacenan en espacios asociados con el usuario y el backend conserva una ruta resoluble para evitar que el cambio temporal de directorio del worker afecte su localización. En la creación del experimento se registra el dataset, la variable objetivo, el tipo de tarea y la configuración efectiva del algoritmo. Esta información queda asociada con el identificador del experimento y se reutiliza cuando se solicita una reejecución, lo que reduce la posibilidad de introducir cambios no documentados entre corridas.

Para la ejecución computacional se diseñó un mecanismo explícito de administración de GPU. La versión actual reserva una GPU para cada proceso activo de M5GP; en un equipo con varios dispositivos pueden ejecutarse simultáneamente tantas tareas como GPUs disponibles, con una tarea por dispositivo. Cuando todos los recursos están ocupados, la API rechaza la nueva solicitud mediante una respuesta controlada y no la incorpora a una cola de trabajos. El worker selecciona `m5gpRegressor` o `m5gpClassifier` según el tipo de tarea, carga los datos, aplica los parámetros almacenados, asigna el dispositivo, actualiza el estado del experimento y libera el recurso al finalizar o ante una excepción controlada. Este procedimiento evita la sobresuscripción de una GPU y hace observable la condición de disponibilidad para el usuario.

De forma paralela se construyó la interfaz web como una aplicación de una sola página con React, TypeScript y Vite. El frontend consume exclusivamente la API REST, por lo que no ejecuta M5GP ni accede directamente al sistema de archivos. Se implementaron recorridos para registro e inicio de sesión, panel general, administración de datasets, creación e historial de experimentos, consulta de recursos GPU, seguimiento de estados y revisión de resultados. El formulario distingue entre regresión simbólica y clasificación y presenta únicamente los métodos y parámetros válidos para la tarea seleccionada. El panel consulta periódicamente datasets, experimentos y disponibilidad de GPU, mientras que la vista de detalle reúne configuración, métricas, progreso y enlaces de descarga. Esta separación permitió comprobar por separado el comportamiento del frontend, la API y el worker antes de validar su integración completa.

La verificación funcional se realizó mediante recorridos controlados. Se comprobó el arranque del servicio, el registro y autenticación de usuarios, la carga y vista previa de archivos, la creación de experimentos, la consulta de GPUs, el inicio de una ejecución, la transición entre estados y la recuperación de resultados. También se consideraron casos negativos, entre ellos credenciales inválidas, acceso a recursos de otro usuario, archivos no admitidos, parámetros incompatibles y ausencia de una GPU libre. Una ejecución concluida produce artefactos separados, entre ellos `metrics.json`, `model.txt`, `predictions.csv` y `experiment.json`, además de los registros e información de seguimiento correspondientes. La posibilidad de descargar estos archivos y conservar la configuración efectiva constituye una evidencia verificable del flujo completo y facilita su auditoría posterior.

Como apoyo a la instalación y transferencia tecnológica se prepararon archivos de entorno, scripts, documentación y mecanismos de despliegue. El núcleo acelerado utiliza un entorno compatible con CUDA, RAPIDS y cuML; el backend y el frontend pueden instalarse por separado, y la configuración con Docker Compose integra los servicios de base de datos, API y aplicación web. El archivo README documenta la instalación del núcleo, la ejecución local, la integración con SRBench, la prueba de la API, el uso del frontend,

las rutas principales, los artefactos y la atención de problemas frecuentes. Las capturas incorporadas al informe muestran las pantallas desarrolladas y complementan la evidencia técnica del estado alcanzado.

Los resultados obtenidos se expresan, por tanto, en dos niveles. En el nivel tecnológico se dispone de una versión integrada de GenLab M5GP con arquitectura modular, autenticación, gestión de datos, configuración de tareas de regresión y clasificación, control de GPU, seguimiento de experimentos, visualización de resultados y descarga de artefactos, sin eliminar el uso local ni la compatibilidad con SRBench. En el nivel metodológico se establecieron procedimientos de verificación, trazabilidad y diseño experimental que permiten someter el sistema a evaluaciones comparativas posteriores. La implementación funcional no se interpreta por sí sola como evidencia de superioridad predictiva o computacional; dichas conclusiones dependen de las ejecuciones repetidas, métricas y pruebas estadísticas definidas en la sección de metodología.

El trabajo también contribuyó al cumplimiento de las metas de formación y difusión. La participación de estudiantes de residencia profesional y servicio social se articuló con tareas concretas de análisis, programación, documentación, pruebas y divulgación. Esta organización permitió que los productos de formación se vincularan directamente con componentes verificables del proyecto y no solamente con actividades auxiliares. En conjunto, los procedimientos seguidos transformaron el planteamiento inicial en una plataforma documentada y técnicamente evaluable, y dejaron identificadas las actividades pendientes de validación experimental exhaustiva, estabilización operativa y consolidación de productos científicos.

e) Conclusiones/Observaciones:

El estado alcanzado permite concluir que el proyecto evolucionó desde la revisión del algoritmo y la definición conceptual de un laboratorio en línea hacia una implementación modular e integrada. La versión actual no constituye únicamente una maqueta de interfaz: articula un núcleo científico independiente, una API REST, persistencia de datos, autenticación, administración de experimentos, control de recursos GPU y una aplicación web. Esta integración representa un cumplimiento sustantivo del objetivo tecnológico, porque habilita un flujo verificable desde el registro del usuario y la carga del dataset hasta la ejecución de M5GP y la recuperación de métricas, predicciones y modelo simbólico.

Una conclusión técnica relevante es que la decisión de conservar una sola fuente del núcleo reduce el riesgo de divergencia entre modalidades de uso. M5GP continúa disponible para ejecución directa mediante scripts Python y para su instalación en SRBench, mientras que GenLab añade acceso por API y navegador sin sustituir esas formas de trabajo. El desacoplamiento entre core, backend, frontend e integraciones permite revisar, probar o actualizar cada componente con menor interferencia sobre los demás. Asimismo, el instalador del núcleo conserva los procedimientos originales de evaluación, operadores evolutivos, kernels CUDA, predicción y construcción simbólica, lo que favorece la continuidad con la base científica previamente validada.

Desde la perspectiva de accesibilidad, la plataforma reduce la necesidad de que cada usuario configure localmente el entorno completo de M5GP. Los docentes, estudiantes e investigadores pueden interactuar mediante el navegador o la API, mientras que el entorno CUDA/RAPIDS/cuML se concentra en el servidor que ejecuta el worker. Esta separación amplía las posibilidades de uso académico y experimental; sin embargo, no elimina la dependencia tecnológica del motor. El backend y el frontend pueden operar sin GPU para autenticación, navegación y administración de información, pero una ejecución real de regresión simbólica o clasificación continúa requiriendo una GPU NVIDIA y las bibliotecas compatibles.

La administración de GPU constituye al mismo tiempo un resultado y una limitación deliberada de la primera versión. La regla de una tarea activa por GPU evita competencia no controlada por memoria y capacidad de cómputo, y la asignación dinámica permite aprovechar varios dispositivos cuando están disponibles. No obstante, la ausencia de una cola implica que una solicitud es rechazada cuando todos los dispositivos se encuentran ocupados. Esta decisión simplifica la trazabilidad y reduce riesgos operativos

durante la etapa inicial, aunque limita la atención simultánea de usuarios. La incorporación futura de una cola, políticas de prioridad o escalamiento distribuido deberá evaluarse sin comprometer la regla de aislamiento por GPU ni la reproducibilidad de las ejecuciones.

También se concluye que la trazabilidad quedó incorporada como parte central del diseño. La plataforma conserva la relación entre usuario, dataset, variable objetivo, parámetros, semilla, dispositivo, estados, tiempos, métricas, expresión y archivos descargables. Los artefactos `metrics.json`, `model.txt`, `predictions.csv` y `experiment.json` permiten revisar qué se ejecutó y reconstruir el contexto de cada resultado. Esta característica es especialmente importante para investigación y docencia, porque evita que las conclusiones dependan exclusivamente de valores mostrados temporalmente en pantalla y proporciona evidencia que puede integrarse a análisis estadísticos, informes o ejercicios de reproducción.

En cuanto a la evaluación científica, debe distinguirse con claridad entre validación funcional y validación comparativa del algoritmo. Las pruebas de integración permiten afirmar que los módulos se comunican, aplican controles y producen artefactos conforme al flujo documentado. Sin embargo, no son suficientes para afirmar que una configuración de M5GP supera a otra o que la plataforma mejora por sí misma la calidad predictiva del motor. Estas afirmaciones requieren múltiples ejecuciones controladas sobre benchmarks, análisis separado de entrenamiento y prueba, comparación de complejidad y tiempos, y aplicación de las pruebas estadísticas previstas. Mantener esta distinción evita sobreinterpretar el avance tecnológico y preserva el rigor del proyecto.

Como observación operativa, la instalación exige compatibilidad entre sistema, controladores NVIDIA, CUDA, RAPIDS/cuML, Python y dependencias del frontend. La documentación, los archivos de entorno y los contenedores reducen esta complejidad, pero no la eliminan. También deben mantenerse controles sobre rutas de archivos, variables de entorno, CORS, sesiones almacenadas en el navegador y versiones de dependencias, ya que estos elementos pueden afectar el funcionamiento aun cuando el núcleo científico permanezca correcto. La inclusión de procedimientos de diagnóstico en el README y la separación de servicios constituyen medidas apropiadas para localizar fallas y evitar que un problema de interfaz se confunda con un problema del algoritmo.

En el plano académico, el proyecto ha generado un entorno de formación en investigación aplicada y desarrollo de software científico. Las actividades realizadas por los estudiantes participantes se relacionaron con análisis del código, diseño de componentes, pruebas, documentación y difusión, lo cual favorece el desarrollo de competencias técnicas y metodológicas. La publicación y documentación del software también amplían las posibilidades de transferencia, revisión y colaboración con otros usuarios del área, siempre que las versiones, dependencias y condiciones experimentales se mantengan explícitamente registradas.

En síntesis, GenLab M5GP cuenta con una base funcional y arquitectónica coherente con el objetivo de ofrecer acceso web y por API a M5GP, sin perder su uso directo ni su integración con herramientas de evaluación. Los principales riesgos técnicos fueron delimitados y atendidos mediante modularidad, autenticación, control de propiedad, validación de datos, reservación de GPU y generación de artefactos. Las actividades siguientes deben concentrarse en completar la validación experimental exhaustiva, ampliar las pruebas automatizadas y operativas, fortalecer el despliegue en infraestructura GPU y consolidar los productos científicos y de transferencia derivados. Estas tareas representan la continuación natural del proyecto y no corrigen una carencia conceptual, sino que profundizan la evaluación y madurez de una plataforma ya integrada.




2. Objetivo del proyecto

Desarrollar GenLab M5GP, Laboratorio en Línea para Aprendizaje Automático utilizando M5GP, una plataforma en línea que permita la ejecución optimizada de M5GP en tareas de regresión simbólica y clasificación, proporcionando acceso escalable y eficiente para la comunidad académica y científica.

3. Metas: cumplimiento de Metas

Metas cuantificables	Cantidad programada	Cantidad lograda	Observaciones
Incorporación de estudiantes de licenciatura al proyecto, créditos complementarios:			
Incorporación de estudiantes de licenciatura al proyecto, servicio social:	2	2	Se termino servicio social
Estudiantes con residencia concluida	1	1	Se concluyo residencia.
Tesis en desarrollo de Licenciatura			
Tesis concluida de Licenciatura			
Tesis en desarrollo de Maestría			
Tesis concluida de Maestría			
Tesis en desarrollo de Doctorado			
Tesis concluida de Doctorado			
Artículos científicos publicados en revistas indizadas enviados	1	1	Articulo enviado
Artículos científicos en revistas arbitradas enviados	1	1	Articulo enviado
Artículos de divulgación enviados			
Artículos científicos en revistas indizadas publicados	1	1	Articulo publicado
Artículos científicos en revistas arbitradas publicados			
Artículos de divulgación publicados			
Memorias en extenso en congresos			
Capítulos de libros enviados para revisión			
Libros enviados para revisión			
Libros editados y publicados			
Registro de Patente (IMPI)			
Registro de Modelo de Utilidad (IMPI)			
Registro de Signos Distintivos (IMPI)			

Registro de Diseño Industrial (IMPI)			
Registro de Esquemas de Trazado de Circuitos Integrados (IMPI):			
Registro de Obra (INDAUTOR):			
Carta de Usuario (Empresa):			

Metas cumplidas

Incorporación de estudiantes de licenciatura al proyecto, servicio social:	
Fecha	Nombre de los estudiantes incorporados y su Programa Educativo
2025-1, 2025-2	Cabrera-Rodríguez, Eric
2025-1, 2025-2	Escalante -Vertti , Ernesto
Estudiantes residentes participantes en el proyecto	
Fecha	Nombre de los estudiantes residentes, nombre del proyecto de residencia y su Programa Educativo
16/octubre/2025 - 16/enero/2026	Cota-Lopez, Pablo-Isai;




4. Metodología: explique el o los procedimientos científico-metodológicos que se siguieron para cumplir los objetivos y metas del proyecto, indicando las pruebas estadísticas, diseño experimental y técnicas utilizadas o previstas (máximo tres cuartillas).

Enfoque científico-metodológico

El proyecto GenLab M5GP se desarrolló como una investigación aplicada con desarrollo experimental de software y validación cuantitativa. El procedimiento metodológico integró cuatro componentes: análisis de la línea base M5GP 2.0; diseño e implementación incremental de una plataforma modular; verificación funcional y técnica de los servicios; y definición de un diseño experimental para evaluar desempeño predictivo, complejidad simbólica, estabilidad y costo computacional. La estrategia fue iterativa: cada componente se revisó, implementó, integró y comprobó antes de incorporar el siguiente, manteniendo trazabilidad entre los objetivos del proyecto, los módulos desarrollados y las evidencias generadas.

Para preservar el rigor del informe, se diferencian dos niveles de evidencia:

1. El primero corresponde a actividades ya realizadas y verificables en el código, el archivo README.md, la documentación de la API y los artefactos de ejecución.
2. El segundo corresponde a la validación estadística y de ejecución prevista para la fase experimental.

Procedimiento científico y técnico seguido

1. Análisis de la línea base y delimitación de requisitos.

Se revisaron la estructura, dependencias, flujo evolutivo y formas de ejecución del núcleo M5GP 2.0, incluyendo su uso directo mediante scripts Python y su integración local con SRBench. Se identificaron como restricciones técnicas la dependencia del núcleo respecto de GPU NVIDIA, CUDA y RAPIDS/cuML, así como la necesidad de conservar sin pérdida los operadores, kernels CUDA, evaluación de individuos, selección, UMAP, supervivencia, reemplazo, predicción y construcción simbólica. También se estableció que backend, frontend y API pueden operar sin GPU para autenticación, navegación y administración de datos, pero la ejecución de M5GP requiere un dispositivo compatible.

2. Diseño arquitectónico y organización del repositorio.

Se adoptó una arquitectura tipo repositorio con separación entre core, integrations/srbench, backend, frontend, tools, docs, data e infraestructura de despliegue. El criterio metodológico fue desacoplar responsabilidades sin duplicar el motor científico. Por ello, el código de M5GP se mantiene en una sola fuente ubicada en core/src/m5gp y se obtiene mediante la herramienta de instalación documentada, en lugar de incorporar permanentemente una copia del repositorio original. Esta decisión permite actualizar o auditar el núcleo de manera independiente, conservar el uso local por terminal y evitar divergencias entre la versión utilizada por la web, la API y SRBench.

3. Implementación y validación de la capa de servicios.

El backend se implementó con FastAPI. Los archivos main.py, config.py, db.py, models.py, security.py y deps.py concentran arranque, configuración, persistencia, modelos, autenticación y dependencias; los routers exponen los endpoints y los services junto con worker.py coordinan almacenamiento, experimentos, recursos GPU y ejecución del núcleo. La autenticación utiliza contraseñas cifradas, tokens JWT de acceso y renovación, y control de propiedad para impedir que un usuario consulte datasets, experimentos o archivos de otro usuario. Los esquemas de entrada y salida validan parámetros antes de invocar el worker y la especificación OpenAPI/Swagger se utiliza para revisar el contrato de la API.

4. Implementación de la capa de acceso.

La interfaz se construyó con React, TypeScript y Vite. Los archivos `main.tsx`, `components.tsx`, `pages.tsx`, `charts.tsx`, `i18n.tsx` y `styles.css` organizan inicialización, componentes reutilizables, páginas funcionales, visualizaciones, traducción y presentación. El frontend consume exclusivamente la API REST y no accede de forma directa al sistema de archivos ni al núcleo. Los recorridos de usuario cubren registro e inicio de sesión, carga y vista previa de CSV/TSV, selección de dataset y variable objetivo, configuración de regresión o clasificación, consulta de GPU, seguimiento del estado y descarga de resultados. Esta separación permitió probar la interfaz y el backend de manera independiente antes de validar el flujo integrado.

5. Control de ejecución, asignación de GPU y trazabilidad.

La primera versión aplica la regla de un proceso M5GP activo por GPU. En equipos con varios dispositivos, la asignación es dinámica y admite una ejecución concurrente por GPU; cuando todos están ocupados, la solicitud se rechaza mediante HTTP 409 y no se incorpora una cola de trabajos no documentada. El worker carga el dataset, identifica los predictores numéricos, aplica la configuración del experimento, reserva el dispositivo, ejecuta `m5gpRegressor` o `m5gpClassifier`, actualiza estado e historial y libera la GPU al terminar o ante un error controlado. Cada ejecución conserva identificador, usuario propietario, configuración efectiva, semilla, tiempos, métricas, expresión, modelo, predicciones, historial por generación y archivos de salida en un directorio independiente.

6. Verificación funcional y de integración.

Se definieron casos de prueba positivos y negativos para comprobar salud del servicio; registro, inicio, renovación y cierre de sesión; autorización por propietario; carga y vista previa de archivos; validación de parámetros; creación de experimentos; consulta de recursos; transición de estados; rechazo por GPU ocupada; manejo de excepciones; lectura de métricas y descarga de artefactos. La comprobación se apoyó en los endpoints documentados, la interfaz OpenAPI/Swagger, los scripts y comandos del README.md y la revisión de los archivos producidos. En la integración completa se verificó que la petición iniciada en el navegador atravesara frontend, API, persistencia, reservación de GPU y worker, y regresara al usuario como estado, métricas y archivos descargables.

7. Gestión de configuración y reproducibilidad.

Se utilizaron control de versiones, variables de entorno, archivos de configuración, entornos Conda para el núcleo acelerado y contenedores para los componentes de despliegue. La reproducibilidad se basa en registrar la versión de GenLab M5GP, la versión del núcleo, parámetros, semilla, partición de datos, dispositivo asignado y artefactos resultantes. La configuración se conserva junto con el experimento para que una ejecución pueda reconstruirse o compararse posteriormente. No se añadió un comando CLI propio: el uso local se mantiene mediante scripts Python, conforme al patrón de pruebas existente, y la plataforma web actúa como una capa adicional de acceso.

Diseño experimental y variables de estudio

La evaluación científica se estructuró como un diseño comparativo de mediciones repetidas y reproducibles. La unidad experimental es una ejecución completa de M5GP sobre un conjunto de datos, una configuración y una semilla determinadas. El factor de comparación está constituido por la configuración algorítmica o la modalidad de tarea, mientras que el conjunto de datos, la partición entrenamiento-prueba y las semillas se mantienen iguales entre configuraciones para producir observaciones pareadas. Este diseño reduce la variación atribuible a particiones distintas y permite estimar con mayor claridad el efecto de los parámetros o extensiones evaluadas.

Las variables independientes consideradas son: tipo de tarea —regresión simbólica o clasificación—, método de evaluación, conjunto de operadores, número de generaciones, tamaño de población, longitud del individuo, probabilidades de generación de operadores y variables, probabilidades de mutación y eliminación, tamaño de torneo y semilla aleatoria. Las variables dependientes son el desempeño predictivo en entrenamiento y prueba, complejidad de la expresión, tiempo de proceso, tiempo de pared, estabilidad entre repeticiones, comportamiento del ajuste por generación y estado final de la ejecución. Como variables de control se registran versión del software, conjunto de datos, variable objetivo, partición, entorno de ejecución y GPU asignada.

La implementación actual utiliza una partición de 75 % para entrenamiento y 25 % para prueba, con semilla configurable y valor predeterminado 42. Para la comparación definitiva se realizarán varias ejecuciones independientes por configuración, conservando el mismo conjunto de semillas y la misma partición para todas las alternativas comparadas. Los resultados de entrenamiento y prueba se analizarán por separado; el desempeño de entrenamiento se utilizará para examinar ajuste y convergencia, mientras que el conjunto de prueba sustentará la valoración de generalización. La plataforma almacenará cada repetición como un experimento trazable, evitando sobrescribir configuraciones o resultados.

Métricas, pruebas estadísticas y criterios de análisis

- **Regresión:** se registrarán error cuadrático medio (MSE), raíz del error cuadrático medio (RMSE), error absoluto medio (MAE) y coeficiente de determinación (R^2). Estas métricas se complementarán con el tamaño o número de nodos de la expresión simbólica y con tiempos de ejecución, para analizar conjuntamente precisión, interpretabilidad y costo computacional.
- **Clasificación:** se registrarán exactitud, precisión macro, exhaustividad macro y F1 macro. Cuando el worker produzca la información correspondiente, se conservarán además el reporte por clase y la matriz de confusión, con el fin de evitar que una sola métrica global oculte comportamientos diferentes entre clases.
- **Seguimiento evolutivo y operación:** se conservarán el valor de ajuste por generación, generación actual y total, estados de la ejecución, tiempos de proceso y pared, dispositivo asignado y condición de terminación. El porcentaje mostrado por la interfaz representa etapas verificables del flujo y no una estimación lineal del tiempo restante.
- **Análisis descriptivo y comparación estadística:** para cada configuración se utilizarán mediana, rango intercuartílico y medidas de dispersión, además de tablas de resultados y rangos por conjunto de datos. En comparaciones pareadas de dos configuraciones se empleará la prueba no paramétrica de rangos con signo de Wilcoxon, cuya hipótesis nula establece que la distribución de las diferencias pareadas está centrada en cero. Cuando se efectúen múltiples contrastes, los valores p se ajustarán mediante el procedimiento de Holm. Se reportarán dirección y magnitud observable de las diferencias, métricas de desempeño, complejidad y costo, y no únicamente significancia estadística. Si la cantidad o calidad de observaciones no permite un contraste válido, el análisis se limitará a estadísticos descriptivos sin forzar conclusiones inferenciales.

Técnicas de aseguramiento de calidad y criterios de aceptación

Durante el desarrollo se aplicaron prácticas orientadas a mantener la calidad, consistencia y trazabilidad del sistema y sus resultados. Entre ellas se incluyeron la separación de responsabilidades entre componentes, la validación de datos y parámetros de entrada, el control de versiones, el manejo externo de la configuración, la autenticación de usuarios, el registro de eventos y el almacenamiento estructurado de la información y de los archivos generados durante cada experimento.

Como criterios de aceptación se estableció conservar una única implementación del núcleo de M5GP y mantener su ejecución local mediante scripts Python. También se verificó que los usuarios únicamente pudieran acceder a sus propios datos y experimentos, que los conjuntos de datos y parámetros fueran revisados antes de iniciar una ejecución y que una solicitud no se pusiera en marcha cuando no hubiera una GPU disponible. De igual forma, se consideró necesario liberar los recursos asignados tanto al finalizar correctamente un experimento como cuando ocurriera algún error durante su ejecución.

Otro criterio se consideró, importante fue conservar el seguimiento de cada experimento desde su configuración inicial hasta los resultados obtenidos. Para ello se relacionaron la solicitud de ejecución, los parámetros utilizados, el estado del proceso, las métricas calculadas, la expresión del modelo obtenido y los archivos generados.

En conjunto, estas prácticas permitieron verificar el funcionamiento de los principales componentes y del flujo de operación de la plataforma. Esta verificación corresponde principalmente a aspectos funcionales y de integración del sistema. La evaluación comparativa del desempeño de la plataforma y del núcleo de M5GP requiere, por su parte, experimentos controlados y análisis estadísticos específicos. Por esta razón, en el presente reporte únicamente se documentan los resultados efectivamente obtenidos y no se presentan conclusiones estadísticas sobre pruebas que todavía no han sido realizadas de manera completa.

5. Difusión

Las actividades de difusión del proyecto se desarrollaron por dos vías principales: la divulgación académica y la transferencia técnica de los resultados mediante documentación y publicación del software. Durante el periodo se mantuvieron las evidencias previamente reportadas, entre ellas la participación en cuatro conferencias dirigidas principalmente a públicos académicos y estudiantiles, así como la presentación de una ponencia en un congreso organizado por IEEE en Tijuana.

En estos espacios se presentaron los fundamentos de M5GP, las ventajas de trabajar con modelos interpretables y los avances del proyecto orientados al desarrollo de un laboratorio en línea para apoyar actividades de docencia e investigación. Estas acciones permitieron dar a conocer el trabajo realizado y acercar sus resultados a estudiantes, docentes e investigadores interesados en el uso de herramientas de aprendizaje automático y programación genética.

Durante el periodo actualizado se incorporó como producto científico relacionado la publicación del artículo "M5GP 2.0: Extensions and Enhancements to a Multidimensional Genetic Programming System", en la revista Expert Systems, con DOI 10.1111/exsy.70356. Este artículo documenta las mejoras algorítmicas que constituyen la base científica y computacional utilizada por GenLab M5GP. La mención se limita al producto verificable y no supone que el artículo describa la totalidad de la plataforma web.

La difusión tecnológica se fortaleció mediante el repositorio oficial GenLab-m5gp y su archivo README.md (<https://github.com/armandocardenasf/GenLab-m5gp>). La documentación describe la instalación del núcleo desde su repositorio independiente, el uso directo con scripts Python, la integración con SRBench, la instalación del backend, el flujo de prueba de la API, el arranque del frontend, el despliegue con Docker Compose, la solución de problemas y el acceso a artefactos. La API agrega documentación interactiva OpenAPI/Swagger, lo que permite consultar y probar los endpoints sin depender exclusivamente de documentación narrativa.

El interfaz bilingüe español/inglés y la página "Acerca de" complementan la transferencia del conocimiento. Esta última consulta un endpoint público que presenta versión, autoría, derechos, instituciones de apoyo, agradecimientos, referencias de M5GP y M5GP 2.0, enlaces de código fuente y aviso legal. En conjunto, estos mecanismos favorecen la reproducibilidad, el aprendizaje autónomo y la incorporación de estudiantes e investigadores al uso de la plataforma.

Medio de difusión	Evidencia o alcance actualizado	Público principal
Conferencias de divulgación	Cuatro participaciones previamente reportadas; exposición de fundamentos, motivación y aplicaciones de M5GP.	Estudiantes, docentes y comunidad académica.
Congreso IEEE en Tijuana	Una ponencia previamente reportada, vinculada con la aplicación de M5GP en investigación educativa.	Investigadores y profesionales.
Artículo científico	Publicación en Expert Systems: DOI 10.1111/exsy.70356.	Comunidad de programación genética y aprendizaje automático.
Repositorio y README.md	Código y documentación de instalación, arquitectura, API, frontend, pruebas, despliegue y solución de problemas.	Desarrolladores, estudiantes e investigadores.
OpenAPI/Swagger y página "Acerca de"	Consulta interactiva de servicios y comunicación pública de versión, autoría, referencias y acceso al código.	Usuarios técnicos y académicos.

6. Beneficios y Problemas

Beneficios obtenidos

- **Acceso tecnológico ampliado.** GenLab M5GP separa la experiencia de uso de la infraestructura especializada: el usuario puede autenticar su cuenta, cargar datos, configurar experimentos y consultar resultados desde el navegador, mientras la ejecución computacional permanece en el servidor con GPU.
- **Conservación de las capacidades existentes.** El núcleo mantiene su ejecución directa con scripts Python y su integración con SRBench. La plataforma no sustituye el uso local ni introduce un comando CLI propio; agrega formas de acceso sin eliminar las anteriores.
- **Arquitectura mantenible y trazable.** La fuente única en core/src/m5gp evita copias divergentes. Backend, frontend e integraciones se encuentran desacoplados, con responsabilidades y archivos identificables, lo que facilita mantenimiento, pruebas y evolución independiente.
- **Seguridad y aislamiento.** La autenticación JWT, las sesiones de renovación y el control de propiedad restringen el acceso a datasets, experimentos y artefactos del usuario correspondiente.
- **Uso responsable de GPU.** El mecanismo de reservación impide que dos procesos utilicen simultáneamente el mismo dispositivo. La asignación multi-GPU permite paralelismo controlado cuando existe más de una GPU.
- **Reproducibilidad experimental.** La configuración efectiva, semilla, métricas, tiempos, expresión, predicciones e historial quedan almacenados en archivos descargables, lo que facilita auditoría y comparación posterior.
- **Utilidad académica.** La interfaz bilingüe, la vista previa de datasets, los formularios explícitos y la documentación paso a paso reducen la barrera de entrada para prácticas docentes y proyectos de investigación.

Problemas, limitaciones y medidas adoptadas

Problema o limitación	Efecto en el proyecto	Tratamiento actual
Dependencia del núcleo respecto de GPU, CUDA y RAPIDS/cuML.	La ejecución de modelos no puede realizarse en equipos sin un entorno GPU compatible.	Se mantuvo la dependencia como requisito explícito del núcleo. Backend y frontend pueden probarse sin GPU, pero la ejecución se rechaza de forma controlada.
Una tarea activa por GPU y ausencia de cola en la versión inicial.	Una solicitud adicional no inicia cuando todas las GPU están ocupadas.	El backend reserva un dispositivo por proceso y responde HTTP 409. La interfaz comunica que el recurso está ocupado; la cola queda fuera del alcance de la primera versión.
Compatibilidad de versiones del entorno.	Cambios en CUDA, RAPIDS, Python, Node.js, Vite o paquetes npm pueden impedir instalación o compilación.	README.md documenta versiones, variables de entorno, reinstalación de dependencias y procedimientos de diagnóstico. El núcleo se instala de forma separada para reducir acoplamiento.
Duración variable de los algoritmos evolutivos.	El tiempo restante no puede inferirse linealmente a partir de la generación actual.	El worker reporta etapas y avance por generación. La documentación aclara que el porcentaje es informativo y no una predicción exacta de duración.
Datos no adecuados para el estimador.	Columnas no numéricas, objetivo inválido, valores faltantes o formatos incorrectos pueden impedir el entrenamiento.	La plataforma valida extensión CSV/TSV, delimitador, columnas y parámetros; conserva errores y registros para diagnóstico. La preparación final de los datos sigue siendo responsabilidad del experimento.
Validación comparativa aún no exhaustiva para todas las configuraciones de la plataforma.	No es posible atribuir a la interfaz web una mejora estadística del algoritmo sin ejecutar el plan completo.	Se separó la verificación funcional del software de la evaluación científica del algoritmo y se definió el esquema de repeticiones, métricas y pruebas estadísticas para la fase de cierre.
Seguridad de producción.	Secretos predeterminados o políticas permisivas son inadecuados para una exposición pública.	La configuración se externaliza en .env; para producción deben sustituirse secretos, restringirse CORS, utilizar HTTPS y aplicar políticas institucionales de respaldo y acceso.

7. Información adicional

7.1 Estado técnico y alcance de la versión

La versión documentada de la plataforma es GenLab M5GP 1.0.0, y utiliza M5GP Core 2.0.0. El alcance funcional incluye autenticación, sesiones, administración de datasets por usuario, creación de experimentos de regresión simbólica y clasificación, reservación de GPU, monitoreo del estado, consulta de métricas, visualización de predicciones y descarga de artefactos. El núcleo continúa siendo GPU-only en su diseño actual; las capas de backend y frontend no requieren GPU para sus funciones administrativas, pero dependen de un worker con entorno CUDA/RAPIDS/cuML para ejecutar M5GP.

El repositorio GenLab-m5gp no incorpora de forma permanente los archivos fuente del repositorio original de M5GP 2.0. La herramienta tools/install_core_from_github.py obtiene la referencia solicitada y coloca una sola copia en core/src/m5gp. Esta decisión conserva la independencia del núcleo, facilita fijar un tag o commit y evita mantener duplicados dentro de la plataforma.

7.2 Arquitectura funcional

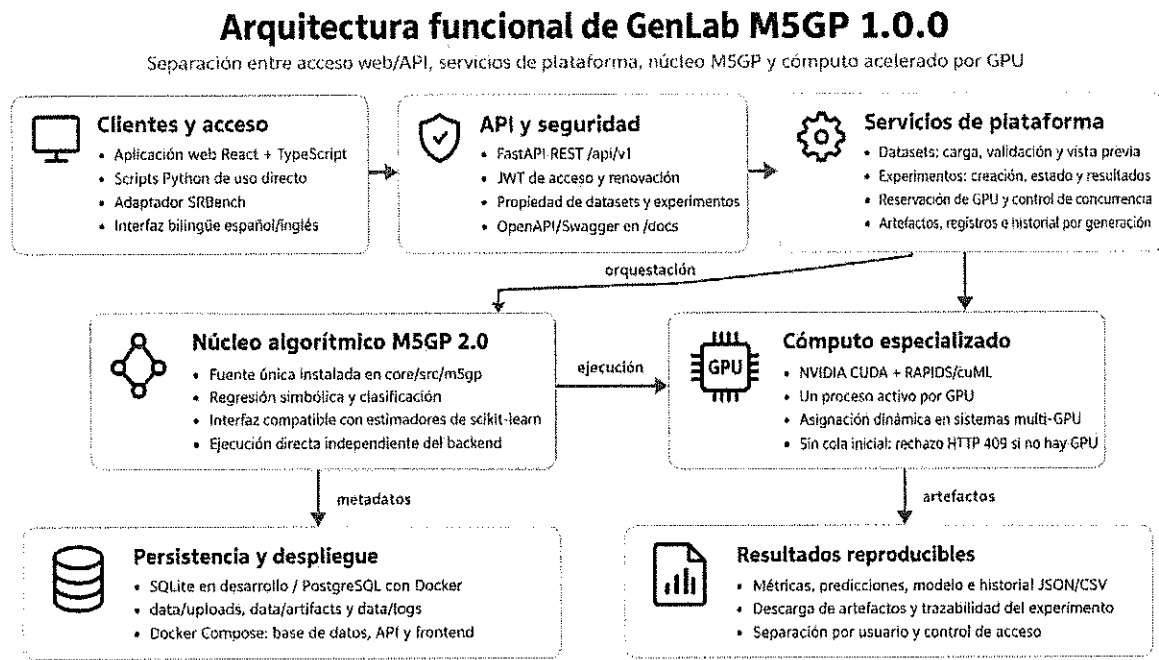


Figura 1. Arquitectura funcional de GenLab M5GP 1.0.0. Fuente: elaboración propia con ayuda de IA para su diseño y con base en documento README.md y los archivos de configuración e implementación del repositorio.

7.2 Modulos y archivos que conforma el proyecto

Módulo	Archivos o carpetas principales	Funcionalidad
Raíz	README.md; .env/.env.example; docker-compose.yml	Documentación integral, configuración del sistema y despliegue coordinado de PostgreSQL, API y frontend.
core/	src/m5gp/; tests/m5gp_Test.py; pyproject.toml	Contiene la única copia instalada del motor M5GP 2.0, sus pruebas directas y el paquete m5gp-core. Permite ejecutar python m5gp_Test.py sin backend ni autenticación.
integrations/srbench/	install.sh	Instala el adaptador m5gpRegressor y la fuente requerida dentro de SRBench v2, fijando una referencia Git cuando sea necesario.

Módulo	Archivos o carpetas principales	Funcionalidad
backend/src/genlab_api/	main.py; config.py; db.py; models.py; schemas.py; security.py; deps.py	Inicializa FastAPI, carga configuración, gestiona persistencia, modelos, validación de datos, cifrado de contraseñas, tokens JWT y dependencias de autorización.
backend/.../routers/	about.py; auth.py; datasets.py; experiments.py; gpus.py	Expone endpoints públicos y protegidos para información del proyecto, cuentas/sesiones, datos, experimentos, artefactos y estado de recursos GPU.
backend/.../services/	files.py; gpu.py; experiments.py	Implementa almacenamiento seguro, detección/reservación/liberación de GPU y lógica de creación, lanzamiento, reejecución, cancelación y eliminación de experimentos.
backend/src/genlab_api/	worker.py	Carga datos, divide entrenamiento/prueba, construye el estimador M5GP, ejecuta fit/predict, calcula métricas, actualiza progreso y genera artefactos.
frontend/src/	main.tsx; components.tsx; pages.tsx; charts.tsx; i18n.tsx; styles.css	Inicia la aplicación; define navegación y componentes; implementa páginas y formularios; dibuja gráficas SVG; gestiona traducciones y apariencia.
frontend/	package.json	Declara la aplicación web y sus dependencias de React, TypeScript y Vite.
tools/	install_core_from_github.py; check_github_repository.py	Instala el núcleo desde GitHub y verifica el estado/estructura esperada del repositorio.
data/	uploads/; artifacts/; logs/	Almacena datasets, resultados descargables y registros de ejecución, separados por usuario y experimento.
docs/ e infra/	GITHUB_PUBLISHING.md y recursos de infraestructura	Concentran lineamientos de publicación y archivos auxiliares de despliegue.

7.3 Flujo funcional del sistema

A continuación se presenta la secuencia de operaciones que sigue GenLab M5GP desde el acceso del usuario hasta la consulta de los resultados obtenidos. El flujo integra las funciones de autenticación, gestión de conjuntos de datos, configuración y ejecución de experimentos, asignación de recursos GPU, procesamiento mediante M5GP y almacenamiento de resultados. Esta secuencia permite visualizar de manera conjunta la interacción entre la interfaz web, la API, el backend, el núcleo de cómputo y los mecanismos de persistencia que intervienen durante una ejecución.

1. El usuario crea una cuenta o inicia sesión. El backend verifica la contraseña cifrada y entrega tokens de acceso y renovación; la interfaz conserva temporalmente la sesión en el navegador.
2. El usuario carga un archivo CSV o TSV. El servicio valida formato, tamaño y delimitador, almacena el archivo dentro del espacio del usuario y devuelve columnas y una vista previa de hasta 20 filas.
3. En "Nuevo experimento" se seleccionan dataset, variable objetivo, tipo de tarea y parámetros de M5GP. El frontend crea el registro y solicita su ejecución mediante la API.
4. El backend comprueba propiedad y estado, reserva una GPU libre y lanza el worker. Si ninguna GPU está disponible, responde HTTP 409 y no crea una cola oculta.
5. El worker carga los datos, selecciona predictores numéricos, realiza la partición entrenamiento/prueba, instancia m5gpRegressor o m5gpClassifier, ejecuta el ciclo evolutivo y actualiza progreso e historial.
6. Al concluir, el sistema calcula métricas, libera la GPU y almacena modelo, predicciones, métricas, expresión, configuración, historial y resultados de prueba. Para clasificación agrega reporte por clase y matriz de confusión.
7. La página de detalle consulta estado, métricas, gráfica por generación, comparación real-predicción y artefactos. Las descargas se autorizan con el JWT del usuario.

Flujo funcional de GenLab M5GP

Desde el acceso del usuario hasta la obtención de resultados

GenLab M5GP 1.0.0
Aprendizaje automático con programación genética
vía web y GPU

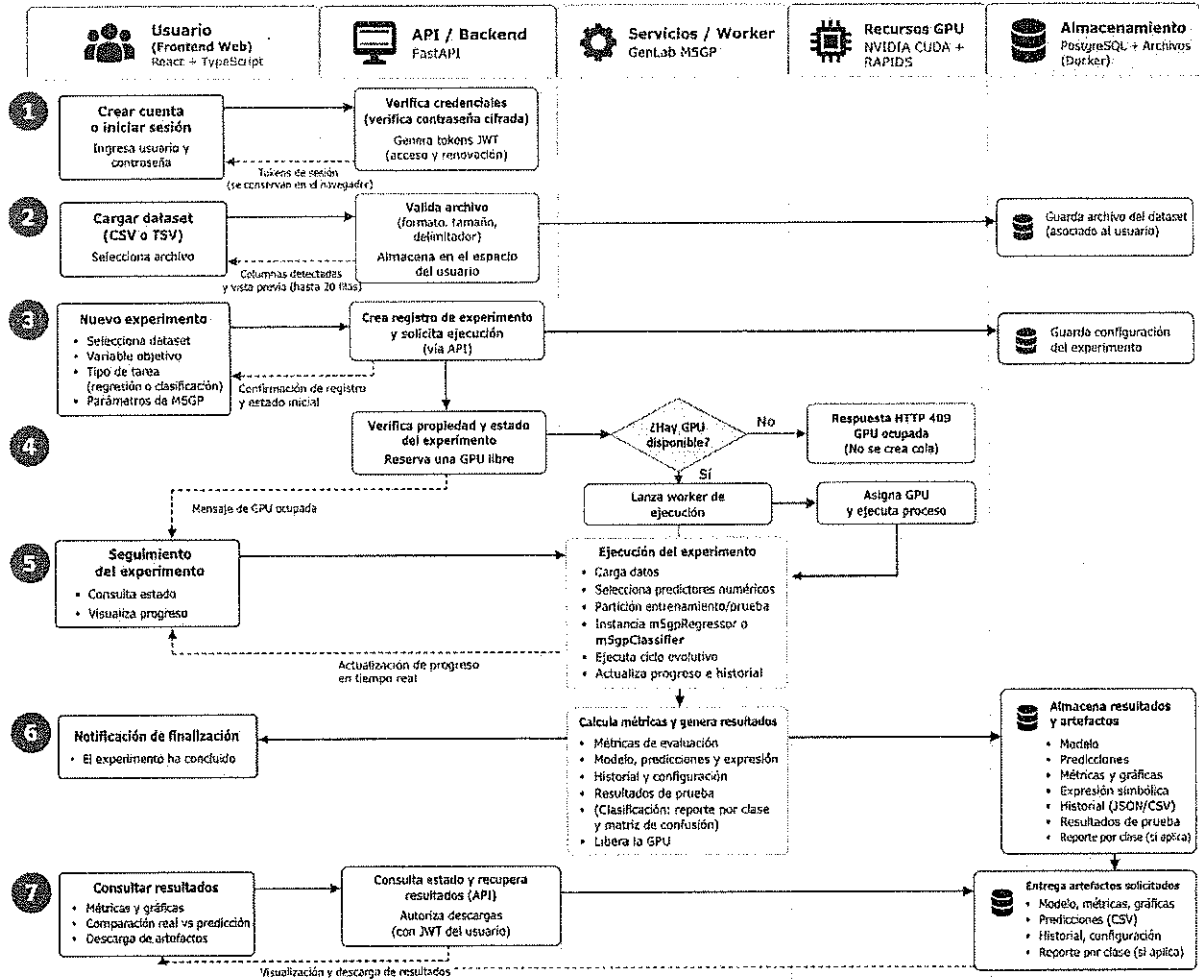


Figura 2. Flujo funcional de GenLab M5GP 1.0.0. Fuente: elaboración propia con ayuda de IA para su diseño y con base en documento README.md y los archivos de configuración e implementación del repositorio.

7.4 Archivos de salida

Archivo	Contenido y utilidad
model.joblib	Objeto del modelo entrenado para conservación o reutilización en un entorno compatible.
predictions.csv	Valores reales y predicciones del conjunto de evaluación.
metrics.json	Métricas de entrenamiento y prueba apropiadas para regresión o clasificación.
model.txt	Representación textual de la expresión o modelo obtenido.
generation_history.json	Historial de ajuste por generación utilizado por la gráfica de evolución.
test_results.json	Resultados reales y predichos en formato JSON para visualización y descarga.
experiment.json	Configuración efectiva, semilla, tiempos, ruido, complejidad, expresión y resultados completos.
classification_report.json y confusion_matrix.json	Archivos adicionales para experimentos de clasificación.

7.5 Capturas de pantalla de la versión documentada

Las siguientes imágenes documentan la interfaz actual con estados de demostración contruidos a partir de sus páginas, etiquetas y estilos. No representan resultados científicos ni sustituyen una evidencia de ejecución en un servidor institucional con GPU; su propósito es mostrar la funcionalidad y organización visual de la versión 1.0.0.

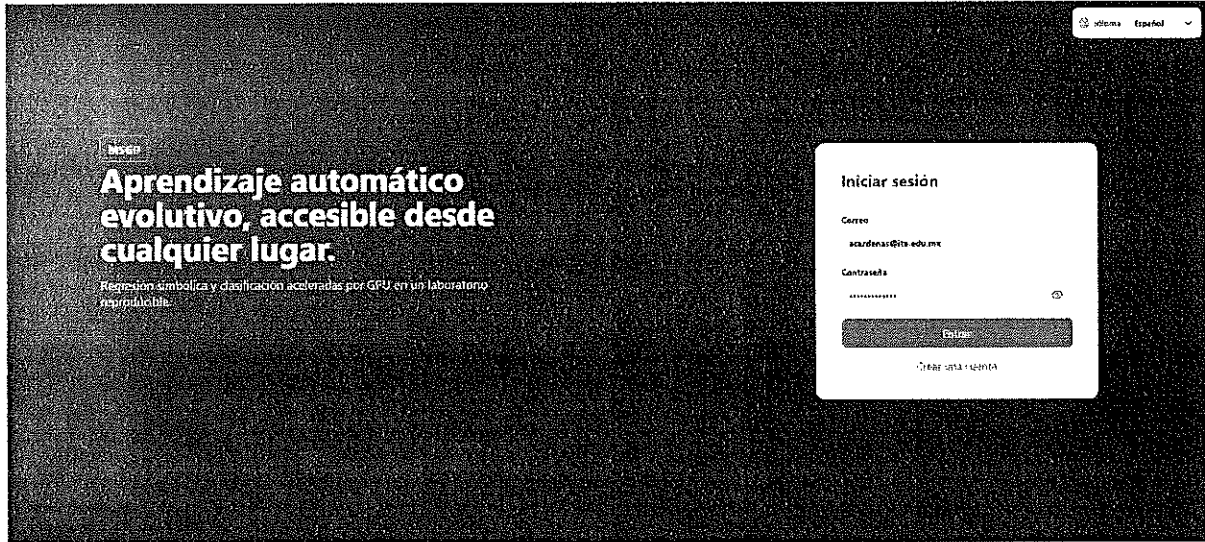


Figura 3. Pantalla de acceso con autenticación de usuario y control de sesión.

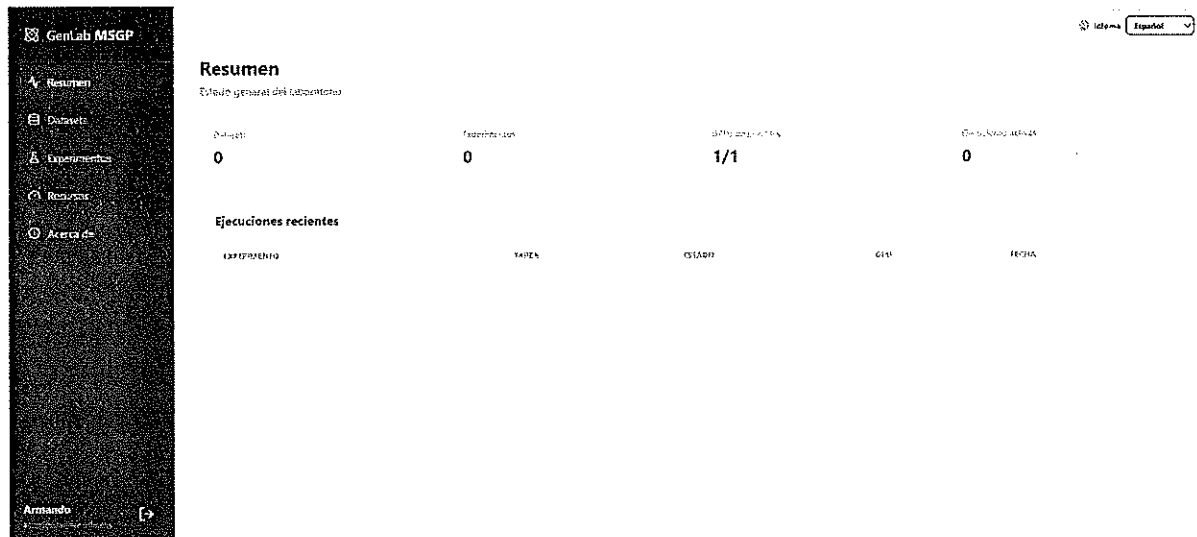


Figura 4. Resumen del laboratorio con indicadores de datasets, experimentos, GPU disponibles y ejecuciones activas.

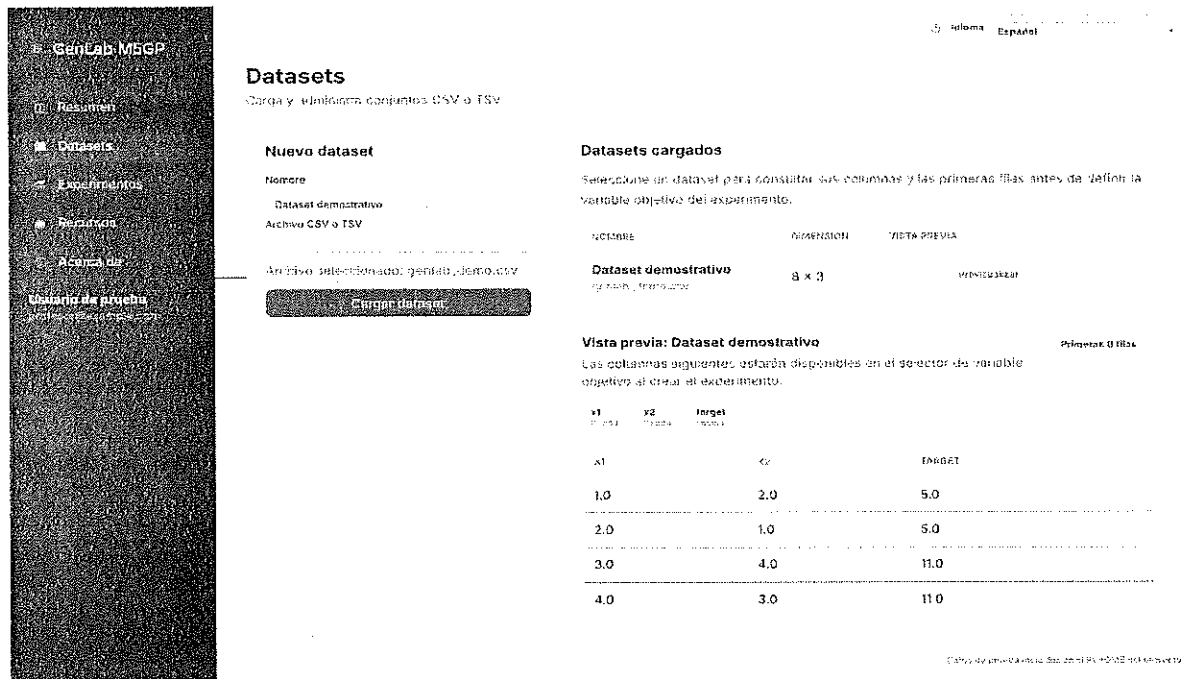


Figura 5. Módulo de datasets con carga CSV/TSV, listado de archivos y vista previa de columnas y registros.



Figura 6. Formulario de nuevo experimento de regresión simbólica con parámetros predeterminados documentados en frontend/src/pages.tsx.

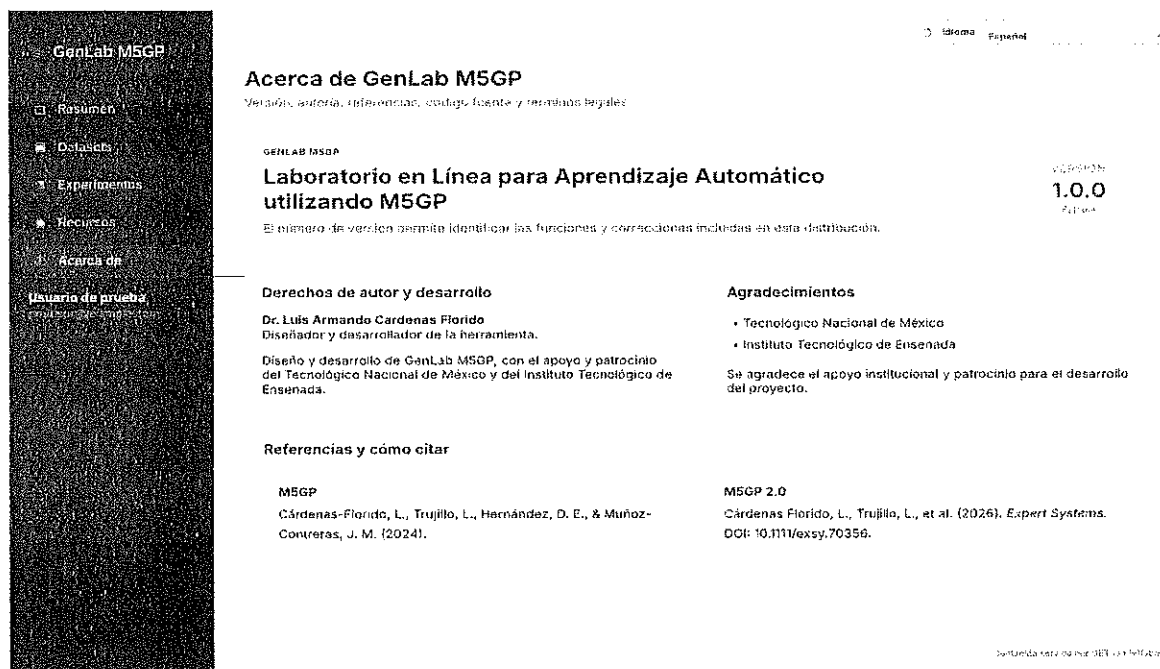


Figura 7. Página “Acerca de”, alimentada por el endpoint público /api/v1/about, con versión, autoría, referencias y acceso al código fuente.

7.6 Fuentes técnicas verificadas

- README.md del repositorio GenLab-m5gp: estructura definitiva, instalación, uso directo, API, frontend, sesiones, artefactos, despliegue y solución de problemas.
- Archivos del backend: main.py, config.py, db.py, models.py, security.py, deps.py, routers, services y worker.py.
- Archivos del frontend: package.json, main.tsx, components.tsx, pages.tsx, charts.tsx, i18n.tsx y styles.css.
- Archivos de empaquetado y despliegue: core/pyproject.toml, backend/pyproject.toml y docker-compose.yml.
- Artículo publicado de M5GP 2.0: DOI 10.1111/exsy.70356.
- El código fuente del proyecto puede ser accedido y descargado desde el repositorio: <https://github.com/armandocardenasf/GenLab-m5gp>

7.7 Referencias bibliográficas

Cárdenas Florido, L., Trujillo, L., Carmona Troyo, J., Cabrera Rodriguez, E., & Cota Lopez, P. (2026). M5GP 2.0: Extensions and enhancements to a constructive feature induction system based on genetic programming. *Expert Systems*, 43(8), e70356. <https://doi.org/10.1111/exsy.70356>

Cárdenas Florido, L., Trujillo, L., Hernandez, D. E., & Muñoz Contreras, J. M. (2024). M5GP: Parallel multidimensional genetic programming with multidimensional populations for symbolic regression. *Mathematical and Computational Applications*, 29(2), Article 25. <https://doi.org/10.3390/mca29020025>

Kune, R., Konugurthi, P. K., Agarwal, A., Chillarige, R. R., & Buyya, R. (2016). The anatomy of big data computing. *Software: Practice and Experience*, 46(1), 79–105. <https://doi.org/10.1002/spe.2374>

Langdon, W. B. (2011). Graphics processing units and genetic programming: An overview. *Soft Computing*, 15(8), 1657–1669. <https://doi.org/10.1007/s00500-011-0695-2>

Letowski, B., Lavayssière, C., Larroque, B., Schröder, M., & Luthon, F. (2020). A fully open source remote laboratory for practical learning. *Electronics*, 9(11), Article 1832. <https://doi.org/10.3390/electronics9111832>

May, D., Jahnke, I., & Moore, S. (2023). Online laboratories and virtual experimentation in higher education from a sociotechnical-pedagogical design perspective. *Journal of Computing in Higher Education*, 35, 203–222. <https://doi.org/10.1007/s12528-023-09380-3>

Pineau, J., Vincent-Lamarre, P., Sinha, K., Larivière, V., Beygelzimer, A., d'Alché-Buc, F., Fox, E., & Larochelle, H. (2021). Improving reproducibility in machine learning research: A report from the NeurIPS 2019 reproducibility program. *Journal of Machine Learning Research*, 22(164), 1–20. <https://doi.org/10.5555/3546258.3546422>

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5), 206–215. <https://doi.org/10.1038/s42256-019-0048-x>

Trujillo, L., Muñoz Contreras, J. M., Hernandez, D. E., Castelli, M., & Tapia, J. J. (2022). GSGP-CUDA—A CUDA framework for geometric semantic genetic programming. *SoftwareX*, 18, Article 101085. <https://doi.org/10.1016/j.softx.2022.101085>

Los anexos administrativos indicados en el formato original —reporte de similitud y carta de reconocimiento de derechos a favor del TecNM— se conservan al final del documento sin modificación. Su integración como evidencia firmada corresponde al proceso institucional de entrega.

Subdirector Académico		Docente
		
MIGUEL ANGEL SIDON AYALA	SECRETARIA DE EDUCACION PUBLICA Instituto Tecnológico de Ensenada	LUIS ARMANDO CARDENAS FLORIDO
Nombre y Firma		Nombre y Firma

Sub-Dirección Aca

- Incluir reporte de software especializado en el que se muestre que **no se tiene un porcentaje mayor al 30% de similitud**, con el visto bueno de la Subdirección Académica.