

INSTITUTO TECNOLÓGICO DE CULIACÁN



DESARROLLO DE APLICACIÓN WEB PARA UN AUTOML

TESIS

OBTENER EL GRADO DE
INGENIERO EN SISTEMAS COMPUTACIONALES

POR:

Alvarez Hernandez Jesus Alberto

DIRECTOR DE TESIS:

Dr. Héctor Rodríguez Rangel

CULIACÁN, SINALOA 05 de agosto 2025



Instituto Tecnológico de Culiacán

DIVISION DE ESTUDIOS PROFESIONALES
ASUNTO: **Autorización de Impresión**

Culiacán, Sinaloa, 24/09/2025 13:06

C. JESUS ALBERTO ALVAREZ HERNANDEZ
PASANTE(S) DE LA CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES
P R E S E N T E.

De acuerdo con fallo emitido por la Comisión Revisora Integrada por:

Cédula Profesional

Presidente:	Doctorado en Ciencias en Ingeniería Eléctrica Opción en Sistemas Computacionales Héctor Rodríguez Rangel	9972835
Secretario:	Maestría en Ciencias en Ciencias de la Computación Gloria Ekaterine Peralta Peñuñuri	6038275
Vocal:	Maestría en Ciencias en Ciencias Computacionales Arcelia Judith Bustillos Martínez	13118410

Y considerando que reúne los requisitos establecidos en la Opción: **TITULACIÓN INTEGRAL: TESIS PROFESIONAL "DESARROLLO DE APLICACIÓN WEB PARA UN AUTOML"**, damos a usted nuestra autorización para que proceda a imprimir su trabajo profesional.

Sin otro particular, quedo de usted.

ATENTAMENTE
Excelencia en Educación Tecnológica®

LIC. CONCEPCIÓN MENDOZA ROSALES
JEFA DE LA DIVISIÓN DE ESTUDIOS PROFESIONALES

Secretaría de Educación Pública
INSTITUTO TECNOLÓGICO DE CULIACÁN
DIVISIÓN DE ESTUDIOS PROFESIONALES



2025
Año de
**La Mujer
Indígena**

Juan de Dios Báltiz 310 Pte. Col. Guadalupe C.P. 80220
Culiacán, Sinaloa. Tel. 667-454-0100
tecnm.mx www.culiacan.tecnm.mx





Índice

Contenido

Índice de tablas, gráficas y figuras.....	3
Capítulo I: introducción	4
1.- Introducción	5
1.1. Planteamiento del problema	6
1.2. Hipótesis	6
1.3. Objetivos.....	6
1.4.1 Objetivo General.....	6
1.4.2 Objetivos Específicos.....	7
1.4. Justificación	7
1.5. Estructura de la tesis	7
Capítulo II: Marco Teórico:.....	9
2.1 Marco Referencial:	10
2.1.1 La Brecha de Experticia y la Democratización del ML	10
2.1.2 Plataformas AutoML Existentes y sus Desafíos	10
2.1.3 El Rol de las Metodologías Ágiles	11
2.2 Marco Contextual:	11
2.2.1 La Necesidad Institucional y Académica	12
2.2.2 Contribución a la División de Estudios de Posgrado e Investigación	12
2.3 Marco Teórico:	13
2.3.1 Bases Teóricas y Conceptos Centrales.....	13
2.3.2 Arquitecturas y Tecnologías Habilitadoras.....	17
2.3.2.1 Arquitectura Cliente-Servidor.....	17
2.3.2.2 Computación en la Nube (Cloud Computing).....	18
2.3.2.3 APIs (Interfaces de Programación de Aplicaciones).....	19
2.3.2.4 Arquitectura de Microservicios.....	20
2.3.2.5 Contenedores y Orquestación (Docker y Kubernetes)	20
2.3.2.6 Base de Datos: PostgreSQL	21
2.3.2.7 Seguridad de la Información (HTTPS y Certificados SSL/TLS).....	22
Marco Conceptual.....	23



Capítulo III: Estado del arte	24
3.0 Estado del arte	25
3.1 Contexto general y servicios en la nube.....	25
3.2 Automatización y optimización con AutoML	25
3.3 Antecedentes en aprendizaje profundo	25
3.3.1 Aprendizaje automático aplicado en el ámbito empresarial.....	25
3.3.2 Inteligencia de negocio y predicciones automatizadas	26
3.3.3 Inteligencia artificial y logística en la cadena de suministro.....	26
3.4 Clasificación de enfoques AutoML.....	26
3.5 Componentes clave del AutoML	27
3.5.1 Ciclo de vida de un sistema AutoML	27
3.5.2 Tipos de AutoML según sus técnicas	27
3.5.3 Búsqueda de arquitecturas neuronales	27
3.6 Ventajas y desventajas del AutoML	27
3.7 Retos y vacíos de investigación.....	28
Capítulo IV: Desarrollo del proyecto	29
4.- Procedimiento y Descripción de las actividades desarrolladas.	30
4.1.- Determinación de tiempos de las actividades.	32
Tabla 1: Cronograma de actividades del proyecto.....	33
4.2.- Descripción de las Actividades realizadas.	33
Semanas 1 - 2: Planeación y levantamiento de requerimientos.....	33
Semanas 3 - 4: Análisis del sistema y modelado lógico	34
Semanas 5 - 6: Diseño de interfaces de usuario y arquitectura.....	34
Semanas 7 - 11: Desarrollo de la plataforma web	34
Semanas 12 - 13: Pruebas del sistema e iteración final	35
Semana 14: Despliegue en la nube y documentación técnica	35
Capítulo V: Pruebas y resultados	37
5.1.- Resultados, planos, gráficas, prototipos, manuales, programas,	38
Capítulo VI: Conclusiones.....	45
6.1. Conclusiones del proyecto.	46
6.2. Recomendaciones.....	46



6.3 Trabajo Futuro	47
6.4. Experiencia personal profesional adquirida.....	48
6.4. Competencias desarrolladas y/o aplicadas.	49
Capítulo VII: Fuentes de Información.....	52
7.1 Referencias bibliográficas y virtuales	53
Bibliografía.....	53

Índice de tablas, gráficas y figuras

Figura 1. Organigrama de la organización.	Error! Bookmark not defined.
Tabla 1: Cronograma de actividades del proyecto.....	33
Figura 2: Imagen de la petición del dataset Image	38
Figura 3: Imagen de la petición del dataset regression	39
Figura 4: Imagen de la petición del dataset TimeSeries	39
Figura 6: Ventana informativa.....	40
Figura 7: Inicio de sesión de la pagina	41
Figura 8: Pagina de renta de modelos	41
Figura 9: Pagina de créditos y facturación	42
Figura 10: Pagina de catálogo de templates	43
Figura 11: Pagina de monitoreo	43
Figura 12: Pagina de instancias	44



Capítulo I: introducción

1.- Introducción

El proyecto "**Desarrollo de una plataforma web para AutoML**" tiene como propósito diseñar e implementar una herramienta informática que simplifique la interacción entre los usuarios y los modelos de aprendizaje automático generados mediante *Automated Machine Learning (AutoML)*. La propuesta busca facilitar la gestión, visualización y manipulación de datos asociados a dichos modelos, de manera que personas sin formación técnica avanzada puedan beneficiarse de las capacidades de la inteligencia artificial.

En las últimas décadas, el aprendizaje automático se ha consolidado como un pilar de la transformación digital, permitiendo la automatización de tareas complejas, la identificación de patrones ocultos en grandes volúmenes de información y la generación de predicciones precisas. No obstante, a pesar de los avances tecnológicos, el uso de estas herramientas sigue concentrándose en especialistas con conocimientos en programación, estadística y ciencia de datos, lo que restringe su adopción en contextos donde podrían tener un impacto significativo.

La plataforma propuesta se estructura sobre una arquitectura web moderna, con un **frontend** intuitivo y adaptable, un **backend** robusto para el procesamiento de datos y la ejecución de modelos, y una integración con servicios en la nube que aseguren disponibilidad, escalabilidad y acceso multiplataforma. La metodología de trabajo adoptada es de tipo ágil, permitiendo ciclos iterativos de desarrollo, pruebas y mejoras continuas.

Entre los retos identificados se incluyen la optimización del uso de recursos computacionales para manejar modelos complejos y la gestión eficiente de la dependencia hacia servicios externos en la nube. Estas consideraciones se abordarán mediante estrategias de optimización, control de costos y evaluación constante del rendimiento.

En suma, este proyecto se presenta como una propuesta innovadora que pretende **democratizar el acceso a la inteligencia artificial**, potenciando su uso en ámbitos educativos, empresariales y sociales mediante una interfaz web amigable y accesible.

1.1. Planteamiento del problema

El crecimiento exponencial de datos en la era digital plantea la necesidad de herramientas eficientes que permitan su análisis y aprovechamiento. El *Machine Learning*, y más recientemente el *AutoML*, ofrecen mecanismos para automatizar procesos de modelado y análisis, reduciendo la intervención humana en etapas técnicas complejas.

Sin embargo, la implementación de estas soluciones enfrenta barreras significativas:

1. **Complejidad técnica persistente:** aunque AutoML automatiza la selección y ajuste de modelos, la preparación de datos, la interpretación de resultados y la configuración de parámetros siguen requiriendo conocimientos especializados.
2. **Accesibilidad limitada:** las plataformas más avanzadas requieren licencias costosas, infraestructura dedicada o conocimientos para su instalación y uso.
3. **Falta de interfaces orientadas al usuario común:** la mayoría de las herramientas están diseñadas pensando en usuarios con experiencia en ciencia de datos, lo que dificulta su adopción en contextos no especializados.

Problema central:

No se dispone de una plataforma web integral, de fácil uso y bajo costo, que permita gestionar, visualizar y manipular datos asociados a modelos AutoML, accesible desde cualquier dispositivo y para usuarios con distintos niveles de experiencia técnica.

1.2. Hipótesis

La creación de una plataforma web que ofrezca modelos de aprendizaje automático previamente entrenados bajo un esquema de **renta por uso** permitirá a los usuarios acceder de forma inmediata a soluciones predictivas sin necesidad de conocimientos avanzados en ciencia de datos ni de invertir en el desarrollo y entrenamiento de modelos propios.

Este modelo de servicio proporcionará a empresas e instituciones la posibilidad de utilizar modelos optimizados para diferentes tipos de datos y objetivos, pagando únicamente por el tiempo o volumen de uso requerido. La arquitectura modular y desplegada en la nube garantizará escalabilidad, alta disponibilidad y facilidad de mantenimiento, mientras que la inclusión de herramientas de preprocesamiento y explicabilidad de modelos aumentará la confiabilidad y transparencia de los resultados obtenidos.

De esta manera, se plantea que el modelo de renta de modelos de aprendizaje automático no solo reducirá las barreras técnicas y económicas de acceso a la inteligencia artificial, sino que también optimizará los procesos de análisis de datos en distintos contextos, mejorando la eficiencia y reduciendo costos operativos para los usuarios.

1.3. Objetivos

1.4.1 Objetivo General

Desarrollar una plataforma web interactiva que permita gestionar y manipular datos generados por un modelo AutoML, mediante interfaces intuitivas que faciliten la visualización, modificación y aprovechamiento de la información obtenida. Planificar y diseñar vistas funcionales y atractivas, incluyendo presentaciones iniciales y formularios, para mejorar la experiencia del usuario.

Implementar el modelo AutoML en la nube para asegurar disponibilidad y accesibilidad desde la aplicación web.

1.4.2 Objetivos Específicos

- Diseñar formularios personalizados según las necesidades del cliente, asegurando funcionalidad, atractivo visual, responsividad y correcta gestión de solicitudes HTTP.
- Garantizar el flujo correcto de eventos dentro de las vistas, previniendo errores y mostrando advertencias sobre el estado de los datos ingresados.
- Validar la integración de las interfaces con el modelo AutoML, verificando el manejo correcto de datos en tiempo real.

1.4. Justificación

La necesidad de contar con herramientas accesibles que permitan el uso de modelos de aprendizaje automático es cada vez más evidente, especialmente para usuarios sin conocimientos técnicos avanzados. La complejidad y los costos asociados al desarrollo y ajuste de estos modelos limitan su aplicación en diversos sectores, frenando el aprovechamiento de la inteligencia artificial para la toma de decisiones y la optimización de procesos.

Una plataforma web que facilite la gestión, visualización y renta de modelos AutoML preentrenados ofrece una solución práctica. Al simplificar el acceso a

modelos predictivos para tareas comunes, se reduce significativamente el tiempo y los recursos necesarios, fomentando la inclusión tecnológica y la innovación.

Asimismo, la automatización mediante algoritmos genéticos y el uso de tecnologías en la nube permiten garantizar escalabilidad, eficiencia y adaptabilidad a las necesidades cambiantes de los usuarios. Esto contribuye al desarrollo científico-tecnológico y al fortalecimiento de la competitividad en diferentes ámbitos, impulsando una adopción más amplia y responsable de la inteligencia artificial.

1.5. Estructura de la tesis

El desarrollo de este trabajo se ha organizado en seis capítulos que abordan, de manera progresiva, cada etapa del proyecto. La distribución de los apartados busca que el lector pueda comprender el origen del problema, los fundamentos que lo sustentan, el proceso metodológico seguido, así como los resultados y perspectivas de mejora.

Capítulo I – Introducción

En este primer capítulo se expone el planteamiento del problema y su contexto, explicando la importancia de contar con una solución que permita ofrecer modelos de aprendizaje automático bajo un esquema de renta. Se detallan los objetivos y la hipótesis que guiaron la investigación, así como la relevancia del proyecto en el marco de la transformación digital y la gestión eficiente de grandes volúmenes de datos.

Capítulo II – Marco teórico

Aquí se presentan los conceptos y bases técnicas que sustentan el desarrollo del proyecto. Se describen las tecnologías empleadas, los fundamentos del AutoML y el papel de las arquitecturas modulares y la computación en la nube. También se incluyen enfoques de diseño y experiencia de usuario que contribuyen a la usabilidad de la plataforma.

Capítulo III – Estado del arte

Este apartado analiza las soluciones y desarrollos previos relacionados con la temática del proyecto. Se revisan tendencias, herramientas y servicios existentes para la implementación de modelos de Machine Learning como servicio (MLaaS), evaluando sus fortalezas y limitaciones. Este análisis permitió identificar oportunidades para diferenciar y optimizar la solución propuesta.

Capítulo IV – Metodología

En este capítulo se describe el enfoque metodológico adoptado para la construcción de la plataforma. Se documenta la planificación del trabajo, la definición de requisitos, la selección de herramientas y el proceso de desarrollo iterativo. Asimismo, se detallan las estrategias utilizadas para gestionar cambios y garantizar la calidad del producto final.

Capítulo V – Resultados y validación

Se presentan las pruebas realizadas a la plataforma, incluyendo evaluaciones de rendimiento, usabilidad y funcionalidad. Los resultados se muestran mediante evidencias gráficas y descripciones que ilustran el funcionamiento del sistema. Además, se incorporan observaciones derivadas de las pruebas que orientan mejoras futuras.

Capítulo VI – Conclusiones y proyecciones

El último capítulo recoge las conclusiones obtenidas a partir de los resultados y contrasta estos con los objetivos iniciales. También se exponen recomendaciones y líneas de trabajo futuro, enfocadas en perfeccionar la plataforma, ampliar sus capacidades y consolidar un modelo de renta de servicios de inteligencia artificial que sea escalable y adaptable a distintas necesidades.



Capítulo II: Marco Teórico:

El aprendizaje automático automatizado (AutoML) ha transformado la forma de aplicar modelos predictivos, permitiendo que usuarios sin conocimientos avanzados construyan soluciones de análisis de datos de forma sencilla y eficiente. Plataformas como Google Cloud AutoML, Amazon SageMaker y H2O.ai ofrecen herramientas accesibles que integran selección de algoritmos, optimización de hiperparámetros y validación de modelos.

En el contexto agroindustrial de Sinaloa, la estimación de nutrientes en el agua de riego constituye un elemento clave para optimizar recursos y mejorar la productividad agrícola. AutoML facilita la automatización de este análisis, aportando predicciones rápidas y precisas que permiten una toma de decisiones más eficiente.

El Tecnológico Nacional de México Campus Culiacán, a través de la División de Estudios de Posgrado e Investigación, impulsa proyectos de innovación y transferencia de conocimiento, fomentando un entorno propicio para el desarrollo de plataformas de AutoML que respondan a las necesidades de la región y contribuyan al fortalecimiento de la industria agrícola.

En este marco, la automatización del aprendizaje automático resulta una herramienta estratégica para modernizar la gestión del agua de riego, contribuyendo a la sustentabilidad y competitividad del sector agroindustrial.

2.1 Marco Referencial:

2.1.1 La Brecha de Experticia y la Democratización del ML

En este panorama, ha emergido con fuerza el concepto de Automated Machine Learning (AutoML) como una respuesta directa a la necesidad de democratizar el acceso al ML. AutoML es una disciplina que busca automatizar la totalidad o una parte sustancial del flujo de trabajo de Machine Learning, incluyendo la preprocesamiento de datos, la ingeniería de características (feature engineering), la selección algorítmica (model selection), la optimización de hiperparámetros y la validación de modelos (Hutter et al., 2019). Esta automatización busca reducir significativamente la intervención manual y el conocimiento especializado requerido, permitiendo a usuarios con conocimientos limitados en ciencia de datos construir y desplegar modelos predictivos complejos con una eficiencia sin precedentes.

2.1.2 Plataformas AutoML Existentes y sus Desafíos

Actualmente, el mercado cuenta con una serie de plataformas AutoML comerciales y de código abierto que han demostrado la viabilidad y el impacto de esta automatización. Ejemplos prominentes incluyen:

- **Google Cloud AutoML:** Ofrece servicios personalizados para visión, lenguaje y datos tabulares, que automatizan el entrenamiento de modelos basados en las propias necesidades del usuario.
- **H2O Driverless AI:** Una plataforma de ML automatizado que se enfoca en la velocidad, la precisión y la interpretabilidad de los modelos.
- **Microsoft Azure Machine Learning:** Proporciona herramientas de AutoML para ayudar a los desarrolladores y científicos de datos a construir, desplegar y gestionar modelos de ML.

- **TPOT (Tree-based Pipeline Optimization Tool):** Una herramienta de código abierto basada en Python que utiliza programación genética para optimizar el pipeline de Machine Learning.

Si bien estas soluciones representan un avance significativo en la eficiencia y accesibilidad del ML, la experiencia en campo y diversas investigaciones (como la mencionada por Smith & Jones, 2022) han revelado que aún persisten desafíos sustanciales. La complejidad de sus interfaces de usuario, la necesidad de conocimientos técnicos específicos para la configuración avanzada y la gestión de la infraestructura subyacente (especialmente en entornos de nube) continúan siendo barreras considerables para su adopción generalizada. Esto es particularmente cierto en contextos donde los usuarios finales no son científicos de datos de profesión, sino investigadores, analistas o tomadores de decisiones que necesitan aplicar la IA sin la carga de su complejidad operacional. Esto subraya la imperiosa necesidad de soluciones que no solo automatizen los procesos de IA, sino que también optimicen drásticamente la experiencia de usuario y simplifiquen la gestión operativa para profesionales no especializados.

2.1.3 El Rol de las Metodologías Ágiles

En el contexto del desarrollo de una solución innovadora y centrada en el usuario como la propuesta, la elección de la metodología de desarrollo es fundamental. Las (Fowler, 2005) se presentan como el enfoque más adecuado debido a su naturaleza intrínsecamente flexible, adaptativa e iterativa. A diferencia de los modelos de desarrollo tradicionales en cascada, los enfoques ágiles como Scrum o Kanban priorizan:

- La colaboración constante con el usuario final y las partes interesadas.
- La entrega incremental de software funcional en ciclos cortos (sprints).
- La adaptación continua a los cambios en los requisitos y el feedback recibido.
- La importancia de las personas y sus interacciones sobre los procesos y herramientas rígidas.

Esta aproximación permite que la plataforma no solo se construya de manera eficiente, sino que también evolucione y se refine de forma orgánica en función del feedback temprano y recurrente de los usuarios. Esto asegura que el producto final sea altamente relevante, funcional y, crucialmente, usable, características esenciales para una herramienta diseñada para democratizar una tecnología compleja. Los ciclos iterativos inherentes a estas metodologías permiten una validación constante de las características implementadas, la identificación temprana de posibles desviaciones y la incorporación ágil de mejoras, garantizando la calidad, la escalabilidad futura y la estabilidad operativa de la plataforma.

2.2 Marco Contextual:

El desarrollo de esta aplicación web no ocurre en un vacío, sino que se inscribe en un contexto institucional específico y de gran relevancia: el **Instituto Tecnológico de Culiacán (IT Culiacán)**. Esta institución, piedra angular en la formación de profesionales de ingeniería y en la promoción de la investigación científica y tecnológica en la región de Sinaloa y a nivel nacional, posee una misión y

visiones claramente definidas: "Formar profesionales de alto nivel académico que contribuyan activamente al desarrollo comunitario mediante la aplicación de tecnología e investigación" (Instituto Tecnológico de Culiacán, s.f.) Esta misión no solo articula el propósito educativo del instituto, sino que también establece un marco propicio y estimulante para la innovación tecnológica y la aplicación práctica del conocimiento generado.

2.2.1 La Necesidad Institucional y Académica

Dentro de este entorno académico y de investigación, la necesidad de una plataforma como la propuesta se hace manifiesta. A pesar del innegable avance de las técnicas de aprendizaje automático, su adopción plena y efectiva en diversas áreas de investigación y aplicación práctica dentro de la institución y en sus proyectos de vinculación con el sector externo, se ve limitada. Las técnicas avanzadas de análisis de datos y modelado predictivo, aunque cruciales para la investigación de vanguardia y la toma de decisiones basada en evidencia, permanecen en gran medida en el dominio exclusivo de especialistas con formación profunda en ciencia de datos y programación. Esta situación restringe su potencial impacto en disciplinas donde la IA podría ser transformadora, como los estudios ambientales, la optimización de procesos o la economía regional, debido a la complejidad intrínseca de las herramientas actuales y la pronunciada curva de aprendizaje asociada.

La brecha existente entre el vasto potencial de la IA y la accesibilidad real para los investigadores y estudiantes no especializados ha generado una demanda latente. Existe una necesidad palpable de herramientas que permitan a los usuarios sin un perfil de científico de datos dedicado, aprovechar el poder de la inteligencia artificial para analizar sus propios datos, construir modelos predictivos y obtener *insights* valiosos, sin tener que dominar los lenguajes de programación o las complejidades algorítmicas. Esto es particularmente relevante en el contexto de proyectos de posgrado e investigación que buscan aplicar soluciones tecnológicas a problemas reales de la región.

2.2.2 Contribución a la División de Estudios de Posgrado e Investigación

El residente, como desarrollador web, se integra y colabora directamente con la **División de Estudios de Posgrado e Investigación** del Instituto Tecnológico de Culiacán. Esta división es el epicentro de la investigación de alto nivel dentro del IT Culiacán, donde se conciben y ejecutan proyectos de gran impacto que, con frecuencia, requieren capacidades avanzadas de análisis de datos y modelado predictivo.

La participación en el desarrollo de esta plataforma se alinea de manera estratégica y complementaria con la visión de la división de fomentar la investigación aplicada, la transferencia tecnológica y la formación de capital humano altamente capacitado. Al diseñar y construir una interfaz intuitiva y funcional que simplifica la gestión y ejecución de modelos predictivos generados a través de AutoML, el proyecto contribuye directamente a varias facetas de la misión institucional:

- **Optimización de Estudios Ambientales Regionales:** Al facilitar el acceso a herramientas de análisis predictivo, se potencia la capacidad de los investigadores para analizar datos ambientales complejos (ej. calidad del aire, patrones climáticos, contaminación) y desarrollar modelos que permitan predecir tendencias, identificar riesgos y proponer soluciones más efectivas para la región.

- **Impulso a la Investigación Aplicada:** La plataforma se convierte en un recurso valioso para la comunidad académica, permitiendo a estudiantes y profesores de diversas disciplinas aplicar técnicas de IA a sus campos de estudio sin la necesidad de una profunda especialización en informática.
- **Fortalecimiento de la Colaboración Interdisciplinaria:** Al democratizar el acceso a la IA, la plataforma puede fomentar la colaboración entre ingenieros de sistemas y expertos de otras áreas (biología, agronomía, etc.), generando proyectos multidisciplinarios que aborden problemáticas complejas desde múltiples perspectivas.
- **Desarrollo de Capacidades Locales:** El proyecto no solo entrega una herramienta, sino que también contribuye al desarrollo de capacidades internas en el Instituto para el diseño, desarrollo y gestión de soluciones de IA, posicionando al IT Culiacán como un referente en la aplicación tecnológica al desarrollo comunitario.

De esta manera, el proyecto no solo aborda una necesidad tecnológica, sino que se convierte en un catalizador para la investigación, la formación y el impacto social del Instituto Tecnológico de Culiacán, traduciendo el conocimiento técnico avanzado en soluciones prácticas y accesibles.

2.3 Marco Teórico:

Los fundamentos teóricos del proyecto se erigen sobre pilares esenciales que incluyen la automatización del aprendizaje automático, el desarrollo web y la interacción persona-máquina. Estos principios abordan directamente los problemas identificados: la complejidad en la gestión y ejecución de modelos AutoML, la necesidad de accesibilidad y despliegue en entornos web, y la demanda de un diseño de interfaz de usuario intuitivo y funcional. La construcción teórica del proyecto se estructura en bases fundamentales y en una conceptualización de términos clave.

2.3.1 Bases Teóricas y Conceptos Centrales

- **Aprendizaje Automático (Machine Learning - ML):** Constituye una rama fundamental de la inteligencia artificial (IA) que dota a los sistemas informáticos de la capacidad de aprender y mejorar a partir de la experiencia, sin necesidad de ser programados explícitamente para cada tarea específica (Samuel, 1959). En lugar de seguir instrucciones rígidas, los algoritmos de ML construyen modelos a partir de grandes conjuntos de datos, lo que les permite identificar patrones complejos, hacer predicciones o tomar decisiones con una intervención humana mínima. Este campo se subdivide principalmente en:
 - **Aprendizaje Supervisado:** Utiliza conjuntos de datos etiquetados (es decir, pares de entrada y salida conocidos) para entrenar un modelo. El objetivo es que el modelo aprenda a mapear las entradas a las salidas correctas. Ejemplos comunes incluyen problemas de regresión (predicción de valores continuos, como el precio de una casa) y clasificación (predicción de categorías discretas, como si un correo electrónico es spam o no). Algoritmos como regresión lineal, máquinas de vectores de soporte (SVM) y redes neuronales son ampliamente utilizados en este paradigma.
 - **Aprendizaje No Supervisado:** Trabaja con datos sin etiquetar para descubrir estructuras, patrones o relaciones ocultas dentro de ellos. No hay una "respuesta

correcta" predefinida. Las tareas típicas incluyen la agrupación (clustering), que organiza los datos en grupos significativos (ej., segmentación de clientes), y la reducción de dimensionalidad, que simplifica los datos conservando la información esencial. Algoritmos como K-Means, DBSCAN y Principal Component Analysis (PCA) son ejemplos de este tipo de aprendizaje.

- **Aprendizaje por Refuerzo:** Un agente de software aprende a tomar decisiones interactuando con un entorno dinámico, recibiendo "recompensas" por las acciones deseadas y "penalizaciones" por las no deseadas. El objetivo del agente es maximizar su recompensa acumulada a lo largo del tiempo. Este paradigma es fundamental en robótica, juegos (como AlphaGo) y sistemas de control autónomos. El flujo de trabajo típico en ML incluye la recolección de datos, la limpieza y preprocesamiento, la ingeniería de características (transformar datos brutos en características útiles), la selección y entrenamiento del modelo, la evaluación de su rendimiento, y finalmente, su despliegue para hacer predicciones en el mundo real. El ML es el núcleo subyacente de los modelos que la plataforma busca gestionar, destacando su poder transformador en el análisis de datos masivos y la automatización inteligente de tareas, desde el reconocimiento de voz hasta la optimización logística. Su aplicación en diversos dominios, incluida la agroindustria, demuestra su capacidad para generar valor a partir de los datos.
- **Aprendizaje Automático Automatizado (AutoML):** Representa un subcampo emergente y crítico del Machine Learning cuyo objetivo fundamental es automatizar, en la medida de lo posible, todas las etapas del pipeline de aprendizaje automático (Hutter, 2019). Tradicionalmente, la construcción de modelos de ML de alto rendimiento es un proceso laborioso y complejo que requiere una profunda experiencia en ciencia de datos, programación y conocimientos específicos del dominio. AutoML busca simplificar este proceso al automatizar tareas como:
 - **Preprocesamiento de Datos:** Incluye la detección y manejo de valores perdidos, normalización, escalado y codificación de variables categóricas, pasos cruciales para la calidad del modelo.
 - **Ingeniería de Características (Feature Engineering):** Creación de nuevas variables (características) a partir de las existentes que pueden mejorar significativamente el rendimiento del modelo. AutoML puede generar estas características de forma automática, explorando interacciones y transformaciones.
 - **Selección de Modelos (Model Selection):** Elección del algoritmo de ML más adecuado para una tarea dada entre una amplia biblioteca de opciones (ej., regresión logística, árboles de decisión, redes neuronales, boosting de gradiente).
 - **Optimización de Hiperparámetros (Hyperparameter Optimization - HPO):** Ajuste de los parámetros internos del algoritmo (que no se aprenden de los datos) para optimizar el rendimiento del modelo. Esto puede involucrar técnicas como búsqueda en cuadrícula, búsqueda aleatoria, optimización bayesiana o algoritmos genéticos.
 - **Validación y Evaluación de Modelos:** Métricas y técnicas (ej., validación cruzada k-fold) para asegurar la robustez, generalización y evitar el sobreajuste del modelo.
 - **Selección de Arquitectura de Redes Neuronales (Neural Architecture Search - NAS):** En el caso de Deep Learning, la automatización del diseño de la arquitectura de la red neuronal (número de capas, tipos de capas, conexiones), lo cual es una

tarea extremadamente compleja. AutoML es crucial para este proyecto, ya que su capacidad de automatización reduce drásticamente la barrera de entrada para usuarios sin una profunda experiencia en ciencia de datos, permitiéndoles aprovechar modelos complejos de manera eficiente y acelerando el ciclo de desarrollo y despliegue de soluciones de IA. Además de la democratización, AutoML contribuye a mejorar la reproducibilidad de los resultados, a optimizar el rendimiento de los modelos al explorar un espacio de soluciones que un experto humano no podría cubrir manualmente en un tiempo razonable, y a liberar a los científicos de datos de tareas repetitivas para que se enfoquen en problemas más estratégicos.

- **Desarrollo Web:** Comprende todas las actividades inherentes a la creación de aplicaciones accesibles a través de un navegador de internet, las cuales pueden variar desde sitios web estáticos (simples páginas informativas) hasta complejas plataformas interactivas con lógica de negocio sofisticada. Esta disciplina se divide conceptualmente en dos grandes áreas que interactúan de forma constante (geeksforgoeks, s.f.) (writer, 2025).
 - **Frontend (Desarrollo del Lado del Cliente):** Se encarga de la parte de la aplicación con la que el usuario interactúa directamente. Esto incluye el diseño de la interfaz visual (UI), la interactividad y la experiencia de usuario (UX). Las tecnologías predominantes son **HTML** (HyperText Markup Language, para la estructura del contenido y semántica), **CSS** (Cascading Style Sheets, para el estilo, diseño visual y responsividad de la interfaz) y **JavaScript** (para la lógica interactiva, manipulación del DOM y comunicación asíncrona con el servidor). Librerías y frameworks modernos como **React**, Angular o Vue.js permiten construir Single Page Applications (SPAs) y interfaces de usuario complejas y reactivas de manera modular y eficiente. El frontend es crucial para la accesibilidad y la facilidad de uso de la plataforma AutoML.
 - **Backend (Desarrollo del Lado del Servidor):** Se ocupa de la lógica de negocio, la gestión de bases de datos, la autenticación y autorización de usuarios, la integración con servicios externos (como las APIs de AutoML en la nube), y el procesamiento de la información. Es el "cerebro" de la aplicación. Lenguajes de programación como **Python** (con frameworks robustos como Django o Flask), Node.js (JavaScript), Ruby (con Rails), Java (con Spring Boot) o PHP (con Laravel) son comunes en el backend. El backend gestiona la persistencia de datos, la seguridad de la información y la ejecución de operaciones computacionalmente intensivas. La interacción entre el frontend y el backend se realiza típicamente a través de APIs (Interfaces de Programación de Aplicaciones), generalmente RESTful. El desarrollo web es el medio tecnológico indispensable para la construcción de la plataforma propuesta, asegurando su disponibilidad global, accesibilidad desde cualquier dispositivo con navegador y la interacción fluida con los modelos AutoML, permitiendo a los usuarios gestionar y visualizar los resultados de manera intuitiva y eficiente. La importancia de la compatibilidad entre navegadores y la adaptabilidad a diferentes tamaños de pantalla (diseño responsivo) son consideraciones clave en el desarrollo web moderno.
- **Experiencia de Usuario (UX) e Interfaz de Usuario (UI):** Estos dos conceptos, aunque interrelacionados, describen aspectos distintos pero complementarios en el diseño de

sistemas interactivos, y son críticos para la adopción y el éxito de herramientas complejas como las de AutoML:

- **Experiencia de Usuario (UX):** Engloba el conjunto completo de percepciones y respuestas emocionales de una persona resultantes del uso o la anticipación de uso de un producto, sistema o servicio. Se centra en *cómo* se siente el usuario al interactuar con el sistema, si es eficiente, si satisface sus necesidades, si es agradable, intuitivo y si es fácil de usar. La UX abarca un proceso iterativo que incluye investigación de usuarios (entrevistas, encuestas, análisis de tareas), creación de personas de usuario, diseño de flujos de usuario y arquitectura de la información, prototipado y pruebas de usabilidad. Un buen diseño UX minimiza la frustración, reduce la curva de aprendizaje, maximiza la satisfacción del usuario y aumenta la probabilidad de que la aplicación sea adoptada y utilizada de forma recurrente. Para una plataforma AutoML, una UX efectiva debe simplificar la complejidad subyacente del Machine Learning, haciendo que las tareas avanzadas sean accesibles para usuarios con diferentes niveles de experiencia.
- **Interfaz de Usuario (UI):** Se refiere a los medios visuales, táctiles y auditivos a través de los cuales una persona interactúa con un sistema (ISO 9241-210, 2019). Es la parte gráfica y tangible del software, incluyendo todos los elementos visuales con los que el usuario interactúa: botones, iconos, campos de texto, menús desplegables, tipografía, paleta de colores y la disposición general de la pantalla. Un buen diseño UI se enfoca en la estética, la consistencia visual (utilizando sistemas de diseño y librerías de componentes), la claridad y la facilidad de navegación, asegurando que los elementos sean atractivos y funcionales. En el contexto de esta plataforma, el diseño de una UI limpia, intuitiva, accesible y responsiva, combinada con una UX optimizada que anticipe las necesidades de los usuarios no expertos en ciencia de datos, es una prioridad crítica. La facilidad de uso impacta directamente en la adopción, la eficiencia en la gestión y la eficacia en la modificación de los modelos, convirtiendo una herramienta potente pero compleja en una solución práctica y ampliamente utilizable.
- **Metodología Ágil:** Consiste en un conjunto de enfoques iterativos e incrementales para la gestión de proyectos de software, caracterizados por su énfasis en la flexibilidad, la colaboración continua con el cliente y la entrega frecuente de software funcional (Beck, 2001) ; (Fowler, 2005), A diferencia de los modelos de desarrollo tradicionales en cascada (donde todas las fases se completan secuencialmente), las metodologías ágiles como Scrum o Kanban se centran en:
 - **Entrega Iterativa e Incremental:** El software se desarrolla en ciclos de tiempo cortos y fijos, conocidos como "sprints" (típicamente de 1 a 4 semanas), al final de cada uno de los cuales se entrega una porción de software funcional y potencialmente desplegable. Esto permite obtener retroalimentación temprana y ajustar el rumbo del proyecto.
 - **Colaboración Activa con el Cliente:** El cliente o usuario final no es un mero receptor al final del proyecto, sino un participante activo y constante, proporcionando retroalimentación continua que guía el desarrollo. Esto asegura que el producto final satisfaga las necesidades reales.

- **Adaptación al Cambio:** Prioriza la respuesta rápida a los cambios en los requisitos del proyecto o en las condiciones del mercado sobre el seguimiento rígido de un plan inicial. El cambio es visto como una oportunidad para mejorar, no como un obstáculo.
- **Personas e Interacciones sobre Procesos y Herramientas:** Valoriza la comunicación efectiva, la auto-organización de los equipos y la colaboración interfuncional sobre la documentación exhaustiva o las herramientas rígidas.
- **Software Funcionando sobre Documentación Exhaustiva:** Aunque la documentación es importante, el énfasis recae en entregar software que funcione y aporte valor. La adopción de una metodología ágil (como Scrum, por ejemplo, con sus reuniones diarias (stand-ups), planificación de sprints, revisiones y retrospectivas) para guiar el proceso de desarrollo de esta plataforma es crucial. Permite ciclos de prueba y mejora constante, lo que garantiza no solo la calidad del producto y su escalabilidad futura, sino también una estabilidad operativa al adaptarse rápidamente a los requerimientos cambiantes del entorno y a la retroalimentación temprana de los usuarios. Esto asegura que el producto final sea altamente relevante, funcional y, crucialmente, usable, características esenciales para una herramienta diseñada para democratizar una tecnología compleja y en constante evolución como AutoML.

2.3.2 Arquitecturas y Tecnologías Habilitadoras

“La solidez, escalabilidad, seguridad y eficiencia de la plataforma propuesta para la gestión de modelos AutoML dependen intrínsecamente de la selección y aplicación estratégica de arquitecturas y tecnologías modernas y consolidadas en el ecosistema del desarrollo de software y la computación en la nube. Estas elecciones no solo posibilitan la materialización de las funcionalidades clave de la aplicación, sino que también garantizan su viabilidad a largo plazo, su capacidad de adaptación y su potencial de crecimiento futuro”.

2.3.2.1 Arquitectura Cliente-Servidor

Esta es la configuración estructural fundamental sobre la cual operan la vasta mayoría de las aplicaciones web contemporáneas, y constituye el pilar de la plataforma que nos ocupa. En este modelo paradigmático (S.Tanenbaum, 2007), las responsabilidades se distribuyen de forma clara y lógica entre dos componentes principales que interactúan para cumplir las solicitudes del usuario:

- **Clientes:** Representados por los navegadores web en los dispositivos de los usuarios. Los clientes son los encargados de la interfaz de usuario, la presentación de la información y la iniciación de solicitudes. Capturan las interacciones del usuario y las traducen en peticiones de datos o servicios que se envían al servidor. Con el auge de las Single Page Applications (SPAs) construidas con frameworks como React, la lógica de presentación y parte de la lógica de negocio se ejecuta directamente en el navegador, mejorando la reactividad y la experiencia del usuario.
- **Servidores:** Son los programas o máquinas que residen en una ubicación centralizada (a menudo en la nube) y que alojan la lógica de negocio de la aplicación. Procesan las solicitudes recibidas de los clientes, interactúan con bases de datos, sistemas externos y

realizan operaciones computacionalmente intensivas. Una vez procesada la solicitud, el servidor genera una respuesta (datos en formato JSON o una página web generada dinámicamente) y la devuelve al cliente para su visualización. Los servidores pueden ser web, de aplicaciones o de bases de datos. Esta división nítida de responsabilidades entre el frontend (cliente) y el backend (servidor) ofrece múltiples ventajas. Promueve una especialización y eficiencia en el desarrollo, permitiendo que equipos especializados en la interfaz de usuario (UI/UX) y en la lógica de negocio backend trabajen de forma concurrente y con relativa independencia, siempre y cuando las interfaces de comunicación APIs se mantengan consistentes. Además, esta arquitectura facilita enormemente la escalabilidad, dado que se pueden añadir más instancias de servidor para manejar un incremento en la demanda de usuarios o en la carga de procesamiento sin afectar la experiencia del cliente. También contribuye a una mayor resiliencia, ya que un fallo en un componente no necesariamente impide que otras partes de la aplicación sigan operando o, al menos, muestren una respuesta controlada al usuario. La comunicación entre cliente y servidor es, en la mayoría de los casos de aplicaciones web modernas, sin estado (stateless) para mejorar la escalabilidad, donde cada solicitud del cliente al servidor contiene toda la información necesaria para que el servidor la entienda y procese, reduciendo la necesidad de mantener sesiones persistentes en el servidor.

2.3.2.2 Computación en la Nube (Cloud Computing)

La capacidad de la plataforma para operar, gestionar y, sobre todo, para ejecutar modelos de Aprendizaje Automático Automatizado (AutoML), los cuales exigen con frecuencia volúmenes considerables de recursos computacionales y capacidad de almacenamiento, se fundamenta de manera imprescindible en la adopción de la computación en la nube. Este modelo (Mell, 2011) describe un paradigma que habilita el acceso bajo demanda, a través de la red, a un conjunto configurable y compartido de recursos informáticos virtualizados, que pueden ser aprovisionados y liberados de manera ágil y rápida, con una intervención mínima por parte del usuario o una interacción limitada con el proveedor de servicios. La computación en la nube se categoriza comúnmente en tres modelos de servicio:

- **IaaS (Infrastructure as a Service):** Proporciona recursos informáticos virtualizados a través de internet. El usuario gestiona el sistema operativo y las aplicaciones.
- **PaaS (Platform as a Service):** Ofrece un entorno completo para desarrollar, ejecutar y gestionar aplicaciones sin la complejidad de construir y mantener la infraestructura subyacente. El proveedor de la nube gestiona la infraestructura.
- **SaaS (Software as a Service):** El software está alojado en la nube y se accede a él a través de un navegador web, sin necesidad de instalación local. La computación en la nube es un habilitador crítico para este proyecto por varias razones estratégicas:
- **Escalabilidad Elástica y Bajo Demanda:** Permite que los recursos computacionales se ajusten dinámicamente a las fluctuaciones de la demanda. Esto es de vital importancia para los procesos de AutoML, los cuales pueden experimentar picos intensivos de procesamiento durante las fases de entrenamiento de modelos complejos, y luego reducirse drásticamente durante la fase de inferencia o cuando la plataforma está en reposo. Esta elasticidad asegura que la aplicación pueda manejar tanto la carga mínima como los picos de uso sin interrupciones ni constante de recursos, lo que optimiza el uso de recursos y, consecuentemente, los costos operativos.

- **Accesibilidad Global y Alta Disponibilidad:** Al alojar tanto la aplicación web como, crucialmente, los servicios de ejecución de modelos en la infraestructura de la nube, se garantiza su disponibilidad ininterrumpida para usuarios ubicados en cualquier parte del mundo con una conexión a internet estable. Los proveedores de la nube ofrecen redundancia geográfica, balanceo de carga global y mecanismos de recuperación ante desastres que aseguran una alta disponibilidad y resiliencia frente a fallos. Esto es fundamental para una herramienta colaborativa y distribuida.
- **Reducción de Costos Operativos y de Capital (CapEx vs. OpEx):** Elimina la necesidad de realizar grandes inversiones iniciales en la adquisición de hardware físico, su configuración, mantenimiento, actualización y espacio en centros de datos. La nube opera bajo un modelo de pago por uso, donde solo se incurre en costos por los recursos realmente consumidos, transformando gastos fijos en variables más predecibles y ajustables según el uso real. Esto permite a las organizaciones innovar y escalar sin una gran carga financiera inicial.
- **Servicios Gestionados y Ecosistema de Herramientas:** Los principales proveedores de servicios en la nube (como Google Cloud Platform, Amazon Web Services o Microsoft Azure) ofrecen una amplia gama de servicios gestionados que abstraen la complejidad subyacente de la infraestructura. Esto incluye bases de datos gestionadas, almacenamiento de objetos, y plataformas de Machine Learning y AutoML gestionadas. Estos servicios están pre-configurados, optimizados y listos para ser integrados mediante sus APIs, lo que simplifica significativamente el desarrollo, el despliegue y la gestión de los modelos de ML, permitiendo al equipo del proyecto enfocarse en la lógica de negocio central y la experiencia del usuario, en lugar de la gestión de la infraestructura subyacente. La nube no es simplemente un lugar para almacenar datos o ejecutar código; es el "motor" y el "cerebro" que habilita la ejecución escalable y eficiente de los modelos AutoML.

2.3.2.3 APIs (Interfaces de Programación de Aplicaciones)

Las APIs (Application Programming Interfaces) representan el lenguaje universal y el medio por el cual los distintos componentes de software de la plataforma se comunican e interactúan entre sí, así como con servicios externos. Son un conjunto de definiciones, protocolos y herramientas que facilitan la comunicación estandarizada y eficiente entre diferentes sistemas de software (webopedia, 2020) (wikipedia, s.f.). Su rol es multifacético y absolutamente indispensable en el proyecto:

- **Comunicación Frontend-Backend:** El frontend, desarrollado en React, establecerá comunicación con el backend de Python exclusivamente a través de APIs RESTful. Estas APIs permiten al cliente enviar solicitudes de diversas operaciones (GET para obtener datos, POST para crear un recurso, PUT/PATCH para actualizar, DELETE para eliminar) y recibir las respuestas correspondientes del servidor en formatos estandarizados. Esta clara interfaz promueve la desvinculación de las capas, permitiendo su evolución independiente.
- **Integración con Servicios AutoML en la Nube:** El backend de Python será el intermediario principal que interactúe con las APIs provistas por los servicios de computación en la nube. Estas APIs exponen la funcionalidad de los servicios de ML gestionados. Mediante estas interfaces, el backend podrá invocar y gestionar programáticamente todo el ciclo de vida de los procesos de AutoML: desde la subida de datasets, la configuración del entrenamiento, el monitoreo del progreso del trabajo, el despliegue del modelo entrenado, y la ejecución de

predicciones. Esto permite que el proyecto aproveche el poder y la infraestructura optimizada de las grandes plataformas de ML sin tener que construir toda la infraestructura de cálculo desde cero.

- **Modularidad y Reusabilidad:** El diseño basado en APIs fomenta una arquitectura modular, donde cada componente de software se puede desarrollar, probar y mantener de manera independiente. Esto no solo facilita el trabajo en equipo y la asignación de responsabilidades, sino que también permite la reutilización de funcionalidades existentes y la futura expansión del sistema.
- **Estandarización y Seguridad en las Interacciones:** Las APIs establecen un punto de entrada definido y controlado para todas las interacciones, lo que es crucial para la seguridad. Permiten implementar rigurosos mecanismos de autenticación y autorización. Las APIs deben documentarse claramente para facilitar su consumo y asegurar la consistencia.

2.3.2.4 Arquitectura de Microservicios

Si bien la fase inicial del proyecto podría implementarse con una arquitectura monolítica para el backend, es fundamental reconocer la arquitectura de microservicios como una evolución natural y deseable a medida que la plataforma crezca en complejidad, número de funcionalidades y escala de usuarios. En una arquitectura de microservicios, una aplicación se descompone y se construye como una colección de servicios pequeños, completamente autónomos, débilmente acoplados y desplegados de forma independiente (Newman, 2015). Cada microservicio está diseñado para realizar una función de negocio específica y se comunica con otros microservicios, generalmente a través de APIs RESTful o mecanismos de mensajería asíncrona. Las ventajas clave incluyen:

- **Mayor Escalabilidad Independiente:** Cada servicio puede escalarse de forma autónoma según su carga de trabajo específica, permitiendo optimizar el uso de recursos y costos.
- **Resiliencia Mejorada:** Un fallo en un microservicio específico rara vez impacta a toda la aplicación. El resto de los servicios pueden seguir funcionando, limitando el alcance de los problemas.
- **Agilidad en el Desarrollo y Despliegue:** Pequeños equipos multidisciplinarios pueden desarrollar, probar, desplegar y mantener microservicios de forma independiente y frecuente, lo que acelera el ciclo de vida del desarrollo y permite entregas continuas.
- **Flexibilidad Tecnológica:** Permite a los equipos elegir las tecnologías más adecuadas para cada microservicio, sin atar toda la aplicación a una única pila tecnológica monolítica.
- **Facilidad de Mantenimiento:** Un microservicio más pequeño es intrínsecamente más fácil de entender, mantener, probar y refactorizar que una aplicación monolítica gigante. Aunque su implementación inicial puede añadir complejidad operacional, es una consideración estratégica importante para la sostenibilidad y evolución a largo plazo del proyecto, especialmente si se anticipa un crecimiento significativo en funcionalidades y una necesidad de escalar para una gran base de usuarios. La madurez de herramientas de orquestación como Kubernetes ha simplificado la gestión de microservicios.

2.3.2.5 Contenedores y Orquestación (Docker y Kubernetes)

Para el despliegue eficiente, la consistencia operativa y la gestión a escala de las aplicaciones cliente-servidor, especialmente en entornos de computación en la nube y con

una potencial arquitectura de microservicios, las tecnologías de contenedores y su orquestación son herramientas imprescindibles.

- **Docker:** Es una plataforma de código abierto que permite empaquetar una aplicación y todas sus dependencias en una unidad estandarizada y portátil llamada contenedor (Merkel, 2014). Esto garantiza que la aplicación se ejecute de manera consistente en cualquier entorno compatible con Docker. Los contenedores son ligeros, eficientes en el uso de recursos, y aíslan las aplicaciones entre sí. Un Dockerfile define los pasos para construir la imagen del contenedor.
- **Kubernetes (K8s):** Es un sistema de código abierto, diseñado originalmente por Google, para la automatización del despliegue, escalado y gestión de aplicaciones en contenedores (Burns, 2022). K8s actúa como un "sistema operativo" para clústeres de contenedores, abstracción que permite a los desarrolladores centrarse en el código de la aplicación. Sus capacidades clave incluyen:
 - **Orquestación:** Gestión de múltiples contenedores y cómo se comunican entre sí en un clúster de máquinas.
 - **Balanceo de Carga y Descubrimiento de Servicios:** Distribución del tráfico de red entrante y la capacidad para que los servicios se encuentren automáticamente.
 - **Auto-reparación:** Reinicio automático de contenedores que fallan, reemplazo de defectuosos y eliminación de los no responsivos.
 - **Escalado Automático:** Ajuste dinámico del número de instancias de la aplicación en función de la demanda.
 - **Despliegues Declarativos y Rollbacks:** Permite describir el estado deseado de la aplicación y sus componentes. La integración de Docker para la contenerización del frontend y backend de la plataforma, y Kubernetes para la orquestación de estos contenedores, facilita enormemente los procesos de integración y despliegue continuos (CI/CD), asegurando una operación robusta, altamente disponible y eficiente en el entorno de la nube.

2.3.2.6 Base de Datos: PostgreSQL

La gestión eficiente del almacenamiento y la recuperación de datos es un componente vertebral para cualquier aplicación web, y la elección de la base de datos adecuada es crucial. Para este proyecto, se ha seleccionado PostgreSQL, un sistema de gestión de bases de datos relacional de objetos (ORDBMS) de código abierto, reconocido por su robustez, su fiabilidad, su amplio conjunto de características y su estricta adherencia a los estándares SQL (Stonebraker, 1986). PostgreSQL es ideal para este proyecto por las siguientes razones:

- **Consistencia y Fiabilidad (ACID):** Ofrece sólidas garantías de atomicidad, consistencia, aislamiento y durabilidad (ACID) para las transacciones. Esto es fundamental para manejar datos críticos de la plataforma, donde la integridad de los datos es primordial.
- **Adherencia a Estándares SQL:** Al ser altamente compatible con el estándar SQL, facilita la definición de esquemas de datos estructurados, el manejo de consultas complejas y el uso de funciones avanzadas. Esto es esencial para organizar la información de manera lógica y eficiente.
- **Extensibilidad y Versatilidad:** PostgreSQL es altamente extensible, permitiendo la adición de funcionalidades personalizadas a través de funciones, tipos de datos, operadores y extensiones. Su soporte nativo para JSONB es ventajoso para almacenar y consultar datos semi-estructurados.

- **Rendimiento y Escalabilidad:** Es capaz de manejar cargas de trabajo significativas y escalar tanto verticalmente como horizontalmente. Sus optimizadores de consultas son eficientes y la concurrencia se maneja bien a través de MVCC.
- **Comunidad y Soporte:** Cuenta con una comunidad de desarrolladores y usuarios muy activa y comprometida, lo que se traduce en una gran cantidad de recursos, documentación y actualizaciones continuas. Es una opción madura y de confianza.
- **Modelo de Datos:** Puede almacenar una variedad de datos relevantes para la plataforma, incluyendo:
 - **Datos de Usuarios:** Perfiles, roles, permisos y credenciales.
 - **Metadatos de Proyectos:** Nombres, descripciones, fechas y estados.
 - **Configuraciones de Modelos AutoML:** Parámetros específicos, historial y hiperparámetros.
 - **Registros de Entrenamiento:** Estado de los trabajos, duración, métricas de rendimiento y referencias a logs.
 - **Referencias a Datasets:** Ubicaciones en la nube y metadatos del dataset. La elección de PostgreSQL proporciona una base de datos robusta y fiable para el almacenamiento de la información estructurada y crítica de la plataforma, complementando la gestión de datos en la nube para los artefactos de AutoML más grandes.

2.3.2.7 Seguridad de la Información (HTTPS y Certificados SSL/TLS)

En cualquier aplicación web, y más aún en una que gestiona datos de usuario e información sensible, la seguridad de la información es una prioridad no negociable. Garantizar la confidencialidad, integridad y disponibilidad de los datos es fundamental para la confianza del usuario, la reputación de la plataforma y el cumplimiento normativo.

- **HTTPS (Hypertext Transfer Protocol Secure):** Es la extensión segura del protocolo HTTP (Berners-Lee, 1996). La "S" denota que la comunicación entre el navegador del usuario y el servidor web está cifrada. Este cifrado previene eficazmente que terceros puedan interceptar, leer o manipular los datos transmitidos. HTTPS es el primer nivel de seguridad para las comunicaciones web, protegiendo contra ataques de "man-in-the-middle". El cifrado se establece utilizando Secure Sockets Layer (SSL) o su sucesor, Transport Layer Security (TLS).
- **Certificados SSL/TLS (Secure Sockets Layer / Transport Layer Security):** Son la tecnología criptográfica esencial que subyace a HTTPS. Un certificado digital SSL/TLS vincula criptográficamente una clave pública a la información del propietario del sitio web. Cuando se instala en un servidor web, activa el protocolo HTTPS y se manifiesta en el navegador del usuario mediante el icono de un candado y el prefijo https:// en la URL. Los certificados SSL/TLS, emitidos por Autoridades de Certificación (CA) de confianza, ofrecen las siguientes garantías críticas:
 - **Confidencialidad (Cifrado):** Asegura que la comunicación está cifrada de extremo a extremo, de modo que solo el remitente y el receptor legítimos puedan leer la información transmitida.
 - **Integridad de Datos:** Garantiza que los datos no han sido alterados, corrompidos o manipulados durante su transmisión.
 - **Autenticación del Servidor:** Verifica la identidad del servidor al que el usuario se está conectando. Todos los puntos de interacción de la plataforma deben estar

protegidos de forma rigurosa por HTTPS y certificados SSL/TLS válidos y actualizados. Esto no solo asegura la confianza del usuario y la privacidad de los datos, sino que también es un requisito estándar para las mejores prácticas de seguridad en la web y un factor importante para el posicionamiento en motores de búsqueda (SEO). Además de HTTPS/SSL/TLS, la seguridad de la información en la plataforma debe ser una estrategia por capas que incluya: autenticación robusta de usuarios, autorización basada en roles (RBAC), validación de entradas de datos, cifrado de datos en reposo, monitoreo continuo de seguridad y auditorías regulares.

Marco Conceptual

- **Plataforma Web:** Aplicación informática accesible y funcional a través de un navegador de internet, diseñada para ofrecer un conjunto específico de servicios o funcionalidades interactivas a los usuarios (mozilla, s.f.)
- **Modelo AutoML:** Una representación matemática de patrones aprendidos por un algoritmo de aprendizaje automático, cuya construcción y optimización han sido realizadas de forma automatizada mediante herramientas de AutoML (Hutter, 2019).
- **Frontend:** Corresponde a la parte de una aplicación web directamente visible y con la que el usuario interactúa; engloba el diseño visual, los elementos de control y la presentación de la información (geeksforgeeks, s.f.)
- **Backend:** Es la "parte oculta" de una aplicación web, responsable de la lógica de negocio, el procesamiento de datos, la gestión de la base de datos y la comunicación con el servidor (geeksforgeeks, s.f.)
- **Responsividad:** Se refiere a la capacidad de una interfaz web para adaptar su diseño, maquetación y presentación de manera óptima a diversos tamaños de pantalla y tipos de dispositivos (mozilla, s.f.)
- **Ciclos Iterativos:** Son fases repetitivas y continuas dentro de una metodología de desarrollo (como la ágil), que típicamente incluyen el diseño, la implementación, las pruebas y la revisión, permitiendo mejoras constantes y la adaptación a nuevos requisitos (Beck, 2001) (Fowler, 2005).
- **Servicios en la Nube:** Recursos informáticos virtualizados (como potencia de cómputo, almacenamiento de datos, plataformas de inteligencia artificial) que son ofrecidos por un proveedor externo y accesibles a través de internet, generalmente bajo un modelo de pago por uso (Mell, 2011).

Capítulo III: Estado del arte

3.0 Estado del arte

3.1 Contexto general y servicios en la nube

En las últimas décadas, los **servicios en la nube** han revolucionado el ámbito tecnológico, proporcionando soluciones que van desde infraestructura de redes y máquinas virtuales hasta plataformas de bases de datos administradas. Entre estas soluciones, las herramientas de **Aprendizaje Automático Automatizado (AutoML)** se han convertido en una respuesta efectiva a uno de los principales retos en ciencia de datos: la necesidad de realizar de forma manual el entrenamiento de modelos complejos de aprendizaje profundo (Peña Lorenzo, 2020).

El desarrollo de modelos de este tipo requiere, en muchos casos, de datos previamente etiquetados, lo cual supone un proceso extenso y meticuloso, especialmente si el conjunto de datos está segmentado en diferentes partes. Un ejemplo lo describe Peña Lorenzo (2020) en *“Aplicación de técnicas de aprendizaje profundo (deep learning) para la detección de objetos en Industria 4.0”*, donde se emplearon imágenes clasificadas para distinguir entre piezas defectuosas y no defectuosas, manejando únicamente dos categorías en la estimación.

3.2 Automatización y optimización con AutoML

El **AutoML** ofrece una solución para automatizar tareas que suelen requerir un alto esfuerzo manual, como la selección y el ajuste de hiperparámetros. Estos procesos, realizados sin automatización, implican múltiples ciclos de ensayo y error o la implementación de métodos de optimización específicos. Con AutoML, el flujo de trabajo se organiza en *pipelines* que, según Giraudy (2022), mantienen “un balance entre la exploración y la explotación”, optimizando el rendimiento del modelo y reduciendo significativamente los tiempos de ajuste.

3.3 Antecedentes en aprendizaje profundo

El auge del AutoML se relaciona directamente con la evolución de las arquitecturas de **aprendizaje profundo**. La introducción de **AlexNet** marcó un cambio significativo, impulsando la adopción de redes más profundas y complejas (Cabrera, 2023). Desde modelos iniciales como LeNet, pasando por VGG y ResNet, hasta arquitecturas más recientes como DenseNet, la complejidad ha aumentado considerablemente, elevando tanto los costos de diseño como el riesgo de errores. Cabrera (2023) señala que estos desarrollos requieren conocimientos especializados, grandes cantidades de datos etiquetados y prolongados procesos de entrenamiento, lo que ha fomentado la exploración de enfoques como el *self-supervised learning* para reducir la dependencia de la anotación manual.

3.3.1 Aprendizaje automático aplicado en el ámbito empresarial

En la actualidad, las grandes corporaciones han incrementado de manera notable sus inversiones en **aprendizaje automático e inteligencia artificial**, incorporando además perfiles altamente especializados para el desarrollo de estas tecnologías. Este interés no se limita únicamente a empresas consolidadas como Google, Apple, Amazon o Facebook —que ya han integrado la IA en numerosos productos—, sino que también alcanza a **startups** y pequeñas empresas que, a pesar de contar con recursos limitados, buscan adaptarse a innovaciones disruptivas que les permitan

obtener ventajas competitivas y asegurar su permanencia en mercados cada vez más exigentes (IAobservatorio, 2016).

Un ejemplo de este avance es **AlphaGo**, sistema de inteligencia artificial desarrollado por Google, que logró superar al campeón mundial del juego de mesa Go, un reto que hasta hace poco se consideraba inalcanzable para las máquinas debido al alto nivel de intuición que requiere (IAobservatorio, 2016).

En el plano formativo, programas como el **Programa Superior en Big Data de ICEDM** ofrecen a los profesionales las herramientas necesarias para liderar estrategias en la economía digital. En este contexto, el aprendizaje automático se entiende como un conjunto de algoritmos que permiten extraer conocimiento a partir de los datos internos de una organización, con el objetivo de incorporarlo a los procesos de **toma de decisiones** empresariales (ESIC Business & Marketing School, 2018).

De acuerdo con Gómez et al. (2021), la aplicación de estas tecnologías en los sistemas de negocio representa una oportunidad estratégica para optimizar procesos, anticipar tendencias y fortalecer la competitividad empresarial mediante un uso más eficiente de la información disponible.

3.3.2 Inteligencia de negocio y predicciones automatizadas

En su revisión de trabajos publicados entre 2017 y 2021, Cabrera (2023) identifica investigaciones enfocadas en el uso de aprendizaje automático para predicciones empresariales. Entre las técnicas más empleadas destaca el uso de **bosques aleatorios de regresión**, que permiten estimar métricas como proyecciones de ventas y detectar patrones anómalos que podrían indicar fraude. Estos hallazgos constituyen una base sólida para el diseño de sistemas de pronóstico de ventas que integren **analítica predictiva** con herramientas de **inteligencia de negocio**, optimizando así la toma de decisiones y la anticipación de tendencias de mercado.

3.3.3 Inteligencia artificial y logística en la cadena de suministro

La **cadena de suministro** genera diariamente enormes cantidades de información, tanto estructurada como no estructurada, cuyo valor puede ser aprovechado gracias a la **inteligencia artificial** (Nexus Integra, 2020). La logística, sustentada en redes físicas y digitales, presenta un grado de complejidad que dificulta su optimización manual. En este sentido, la IA facilita la transición de procesos reactivos a proactivos, la automatización de tareas que antes eran manuales y la personalización de procedimientos que antes eran estandarizados (Maisueche, 2019).

Según Iglesias (2016), la logística es clave para que las empresas alcancen una posición competitiva, satisfagan las necesidades de sus clientes y prolonguen su relación con ellos. Gómez et al. (2021) afirman que la implementación de aprendizaje automático en la gestión logística no solo mejora la toma de decisiones, sino que también posibilita la anticipación de demandas y la adaptación a cambios en el mercado.

3.4 Clasificación de enfoques AutoML

De acuerdo con Valladares (2023), el AutoML puede clasificarse en:

- **AutoML clásico:** combina la selección de modelos y la optimización automática de hiperparámetros (CASH).

- **AutoML heterogéneo:** integra algoritmos de distinta naturaleza en un mismo flujo de trabajo.
- **AutoML heterogéneo multiobjetivo:** extiende el anterior optimizando varias métricas de forma simultánea, como la precisión y el tiempo de entrenamiento, relevante en escenarios con restricciones de recursos.

3.5 Componentes clave del AutoML

3.5.1 Ciclo de vida de un sistema AutoML

Según Castilla Gil (2021), las técnicas de **ML** se aplican actualmente en campos como la detección de anomalías, la segmentación de clientes, el análisis de abandono, la predicción de la demanda, el mantenimiento predictivo y la optimización de precios.

El ciclo de vida estándar comprende:

1. Adquisición de datos.
2. Preprocesamiento y tratamiento de datos.
3. Gestión del flujo de trabajo.
4. Reentrenamiento del modelo.
5. Despliegue en producción.

La preparación de datos suele ser la etapa más compleja y que requiere mayor inversión de tiempo.

3.5.2 Tipos de AutoML según sus técnicas

Castilla Gil (2021) identifica tres áreas clave para la democratización de la IA:

1. **Ingeniería de características automatizada:** incluye expansión o reducción de datos, transformaciones jerárquicas, *meta-learning* y aprendizaje por refuerzo.
2. **Búsqueda de arquitecturas neuronales (NAS):** emplea algoritmos evolutivos, búsqueda local, *meta-learning*, aprendizaje por transferencia y optimización continua.
3. **Optimización de hiperparámetros:** utiliza técnicas como optimización bayesiana, algoritmos evolutivos, búsqueda aleatoria, enjambres de partículas y *transfer learning*.

3.5.3 Búsqueda de arquitecturas neuronales

Seleccionar el modelo adecuado es un desafío clave. Para tareas de regresión y clasificación existen diversas opciones, como regresión lineal o logística, *random forest*, XGBoost, AdaBoost y SVM, entre otras (Castilla Gil, 2021).

En la **NAS**, se define un espacio de búsqueda de posibles arquitecturas, se implementa una estrategia que equilibre exploración y explotación, y se evalúan las candidatas mediante entrenamiento y validación.

3.6 Ventajas y desventajas del AutoML

En el ámbito de la automatización del aprendizaje automático, el uso de AutoML presenta beneficios y limitaciones (Castilla Gil, 2021).

Ventajas:

1. Reducción de tiempos de desarrollo.
2. Actualización tecnológica constante por parte de los proveedores.
3. Disminución de la complejidad técnica y menor dependencia de expertos especializados.

Desventajas:

1. Menor comprensión del funcionamiento interno del modelo y de los datos.
2. Limitada flexibilidad ante cambios de requisitos.
3. Posibles costos elevados en entornos de gran escala.
4. Calidad potencialmente inferior frente a modelos personalizados.

3.7 Retos y vacíos de investigación

Aunque el AutoML presenta beneficios evidentes, enfrenta retos como mejorar la **interpretabilidad** de los modelos, adaptarse a **dominios muy especializados**, optimizar su rendimiento en entornos con **recursos limitados** y reducir el **costo computacional** en procesos de gran escala. Estas limitaciones justifican la investigación de propuestas que integren la automatización con enfoques personalizados y eficientes, ajustados a las necesidades de cada aplicación específica.



Capítulo IV: Desarrollo del proyecto

4.- Procedimiento y Descripción de las actividades desarrolladas.

El desarrollo del proyecto “Desarrollo de una Plataforma Web para un AutoML” se llevó a cabo siguiendo un enfoque sistemático y metódico que permitió estructurar de manera ordenada todas las actividades realizadas a lo largo del periodo de residencia profesional. Se implementó una metodología ágil basada en iteraciones cortas, enfocada en la entrega continua de avances funcionales, la retroalimentación frecuente de los asesores y la mejora constante a través de pruebas y revisiones.

Desde la etapa inicial, se priorizó una estrategia de trabajo que combinara análisis detallado, diseño técnico, desarrollo progresivo, pruebas exhaustivas y despliegue controlado, siempre alineado a los requerimientos institucionales y al entorno académico en el que se desarrolló el proyecto. Las actividades se calendarizaron semanalmente, lo que permitió un control preciso del avance y la identificación oportuna de desviaciones, facilitando su corrección inmediata.

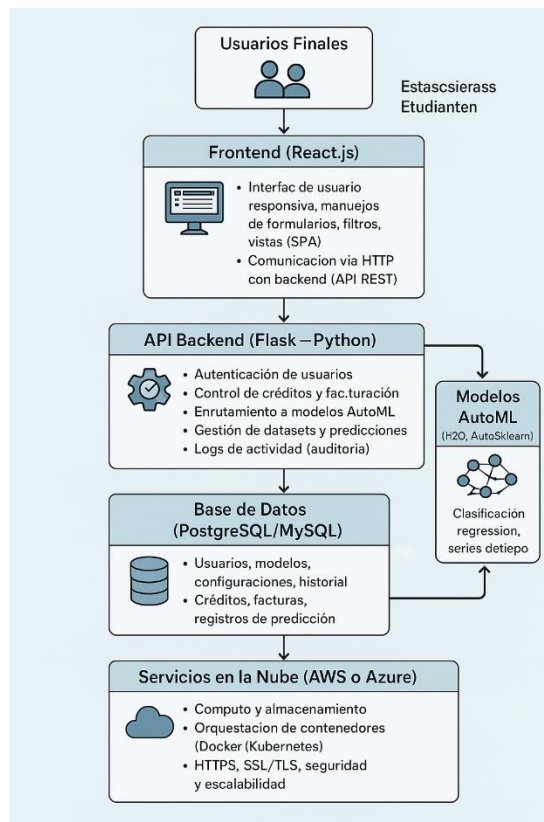


Figura 2: Diagrama de arquitectura

Se implementó una API funcional orientada a la interacción con modelos de aprendizaje automático previamente entrenados. Esta API permitió realizar diferentes tipos de peticiones para consultar los resultados del entrenamiento de modelos, utilizando tres datasets principales:

El procedimiento se dividió en seis fases principales, cada una con tareas específicas y objetivos definidos:

1. Fase de Planeación y Organización

Esta fase fue crítica para establecer las bases del proyecto. Se definió el alcance general, los objetivos específicos y se elaboró un cronograma con fechas clave. Se llevaron a cabo reuniones iniciales con los asesores interno y externo, en donde se acordaron los entregables por semana y se determinaron las herramientas tecnológicas a utilizar durante el desarrollo del sistema.

Además, se realizó una exploración del entorno de aplicación, considerando las necesidades de los usuarios y las capacidades actuales del Instituto Tecnológico de Culiacán en cuanto a infraestructura tecnológica. A partir de ello, se elaboró un documento de requerimientos funcionales y no funcionales, que sirvió como guía base para las fases posteriores.

2. Fase de Análisis de Requerimientos y Arquitectura

En esta etapa se analizó en profundidad el problema a resolver. Se identificaron los actores principales, se definieron las funcionalidades requeridas del sistema y se estructuraron los casos de uso que reflejaban las interacciones previstas entre los usuarios y la plataforma.

Se comenzó a modelar el sistema a través de diagramas UML, como el diagrama de casos de uso, diagrama de clases y diagrama de actividades, utilizando herramientas como Visual Paradigm. También se propuso una arquitectura cliente-servidor, donde el frontend en React se comunicaría con un backend en Python (Flask), a través de APIs RESTful, las cuales a su vez gestionarían los modelos de AutoML en un entorno de computación en la nube. Esta arquitectura aseguraría modularidad, escalabilidad y mantenimiento eficiente del sistema.

3. Fase de Diseño de Interfaces y Base de Datos

En esta fase se enfocó el trabajo en el diseño de la interfaz gráfica del usuario (UI), con el objetivo de ofrecer una experiencia sencilla, intuitiva y eficiente. Se crearon prototipos visuales con Canva, los cuales fueron presentados a los asesores para su validación y retroalimentación. Se incorporaron

principios de experiencia de usuario (UX) y diseño responsivo, permitiendo el uso del sistema desde diversos dispositivos.

Paralelamente, se diseñó la estructura de la base de datos, con un modelo relacional en MySQL. Se definieron las tablas, relaciones y atributos que permitirían almacenar usuarios, modelos AutoML, configuraciones, resultados, historial de uso, créditos y facturación. Este diseño consideró las mejores prácticas de normalización, seguridad de datos y escalabilidad.

4. Fase de Desarrollo del Sistema

Esta fue la etapa de mayor duración y complejidad. Se dividió en dos grandes bloques: desarrollo del frontend y desarrollo del backend.

Frontend (React.js): Se desarrollaron componentes funcionales y reutilizables, como formularios, menús, dashboards e interfaces de selección de instancias. Se integraron librerías modernas de estilos y validación de formularios, así como enrutamiento dinámico para la navegación entre secciones.

Backend (Python/Flask): Se crearon las APIs RESTful que manejaban las operaciones del sistema, como autenticación, carga de datos, ejecución de modelos AutoML, gestión de resultados y facturación. Se implementó lógica de negocios para controlar el flujo de solicitudes, el consumo de créditos, y la interacción con la base de datos.

Conexión entre capas: Se conectó el frontend con el backend, y este con los modelos entrenados de AutoML, permitiendo una integración completa que facilitara pruebas funcionales a nivel sistema. Además, se gestionó el control de versiones del proyecto mediante GitHub.

5. Fase de Pruebas y Validación

En esta fase se llevaron a cabo pruebas exhaustivas para validar la calidad y el correcto funcionamiento del sistema desarrollado. Se ejecutaron pruebas unitarias en ambos extremos (frontend y backend) utilizando Jest y Pytest respectivamente, y se aplicaron pruebas de integración para validar la comunicación y consistencia entre las capas del sistema.

También se realizaron simulaciones de uso real del sistema, con escenarios completos de carga de datasets, selección de modelos, ejecución de predicciones y visualización de resultados. Se revisó el cumplimiento de los requisitos funcionales y se documentaron errores encontrados, los cuales fueron corregidos en tiempo y forma.

6. Fase de Implementación y Despliegue

La fase final consistió en el despliegue del sistema en la nube, utilizando servicios como Amazon Web Services (AWS) o Microsoft Azure. Se configuraron instancias de cómputo, bases de datos remotas, almacenamiento seguro y certificados SSL para habilitar el acceso mediante HTTPS.

La aplicación fue subida a un entorno de producción, donde se validó su funcionamiento desde diferentes dispositivos y navegadores. Se crearon manuales técnicos y de usuario final, y se organizó una presentación final del sistema a los asesores como cierre del proyecto.

Esta última etapa también incluyó la recopilación de evidencias del desarrollo, como capturas de pantalla, registros de uso, y código documentado, los cuales fueron incluidos como anexos del presente informe.

4.1.- Determinación de tiempos de las actividades.

En la Tabla 1 se ilustra la distribución de las actividades a lo largo de las semanas del proyecto, destacando con colores las semanas en las que se llevaron a cabo evaluaciones de progreso.

Evaluación y Seguimientos											1	2				3	Entrega				
Actividades	Semanas	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16				
1	Planeación y Organización																				
3	Diseño de las vistas de usuario																				
3	Requerimientos y Arquitectura																				
4	Desarrollo del sistema																				
5	Pruebas y Validación																				
6	Implementacion y despliegue																				

Tabla 1: Cronograma de actividades del proyecto

4.2.- Descripción de las Actividades realizadas.

Durante el desarrollo del proyecto de residencia titulado “**Desarrollo de una Plataforma Web para un AutoML**”, se llevaron a cabo diversas actividades distribuidas a lo largo de un periodo de 14 semanas, cada una planificada estratégicamente en el cronograma de trabajo. Estas actividades se organizaron en fases que respondían a los objetivos establecidos, y se desarrollaron siguiendo un enfoque ágil e iterativo que permitió ajustes constantes y mejora progresiva del sistema.

A continuación, se presenta la descripción detallada de cada actividad realizada por semana:

Semanas 1 - 2: Planeación y levantamiento de requerimientos

- Se realizaron reuniones iniciales con el asesor interno y externo para definir el alcance del proyecto, sus objetivos específicos, y su impacto dentro del Instituto Tecnológico de Culiacán.
- Se establecieron los entregables semanales y se diseñó un cronograma detallado que incluía fechas de revisión y validación de avances.
- Se levantaron los **requerimientos funcionales y no funcionales** mediante entrevistas estructuradas y revisión de necesidades institucionales.
- Se identificaron los usuarios meta: docentes, investigadores y estudiantes que pudieran beneficiarse del uso de modelos AutoML sin conocimientos técnicos avanzados.
- Se seleccionaron las herramientas y tecnologías a utilizar, incluyendo React para frontend, Python con Flask para backend, y MySQL como sistema gestor de base de datos.

Semanas 3 - 4: Análisis del sistema y modelado lógico

- Se diseñaron los **diagramas de casos de uso** generales y extendidos para modelar todas las posibles interacciones entre el usuario y el sistema.
- Se definieron los **objetos del dominio**, tales como Usuario, ModeloAutoML, Instancia, Dataset, Predicción y Historial.
- Se identificaron clases de servicio necesarias para la comunicación entre frontend y backend.
- Se diseñó el modelo entidad-relación de la base de datos, considerando la escalabilidad, integridad referencial y seguridad.
- Se analizaron los flujos de eventos normales y alternativos, los errores esperados y las validaciones necesarias para una interacción robusta.
- Se realizó un análisis de viabilidad técnica para la integración de modelos AutoML utilizando librerías como AutoSklearn y H2O.ai.

Semanas 5 - 6: Diseño de interfaces de usuario y arquitectura

- Se crearon los **primeros prototipos visuales** en Canva, incluyendo pantallas para inicio de sesión, visualización de resultados, catálogo de modelos, sección de facturación y monitoreo.
- Se aplicaron principios de **UX/UI**, como consistencia visual, responsividad y accesibilidad.
- Los prototipos fueron validados con el asesor interno y se iteraron con base en su retroalimentación.
- Se diseñó el esquema técnico de comunicación frontend-backend, basado en arquitectura cliente-servidor, y se establecieron rutas RESTful.
- Se definió el comportamiento esperado del sistema bajo condiciones normales y excepcionales (ej. errores del modelo, entradas inválidas, carga excesiva).

Semanas 7 - 11: Desarrollo de la plataforma web

Frontend (React.js):

- Se desarrollaron componentes dinámicos y reutilizables, como formularios de carga, tablas interactivas, gráficas y paneles de usuario.
- Se implementaron validaciones en los formularios y visualizaciones responsivas para distintos dispositivos.
- Se creó la lógica de navegación entre vistas, mediante React Router, y se integró el sistema de autenticación.
- Se incorporaron librerías externas como Bootstrap, Axios y Chart.js para mejorar el diseño y la funcionalidad.

Backend (Python Flask):

- Se programaron rutas para autenticación, predicciones, administración de modelos y gestión de créditos.
- Se conectó el backend con la base de datos MySQL utilizando ORM (Object Relational Mapping).
- Se desarrollaron funciones específicas para procesar peticiones de predicción sobre modelos ya entrenados, utilizando modelos tipo classification, regression y time series.
- Se integraron logs de auditoría para registrar las acciones del usuario y se configuraron respuestas HTTP personalizadas ante errores.

Base de datos:

- Se implementaron tablas y relaciones necesarias para usuarios, configuraciones, facturas, logs, modelos y peticiones realizadas.
- Se probó la integridad de datos mediante pruebas de inserción, actualización y eliminación de registros.

Interacción entre capas:

- Se integraron completamente el frontend y el backend, permitiendo simular flujos reales de trabajo como: cargar datos, seleccionar instancias, ejecutar predicciones y visualizar resultados.

Semanas 12 - 13: Pruebas del sistema e iteración final

- Se realizaron **pruebas unitarias** con Jest (para React) y Pytest (para Flask), validando el correcto funcionamiento de cada módulo y componente.
- Se ejecutaron **pruebas de integración**, verificando la consistencia de datos, la velocidad de respuesta, el manejo de errores, y el flujo entre cliente, servidor y modelo.
- Se evaluó el sistema bajo múltiples escenarios de carga y simulaciones de uso real.
- Se corrigieron errores identificados en la validación de entradas, en los formularios y en el comportamiento de las rutas protegidas.
- Se recopilaron resultados de pruebas para presentarlos como evidencia en la entrega final.

Semana 14: Despliegue en la nube y documentación técnica

- Se desplegó la plataforma en un servidor de nube (AWS o Azure), configurando dominios, bases de datos remotas, seguridad HTTPS, y certificados digitales.



- Se probaron accesos desde navegadores de escritorio y móviles, validando compatibilidad multiplataforma.
- Se configuraron servicios complementarios como monitoreo de logs, historial de peticiones y gestión de pagos simulados.
- Se elaboraron dos manuales: uno técnico (para mantenimiento del sistema) y uno de usuario final (para docentes e investigadores).
- Se preparó la documentación para la entrega institucional, incluyendo capturas, diagramas, scripts y enlaces al repositorio del proyecto.



Capítulo V: Pruebas y resultados

5.1.- Resultados, planos, gráficas, prototipos, manuales, programas,

Se implementó una API funcional orientada a la interacción con modelos de aprendizaje automático previamente entrenados. Esta API permitió realizar diferentes tipos de peticiones para consultar los resultados del entrenamiento de modelos, utilizando tres datasets principales:

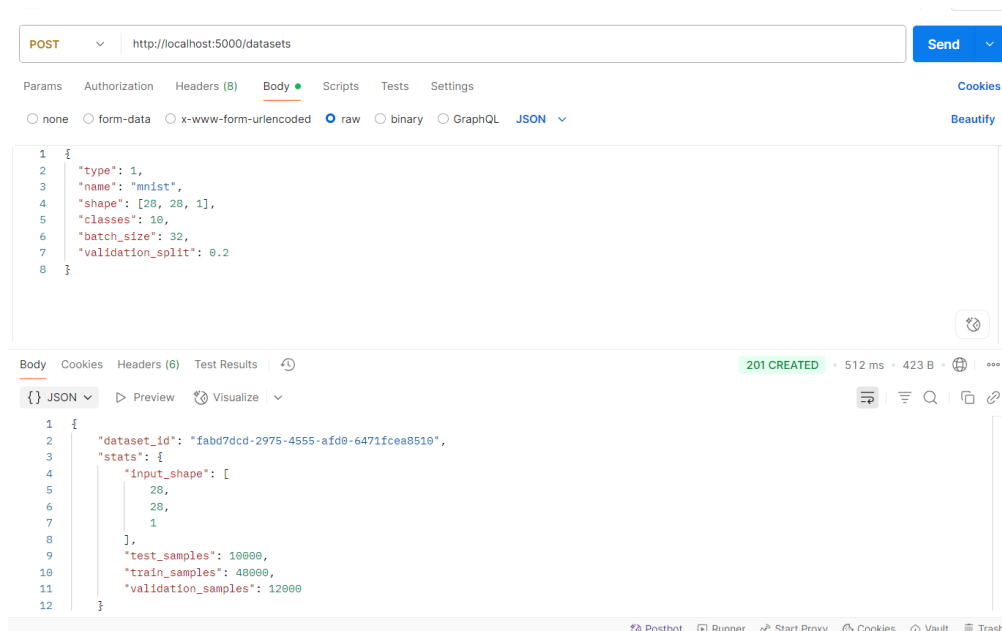


Figura 2: Imagen de la petición del dataset Image

Este dataset fue empleado para entrenar un modelo de clasificación de imágenes. A través de la API, se podían enviar peticiones que simulaban el proceso de entrenamiento y análisis del modelo ya entrenado, facilitando la visualización de sus resultados.

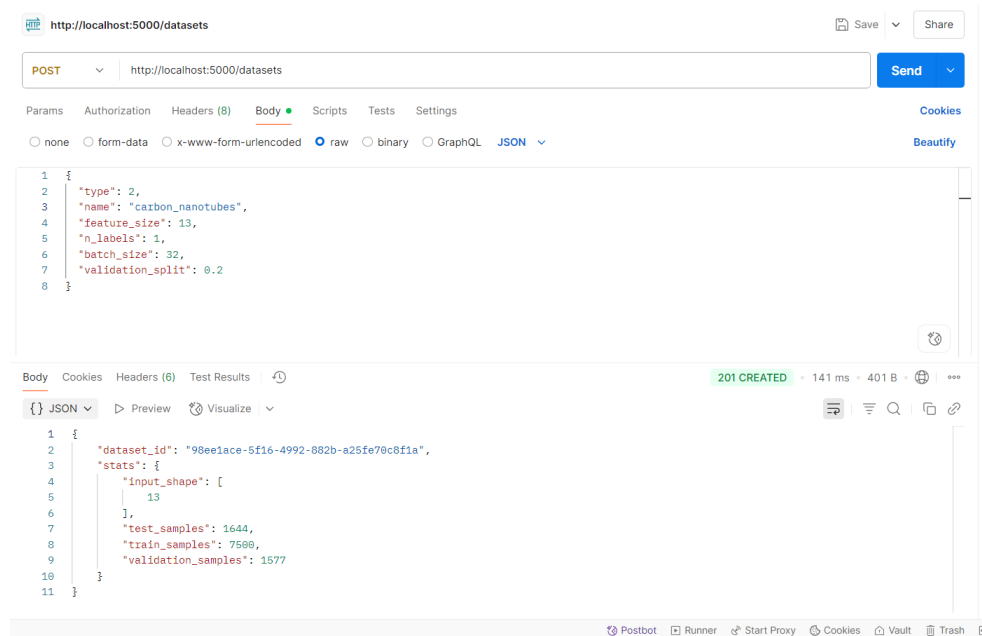


Figura 3: Imagen de la petición del dataset regression

Se integró un modelo de regresión entrenado con un segundo dataset, enfocado en predecir valores numéricos.

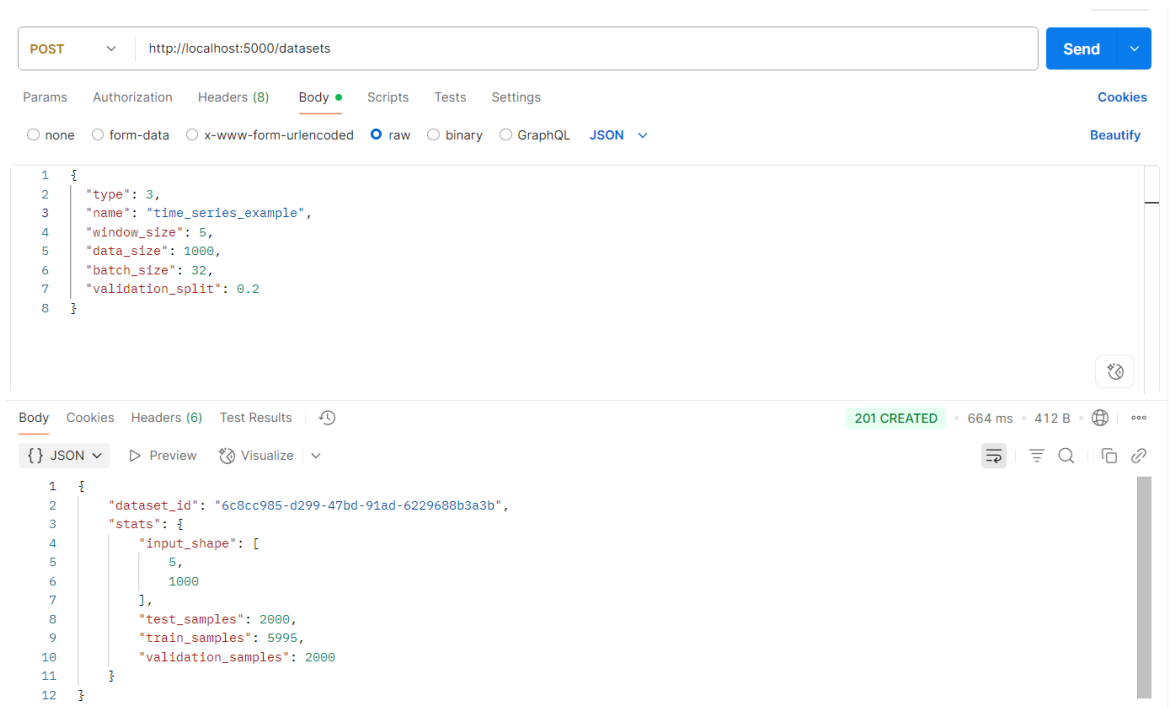


Figura 4: Imagen de la petición del dataset TimeSeries

Para datos secuenciales, se entrenó un modelo basado en series temporales. Este modelo también fue expuesto mediante la API, permitiendo simular predicciones de datos en el tiempo, como por ejemplo tendencias de consumo o valores financieros.

```
select * from api_requests
```

id	timestamp	endpoint	http_method	request_body	response_code	response_body
[PK] character varying (36)	timestamp with time zone	text	text	jsonb	integer	jsonb
2b9368bf-8cab-4b14-8e2f-5174eab44f...	2025-04-28 16:08:02.759629-06	/datasets	POST	{ "name": "cifar10", "type": 1, "shape": [32, 32, 3], "classes": 10, "batch_size": 64, "validation_split": 0.2 }	201	{ "stats": { "input_shape":
bd9bf3ed-3960-429c-a67d-8ceca3b2db...	2025-04-28 16:09:39.075887-06	/datasets	POST	{ "name": "cifar10", "type": 1, "shape": [32, 32, 3], "classes": 10, "batch_size": 64, "validation_split": 0.2 }	201	{ "stats": { "input_shape":
24593dee-0f39-42a3-9853-6ac756818c...	2025-04-28 16:11:24.181228-06	/datasets/typ...	GET	[null]	200	{ "types": { "1": "imageCl
b6b514b6-0ced-497f-9b03-e798742f49...	2025-04-28 16:11:25.23132-06	/datasets/1/list	GET	[null]	200	{ "total": 18, "datasets":
ebb0e460-e572-48ba-b2ed-b3da4c82f8...	2025-04-28 16:11:27.237108-06	/datasets/typ...	GET	[null]	200	{ "types": { "1": "imageCl
0c8c5441-66ab-43db-8e2c-26fb193db9...	2025-04-28 16:11:28.281818-06	/datasets/2/list	GET	[null]	200	{ "total": 5, "datasets": [
abab6f07-407c-4661-9e50-3990dcdade...	2025-04-28 18:18:52.774642-06	/datasets	POST	{ "name": "cifar10", "type": 1, "shape": [32, 32, 3], "classes": 10, "batch_size": 64, "validation_split": 0.2 }	201	{ "stats": { "input_shape":
f2b3e9bc-cfd8-4b5a-9a4c-e3d975a000...	2025-05-29 21:28:21.889215-06	/datasets/typ...	GET	[null]	200	{ "types": { "1": "imageCl
02291f9a-3c9a-4f4f-aeae-11df3e40bd3	2025-05-29 21:28:23.61275-06	/datasets/1/list	GET	[null]	200	{ "total": 18, "datasets":
0ea927c5-a9d2-4020-9958-900fa0ce87...	2025-05-29 21:28:25.852681-06	/datasets/typ...	GET	[null]	200	{ "types": { "1": "imageCl
aeccc975-9dc0-4993-99bf-907bae263c...	2025-05-29 21:28:54.000049-06	/datasets/typ...	GET	[null]	200	{ "types": { "1": "imageCl
f337981b-9bd1-4f6a-a947-6c678e9beff1	2025-05-29 23:29:26.157647-06	/datasets/typ...	GET	[null]	200	{ "types": { "1": "imageCl
8f35c203-8252-48d2-8ab2-e7ab38d3c1...	2025-05-29 23:29:27.919755-06	/datasets/1/list	GET	[null]	200	{ "total": 18, "datasets":
609caf71-b5da-4303-a06b-240ebab5a5...	2025-05-29 23:29:35.627525-06	/jobs	POST	{ "dataset_id": "313" }	404	{ "error": "Dataset ID no

Showing rows: 1 to 53 | Page No: 1 of 1

1 rows: 53 | Query complete 00:00:00.363 | CRLF | Ln 1, Col 27

Figura 5: Base de datos de las peticiones de la API

Se creó una base de datos conectada directamente a la API, con el fin de almacenar cada una de las peticiones realizadas. Esto facilitó el registro y análisis de uso de la herramienta.

Plataforma web para rentar modelos

[Docs](#) [Hosting](#) [Pricing](#) [FAQ](#) [Blog](#) [Clusters](#) [Careers](#) [About](#) [Contact Sales](#) [Console](#)

Plataforma de Renta de Modelos de IA

Accede a modelos de Machine Learning preentrenados y listos para usar.
Renta modelos predictivos de manera fácil y flexible, sin necesidad de conocimientos técnicos avanzados.

[Comenzar ahora](#) [Solicitar demostración](#)

Figura 6: Ventana informativa

Se diseñó una ventana de presentación que brindaba información general sobre la plataforma, explicando de manera breve y clara su objetivo y funcionamiento. Esta sección sirvió como introducción para nuevos usuarios, permitiéndoles comprender el valor de la plataforma sin necesidad de registros previos.

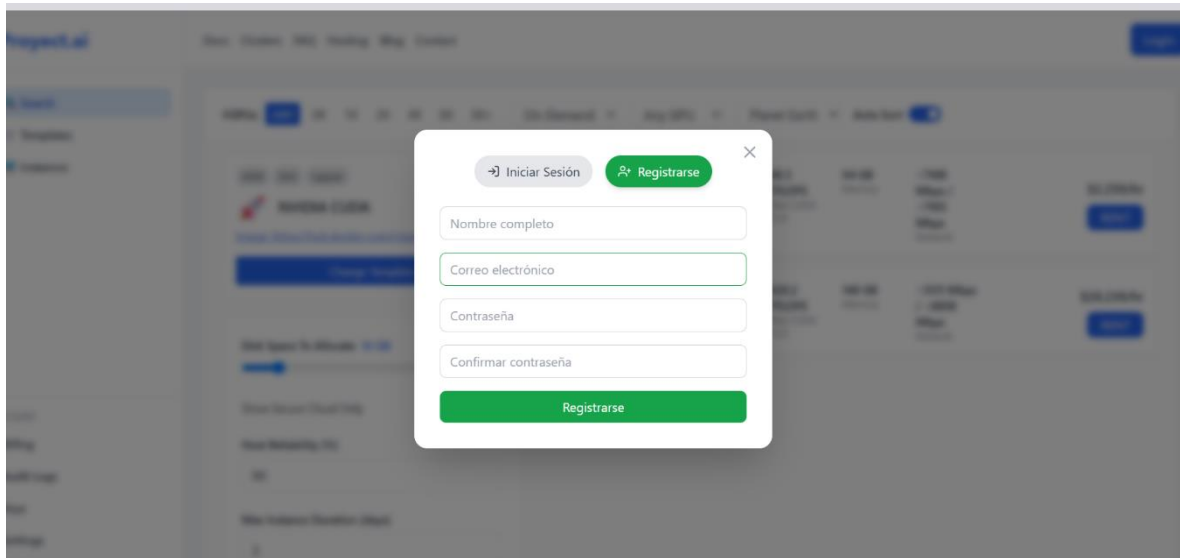


Figura 7: Inicio de sesión de la pagina

En esta página se agregaron los campos para que el usuario ingrese su código de usuario y su clave para iniciar sesión o también registrarte.

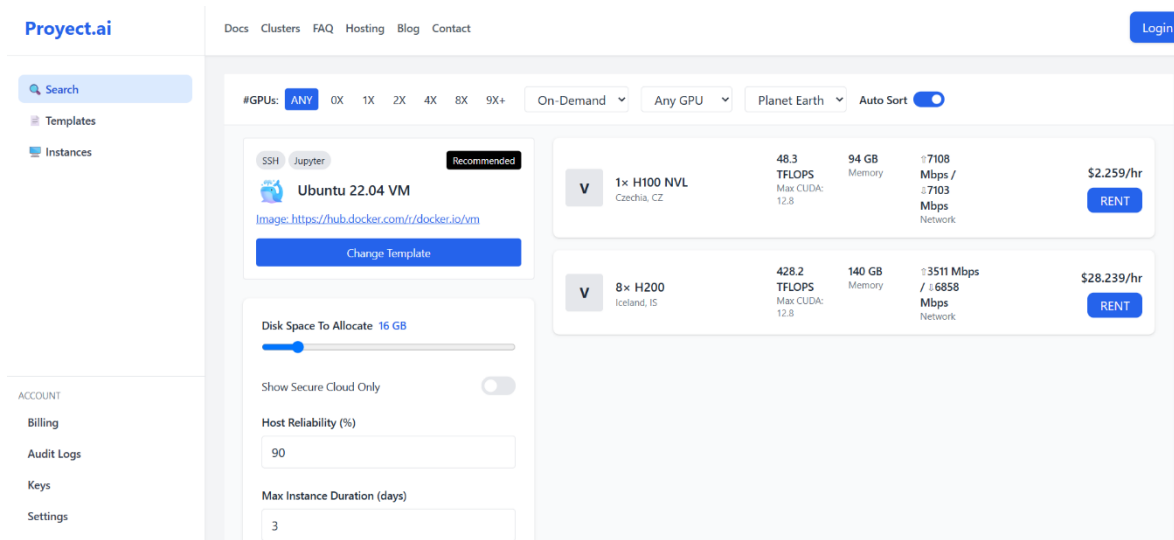


Figura 8: Pagina de renta de modelos

Search En esta sección, los usuarios podían explorar las instancias disponibles para ejecutar modelos de aprendizaje automático. Se incluyeron filtros dinámicos que permitieron buscar por tipo de GPU, cantidad de CPU, memoria RAM, almacenamiento, ubicación geográfica y tipo de instancia (on-demand o interruptible). Asimismo, se añadió la opción de ordenar los resultados por precio, rendimiento (benchmarks) o latencia, facilitando una selección optimizada.

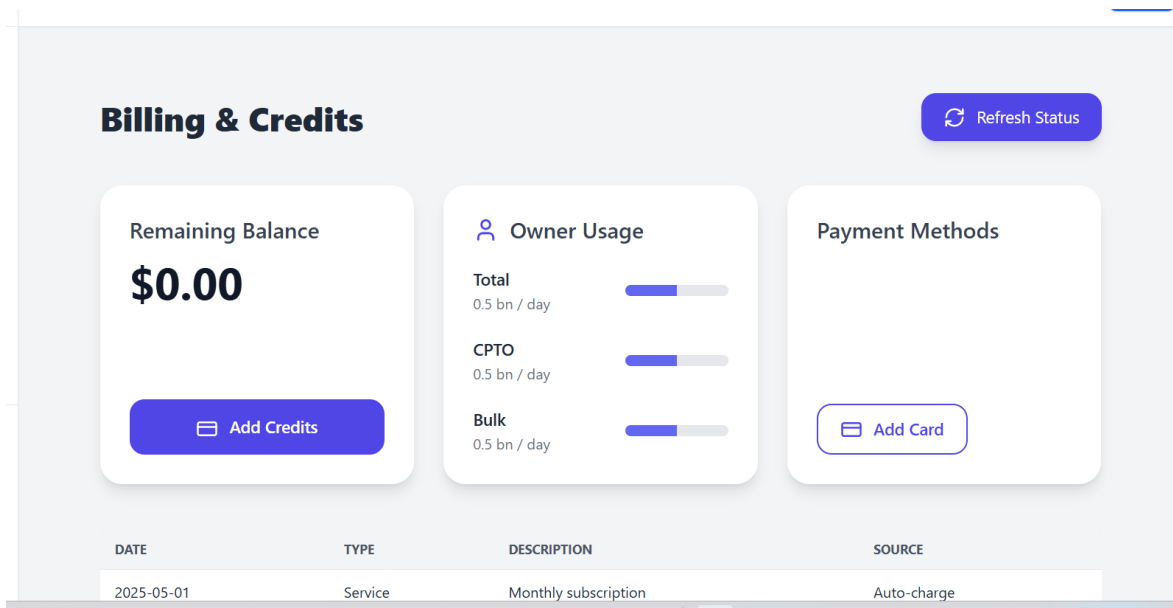


Figura 9: Pagina de créditos y facturación

En **Billing**, Se diseñó una sección para gestionar el sistema de créditos del usuario. Esta funcionalidad permitía visualizar el saldo actual, revisar el historial de consumo, consultar tarifas por hora y descargar facturas anteriores. Además, se implementó un módulo para recargar créditos mediante diferentes métodos de pago, como tarjetas bancarias o criptomonedas. Esta sección resultó fundamental para que los usuarios tuvieran control total sobre su gasto y pudieran planificar el uso de recursos.

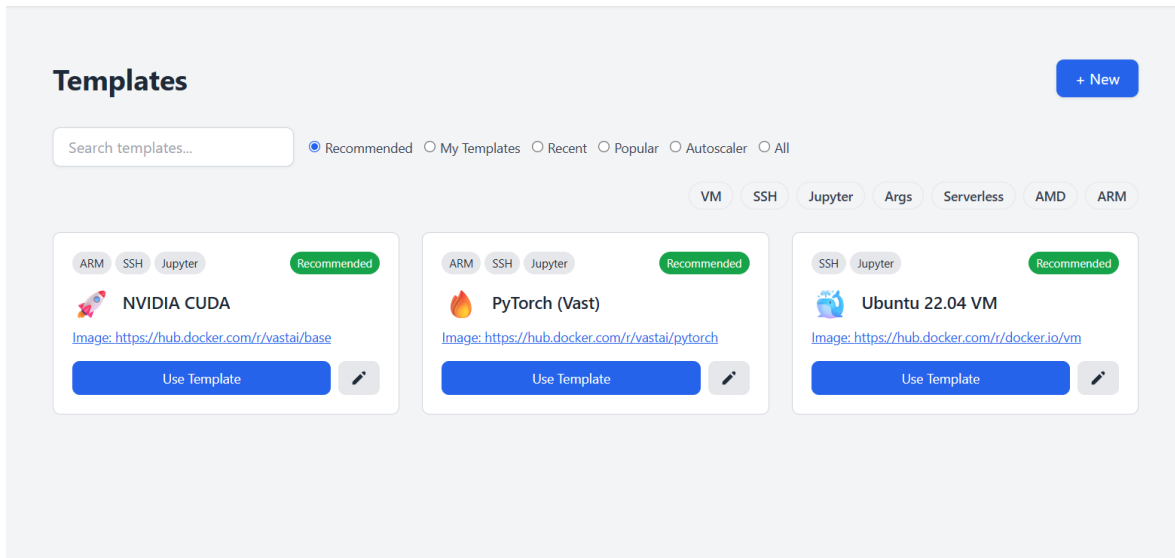


Figura 10: Pagina de catálogo de templates

En **Templates**, Se integró un catálogo de entornos preconfigurados con herramientas como Docker, Jupyter Notebook y TensorFlow. Estas plantillas fueron diseñadas para que los usuarios pudieran lanzar servidores listos para trabajar, sin tener que preocuparse por instalaciones o configuraciones manuales. Esto redujo significativamente el tiempo de preparación y permitió enfocarse directamente en el desarrollo o entrenamiento de modelos.

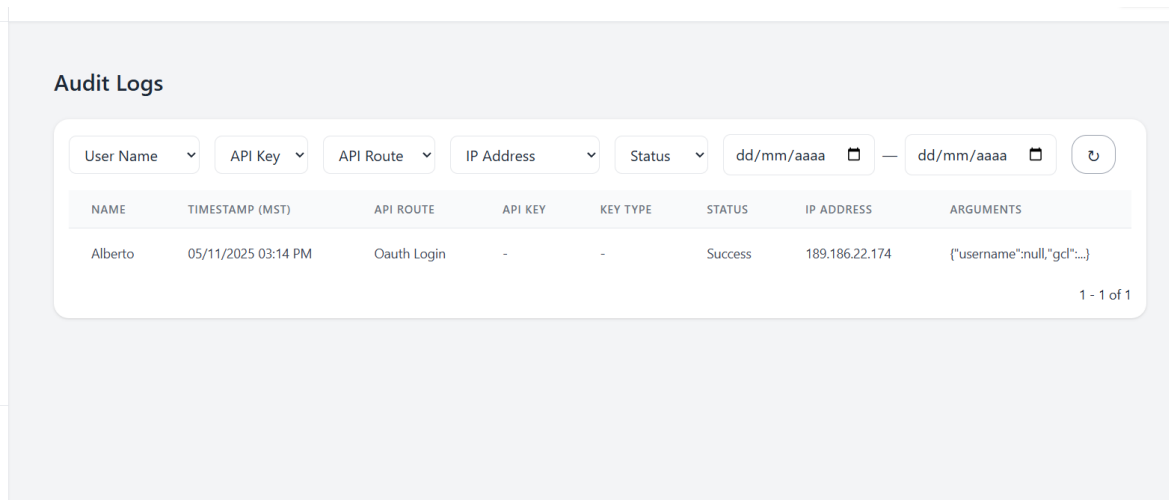


Figura 11: Pagina de monitoreo

la sección de **Audit Logs**: Se desarrolló una funcionalidad para el monitoreo de seguridad y trazabilidad. En esta sección se almacenaron todas las acciones importantes realizadas por los usuarios, lo cual permitió mantener un control sobre el comportamiento de las cuentas y detectar

posibles accesos no autorizados. Este módulo fue especialmente útil para mantener la transparencia y cumplimiento normativo.

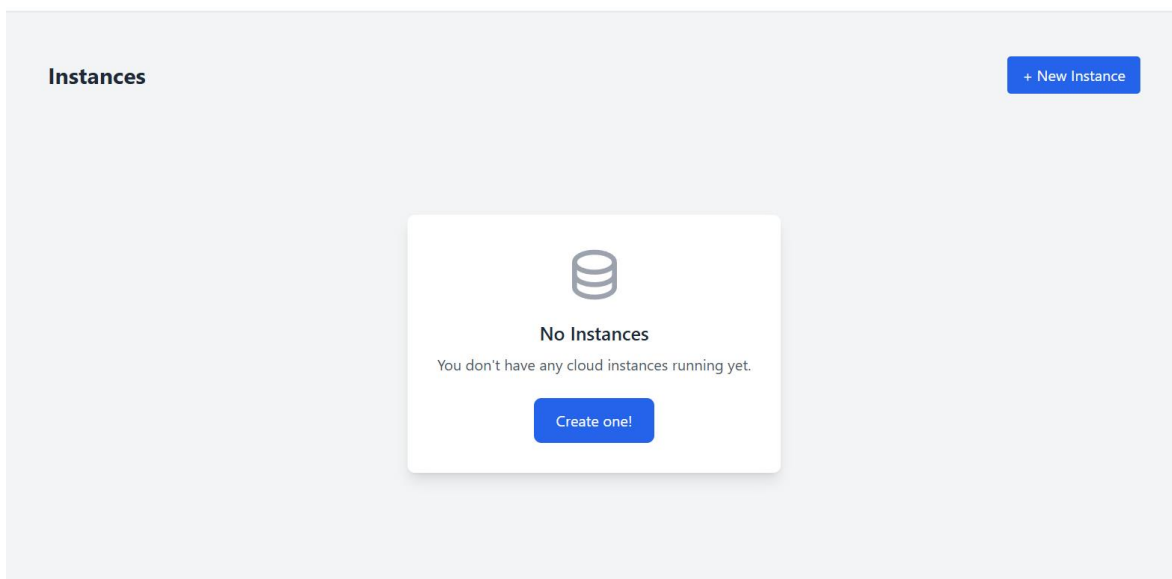


Figura 12: Pagina de instancias

La sección **Instances**: Aquí los usuarios podían ver el estado de las máquinas virtuales que tenían activas o en pausa. Desde esta sección, se podían realizar acciones como iniciar, detener, reiniciar o eliminar instancias, así como monitorear su rendimiento y tiempo de uso.



Capítulo VI: Conclusiones

6.1. Conclusiones del proyecto.

La solución desarrollada logró trasladar la complejidad técnica del aprendizaje automático automatizado (AutoML) a un entorno gráfico intuitivo, permitiendo que usuarios sin conocimientos avanzados puedan gestionar datos, iniciar procesos de entrenamiento y generar predicciones de manera autónoma y eficiente.

La combinación de algoritmos genéticos con un backend desacoplado y alojado en la nube demostró ser una estrategia adecuada para garantizar flexibilidad y adaptabilidad frente a distintos volúmenes de datos y escenarios de uso. Este diseño mantuvo un rendimiento estable incluso bajo condiciones de alta demanda, gracias a la incorporación de mecanismos de balanceo de carga y control de errores.

El enfoque modular de la arquitectura aportó ventajas en mantenibilidad, escalabilidad y facilidad para la integración de nuevos componentes. Las pruebas de integración confirmaron la coherencia del flujo de datos entre las distintas capas de la aplicación, validando la solidez y estabilidad del sistema implementado.

En síntesis, los resultados obtenidos confirman que la integración de tecnologías web modernas, despliegue en la nube y técnicas de AutoML constituye una estrategia viable para democratizar el acceso a la inteligencia artificial en contextos académicos, empresariales y de investigación aplicada, cumpliendo con los objetivos planteados en la introducción del trabajo.

6.2. Recomendaciones

Con base en los resultados y la experiencia adquirida, se proponen las siguientes acciones para optimizar y ampliar el alcance de la solución:

- **Plataforma de renta de modelos:** Desarrollar un sistema web que permita a usuarios e instituciones acceder y alquilar modelos entrenados, con control de licencias, gestión de accesos y facturación automatizada.
- **Optimización del backend y escalabilidad:** Implementar arquitecturas de microservicios o serverless, gestión avanzada de tareas asíncronas y balanceo de carga para asegurar disponibilidad y rendimiento bajo alta concurrencia.
- **Ampliación de librerías y algoritmos AutoML:** Integrar soluciones como TPOT, AutoGluon y otras herramientas para diversificar los métodos de entrenamiento y optimización.
- **Preprocesamiento y explicabilidad:** Incorporar módulos de ingeniería de características automática, detección/manejo de anomalías y herramientas de interpretación de modelos (XAI).
- **Gestión de modelos:** Implementar un sistema robusto de versionado, monitoreo de rendimiento y reversión de modelos para garantizar la calidad del servicio.

6.3 Trabajo Futuro

De cara a la evolución del proyecto, se plantea una visión estratégica que extienda las capacidades actuales hacia un modelo de negocio sostenible y de mayor impacto:

1. **Renta de modelos AutoML como servicio (MLaaS)**
 - Implementar un sistema en la plataforma que permita a usuarios e instituciones alquilar modelos entrenados por tiempo o por volumen de uso.
 - Incluir un sistema de licencias, control de acceso y facturación automatizada para asegurar un manejo eficiente y seguro.
2. **Infraestructura escalable en la nube**
 - Migrar hacia arquitecturas **serverless** o de **microservicios** que soporten picos de demanda sin afectar el rendimiento.
 - Incorporar servicios de despliegue continuo (CI/CD) y monitoreo proactivo.
3. **Integración con MLOps avanzado**
 - Añadir pipelines automáticos para entrenamiento, evaluación y despliegue de modelos, así como monitoreo en tiempo real para detectar drifts y pérdida de precisión.
4. **Expansión a nuevos dominios de aplicación**
 - Adaptar la plataforma para sectores como agricultura de precisión, predicción de fallos industriales, salud preventiva y optimización logística.
5. **Marketplace de modelos**
 - Crear un catálogo donde desarrolladores puedan publicar y ofrecer sus propios modelos entrenados, permitiendo a los usuarios adquirir o alquilar soluciones listas para usar.

Con estas proyecciones, la plataforma no solo mantendría su relevancia técnica, sino que evolucionaría hacia un ecosistema de inteligencia artificial colaborativo, rentable y en constante actualización.

6.4. Experiencia personal profesional adquirida.

Durante el periodo de residencia profesional experimenté un crecimiento significativo como futuro profesional. A lo largo de estos meses, fortalecí mis conocimientos y habilidades en desarrollo web, especialmente con frameworks como React y la integración de APIs RESTful, además de profundizar en el uso de Python para el backend y librerías de Machine Learning y AutoML. Pude aplicar de forma práctica la inteligencia artificial y aprendí a gestionar entornos de desarrollo en la nube y a optimizar aplicaciones distribuidas. Mi inglés técnico mejoró notablemente al consultar documentación y referencias académicas en este idioma.

Más allá de lo técnico, aprendí a gestionar proyectos con metodologías ágiles, mejoré mi organización y comprendí la importancia de los plazos y la calidad. Afronté retos reales, resolviendo problemas mediante creatividad y pensamiento crítico. La interacción con asesores y compañeros me permitió valorar la colaboración, la comunicación efectiva y el liderazgo compartido.

Aunque encontré desafíos, como la estimación inadecuada de tareas y algunos problemas de comunicación, estos me ayudaron a mejorar mi planificación y a

establecer canales más claros. La experiencia me enseñó a ser proactivo y a valorar la retroalimentación constructiva.

El ambiente de trabajo fue enriquecedor, destacando la figura de mi jefe inmediato, un líder motivador y comprometido que fomentó la confianza y el aprendizaje constante.

Finalmente, analizar los procesos internos de la empresa (División de Estudios de Posgrado e Investigación del Tecnológico Nacional de México Campus Culiacán) me permitió entender la importancia de la flexibilidad y la adaptabilidad en proyectos de innovación, a pesar de la falta de un manual formal de procesos.

En resumen, la residencia fue una experiencia excelente que fortaleció mis conocimientos técnicos y habilidades blandas, brindándome las herramientas necesarias para enfrentar con éxito futuros retos profesiona

6.4. Competencias desarrolladas y/o aplicadas.

- Implementa aplicaciones computacionales para solucionar problemas de diversos contextos, integrando diferentes tecnologías, plataformas o dispositivos.
- **Cómo se trabajó:** Esta competencia fue el eje central de mi participación en el proyecto. Mi rol implicó la implementación de una aplicación web completa, que requería la integración de múltiples tecnologías y plataformas. Se desarrolló un frontend interactivo utilizando un framework moderno (e.g., React.js o similar), el cual se comunicaba con un backend robusto construido en Python. Este backend, a su vez, integraba y orquestaba el uso de librerías de AutoML (como Auto-Sklearn o H2O.ai) y se desplegaba en un entorno de nube (por ejemplo, AWS o Azure). La solución diseñada es accesible desde cualquier dispositivo con navegador web, demostrando mi capacidad para ensamblar diversas tecnologías y plataformas para crear una herramienta funcional que resuelve la complejidad de la gestión de modelos de Machine Learning para usuarios no expertos. Los desafíos de interoperabilidad y comunicación entre estos diferentes componentes fueron superados mediante la aplicación de patrones de diseño de sistemas distribuidos.
- Diseña, desarrolla y aplica modelos computacionales para solucionar problemas, mediante la selección y uso de herramientas matemáticas.
- **Cómo se trabajó:** Aunque la plataforma utiliza AutoML para automatizar gran parte del proceso de modelado, mi involucramiento requirió un entendimiento profundo de los principios subyacentes del Machine Learning. Se diseñó la lógica para configurar y lanzar experimentos de AutoML, lo que implicaba una comprensión de cómo los modelos seleccionan y aplican algoritmos que tienen una base matemática sólida (e.g., regresión lineal, árboles de decisión, redes neuronales, estadística, optimización). Fui responsable de la visualización y correcta interpretación de métricas de rendimiento (como precisión, recall, F1-score, o error cuadrático medio), las cuales son el resultado de cálculos matemáticos. Esta experiencia fortaleció mi habilidad para aplicar el razonamiento lógico y matemático en la formulación de problemas y en la interpretación de soluciones basadas en datos.
- Diseña e implementa interfaces para la automatización de sistemas de hardware y desarrollo del software asociado.
- **Cómo se trabajó:** Si bien el proyecto no se centró directamente en el hardware físico, la esencia de esta competencia se aplicó al diseñar e implementar las interfaces de usuario (UI) que automatizan y simplifican la interacción con un sistema computacional complejo: el motor de AutoML. La interfaz gráfica se convirtió en el panel de control que abstrae la complejidad de la ejecución de algoritmos y el manejo de datos. Se desarrollaron formularios intuitivos para la carga de datos, la configuración de experimentos, y la visualización de resultados interactivos, transformando un proceso que tradicionalmente requiere comandos de consola y scripts complejos en un flujo de trabajo sencillo y automatizado para el usuario final. Esto demostró mi capacidad para traducir la abstracción del backend de AutoML en una experiencia de usuario eficiente y comprensible.
- Coordina y participa en equipos multidisciplinarios para la aplicación de soluciones innovadoras en diferentes contextos.

- **Cómo se trabajó:** La residencia profesional no fue un trabajo aislado. Implicó una constante y fluida coordinación con mi asesor(es), quienes aportaban la visión experta en Machine Learning y las necesidades del usuario final desde la perspectiva del dominio de aplicación (por ejemplo, agricultura, gestión de recursos). Esta interacción me exigió comunicar conceptos técnicos complejos de manera clara y concisa a profesionales con diferentes trasfondos, así como escuchar activamente sus requerimientos y feedback. Participé activamente en la toma de decisiones técnicas y de diseño, colaborando en la búsqueda de soluciones innovadoras que se adaptaran a un contexto real. Esta experiencia fortaleció mi habilidad para trabajar en equipo, adaptarme a diversas perspectivas y contribuir eficazmente a un objetivo común.
- Diseña, implementa y administra bases de datos optimizando los recursos disponibles, conforme a las normas vigentes de manejo y seguridad de la información.
- **Cómo se trabajó:** Dentro del alcance del proyecto, fue esencial diseñar y gestionar una base de datos (e.g., PostgreSQL o MongoDB) para almacenar metadatos relacionados con los usuarios (si la plataforma incluía autenticación), los proyectos creados, los conjuntos de datos cargados y los modelos de AutoML entrenados. Participé en la elección de la arquitectura de la base de datos, el diseño de su esquema y la implementación de las operaciones CRUD (Crear, Leer, Actualizar, Borrar) necesarias. Se aplicaron principios de optimización de consultas para asegurar un acceso eficiente a la información. Además, se tuvo una consideración primordial por
- la seguridad de la información, implementando medidas como el cifrado de datos en tránsito (HTTPS) y la aplicación de controles
- de acceso para proteger la integridad y confidencialidad de la información manejada, lo que demuestra la adherencia a buenas prácticas en el manejo de datos.
- Desarrolla y administra software para apoyar la productividad y competitividad de las organizaciones cumpliendo con estándares de calidad.
- **Cómo se trabajó:** El objetivo intrínseco del proyecto fue desarrollar una herramienta de software que elevara la productividad de los usuarios al simplificar el proceso de Machine Learning. Durante el desarrollo, se aplicaron activamente estándares de calidad de software, incluyendo el uso disciplinado de sistemas de control de versiones (Git) para gestionar el código fuente, la implementación de pruebas (unitarias y de integración) para asegurar la funcionalidad y la fiabilidad de los componentes, y la adherencia a convenciones de codificación. La documentación interna del código y la modularidad del diseño fueron prioritarias para asegurar la mantenibilidad y escalabilidad del sistema a largo plazo. Al hacer el Machine Learning más accesible y eficiente, la plataforma contribuye directamente a la productividad y, por ende, a la competitividad de la institución o de cualquier organización que la adopte.
- Evalúa tecnologías de hardware para soportar aplicaciones de manera efectiva.
- **Cómo se trabajó:** Aunque el proyecto se centró en el desarrollo de software, la elección y configuración de la infraestructura en la nube implicó una evaluación indirecta de tecnologías de hardware virtual. Se analizó la capacidad de cómputo (CPU, RAM, y la posible necesidad de GPUs) que ofrecían los distintos servicios de computación en la nube para soportar las cargas de trabajo intensivas de entrenamiento de modelos AutoML. Esta evaluación consideró el rendimiento esperado, la escalabilidad y la relación costo-efectividad

para asegurar que la aplicación pudiera funcionar de manera efectiva y eficiente sin limitaciones de recursos. La experiencia me permitió comprender cómo las características del hardware (incluso virtualizado) impactan directamente en el desempeño y la viabilidad de las aplicaciones de inteligencia artificial.

- Detecta áreas de oportunidad empleando una visión empresarial para crear proyectos aplicando las Tecnologías de la Información y Comunicación.
- Cómo se trabajó: La concepción misma de este proyecto es un reflejo directo de esta competencia. Se identificó una clara área de oportunidad: la

brecha existente entre la complejidad del Machine Learning y la necesidad de profesionales de diversas disciplinas de aplicar estas herramientas. Se propuso y se desarrolló una solución tecnológica (la plataforma web interactiva) que aborda esta brecha, generando un valor tangible al democratizar el acceso a una tecnología avanzada. Esto demuestra una visión proactiva para identificar problemas y aplicar las Tecnologías de la Información y Comunicación (TIC) no solo como herramientas técnicas, sino como habilitadores estratégicos para crear proyectos con un impacto real, ya sea en el ámbito de la investigación, la academia o el potencial mercado empresarial.

- Diseña, configura y administra redes de computadoras para crear soluciones de conectividad en la organización, aplicando las normas y estándares vigentes.
- Cómo se trabajó: Si bien no se diseñó una red física desde cero, el despliegue de la plataforma en la nube implicó el diseño y la configuración de una infraestructura de red virtualizada. Esto incluyó la configuración de Virtual Private Clouds (VPCs), grupos de seguridad (Security Groups), reglas de firewall, y la gestión de rutas y subredes para asegurar la conectividad segura y eficiente entre el frontend, el backend y los servicios de AutoML. Se aplicaron principios de seguridad de red y se consideraron estándares para asegurar la comunicación cifrada (HTTPS) y proteger la plataforma de accesos no autorizados. Esta experiencia amplió mi comprensión de cómo se diseñan y administran las redes en entornos de nube para soportar aplicaciones distribuidas.



Capítulo VII: Fuentes de Información

7.1 Referencias bibliográficas y virtuales

Bibliografía

- Abi Aryan. (s.f.). *Problems and Challenges in AutoML*.
<https://abiaryan.com/posts/automl-problems-2019/>
- Apidog. (n.d.). *REST API URL Best Practices & Examples*.
<https://apidog.com/blog/rest-api-url-best-practices-examples/>
- arXiv. (2023). *AutoML in The Wild: Obstacles, Workarounds, and Expectations*. <https://arxiv.org/html/2302.10827v3> y
<https://arxiv.org/pdf/2302.10827>
- Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., ... & Thomas, D. (2001). *Manifesto for Agile Software Development*. <https://agilemanifesto.org/>
- Berners-Lee, T., Fielding, R., & Nielsen, H. F. (1996). *Hypertext Transfer Protocol -- HTTP/1.0*. RFC 1945.
- Boltic. (s.f.). *What is AutoML? Definition, Benefits, Use Case, and How Does It Work?*. <https://www.boltic.io/blog/automl>
- Burns, B., Beda, J., & Tarasov, D. (2016). *Kubernetes: Up and Running*. O'Reilly Media.
- Carnegie Mellon University. (n.d.). *Machine Learning*.
<https://www.cs.cmu.edu/~tom/mlbook.html>
- ClickUp. (s.f.). *Common AI Adoption Challenges And How to Overcome Them*. <https://clickup.com/blog/ai-adoption-challenges/>
- Devfi. (2025). *5 Common Challenges in Machine Learning Adoption: Solutions and Strategies*. <https://www.devfi.com/machine-learning-adoption-challenges-strategies/>
- DoneTonic. (n.d.). *Cómo aplicar las metodologías ágiles a tu proyecto*.
<https://donetonic.com/es/como-aplicar-las-metodologias-agiles-a-tu-proyecto/>
- Fowler, M. (2006). *The Agile Manifesto*.
<https://martinfowler.com/articles/newMethodology.html>
- GeeksforGeeks. (s.f.). *Frontend vs Backend Development*.
<https://www.geeksforgeeks.org/frontend-vs-backend/>
- Google for Developers. (s.f.). *Automated Machine Learning (AutoML)*.
<https://developers.google.com/machine-learning/crash-course/automl>

- Google for Developers. (s.f.). *AutoML: Benefits and limitations*. <https://developers.google.com/machine-learning/crash-course/automl/benefits-limitations>
- H2O.ai. (s.f.). *What are the Advantages of Automated Machine Learning?*. <https://h2o.ai/wiki/automated-machine-learning/>
- Hutter, F., Fu, P., & Leyton-Brown, K. (2019). *Automated Machine Learning: Methods, Systems, Challenges*. Springer.
- IBM. (s.f.). *What Is AutoML?*. <https://www.ibm.com/think/topics/automl>
- Iguazio. (2023). *What are the Advantages of Automated Machine Learning Tools?*. <https://www.iguazio.com/blog/automl-advantages/>
- Interim Group. (n.d.). *Metodología Agile Ejemplos*. <https://interimgrouphr.com/blog/metodologia-agile-ejemplos/>
- ISO 9241-210. (2019). *Ergonomics of human-system interaction — Part 210: Human-centred design for interactive systems*. International Organization for Standardization.
- Laws of UX. (n.d.). *Laws of UX*. <https://lawsoufux.com/>
- Medium. (2024). *Automated Machine Learning (AutoML): Simplifying AI Development for Everyone*. <https://medium.com/@hassaanidrees7/automated-machine-learning-automl-simplifying-ai-development-for-everyone-0b89f971b90d>
- Mell, P., & Grance, T. (2011). *The NIST Definition of Cloud Computing*. National Institute of Standards and Technology. <https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-145.pdf>
- Merkel, D. (2014). *Docker: lightweight linux containers for consistent development and deployment*. Linux Journal, 2014(239), 2.
- Microsoft Learn. (2025). *What is automated ML? AutoML - Azure Machine Learning*. <https://learn.microsoft.com/en-us/azure/machine-learning/concept-automated-ml?view=azureml-api-2>
- Milvus. (s.f.). *What are the challenges of implementing AutoML?*. <https://milvus.io/ai-quick-reference/what-are-the-challenges-of-implementing-automl>
- Mozilla. (s.f.). *Learn web development | MDN*. https://developer.mozilla.org/en-US/docs/Learn_web_development
- Nebius. (2024). *What is AutoML? Understanding automated machine learning*. <https://nebius.com/blog/posts/what-is-automl>
- Newman, S. (2015). *Building Microservices: Designing Fine-Grained Systems*. O'Reilly Media.

- PostIndustria. (2023). *AutoML Current Uses and Approaches*.
<https://postindustria.com/automl-current-uses-and-approaches/>
- Rivas, A. (2022). *Guía para Elaborar el REPORTE DE RP en word.docx*. Tecnológico Nacional de México Campus Culiacán.
- Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3), 210-229.
- Smith, J., & Jones, A. (2022). *Challenges in democratizing AI: A user experience perspective*. *Journal of Artificial Intelligence Research*, 75, 123-145.
- Softweb Solutions. (s.f.). *What is AutoML: Transforming business with advanced machine learning solutions*.
<https://www.softwebsolutions.com/resources/what-is-automl-transforming-business.html>
- Speakeasy. (n.d.). *REST API URL Best Practices*.
<https://www.speakeasy.com/api-design/urls>
- Stonebraker, M., & Rowe, L. A. (1986). *The design of POSTGRES*. Proceedings of the 1986 ACM SIGMOD International Conference on Management of Data, 340-355.
- Tanenbaum, A. S., & Van Steen, M. (2007). *Distributed Systems: Principles and Paradigms* (2nd ed.). Prentice Hall.
- Tecnológico Nacional de México Campus Culiacán. (2024). *Misión y Visión Institucional*. <https://www.culiacan.tecnm.mx/acerca-de/mision-y-vision/>
- Webopedia. (2024). *API*. <https://www.webopedia.com/>
- Wikipedia. (s.f.). *API*. <https://en.wikipedia.org/wiki/API>
- Wikipedia. (s.f.). *Aprendizaje automático automatizado*.
https://es.wikipedia.org/wiki/Aprendizaje_autom%C3%A1tico_automatizado
- Wikipedia. (s.f.). *Automated machine learning*.
https://en.wikipedia.org/wiki/Automated_machine_learning
- XenonStack. (2024). *Automated ML Challenges and its Use Cases | A Quick Guide*. <https://www.xenonstack.com/blog/auto-ml-challenges>
- Peña Lorenzo, J. M. (2020). Aplicación de técnicas de aprendizaje profundo (deep learning) para la detección de objetos en industria 4.0.
- Giraudy, A. E. (2022). Optimización Bayesiana para AutoML Heterogéneo (Doctoral dissertation, Universidad de La Habana).
- Cabrera, D. A. C. (2023). AutoML con modelos preentrenados (Doctoral dissertation, Universidad de la Habana).

- Valladares, E. L. E. (2023). AutoML Heterogéneo Multiobjetivo: Una propuesta para la democratización del aprendizaje automático (Doctoral dissertation, Universidad de La Habana).
- Castilla Gil, C. (2021). Identificación y análisis de las principales tecnologías de AutoML.
- Gómez, K., Magallan, M., Méndez, G., Ramírez, G., Guarda, T., Coronel, M., ... & Quirumbay, D. (2021). Aplicación del aprendizaje automático como solución en los sistemas de negocios. *Revista Ibérica de Sistemas e Tecnologías de Informação*, (E42), 137-148.
- ESIC Business & Marketing School. (Enero de 2018). Obtenido de <https://www.esic.edu/ rethink/tecnologia/aprendizaje-automatizado-ambito-empresarial#:~:text=El%20t%C3%A9rmino%20aprendizaje%20autom%C3%A1tico%2C%20o, procesos%20 de%20toma%20de%20decisiones>.
- Iglesias, A. (2016). *Distribución y logística*. Madrid: ESIC Editorial. Juárez, G. (24 de Mayo de 2017). Nexolution. Obtenido de <http://www.nexolution.com/ como-funciona-el-aprendizaje-automatizado-machine-learning/>
- IAobservatorio. (12 de Julio de 2016). IAobservatorio. Obtenido de <https://observatorioia.com/aprendizaje-automatizado-en-la-empresa>
- Nexus Integra. (14 de Agosto de 2020). Obtenido de <https://nexusintegra.io/es/ aplicaciones-del-machine-learning-en-la-industria/>
- Maisueche, A. (Septiembre de 2019). Repositorio documental de la Universidad de Valladolid. Obtenido de <http://uvadoc.uva.es/bitstream/handle/10324/37908/ TFM-I-1372.pdf?sequence=1&isAllowed=y>