CoProof - Plataforma para la demostración matemática formal colaborativa asistida por IA

David Alejandro López Torres 22310432
Daniel Tejeda Saavedra 22310431
Emiliano Flores Márquez 22110044

7N1

Proyecto 1.
Ingeniería en Desarrollo de Software

Prof. Carlos Darío Arenas Yerena

Centro de Enseñanza Técnica Industrial, Plantel Colomos
Guadalajara, México. Septiembre 8, 2025

Índice

Propósito del documento

El presente documento tiene como propósito describir de manera detallada la propuesta de proyecto **CoProof**, tanto desde una perspectiva técnica como administrativa. Se busca establecer una visión clara de los objetivos, el alcance, la justificación y la estructura inicial de gestión del proyecto. Este documento servirá como base de referencia para los responsables técnicos, los patrocinadores académicos y cualquier parte interesada en la evaluación de la viabilidad y relevancia del proyecto.

Desde el punto de vista técnico, se plantea el diseño e implementación de un entorno colaborativo en línea orientado a la creación, exploración y validación de demostraciones formales. El sistema integrará inteligencia artificial como mediador entre el lenguaje natural y el motor de verificación Lean, con el fin de reducir la barrera de entrada a la verificación formal y acelerar los procesos de razonamiento matemático.

Desde la perspectiva administrativa, el documento expone los objetivos generales y específicos, los entregables esperados, los alcances y las restricciones del proyecto. Asimismo, introduce lineamientos básicos para la administración de riesgos y personal, asegurando que la propuesta pueda desarrollarse de forma organizada, factible y dentro de un marco académico-profesional.

1 Justificación

La verificación formal de demostraciones matemáticas ha sido desde siempre una piedra angular en la matemática rigurosa y en la ingeniería de software orientada a la verificación. No obstante, a pesar del éxito y adopción creciente de asistentes de prueba como Lean 4, escribir y validar demostraciones continua siendo un reto técnico significativo que limita su accesibilidad a una audiencia amplia. Recientemente, Tang (2025) presentó un análisis exhaustivo de la arquitectura, la automatización y las aplicaciones de Lean 4, destacando sus ventajas en términos de usabilidad, rendimiento y potencial para metaprogramación, pero también resaltando la necesidad de capacidades adicionales que faciliten su manejo y colaboración Tang, 2025.

Asimismo, el campo del aprendizaje profundo aplicado a la demostración formal muestra un auge notable. Song, Yang y Anandkumar (2025) presentaron Lean Copilot, un marco que incorpora modelos de lenguaje como copilotos dentro del flujo de trabajo de Lean, sugiriendo pasos de prueba, completando objetivos intermedios y seleccionando premisas automáticamente, lo cual evidencia avances concretos en la asistencia inteligente integrada Song et al., 2025. No obstante, el uso práctico de Lean Copilot presenta desafíos importantes: por ejemplo, se ha reportado que en ocasiones Lean puede fallar o reiniciarse al tratar ciertos archivos durante la edición, y algunas funciones que pueden retornar formas sintácticas inapropiadas de premisas que no se alinean con la intención del usuario, lo que refleja limitaciones en la estabilidad y precisión de la integración actual contributors, 2025. En consecuencia, aunque Lean Copilot es prometedor, está limitado por problemas de resiliencia e interoperabilidad dentro de un entorno de trabajo continuo.

En una línea similar, Baba et al. (2025) propusieron Prover Agent, un agente de IA que combina razonamiento informal mediante modelos de lenguaje con verificación formal en Lean, alcanzando una tasa de éxito del 86.1% en el benchmark MiniF2F Baba et al., 2025. A pesar de este desempeño, dicha solución sigue siendo principalmente un agente autónomo, sin ofrecer una plataforma colaborativa ni capacidades de trazabilidad o visualización integradas, lo que restringe su uso en entornos de cooperación académica o educativa. De modo paralelo, Piotrowski, Fernández Mir y Ayers (2023) crearon una herramienta de aprendizaje automático dentro de Lean que sugiere premisas relevantes —muy útil, integrada y ligera— Piotrowski et al., 2023. Sin embargo, su enfoque se limita exclusivamente a la selección de premisas, sin abordar la traducción bidireccional, la asistencia activa en la construcción de la demostración, ni la visualización dinámica de cómo los teoremas se conectan y evolucionan. En resumen, aunque estas herramientas avanzadas representan pasos significativos, ninguna cubre de manera integral la visión colaborativa, asistida por IA y visualmente dinámica que propone *CoProof*.

En este contexto, el proyecto *CoProof* encuentra una sólida base justificada: pretende combinar la fuerza del motor de verificación Lean con inteligencia artificial como mediadora experta. Al habilitar la traducción bidireccional entre lenguaje natural y formal, asistencia automatizada en pasos de prueba, detección temprana de inconsistencias y colaboración visual dinámica, *CoProof* responde a las restricciones actuales de usabilidad, velocidad y accesibilidad, así como capitaliza las tendencias emergentes en investigación para democratizar el acceso a la demostración formal verificable.

2 Resumen ejecutivo

El proyecto CoProof propone el desarrollo de un entorno colaborativo de demostraciones formales asistido por inteligencia artificial e impulsado por el motor de verificación Lean. La solución ataca el problema de la complejidad y lentitud de las demostraciones formales mediante un sistema que traduce entre lenguaje natural y formal, asiste en la construcción de pruebas y visualiza relaciones entre teoremas de manera dinámica.

El mercado objetivo incluye principalmente a investigadores, estudiantes de matemáticas e informática, y equipos académicos que requieren verificación rigurosa en sus proyectos. Asimismo, la herramienta puede extenderse a áreas industriales donde la verificación formal de software y algoritmos es crítica. En conjunto, CoProof representa una innovación tecnológica y educativa que integra accesibilidad, colaboración y rigor lógico en un mismo entorno.

3 Plática de elevador

Nuestro equipo está desarrollando CoProof, una plataforma colaborativa para construir y validar demostraciones matemáticas. Hoy en día, escribir en lenguajes formales como Lean es muy complejo, y la verificación manual consume mucho tiempo en los equipos de investigación. Con nuestra solución, cualquier persona podrá escribir una demostración en lenguaje natural y obtener la traducción automática a Lean, validada y asistida por inteligencia artificial. La plataforma permitirá colaborar en línea, visualizar relaciones entre teoremas y diseñar pruebas por cómputo exhaustivo de manera accesible.

4 Objetivos del proyecto

- Diseñar e implementar un entorno colaborativo en línea que integre inteligencia artificial y Lean para la verificación formal de demostraciones matemáticas.
- Reducir las barreras técnicas en la elaboración de pruebas mediante traducción bidireccional entre lenguaje natural y formal.
- Acelerar el proceso de validación y exploración de teoremas en contextos académicos y de investigación.
- Promover un modelo de colaboración científica que combine accesibilidad, rigor lógico y trazabilidad.

5 Objetivos específicos

- Integrar Lean como motor central de verificación formal en el backend del sistema.
- Implementar un módulo de traducción bidireccional entre lenguaje natural y Lean mediante IA.
- Desarrollar agentes de IA especializados en estilos de demostración y áreas.
- Incluir mecanismos de detección temprana de inconsistencias y sugerencias inteligentes de pasos intermedios.
- Crear un editor colaborativo en línea con control de versiones y comunicación integrada.
- Desarrollar un módulo gráfico para la construcción de pruebas por cómputo exhaustivo.
- Diseñar una interfaz visual para mostrar relaciones y evolución entre teoremas.

6 Alcance

El proyecto comprende el diseño arquitectónico, el desarrollo del backend y frontend, la integración de modelos de IA para traducción y asistencia, la creación de un módulo especializado para pruebas de cómputo exhaustivo y paralelo con una interfaz de diseño por bloques, y la definición de un conjunto de artefactos de soporte (documentación técnica, pruebas automatizadas y material de capacitación). Se prioriza la entrega de una plataforma usable para evaluaciones con usuarios reales (investigadores y estudiantes) y la generación de evidencias cuantitativas sobre la eficacia de los componentes IA-Lean.

6.1 Entregables

A continuación se enumeran los entregables concretos que el equipo desarrollará y entregará al cierre del proyecto:

- **Especificación de requisitos y diseño arquitectónico.** Documento que incluye requisitos funcionales y no funcionales, diseño de alto nivel, modelo de datos, API especificadas y plan de pruebas.
- **Backend de verificación Lean.** Servicio (con API REST/gRPC) que orquesta sesiones con Lean, valida pruebas, administra colas de verificación y registra resultados y trazas para auditoría.
- **Módulo traductor bidireccional NL $\leftrightarrow$ Lean.** Conjunto de modelos y utilidades para convertir enunciados y pasos en lenguaje natural a términos/formas en Lean y viceversa; incluye pipeline de preprocesamiento, dataset paralelo para entrenamiento/validación y scripts de evaluación.
- **Agentes IA de asistencia.** Agentes especializados (por ejemplo: sugeridor de pasos, selector de premisas, agente de estilo de demostración por área matemática) que interactúan con el editor y con Lean para proponer, verificar y refinar pasos.
- **Editor colaborativo (IDE) en línea.** Interfaz web con edición simultánea (basada en CRDT/OT), resalte sintáctico para Lean, integración con los agentes IA, notificaciones y control de versiones (history / branches).
- **Distribución local y contenedorización.** Paquetes Docker/Compose y scripts para desplegar la plataforma en modo local (para investigación y evaluación) y en la nube (entorno web).
- **Módulo de cómputo exhaustivo y diseño por bloques.** Interfaz gráfica para construir pruebas por cómputo (configuración de pruebas, particionado de espacio de búsqueda, parámetros de paralelización) y backend que automatiza ejecución en serie o paralelo sobre recursos locales o remotos.
- **Visualizador dinámico de relaciones entre teoremas.** Componente que muestra grafo/lineage de teoremas, dependencias, versiones y anotaciones colaborativas (con zoom, filtros y timeline).
- **Dataset y suite de evaluación.** Conjunto de problemas/teoremas, pares NL-Lean y scripts para evaluar desempeño del traductor, tasa de verificación, latencia y robustez de agentes.

- **Pruebas automatizadas y plan de QA.** Test unitarios, integración y pruebas de carga que permiten validar regresiones y estabilidad.
- **Manuales y capacitación.** Documentación de usuario, guía de administración, tutoriales y al menos un taller de inducción para evaluadores.
- **Reporte final y recomendaciones.** Documento que compila resultados cuantitativos, lecciones aprendidas y roadmap para continuidad/escala.

Criterios de aceptación global

El proyecto se considerará aceptado si, al cierre, se cumplen simultáneamente los siguientes criterios mínimos: despliegue estable del sistema (entorno web y distribución local), disponibilidad de los módulos clave (traductor, agentes IA, backend Lean, editor colaborativo), y el cumplimiento de al menos el 70% de las métricas en sus umbrales de aceptación mínima definidos arriba (por ejemplo, alcanzar al menos 7 de 10 métricas en sus valores mínimos). Además, se exigirá la entrega de la documentación y la suite de evaluación que permitan replicar los experimentos y continuar el desarrollo.

7 Restricciones

A pesar del amplio espectro de funcionalidades que se propone, el sistema tiene límites bien definidos que evitan confusiones respecto a lo que no podrá realizar. Estas restricciones garantizan que los objetivos permanezcan claros y alcanzables, y que no se desvíen recursos hacia actividades fuera de la misión principal. Entre las restricciones más relevantes destacan:

- El sistema no demostrará teoremas de manera completamente automática sin intervención humana; su rol es de asistente, no de reemplazo del matemático.
- No resolverá problemas matemáticos abiertos o no formalizados en la literatura, ya que su ámbito se limita a teoremas con representación formalizable. *Nota: Puede ser utilizado por matemáticos e investigadores para este fin, pero no es su propósito fundamental.*
- No reemplazará a traductores generales de lenguaje natural: su especialización está en la traducción de lenguaje matemático natural hacia lenguaje formal.
- No garantizará la resolución de todos los problemas formales generados: algunos problemas pueden quedar sin resolver incluso con computación exhaustiva o paralela.
- No ofrecerá una interfaz de usuario en todos los idiomas; inicialmente se centrará en inglés y español.
- No tendrá soporte para todas las librerías de Lean ni de otros asistentes de prueba, sino un subconjunto definido en función de la compatibilidad y estabilidad.
- No incluirá por defecto algoritmos de aprendizaje autónomo continuo (auto-training), sino que dependerá de modelos previamente entrenados y de ajustes supervisados.

Estas restricciones no limitan el valor del sistema, sino que lo enfocan en aquello que es factible y medible dentro del marco del proyecto.

Característica	**Lo que sí hace el sistema**	**Lo que no hace el sistema**
Traducción matemática	Traduce enunciados matemáticos en lenguaje natural a Lean de forma semiautomática	No traduce textos generales no matemáticos ni contextos literarios
Resolución de problemas	Genera representaciones formales resolubles por métodos computacionales	No garantiza la solución de todos los problemas ni aborda problemas abiertos
Interfaz de usuario	Provee una plataforma web y local para interacción con usuarios	No incluye soporte para todos los idiomas ni compatibilidad universal con otros sistemas
Integración con Lean	Soporta un subconjunto relevante de librerías Lean	No asegura compatibilidad con todas las librerías ni con otros asistentes de prueba como Coq o Isabelle
Capacidad de cómputo	Implementa mecanismos para delegar problemas a cómputo exhaustivo o paralelo	No escala automáticamente a supercomputadoras sin configuración específica
Aprendizaje automático	Utiliza modelos entrenados previamente y con supervisión limitada	No incluye autoentrenamiento autónomo continuo sin intervención humana

Table 1: Comparación entre lo que el sistema sí y no realiza.

8 Administración de riesgos

La administración de riesgos en este proyecto constituye una actividad crítica para garantizar la estabilidad, confiabilidad y sostenibilidad del sistema a lo largo de todas sus fases de desarrollo, implementación y liberación al público. Dada la complejidad del traductor de lenguaje natural a lenguaje formal y su integración con motores de cómputo paralelos y plataformas web, es necesario establecer un plan de identificación, gestión y resolución de riesgos que permita anticipar y mitigar problemas antes de que estos afecten de manera grave al proyecto.

El enfoque adoptado se estructura en tres niveles: **prevención**, **detección temprana** y **contingencia**. La prevención incluye prácticas de calidad de software, revisiones de código, auditorías de seguridad y pruebas continuas. La detección temprana se basa en la monitorización automatizada y manual de métricas críticas, tales como disponibilidad de la plataforma, desempeño en benchmarks y retroalimentación de usuarios. Finalmente, la contingencia considera mecanismos de respuesta rápida ante fallos inevitables, con protocolos definidos de comunicación, recuperación y reanudación de operaciones.

8.1 Plan de gestión de riesgos

El plan de gestión de riesgos se divide en las siguientes fases y pasos:

1. **Identificación de riesgos:**
   - Realizar sesiones periódicas de lluvia de ideas con el equipo de desarrollo para enlistar riesgos técnicos, legales y operativos.
   - Documentar riesgos asociados a dependencias externas (bibliotecas, frameworks, servicios de terceros).
   - Revisar riesgos de seguridad en la manipulación de datos sensibles.
2. **Evaluación y priorización:**
   - Asignar un nivel de probabilidad (baja, media, alta) y un nivel de impacto (leve, moderado, crítico).
   - Clasificar los riesgos en una matriz de priorización que determine cuáles requieren atención inmediata.
3. **Definición de estrategias de mitigación:**
   - Establecer planes de prevención específicos (p. ej., pruebas unitarias automatizadas, redundancia en servidores).
   - Asignar responsables claros para cada riesgo identificado.
4. **Monitoreo y detección temprana:**
   - Configurar herramientas de monitoreo de rendimiento y seguridad (alertas sobre uso anómalo de CPU, memoria o accesos sospechosos).
   - Implementar tableros de métricas que permitan seguimiento en tiempo real.

5. **Respuesta y contingencia:**
    - Activar protocolos de recuperación rápida en caso de incidentes (rollback de versiones, activación de servidores de respaldo).
    - Comunicar de inmediato al equipo y, si aplica, a los usuarios afectados sobre la situación y los tiempos estimados de resolución.
6. **Evaluación posterior y documentación:**
    - Registrar el incidente, su causa raíz y las acciones correctivas implementadas.
    - Incorporar las lecciones aprendidas en nuevas versiones del plan de riesgos.

8.2 Plan de contingencia

El plan de contingencia se basa en mantener la continuidad del proyecto y reducir al mínimo los efectos adversos de eventos no deseados, tanto técnicos como administrativos. Se establecen medidas preventivas y soluciones tentativas para los riesgos más relevantes identificados:

Riesgo	**Posible impacto**	**Prevención**	**Plan de contingencia**
Sobrecarga en los servidores	Caídas del sistema y mala experiencia de usuario	Arquitectura escalable basada en microservicios y balanceadores de carga	Escalamiento automático en la nube y colas de espera temporales
Errores en la traducción formal	Resultados inconsistentes o incorrectos en problemas matemáticos	Entrenamiento con corpus amplio y pruebas unitarias en diversos contextos	Monitoreo de logs y liberación de *hotfixes* frecuentes
Bajo rendimiento en benchmarks	Pérdida de credibilidad académica	Validación frente a datasets estándar y entrenamiento supervisado	Optimización de algoritmos de cómputo paralelo
Dependencia de bibliotecas externas	Incompatibilidades técnicas y riesgo de abandono	Selección de librerías con soporte activo y documentación clara	Migración planificada hacia alternativas mantenidas con pruebas de regresión
Problemas de seguridad y filtrado de datos	Riesgos legales y pérdida de confianza	Cifrado de datos, cumplimiento de normativas y auditorías internas	Parcheo inmediato, comunicación a usuarios y refuerzo de políticas
Falta de adopción por la comunidad	Impacto limitado y baja justificación del proyecto	Difusión en universidades, foros y publicaciones	Adaptación del sistema a necesidades detectadas, incluyendo soporte multilingüe y nuevos dominios
Retrasos en el desarrollo por disponibilidad de personal	Demoras en entregables y desviación de cronograma	Planificación con márgenes de tiempo, asignación clara de responsabilidades	Redistribución de tareas, contratación temporal o ajuste de cronograma
Conflictos de comunicación entre equipos	Malentendidos, duplicación de trabajo, pérdida de eficiencia	Reuniones periódicas, herramientas de comunicación documentadas, actas	Mediación de conflictos, actualización de protocolos internos y seguimiento cercano
Cambios de alcance no planificados	Sobrecarga de trabajo y desviación de objetivos	Control de cambios formal, revisiones periódicas del alcance	Revisión y priorización de nuevos requerimientos, ajuste de entregables y recursos
Falta de documentación adecuada	Dificultad en mantenimiento y transferencia de conocimiento	Normas de documentación obligatorias y revisiones internas	Auditorías de documentación y actualización inmediata de secciones críticas

Table 2: Riesgos técnicos y administrativos, medidas preventivas y planes de contingencia.

9 Administración de personal

El éxito de un proyecto depende en gran medida de la forma en que se gestiona al equipo de trabajo. La administración de personal tiene como objetivo garantizar que los recursos humanos estén alineados con las metas del proyecto, se distribuyan de manera eficiente y cuenten con la motivación necesaria para cumplir sus responsabilidades. Este proceso abarca la planificación de roles, la asignación de tareas, la capacitación y la supervisión.

En primer lugar, se definirá un organigrama que muestre la estructura jerárquica y funcional del equipo. Se identificarán roles clave como el director del proyecto, los líderes técnicos, los analistas de requisitos, los desarrolladores, los responsables de calidad y los encargados de documentación. Cada rol tendrá asignadas responsabilidades específicas, lo que evita duplicidad de esfuerzos y facilita la rendición de cuentas. La asignación de recursos humanos se hará considerando la experiencia y competencias individuales, asegurando que cada miembro trabaje en tareas acordes a sus fortalezas.

En segundo lugar, se implementará un plan de comunicación interna que promueva la coordinación y el flujo de información. Se establecerán reuniones de seguimiento semanales, informes de avance y canales de comunicación ágiles mediante plataformas digitales colaborativas. Esto reducirá la posibilidad de malentendidos y fomentará un ambiente de transparencia. Asimismo, se promoverá la retroalimentación constante entre los miembros del equipo, con énfasis en la resolución temprana de conflictos.

Otro aspecto fundamental será la capacitación y el desarrollo profesional. El proyecto contempla la realización de talleres de inducción en las tecnologías utilizadas, así como la asignación de mentores para miembros con menor experiencia. Este enfoque no solo incrementa la eficiencia del trabajo, sino que también fortalece la cohesión del grupo al fomentar un aprendizaje colectivo. Además, se incentivará la rotación de tareas para que los integrantes adquieran una visión más amplia del proyecto.

Por último, la motivación y el reconocimiento jugarán un papel clave en la administración del personal. Se promoverán dinámicas de integración y se reconocerán públicamente los logros individuales y colectivos. El liderazgo se ejercerá bajo un enfoque participativo, en el que las decisiones se tomen de forma colaborativa siempre que sea posible. De esta manera, se construye un equipo comprometido, resiliente y capaz de afrontar los desafíos del proyecto con una actitud positiva.

9.1 Perfiles del equipo de desarrollo

Se anticipa que, en un escenario de despliegue completo, el equipo de desarrollo deberá cubrir áreas de desarrollo de software, inteligencia artificial, pruebas, infraestructura y coordinación de proyectos. Los perfiles y competencias técnicas requeridas son los siguientes:

- **Líder de proyecto:** Responsable de la planificación general, asignación de recursos y seguimiento de cronogramas. Debe poseer habilidades de gestión de proyectos, comunicación efectiva, resolución de conflictos y experiencia en coordinación de equipos técnicos multidisciplinarios.

- **Ingeniero de backend:** Encargado del diseño e implementación de la lógica del servidor, bases de datos y APIs. Conocimientos avanzados en Python, Node.js o Java, desarrollo de microservicios, bases de datos relacionales y no relacionales, y familiaridad con integración de motores de demostración formal como Lean.
- **Ingeniero de frontend:** Responsable de la interfaz de usuario y experiencia interactiva en la plataforma web y local. Dominio de frameworks como React, Angular o Vue, habilidades en diseño UX/UI, manejo de bibliotecas de visualización dinámica y desarrollo multiplataforma.
- **Ingeniero de inteligencia artificial:** Desarrolla los modelos de traducción bidireccional y agentes de inferencia de teoremas. Conocimientos en aprendizaje profundo, NLP, integración de modelos con APIs de backend y experiencia en frameworks como PyTorch o TensorFlow.
- **Especialista en cómputo paralelo y exhaustivo:** Diseña y optimiza módulos para demostraciones mediante cómputo paralelo y exhaustivo. Experiencia en programación concurrente y distribuida, HPC y optimización de algoritmos.
- **Ingeniero de calidad y pruebas:** Encargado de implementar pruebas unitarias, de integración y de rendimiento, y validar la precisión de traducciones y demostraciones. Domina frameworks de testing, metodologías de aseguramiento de calidad y métricas de evaluación de sistemas de IA.
- **Administrador de infraestructura y DevOps:** Responsable de la gestión de servidores, despliegue de la plataforma, integración continua y automatización. Conocimientos en sistemas Linux, contenedores (Docker, Kubernetes), CI/CD, monitoreo y seguridad informática.
- **Documentación y soporte técnico:** Produce y mantiene documentación técnica, guías de usuario y soporte a la comunidad. Habilidades de redacción técnica, comprensión del flujo de trabajo de la plataforma y capacidad de sintetizar información compleja.
- **Coordinador de comunidad y adopción académica:** Encargado de la difusión, capacitación y retroalimentación de usuarios. Habilidades de comunicación, divulgación científica, manejo de comunidades online y organización de talleres o webinars.

Este enfoque permite que cada miembro del equipo conozca sus responsabilidades, aproveche sus habilidades técnicas de manera óptima y que el proyecto pueda escalar según las necesidades futuras, manteniendo un desarrollo ordenado y eficiente.

10 Adminsitración de recursos materiales

10.1 Equipo de cómputo

- **Estaciones de trabajo de desarrollo**: Computadoras personales con procesadores de al menos 8 núcleos (Intel i7 o AMD Ryzen 7), 32 GB de memoria RAM y almacenamiento SSD NVMe de 1 TB. Utilizadas por los desarrolladores para programación, pruebas locales y documentación.
- **GPU de alto rendimiento**: Una tarjeta gráfica NVIDIA RTX 4090 (24 GB VRAM GDDR6X), destinada al entrenamiento y ajuste fino de modelos de inteligencia artificial y razonamiento matemático automatizado.
- **Mini clúster de cómputo**: Conformado por 4 Raspberry Pi 4 (8 GB RAM, quad-core ARM Cortex-A72 a 1.5 GHz), corriendo Ubuntu 24.04 LTS, conectadas mediante un switch Netgear de 8 puertos Gigabit. Este clúster se utilizará para experimentación en cómputo paralelo y despliegues distribuidos de prototipos.
- **Servidor central de integración**: Equipo con procesador de 32 núcleos, 64 GB de RAM y almacenamiento híbrido (SSD + HDD de 10 TB) destinado a la coordinación de pruebas de integración, almacenamiento de modelos y ejecución de contenedores.
- **Laptops de soporte y pruebas**: 2 laptops adicionales con 16 GB RAM y procesadores de 6 núcleos, destinadas a pruebas de usuario, documentación y despliegues ligeros.
- **Dispositivos móviles**: 2 tablets Android y 2 iPads, utilizadas para pruebas de accesibilidad de la plataforma en navegadores móviles.
- **Monitores adicionales**: Pantallas de 27” con resolución 4K, para facilitar el trabajo en interfaces gráficas, depuración y comparación de resultados.

10.2 Software y herramientas

- **Modelos de inteligencia artificial**: Modelos open-source como LLaMA, Mistral, GPT-NeoX y LeanDojo, utilizados en versiones preentrenadas o con reentrenamiento parcial en GPU local.
- **Lenguajes de programación**: Python, Lean, Haskell y C++ para el desarrollo del traductor, integración con entornos de razonamiento formal y módulos de cómputo paralelo.
- **Entornos de desarrollo**: Visual Studio Code, PyCharm y JupyterLab para la programación colaborativa y pruebas rápidas.
- **Sistemas de virtualización y contenedores**: Docker y Kubernetes para la gestión de despliegues en múltiples entornos.
- **Gestión de versiones**: Git y GitHub para control de versiones y colaboración.

- **Plataformas de documentación**: LaTeX para reportes académicos, MkDocs para documentación técnica en línea y Notion para la coordinación interna del equipo.
- **Herramientas de benchmarking**: Librerías como `pytest-benchmark` y métricas específicas para medir desempeño en resolución de teoremas y traducción automática.

10.3 Servicios de apoyo

- **Almacenamiento en la nube**: Espacios en Google Cloud, AWS o Azure para respaldo de datos, modelos pesados y pruebas de escalabilidad.
- **Servicios de cómputo bajo demanda**: Instancias de GPU en la nube (por ejemplo, NVIDIA A100 o H100) para experimentos que excedan la capacidad de la RTX 4090.
- **Plataformas de comunicación**: Slack, Microsoft Teams o Discord para coordinación interna del equipo.
- **Sistema de gestión de proyectos**: Jira o Trello para seguimiento de tareas y organización de backlog.
- **Control de acceso y seguridad**: Autenticación multifactor y cifrado de datos para proteger los resultados del proyecto y la infraestructura en la nube.

Table 3: Resumen de recursos de cómputo

Equipo	**Procesador**	**RAM**	**Almacenamiento**	**Propósito**
Estaciones de trabajo	Intel i7 / AMD Ryzen 7 (8 núcleos)	32 GB	SSD NVMe 1 TB	Desarrollo, pruebas locales y documentación
GPU de alto rendimiento	NVIDIA RTX 4090 (24 GB VRAM)	–	–	Entrenamiento y ajuste de modelos de IA
Mini clúster	4 × Raspberry Pi 4 (ARM Cortex-A72, 4 núcleos)	8 GB c/u	microSD 128 GB c/u	Cómputo paralelo y despliegues distribuidos
Servidor central	32 núcleos multinúcleo	64 GB	SSD + HDD 10 TB	Integración, almacenamiento de modelos y contenedores
Laptops de soporte	CPU 6 núcleos	16 GB	SSD 512 GB	Pruebas de usuario y despliegues ligeros
Dispositivos móviles	ARM multinúcleo	4–6 GB	64–128 GB	Pruebas en navegadores móviles
Monitores 4K	–	–	–	Interfaces gráficas, depuración y visualización de resultados

11 Relación con áreas externas o internas

El proyecto *CoProof* contempla establecer vínculos estratégicos tanto dentro como fuera del Centro de Enseñanza Técnica Industrial (CETI). Estas relaciones tienen como finalidad enriquecer el proceso de desarrollo mediante retroalimentación especializada y garantizar que la solución propuesta cuente con un nivel adecuado de aceptabilidad dentro de la comunidad académica y científica.

En el ámbito interno, se prevé la participación de docentes y estudiantes del CETI Colomos, quienes aportarán su experiencia en áreas de desarrollo de software, metodologías de investigación aplicada y validación temprana de prototipos. Este vínculo permitirá que el equipo de trabajo se mantenga alineado con las competencias técnicas promovidas por la institución y que el proyecto se enmarque dentro de la formación académica del alumnado.

En el ámbito externo, se pretende mantener comunicación formal por medio de correo electrónico con investigadores, profesores, estudiantes y egresados de instituciones nacionales e internacionales de prestigio. Los contactos previstos son los siguientes:

- **Instituto Tecnológico y de Estudios Superiores de Monterrey (ITESM, Campus Monterrey)**: se busca retroalimentación temprana en cuanto a la usabilidad y accesibilidad de la plataforma, dado su enfoque en innovación educativa. Se contempla establecer comunicación inicial en **enero de 2026**.
- **Universidad Autónoma de Nuevo León (UANL)**: se pretende involucrar a investigadores en matemáticas aplicadas y ciencias computacionales para la validación de los módulos de traducción entre lenguaje natural y formal. El contacto se proyecta para **febrero de 2026**.
- **Universidad de Guanajuato, Departamento de Matemáticas (DEMAT)**: su experiencia en lógica y teoría de números puede servir para probar la solidez del motor de inferencia de la plataforma. Se plantea una interacción en **enero-febrero de 2026**.
- **Centro de Investigación en Matemáticas (CIMAT), Guanajuato**: se buscará colaboración en torno al módulo de demostraciones por cómputo exhaustivo y paralelo, aprovechando su experiencia en supercómputo y algoritmos. La comunicación se prevé para **marzo de 2026**.
- **Universidad Veracruzana (UV)**: se pretende contar con apoyo en la fase de validación educativa, evaluando el potencial del entorno como herramienta didáctica en cursos de matemáticas. El contacto inicial se proyecta en **enero de 2026**.
- **Universidad Nacional Autónoma de México (UNAM)**: dada su relevancia en investigación matemática a nivel nacional, se buscará colaboración para validar el repositorio de teoremas y su memoria evolutiva. Se prevé contacto en **febrero de 2026**.
- **University of Chicago**: el objetivo será validar la relevancia y aplicabilidad del sistema en contextos de investigación de frontera en teoría de números y demostraciones formales. La comunicación se proyecta para **marzo de 2026**.

- **Massachusetts Institute of Technology (MIT)**: se busca retroalimentación en torno al diseño colaborativo y a la integración de herramientas de inteligencia artificial en entornos formales, con miras a robustecer la parte de innovación tecnológica. Se planea establecer contacto en **marzo-abril de 2026**.

El propósito de estas interacciones será doble: por un lado, obtener retroalimentación técnica y académica sobre los prototipos funcionales del sistema; por otro, garantizar que el producto final responda a las expectativas de investigadores y docentes que trabajan con demostraciones formales.

Este acercamiento permitirá contrastar los avances del proyecto con estándares internacionales, detectar de manera temprana mejoras que aumenten la utilidad del sistema en entornos de investigación colaborativa y construir una red de validación que respalde el potencial de transferencia tecnológica del proyecto. En particular, se espera que hacia **abril de 2026** se cuente con un conjunto sólido de resultados preliminares derivados de estas interacciones, lo que permitirá afinar las últimas etapas de desarrollo.

12 Enfoque de desarrollo

El enfoque de desarrollo seleccionado para el proyecto será la metodología ágil **SCRUM**. La elección de esta metodología se justifica debido a la naturaleza del proyecto, el cual requiere un desarrollo flexible y adaptable, especialmente porque involucra la experimentación con modelos de inteligencia artificial (tanto pre-entrenados como entrenados en hardware local), así como el diseño iterativo de prototipos de la aplicación *CoProof*.

Justificación

SCRUM se adapta mejor que enfoques tradicionales como *cascada*, ya que permite:

- Ajustar los objetivos de cada iteración con base en la retroalimentación temprana recibida de profesores, estudiantes y egresados.
- Entregar prototipos funcionales en periodos cortos, lo que facilita validar la aceptabilidad de la aplicación en etapas tempranas.
- Favorecer la comunicación continua dentro del equipo y con las instituciones externas involucradas.
- Reducir riesgos al identificar problemas técnicos o de usabilidad de manera temprana, evitando retrasos acumulados.

Implementación de SCRUM en el proyecto

El desarrollo del proyecto se organizará en **sprints** de **tres semanas**, con entregables funcionales al final de cada ciclo. El proceso de implementación será el siguiente:

- **Product Owner**: Representado por el líder del proyecto, quien definirá y priorizará las características clave de *CoProof*.
- **Scrum Master**: Persona encargada de garantizar que se sigan las prácticas ágiles y de resolver bloqueos del equipo.

- **Equipo de Desarrollo**: Integrado por los responsables de programación, pruebas y experimentación con modelos de IA.
- **Sprints**: Cada sprint tendrá objetivos claros, como la integración de un modelo de IA, el desarrollo de la interfaz, o la validación de prototipos.
- **Reuniones**: Se realizarán reuniones de sincronización cortas (*daily scrums*) y una reunión de revisión al final de cada sprint con retroalimentación de usuarios de prueba (profesores, estudiantes y egresados).
- **Prototipos Incrementales**: En cada sprint se presentará una versión mejorada de *CoProof*, desde una interfaz básica hasta un sistema funcional con pruebas de aceptabilidad.

De esta manera, la metodología SCRUM permitirá que el desarrollo avance de forma ordenada, con una clara orientación a resultados, manteniendo la flexibilidad para responder a cambios en los requerimientos y en la retroalimentación de la comunidad académica.

13 Mejores prácticas

Durante el desarrollo del proyecto *CoProof* se implementarán un conjunto de buenas prácticas orientadas a garantizar la calidad, continuidad y eficiencia en cada etapa del ciclo de vida del software. Estas prácticas se alinearán tanto con marcos de referencia internacionales como con metodologías modernas de desarrollo colaborativo.

Marco de referencia ITIL v4

Se adoptarán principios del marco de referencia **ITIL v4** para garantizar la calidad y continuidad del servicio, especialmente en lo referente a:

- **Gestión de incidencias**: Documentación y atención sistemática de errores detectados durante pruebas o en producción.
- **Gestión de cambios**: Procedimientos claros para introducir mejoras o modificaciones sin comprometer la estabilidad del sistema.
- **Gestión de disponibilidad y continuidad**: Definición de políticas para asegurar que el servicio se mantenga operativo y con planes de contingencia establecidos.
- **Monitoreo y mejora continua**: Ciclos de retroalimentación para evaluar y optimizar el desempeño de la aplicación.

Flujo de desarrollo colaborativo

Con el fin de estructurar y profesionalizar el trabajo del equipo, se aplicarán prácticas ampliamente utilizadas en proyectos de software:

- **Control de versiones**: Uso de GitHub/GitLab como repositorio central.
- **Pull Requests (PRs)**: Todo cambio al código será sometido a revisión por pares mediante PRs antes de ser integrado a la rama principal.

- **Issues y Kanban boards**: Registro de tareas, errores y mejoras mediante issues, gestionados en tableros Kanban para visualizar el progreso.
- **Pruebas unitarias y de integración**: Implementación de un entorno de pruebas automatizadas para asegurar la calidad del código y prevenir regresiones.
- **Integración continua (CI)**: Configuración de pipelines automáticos para ejecutar pruebas y análisis estáticos de código en cada commit.

Estándares de documentación y estilo de código

Para favorecer la claridad y mantenibilidad del proyecto, se implementarán estándares consistentes:

- **Framework de documentación**: Uso de herramientas como `Sphinx` o `MkDocs` para centralizar la documentación técnica.
- **Convenciones de sintaxis**: Adopción de guías de estilo específicas para los lenguajes principales:
  - `PEP8` para Python.
  - `Airbnb JavaScript Style Guide` para JavaScript.
  - Convenciones de la comunidad para LEAN.

  En caso de utilizarse otros lenguajes de programación, se adoptarán los lineamientos de estilo recomendados para cada uno.
- **Comentarios y docstrings**: Uso disciplinado de comentarios y docstrings para facilitar la comprensión del código.
- **Versionado de entregables**: Asignación de números de versión siguiendo el esquema `SemVer` (ej. v1.0.0).

Prácticas de comunicación y retroalimentación

Finalmente, se fomentará un flujo de comunicación ágil y transparente:

- **Revisiones periódicas**: Reuniones de retroalimentación al final de cada sprint (SCRUM).
- **Documentación compartida**: Uso de plataformas colaborativas (ej. Notion, Confluence o Google Drive).
- **Cultura de mejora continua**: Retroalimentación constante entre los miembros del equipo para detectar áreas de optimización tanto técnicas como administrativas.

14 Tabla de hitos

A continuación, se presenta la planificación tentativa de los principales hitos del proyecto *CoProof*. Cada hito corresponde a un entregable clave o a una actividad fundamental que marca el avance hacia la consolidación del producto final.

Hito	**Descripción**	**Fecha tentativa**
Entrega del Project Charter	Documento inicial que define el alcance, objetivos, roles y plan de trabajo del proyecto.	Octubre 2025
Especificación de requisitos y diseño arquitectónico	Documento con requisitos funcionales y no funcionales, diseño de alto nivel, modelo de datos, API y plan de pruebas.	Noviembre 2025
Backend de verificación Lean	Desarrollo del servicio de verificación con Lean, incluyendo APIs y registro de trazas.	Diciembre 2025
Módulo traductor bidireccional NL $\leftrightarrow$ Lean	Primer prototipo funcional del traductor con dataset paralelo inicial.	Enero 2026
Agentes IA de asistencia	Desarrollo y pruebas de agentes especializados (sugeridor de pasos, selector de premisas, etc.).	Febrero 2026
Editor colaborativo (IDE) en línea	Versión preliminar del IDE con edición colaborativa, resaltado y conexión con agentes.	Marzo 2026
Prueba piloto con comunidades académicas	Liberación temprana de un prototipo a profesores, egresados y estudiantes de instituciones colaboradoras para retroalimentación.	Marzo 2026
Distribución local y contene-dorización	Publicación de contenedores Docker/Compose para despliegue local y en la nube.	Marzo 2026
Módulo de cómputo exhaustivo y diseño por bloques	Implementación de interfaz gráfica y backend de cómputo paralelo/serie.	Marzo 2026
Visualizador dinámico de relaciones entre teoremas	Desarrollo de prototipo del grafo interactivo de dependencias.	Marzo 2026

Hito	**Descripción**	**Fecha tentativa**
Dataset y suite de evaluación	Ampliación del dataset y validación del traductor y agentes en benchmarks relevantes.	Marzo 2026
Pruebas automatizadas y plan de QA	Ejecución de pruebas unitarias, integración y carga para validar regresiones.	Marzo 2026
Manuales y capacitación	Documentación de usuario, guía de administración y talleres de inducción.	Abril 2026
Presentación del proyecto	Presentación oficial del estado final del proyecto ante la comunidad académica y evaluadores.	Abril 2026
Reporte final y recomendaciones	Documento con resultados cuantitativos, lecciones aprendidas y roadmap para continuidad.	Abril 2026
Liberación del producto final	Publicación oficial del sistema con prototipos probados y documentación completa.	Abril 2026

15 Recursos físicos o virtuales

El proyecto *CoProof* requiere de una serie de recursos físicos y virtuales que no serán desarrollados por el equipo, sino que se emplearán como base para la construcción de la solución. Estos recursos incluyen lenguajes de programación, plataformas de ejecución, librerías y modelos de inteligencia artificial preexistentes que permitirán acelerar el desarrollo y asegurar la calidad de los resultados. A continuación, se detallan los principales elementos contemplados:

Recurso	**Descripción y uso en el proyecto**
Lenguaje Python	Utilizado para el desarrollo de los módulos de inteligencia artificial, en particular el traductor bidireccional NL $\leftrightarrow$ Lean, los agentes de inferencia y la integración con librerías de procesamiento del lenguaje natural.
Lenguaje JavaScript (Node.js / React)	Base para el desarrollo del editor colaborativo en línea (IDE) y la interfaz gráfica. React será empleado para el frontend interactivo, mientras que Node.js facilitará el backend de la plataforma web.
Lenguaje Lean	Herramienta formal de verificación que servirá como motor base del sistema. Será utilizado para definir axiomas, enunciar teoremas y verificar pruebas generadas o traducidas por el sistema.

Plataforma Docker	Contenerización del backend, agentes y servicios auxiliares, facilitando el despliegue en entornos locales y en la nube.
Plataforma GitHub	Gestión del código fuente, issues, pull requests y control de versiones, así como espacio de colaboración del equipo.
Framework PyTorch / TensorFlow	Bibliotecas para el entrenamiento y ajuste fino de modelos de IA, principalmente en los módulos de traducción y sugerencia de pasos en demostraciones.
Modelos de lenguaje preentrenados (ej. GPT, LLaMA, Mistral)	Puntos de partida para el desarrollo de agentes inteligentes de asistencia. Se utilizarán variantes open source o accesibles vía APIs, adaptadas mediante ajuste fino a los contextos de demostración formal.
Kubernetes*	Para la orquestación de contenedores en entornos de despliegue escalables, en caso de que se requiera manejar múltiples usuarios concurrentes en producción.
Herramientas de colaboración (Slack, Teams, o Discord)	Canales de comunicación interna y con colaboradores externos, esenciales para la coordinación del equipo.
Suite de pruebas (PyTest, Jest, Lean Unit Tests)	Herramientas de validación para asegurar la calidad del software en cada etapa de desarrollo.

16 Costo presupuestado del proyecto

Período de ejecución: septiembre de 2025 a abril de 2026 (8 meses). Todos los montos están expresados en pesos mexicanos (MXN).

1. Equipo de cómputo y software

Concepto	**Cantidad**	**Costo estimado (MXN)**
Estaciones de trabajo de alto rendimiento (3 desarrolladores)	3	$120,000
Estaciones de trabajo de medio rendimiento (6 administrativos)	6	$60,000
Monitores 4K	3	$30,000
Herramientas de desarrollo open source (IDE, frameworks, bibliotecas)	Global	$0

Nota: El servidor central puede reemplazarse con instancias en la nube para pruebas, lo que reduce inversión inicial, aunque mantenerlo local ofrece ventajas de control y latencia.	
Subtotal equipo y software	**$210,000**

2. Sueldos del equipo (junior, 8 meses)

Se consideran sueldos mensuales promedio de $25,000 MXN para los desarrolladores junior. Para los perfiles de soporte, coordinación y especialistas, se estiman $20,000 MXN mensuales.

Perfil	**Duración (meses)**	**Costo estimado total (MXN)**
Líder de proyecto	8	$160,000
Ingeniero de backend (junior)	8	$200,000
Ingeniero de frontend (junior)	8	$200,000
Ingeniero de IA (junior)	8	$200,000
Especialista cómputo paralelo	8	$160,000
QA / pruebas	8	$160,000
DevOps / infraestructura	8	$160,000
Documentación / soporte técnico	8	$160,000
Coordinador comunidad	8	$160,000
Subtotal sueldos del equipo		**$1,560,000**

3. Costos de despliegue y producción (nube)

Concepto	**Cantidad**	**Costo estimado (MXN)**
Despliegue en la nube (3 instancias activas, 24/7, 8 meses)	Global	$69,120
Dominio + SSL	Global	$20,000
Monitoreo y respaldo (cloud)	Global	$60,000
Difusión académica	Global	$100,000
Subtotal despliegue y producción		**$249,120**

4. Otros costos adicionales

Concepto	Cantidad	Costo estimado (MXN)
Mantenimiento de equipo y actualizaciones	Global	$80,000
Fondo de contingencia (10%)	Global	$140,000
Subtotal otros costos		**$220,000**

Resumen general del presupuesto

Categoría	Total (MXN)
Equipo y software	$210,000
Sueldos del equipo	$1,560,000
Despliegue y producción	$249,120
Otros costos adicionales	$220,000
Monto total estimado	**$2,239,120**

17 Matriz RACI

Símbolos de roles:

L: Líder de proyecto

B: Ingeniero Backend

F: Ingeniero Frontend

I: Ingeniero IA

C: Especialista Cómputo

Q: QA / Pruebas

D: DevOps / Infraestructura

S: Documentación / Soporte

O: Coordinador comunidad

Símbolos de entregables:

E1: Especificación de requisitos y diseño arquitectónico

E7: Módulo de cómputo exhaustivo

E2: Backend de verificación Lean

E8: Visualizador de relaciones entre teoremas

E3: Traductor NL $\leftrightarrow$ Lean

E9: Dataset y suite de evaluación

E4: Agentes IA de asistencia

E10: Pruebas automatizadas y plan de QA

E5: Editor colaborativo (IDE) en línea

E11: Manuales y capacitación

E6: Distribución local y contenedorización

E12: Reporte final y recomendaciones

Entregable / Rol	L	B	F	I	C	Q	D	S	O
E1	A	C	C	C	C	I	I	I	I
E2	I	R	I	C	C	C	A	I	I
E3	I	C	I	R	I	C	I	I	I
E4	I	C	I	R	I	C	I	I	I
E5	I	C	R	C	I	C	I	I	I
E6	I	C	C	C	I	I	R	I	I
E7	I	C	I	C	R	C	I	I	I
E8	I	C	R	C	I	C	I	I	I
E9	I	R	I	R	I	C	I	I	I
E10	I	C	C	C	C	R	I	I	I
E11	I	I	I	I	I	C	I	R	C
E12	A	C	C	C	C	C	I	R	C

Leyenda de RACI: R = Responsable, A = Accountable, C = Consultado, I = Informado.

18 Criterios de éxito

El éxito del proyecto CoProof se medirá a través de un conjunto de criterios cuantitativos y cualitativos que evalúan tanto la funcionalidad del sistema como la aceptación por parte de la comunidad académica. Estas métricas aseguran que los objetivos del proyecto se cumplan de manera tangible y verificable, y que la plataforma cumpla con su propósito de facilitar la construcción, traducción y verificación de demostraciones formales en un entorno colaborativo. Además de las métricas cuantificables, el proyecto también considerará la retroalimentación de usuarios expertos y estudiantes para validar la utilidad, usabilidad y escalabilidad del sistema.

18.1 Métricas y criterios cuantificables

Para evitar ambigüedades sobre el alcance se definirá un conjunto de métricas cuantificables con objetivos (targets) y criterios mínimos de aceptación. Estas métricas se medirán mediante las herramientas y suites de evaluación incluidas en el entregable "Dataset y suite de evaluación".

Usuarios registrados (meta):

Target: 500 usuarios registrados en el entorno (prototipo abierto a la comunidad académica).

Aceptación mínima: 100 usuarios registrados y 20 usuarios activos mensuales durante la etapa de evaluación.

Medición: registros de cuenta en la base de datos y métricas de actividad (login, ediciones, sesiones activas).

Número de teoremas verificados (meta):

Target: 200 teoremas/lemmas verificados por Lean a través del pipeline NL→Lean.

Aceptación mínima: 50 teoremas verificados.

Medición: conteo de artefactos aceptados por Lean y almacenados en el repositorio del proyecto.

Tasa de éxito de agentes en benchmark (meta):

Target: $\geq 50\%$ de éxito promedio en benchmarks públicos relevantes.

Aceptación mínima: $\geq 30\%$.

Medición: ejecución automatizada sobre benchmarks y registro de casos resueltos por Lean.

Precisión de la traducción NL → Lean (meta):

Target: $\geq 65\%$ exact-match semántico en un conjunto de evaluación.

Aceptación mínima: $\geq 45\%$.

Medición: evaluación automática (exact match, BLEU o métricas de similaridad) complementada con evaluación humana.

Calidad de la traducción Lean → NL (meta):

Target: puntuación de fluidez y fidelidad ≥ 4.0 en escala Likert (1–5).

Aceptación mínima: promedio ≥ 3.0.

Medición: encuesta ciega a evaluadores expertos comparando la traducción automática con referencia humana.

Latencia de verificación (meta):

Target: tiempo medio por verificación de objetivo simple < 2 s y de objetivo complejo < 30 s.
Aceptación mínima: < 5 s (simple) y < 120 s (complejo).
Medición: pruebas de rendimiento automatizadas registrando latencias end-to-end.

Escalabilidad / concurrencia (meta):

Target: soportar 50 sesiones concurrentes de edición con degradación controlada del rendimiento ($< 30\%$ aumento de latencia promedio).
Aceptación mínima: 10 sesiones concurrentes.
Medición: pruebas de carga y stress sobre el editor colaborativo y el backend.

Disponibilidad (meta):

Target: uptime $\geq 99.0\%$ durante la fase beta.
Aceptación mínima: uptime $\geq 95.0\%$.
Medición: monitoreo continuo y reporte de incidentes.

Satisfacción de usuarios (meta):

Target: puntuación SUS ≥ 70 en pruebas con al menos 25 usuarios.
Aceptación mínima: SUS ≥ 55.
Medición: encuestas estandarizadas aplicadas tras sesiones de uso dirigidas.

Reproducción de comprobaciones por cómputo (meta):

Target: ejecución correcta y reproducible de al menos 20 diseños de pruebas exhaustivas creados con el módulo por bloques.
Aceptación mínima: 5 diseños reproducibles.
Medición: registros de ejecución y verificación de integridad de resultados en la base de datos.

18.2 Criterios cualitativos

Además de las métricas cuantificables, se considerará como éxito la aceptación por parte de la comunidad académica y de investigación formal. Esto incluirá retroalimentación positiva en pruebas de prototipo, adopción por parte de usuarios avanzados y principiantes, y la percepción de que la plataforma facilita significativamente la construcción, comprensión y verificación de demostraciones formales.

19 Plan de comunicaciones

El plan de comunicaciones de CoProof tiene como objetivo asegurar que toda la información relevante fluya de manera oportuna y eficiente tanto dentro del equipo de desarrollo como con los asesores externos. Esto permite una coordinación adecuada, facilita la toma de decisiones y asegura la transparencia en todas las etapas del proyecto.

19.1 Comunicación interna del equipo

La comunicación interna se realizará principalmente a través de Slack, complementada con reuniones semanales y documentación compartida. Slack permitirá mensajes instantáneos, creación de canales específicos por módulo o tema, notificaciones de tareas críticas y seguimiento de issues. Las reuniones internas semanales permitirán coordinar tareas, discutir avances, identificar obstáculos y priorizar actividades. Adicionalmente, se mantendrán registros de decisiones importantes y acuerdos dentro de Slack y en documentos compartidos para garantizar la trazabilidad de las acciones.

19.2 Comunicación con asesores y revisores externos

Se establecerá contacto formal con los asesores del proyecto y expertos de las instituciones colaboradoras mediante correo electrónico y videoconferencias programadas. La comunicación se enfocará en la revisión de prototipos funcionales, validación de resultados y orientación estratégica del desarrollo. Se garantizará que los asesores tengan acceso a reportes semanales y a la documentación relevante compartida de manera centralizada.

19.3 Calendario de reuniones

Se definirá un calendario de reuniones semanal para mantener un flujo de comunicación consistente y organizado:

- **Lunes:** Reunión de planificación semanal interna vía videollamada. Revisión de objetivos de la semana, asignación de tareas y detección de riesgos operativos.
- **Miércoles:** Reunión de seguimiento técnico vía videollamada. Discusión de avances en el desarrollo de módulos críticos, integración de sistemas y resolución de problemas de implementación.
- **Viernes:** Reunión de revisión con el asesor del proyecto vía videoconferencia. Presentación de resultados de la semana, validación de prototipos funcionales, retroalimentación sobre metodología y ajuste de prioridades.
- Comunicación diaria mediante Slack para dudas rápidas, coordinación de tareas urgentes y actualización de estatus de tickets o issues en la plataforma de gestión de proyectos (ej. GitHub Projects o Trello).

19.4 Reportes y documentación de comunicación

Se generarán reportes semanales que incluyan:

- Avances alcanzados y tareas completadas.

- Problemas encontrados y acciones correctivas.
- Ajustes a cronogramas o prioridades.
- Decisiones relevantes y acuerdos del equipo y asesores.

Estos reportes se compartirán con todos los miembros del equipo y el asesor del proyecto mediante Slack y documentos centralizados, asegurando transparencia y continuidad en la gestión de la información.

20 Esquema de negocio

El proyecto CoProof se plantea como una iniciativa sin fines de lucro, orientada a proporcionar a la comunidad académica y de investigación una plataforma de demostración formal colaborativa, accesible de manera gratuita. La intención es democratizar el acceso a herramientas avanzadas de verificación formal y promover la educación, la investigación y la colaboración en matemáticas, ciencias de la computación y áreas afines.

La monetización directa no es un objetivo del proyecto, pero se contempla la posibilidad de aceptar donativos voluntarios de individuos, instituciones o empresas que deseen apoyar el mantenimiento y la mejora continua de la plataforma. Estos fondos se destinarán exclusivamente a la operación, infraestructura, actualizaciones, capacitación y mejoras de seguridad del sistema, asegurando que la plataforma permanezca gratuita para todos los usuarios finales.

Parte de la retribución del proyecto se basa en la satisfacción de la comunidad y del equipo de desarrollo al ofrecer un entorno accesible, confiable y de alta calidad para la construcción y verificación de teoremas, así como para el aprendizaje de la lógica formal. Este aspecto intangible representa un valor significativo, ya que permite fortalecer la reputación de la iniciativa dentro de la comunidad académica y científica.

Adicionalmente, se contempla la integración opcional de modelos de lenguaje de pago (LLM comerciales) que podrían ofrecer capacidades avanzadas de inferencia, traducción o asistencia en demostraciones. En este caso, el acceso a dichas funcionalidades estaría sujeto a licencias externas, siendo la plataforma en sí misma gratuita y abierta, pero con la posibilidad de que los usuarios elijan voluntariamente si desean usar estas herramientas adicionales, generando un modelo híbrido que combina acceso universal con opciones premium opcionales.

Finalmente, se fomentará la participación activa de la comunidad open source mediante contribuciones de código, reportes de errores, mejoras en la documentación y desarrollo de nuevos agentes de IA, asegurando que el proyecto evolucione de manera colaborativa y sostenible en el largo plazo.

Referencias bibliográficas

Baba, K., Liu, C., Kurita, S., & Sannai, A. (2025). Prover agent: An agent-based framework for formal mathematical proofs. *arXiv preprint arXiv:2506.19923*. https://arxiv.org/abs/2506.19923

contributors, M. (2025). Lean copilot known caveats [Consultado el 1 de septiembre de 2025]. https://github.com/lean-dojo/LeanCopilot

Piotrowski, B., Fernández Mir, R., & Ayers, E. (2023). Machine-learned premise selection for lean. *arXiv preprint arXiv:2304.00994*. https://arxiv.org/abs/2304.00994

Song, P., Yang, K., & Anandkumar, A. (2025). Lean copilot: Large language models as copilots for theorem proving in lean. *Proceedings of the International Conference on Neuro-symbolic Systems*, *288*, 144–169. https://proceedings.mlr.press/v288/song25a.html

Tang, X. (2025). A comprehensive survey of the lean 4 theorem prover: Architecture, applications, and advances. *arXiv preprint arXiv:2501.18639*. https://arxiv.org/abs/2501.18639