UNIVERSIDAD CONTINENTAL

FACULTAD DE INGENIERÍA

ESCUELA ACADÉMICO PROFESIONAL DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

PROYECTO

“Tutor Virtual de Lectura Comprensiva Escolar - I.E.P. San Carlos”

PRESENTADO POR:

APELLIDOS Y NOMBRES	CÓDIGO
Huaccho Mancilla Steven José	72620878
Lazo Maraví Nilton Joel	74091904
Poma Goche Abigail Karim	72541050
Ramirez Basualdo Lenin Sebasthian	72695645
Robles Sanchez Britney Sheyla	72855637
Rojas Mellado Andrea Mirella	75411510

ASESOR:

Daniel Gamarra Moreno

HUANCAYO – PERÚ

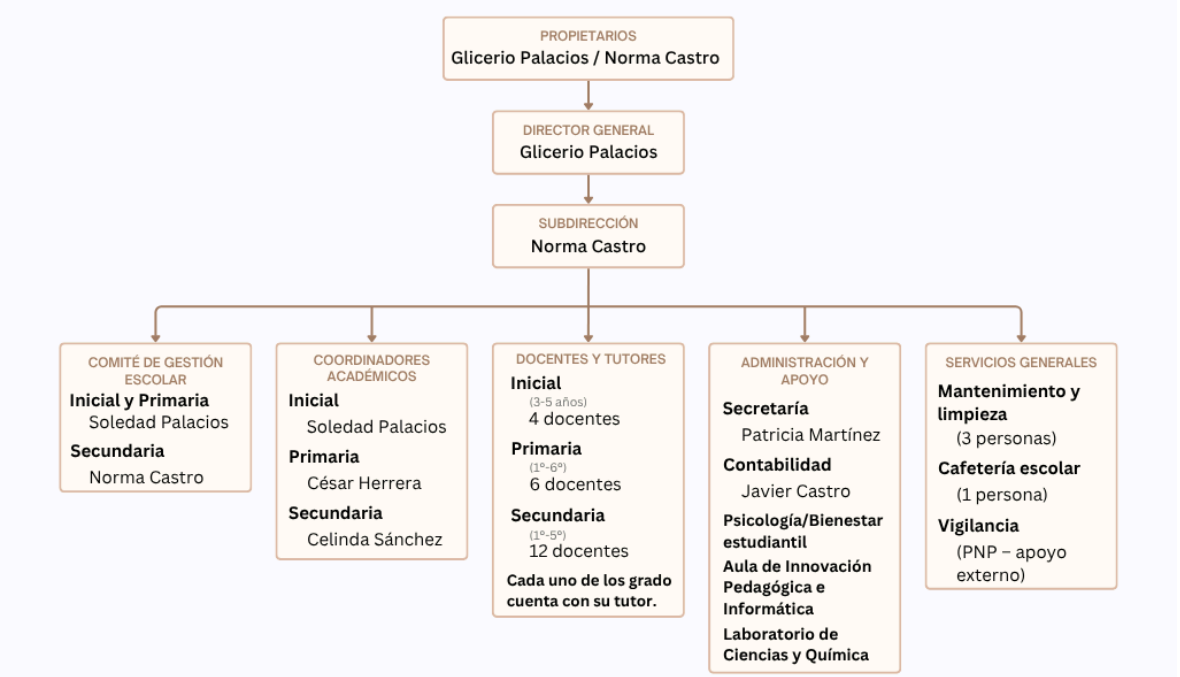
2023

LISTA DE CONTENIDO

CAPÍTULO 1 PLANTEAMIENTO DEL ESTUDIO

1.1. Aspectos Generales de la Empresa

1.1.1. Organigrama



1.1.2. Misión y Visión

- Misión

  “Somos una Institución Educativa que formamos líderes con una sólida formación en valores que les permita enfrentar con éxito los retos de este nuevo milenio.”

- Visión

  “Ser una institución Educativa, acreditada y líder, con lo más exigentes estándares de calidad que contribuya al logro de un sociedad desarrollada; con justicia, libertad, soberanía y verdadera democracia”

1.2. Diagnóstico del Problema

En la Institución Educativa “San Carlos” se ha identificado una problemática relacionada con el desarrollo limitado de la comprensión lectora y el pensamiento crítico en los estudiantes de los niveles inicial, primaria y secundaria.

Aunque el colegio promueve la lectura como eje de su propuesta pedagógica, existen dificultades en el seguimiento individual del progreso lector, la creación de actividades personalizadas y la retroalimentación inmediata a los alumnos.

Actualmente, los docentes elaboran manualmente las preguntas y ejercicios para cada texto, lo que demanda tiempo y dificulta una atención diferenciada según el ritmo de aprendizaje. Además, las lecturas se realizan de forma tradicional, sin registros sistemáticos de avance ni herramientas que permitan analizar resultados.

Como consecuencia, muchos estudiantes no consolidan habilidades de análisis e inferencia, y los docentes carecen de medios automatizados para monitorear y mejorar la comprensión lectora.

Frente a ello, se plantea el desarrollo del Tutor Virtual de Lectura Comprensiva Escolar, una aplicación web que integra inteligencia artificial y automatización para generar preguntas automáticas, brindar retroalimentación y ofrecer a los docentes un panel de seguimiento del rendimiento lector. El proyecto busca optimizar la gestión del proceso lector, fortalecer las competencias comunicativas y fomentar el hábito de lectura desde edades tempranas mediante el uso innovador de la tecnología educativa.

1.3. Procesos de la Empresa

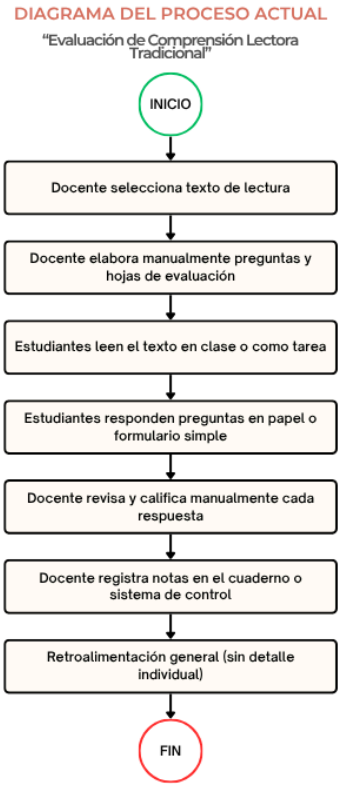
En la Institución Educativa “San Carlos”, el proceso de evaluación constituye uno de los pilares fundamentales para medir el avance académico y la formación integral de los estudiantes en los niveles de inicial, primaria y secundaria.

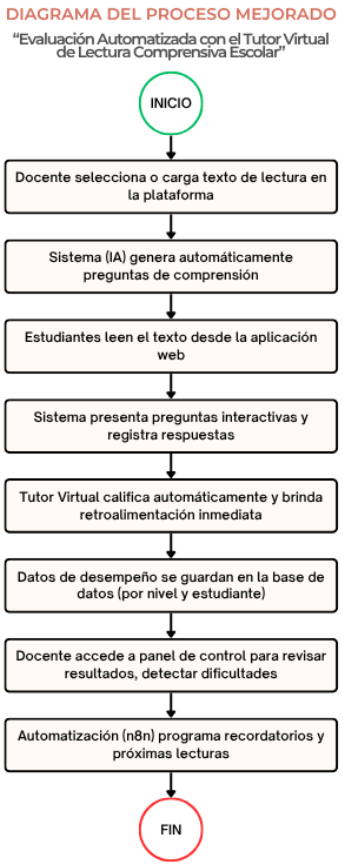
Actualmente, el sistema de evaluación se desarrolla mediante evaluaciones mensuales y bimestrales, complementadas en el nivel secundario con exámenes tipo admisión cada quince días, los cuales permiten medir la preparación de los alumnos frente a evaluaciones externas y reforzar el aprendizaje continuo.

Si bien este modelo permite mantener un control periódico del rendimiento académico, presenta limitaciones en la evaluación de competencias lectoras y de pensamiento crítico, ya que la valoración se centra principalmente en resultados cuantitativos y no en el proceso de comprensión o análisis del texto. Además, la corrección manual de los ejercicios y exámenes demanda un esfuerzo considerable del docente, dificultando la entrega de retroalimentación inmediata al estudiante.

Frente a ello, se identifica la necesidad de **optimizar el proceso de evaluación de la lectura comprensiva**, integrando herramientas tecnológicas que permitan **automatizar la generación de preguntas, calificación y seguimiento del desempeño lector**.

El **Tutor Virtual de Lectura Comprensiva Escolar** surge como una solución que complementa los procesos evaluativos tradicionales, permitiendo a los docentes realizar evaluaciones formativas continuas con apoyo de inteligencia artificial, mientras que los estudiantes pueden practicar la lectura y recibir retroalimentación automática antes de las evaluaciones mensuales o tipo admisión.





1.4. Oportunidad Encontrada

La Institución Educativa "San Carlos" cuenta con un entorno académico comprometido con la mejora continua del aprendizaje, lo que representa una oportunidad favorable para incorporar **herramientas tecnológicas que fortalezcan la comprensión lectora y el pensamiento crítico**.

El uso de metodologías tradicionales en las evaluaciones lectoras ha evidenciado la necesidad de **modernizar los procesos de enseñanza y evaluación**, permitiendo que los estudiantes practiquen de forma autónoma y los docentes dispongan de información precisa sobre el rendimiento de sus alumnos.

En este contexto, la implementación del **Tutor Virtual de Lectura Comprensiva Escolar** surge como una oportunidad para **integrar la inteligencia artificial y la automatización** dentro del proceso educativo, facilitando la generación de preguntas, la retroalimentación inmediata y el seguimiento del progreso lector por niveles.

De esta forma, el colegio puede **optimizar su gestión académica, reducir la carga manual de los docentes y mejorar la calidad del aprendizaje**, consolidándose como una institución innovadora que incorpora tecnología educativa al servicio de la formación integral de sus estudiantes.

1.5. Detalles del Proyecto

El proyecto Tutor Virtual de Lectura Comprensiva Escolar consiste en el desarrollo de una aplicación web educativa basada en el stack MERN (MongoDB, Express, React y Node.js), que busca fortalecer las habilidades de lectura y comprensión de los estudiantes de los niveles inicial, primaria y secundaria.

El sistema integra inteligencia artificial (IA) para generar preguntas automáticas a partir de textos y proporcionar retroalimentación inmediata, así como automatización mediante n8n para el envío de recordatorios, registro de actividades y programación de sesiones lectoras.

Además, incluye un panel de control para docentes, donde se visualiza el progreso de los estudiantes, su nivel de comprensión y los resultados de las evaluaciones por periodo (mensual, bimestral o tipo admisión).

La solución se desplegará en la nube, permitiendo acceso desde cualquier dispositivo con conexión a internet. Su diseño busca ser intuitivo, accesible y adaptable a cada nivel educativo, con el fin de promover la lectura comprensiva y reducir la carga administrativa de los docentes.

En conjunto, el proyecto contribuirá a digitalizar el proceso de evaluación lectora, mejorar la gestión académica del colegio y fomentar el uso responsable e innovador de la tecnología en el aula.

CAPÍTULO 2 ESTUDIO DE FACTIBILIDAD

2.1. Alternativas de Solución

Tras analizar la problemática identificada en la Institución Educativa “San Carlos”, se evaluaron distintas alternativas orientadas a mejorar el proceso de evaluación lectora y el desarrollo de habilidades de comprensión en los estudiantes:

- **Alternativa 1: Mantener el proceso actual de evaluación manual.**
  Consiste en continuar utilizando exámenes impresos o formularios digitales básicos elaborados por los docentes. Aunque no requiere inversión tecnológica, mantiene las mismas limitaciones: alta carga de trabajo docente, ausencia de retroalimentación inmediata y escaso seguimiento individual del progreso lector.
- **Alternativa 2: Implementar plataformas externas de lectura.**
  Implica adoptar herramientas o sitios web ya existentes para practicar comprensión lectora. Sin embargo, la mayoría de estas plataformas están orientadas a contextos generales o internacionales, carecen de adaptación curricular al entorno nacional y no ofrecen control directo a los docentes del colegio.
- **Alternativa 3: Desarrollar el Tutor Virtual de Lectura Comprensiva Escolar.**
  Propone crear una plataforma propia e integrada al modelo pedagógico institucional. El sistema automatiza la generación de preguntas mediante inteligencia artificial, evalúa las respuestas en tiempo real y proporciona a los docentes un panel de seguimiento del progreso lector. Además, incorpora automatización (n8n) para programar recordatorios, registrar resultados y facilitar la gestión académica.

Entre las alternativas analizadas, esta última resulta la más viable e innovadora, ya que digitaliza el proceso evaluativo, optimiza el tiempo docente y fortalece las competencias lectoras de los estudiantes en todos los niveles educativos.

2.2. Factibilidad Técnica

El proyecto **Tutor Virtual de Lectura Comprensiva Escolar** es técnicamente viable, dado que utiliza tecnologías probadas y de libre acceso, lo que reduce significativamente los costos de desarrollo y mantenimiento.

El sistema se desarrollará bajo la arquitectura **MERN (MongoDB, Express, React y Node.js)**, con integración de **inteligencia artificial** mediante API NLP (Hugging Face u Ollama) y **automatización** con **n8n**, todas herramientas disponibles en versiones gratuitas o de código abierto.

Asimismo, el proyecto se apoyará en plataformas de despliegue **Vercel** y **MongoDB Atlas Free Tier**, lo que garantiza accesibilidad sin requerir inversión en infraestructura. Las pruebas y control de calidad se gestionan con **Jest** y **Cypress**, y la integración continua se automatizará mediante **GitHub Actions** y **Docker**.

El equipo cuenta con los conocimientos necesarios en desarrollo web full-stack, control de versiones y metodologías ágiles, por lo que la factibilidad técnica del proyecto es **alta**.

2.2.1. Recursos tecnológicos

- **Frontend:** React.js con diseño responsive y componentes reutilizables.
- **Backend:** Node.js con Express, autenticación JWT y conexión a MongoDB Atlas.
- **Base de datos:** MongoDB (Free Tier en la nube).
- **IA:** Modelos NLP (Hugging Face / Ollama) para generación de preguntas.
- **Automatización:** n8n self-hosted para notificaciones y registro automático.
- **Integración y despliegue:** GitHub, Docker y Vercel.

2.2.2. Requerimientos del entorno

- **Hardware mínimo de desarrollo:** Procesador Intel i5 o superior, 8 GB RAM, 512 GB SSD.
- **Software base:** Node.js 20+, MongoDB Compass, VS Code, Docker Desktop.
- **Recursos en la nube:** Vercel y MongoDB Atlas (planes gratuitos).
- **Compatibilidad:** Navegadores modernos (Chrome, Edge, Firefox).

  El entorno propuesto asegura una ejecución estable, escalable y sin costo operativo significativo.

2.3. Factibilidad Económica

El proyecto es económicamente viable, ya que aprovecha herramientas gratuitas y la participación de un equipo académico multidisciplinario.

Según el presupuesto consolidado (informe técnico y hoja de cálculo), el **costo total del proyecto** asciende a **4 586 USD**, distribuidos en recursos humanos, desarrollo, pruebas y documentación.

2.3.1. Gastos Generales

Fase	**Costo estimado (USD)**
Inicio	122
Planificación	308
Planificación 2	265
Desarrollo	2 776
Pruebas y entrega	1 115
Total	**4 586 USD**

Los principales gastos se asocian al tiempo de dedicación del equipo (horas de programación, diseño y pruebas), sin requerir inversiones en infraestructura o licencias.

El uso de tecnologías open source y servicios en la nube gratuitos permite mantener la sostenibilidad del sistema en el mediano plazo.

Por tanto, la factibilidad económica se califica como alta, con retorno académico y funcional inmediato.

2.4. Factibilidad Operacional

La solución propuesta es operativamente viable, ya que se integra de forma natural a la dinámica pedagógica del colegio. Los docentes podrán utilizar la plataforma para asignar lecturas y realizar evaluaciones automáticas, mientras que los estudiantes accederán a las actividades desde cualquier dispositivo con conexión a internet.

El sistema está diseñado con una **interfaz intuitiva y adaptable por nivel educativo**, lo que garantiza su aceptación tanto por estudiantes de primaria y secundaria como por el personal docente. Además, los flujos automatizados reducen la carga administrativa y facilitan la evaluación continua.

2.4.1. Sistema de Tutor de lectura Crítica

El sistema permitirá:

- Generar y presentar **preguntas automáticas** sobre los textos leídos.
- Brindar **retroalimentación inmediata** según las respuestas del estudiante.

- Registrar resultados y progreso en la **base de datos central**.

- Ofrecer un **panel docente** para monitorear comprensión y rendimiento.
- Automatizar notificaciones y recordatorios mediante **n8n**.

Gracias a su diseño modular y escalable, el sistema puede implementarse progresivamente en todos los niveles del colegio, consolidando una **gestión educativa moderna, eficiente y digitalizada**.

CAPÍTULO 3 ANÁLISIS DE REQUERIMIENTOS

3.1. Metas del Sistema de Información

El proyecto Tutor Virtual de Lectura Comprensiva Escolar tiene como meta principal fortalecer las habilidades de comprensión lectora y pensamiento crítico de los estudiantes de la Institución Educativa “San Carlos”, mediante una plataforma web con inteligencia artificial que automatice el proceso de evaluación y retroalimentación.

3.2. Requisitos del Sistema

El sistema se desarrolla con la arquitectura MERN (MongoDB, Express, React y Node.js) y servicios complementarios como n8n para automatización y modelos de IA para análisis de texto.
Estas herramientas permiten un sistema escalable, seguro y compatible con los recursos del colegio sin requerir grandes inversiones.

3.2.1. Requerimientos Funcionales

Código	Requerimiento funcional	Descripción
RF01	Gestión de usuarios	Permitir el registro e inicio de sesión de alumnos y docentes
RF02	Asignación de lecturas	El docente podrá subir o seleccionar textos según grado y nivel
RF03	Generación automática de preguntas	El sistema, mediante IA, generará preguntas de comprensión lectora adaptadas al texto.

RF04	Resolución de actividades	El alumno podrá responder las preguntas y enviar sus resultados.
RF05	Retroalimentación inmediata	El sistema mostrará comentarios automáticos sobre aciertos y errores.
RF06	Registro de resultados	Guardar el puntaje y desempeño de cada estudiante.
RF07	Reportes de desempeño	Generar gráficos e informes por alumno, aula o nivel educativo.
RF08	Gestión de contenidos	El coordinador podrá modificar o eliminar textos y usuarios.
RF09	Comunicación docente–alumno	Habilitar mensajes automáticos de recordatorio o aviso de evaluación.

3.2.2. Requerimientos no Funcionales

Código	Requerimiento no funcional	Descripción
RNF01	Accesibilidad	Interfaz intuitiva, adaptable a dispositivos móviles.
RNF02	Rendimiento	El sistema responderá a cada solicitud en menos de 3 segundos.
RNF03	Seguridad	Autenticación JWT y contraseñas cifradas con bcrypt.
RNF04	Escalabilidad	Permitir la incorporación de nuevos grados y usuarios sin reconfiguración.
RNF05	Disponibilidad	Acceso en línea 24/7 mediante Vercel y MongoDB Atlas Free Tier.
RNF06	Usabilidad	Diseñado con componentes claros y navegación guiada.
RNF07	Mantenibilidad	Código modular y documentado bajo estándares MERN.

3.3. Identificación de Actores del Sistema

El usuario cuenta con tres actores principales:

Actor	Descripción	Funciones Principales
Coordinador Académico	Encargado de la gestión global del sistema.	Administra usuarios (docentes/alumnos), supervisa reportes y aprueba contenidos
Docente	Responsable de asignar lecturas y evaluar resultados.	Asigna textos, revisa reportes, orienta a los estudiantes y analiza el avance.
Alumno	Usuario principal del sistema. Estudiantes de primaria o secundaria.	Inicia sesión, realiza lecturas, responde preguntas, revisa retroalimentación y consulta su progreso.

3.3.1. Relación con los procesos Educativos

El sistema se integra con los procesos de evaluación mensual y bimestral que realiza la institución, automatizando la parte de comprensión lectora y generando resultados comparativos por grupo y periodo.

CAPÍTULO 4 PLANIFICACIÓN DEL PROYECTO

4.1. Definición de Roles de Trabajo

El proyecto “Tutor Virtual de Lectura Comprensiva Escolar - I.E.P. San Carlos” se gestiona en Jira con enfoque Scrum. Los roles que se consignan a continuación provienen de las actas y documentos del equipo.

4.1.1. Product Owner

Nombre: Nilton Joel Lazo Maraví

Responsabilidades: Proteger la visión del producto; priorizar y mantener el Product Backlog en Jira, validar criterios de aceptación, aceptar o rechazar incrementos, coordinar con las partes interesadas académicas y velar por el valor entregado en cada sprint.

4.1.2. Scrum master

Nombre: Rol compartido por el equipo

Responsabilidades: Facilitar ceremonias, proteger el time-box, remover impedimentos, promover la mejora continua del proceso, limitar reuniones a ≤60 minutos y dar a conocer los criterios de aceptación en el refinamiento.

4.1.3. Team member

Nombres: Huaccho Mancilla Steven José, Robles Sánchez Britney Sheyla, Poma Goche Abigail Karim, Rojas Mellado Andrea Mirella

Responsabilidades: análisis y diseño, desarrollo MERN (React + Vite + Tailwind, Express, MongoDB) e integración de Ollama y n8n, pruebas unitarias, de integración y E2E, documentación técnica y demos de sprint.

4.1.4. Tester

Nombre: Rol compartido por el equipo
Responsabilidades: Definir y mantener DoR/DoD, diseñar y ejecutar pruebas, resguardar evidencia en Jira, medir cobertura, apoyar la automatización (CI/CD con Jest) y el aseguramiento de calidad.

Patrocinio y partes interesadas: Ramírez Basualdo Lenin Sebasthian (patrocinador); docente evaluador del curso (interesado académico).

4.2. Product Backlog

Criterios de priorización: Valor educativo (impacto en estudiantes y docentes), factibilidad técnica (modelo local con Ollama y automatizaciones en n8n) y necesidad de analítica para seguimiento institucional.

Historias de usuario y estado actual:

- **HU01 – Recibir preguntas (8 pts, Finalizada):** El sistema genera ≥5 preguntas por texto; si falla el procesamiento, muestra error sin resultados vacíos.
- **HU02 – Recibir retroalimentación (5 pts, Finalizada):** Tras responder, muestra correcto/incorrecto con breve explicación; si no hay respuestas, advierte sin bloquear.
- **HU05 – Acceder a reportes (5 pts, Finalizada):** Reporte por estudiante con % de comprensión y actividad; si no hay datos, lo informa.
- **HU06 – Obtener informe (5 pts, Finalizada):** Dashboard con promedios de curso; si no hay estudiantes, muestra “No hay información disponible”.
- **HU03 – Progreso acumulado (3 pts, Finalizada):** Visualización del progreso del estudiante; si no hay registros, “Sin datos disponibles”.
- **HU07 – Indicar sesgos del texto (13 pts, Finalizada):** Detección y etiquetas de sesgos/falacias; si no se puede analizar, notifica sin resultados vacíos.
- **HU04 – Asignar actividades (8 pts, Finalizada):** El docente sube texto y lo asigna a estudiantes; si el formato no es soportado, se notifica y no se asigna.
- **HU08 – Enviar recordatorios de lectura (5 pts, En curso):** n8n agenda y envía recordatorios; ante fallo de envío, registra y reintenta.
- **HU09 – Mostrar panel de comprensión (3 pts, En curso):** KPIs agregados (promedio general); si no hay datos suficientes, se indica.

4.3. Sprint Backlog

4.3.1. Sprint 1

Objetivo: MVP navegable con generación de preguntas, retroalimentación y visualización mínima de desempeño.

Resultado: HU01 (8), HU02 (5), HU05 (5), HU06 (5): 23 puntos entregados.

4.3.2. Sprint 2

Objetivo de producto: Analítica personal (HU03) y análisis de sesgos con IA (HU07).

Resultado: HU03 (3) y HU07 (13): 16 puntos entregados.

4.3.3. Sprint 3

Objetivo de producto: Operación docente (HU04), hábitos de lectura con n8n (HU08) y panel global (HU09).

Carga planificada: 16 puntos.

Hito de calendario: Martes 2 de diciembre del 2025.

4.4. Planificación de Sprints

4.4.1. Historias de usuario

Todas las HU están documentadas en Jira con descripción y criterios de aceptación verificables. La DoR exige historia comprensible, dependencias explícitas y criterios de prueba; la DoD exige pruebas ejecutadas, evidencia en Jira y demo en la revisión de sprint.

4.4.2. Priorización de historias de usuario

Orden establecido: (1) aprendizaje esencial del estudiante y retroalimentación (HU01, HU02), (2) visibilidad docente mínima (HU05, HU06), (3) analítica personal y sesgos (HU03, HU07), (4) flujo operativo docente y hábitos (HU04, HU08), (5) analítica institucional (HU09). Este orden maximiza valor temprano y reduce riesgo técnico.

4.5. Cronograma de Actividades

Fase / Sprint	Fechas	Entregables principales
Sprint 1	10/sep - 08/oct	HU01, HU02, HU05, HU06 (MVP validado)
Sprint 2	09/oct - 04/nov	HU03, HU07 (analítica personal + IA de sesgos)
Sprint 3	05/nov - 02/dic	HU04, HU08, HU09 (operación docente, hábitos con n8n y panel global)
Cierre	03/dic	Demo final, documentación, empaquetado Docker, checklist de calidad

4.6. Gestión de riesgos

ID	Riesgo	Probabilidad	Impacto	Respuesta / Mitigación	Responsable
R1	Latencia/precisión del modelo (Ollama) afecta experiencia (HU01/HU02/HU07).	Media	Alta	Spike técnico de latencia; profiling; caching; control de tokens; métricas p95; mensajes de estado claros.	Equipo (dev)
R2	Cobertura de pruebas insuficiente retrasa cierres DoD.	Media	Alta	CI/CD con Jest en cada PR; umbral objetivo ≥80%; casos de prueba alineados a criterios de aceptación.	Equipo (QA compartido)
R3	Datos insuficientes para reportes/panel (HU05–HU09).	Media	Media	Estados “sin datos”; datos semilla para pruebas; validación previa a demo.	Equipo (dev)
R4	Reuniones extensas reducen capacidad efectiva del sprint.	Media	Media	Time-box ≤60 min; agenda y objetivos por ceremonia; seguimiento del cumplimiento.	Rol SM (compartido)
R5	Integración y conectividad de n8n (HU08) genera fallas de envío.	Media	Media	Reintentos con backoff; logging y trazabilidad; cola de reenvíos; notificación de error al usuario.	Equipo (dev)
R6	Limitaciones de free-tiers (Atlas, n8n, etc.) producen cuellos de botella.	Baja	Media	Plan B local; monitoreo de cuotas; tareas batch fuera de horas pico.	PO + equipo

4.7. Aseguramiento de la calidad

4.7.1. Heurísticas de Nielsen (10 ítems, Likert 1–5)

1. El sistema informa claramente al usuario sobre las acciones en curso o resultados.
2. Los términos usados son comprensibles para el usuario promedio.
3. El sistema permite deshacer o cancelar acciones fácilmente.
4. Colores, tipografías y estilos son coherentes.
5. El sistema evita errores mediante validaciones previas.
6. Se muestran alertas antes de ejecutar acciones críticas.
7. El sistema ofrece menús que evitan depender de la memoria del usuario.
8. Existen atajos o funciones rápidas para usuarios expertos.
9. La interfaz evita elementos innecesarios que distraen.
10. Los mensajes de error indican la causa y la posible solución.

Aplicación y reporte: Al cierre del Sprint 3 (post-demo) con estudiantes y docentes, calcular media por ítem, promedio global y desviación estándar, presentar 3 hallazgos clave con acciones de mejora asociadas.

4.7.2. SUS – System Usability Scale (10 ítems, Likert 1–5)

1. Creo que me gustaría usar este sistema con frecuencia.
2. Encontré el sistema innecesariamente complejo.
3. Considero que el sistema fue fácil de usar.
4. Pienso que necesitaría apoyo técnico para usar el sistema.
5. Las funciones del sistema estaban bien integradas.
6. Me parece que hubo demasiada inconsistencia en el sistema.
7. Imagino que la mayoría de las personas aprenderían a usarlo rápidamente.
8. Encontré el sistema muy engorroso de usar.
9. Me sentí confiado al usar el sistema.
10. Necesité aprender muchas cosas antes de poder usar el sistema.

Cálculo del puntaje SUS: recodificar ítems positivos (1,3,5,7,9) como (respuesta − 1) e ítems negativos (2,4,6,8,10) como (5 − respuesta); sumar y multiplicar × 2.5 → SUS (0–100). Interpretación: ≥68 aceptable, ≥80 excelente.

CAPÍTULO 5 DISEÑO DEL SISTEMA DE INFORMACIÓN

5.1. Diseño de Diagramas UML

El diseño UML describe gráficamente la estructura y el comportamiento del **Tutor Virtual de Lectura Comprensiva Escolar**, permitiendo visualizar las interacciones entre alumnos, docentes y el sistema.

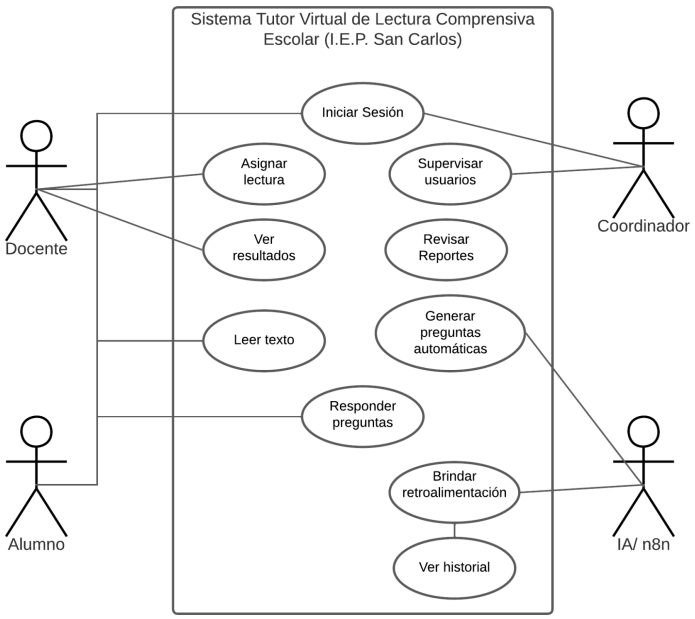
El modelo se elaboró considerando los requerimientos funcionales definidos y las principales operaciones: **asignar lecturas, generar preguntas automáticas, responder actividades, registrar resultados y brindar retroalimentación.**

5.1.1. Diagramas de casos de uso

El Tutor Virtual de Lectura Comprensiva Escolar permite a los alumnos realizar lecturas interactivas y responder preguntas automáticas, a los docentes asignar lecturas y monitorear resultados, y al administrador gestionar usuarios y contenidos.

Actores principales:

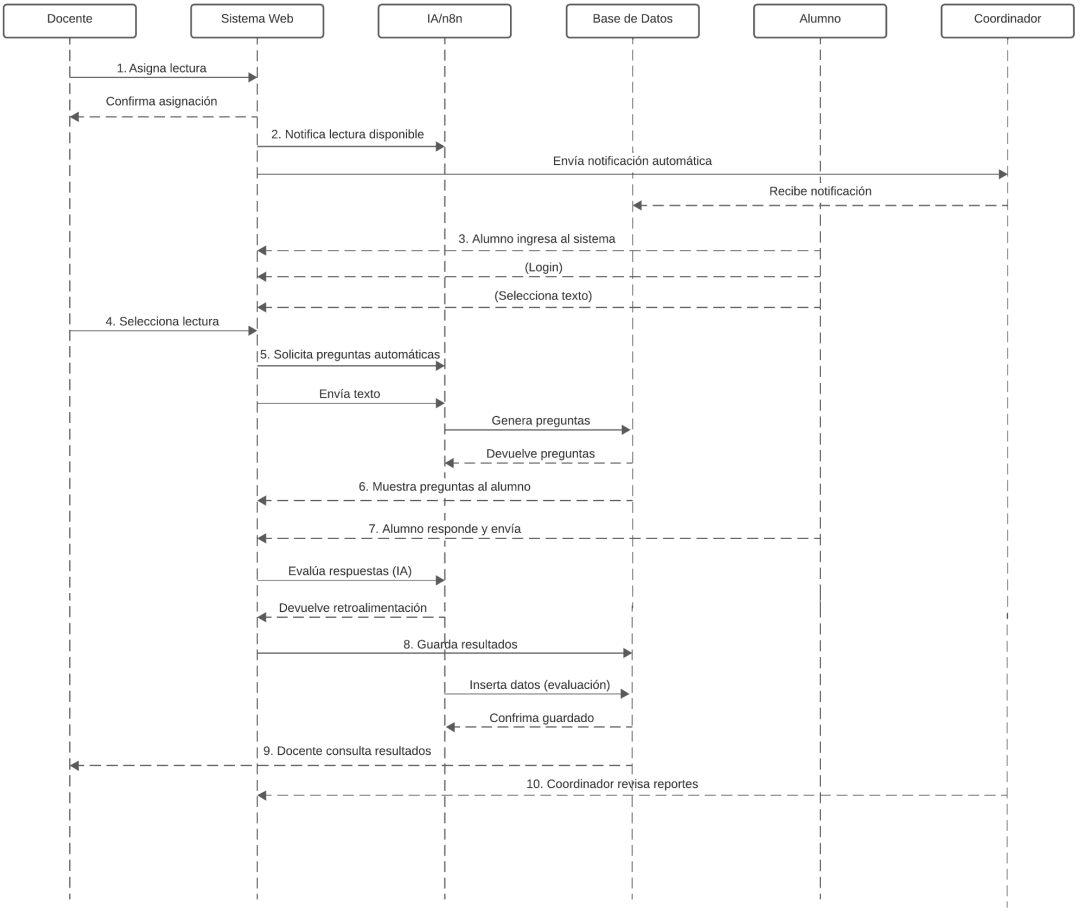
- **Docente:** Asigna lecturas, revisa resultados y genera reportes.
- **Alumno:** Lee textos, responde preguntas y recibe retroalimentación.
- **Coordinador:** Supervisa cuentas, reportes y controla el correcto funcionamiento del sistema.
- **Sistema (IA / n8n):** Genera preguntas automáticas, evalúa respuestas y envía recordatorios.



5.1.2. Diagramas de secuencia

El diagrama de secuencia muestra cómo interactúan los actores (Alumno, Docente, Coordinador y el Sistema con IA/n8n) en el flujo principal de uso: evaluar la comprensión lectora.

Este proceso comienza cuando el docente asigna una lectura y culmina con la generación automática de resultados y retroalimentación para el alumno.



5.1.3. Diagramas de colaboración

5.1.4. Diagramas de clases

5.2. Diseño de base de datos

5.2.1. Diseño conceptual (E/R)

5.2.2. Diseño lógico

5.2.3. Diseño físico

5.2.4. Modelado de base de datos

5.3. Diseño de Interfaces Básicas

5.3.1. Acceso login

5.3.2. Interfaz

5.3.3. ….

5.3.4. ….

CAPÍTULO 6 CODIFICACIÓN DEL SOFTWARE

6.1. Desarrollo del Sprint 1

6.1.1. Sprint planning

Objetivo del Sprint 1:

Desarrollar el MVP (Producto Mínimo Viable) del Tutor Virtual de Lectura Comprensiva Escolar, que permita al estudiante recibir preguntas generadas automáticamente, obtener retroalimentación inmediata y al docente visualizar reportes e informes básicos de desempeño.

Duración: 10 de septiembre – 8 de octubre de 2025

Historias de usuario planificadas:

- **HU01 – Recibir preguntas:** Generación automática de preguntas mediante IA a partir del texto proporcionado.
- **HU02 – Recibir retroalimentación:** Mostrar al estudiante comentarios automáticos sobre sus respuestas.
- **HU05 – Acceder a reportes:** Permitir la visualización de resultados individuales por parte del docente.
- **HU06 – Obtener informe:** Mostrar un dashboard con indicadores generales de comprensión.

Objetivos técnicos:

- Implementar el flujo básico de interacción entre frontend y backend.
- Conectar la API de generación de preguntas (modelo local con Ollama).
- Implementar el módulo de retroalimentación en tiempo real.
- Crear endpoints para reportes e informes de comprensión lectora.
- Diseñar el dashboard docente inicial con estadísticas básicas.

6.1.2. Sprint backlog

Nº	Historia	Tarea técnica	Responsable
1	HU01	Crear endpoint `/api/generate-questions` con conexión a modelo IA local (Ollama).	Joel
2	HU01	Implementar vista de lectura y renderizado dinámico de preguntas en React.	Steven
3	HU02	Diseñar función de evaluación automática de respuestas y mostrar retroalimentación.	Andrea
4	HU02	Conectar respuestas del usuario con la API y guardar resultados en MongoDB.	Britney
5	HU05	Crear endpoint `/api/reports/student/:id` para devolver datos de desempeño individual.	Karim

6	HU06	Implementar componente `DashboardDocente.tsx` con gráficos básicos (Chart.js).	Steven
7	—	Validar comunicación front–back y control de errores (status 400–500).	Equipo
8	—	Pruebas unitarias con Jest y documentación técnica en Jira.	Equipo

6.1.3. Historias de usuario

HU01 – Recibir preguntas

Como estudiante, quiero recibir preguntas generadas automáticamente según el texto que lea, para evaluar mi comprensión.

Criterio de aceptación:
El sistema genera al menos 5 preguntas coherentes por texto y muestra error controlado si falla el procesamiento.

HU02 – Recibir retroalimentación

Como estudiante, quiero recibir comentarios automáticos después de responder, para conocer mis aciertos y errores.

Criterio de aceptación:
Tras enviar las respuestas, el sistema muestra resultado correcto/incorrecto con explicación breve.

HU05 – Acceder a reportes

Como docente, quiero acceder a reportes individuales de los estudiantes, para analizar su nivel de comprensión.

Criterio de aceptación:
El sistema genera reportes por estudiante con porcentaje de comprensión y actividades realizadas.

HU06 – Obtener informe

Como docente, quiero visualizar un panel con indicadores generales del aula, para tener una vista global del desempeño.

Criterio de aceptación:
El dashboard muestra promedio de comprensión por curso y el estado de participación de los estudiantes.

6.1.4. Taskboard

Tareas "Done":

- Endpoint /api/generate-questions funcional con conexión al modelo Ollama local.
- Interfaz de lectura y preguntas en React implementada.
- Lógica de retroalimentación automática completada.
- Reporte individual y dashboard docente funcionales.
- Validaciones y manejo de errores implementados.
- Evidencias de ejecución y pruebas registradas en Jira.

6.1.5. Daily scrum

Reuniones diarias realizadas durante el sprint:

Día	Temas clave
1	Definición de arquitectura MERN y flujo general de endpoints.
3	Integración inicial con modelo IA (Ollama) y control de respuestas vacías.
5	Diseño del componente de retroalimentación y pruebas unitarias iniciales.
7	Pruebas de conexión entre backend y frontend, corrección de latencias.
9	Implementación del dashboard y revisión de métricas en MongoDB.
10	Validación final del MVP y preparación para la demo del sprint.

6.1.6. Sprint review

Logros principales:

- Generación automática de preguntas funcional y estable.
- Retroalimentación inmediata operativa con respuestas correctas/incorrectas.
- Reportes individuales y dashboard docente accesibles.
- Flujo de datos coherente entre el backend (Node.js/Express) y el frontend (React).

Feedback recibido:

- Mejorar la variedad de preguntas generadas por la IA.
- Incorporar colores diferenciados en las respuestas correctas e incorrectas.
- Añadir opción de exportar los reportes en formato PDF en sprints posteriores.

6.1.7. Criterios de aceptación

- El sistema genera un mínimo de 5 preguntas por texto.
- Las respuestas muestran retroalimentación inmediata.
- Los reportes individuales y globales son accesibles sin errores.
- Los datos se almacenan correctamente en MongoDB Atlas.
- Todos los endpoints responden en menos de 3 segundos.
- La demo del MVP es navegable y funcional.

6.1.8. Resultados del sprint

6.1.8.1. Evidencias

6.1.8.2. Prueba de desarrollo

6.1.9. Sprint retrospective

Fortalezas:

- Excelente coordinación del equipo durante el desarrollo.
- Integración efectiva del modelo IA con el backend.
- Cumplimiento del 100% de las HU planificadas.
- Base sólida para los siguientes incrementos (Sprint 2 y 3).

Debilidades:

- Falta de validación avanzada de calidad de preguntas.
- UI inicial simple, con margen para mejoras de usabilidad.

Acciones para el siguiente sprint:

- Implementar barra de progreso por actividad lectora (HU03).
- Mejorar la precisión del análisis de sesgos (HU07).
- Añadir pruebas automatizadas para generación de preguntas.

6.2. Desarrollo del Sprint 2

6.2.1. Sprint planning

Objetivo del Sprint 2:

Implementar la gestión y visualización del progreso del estudiante y la detección de sesgos en los textos de lectura, conectando la base de datos con el panel de análisis interactivo.

Duración: 9 de octubre – 4 de noviembre de 2025

Historias de usuario planificadas:

- **HU03 – Progreso acumulado:** permitir que el sistema registre y muestre el avance porcentual del estudiante según las actividades desarrolladas.
- **HU07 – Indicar sesgos del texto:** habilitar el análisis asistido por IA para detectar posibles sesgos o subjetividades dentro del texto leído.

Objetivos técnicos:

- Integrar la API backend con autenticación JWT.
- Diseñar endpoints REST para obtener y actualizar el progreso del estudiante (/student/activities, /student/progress).
- Implementar el middleware de control de roles (docente / estudiante).
- Desarrollar componentes React para mostrar el avance y análisis textual.

6.2.2. Sprint backlog

Nº	Historia	Tarea técnica	Responsable
1	HU03	Crear modelo **`Submission`** para registrar progreso	Steven
2	HU03	Endpoint **`GET / student/activities`** y **`POST / student/progress`**	Joel
3	HU07	Endpoint **`POST / ai/analyze-text`** para detectar sesgos con IA (local stub)	Karim

4	HU07	Integrar resultados de IA en interfaz de lectura	Andrea
5	HU03	Mostrar porcentaje de progreso en la interfaz **`StudentActivities.tsx`**	Sebas
6	—	Pruebas de API y conexión front–back	Sheyla

6.2.3. Historias de usuario

HU03 – Progreso acumulado

Como estudiante, quiero ver mi progreso general en cada actividad de lectura para conocer cuánto he avanzado.
Criterio de aceptación: El sistema guarda el porcentaje de avance y lo actualiza en tiempo real según las respuestas enviadas.

HU07 – Indicar sesgos del texto

Como docente, quiero que el sistema resalte posibles sesgos o subjetividades en el texto analizado para guiar la discusión en clase.
Criterio de aceptación: El asistente virtual marca en el texto las frases subjetivas o de juicio de valor.

6.2.4. Taskboard

Tareas “Done”:

- Endpoints /student/activities y /student/progress.
- Middleware auth.js con JWT y roles.
- Interfaz de progreso en panel de estudiante.
- Prototipo de IA para detección de sesgos.

6.2.5. Daily scrum

Reuniones diarias para revisar avances:

Día	Temas clave
1	Revisión de estructura de modelos y autenticación.
3	Integración de middleware y autorización por rol.
5	Pruebas de carga de progreso y visualización en frontend.
7	Ajustes de análisis de sesgos y optimización de estado React.

6.2.6. Sprint review

Logros:

- Los estudiantes pueden visualizar sus actividades y el porcentaje de progreso.
- Docentes observan los sesgos textuales analizados por el módulo de IA.
- Integración funcional entre backend y frontend.

Feedback:

- Reforzar la precisión del análisis de sesgos para el sprint siguiente.
- Agregar métricas visuales de progreso por color o barras.

6.2.7. Criterios de aceptación

- El estudiante puede ver todas sus actividades asignadas con su porcentaje de avance.
- Solo los usuarios autenticados acceden a sus datos.
- El docente recibe el resultado del análisis de sesgos en texto.
- Los datos se guardan en MongoDB y se actualizan sin errores de autorización.

6.2.8. Resultados del sprint

6.2.8.1. Evidencias

- Capturas del panel de progreso del estudiante.

- Logs de peticiones **`POST / student/progress`** y **`GET / student/activities`**.
- Demostración de análisis de sesgos IA en texto de lectura.
- Validación exitosa de tokens JWT y roles.

6.2.8.2. Prueba de desarrollo

Caso	Acción	Resultado
1	Enviar progreso desde interfaz	201 Created + % actualizado
2	Acceder sin token	401 No autorizado
3	Usuario sin rol válido	403 No autorizado
4	Analizar texto con IA local	Retorna palabras marcadas por sesgo

6.2.9. Sprint retrospective

Fortalezas:

- Integración completa entre API, roles y base de datos.
- Buen manejo de autenticación y validación de roles.

Debilidades:

- Tiempo limitado para ajustar el análisis IA.
- Falta de visualización gráfica más intuitiva del progreso.

Acciones para el Sprint 3:

- Consolidar el panel docente para asignar actividades (HU04).
- Añadir módulo de recordatorios automáticos (HU08).
- Generar reportes con métricas de comprensión (HU09).

6.3. Desarrollo del Sprint 3

6.3.1. Sprint planning

Objetivo del Sprint 3:

Implementar la funcionalidad completa de gestión docente,incluyendo la asignación de actividades, el sistema de recordatorios automatizados mediante n8n y el panel de comprensión global para administradores, consolidando así todas las capacidades del Tutor Virtual de Lectura Comprensiva.

Duración: 5 de noviembre - 2 de diciembre de 2025

Historias de usuario planificadas:

- **HU04 – Asignar actividades:** Permitir al docente subir textos y asignarlos a estudiantes específicos.
- **HU08 – Enviar recordatorios de lectura:** Implementar flujos automatizados con n8n para programar y enviar notificaciones.
- **HU09 – Mostrar panel de comprensión:** Desarrollar dashboard con métricas agregadas de comprensión lectora.

Objetivos técnicos:

- Implementar módulo completo de gestión docente con interfaz React.
- Configurar flujos de n8n para recordatorios automatizados.
- Desarrollar panel administrativo con gráficos avanzados.
- Integrar sistema de notificaciones por correo electrónico.
- Realizar pruebas de integración end-to-end.

6.3.2. Sprint backlog

N°	Historia	Tarea técnica	Responsable
1	HU04	Crear endpoint POST /api/assignments para asignar textos	Joel
2	HU04	Implementar componente AssignmentForm en React con validaciones	Steven

3	HU08	Configurar workflow n8n para recordatorios automáticos	Karim
4	HU08	Integrar webhooks n8n con backend para registro de actividades	Andrea
5	HU09	Desarrollar componente AdminDashboard con Chart.js	Britney
6	HU09	Crear endpoint GET /api/admin/metrics para datos agregados	Sebas
7	-	Implementar sistema de notificaciones por email	Equipo
8	-	Pruebas E2E con Cypress para flujos completos	Sheyla

6.3.3. Historias de usuario

HU04 – Asignar actividades

Como docente, quiero asignar textos con actividades a mis estudiantes para gestionar su aprendizaje.

Criterio de aceptación:

- El docente puede subir texto en formatos soportados (txt, pdf, docx).
- Puede seleccionar estudiantes o grupos específicos para asignar.
- Si el formato no es soportado, muestra error claro y no asigna.

HU08 – Enviar recordatorios de lectura

Como sistema, quiero programar y enviar recordatorios con n8n para mantener el engagement de los estudiantes.

Criterio de aceptación:

- n8n envía notificaciones en fecha/hora configurada.
- Ante fallo de envío, registra error y realiza reintentos automáticos.
- Integración completa con base de datos para registro de actividades.

HU09 – Mostrar panel de comprensión

Como administrador, quiero ver un panel con métricas de comprensión promedio para tener una visión global del rendimiento.
Criterio de aceptación:

- El panel muestra KPIs agregados (promedio general, tendencias).
- Si no hay datos suficientes, indica "No hay datos suficientes para mostrar".
- Gráficos interactivos con filtros por periodo y nivel educativo.

6.3.4. Taskboard

Tareas “Done”:

- Endpoint /api/assignments funcional con validación de archivos.
- Interfaz de asignación docente implementada en React.
- Workflows n8n configurados y operativos para recordatorios.
- Panel administrativo con métricas de comprensión.
- Sistema de notificaciones por email integrado.
- Pruebas E2E ejecutadas y aprobadas.

6.3.5. Daily scrum

Reuniones diarias realizadas durante el sprint:

Día	Temas clave
1	Definición de estructura de datos para asignaciones
3	Integración inicial de n8n con backend MongoDB
5	Desarrollo de componentes de gráficos para dashboard
7	Pruebas de flujo completo: asignación → notificación → registro
9	Optimización de rendimiento del panel administrativo
10	Validación final y preparación para demo del sprint

6.3.6. Sprint review

Logros:

- Módulo completo de gestión docente operativo.
- Sistema de recordatorios automatizados funcionando con n8n.
- Panel administrativo con métricas de comprensión implementado.
- Integración exitosa de todos los flujos del sistema.

Feedback:

- Mejorar la interfaz de asignación para selección masiva de estudiantes.
- Agregar más tipos de gráficos en el panel administrativo.
- Optimizar frecuencia de recordatorios para no saturar a usuarios.

6.3.7. Criterios de aceptación

- Los docentes pueden asignar textos a estudiantes/grupos específicos.
- El sistema envía recordatorios automáticos según programación.
- El panel administrativo muestra métricas agregadas de comprensión.
- Todos los flujos funcionan de extremo a extremo sin errores.
- La integración n8n-backend es estable y confiable.

6.3.8. Resultados del sprint

6.3.8.1. Evidencias

- Capturas del módulo de asignación docente.
- Logs de ejecución de workflows n8n.
- Demostración del panel administrativo con datos reales.
- Evidencias de notificaciones enviadas por correo.

6.3.8.2. Prueba de desarrollo

Caso	Acción	Resultado
1	Docente asigna texto a grupo	201 Created + notificación programada
2	n8n ejecuta recordatorio	Email enviado + registro en BD
3	Administrador consulta panel	Métricas mostradas correctamente
4	Intento de asignar archivo inválido	Error 400 con mensaje claro

6.3.9. Sprint retrospective

Fortalezas:

- Excelente coordinación en la integración de componentes complejos.
- n8n demostró ser una herramienta robusta para automatización.
- Panel administrativo superó expectativas de funcionalidad.
- Cumplimiento del 100% de las HU planificadas.

Debilidades:

- Curva de aprendizaje inicial con n8n más pronunciada de lo esperado.
- Tiempo limitado para optimizaciones de rendimiento en gráficos.

Acciones para mejora continua:

- Documentar mejores prácticas de uso de n8n.
- Planificar sprint de optimización post-entrega.
- Establecer monitoreo continuo de flujos automatizados.

Lecciones aprendidas:

- La planificación detallada de flujos de automatización es crucial.
- Las pruebas E2E son esenciales para sistemas con múltiples integraciones.
- La retroalimentación temprana de usuarios mejora significativamente la usabilidad.

CAPÍTULO 7 PRUEBAS DE SOFTWARE

7.1. Plan de Pruebas

CONCLUSIONES

RECOMENDACIONES

ANEXOS

Anexo 01. Manual Técnico

Anexo 02. Manual de Usuario

LISTA DE TABLAS

LISTA DE FIGURAS