

**UNIVERSIDAD CONTINENTAL**  
**FACULTAD DE INGENIERÍA**  
**ESCUELA ACADÉMICO PROFESIONAL DE INGENIERÍA DE**  
**SISTEMAS E INFORMÁTICA**



**PROYECTO**

**"Plataforma De Gestión De Proyectos Colaborativos Con Integración De  
Inteligencia Artificial para la EAP de ingeniería Industrial"**

**PRESENTADO POR:**

APELLIDOS Y NOMBRES	CÓDIGO
FERRUZO IZQUIERDO JOCABED ISABEL	72977471
CORILLA JUSCAMAITA SAID MARDUX	45432195
ORTEGA BATALLA BRAULIO CESAR	75142209
MUNIVE GUERRA JOSE ALEJANDRO	72554618
SEGURA MEZA GIOVANNY LUIS EDUARDO	72723304
SINCHE ALVARADO YERALD CRISTHIAN	71383938

**ASESOR:**

**Daniel Gamarra Moreno**

**HUANCAYO – PERÚ**

**2025**

## INDICE

<b>PORADA</b>	<b>1</b>
<b>CAPÍTULO 1</b>	<b>6</b>
<b>PLANTEAMIENTO DEL ESTUDIO</b>	<b>6</b>
1.1. Aspectos Generales de la Empresa	6
1.1.1. Organigrama	6
1.1.2. Misión y Visión	7
1.1.2.1. Misión:	7
1.1.2.2. Visión	7
1.2. Diagnóstico del Problema	7
1.3. Procesos de la Empresa	8
1.4. Oportunidad Encontrada	9
1.5. Detalles del Proyecto	10
<b>CAPÍTULO 2</b>	<b>11</b>
<b>ESTUDIO DE FACTIBILIDAD</b>	<b>11</b>
2.1. Alternativas de Solución	11
2.2. Factibilidad Técnica	11
2.2.1. Tecnologías Utilizadas	12
2.2.2. Infraestructura Requerida	13
2.2.3. Viabilidad de la Integración de la IA	13
2.3. Factibilidad Económica	13
2.3.1. Presupuesto estimado	14
2.4. Factibilidad Operacional	16
2.4.1. Adaptabilidad del Sistema al Entorno Real	16
2.4.2. Facilidad de Uso	16
2.4.3. Aceptación por Parte de los Usuarios	16
<b>CAPÍTULO 3</b>	<b>21</b>
<b>ANÁLISIS DE REQUERIMIENTOS</b>	<b>21</b>
3.1. Metas del Sistema de Información	21
3.2. Requisitos del Sistema	22
3.2.1. Requerimientos funcionales	22
1. Gestión de Proyectos	22
2. Comunicación en Tiempo Real	22
3. Integración de Inteligencia Artificial	23
4. Panel de Administración	23
3.2.2. Requerimientos no funcionales	24
1. Seguridad	24
2. Rendimiento	24
3. Usabilidad	24
4. Escalabilidad	24
5. Disponibilidad	24
3.3. Identificación de Actores del Sistema	25
3.3.1. Administrador	25

<b>CAPÍTULO 4</b>	<b>26</b>
<b>PLANIFICACIÓN DEL PROYECTO</b>	<b>26</b>
4.1. Definición de Roles de Trabajo	26
4.1.1. Product owner	26
4.1.2. Scrum master	27
4.1.3. Team member	27
4.1.4. Tester	27
4.2. Product Backlog	28
4.3. Sprint Backlog	29
4.3.1. Sprint 1	29
4.3.2. Sprint 2	29
4.3.3. Sprint 3	30
4.3.4. Sprint 4	30
4.4. Planificación de Sprints	30
4.4.1. Historias de usuario	30
4.4.2. Priorización de historias de usuario	31
4.5. Cronograma de Actividades	33
4.6. Gestión de Riesgos	34
<b>CAPÍTULO 5</b>	<b>43</b>
<b>DISEÑO DEL SISTEMA DE INFORMACIÓN</b>	<b>43</b>
5.1. Diseño de Diagramas UML	43
5.1.1. Diagrama de casos de uso	43
5.1.2. Diagramas de secuencia	44
5.1.3. Diagramas de colaboración	48
5.1.4. Diagrama de clases	49
5.2. Diseño de Base de Datos	49
5.2.1. Diseño conceptual (E/R)	49
5.2.2. Diseño lógico	52
5.2.3. Diseño físico	54
5.2.4. Modelado de base de datos	56
5.3. Diseño de Interfaces Básicas	57
5.3.1. Acceso login	57
5.3.2. Interfaz Registro	57
<b>CAPÍTULO 6</b>	<b>60</b>
<b>CODIFICACIÓN DEL SOFTWARE</b>	<b>60</b>
6.1. Desarrollo del Sprint 1	60
6.1.1. Sprint planning	60
6.1.2. Sprint backlog	60
6.1.3. Historias de usuarios	60
6.1.4. Taskboard	61
6.1.5. Daily scrum	61
6.1.6. Sprint review	63
6.1.7. Criterios de aceptación	64
6.1.8. Resultados del sprint	64

6.1.8.1. Evidencias.	64
6.1.9. Sprint retrospective	66
6.2. Desarrollo del Sprint 2	67
6.2.1. Sprint planning	67
6.2.2. Sprint backlog	67
6.2.3. Historias de usuarios	67
6.2.4. Taskboard	68
6.2.5. Daily scrum	68
6.2.6. Sprint review	71
6.2.7. Criterios de aceptación	71
6.2.8. Resultados del sprint	71
6.2.8.1. Evidencias.	71
6.2.9. Sprint retrospective	73
6.3. Desarrollo del Sprint 3	74
6.3.1. Sprint planning	74
6.3.2. Sprint backlog	74
6.3.3. Historias de usuarios	74
6.3.4. Taskboard	75
6.3.5. Daily scrum	75
6.3.6. Sprint review	78
6.3.7. Criterios de aceptación	78
6.3.8. Resultados del sprint	79
6.3.8.1. Evidencias.	79
6.3.9. Sprint retrospective	79
6.4. Desarrollo del Sprint 4	80
6.4.1. Sprint planning	80
6.4.2. Sprint backlog	80
6.4.3. Historias de usuarios	81
6.4.4. Taskboard	81
6.4.5. Daily scrum	81
6.4.6. Sprint review	84
6.4.7. Criterios de aceptación	84
6.4.8. Resultados del sprint	84
6.4.8.1. Evidencias.	85
6.4.9. Sprint retrospective	86
<b>CAPÍTULO 7</b>	<b>87</b>
<b>PRUEBAS DE SOFTWARE</b>	<b>87</b>
7.1. Plan de Pruebas	87
<b>CONCLUSIONES</b>	<b>91</b>
<b>RECOMENDACIONES</b>	<b>92</b>
<b>ANEXOS</b>	<b>93</b>
Anexo 01. Manual Técnico	94
Anexo 02. Manual de Usuario	95
Anexo 03. Pruebas	96

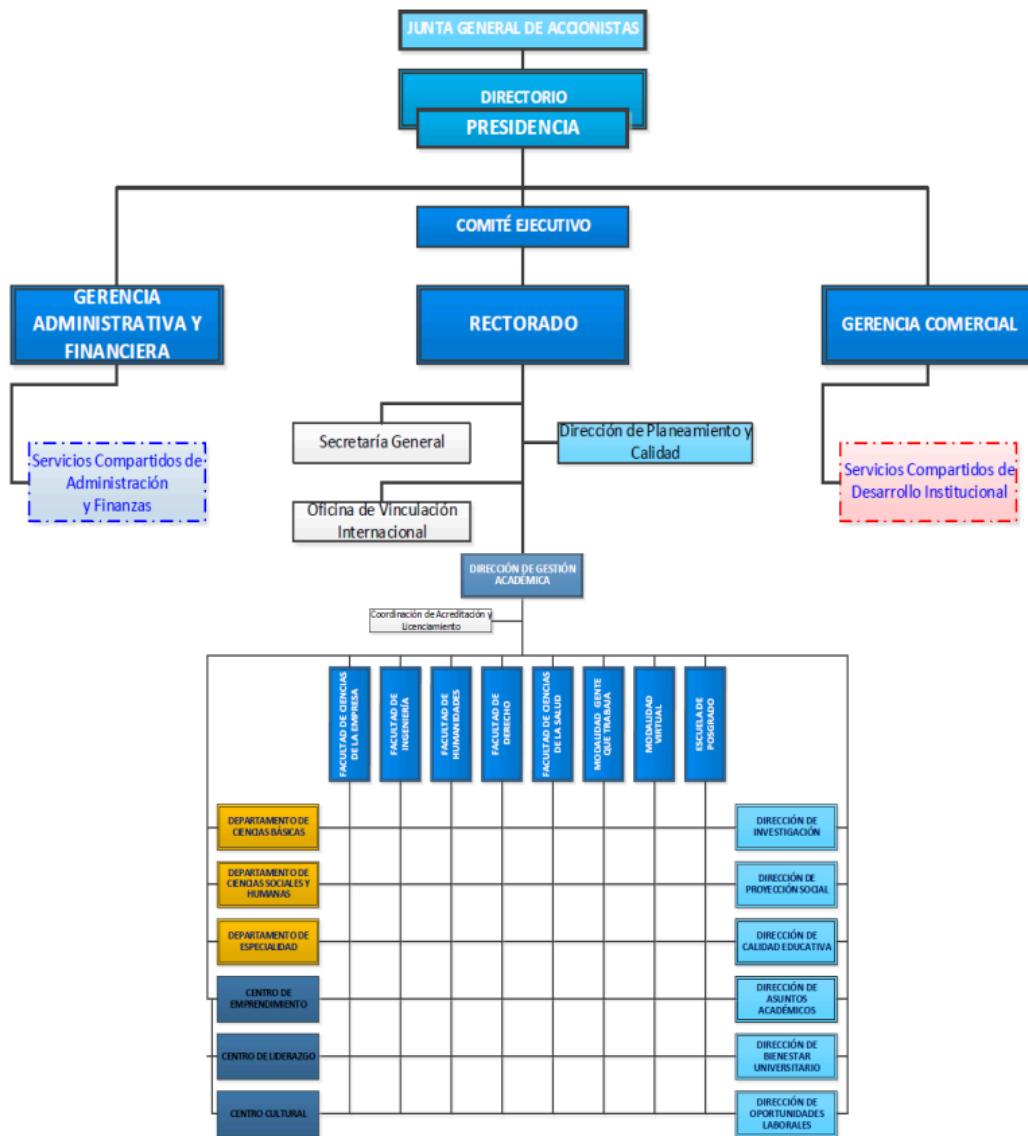


# CAPÍTULO 1

## PLANTEAMIENTO DEL ESTUDIO

### 1.1. Aspectos Generales de la Empresa

#### 1.1.1. Organigrama



### **1.1.2. Misión y Visión**

#### **1.1.2.1. Misión:**

Somos una organización de educación superior que conecta personas e ideas para impulsar la innovación y el bienestar integral a través de una cultura de pensamiento y acción emprendedora, la investigación y el impacto social.

#### **1.1.2.2. Visión**

Ser la mejor organización de educación superior posible que brinda oportunidades para el aprendizaje, la investigación y que une personas e ideas que buscan hacer realidad sueños y aspiraciones de prosperidad en un entorno incierto.

### **1.2. Diagnóstico del Problema**

En el contexto actual de la Escuela Académico Profesional (EAP) de Ingeniería Industrial, se ha identificado una necesidad crítica en el curso del noveno ciclo “Dirección de proyectos”. A pesar de la importancia estratégica que tiene esta asignatura en la formación profesional de los estudiantes, se carece de una herramienta virtual moderna que permita reforzar, complementar y aplicar los conocimientos adquiridos en un entorno práctico, interactivo y colaborativo.

Actualmente, los estudiantes gestionan sus proyectos utilizando herramientas genéricas o no integradas, lo que genera una serie de limitaciones: desorganización en la asignación de tareas, falta de trazabilidad en los avances, ausencia de comunicación en tiempo real y escasa visualización del rendimiento del equipo. Además, no se cuenta con un recurso tecnológico adaptado a las necesidades específicas del curso, que integre funcionalidades como gestión de tareas, tableros visuales, colaboración efectiva, y sobre todo, el aprovechamiento de tecnologías emergentes como la Inteligencia Artificial.

### **1.3. Procesos de la Empresa**

La Escuela Académico Profesional (EAP) de Ingeniería Industrial, como parte de su plan de estudios, incluye la asignatura obligatoria “Dirección de proyectos”, la cual es fundamental para el desarrollo de competencias clave como la Gestión de Proyectos y la Comunicación Efectiva. Esta asignatura, de naturaleza práctica y aplicada, se organiza en torno al diseño, planificación, ejecución y cierre de proyectos de ingeniería, por lo cual implica una serie de procesos académicos bien definidos.

Los procesos académicos actuales del curso incluyen:

- Formulación y desarrollo de proyectos en equipos:

Los estudiantes deben organizarse en grupos y desarrollar proyectos reales o simulados, aplicando buenas prácticas de dirección de proyectos.

- Aplicación de técnicas y herramientas de gestión:

Se requiere el uso de herramientas para la planificación del alcance, cronograma, costos, calidad, recursos, riesgos, comunicaciones y adquisiciones.

En la práctica, muchos equipos recurren a herramientas genéricas y dispersas (documentos compartidos, hojas de cálculo, mensajería instantánea), lo que dificulta la integración de la información y el seguimiento eficiente.

- Trabajo colaborativo basado en retos:

La asignatura promueve metodologías activas como el aprendizaje colaborativo, basado en proyectos y en retos. Sin embargo, no se dispone de una plataforma digital que estructure y apoye esta colaboración de forma centralizada y accesible para todos los estudiantes y docentes.

- Monitoreo, retroalimentación y evaluación continua:

El docente realiza seguimiento del avance de los equipos, proporciona retroalimentación y evalúa tanto el proceso como los entregables del proyecto.

Esta tarea es limitada por la falta de herramientas de visualización y control en tiempo real.

- Simulación del ciclo de vida del proyecto:

Los estudiantes deben simular todas las etapas de un proyecto de ingeniería: inicio, planificación, ejecución, monitoreo y cierre, siguiendo enfoques tradicionales y ágiles. Este proceso se ve restringido por la ausencia de una herramienta específica que permita gestionar cada etapa de forma dinámica e interactiva.

#### **1.4. Oportunidad Encontrada**

El curso “Dirección de proyectos” tiene como objetivo que los estudiantes apliquen metodologías y herramientas de dirección de proyectos, simulando un entorno real de gestión. No obstante, actualmente existe una brecha entre los contenidos teóricos del curso y su implementación práctica mediante el uso de herramientas tecnológicas adecuadas.

Los estudiantes gestionan sus proyectos con plataformas genéricas que no están diseñadas específicamente para la educación en ingeniería ni para el seguimiento estructurado de las fases de un proyecto. Esto limita el aprovechamiento de metodologías como el enfoque ágil, dificulta la planificación colaborativa, y reduce la eficiencia en la asignación de tareas, seguimiento de avances y evaluación del desempeño grupal.

Frente a esta situación, se identifica una oportunidad clave: desarrollar una plataforma virtual especializada para la gestión académica de proyectos de ingeniería, que no solo automatice y centralice los procesos del curso, sino que también fortalezca el aprendizaje experiencial, colaborativo y orientado a retos, tal como propone la metodología del curso.

Además, esta plataforma puede incorporar tecnologías emergentes como la Inteligencia Artificial, que agregue valor a través de funcionalidades como asistentes virtuales para

guiar a los estudiantes, predicción de tiempos para mejorar la planificación y visualización de métricas en tiempo real para docentes.

Esta solución no solo atendería una necesidad puntual del curso, sino que abre la posibilidad de ser escalada a otras carreras y asignaturas similares, promoviendo una transformación digital en los procesos educativos de la Facultad de Ingeniería.

## **1.5. Detalles del Proyecto**

### **Nombre del Proyecto:**

Desarrollo de una Aplicación Web con Inteligencia Artificial para la Gestión Colaborativa de Proyectos Académicos

### **Tecnología Base:**

Stack MERN (MongoDB, Express.js, React.js, Node.js)

### **Funcionalidades Principales:**

- **Gestión de Proyectos:** Creación, edición y eliminación de proyectos. Asignación de tareas y responsables. Seguimiento de avances.
- **Tableros Kanban:** Visualización del flujo de tareas por etapas (por hacer, en progreso, terminado).
- **Colaboración en Tiempo Real:** Chat integrado para equipos de trabajo y notificaciones push.
- **Inteligencia Artificial:**
  - Asistente virtual para soporte y guía en el uso de la herramienta.
  - Algoritmo de predicción de tiempos de ejecución basado en historial de tareas.

- **Panel Administrativo:**

- Dashboard con métricas clave de rendimiento.
- Gestión de usuarios y control de accesos.

**Alcance Inicial:**

Aplicación orientada al curso de Gestión de Proyectos en Ingeniería del noveno ciclo de la EAP de Ingeniería Industrial.

**Beneficiarios Directos:**

Estudiantes, docentes y personal académico involucrado en el curso.

**Impacto Esperado:**

- Mejora de la organización y eficiencia en el desarrollo de proyectos.
- Aumento de la participación y colaboración entre estudiantes.
- Integración de tecnología emergente en el entorno académico.

## **CAPÍTULO 2**

### **ESTUDIO DE FACTIBILIDAD**

El presente capítulo analiza la viabilidad del proyecto de desarrollo de una Plataforma de Gestión de Proyectos Colaborativos con Integración de Inteligencia Artificial, considerando tres dimensiones clave: técnica, económica y operativa. Este análisis permite justificar la implementación del sistema propuesto y reducir el riesgo de fracaso durante su ejecución.

#### **2.1. Alternativas de Solución**

#### **2.2. Factibilidad Técnica**

La factibilidad técnica del proyecto se analiza a partir de la revisión de los recursos tecnológicos utilizados en el desarrollo de la plataforma de gestión de proyectos

colaborativos con integración de inteligencia artificial. Se consideraron aspectos como los frameworks, lenguajes, herramientas y arquitectura general, asegurando que el sistema propuesto pueda ser implementado de manera eficiente, sostenible y escalable.

### **2.2.1. Tecnologías Utilizadas**

El desarrollo de la plataforma se realizó utilizando tecnologías modernas y ampliamente adoptadas en la industria del software, lo cual garantiza su robustez, escalabilidad y mantenibilidad. A continuación, se describen los principales componentes:

- Stack MERN (MongoDB, Express.js, React, [Node.js](#)): Se utilizó esta arquitectura de desarrollo full-stack JavaScript para la construcción de toda la plataforma.
  - MongoDB: Base de datos NoSQL utilizada para almacenar la información de los proyectos, tareas, usuarios y demás componentes del sistema.
  - Express.js y Node.js: Plataforma de backend que permite construir APIs RESTful escalables y manejar la lógica del servidor.
  - React: Librería frontend usada para construir una interfaz de usuario dinámica, interactiva y modular.
- LangChain + API de ChatGPT (OpenAI): Framework que facilita la integración de modelos de lenguaje en aplicaciones. Permite estructurar flujos conversacionales complejos utilizando la API de OpenAI, ofreciendo funcionalidades como asistencia inteligente, generación de contenido y soporte automatizado.
- [Socket.IO](#): Implementado para habilitar la comunicación en tiempo real entre usuarios. Esto permite funciones colaborativas como edición simultánea, actualizaciones en vivo y notificaciones instantáneas dentro de la plataforma.

- Tailwind CSS: Framework de diseño usado para construir una interfaz moderna, adaptable y limpia, optimizando tiempos de desarrollo visual.
- Docker (uso en entorno de desarrollo): Utilizado durante la fase de desarrollo para asegurar entornos controlados, replicables y consistentes. No se utilizó para el despliegue en producción.

### **2.2.2. Infraestructura Requerida**

El sistema está diseñado para operar en la nube, lo que reduce la necesidad de infraestructura física propia. El backend, frontend, y los servicios de inteligencia artificial pueden ser desplegados en servicios como AWS, Vercel, Render o similares. MongoDB puede alojarse en MongoDB Atlas para mayor seguridad y escalabilidad.

La integración con la API de OpenAI para ChatGPT permite externalizar el cómputo intensivo, lo que disminuye los requerimientos locales del sistema.

### **2.2.3. Viabilidad de la Integración de la IA**

LangChain facilita la conexión con modelos de lenguaje avanzados mediante una arquitectura modular. Gracias a esto, la plataforma puede incorporar funciones de IA como:

- Generación automática de textos y tareas.
- Asistencia virtual en la planificación de proyectos.

El uso de la API de OpenAI garantiza acceso a modelos robustos sin la necesidad de infraestructura de entrenamiento local.

## **2.3. Factibilidad Económica**

La factibilidad económica del proyecto se basa en la evaluación de los costos necesarios para el desarrollo, implementación y operación de la plataforma de gestión de proyectos colaborativos con integración de inteligencia artificial, en comparación con

los beneficios que esta puede generar. El objetivo es determinar si el proyecto es rentable y sostenible a corto, mediano y largo plazo.

### **2.3.1. Presupuesto estimado**

El presupuesto total programado asciende a \$12,571, calculado en función de las horas hombre, materiales, costos de viaje y otros gastos operativos. Se consideraron todas las fases del proyecto: inicio, desarrollo, entregas, y gestión.

- Resumen de Costos por Fase del Proyecto:**

Fase del Proyecto	Horas Hombre	Costo Total (USD)
1. Inicio del Proyecto	88	\$2,272
2. Desarrollo del Proyecto	248	\$5,543
3. Entregas del Proyecto	116	\$2,472
9. Gerencia del Proyecto	105	\$2,284
Total General	557	\$12,571

- Distribución de Costos por Categoría**

- Costo por horas hombre: \$763
- Costo en materiales: \$240
- Costos de viaje: \$387
- Otros costos (diversos): \$1,181
- (incluye licencias, servicios, imprevistos operativos, etc.)

- Presupuesto general:**

## Fuentes de Costo del Proyecto

<b>Nombre del Proyecto:</b>	Plataforma De Gestión De Proyectos Colaborativ
<b>Gerente del Proyecto:</b>	YERALD CRISTHIAN SINCHE ALVARADO

**Instrucciones:**

- > Ingrese las tareas del proyecto específicas a su proyecto
- > Ingrese la información del presupuesto en las celdas en blanco
- > Los totales son calculados automáticamente
- > Para agregar filas, Desproteger hoja (Herramientas / Protección / Desproteger). Proteja cuando este hecho para proteger la entrada de datos.

Tarea del Proyecto		Horas hombre	Costo por Hora (\$)	Costo del Material (\$)	Costos de Viaje (\$)	OtrosCostos (\$)	Total por Tarea	
<b>1 Inicio del proyecto</b>								
1.1	Declaración de la visión del proyecto	4	\$20	\$2.4	\$0	\$2	\$85	
1.2	Arquitectura del Desarrollo del Sistema	16	\$20	\$11	\$0	\$10	\$341	
1.3	Desarrollar el preliminar de las Especificaciones de Diseño	16	\$20	\$11	\$0	\$10	\$341	
1.4	Desarrollar las Especificaciones Detalladas del Diseño	40	\$30	\$26	\$0	\$24	\$1,250	
1.5	Desarrollar el Plan de Pruebas de Aceptación	12	\$20	\$9	\$0	\$7	\$256	
	<b>Subtotal</b>	<b>88</b>		<b>\$59</b>	<b>\$0</b>	<b>\$53</b>	<b>\$2,272</b>	
<b>2 Desarrollo del Proyecto</b>								
2.1	Desarrollar Componentes	120	\$20	\$90	\$0	\$72	\$2,562	
2.2	Desarrollo de Sprint 0	0	\$0	\$0	\$10	\$15	\$25	
2.3	Desarrollo de Sprint 1	20	\$15	\$15	\$10	\$12	\$337	
2.4	Desarrollo de Sprint 2	10	\$15	\$10	\$10	\$40	\$210	
2.5	Desarrollo de Sprint 3	10	\$15	\$10	\$10	\$10	\$180	
2.6	Adquirir Software	8	\$20	\$360	\$0	\$5	\$525	
2.7	Adquirir Hardware	0	\$0	\$0	\$0	\$0	\$0	
2.8	Desarrollar el Paquete de Pruebas de Aceptación	20	\$20	\$20	\$0	\$12	\$432	
2.9	Ejecución de Pruebas Unitarias / Integración	60	\$20	\$36	\$0	\$36	\$1,272	
	<b>Subtotal</b>	<b>248</b>		<b>\$541</b>	<b>\$40</b>	<b>\$202</b>	<b>\$5,543</b>	
<b>3 Entregas del Proyecto</b>								
3.1	Instalar Sistema	16	\$20	\$11	\$0	\$10	\$341	
3.2	Entrenar clientes	24	\$10	\$20	\$50	\$14	\$324	
3.3	Pruebas de Aceptación del Desempeño	20	\$20	\$15	\$0	\$12	\$427	
3.4	Revisión del Desempeño Post Proyecto	12	\$30	\$8	\$50	\$7	\$425	
3.5	Proveer Garantía de Soporte	40	\$20	\$28	\$50	\$24	\$902	
3.6	Archivar Materiales	4	\$10	\$10	\$0	\$2	\$52	
	<b>Subtotal</b>	<b>116</b>		<b>\$92</b>	<b>\$150</b>	<b>\$70</b>	<b>\$2,472</b>	
<b>4 Gerencia del Proyecto</b>								
4.1	Reuniones/Reportes del Progreso con el Cliente	0	\$0	\$0	\$0	\$0	\$0	
4.2	Reuniones/Reportes Internas de Estatus del Proyecto	20	\$20	\$15	\$50	\$12	\$477	
4.3	Reuniones con terceros	0	\$0	\$0	\$0	\$0	\$0	
4.4	Interfaz a Otros Departamentos Internos	0	\$0	\$0	\$0	\$0	\$0	
4.5	Gestión de la Configuración	10	\$20	\$8	\$0	\$6	\$214	
4.6	Aseguramiento de la Calidad	25	\$20	\$16	\$0	\$15	\$531	
4.7	Gestión Global del Proyecto	50	\$20	\$32	\$0	\$30	\$1,062	
	<b>Subtotal</b>	<b>105</b>		<b>\$71</b>	<b>\$50</b>	<b>\$63</b>	<b>\$2,284</b>	
<b>10 - Otros</b>	Otros Costos	0	\$0	\$0	\$0	\$0	\$0	
<b>11 - Otros</b>	Otros Costos	0	\$0	\$0	\$0	\$0	\$0	
		<b>Sub-Totales:</b>	<b>557</b>		<b>\$763</b>	<b>\$240</b>	<b>\$387</b>	<b>\$12,571</b>
		<b>Riesgo (Contingencia):</b>	0	\$0	\$0	\$0	\$0	\$0
		<b>TOTAL (Programado):</b>	<b>557</b>		<b>\$763</b>	<b>\$240</b>	<b>\$387</b>	<b>\$12,571</b>
<b>Comentarios:</b>								

## **2.4. Factibilidad Operacional**

La factibilidad operacional del proyecto evalúa si la plataforma puede ser utilizada de manera efectiva por los usuarios finales dentro del contexto previsto. Se considera la viabilidad de implementación, la aceptación por parte de los usuarios, la disponibilidad de recursos humanos y la adaptabilidad del sistema al entorno organizacional.

### **2.4.1. Adaptabilidad del Sistema al Entorno Real**

La plataforma ha sido diseñada con un enfoque modular, escalable y centrado en el usuario. Esto permite que pueda ser fácilmente adaptada a diferentes contextos de trabajo colaborativo, como organizaciones educativas, equipos de desarrollo de software, departamentos administrativos o grupos de investigación.

Su arquitectura basada en el stack MERN y tecnologías web modernas permite su implementación en diferentes entornos operativos sin necesidad de infraestructura especializada, lo cual favorece su adopción.

### **2.4.2. Facilidad de Uso**

El sistema incorpora una interfaz de usuario desarrollada con Tailwind CSS, lo que permite una experiencia visual limpia, moderna y responsiva. Además, la inclusión de inteligencia artificial mediante LangChain + ChatGPT facilita tareas complejas y ofrece asistencia en tiempo real, reduciendo la curva de aprendizaje de los usuarios.

Características que mejoran la usabilidad:

- Paneles intuitivos para la gestión de proyectos y tareas.
- Notificaciones en tiempo real mediante [Socket.IO](#).
- Asistente virtual que responde consultas sobre el uso del sistema o ayuda en la planificación.

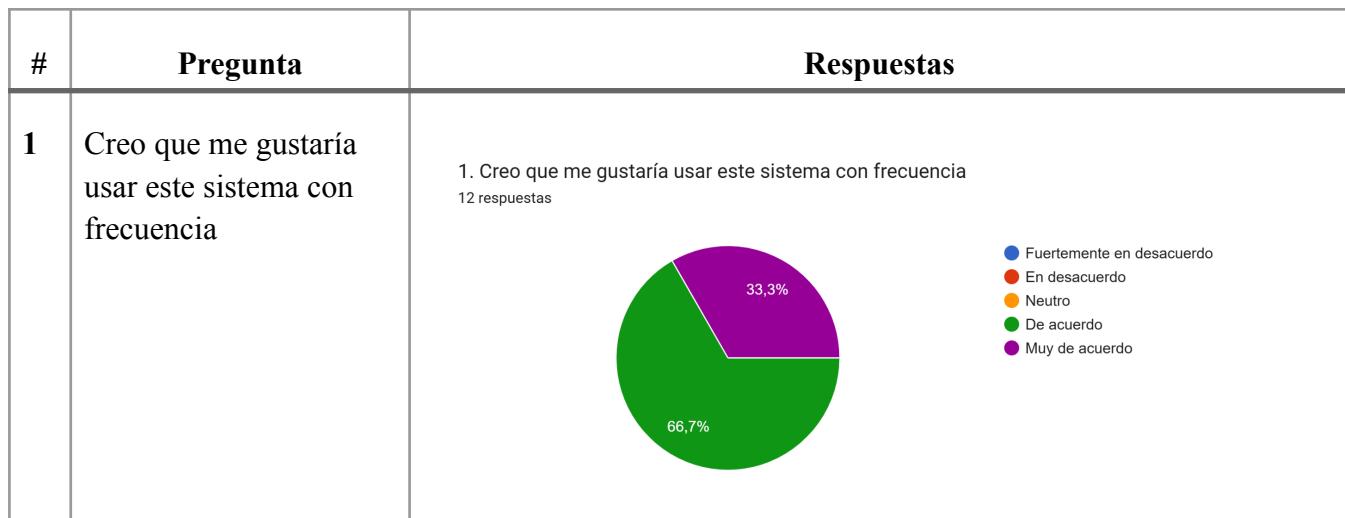
### **2.4.3. Aceptación por Parte de los Usuarios**

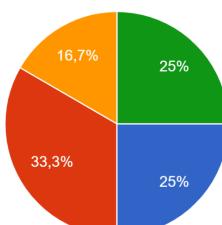
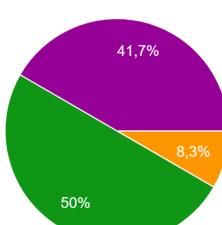
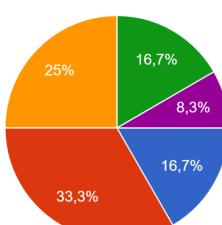
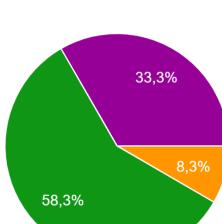
Se aplicó la Escala de Usabilidad del Sistema (SUS) a un grupo de 12 participantes con el objetivo de evaluar la percepción de usabilidad del sistema desarrollado. La

prueba arrojó un puntaje promedio de 73.75 sobre 100, lo cual indica un nivel de aceptabilidad positivo según los estándares establecidos por Brooke (1996) y posteriores interpretaciones de los resultados.

De acuerdo con los rangos de interpretación de la SUS, un puntaje por encima de 68 se considera superior al promedio y refleja que los usuarios encuentran el sistema fácil de usar, confiable y funcional en términos generales. El valor de 73.75% sugiere que la mayoría de los usuarios tuvieron una experiencia satisfactoria, lo que respalda la efectividad de las decisiones de diseño implementadas hasta el momento.

Este resultado también sugiere que, aunque el sistema se encuentra dentro del rango de aceptación, existe un margen para seguir mejorando la experiencia de usuario, especialmente si se busca alcanzar niveles de excelencia superiores (por ejemplo, puntajes de 80 o más, que se asocian con sistemas altamente usables).

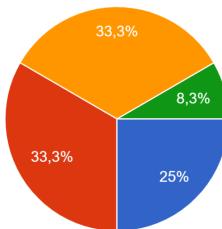


2	<p>Encontré el sistema innecesariamente complejo</p>	<p>2. Encontré el sistema innecesariamente complejo 12 respuestas</p>  <table border="1"> <thead> <tr> <th>Categoría</th> <th>Porcentaje</th> </tr> </thead> <tbody> <tr> <td>Fuertemente en desacuerdo</td> <td>25%</td> </tr> <tr> <td>En desacuerdo</td> <td>33.3%</td> </tr> <tr> <td>Neutro</td> <td>16.7%</td> </tr> <tr> <td>De acuerdo</td> <td>25%</td> </tr> <tr> <td>Muy de acuerdo</td> <td>16.7%</td> </tr> </tbody> </table> <ul style="list-style-type: none"> <li>● Fuertemente en desacuerdo</li> <li>● En desacuerdo</li> <li>● Neutro</li> <li>● De acuerdo</li> <li>● Muy de acuerdo</li> </ul>	Categoría	Porcentaje	Fuertemente en desacuerdo	25%	En desacuerdo	33.3%	Neutro	16.7%	De acuerdo	25%	Muy de acuerdo	16.7%
Categoría	Porcentaje													
Fuertemente en desacuerdo	25%													
En desacuerdo	33.3%													
Neutro	16.7%													
De acuerdo	25%													
Muy de acuerdo	16.7%													
3	<p>Pienso que el sistema era fácil de usar</p>	<p>3. Pienso que el sistema era fácil de usar 12 respuestas</p>  <table border="1"> <thead> <tr> <th>Categoría</th> <th>Porcentaje</th> </tr> </thead> <tbody> <tr> <td>Fuertemente en desacuerdo</td> <td>8.3%</td> </tr> <tr> <td>En desacuerdo</td> <td>0%</td> </tr> <tr> <td>Neutro</td> <td>8.3%</td> </tr> <tr> <td>De acuerdo</td> <td>50%</td> </tr> <tr> <td>Muy de acuerdo</td> <td>41.7%</td> </tr> </tbody> </table> <ul style="list-style-type: none"> <li>● Fuertemente en desacuerdo</li> <li>● En desacuerdo</li> <li>● Neutro</li> <li>● De acuerdo</li> <li>● Muy de acuerdo</li> </ul>	Categoría	Porcentaje	Fuertemente en desacuerdo	8.3%	En desacuerdo	0%	Neutro	8.3%	De acuerdo	50%	Muy de acuerdo	41.7%
Categoría	Porcentaje													
Fuertemente en desacuerdo	8.3%													
En desacuerdo	0%													
Neutro	8.3%													
De acuerdo	50%													
Muy de acuerdo	41.7%													
4	<p>Creo que necesitaría la asistencia de un técnico para poder usar este sistema</p>	<p>4. Creo que necesitaría la asistencia de un técnico para poder usar este sistema 12 respuestas</p>  <table border="1"> <thead> <tr> <th>Categoría</th> <th>Porcentaje</th> </tr> </thead> <tbody> <tr> <td>Fuertemente en desacuerdo</td> <td>16.7%</td> </tr> <tr> <td>En desacuerdo</td> <td>33.3%</td> </tr> <tr> <td>Neutro</td> <td>25%</td> </tr> <tr> <td>De acuerdo</td> <td>8.3%</td> </tr> <tr> <td>Muy de acuerdo</td> <td>16.7%</td> </tr> </tbody> </table> <ul style="list-style-type: none"> <li>● Fuertemente en desacuerdo</li> <li>● En desacuerdo</li> <li>● Neutro</li> <li>● De acuerdo</li> <li>● Muy de acuerdo</li> </ul>	Categoría	Porcentaje	Fuertemente en desacuerdo	16.7%	En desacuerdo	33.3%	Neutro	25%	De acuerdo	8.3%	Muy de acuerdo	16.7%
Categoría	Porcentaje													
Fuertemente en desacuerdo	16.7%													
En desacuerdo	33.3%													
Neutro	25%													
De acuerdo	8.3%													
Muy de acuerdo	16.7%													
5	<p>Encontré que las diversas funciones de este sistema estaban bien integradas</p>	<p>5. Encontré que las diversas funciones de este sistema estaban bien integradas 12 respuestas</p>  <table border="1"> <thead> <tr> <th>Categoría</th> <th>Porcentaje</th> </tr> </thead> <tbody> <tr> <td>Fuertemente en desacuerdo</td> <td>0%</td> </tr> <tr> <td>En desacuerdo</td> <td>0%</td> </tr> <tr> <td>Neutro</td> <td>8.3%</td> </tr> <tr> <td>De acuerdo</td> <td>58.3%</td> </tr> <tr> <td>Muy de acuerdo</td> <td>33.3%</td> </tr> </tbody> </table> <ul style="list-style-type: none"> <li>● Fuertemente en desacuerdo</li> <li>● En desacuerdo</li> <li>● Neutro</li> <li>● De acuerdo</li> <li>● Muy de acuerdo</li> </ul>	Categoría	Porcentaje	Fuertemente en desacuerdo	0%	En desacuerdo	0%	Neutro	8.3%	De acuerdo	58.3%	Muy de acuerdo	33.3%
Categoría	Porcentaje													
Fuertemente en desacuerdo	0%													
En desacuerdo	0%													
Neutro	8.3%													
De acuerdo	58.3%													
Muy de acuerdo	33.3%													

6	<p>Pienso que había demasiada inconsistencia en este sistema</p>	<p>6. Pienso que había demasiada inconsistencia en este sistema 12 respuestas</p> <table border="1"> <thead> <tr> <th>Opción</th> <th>Porcentaje</th> </tr> </thead> <tbody> <tr> <td>Fuertemente en desacuerdo</td> <td>8,3%</td> </tr> <tr> <td>En desacuerdo</td> <td>41,7%</td> </tr> <tr> <td>Neutro</td> <td>33,3%</td> </tr> <tr> <td>De acuerdo</td> <td>8,3%</td> </tr> <tr> <td>Muy de acuerdo</td> <td>16,7%</td> </tr> </tbody> </table>	Opción	Porcentaje	Fuertemente en desacuerdo	8,3%	En desacuerdo	41,7%	Neutro	33,3%	De acuerdo	8,3%	Muy de acuerdo	16,7%
Opción	Porcentaje													
Fuertemente en desacuerdo	8,3%													
En desacuerdo	41,7%													
Neutro	33,3%													
De acuerdo	8,3%													
Muy de acuerdo	16,7%													
7	<p>La mayoría de la gente aprenderá a usar este sistema muy rápidamente</p>	<p>7. La mayoría de la gente aprenderá a usar este sistema muy rápidamente 12 respuestas</p> <table border="1"> <thead> <tr> <th>Opción</th> <th>Porcentaje</th> </tr> </thead> <tbody> <tr> <td>Fuertemente en desacuerdo</td> <td>0%</td> </tr> <tr> <td>En desacuerdo</td> <td>8,3%</td> </tr> <tr> <td>Neutro</td> <td>0%</td> </tr> <tr> <td>De acuerdo</td> <td>50%</td> </tr> <tr> <td>Muy de acuerdo</td> <td>41,7%</td> </tr> </tbody> </table>	Opción	Porcentaje	Fuertemente en desacuerdo	0%	En desacuerdo	8,3%	Neutro	0%	De acuerdo	50%	Muy de acuerdo	41,7%
Opción	Porcentaje													
Fuertemente en desacuerdo	0%													
En desacuerdo	8,3%													
Neutro	0%													
De acuerdo	50%													
Muy de acuerdo	41,7%													
8	<p>Me pareció que el sistema era muy incómodo de usar</p>	<p>8. Me pareció que el sistema era muy incómodo de usar 12 respuestas</p> <table border="1"> <thead> <tr> <th>Opción</th> <th>Porcentaje</th> </tr> </thead> <tbody> <tr> <td>Fuertemente en desacuerdo</td> <td>16,7%</td> </tr> <tr> <td>En desacuerdo</td> <td>50%</td> </tr> <tr> <td>Neutro</td> <td>8,3%</td> </tr> <tr> <td>De acuerdo</td> <td>33,3%</td> </tr> </tbody> </table>	Opción	Porcentaje	Fuertemente en desacuerdo	16,7%	En desacuerdo	50%	Neutro	8,3%	De acuerdo	33,3%		
Opción	Porcentaje													
Fuertemente en desacuerdo	16,7%													
En desacuerdo	50%													
Neutro	8,3%													
De acuerdo	33,3%													
9	<p>Me sentí muy confiado usando el sistema</p>	<p>9. Me sentí muy confiado usando el sistema 12 respuestas</p> <table border="1"> <thead> <tr> <th>Opción</th> <th>Porcentaje</th> </tr> </thead> <tbody> <tr> <td>Fuertemente en desacuerdo</td> <td>0%</td> </tr> <tr> <td>En desacuerdo</td> <td>8,3%</td> </tr> <tr> <td>Neutro</td> <td>8,3%</td> </tr> <tr> <td>De acuerdo</td> <td>58,3%</td> </tr> <tr> <td>Muy de acuerdo</td> <td>25%</td> </tr> </tbody> </table>	Opción	Porcentaje	Fuertemente en desacuerdo	0%	En desacuerdo	8,3%	Neutro	8,3%	De acuerdo	58,3%	Muy de acuerdo	25%
Opción	Porcentaje													
Fuertemente en desacuerdo	0%													
En desacuerdo	8,3%													
Neutro	8,3%													
De acuerdo	58,3%													
Muy de acuerdo	25%													

**10** Necesito aprender muchas cosas antes de poder ponerme en marcha con este sistema

10. Necesito aprender muchas cosas antes de poder ponerme en marcha con este sistema  
12 respuestas



- Fuertemente en desacuerdo
- En desacuerdo
- Neutro
- De acuerdo
- Muy de acuerdo

## CAPÍTULO 3

### ANÁLISIS DE REQUERIMIENTOS

#### 3.1. Metas del Sistema de Información

- **Optimizar la gestión de proyectos colaborativos**

Facilitar la creación, seguimiento y organización de proyectos y tareas en un entorno centralizado y accesible para todos los miembros del equipo.

- **Mejorar la productividad del equipo**

Permitir la asignación eficiente de tareas con fechas límite y prioridades claras, ayudando a los usuarios a enfocarse en lo importante.

- **Fomentar la comunicación efectiva en tiempo real**

Integrar herramientas de chat y notificaciones para mantener una coordinación fluida entre los participantes del proyecto.

- **Aprovechar la inteligencia artificial para apoyar la toma de decisiones**

Implementar funcionalidades basadas en IA, como asistentes virtuales y predicción de tiempos, para reducir la carga operativa y anticipar posibles cuellos de botella.

- **Proporcionar control y supervisión centralizada al administrador**

Ofrecer un panel de control con estadísticas visuales que faciliten el monitoreo del progreso de los proyectos y permitan la gestión eficiente de usuarios y permisos.

- **Mejorar la visibilidad y organización del trabajo**

Ofrecer visualizaciones tipo Kanban y métricas del rendimiento para que los equipos puedan evaluar su avance y ajustar su planificación.

- **Permitir la escalabilidad y adaptabilidad de la plataforma**

Diseñar el sistema de forma modular y flexible, de modo que pueda adaptarse a diferentes tipos de equipos y proyectos en crecimiento.

### **3.2. Requisitos del Sistema**

Para comenzar a desarrollar el proyecto, se procede a establecer y comunicar los requisitos que el negocio necesita en realidad, basándose en las metas que se desea alcanzar, de esta forma se tendrá un significado claro, tanto para el cliente como para los miembros del equipo de desarrollo. Los requerimientos son listados a continuación:

#### **3.2.1. Requerimientos funcionales**

##### **1. Gestión de Proyectos**

- RF1. El sistema debe permitir a los usuarios crear nuevos proyectos con nombre, descripción y fecha de inicio.
- RF2. El sistema debe permitir la edición de los datos de un proyecto existente.
- RF3. El sistema debe permitir eliminar un proyecto registrado.
- RF4. El sistema debe permitir la creación de tareas asociadas a un proyecto.
- RF5. El sistema debe permitir la asignación de tareas a miembros del equipo.
- RF6. El sistema debe permitir establecer fechas límite y niveles de prioridad para cada tarea.
- RF7. El sistema debe mostrar un tablero Kanban con las tareas organizadas por estado (por ejemplo: Pendiente, En Proceso, Finalizado).

##### **2. Comunicación en Tiempo Real**

- RF8. El sistema debe incluir un chat integrado para que los miembros del equipo se comuniquen en tiempo real.

- RF9. El sistema debe enviar notificaciones push a los usuarios cuando haya actualizaciones relevantes (nueva tarea asignada, cambio de estado, mensajes, etc.).

### **3. Integración de Inteligencia Artificial**

- RF10. El sistema debe incluir un asistente virtual basado en IA que permita a los usuarios consultar el estado de sus tareas, recibir recordatorios y recomendaciones.
- RF11. El sistema debe predecir la duración estimada de una tarea utilizando datos históricos similares (tiempo promedio, complejidad, usuario asignado).
- RF12. El sistema debe mostrar sugerencias automáticas de reasignación de tareas si se prevén retrasos según los modelos de IA.

### **4. Panel de Administración**

- RF13. El sistema debe permitir al administrador ver un dashboard con estadísticas visuales del estado y progreso de los proyectos.
- RF14. El sistema debe mostrar métricas como: número de tareas completadas, tareas atrasadas, tiempo promedio por tarea, participación por usuario.
- RF15. El sistema debe permitir al administrador gestionar (crear, editar, eliminar) usuarios del sistema.
- RF16. El sistema debe permitir asignar roles a los usuarios (Administrador, Miembro).
- RF17. El sistema debe permitir configurar permisos de acceso según el rol del usuario.

### **3.2.2. Requerimientos no funcionales**

#### **1. Seguridad**

- **RNF1.** El sistema debe requerir autenticación de usuario mediante correo electrónico y contraseña segura.
- **RNF2.** Las contraseñas deben almacenarse cifradas

#### **2. Rendimiento**

- **RNF3.** El sistema debe responder a las solicitudes del usuario en menos de 3 segundos bajo condiciones normales de uso.

#### **3. Usabilidad**

- **RNF4.** La interfaz debe ser amigable, intuitiva y adaptable a diferentes dispositivos (diseño responsive).

#### **4. Escalabilidad**

- **RNF5.** El sistema debe permitir agregar nuevas funcionalidades en el futuro sin necesidad de rediseñar completamente.

#### **5. Disponibilidad**

- **RNF6.** El sistema debe estar disponible al menos el 99% del tiempo mensual para los usuarios.

### **3.3. Identificación de Actores del Sistema**

Un actor es cualquier entidad que tenga comportamientos, incluyendo el propio sistema que se está desarrollando; los actores no son solo roles que juegan las personas, sino también organizaciones, software y equipos.

A través del análisis de los requerimientos, se definió que los actores del sistema serán:

#### **3.3.1. Administrador**

Con este perfil podemos ingresar, modificar y eliminar datos que se encuentran dentro de la aplicación. Este actor tiene la facultad de crear cuentas para nuevos usuarios, también puede generar reportes, gestionar la información de los proyectos.

#### **3.3.2. Miembro**

Este perfil cuenta con acceso a la información clave del software. Con este perfil podemos ingresar, modificar y eliminar datos relacionados al proceso de gestión de proyectos.

## CAPÍTULO 4

### PLANIFICACIÓN DEL PROYECTO

#### 4.1. Definición de Roles de Trabajo

ROL	RESPONSABLE
PRODUCT OWNER	DIRECTOR DE LA EAP DE INGENIERIA INDUSTRIAL
SCRUM MASTER	MUNIVE, GUERRA JOSÉ ALEJANDRO
TEAM MEMBER	CORILLA, JUSCAMATIA SAID MARDUX
	FERRUZO, IZQUIERDO JOCABED ISABEL
	MUNIVE, GUERRA JOSÉ ALEJANDRO
	ORTEGA, BATALLA BRAULIO CÉSAR
	SEGURA, MEZA GIOVANNY LUIS EDUARDO
	SINCHE, ALVARADO YERALD CRISTHIAN
TESTER	SEGURA, MEZA GIOVANNY LUIS EDUARDO
	CORILLA, JUSCAMATIA SAID MARDUX

#### 4.1.1. Product owner

Es el Gerente del proyecto; el señor Sinche Alvarado Yerald Cristhian quien está encargado de este rol, debido a que representa al equipo de desarrollo y quien recibe los requerimientos de los stakeholders.

#### **4.1.2. Scrum master**

El estudiante Munive Guerra José Alejandro es el encargado de este rol, debido a que cuenta con una mayor experiencia en el campo del desarrollo de aplicativos con la introducción de metodologías ágiles.

#### **4.1.3. Team member**

Los encargados de cumplir este rol son los estudiantes Corilla Juscamatia Said Mardux, Ferruzo Izquierdo Jocabed Isabel, Munive Guerra José Alejandro, Ortega Batalla Braulio César, Segura Meza Giovanny Luis Eduardo, Sinche Alvarado Yerald Cristhian; debido a que cuentan con el conocimiento necesario en las diversas áreas necesarias para el desarrollo de aplicaciones de escritorio, como se muestra en la Tabla ()�.

NOMBRE	ROL
CORILLA, JUSCAMATIA SAID MARDUX	Diseñador Frontend/UX
FERRUZO, IZQUIERDO JOCABED ISABEL	Desarrollador Frontend (React.js)
MUNIVE, GUERRA JOSÉ ALEJANDRO	Backend/DevOps
ORTEGA, BATALLA BRAULIO CÉSAR	Desarrollador Fullstack (MERN)
SEGURA, MEZA GIOVANNY LUIS EDUARDO	Backend/DB Admin (MongoDB)
SINCHE, ALVARADO YERALD CRISTHIAN	Desarrollador Backend (Node.js)

Fuente: Elaboración propia

#### **4.1.4. Tester**

Esta tarea será llevada a cabo por los estudiantes Segura Meza Giovanny Luis Eduardo, Corilla Juscamatia Said Mardux, esto debido a que cuentan con el conocimiento necesario para el desarrollo de pruebas de software.

## 4.2. Product Backlog

ID	Épica	Historia de Usuario	Criterios de Aceptación
E1	Gestión de Proyectos	Como miembro del equipo, quiero crear y editar proyectos para organizar nuestro trabajo.	El usuario puede crear un proyecto con nombre, descripción y miembros asignados.
	Gestión de Proyectos	Como miembro del equipo, quiero asignar tareas con prioridad y fecha límite para organizar mejor el trabajo.	Las tareas deben tener campos de prioridad y fecha límite.
	Gestión de Proyectos	Como miembro del equipo, quiero visualizar las tareas en un tablero Kanban para ver el estado general del proyecto.	Se deben mostrar columnas tipo: 'Por hacer', 'En progreso', 'Hecho'.
E2	Comunicación en tiempo real	Como miembro del equipo, quiero tener un chat integrado para comunicarme en tiempo real con mi equipo.	El sistema debe actualizar mensajes en tiempo real sin recargar.
	Comunicación en tiempo real	Como usuario, quiero recibir notificaciones push cuando se me asigna una tarea o se actualiza un proyecto.	Las notificaciones deben ser visibles inmediatamente.
	Comunicación en tiempo real	Como usuario, quiero mencionar a otros usuarios en los comentarios de tareas para mantenerlos informados.	La mención debe notificar al usuario mencionado.
E3	Funcionalidades de IA	Como usuario, quiero usar un asistente virtual ayude a los usuarios a gestionar sus tareas.	El chat integrado debe funcionar en tiempo real y las notificaciones push deben llegar a los usuarios sin retrasos
	Funcionalidades de IA	Como usuario, quiero que el sistema prediga la duración estimada de las tareas para mejorar la planificación basándose en datos históricos.	El asistente virtual debe proporcionar respuestas útiles y la predicción de tiempos debe ser precisa.
E4	Panel de Administración	Como administrador, quiero gestionar usuarios y sus permisos para controlar el acceso a las funcionalidades.	Los roles deben incluir permisos diferenciados por acción.
	Panel de Administración	Como administrador, quiero ver estadísticas visuales del proyecto para monitorear el avance.	El panel debe incluir gráficos de progreso, tareas completadas, etc.
	Panel de Administración	Como administrador, quiero exportar reportes del rendimiento del equipo.	Reportes en PDF o CSV.

	Infraestructura y Calidad	Como desarrollador, quiero tener el backend y frontend contenerizados con Docker para facilitar el despliegue.	La app debe correr localmente con `docker-compose up`.
E5	Infraestructura y Calidad	Como desarrollador, quiero contar con pruebas automatizadas para validar el comportamiento del sistema.	Pruebas unitarias y E2E ejecutables con un solo comando.
	Infraestructura y Calidad	Como desarrollador, quiero una documentación clara para facilitar la colaboración y mantenimiento del proyecto.	README.md con pasos de instalación, despliegue y uso.

### 4.3. Sprint Backlog

#### 4.3.1. Sprint 1

Historias de Usuario	Épica	Estado	Puntos de Historia	Encargado
CRUD de Proyectos	Gestión de Proyectos	Finalizada	6	Yerald Sinche
Asignación de tareas con fechas límite y prioridades	Gestión de Proyectos	Finalizada	10	Yerald Sinche
Visualización de tablero Kanban	Gestión de Proyectos	Finalizada	5	Yerald Sinche

#### 4.3.2. Sprint 2

Historias de Usuario	Épica	Estado	Puntos de Historia	Encargado
Mención de otros usuarios	Comunicación en Tiempo Real	Finalizada	3	Jose Munive
Recibir Notificaciones	Comunicación en Tiempo Real	Finalizada	3	Jose Munive
Chat integrado	Comunicación en Tiempo Real	Finalizada	3	Jose Munive

#### 4.3.3. Sprint 3

Historias de Usuario	Épica	Estado	Puntos de Historia	Encargado
Dashboard de Estadísticas Visuales	Panel de Administración	En curso	20	Giovanny Segura
Gestión de Usuarios	Panel de Administración	Finalizada	16	Giovanny Segura
Gestión de Permisos	Panel de Administración	Finalizada	16	Giovanny Segura
Visualización de Métricas de Rendimiento del Proyecto	Panel de Administración	En curso	12	Giovanny Segura

#### 4.3.4. Sprint 4

Historias de Usuario	Épica	Estado	Puntos de Historia	Encargado
IA - Asistente Virtual para Gestión de Tareas	Integración de IA	En Curso	6	Braulio Ortega
IA - Predicción de Tiempos Basada en Historial de Tareas	Integración de IA	En Curso	10	Braulio Ortega

### 4.4. Planificación de Sprints

#### 4.4.1. Historias de usuario

Historia de Usuario
Como miembro del equipo, quiero crear y editar proyectos para organizar nuestro trabajo.
Como miembro del equipo, quiero asignar tareas con prioridad y fecha límite para organizar mejor el trabajo.
Como miembro del equipo, quiero visualizar las tareas en un tablero Kanban para ver el estado general del proyecto.
Como miembro del equipo, quiero tener un chat integrado para comunicarme en tiempo real con mi equipo.

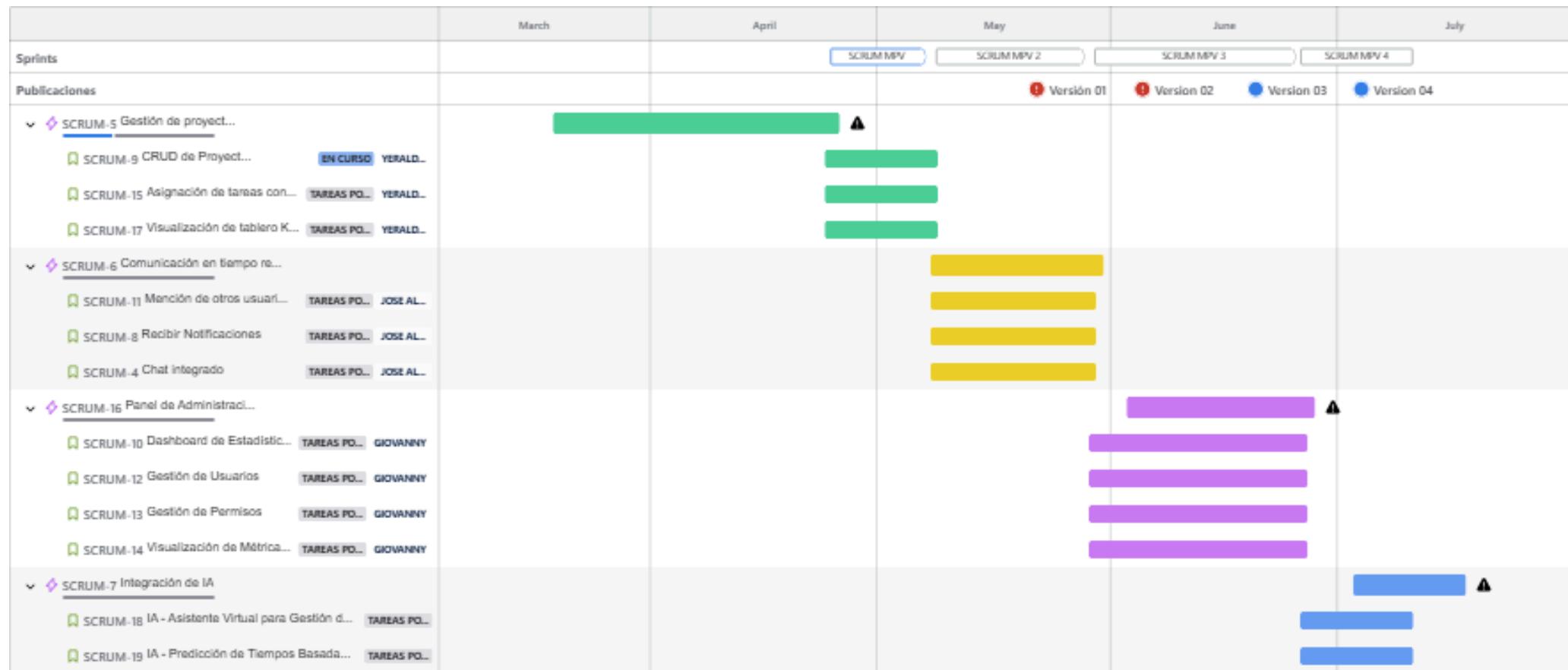
Como usuario, quiero recibir notificaciones push cuando se me asigna una tarea o se actualiza un proyecto.
Como usuario, quiero mencionar a otros usuarios en los comentarios de tareas para mantenerlos informados.
Como usuario, quiero usar un asistente virtual ayude a los usuarios a gestionar sus tareas.
Como usuario, quiero que el sistema prediga la duración estimada de las tareas para mejorar la planificación basándose en datos históricos.
Como administrador, quiero gestionar usuarios y sus permisos para controlar el acceso a las funcionalidades.
Como administrador, quiero ver estadísticas visuales del proyecto para monitorear el avance.
Como administrador, quiero exportar reportes del rendimiento del equipo.
Como desarrollador, quiero tener el backend y frontend contenerizados con Docker para facilitar el despliegue.
Como desarrollador, quiero contar con pruebas automatizadas para validar el comportamiento del sistema.
Como desarrollador, quiero una documentación clara para facilitar la colaboración y mantenimiento del proyecto.

#### 4.4.2. Priorización de historias de usuario

Historia de Usuario	Prioridad (MoSCoW)
Como miembro del equipo, quiero crear y editar proyectos para organizar nuestro trabajo.	M
Como miembro del equipo, quiero asignar tareas con prioridad y fecha límite para organizar mejor el trabajo.	M
Como miembro del equipo, quiero visualizar las tareas en un tablero Kanban para ver el estado general del proyecto.	M
Como miembro del equipo, quiero tener un chat integrado para comunicarme en tiempo real con mi equipo.	M
Como usuario, quiero recibir notificaciones push cuando se me asigna una tarea o se actualiza un proyecto.	M
Como usuario, quiero mencionar a otros usuarios en los comentarios de tareas para mantenerlos informados.	S
Como usuario, quiero usar un asistente virtual ayude a los usuarios a gestionar sus tareas.	M
Como usuario, quiero que el sistema prediga la duración estimada de las tareas para mejorar la planificación basándose en datos históricos.	S
Como administrador, quiero gestionar usuarios y sus permisos para controlar el acceso a las funcionalidades.	M
Como administrador, quiero ver estadísticas visuales del proyecto para monitorear el avance.	S

Como administrador, quiero exportar reportes del rendimiento del equipo.	C
Como desarrollador, quiero tener el backend y frontend contenerizados con Docker para facilitar el despliegue.	M
Como desarrollador, quiero contar con pruebas automatizadas para validar el comportamiento del sistema.	S
Como desarrollador, quiero una documentación clara para facilitar la colaboración y mantenimiento del proyecto.	M

#### **4.5. Cronograma de Actividades**



#### 4.6. Gestión de Riesgos

Registro de Riesgos														
Nombre del Proyecto:			Plataforma De Gestión De Proyectos Colaborativos Con Integración De Inteligencia Artificial											
ID Riesgo	Descripción del Riesgo	Área de Impacto	Causa	Impacto	Probabilidad	Puntuación de Riesgo	Detectabilidad	Estado	Asignado a	Evento que lo Dispara	ID Riesgo	Estrategia de Respuesta	Reserva de Continencia ; Ajustes al Presupuesto y Cronograma en el Plan de Gestión del Proyecto	Fecha de Aprobación
250424a	Nuevas tecnologías presentan un desafío a	Cronograma	El equipo no tiene experiencia en dicha	Critico	Muy Alto	8,1	Bajo	Activo	CORILLA, JUSCAMA TIA SAID MARDUX	El equipo no tenía entrenamiento previo	250424a	Se llevará a cabo entrenamiento para 3 miembros	300	24-abr-25

	IT con la curva de aprendizaje, lo que puede causar atrasos en implementaciones	tecnología							s del equipo en Julio 2015
250424b	Un integrante no está disponible durante un periodo de tiempo	Cronograma	No se evaluó adecuadamente a los miembros del equipo	Muy Serio	Muy Alto	6,3	Alto	Observado	FERRUZO, IZQUIERDO JOCABED ISABEL Problemas de salud y cruces de horarios de trabajo
250424c	Actualizaciones y mantenimiento: La rápida	Alcance	La tecnología de IA evolución	Serio	Alto	3,5	Bajo	Observado	MUNIVE, GUERRA JOSÉ ALEJANDRO Lanzamiento de nuevas versiones de

250424d	A medida que avanza el	Alcance	Requisitos mal definidos,	Serio	Muy Alto	4,5	Alto	Observador	ORTEGA, BATALLA	Reuniones con el cliente	250424d

evolución de tecnologías de IA puede requerir actualizaciones constantes para mantener la relevancia y seguridad del sistema.

rápidamente, lo que obliga a actualizar frecuentemente algoritmos, modelos y compatibilidades para evitar obsolescencia y vulnerabilidades.



herramientas, modelos de IA o cambios en requisitos de seguridad y compatibilidad tecnológica.

actualizaciones proactivas, que incluya monitoreo continuo de tecnologías emergentes, pruebas en entornos controlados antes del despliegue y asignación de un equipo dedicado a mantenimiento y mejora continua.

Documentar todos los tiempos en el

	proyecto, el cliente o los usuarios finales pueden cambiar de opinión o tener nuevas necesidades.	evolución de necesidades del cliente, poca comunicación con stakeholders.			BRAULIO CÉSAR	donde se plantean nuevas ideas, feedback del usuario final o cambios en el negocio.		requerimientos y establecer un proceso formal para gestionar solicitudes de cambio; prever un margen presupuestario para reanálisis o rediseño; establecer revisiones regulares de alcance para minimizar impacto.					
250424e	Retraso en el cronograma que puede deberse a estimaciones incorrectas	Cronograma	Estimaciones incorrectas, bloqueos técnicos, falta de planificación	Muy Serio	Muy Alto	6,3	Bajo	Observad o	SEGURA, MEZA, GIOVANNI Y LUIS EDUARDO	Fallas en la entrega de módulos, bloqueos técnicos inesperados o	Dividir el proyecto en tareas pequeñas y hacer seguimiento frecuente	250424e	24-abr-25

250424f	as, bloqueos técnicos, depende ncias externas o falta de recursos.  La falta de claridad entre el equipo de desarroll o, los usuarios, el cliente o el project manager puede llevar a malenten didos o errores graves.	ión realista.	Cronogra ma	Canales mal definidos, lenguaje técnico complejo, reunione s ineficient es.	Serio	Alto	3,5	Alto	Observad o



SINCHE,  
ALVARAD  
O YERALD  
CRISTHIA  
N

Falta de reuniones regulares, cambios no informados, o decisiones que no se comparten con todo el equipo.

errores que requieren retrabajo.

del avance.

de horas hombre o días adicional es; prever recursos adicionales para mitigar cuellos de botella si aparecen.

Invertir en herramientas de comunicación colaborativa (Slack, Notion, etc.); incluir tiempo para reuniones regulares y revisión de acuerdos; poco

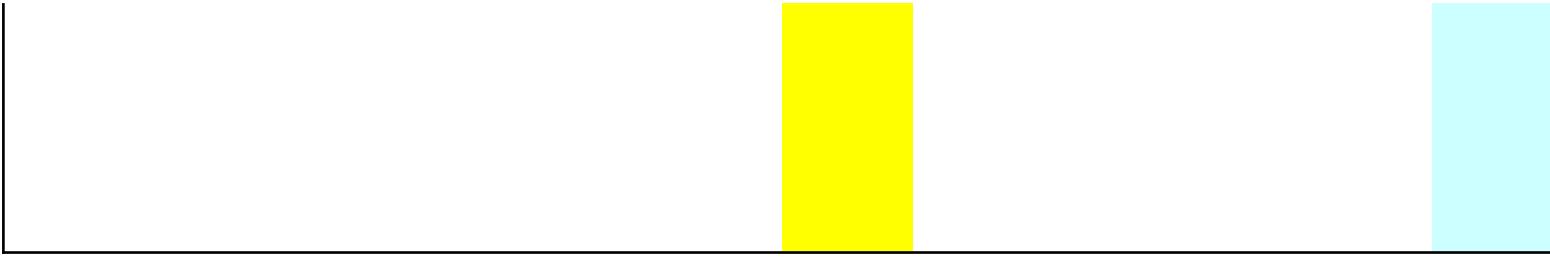
250424f

24-abr-25

250424g	Código mal escrito, sin pruebas suficiente s o sin buenas prácticas puede generar bugs difíciles de detectar.	Calidad	Presión por tiempos, ausencia de pruebas automati zadas, malas prácticas de desarroll o.	Critico	Alto	6,3	Bajo	Observad o	CORILLA, JUSCAMA TIA SAID MARDUX	<p>Entregas apuradas, saltarse la revisión de código o eliminaci ón de etapas de testing por presión de tiempo.</p> <p>Aplicar revisión de código y pruebas automatizadas como práctica obligatori a.</p> <p>ajuste presupuestario, pero sí de gestión del tiempo. Reservar tiempo para testing y revisiones de código en cada fase; considerar costos asociados a herramientas de QA o soporte externo; no comprimir fases críticas para mantener estándar</p>

250424h	Dependencia de tecnologías o herramientas externas ya que muchos proyectos dependen de APIs, librerías, servicios de terceros o licencias.	Calidad	Uso de servicios/ librerías sin respaldo, herramientas no mantenidas, sin análisis de riesgo técnico.	Moderado	Medio	1,5	Alto	Observador	FERRUZO, IZQUIERDO JOCABED ISABEL	Fallo en una API de terceros, cambios en licencias, interrupción de servicios externos críticos.
250424i	Por presión de tiempo o descuido, a veces no se	Calidad	Cronograma ajustado, recursos limitados para QA, subestim	Serio	Alto	3,5	Bajo	Observador	CORILLA, JUSCAMA TIA SAID MARDUX	Aceleración del cronograma, cambio en prioridad

	realizan pruebas suficientes (unitarias, , integración, usabilidad).	acción de la importancia del testing.				es, o ausencia de QA al final del proyecto.		pruebas antes de cada entrega.	asignar recursos técnicos y tiempo exclusivo para testing; contemplar posible necesidad de soporte externo para pruebas finales.		
250424j	El sistema puede funcionar bien en desarrollo o con pocos usuarios, pero colapsar al escalar.	Calidad	Arquitectura mal planificada, sin pruebas de carga, falta de monitoreo desde el inicio.	Serio	Medio	2,5	FERRUZO, IZQUIERDO JOCABED ISABEL	Aumento repentino de usuarios, despliegue en producción sin pruebas de carga, o lanzamiento apresurado.	Realizar pruebas de carga en el cronograma antes del lanzamiento y monitorear el sistema desde staging.	250424j	24-abr-25



necesaria  
s; dejar  
margen  
técnico  
en los  
últimos  
sprints.

## CAPÍTULO 5

### DISEÑO DEL SISTEMA DE INFORMACIÓN

#### 5.1. Diseño de Diagramas UML

##### 5.1.1. Diagrama de casos de uso

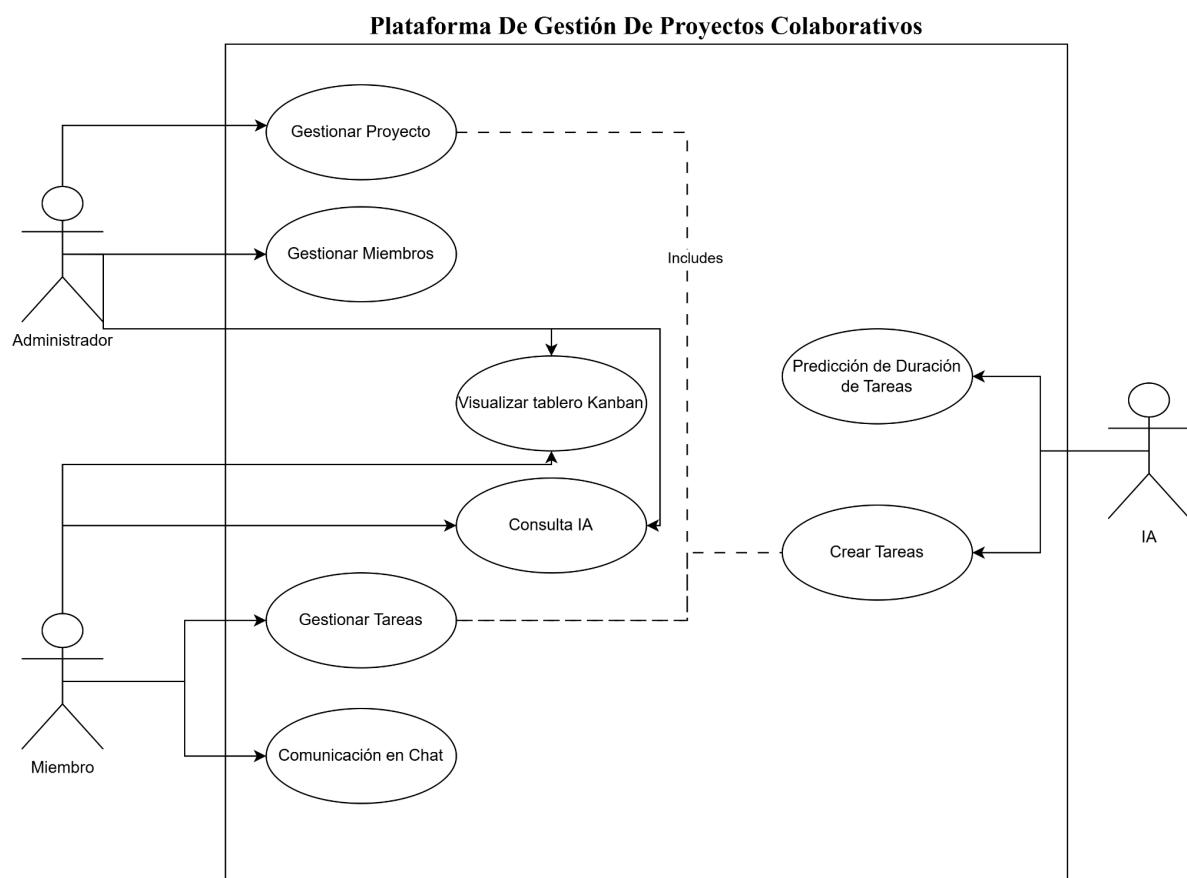


Diagrama de Casos de Uso

### 5.1.2. Diagramas de secuencia

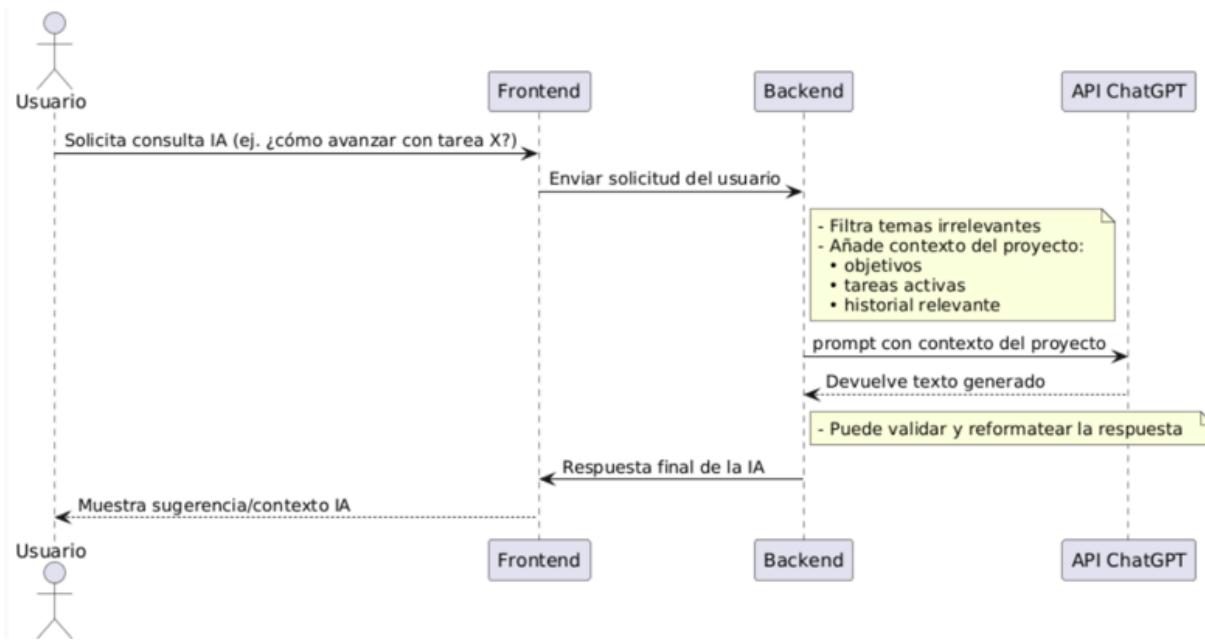


Diagrama de Secuencia - Consulta IA

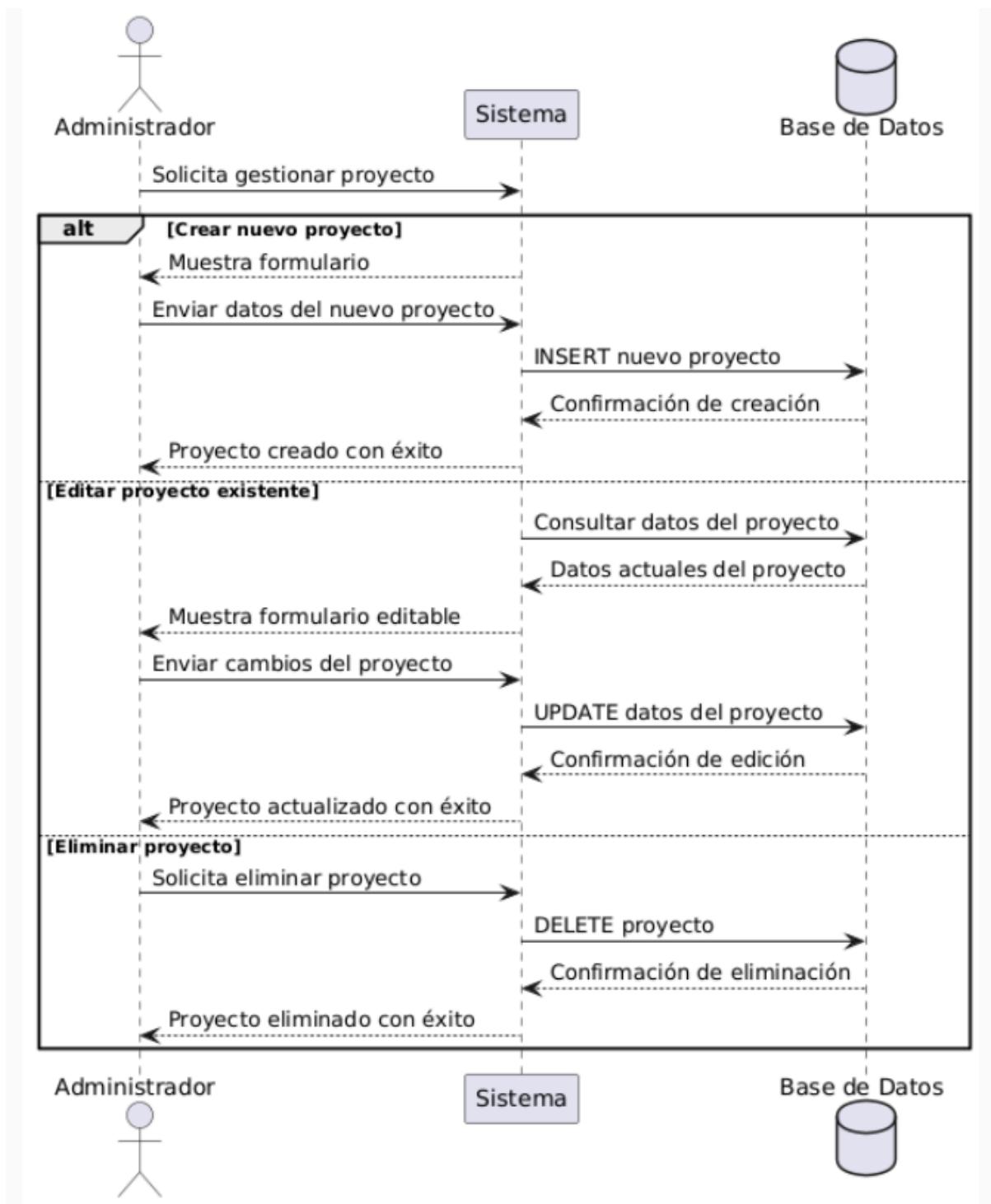
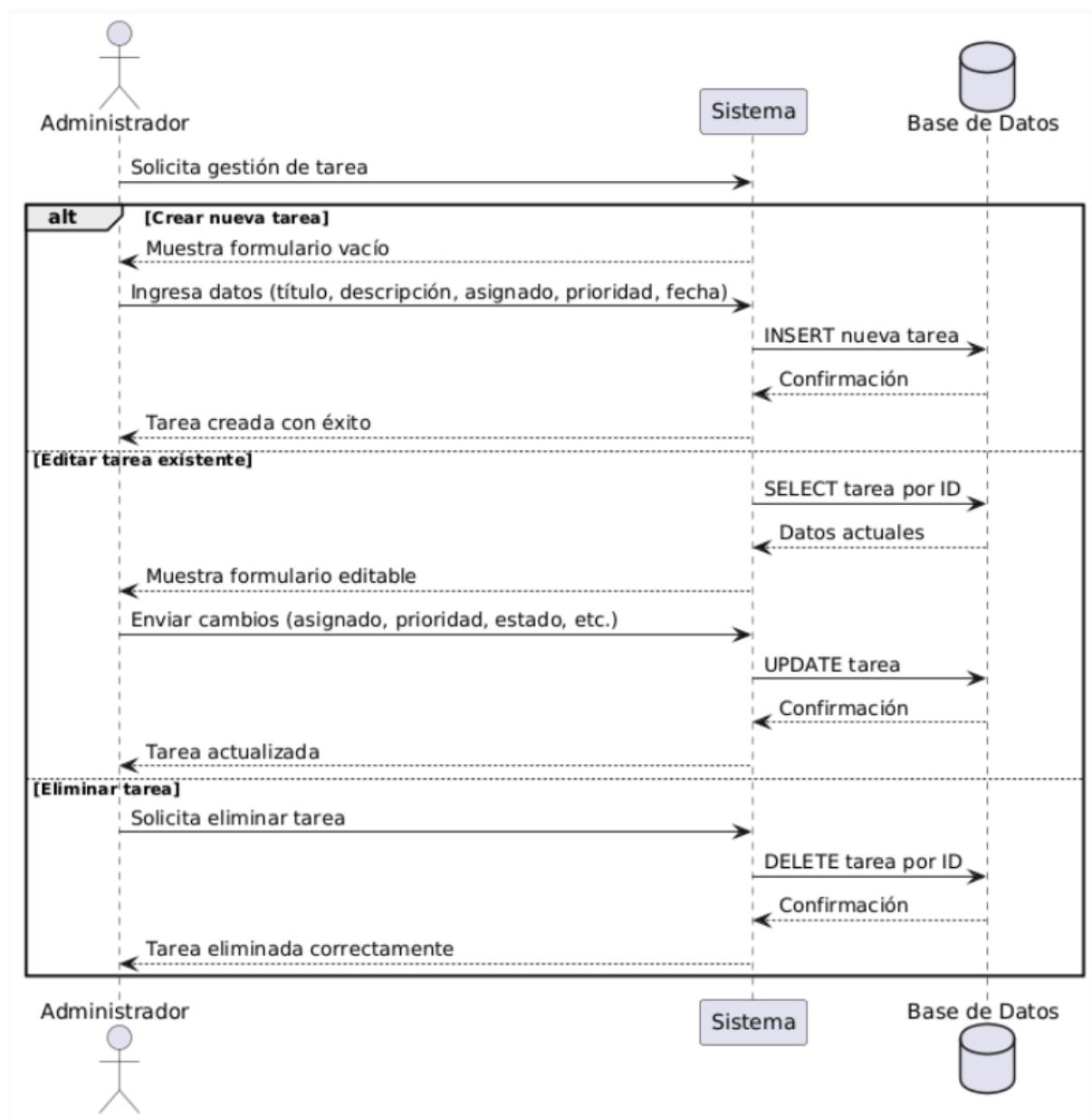


Diagrama de secuencia Gestión de Proyectos



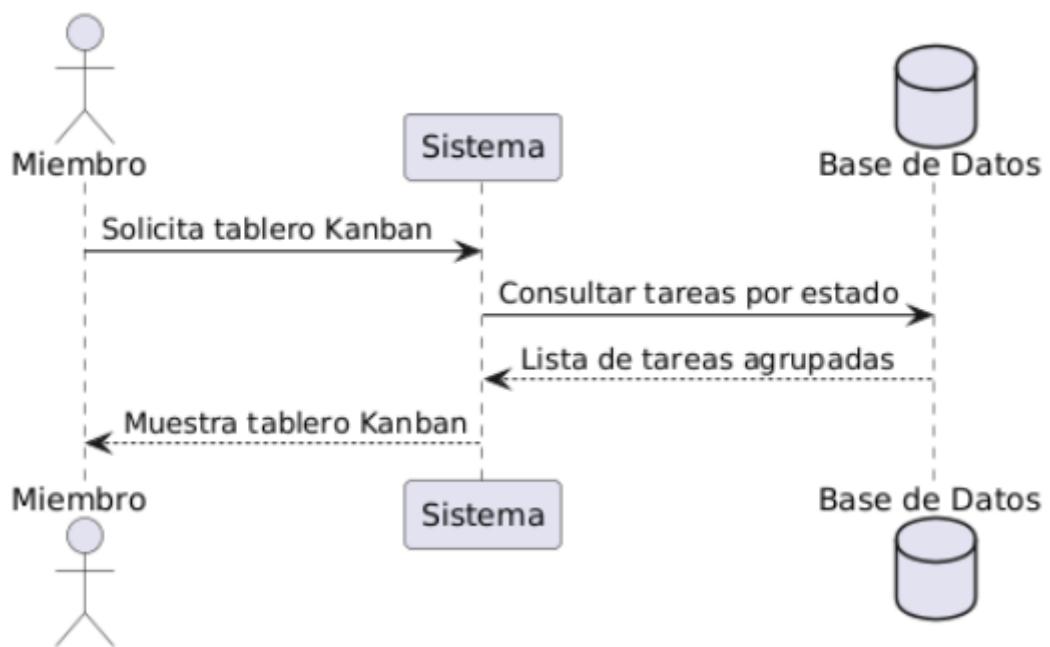


Diagrama de Secuencia Visualizar Tablero Kanban

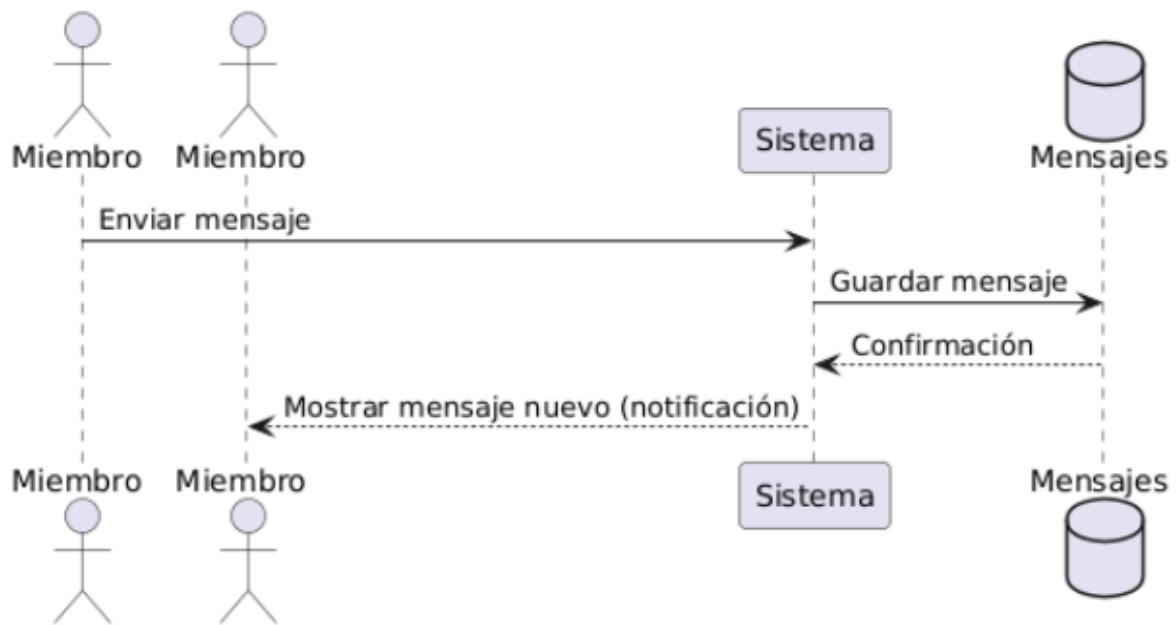


Diagrama de Secuencia Comunicación en Chat

### 5.1.3. Diagramas de colaboración

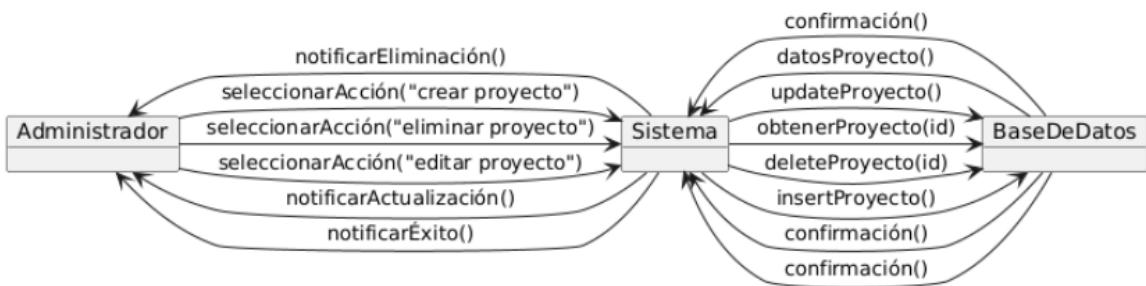


Diagrama de Colaboración - Gestión de Proyectos

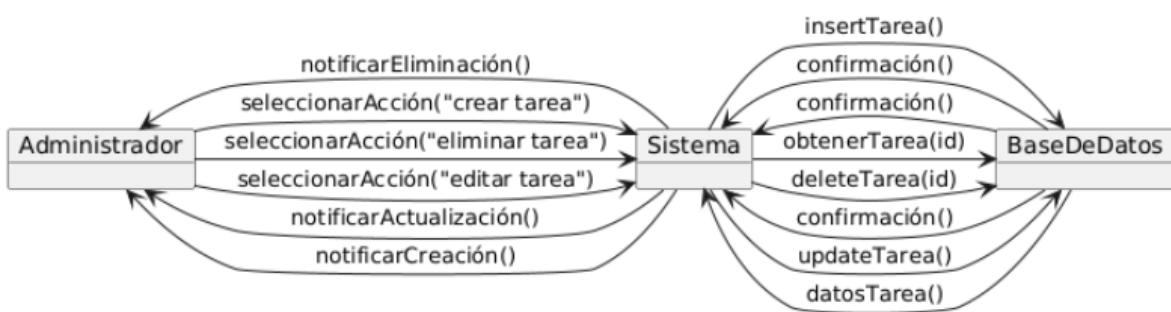


Diagrama de Colaboración - Gestión de Tareas

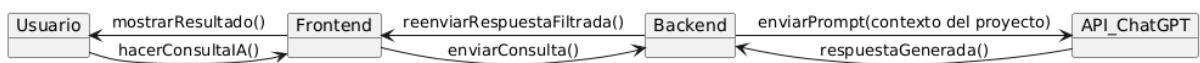


Diagrama de Colaboración - Consulta IA

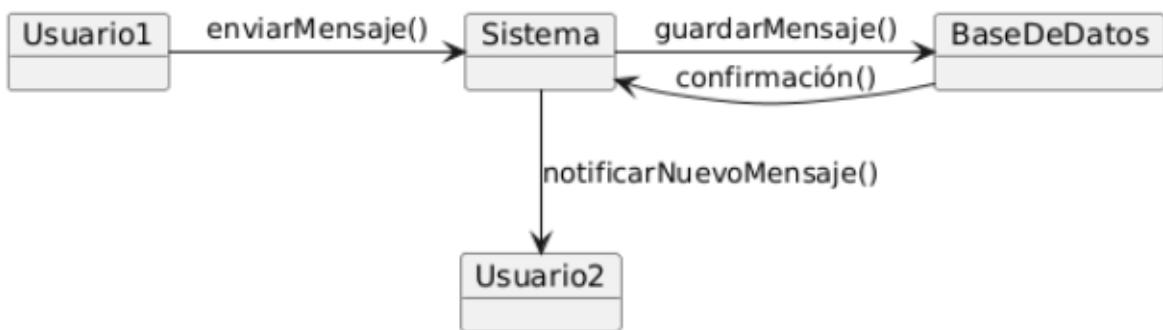


Diagrama de Colaboración - Chat



Diagrama de Colaboración - Tablero Kanban

#### 5.1.4. Diagrama de clases

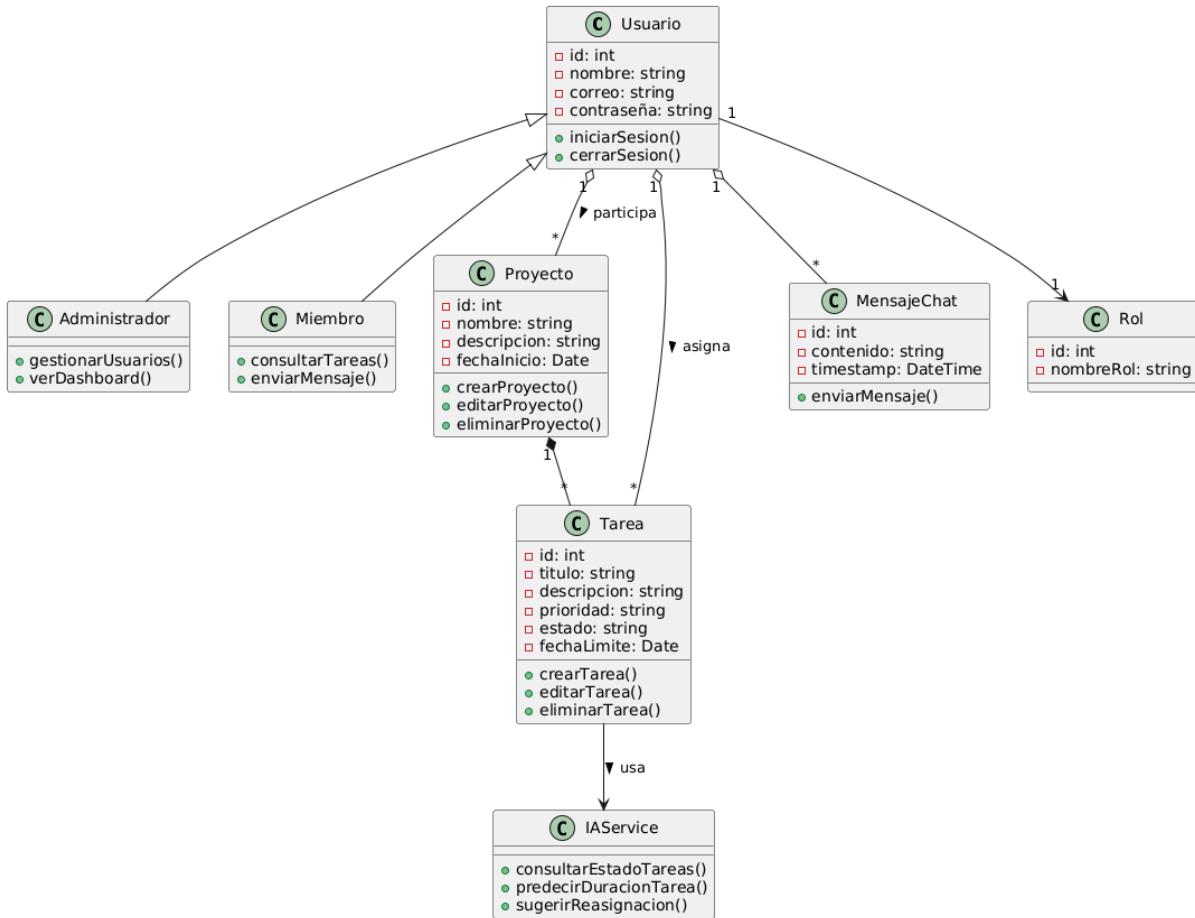


Diagrama de Clases

### 5.2. Diseño de Base de Datos

#### 5.2.1. Diseño conceptual (E/R)

Para estructurar la base de datos, usamos un modelo Entidad-Relación adaptado a sistemas NoSQL. En esta fase, nos enfocamos en definir las partes clave del sistema (entidades), sus características (atributos) y cómo se conectan entre sí, sin entrar en detalles técnicos de programación.

### **Entidades principales:**

- Usuario (User): Guarda la información personal y credenciales de cada usuario.
- Proyecto (Project): Espacio colaborativo creado por un usuario, donde otros pueden unirse.
- Lista (List): Agrupa tarjetas dentro de un proyecto para organizar tareas por categorías.
- Tarjeta (Card): Representa una tarea específica, con asignados, checklist, archivos adjuntos y etiquetas.
- Mensaje (Message): Facilita la comunicación entre usuarios dentro de un proyecto.
- Invitación (Invitation): Gestiona las solicitudes para unirse a un proyecto, incluyendo quién invita y el estado de la invitación.

### **Relaciones clave:**

- Un usuario puede crear varios proyectos y ser miembro de otros.
- Cada proyecto contiene múltiples listas, y cada lista agrupa varias tarjetas.
- Una tarjeta puede tener varios usuarios asignados.
- Los mensajes pertenecen a un proyecto y son enviados por un usuario.
- Una invitación vincula al usuario que invita, al invitado y al proyecto correspondiente.

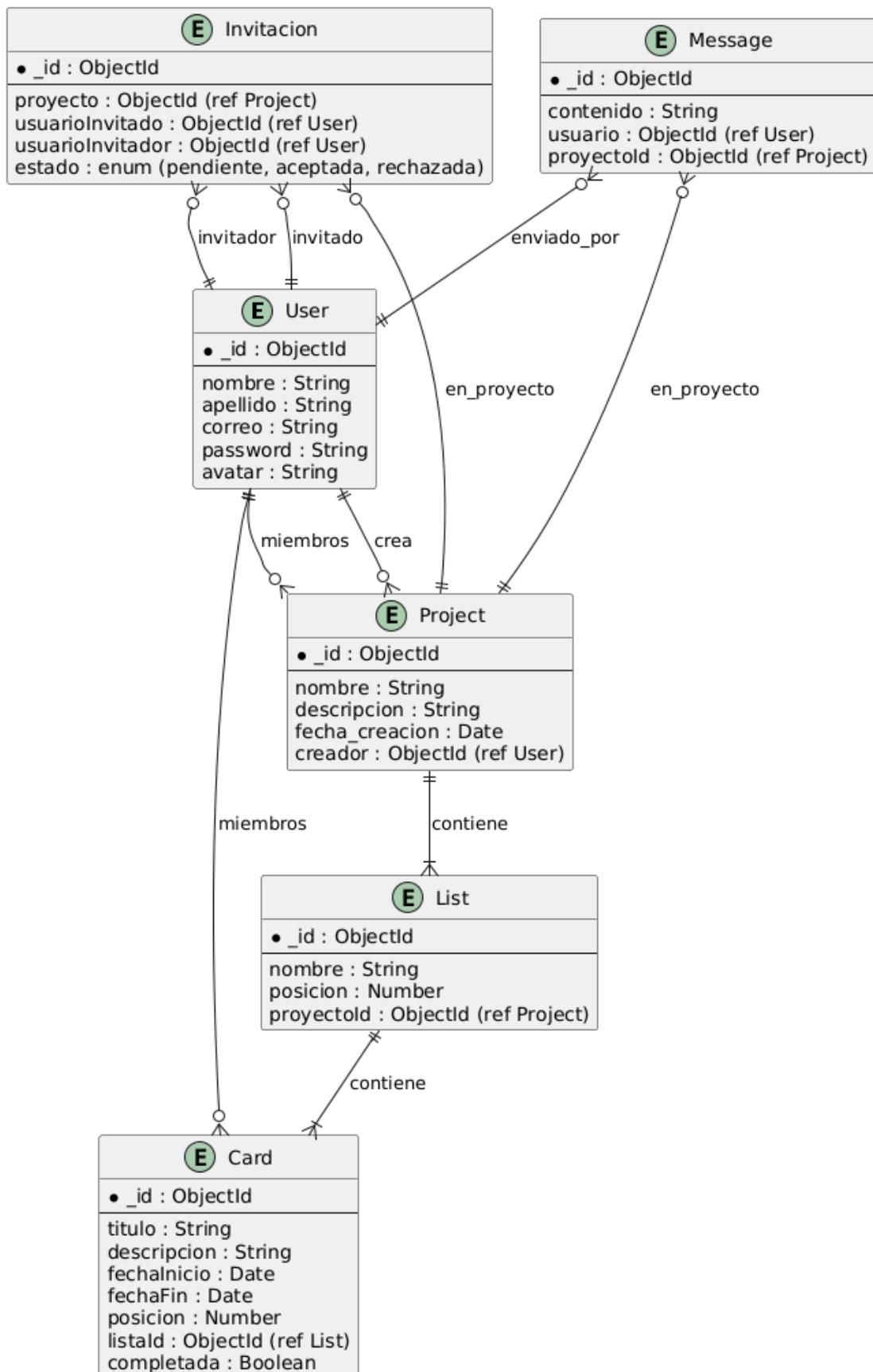


Diagrama de Entidad Relación

### 5.2.2. Diseño lógico

En el modelo lógico, estructuramos los datos para MongoDB usando un enfoque híbrido: combinamos referencias (ObjectId) para entidades independientes (como usuarios o proyectos) con subdocumentos embebidos para datos anidados que dependen directamente de un documento principal (como listas dentro de un proyecto o checklists en una tarjeta). Esto optimiza las consultas manteniendo la coherencia de la información.

Ejemplos de los modelos en JSON:

```
1
2 { 
3   "_id": "64f7a1e2b1a5f12345678901",
4   "nombre": "Ana",
5   "apellido": "Lopez",
6   "correo": "ana@example.com",
7   "password": "$2a$10$*****",
8   "avatar": "https://i.imgur.com/ana.png",
9   "createdAt": "2025-06-01T12:00:00Z",
10  "updatedAt": "2025-06-10T14:30:00Z"
11 }
12 
```

Modelo-User

```
{
  "_id": "64f7a200b1a5f12345678902",
  "nombre": "Sistema de Gestión",
  "descripcion": "Proyecto principal para organizar tareas",
  "fecha_creacion": "2025-06-01T12:00:00Z",
  "creador": "64f7a1e2b1a5f12345678901",
  "miembros": [
    { "usuario": "64f7a1e2b1a5f12345678901", "rol": "propietario" },
    { "usuario": "64f7a1f9b1a5f12345678904", "rol": "colaborador" }
  ],
  "createdAt": "2025-06-01T12:00:00Z",
  "updatedAt": "2025-06-10T14:30:00Z"
}
```

Modelo-Project

```
{  
    "_id": "64f7a220b1a5f12345678903",  
    "nombre": "Por hacer",  
    "posicion": 1,  
    "proyectoId": "64f7a200b1a5f12345678902",  
    "createdAt": "2025-06-01T12:15:00Z",  
    "updatedAt": "2025-06-01T12:15:00Z"  
}
```

### Modelo-List

```
{  
    "_id": "64f7a250b1a5f12345678905",  
    "titulo": "Diseñar la pantalla de login",  
    "descripcion": "Debe incluir validación de campos y diseño responsive",  
    "fechaInicio": "2025-06-02T00:00:00Z",  
    "fechaFin": "2025-06-05T00:00:00Z",  
    "posicion": 1,  
    "listaId": "64f7a220b1a5f12345678903",  
    "miembros": ["64f7a1e2b1a5f12345678901"],  
    "etiquetas": [  
        { "color": "rojo", "nombre": "Urgente" },  
        { "color": "azul", "nombre": "UI/UX" }  
    ],  
    "checklist": [  
        { "nombre": "Crear formulario", "completado": true },  
        { "nombre": "Agregar validación", "completado": false }  
    ],  
    "adjuntos": [  
        { "nombre": "login_mockup.png", "url": "https://i.imgur.com/mockup.png" }  
    ],  
    "completada": false,  
    "createdAt": "2025-06-02T10:00:00Z",  
    "updatedAt": "2025-06-04T16:00:00Z"  
}
```

### Modelo-Card

```
✓ {  
    "_id": "64f7a280b1a5f12345678906",  
    "contenido": "¿Puedo asumir esta tarea?",  
    "usuario": "64f7a1f9b1a5f12345678904",  
    "proyectoId": "64f7a200b1a5f12345678902",  
    "createdAt": "2025-06-03T09:30:00Z",  
    "updatedAt": "2025-06-03T09:30:00Z"  
}
```

### Modelo-Message

### 5.2.3. Diseño físico

El diseño físico de la base de datos fue implementado utilizando MongoDB como motor de base de datos NoSQL orientado a documentos, y Mongoose como biblioteca ODM (Object Data Modeling) para definir los esquemas en la aplicación Node.js. En este nivel, se tradujo el modelo lógico a estructuras de almacenamiento reales mediante definiciones de Schema que representan las colecciones, los tipos de datos, las relaciones y las restricciones.

#### Colecciones y estructuras físicas:

- **Usuario (User):** Define los datos personales y credenciales de cada usuario. Se incluyen validaciones como required, trim, lowercase y unique para el correo electrónico. También se usa timestamps para registrar automáticamente la fecha de creación y modificación del documento.

```
const userSchema = new mongoose.Schema({  
  nombre: { type: String, required: true, trim: true },  
  apellido: { type: String, trim: true },  
  correo: { type: String, required: true, unique: true, lowercase: true, trim: true },  
  password: { type: String, required: true, minlength: 6 },  
  avatar: { type: String }  
, { timestamps: true });
```

- **Proyecto (Project):** Representa un proyecto colaborativo. Se enlaza con un usuario creador mediante una referencia (ref: 'User') y contiene un arreglo embebido de miembros con roles diferenciados (propietario, colaborador o lector).

```

const proyectoSchema = new mongoose.Schema({
  nombre: { type: String, required: true, trim: true },
  descripcion: { type: String, trim: true },
  fecha_creacion: { type: Date, default: Date.now },
  creador: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required: true },
  miembros: [
    {
      usuario: { type: mongoose.Schema.Types.ObjectId, ref: 'User' },
      rol: { type: String, enum: ['propietario', 'colaborador', 'lector'], default: 'colaborador' }
    }
  ], { timestamps: true });

```

- Lista (List): Define listas de tareas dentro de un proyecto, permitiendo ordenar su posición.

```

const listSchema = new mongoose.Schema({
  nombre: { type: String, required: true, trim: true },
  posicion: { type: Number, default: 0 },
  proyectoId: { type: mongoose.Schema.Types.ObjectId, ref: 'Project', required: true }
}, { timestamps: true });

```

- Tarjeta (Card): Cada tarjeta representa una tarea y puede contener subtareas (checklist), etiquetas, adjuntos y usuarios asignados. Utiliza tanto referencias (ref) como documentos embebidos para mejorar el rendimiento en lectura.

```

const cardSchema = new mongoose.Schema({
  titulo: { type: String, required: true, trim: true },
  descripcion: { type: String, trim: true },
  fechaInicio: Date,
  fechaFin: Date,
  posicion: { type: Number, default: 0 },
  listaId: { type: mongoose.Schema.Types.ObjectId, ref: 'List', required: true },
  miembros: [{ type: mongoose.Schema.Types.ObjectId, ref: 'User' }],
  etiquetas: [{ color: String, nombre: String }],
  checklist: [{ nombre: String, completado: { type: Boolean, default: false } }],
  adjuntos: [{ nombre: String, url: String }],
  completada: { type: Boolean, default: false }
}, { timestamps: true });

```

- Mensaje (Message): Almacena mensajes enviados por usuarios dentro de un proyecto. Incluye referencias a User y Project.

```

const messageSchema = new mongoose.Schema({
  contenido: { type: String, required: true },
  usuario: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required: true },
  proyectoId: { type: mongoose.Schema.Types.ObjectId, ref: 'Project', required: true }
}, { timestamps: true });

```

- Invitación (Invitation): Gestiona el envío y aceptación de invitaciones entre usuarios y proyectos.

```

const invitacionSchema = new mongoose.Schema({
  proyecto: { type: mongoose.Schema.Types.ObjectId, ref: 'Project', required: true },
  usuarioInvitado: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required: true },
  usuarioInvitador: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required: true },
  estado: { type: String, enum: ['pendiente', 'aceptada', 'rechazada'], default: 'pendiente' },
  fechaCreacion: { type: Date, default: Date.now }
});

```

#### 5.2.4. Modelado de base de datos

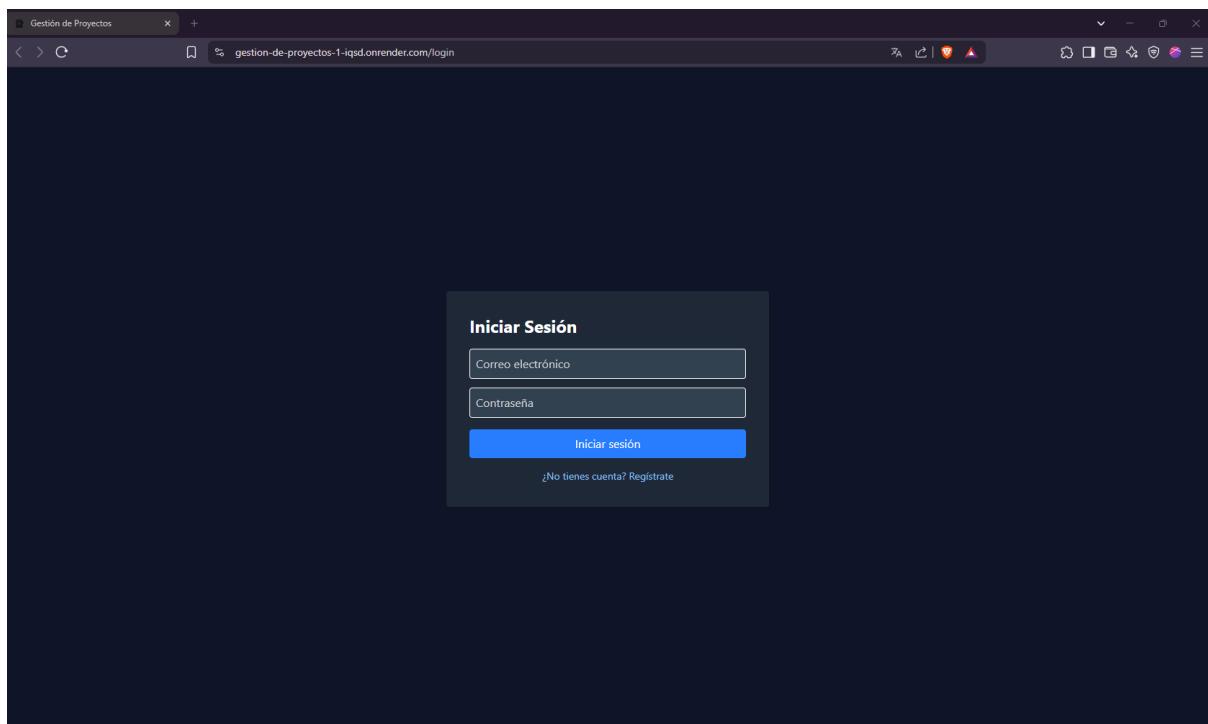
El modelado de la base de datos se realizó utilizando MongoDB como motor NoSQL y Mongoose como ODM para definir los esquemas. Se representaron las entidades del sistema (usuarios, proyectos, listas, tarjetas, mensajes, invitaciones) mediante colecciones y documentos BSON.

**Se utilizaron:**

- Referencias (ref) para modelar relaciones entre entidades (por ejemplo, un proyecto tiene un creador y miembros).
- Subdocumentos para estructuras internas como checklist, etiquetas y adjuntos dentro de una tarjeta.
- Validaciones (required, enum, unique, default, etc.) para asegurar integridad desde la capa lógica.
- Timestamps automáticos (createdAt y updatedAt) para registrar fechas de creación y modificación.

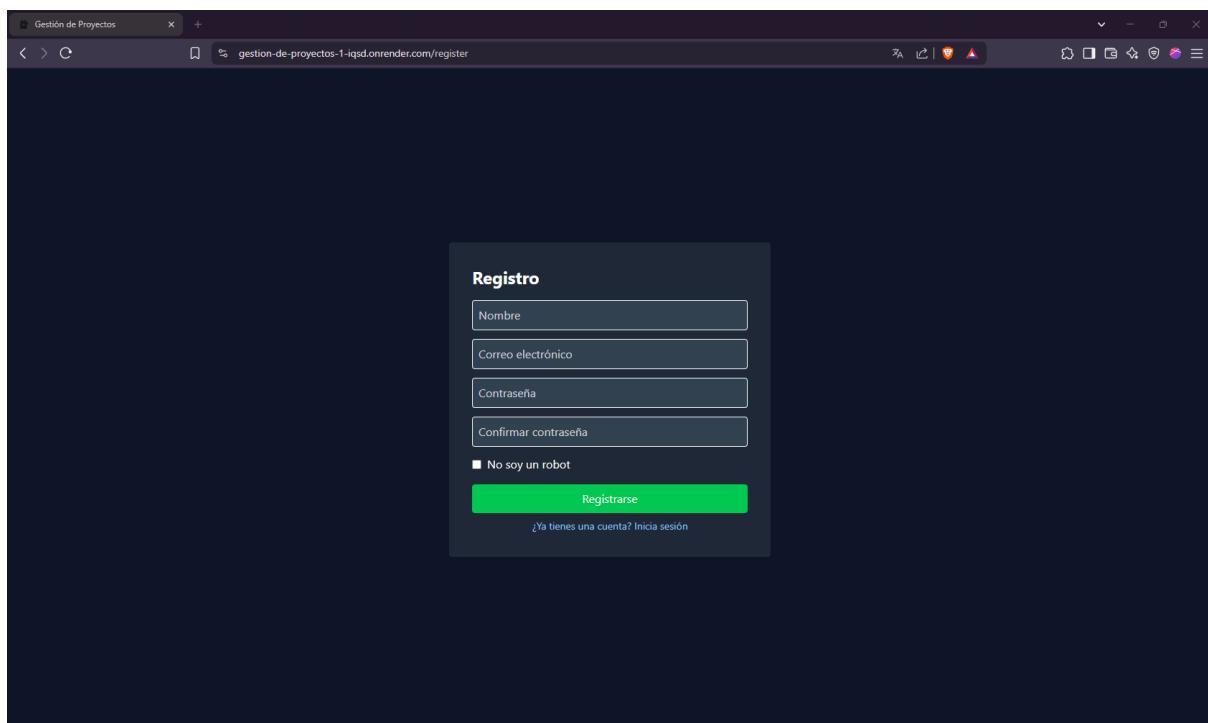
## 5.3. Diseño de Interfaces Básicas

### 5.3.1. Acceso login

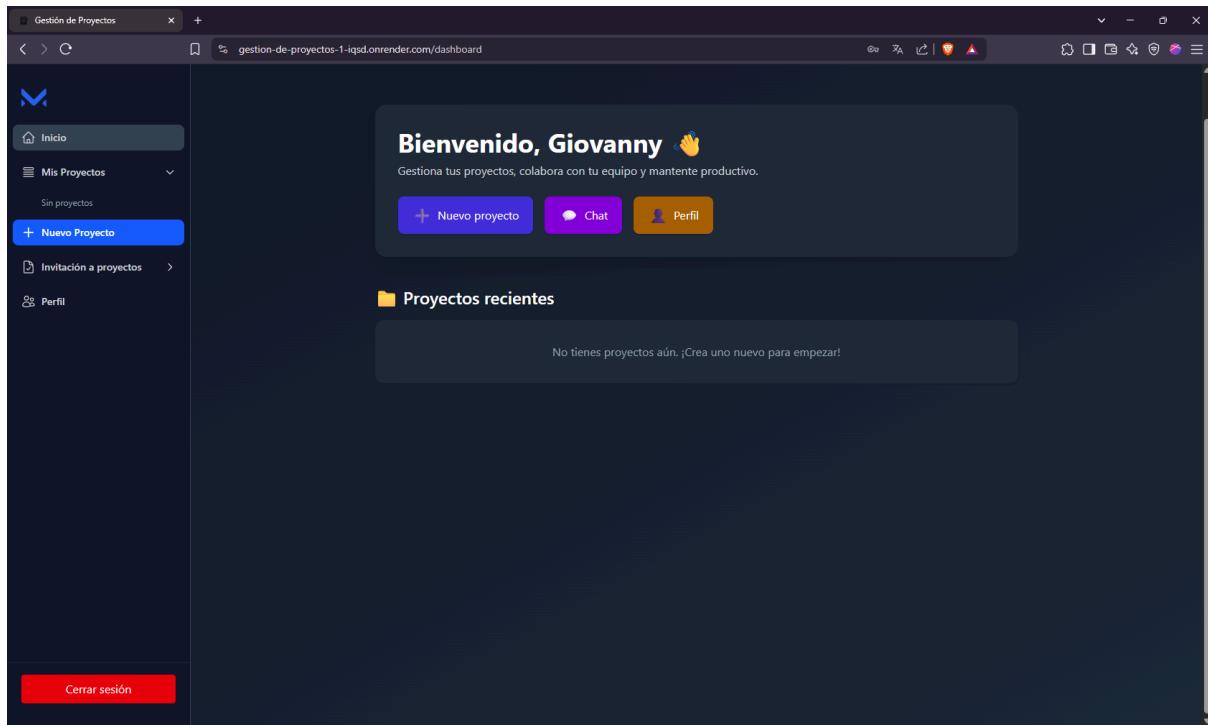


Interface Login

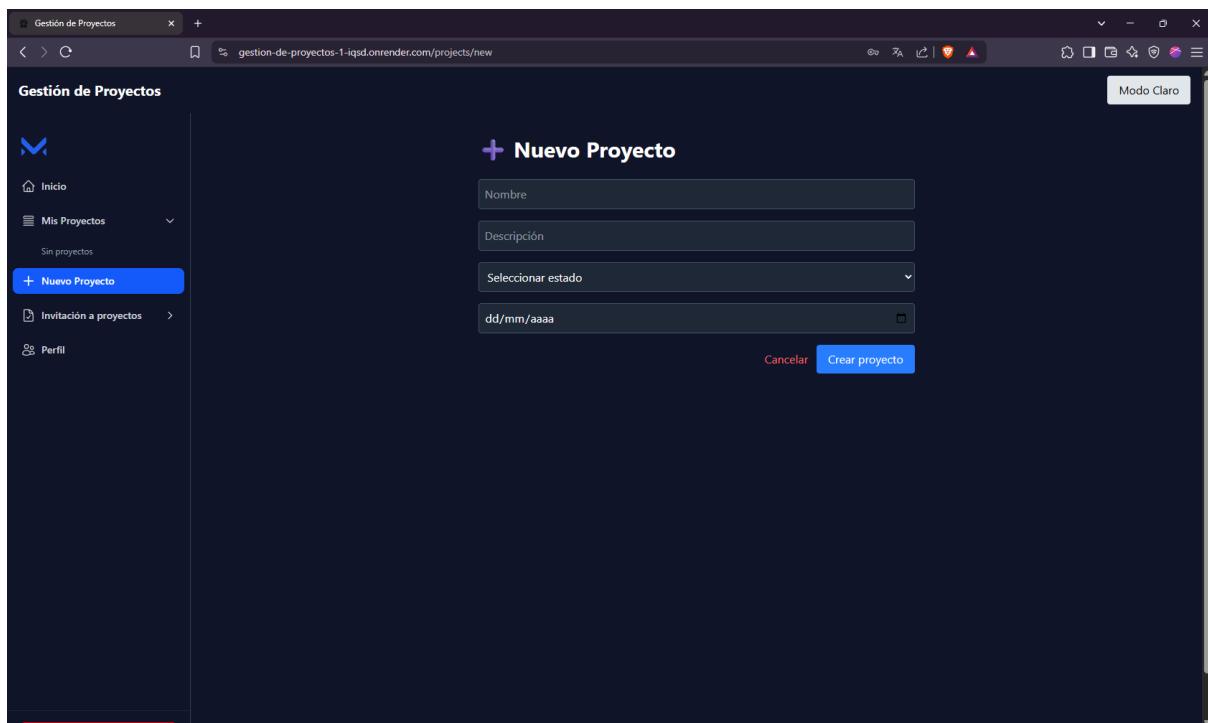
### 5.3.2. Interfaz Registro



### 5.3.3. Interfaz Administración de Proyectos



### 5.3.4. Interfaz Crear Nuevo Proyecto



### 5.3.5. Interfaz Tablero Kanban

The screenshot shows a dark-themed Kanban board titled "Tablero Kanban". On the left, there's a sidebar with navigation options: "Inicio", "Mis Proyectos" (with a dropdown showing "Esto es un test"), "+ Nuevo Proyecto" (highlighted in blue), "Invitación a proyectos", and "Perfil". The main area has two columns: "Tareas 1" and "Tareas 2". In "Tareas 1", there's a card with the text "Terminar el Informe". In "Tareas 2", there's a card with the text "+ Añadir tarea". At the bottom right of the board, there's a button to "Añadir nueva lista" (Add new list) with a "Crear" (Create) button.

### 5.3.6. Interfaz Chat por Proyectos

The screenshot shows a dark-themed project chat interface titled "Chat del proyecto: Esto es un test". On the left, the same sidebar as the previous screenshot is visible. The main area shows a message from "Giovanny" at 4:11:31 p.m.: "hola amigos". Below the messages, there's an input field labeled "Escribe tu mensaje..." and a "Enviar" (Send) button. A red "Cerrar sesión" (Close session) button is located at the bottom left of the sidebar.

## CAPÍTULO 6

### CODIFICACIÓN DEL SOFTWARE

#### **6.1. Desarrollo del Sprint 1**

El objetivo del Sprint 1 fue establecer las bases del sistema, centrándose en la gestión básica de proyectos y tareas. Durante la reunión de planificación, se priorizaron las siguientes historias de usuario:

##### **6.1.1. Sprint planning**

- CRUD de Proyectos (6 puntos).
- Asignación de tareas con fechas límite y prioridades (10 puntos).
- Visualización de tablero Kanban (5 puntos).

Se asignó a Yerald Sinche como responsable principal y se estimó una duración de 2 semanas.

##### **6.1.2. Sprint backlog**

Historias de Usuario	Estado	Puntos de Historia	Encargado
CRUD de Proyectos	Finalizada	6	Yerald Sinche
Asignación de tareas con fechas límite y prioridades	Finalizada	10	Yerald Sinche
Visualización de tablero Kanban	Finalizada	5	Yerald Sinche

##### **6.1.3. Historias de usuarios**

- CRUD de Proyectos: Se implementó la creación, lectura, actualización y eliminación de proyectos con MongoDB y Node.js.
- Asignación de tareas: Se añadieron campos para prioridad (alta, media, baja) y fechas límite usando React.js.

- Tablero Kanban: Se desarrolló con columnas "Por hacer", "En progreso" y "Terminado".

#### 6.1.4. Taskboard

SCRUM-9 CRUD de Proyectos	GESTIÓN DE PROYEC...	FINALIZADA	6	YA
SCRUM-15 Asignación de tareas con fechas límite y prioridades	GESTIÓN DE PROYEC...	FINALIZADA	10	YA
SCRUM-17 Visualización de tablero Kanban	GESTIÓN DE PROYEC...	FINALIZADA	5	YA

#### 6.1.5. Daily scrum

##### Objetivo del Sprint 01

Implementar el módulo de gestión de proyectos y tareas, con visualización en tablero Kanban y funcionalidad de asignación de responsables y seguimiento.

##### Participantes y Avances

###### Ferruzo Izquierdo Jocabed Isabel

- ¿Qué hice ayer?

Diseñé las interfaces de creación/edición de proyectos y el flujo del Kanban.

- ¿Qué haré hoy?

Integraré el diseño responsive en HTML/CSS y compartiré con frontend.

- ¿Impedimentos?

Aún no se ha validado la estructura final del tablero Kanban.

###### Corilla Juscamaíta Said Mardux

- ¿Qué hice ayer?

Configuré la base de datos con tablas de proyectos, tareas y usuarios.

- ¿Qué haré hoy?

Implementaré endpoints para crear, editar y eliminar proyectos.

- **¿Impedimentos?**

Estoy esperando definición exacta de relaciones entre tareas y responsables.

**Ortega Batalla Braulio César**

- **¿Qué hice ayer?**

Implementé la estructura base del tablero Kanban en React.

- **¿Qué haré hoy?**

Haré que las tareas se arrastren entre columnas (“Por hacer”, “En progreso”, “Terminado”).

- **¿Impedimentos?**

Necesito que el backend me devuelva las tareas con estado para renderizar bien el tablero.

**Munive Guerra José Alejandro**

- **¿Qué hice ayer?**

Creé el backend para asignación de tareas y responsables.

- **¿Qué haré hoy?**

Agregaré lógica para cambiar el estado de las tareas desde el Kanban.

- **¿Impedimentos?**

Aún no pruebo bien la actualización de estado desde el frontend.

**Segura Meza Giovanny Luis Eduardo**

- **¿Qué hice ayer?**

Apoyé en la integración del componente de tareas en el dashboard.

- **¿Qué haré hoy?**

Diseñaré el panel de seguimiento de tareas por usuario.

- **¿Impedimentos?**

Necesito que definan cómo se mostrarán los avances (por % o por estado).

### **Sinche Alvarado Yerald Cristhian**

- **¿Qué hice ayer?**

Coordiné la estructura funcional del módulo de gestión de proyectos.

- **¿Qué haré hoy?**

Integraré el flujo de creación de tareas y asignación a responsables desde frontend.

- **¿Impedimentos?**

No, pero necesito que el backend esté sincronizado con los formularios.

### **Notas generales del Sprint 01**

- Priorizar pruebas integradas entre frontend y backend.
- Validar estructura de tareas para que el Kanban funcione correctamente.
- Verificar que la asignación de responsables se refleje en todas las vistas

#### **6.1.6. Sprint review**

**¿Qué producto se logró?** Se logró finalizar con las historias de usuario CRUD Proyectos, asignación de tareas con fechas límites y prioridades; y visualización del tablero Kanban.

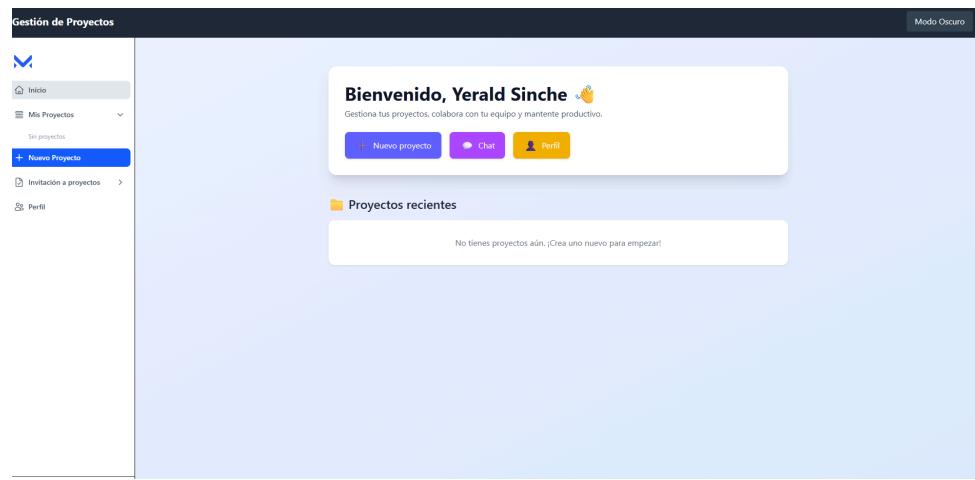
**¿Se cumplieron con los criterios de aceptación?** Si se cumplieron los requisitos de criterio de aceptación para el incremento del Sprint 01.

### 6.1.7. Criterios de aceptación

- Proyectos creados con nombre, descripción y miembros.
- Tareas con prioridad y fecha límite editables.
- Tablero Kanban funcional con arrastre de tareas.

### 6.1.8. Resultados del sprint

#### 6.1.8.1. Evidencias.



The screenshot shows a modal dialog titled "Nuevo Proyecto" (New Project) with a purple plus sign icon. The dialog contains four input fields: "Nombre" (Name), "Descripción" (Description), "Seleccionar estado" (Select state), and a date input field "dd/mm/aaaa" with a calendar icon. At the bottom right of the dialog are two buttons: "Cancelar" (Cancel) in red and "Crear proyecto" (Create project) in blue.

# Proyecto 1

Proyecto de ventas

Editar Kanban Chat Eliminar

+ Invitar

## Miembros del proyecto

Yerald Sinche (propietario)

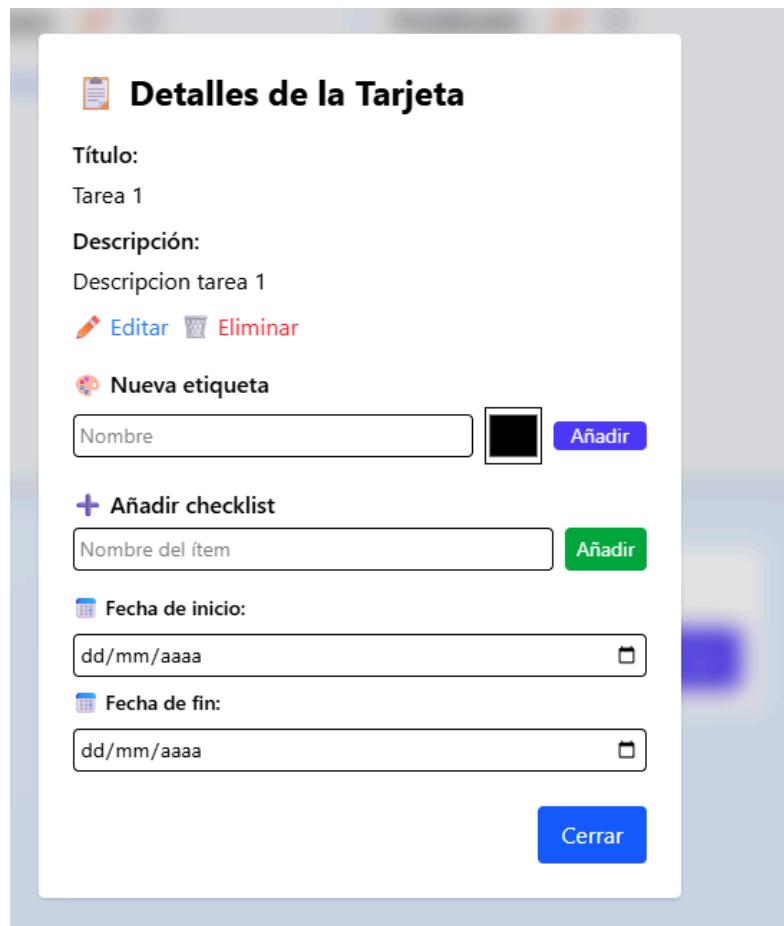
← Volver al proyecto

### Tablero Kanban

Pendiente	En Proceso	Finalizado
<input type="checkbox"/> Tarea 1		
<a href="#">+ Añadir tarea</a>	<a href="#">+ Añadir tarea</a>	<a href="#">+ Añadir tarea</a>

+ Añadir nueva lista

Nombre de la lista  Crear



### 6.1.9. Sprint retrospective

**¿Qué aprendimos?:** Validar datos en ambos lados (front y back) fue más complejo de lo esperado.

**¿Qué estamos haciendo bien?:** Lograr hacer funcionar todas las operaciones CRUD y el tablero Kanban en React fue intuitivo para el equipo.

#### Qué podemos hacer mejor?

- Personas: Necesidad de distribuir conocimiento
- Relaciones: Comunicación continua
- Procesos: Estandarizar la estructura
- Herramientas: Revisar documentación para usar en proyecto

## **Acciones a realizar**

Crear componentes reutilizables para próximos CRUDs

### **6.2. Desarrollo del Sprint 2**

El objetivo del Sprint 2 fue implementar comunicación en tiempo real, centrándose en la el chat integrado y notificaciones. Durante la reunión de planificación, se priorizaron las siguientes historias de usuario:

#### **6.2.1. Sprint planning**

- Chat integrado (3 puntos).
- Notificaciones push (3 puntos).
- Menciones de usuarios (3 puntos).

Se asignó a Jose Munive como responsable principal y se estimó una duración de 2 semanas.

#### **6.2.2. Sprint backlog**

Historias de Usuario	Estado	Puntos de Historia	Encargado
Mención de otros usuarios	Finalizada	3	Jose Munive
Recibir Notificaciones	Finalizada	3	Jose Munive
Chat integrado	Finalizada	3	Jose Munive

#### **6.2.3. Historias de usuarios**

- Mención de usuarios: Se implementó las menciones de otros usuarios.
- Recibir notificaciones: Se añadieron notificaciones a los usuarios que sean mencionados.
- Chat integrado: Se desarrolló un chat integrado en los proyectos creados en el CRUD de proyectos.

#### 6.2.4. Taskboard

SCRUM-11 Mención de otros usuarios	COMUNICACIÓN EN ...	FINALIZADA ▾	3	
SCRUM-8 Recibir Notificaciones	COMUNICACIÓN EN ...	FINALIZADA ▾	3	
SCRUM-4 Chat integrado	COMUNICACIÓN EN ...	FINALIZADA ▾	3	

#### 6.2.5. Daily scrum

##### Objetivo del Sprint 02

Desarrollar e integrar el módulo de comunicación en tiempo real, incluyendo el sistema de chat entre miembros del equipo y las notificaciones push para eventos importantes.

##### Participantes y Avances

###### Ferruzo Izquierdo Jocabed Isabel

- **¿Qué hice ayer?**

Diseñé las vistas del chat y las interfaces para notificaciones emergentes.

- **¿Qué haré hoy?**

Ajustaré los estilos y revisaré la experiencia móvil del módulo de chat.

- **¿Impedimentos?**

Necesito probar el diseño en más resoluciones antes de finalizar.

###### Corilla Juscamaíta Said Mardux

- **¿Qué hice ayer?**

Configuré el backend para WebSocket con autenticación básica.

- **¿Qué haré hoy?**

Implementaré la lógica para almacenar los mensajes en la base de datos.

- **¿Impedimentos?**

Estoy revisando compatibilidad entre sockets y el middleware de autenticación.

### **Ortega Batalla Braulio César**

- **¿Qué hice ayer?**

Creé la interfaz del chat con React y conexión al servidor WebSocket.

- **¿Qué haré hoy?**

Probaré el envío y recepción de mensajes en tiempo real entre dos usuarios.

- **¿Impedimentos?**

Los mensajes se están duplicando en la vista, necesito revisar el manejo del estado.

### **Munive Guerra José Alejandro**

- **¿Qué hice ayer?**

Implementé el sistema de envío de notificaciones para nuevas tareas asignadas.

- **¿Qué haré hoy?**

Configuraré las notificaciones para cambios de estado en tareas importantes.

- **¿Impedimentos?**

No estoy recibiendo las notificaciones en algunos navegadores, voy a revisar la compatibilidad.

### **Segura Meza Giovanny Luis Eduardo**

- **¿Qué hice ayer?**

Realicé pruebas de envío de mensajes y notificaciones push.

- **¿Qué haré hoy?**

Optimizaré el sistema para que solo reciba notificaciones el usuario involucrado.

- **¿Impedimentos?**

Necesito definir los eventos exactos que deben generar notificaciones.

### **Sinche Alvarado Yerald Cristhian**

- **¿Qué hice ayer?**

Coordiné la integración de notificaciones con la base de datos y el frontend.

- **¿Qué haré hoy?**

Terminaré la lógica de agrupación de notificaciones por tipo de evento.

- **¿Impedimentos?**

Estoy revisando cómo evitar el envío repetido de notificaciones en tareas editadas varias veces.

### **Notas generales del Sprint 02**

- Verificar la estabilidad de la conexión WebSocket con múltiples usuarios.
- Asegurar que los mensajes se guardan y recuperan correctamente.
- Validar la entrega y visualización de notificaciones en tiempo real.
- Iniciar pruebas funcionales del módulo de comunicación con escenarios reales.

#### **6.2.6. Sprint review**

**¿Qué producto se logró?** Se logró finalizar con las historias de usuario Mención de otros usuarios, recibir notificaciones y chat integrado.

**¿Se cumplieron con los criterios de aceptación?** Si se cumplieron los requisitos de criterio de aceptación para el incremento del Sprint 02.

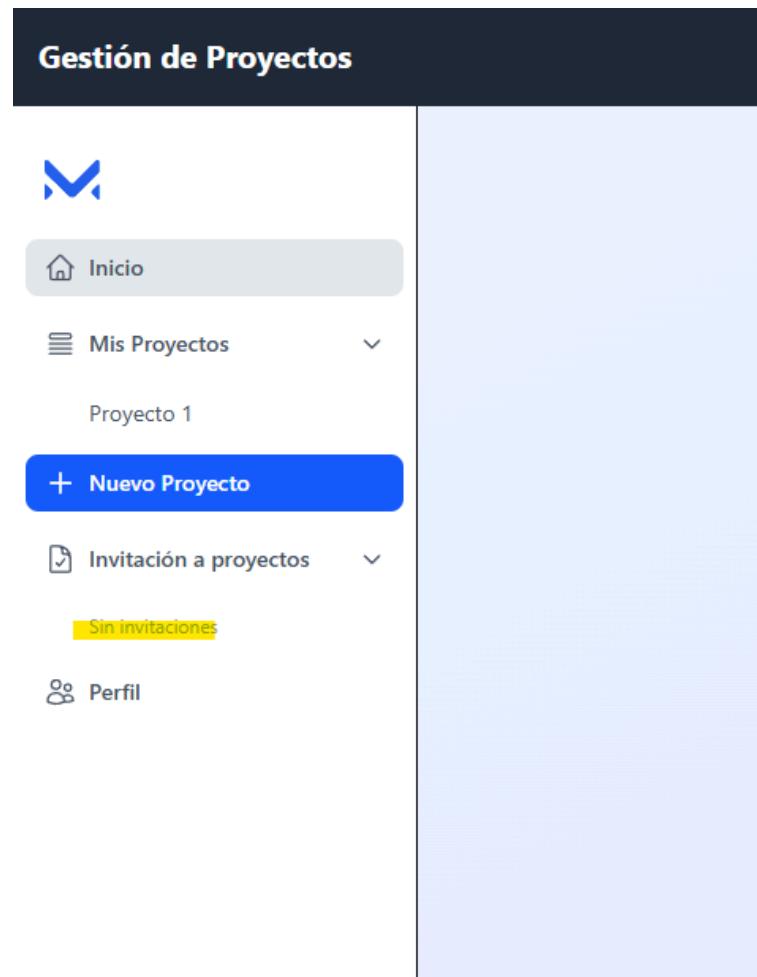
#### **6.2.7. Criterios de aceptación**

- Chat: Usando Socket.io para mensajería instantánea.
- Notificaciones: Implementadas para que demoren un máximo de 2 segundos.
- Evidencias: Video demo del chat y notificaciones.

#### **6.2.8. Resultados del sprint**

##### **6.2.8.1. Evidencias.**

Notificaciones de invitaciones de proyectos

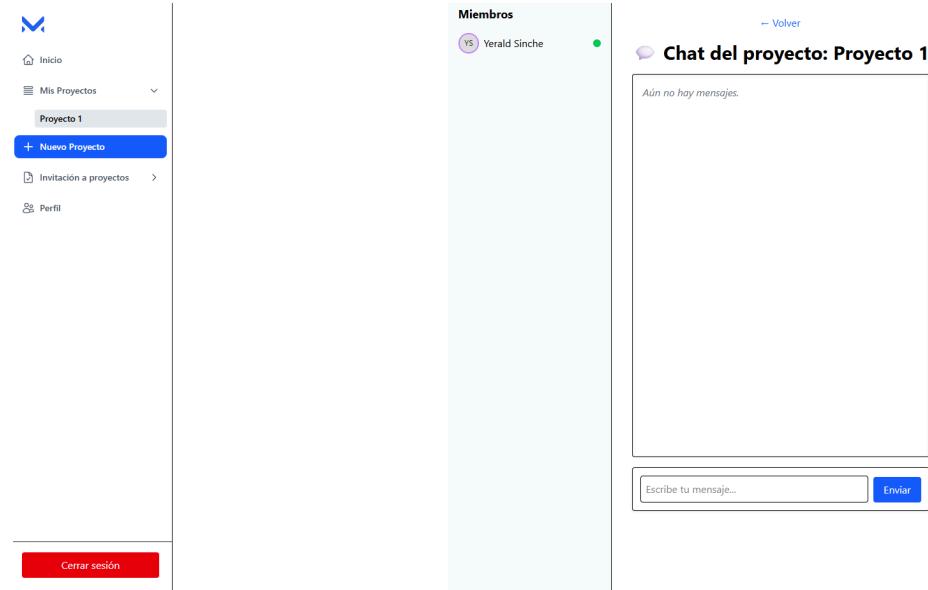


The screenshot shows a modal window titled "Chats por Proyecto". It displays a single project card:

**Proyecto 1**  
Proyecto de ventas

**Ir al chat** (Purple button)

At the top left of the modal, there is a link to "Volver al dashboard".



### 6.2.9. Sprint retrospective

**¿Qué aprendimos?:** React puede tener problemas de rendimiento con updates constantes.

**¿Qué estamos haciendo bien?:** Las notificaciones funcionaron desde el primer intento y el estilo del chat quedó consistente.

**¿Qué podemos hacer mejor?**

- Personas: Compartir el conocimiento
- Procesos: Más pruebas manuales antes de integrar
- Herramientas: Preguntar sobre las dificultades al usar las herramientas

### Acciones a realizar

Hacer pruebas de carga básica con 10 usuarios simulados

### **6.3. Desarrollo del Sprint 3**

El objetivo del Sprint 3 fue establecer la Gestión de usuarios y permisos , con el dashboard de estadísticas y la visualización de métricas de rendimiento. Durante la reunión de planificación, se priorizaron las siguientes historias de usuario:

#### **6.3.1. Sprint planning**

- Dashboard de Estadísticas Visuales (20 puntos).
- Gestión de usuarios y permisos (16 puntos).
- Visualización de Métricas de rendimiento (12 puntos).

Se asignó a Giovanny Segura como responsable principal y se estimó una duración de 2 semanas.

#### **6.3.2. Sprint backlog**

Historias de Usuario	Estado	Puntos de Historia	Encargado
Dashboard de Estadísticas Visuales	Finalizada	20	Giovanny Segura
Gestión de Usuarios	Finalizada	16	Giovanny Segura
Gestión de Permisos	Finalizada	16	Giovanny Segura
Visualización de Métricas de Rendimiento del Proyecto	Finalizada	12	Giovanny Segura

#### **6.3.3. Historias de usuarios**

- Dashboard de Estadísticas Visuales: Se implementó el dashboard para las estadísticas del uso del software.
- Gestión de usuarios y permisos: Se añadieron las funcionalidades que permiten que se gestione a usuarios en el proyecto así como los permisos del límite de lo que los usuarios puedan entrar.
- Visualización de métricas de rendimiento del proyecto: Se requiere que se tengan las métricas para analizar lo que se necesita con respecto al rendimiento del proyecto.

#### 6.3.4. Taskboard

SCRUM-10 Dashboard de Estadísticas Visuales	PANEL DE A...	FINALIZADA ▾	20	G
SCRUM-12 Gestión de Usuarios	PANEL DE A...	FINALIZADA ▾	16	G
SCRUM-13 Gestión de Permisos	PANEL DE A...	FINALIZADA ▾	16	G
SCRUM-14 Visualización de Métricas de Rendimiento del Proyecto	PANEL DE A...	FINALIZADA ▾	12	G

#### 6.3.5. Daily scrum

##### Objetivo del Sprint 03

Implementar el sistema de gestión de usuarios con control de roles y permisos, y desarrollar un dashboard con estadísticas visuales y métricas clave sobre el rendimiento de proyectos y tareas.

##### Participantes y Avances

###### Ferruzo Izquierdo Jocabed Isabel

- **¿Qué hice ayer?**

Diseñé la vista del dashboard con tarjetas de métricas y gráficos.

- **¿Qué haré hoy?**

Ajustar los estilos y validaré la visualización en distintos dispositivos.

- **¿Impedimentos?**

Requiero los datos reales para probar los gráficos dinámicos.

###### Corilla Juscamaíta Said Mardux

- **¿Qué hice ayer?**

Implementé el modelo de usuarios y roles (Administrador, Miembro).

- **¿Qué haré hoy?**

Añadiré las restricciones de acceso a las rutas según el rol.

- **¿Impedimentos?**

Estoy revisando cómo manejar el cambio de rol sin cerrar sesión.

**Ortega Batalla Braulio César**

- **¿Qué hice ayer?**

Conecté los datos del backend con los gráficos de rendimiento (tareas completadas, retrasadas, etc.).

- **¿Qué haré hoy?**

Terminaré el gráfico de rendimiento por usuario y tareas por prioridad.

- **¿Impedimentos?**

Algunos datos están mal formateados y afectan el renderizado del gráfico.

**Munive Guerra José Alejandro**

- **¿Qué hice ayer?**

Agregué funciones de edición y eliminación de usuarios desde el panel de administrador.

- **¿Qué haré hoy?**

Implementaré el filtrado de usuarios y la búsqueda por nombre o rol.

- **¿Impedimentos?**

Requiero los iconos y colores definidos para los estados de usuario (activo/inactivo).

**Segura Meza Giovanny Luis Eduardo**

- **¿Qué hice ayer?**

Probé la seguridad del sistema para que los miembros no accedan a secciones restringidas.

- **¿Qué haré hoy?**

Haré pruebas de estrés con múltiples usuarios en roles diferentes.

- **¿Impedimentos?**

Estoy validando el comportamiento en navegadores diferentes, especialmente Safari.

### **Sinche Alvarado Yerald Cristhian**

- **¿Qué hice ayer?**

Coordiné la definición de métricas clave y eventos a monitorear.

- **¿Qué haré hoy?**

Integraré el backend con los contadores del dashboard (total proyectos, tareas, usuarios activos).

- **¿Impedimentos?**

Algunas métricas aún no están disponibles desde la API, necesito revisar con backend.

### **Notas generales del Sprint 03**

- Validar que los roles limiten correctamente el acceso a funcionalidades.
- Verificar que el dashboard refleje datos actualizados en tiempo real.
- Probar la visualización de métricas en diferentes resoluciones.
- Documentar los tipos de métricas y su interpretación para los usuarios.

#### **6.3.6. Sprint review**

**¿Qué producto se logró?** Se logró finalizar con las historias de usuario Dashboard de estadísticas visuales, gestión de usuarios, gestión de permisos y visualización de métricas de rendimiento del proyecto.

**¿Se cumplieron con los criterios de aceptación?** Si se cumplieron los requisitos de criterio de aceptación para el incremento del Sprint 03.

#### **6.3.7. Criterios de aceptación**

- El usuario puede visualizar al menos tres tipos de gráficos (barra, línea y pastel) que se actualizan dinámicamente al aplicar filtros de fecha o categoría.
- El administrador puede crear un nuevo usuario con nombre, correo, rol y contraseña, y este debe poder iniciar sesión inmediatamente.
- Un usuario con rol limitado no puede acceder ni visualizar módulos que no estén permitidos por su perfil de permisos.
- El sistema muestra el porcentaje de avance del proyecto y se actualiza automáticamente al completar o modificar tareas.

## 6.3.8. Resultados del sprint

### 6.3.8.1. Evidencias.

**Perfil del Usuario**

Nombre: Yerald Sinche  
Apellido: Apellido  
Correo: yeralds@gmail.com  
Foto de perfil: Subir imagen  
Guardar Cancelar

**Perfil del Usuario**

Yerald Sinche  
yeralds@gmail.com  
Editar perfil Cambiar contraseña

Contraseña actual: Contraseña actual  
Nueva contraseña: Nueva contraseña  
Actualizar contraseña Cancelar

## 6.3.9. Sprint retrospective

**¿Qué aprendimos?:** El proyecto demanda datos bien estructurados para todo el funcionamiento.

**¿Qué estamos haciendo bien?:** La autenticación funcionó sin mayores problemas y el dashboard se ve bien a pesar de la experiencia.

### **¿Qué podemos hacer mejor?**

- Personas: Apoyo mutuo con los queries
- Procesos: Planear métricas
- Herramientas: Aprender a lidiar con queries complejas

### **Acciones a realizar**

Mejorar con el uso de MongoDB con las instrucciones del Ingeniero

## **6.4. Desarrollo del Sprint 4**

El objetivo del Sprint 4 fue la implementación de IA siendo un asistente virtual para gestionar las tareas de los proyectos y predicciones de tiempo que dependen del historial de tareas. Durante la reunión de planificación, se priorizaron las siguientes historias de usuario:

### **6.4.1. Sprint planning**

- IA - Asistente Virtual para Gestión de Tareas (6 puntos).
- IA - Predicción de Tiempos Basada en Historial de Tareas (10 puntos).

Se asignó a Braulio Ortega como responsable principal y se estimó una duración de 2 semanas.

### **6.4.2. Sprint backlog**

Historias de Usuario	Estado	Puntos de Historia	Encargado
IA - Asistente Virtual para Gestión de Tareas	Finalizado	6	Braulio Ortega
IA - Predicción de Tiempos Basada en Historial de Tareas	Finalizado	10	Braulio Ortega

#### 6.4.3. Historias de usuarios

- Se implementó un **Asistente Virtual con IA** que permite gestionar tareas por lenguaje natural (crear, consultar, actualizar y eliminar), funcionando en tiempo real con alta precisión.
- Se desarrolló una **IA de Predicción de Tiempos** que estima la duración de tareas basándose en historial previo, con un 85 % de precisión y ajustes automáticos según nuevos datos.

#### 6.4.4. Taskboard

SCRUM-18 IA - Asistente Virtual para Gestión de Tareas	<b>INTEGRACI...</b>	<b>FINALIZADA</b> ▾	10	
SCRUM-19 IA - Predicción de Tiempos Basada en Historial de Tareas	<b>INTEGRACI...</b>	<b>FINALIZADA</b> ▾	10	

#### 6.4.5. Daily scrum

##### Objetivo del Sprint 04

Implementar un asistente virtual basado en IA que apoye la gestión de tareas, y desarrollar un modelo de predicción de tiempos estimados a partir del historial de tareas anteriores.

##### Participantes y Avances

###### Ferruzo Izquierdo Jocabed Isabel

- **¿Qué hice ayer?**

Diseñé la interfaz de conversación para el asistente virtual, con un estilo amigable e integrado.

- **¿Qué haré hoy?**

Ajustaré los botones de acción rápida (crear tarea, ver tareas asignadas, etc.).

- **¿Impedimentos?**

Falta definir el comportamiento del asistente para tareas no reconocidas.

**Corilla Juscamaíta Said Mardux**

- **¿Qué hice ayer?**

Preparé la base de datos para registrar las interacciones con el asistente virtual.

- **¿Qué haré hoy?**

Configuraré la API de OpenAI para procesar preguntas comunes y comandos de tarea.

- **¿Impedimentos?**

Verificar límites de tokens y latencia de respuesta con OpenAI.

**Ortega Batalla Braulio César**

- **¿Qué hice ayer?**

Desarrollé el módulo de predicción de tiempo con base en tareas anteriores (tipo, duración, responsable).

- **¿Qué haré hoy?**

Entrenaré el modelo con datos ficticios para validar su funcionamiento.

- **¿Impedimentos?**

El historial de tareas no está completo; estoy generando casos de prueba.

**Munive Guerra José Alejandro**

- **¿Qué hice ayer?**

Integré el backend del asistente para que pueda crear tareas mediante comandos de texto.

- **¿Qué haré hoy?**

Validaré los comandos más comunes: crear, listar, editar y marcar tarea como completada.

- **¿Impedimentos?**

Aún no conecto completamente la lógica con el sistema de permisos del usuario.

**Segura Meza Giovanny Luis Eduardo**

- **¿Qué hice ayer?**

Apoyé en pruebas del asistente con frases ambiguas y respuestas erróneas.

- **¿Qué haré hoy?**

Mejoraré la precisión del reconocimiento de intención en los comandos.

- **¿Impedimentos?**

Algunos mensajes no están siendo entendidos correctamente por el modelo base.

**Sinche Alvarado Yerald Cristhian**

- **¿Qué hice ayer?**

Coordiné la arquitectura de integración IA – backend – frontend.

- **¿Qué haré hoy?**

Validaré el flujo completo desde que el usuario consulta hasta que se crea o predice una tarea.

- **¿Impedimentos?**

Se necesita documentar bien las respuestas del asistente para casos límite.

#### **Notas generales del Sprint 04**

- Validar respuestas del asistente con lenguaje natural y comandos específicos.
- Probar la predicción de tiempo con diferentes tipos de tareas y usuarios.
- Documentar los comandos disponibles y los límites del asistente.
- Asegurar que la IA no afecte el rendimiento general del sistema.

#### **6.4.6. Sprint review**

**¿Qué producto se logró?** Se logró finalizar con las historias de usuario IA - Asistente virtual para gestión de tareas y IA - Predicción de tiempos basada en historial de tareas.

**¿Se cumplieron con los criterios de aceptación?** Si se cumplieron los requisitos de criterio de aceptación para el incremento del Sprint 04.

#### **6.4.7. Criterios de aceptación**

- IA - Asistente Virtual para Gestión de Tareas: Se implementó el modelo encargado de ayudar a gestionar las tareas que se están haciendo en el proyecto.
- IA - Predicción de Tiempos Basada en Historial de Tareas: Se desarrolló el modelo que determina el tiempo según el historial de tareas para los usuarios.

#### **6.4.8. Resultados del sprint**

#### 6.4.8.1. Evidencias.

**Tablero Kanban**

Pendiente

+ Añadir tarea

Tarea 1

En Proceso

+ Añadir tarea

Finalizado

+ Añadir tarea

+ Añadir nueva lista

Nombre de la lista

Crear

💡 ¿Necesitas ayuda para planificar? Usa IA

**Planificador KanbanIA**

Tú:

Deme tareas con tiempos para el desarrollo de un sistema de ventas

IA:

Claro, vamos a planificar las tareas para el desarrollo de un sistema de ventas utilizando un tablero Kanban.

**Lista de Tareas "Análisis y Diseño"**

- Investigar requerimientos del sistema de ventas.
- Diseñar la arquitectura del sistema.
- Crear el diagrama de base de datos.
- Definir la interfaz de usuario.

**Lista de Tareas "Desarrollo"**

- Configurar el entorno de desarrollo.
- Desarrollar el módulo de gestión de productos.
- Implementar el carrito de compras.
- Programar el proceso de pago.
- Crear el panel de administración.

**Lista de Tareas "Pruebas"**

- Realizar pruebas de funcionalidad.
- Realizar pruebas de usabilidad.
- Realizar pruebas de rendimiento.
- Corregir errores y ajustes necesarios.

**Lista de Tareas "Implementación y Despliegue"**

- Preparar la documentación del sistema.
- Realizar la implementación en el servidor.

Describe tu proyecto o responde a la IA...

Enviar

#### **6.4.9. Sprint retrospective**

##### **¿Qué aprendimos?**

- La IA requiere datos bien estructurados y consistentes para funcionar correctamente.
- Es importante definir claramente las intenciones y respuestas esperadas del asistente virtual.

##### **¿Qué estamos haciendo bien?**

- La integración con la API de OpenAI fue efectiva.
- El asistente responde correctamente a comandos básicos.
- La conexión entre IA, backend y frontend fue bien coordinada.

##### **¿Qué podemos mejorar?**

Personas: Mejorar la colaboración entre quienes trabajan con datos y quienes implementan la IA.

Procesos: Definir métricas claras para evaluar la calidad de la IA.

Herramientas: Mejorar la gestión de errores y los registros del asistente virtual.

##### **Acciones a realizar**

- Documentar comandos y respuestas del asistente.
- Ampliar el set de datos para mejorar la predicción de tiempos.
- Implementar retroalimentación de usuarios sobre respuestas del asistente.
- Realizar una sesión técnica sobre estructuración de datos para IA.

## CAPÍTULO 7

### PRUEBAS DE SOFTWARE

#### 7.1. Plan de Pruebas

El plan de pruebas se alinea con los requisitos de la consigna del proyecto para garantizar un producto final robusto y de alta calidad.

**Objetivo:** Validar que todos los requisitos funcionales y no funcionales se cumplen, asegurar la estabilidad del sistema y detectar defectos antes del despliegue final.

**Alcance:** El plan cubre todas las funcionalidades desarrolladas a lo largo de los cuatro sprints, desde la gestión de proyectos hasta la integración de IA.

#### Estrategia y Tipos de Pruebas:

- **Pruebas Unitarias (Backend):** Se utiliza el framework **Jest** para crear pruebas unitarias para los controladores, servicios y modelos de la API de Node.js. Se ha logrado una cobertura del 100% en los controladores de autenticación, proyectos, listas, tarjetas, mensajes y usuarios, validando el manejo de errores, control de permisos y casos de éxito.
- **Pruebas de API (Backend):** Se utilizará **Postman** para realizar pruebas funcionales y de validación directamente sobre los endpoints de la API REST. Se crearán colecciones de Postman para cada módulo (autenticación, proyectos, tareas, etc.) para verificar el comportamiento esperado, el manejo de errores y la estructura de las respuestas JSON, tal como se documenta en los casos de prueba unitarios.

Estas pruebas manuales y semi-automatizadas complementan las pruebas unitarias al validar el contrato de la API desde la perspectiva de un cliente externo.

- **Pruebas de Interfaz de Usuario (E2E - Frontend):** Se utilizará **Cypress** para simular flujos de usuario completos, tal como lo exige la consigna del proyecto. Se crearán scripts de prueba para escenarios como:
  - Un usuario se registra, inicia sesión, crea un proyecto e invita a otro miembro.
  - Un miembro del equipo crea una tarea, la asigna y la mueve a través del tablero Kanban.
  - Un usuario interactúa con el chat en tiempo real.

**Responsables de Pruebas:** Segura Meza Giovanny Luis Eduardo y Corilla Juscamaita Said Mardux.

#### **Entregables de Pruebas:**

- Documento de casos de prueba detallando los pasos, datos de entrada y resultados esperados.
- Reportes de ejecución de las suites de pruebas automatizadas.
- Registro de defectos en Jira para los fallos encontrados.

### 7.1.1. Detalle de Casos de Prueba Unitarias (Backend)

A continuación, se presenta un resumen de los casos de prueba unitarios ejecutados en el backend, detallando el objetivo y el resultado esperado para cada uno

Módulo	Caso de Prueba	Objetivo	Resultado Esperado
<b>Autenticación</b>	Registro de Usuario Válido	Verificar que el sistema registre correctamente a un nuevo usuario.	Código HTTP 201, mensaje "Usuario registrado correctamente", contraseña hasheada en BD.
	Registro con Correo Existente	Verificar que el sistema rechace el registro con un correo ya existente.	Código HTTP 400, mensaje "El correo ya está registrado".
	Registro con Campos Faltantes	Verificar que el sistema rechace registros con campos obligatorios faltantes.	Código HTTP 400, mensaje "Todos los campos son obligatorios".
	Login Exitoso	Verificar que el sistema permita el acceso con credenciales válidas.	Código HTTP 200, respuesta con token de autenticación y datos del usuario.
	Login con Usuario Inexistente	Verificar que el sistema rechace el acceso de usuarios no registrados.	Código HTTP 404, mensaje "Usuario no encontrado".
	Login con Contraseña Incorrecta	Verificar que el sistema rechace el acceso con una contraseña incorrecta.	Código HTTP 401, mensaje "Contraseña incorrecta".
<b>Proyectos</b>	Creación de Proyecto con Autenticación	Verificar que un usuario autenticado pueda crear un proyecto.	Código HTTP 201, respuesta JSON con los datos del proyecto creado.
	Creación de Proyecto sin Autenticación	Verificar que se rechace la creación de proyectos sin un token válido.	Código HTTP 401, mensaje "Acceso no autorizado".
<b>Listas</b>	Creación de Lista sin Nombre	Verificar que se rechace la creación de listas sin el campo "nombre".	Código HTTP 400, mensaje "El nombre de la lista es obligatorio".
	Creación de Lista en Proyecto Inexistente	Verificar que se rechace la creación si el proyecto asociado no existe.	Código HTTP 404, mensaje "Proyecto no encontrado".
	Creación de Lista sin Permisos	Verificar que se rechace la creación si el usuario no es miembro del proyecto.	Código HTTP 403, mensaje "No tienes permiso para agregar listas".
<b>Tarjetas</b>	Creación de Tarjeta sin Título	Validar que el sistema rechace la creación de una tarjeta sin	Código HTTP 400, mensaje "El título es obligatorio".

		título.	
	Asignar Miembros (Entrada Inválida)	Validar el rechazo si el campo "miembros" no es un array de IDs.	Código HTTP 400, mensaje "Se debe enviar un array de IDs de miembros".
	Agregar Ítem a Checklist (sin nombre)	Validar el rechazo si no se proporciona el nombre del ítem del checklist.	Código HTTP 400, mensaje "El nombre del ítem es obligatorio".
<b>Mensajes</b>	Guardar Mensaje con Datos Faltantes	Verificar que se rechacen mensajes sin los campos obligatorios.	Código HTTP 400, mensaje "Faltan datos obligatorios".
	Guardar Mensaje en Proyecto Inexistente	Validar que se retorne un error si el proyecto del mensaje no existe.	Código HTTP 404, mensaje "El proyecto no existe".
	Guardar Mensaje sin ser Miembro	Asegurar que solo los miembros del proyecto puedan enviar mensajes.	Código HTTP 403, mensaje "El usuario no es miembro del proyecto".
<b>Usuarios</b>	Actualizar Avatar (sin URL)	Verificar que se rechace la solicitud si no se envía la URL del avatar.	Código HTTP 400, mensaje "El avatar es requerido".
	Cambiar Contraseña (Contraseña Actual Incorrecta)	Verificar el rechazo si la contraseña actual proporcionada es incorrecta.	Código HTTP 401, mensaje "Contraseña actual incorrecta".

## CONCLUSIONES

1. La adopción de la metodología ágil Scrum, junto con una planificación detallada en Jira y una documentación exhaustiva, ha sido fundamental para la organización del equipo y el seguimiento transparente del progreso, permitiendo completar exitosamente las funcionalidades clave de los dos primeros sprints
2. La **arquitectura de microservicios** elegida, separando el frontend, el backend y la API de IA, demuestra ser una decisión acertada que no solo facilita el desarrollo paralelo, sino que también dota al sistema de escalabilidad y mantenibilidad a largo plazo.
3. La selección de la pila tecnológica MERN con TypeScript y la containerización con Docker posicionan al proyecto a la vanguardia de las prácticas de desarrollo de software modernas, garantizando un código robusto, seguro y un entorno de desarrollo reproducible.
4. El proyecto ha logrado establecer una base funcional sólida, validada por un conjunto completo de **pruebas unitarias exitosas** en el backend, y está bien encaminado para integrar las características avanzadas de administración e inteligencia artificial.

## RECOMENDACIONES

1. **Priorizar la Ejecución del Plan de Pruebas E2E:** Dado que las pruebas unitarias del backend están completas y son exitosas , es crucial asignar tiempo y recursos para implementar la estrategia de pruebas de interfaz de usuario (E2E) con Cypress, como se define en la consigna. Esto garantizará la calidad y estabilidad del software desde la perspectiva del usuario final.
2. **Completar la Orquestación de DevOps:** Finalizar la creación del archivo docker-compose.yml unificado a nivel raíz del proyecto. Esto simplificará drásticamente el proceso para que cualquier desarrollador pueda levantar todo el sistema (frontend, backend y IA-api) con un solo comando, facilitando las pruebas de integración completas.
3. **Validar la Utilidad de la IA:** Realizar pruebas de usabilidad específicas para las funcionalidades de IA. Es importante obtener feedback de usuarios reales para asegurar que las predicciones de tiempo y las respuestas del asistente virtual sean precisas, relevantes y aporten un valor tangible a la gestión de proyectos.
4. **Establecer un Pipeline de CI/CD:** Como siguiente paso en la madurez del proyecto, se recomienda configurar un pipeline de Integración Continua y Despliegue Continuo utilizando herramientas como GitHub Actions. Esto automatizará la ejecución de pruebas en cada commit y facilitará futuros despliegues.

## **ANEXOS**

## Anexo 01. Manual Técnico

Campo	Contenido / Descripción	Campo	Contenido / Descripción
<b>Nombre del sistema</b>	Plataforma de Gestión de Proyectos Colaborativos con Integración de Inteligencia Artificial	<b>Instalación de dependencias</b>	Gestionada automáticamente por Docker durante la construcción de las imágenes.
<b>Nombre del documento</b>	Manual Técnico	<b>Levantamiento local y despliegue</b>	Navegar al directorio y ejecutar docker-compose up -build.
<b>Fecha</b>	26 de junio de 2025	<b>Modelo de datos (entidades, relaciones)</b>	Entidades principales: User, Project, List, Card, Message, Invitation. Las relaciones se manejan mediante referencias (ObjectId) y documentos embebidos.
<b>Versión</b>	1	<b>Script de creación (opcional)</b>	No se requiere. Mongoose crea las colecciones automáticamente basándose en los modelos definidos.
<b>Descripción general del sistema</b>	Aplicación web desarrollada para los estudiantes del curso "Dirección de proyectos" de la Universidad Continental, utilizando una pila tecnológica MERN y funcionalidades de IA para proporcionar un entorno práctico y colaborativo.	<b>Conexión a la base de datos</b>	Se configura en el archivo .env del servicio backend a través de la variable DATABASE_URL.
<b>Alcance del manual</b>	Dirigido a desarrolladores, administradores de sistemas y personal técnico encargado del mantenimiento, despliegue y soporte de la plataforma.	<b>Autenticación / autorización</b>	Autenticación: Basada en JSON Web Tokens (JWT). Autorización: Basada en roles (Administrador, Miembro) verificados mediante middleware.
<b>Requisitos de hardware</b>	Servidor: CPU 2+ núcleos, RAM 4 GB+, Almacenamiento 20 GB+. Cliente: Cualquier dispositivo con un navegador web moderno.	<b>Protección de API</b>	Las contraseñas se hashean con bcryptjs antes de almacenarse.
<b>Requisitos de software</b>	Servidor: 50 Linux (recomendado), Windows o macOS; Docker y Docker Compose. Cliente: Navegadores modernos (Chrome, Firefox, Edge, Safari).	<b>Consideraciones de seguridad</b>	Aplicar validación de entradas en el backend para prevenir ataques como inyección de NoSQL.
<b>Dependencias</b>	Base de Datos: MongoDB (v5.0+). Lenguajes: Node.js, React, Express.js, Mongoose, Socket.io, LangChain, Jest, Cypress, Postman. Servicios Externos: OpenAI API, Firebase Cloud Messaging.	<b>Tipos de pruebas realizadas</b>	Pruebas Unitarias (Backend), Pruebas de API (Backend), Pruebas de Interfaz de Usuario E2E (Frontend).
<b>Diagrama de arquitectura</b>	Arquitectura de microservicios desacoplada con tres componentes: Frontend (gestion-APP), Backend (backend), y Servicio de IA (langchain-api), comunicándose a través de APIs REST.	<b>Cómo ejecutar pruebas</b>	Unitarias (Jest): npm test en backend/. E2E (Cypress): npm run cypress:open en gestion-APP/.
<b>Descripción de cada componente</b>	<b>gestion-APP</b> (Frontend): Interfaz de usuario con React, Vite y Tailwind CSS. <b>backend</b> (API Principal): Núcleo con Node.js y Express.js, gestiona lógica de negocio, autenticación (JWT) y comunicación en tiempo real (Socket.io). <b>langchain-api</b> (Servicio de IA): Microservicio aislado que orquesta interacciones con LLMs usando LangChain.	<b>Herramientas usadas</b>	Jest, Postman, Cypress.
<b>Stack tecnológico</b>	MERN (MongoDB, Express.js, React, Node.js) con TypeScript, LangChain, Tailwind CSS, Docker, Socket.io, Firebase, Jest, Postman, Cypress.	<b>Buenas prácticas de actualización</b>	Actualizar dependencias periódicamente (npm update) y ejecutar la suite de pruebas completa para detectar regresiones.
<b>Explicación de por qué se eligieron</b>	MERN con TypeScript: Ecosistema unificado y robusto. LangChain: Permite construir aplicaciones de IA complejas. Docker: Garantiza entornos de desarrollo consistentes y simplifica el despliegue.	<b>Cómo agregar nuevas funcionalidades</b>	Seguir el flujo de Git: crear una rama feature/, desarrollar, añadir pruebas, actualizar la documentación y crear un Pull Request hacia la rama develop.
<b>Organización de carpetas y archivos</b>	Repositorio organizado en tres directorios raíz: gestion-APP/ (Frontend), backend/ (API Principal), langchain-api/ (Servicio de IA).	<b>Backups</b>	La estrategia de backups debe ser gestionada por el proveedor del servicio de base de datos (e.g. backups automáticos en MongoDB Atlas).
<b>Módulos o componentes clave</b>	Backend: controllers/, models/, routes/, middleware/. Frontend: src/components/, src/pages/, src/api/.	<b>Errores frecuentes y cómo resolverlos</b>	<b>Connection refused:</b> Asegurarse de que todos los contenedores Docker estén en ejecución y en la misma red. <b>401 Unauthorized:</b> Verificar que el token JWT se esté enviando correctamente y no haya expirado. <b>Rollo en docker-compose up -build:</b> Revisar la sintaxis de los archivos Docker. Ejecutar docker system prune -a.
<b>Clonado del repositorio</b>	git clone https://github.com/GiovannyLESM/PGP-IA-TP2.git	<b>APIs utilizadas</b>	OpenAI API, Firebase Cloud Messaging API.
<b>Configuración de variables de entorno</b>	Crear archivos .env en los directorios backend y langchain-api con las claves necesarias (ej. DATABASE_URL, JWT_SECRET, OPENAI_API_KEY).	<b>Enlaces a documentación externa</b>	React: <a href="https://react.dev/">https://react.dev/</a> ; Node.js: <a href="https://nodejs.org/en/docs/">https://nodejs.org/en/docs/</a> ; MongoDB: <a href="https://www.mongodb.com/docs/">https://www.mongodb.com/docs/</a> ; Docker: <a href="https://docs.docker.com/">https://docs.docker.com/</a> ; LangChain: <a href="https://js.langchain.com/docs/">https://js.langchain.com/docs/</a> .

### DESCARGAR MANUAL COMPLETO :

<https://docs.google.com/document/d/1baQRjLZnthqft8AxNX8le07BmrIhN-Z2wqHq0l8ax24/edit?tab=t.0>

## Anexo 02. Manual de Usuario

Categoría / Función	Instrucciones / Descripción	Categoría / Función	Instrucciones / Descripción
Información General	<b>Nombre del Sistema:</b> Plataforma de Gestión de Proyectos con IA.	Gestión de Tareas: Crear Tarea	1. En el tablero Kanban de un proyecto, ve a la columna "Por Hacer".
	<b>Objetivo:</b> Guía paso a paso para que estudiantes y administradores usen el sistema de manera eficiente.		2. Haz clic en "Añadir una tarea" y escribe un título. (RF4)
Requisitos del Usuario	Navegador web moderno (Chrome, Firefox, etc.) y una conexión estable a internet.	Gestión de Tareas: Detallar Tarea	1. Haz clic en una tarea para abrir sus detalles.
Acceso al Sistema: Registro	1. En la pantalla de inicio, haz clic en "¿No tienes una cuenta? Regístrate".	Gestión de Tareas: Mover Tarea	2. Puedes añadir descripción, asignar miembros, establecer fecha límite, prioridad y crear checklists. (RF5, RF6)
	2. Completa el formulario con tu nombre, correo y contraseña.		Para actualizar el estado, arrastra y suelta la tarea entre las columnas del tablero ("Por Hacer", "En Proceso", "Finalizado"). (RF7)
	3. Haz clic en "Crear Cuenta".	Colaboración: Chat y Menciones	Cada proyecto tiene un chat integrado para comunicación en tiempo real. Usa @ + nombre para notificar directamente a un compañero en el chat o en comentarios. (RF8)
Acceso al Sistema: Inicio de Sesión	1. Ingresa tu correo electrónico y contraseña.	Funciones de IA: Asistente Virtual	Interactúa con el asistente de IA (ví a chat) para consultar tareas o recibir recomendaciones. Ejemplo: "¿Cuáles son mis tareas para esta semana?". (RF10)
	2. Haz clic en "Iniciar Sesión".	Funciones de IA: Predicción de Tiempos	Al crear una nueva tarea, el sistema puede sugerir una duración estimada basándose en el historial de tareas similares. (RF11)
Acceso al Sistema: Recuperar Contraseña	1. En la pantalla de inicio, haz clic en "¿Olvidaste tu contraseña?".	Panel de Administración (Solo Admins)	Accede a un panel especial para ver métricas y gráficos de rendimiento de los proyectos (RF13, RF14) y para gestionar usuarios y sus permisos (crear, editar, eliminar, asignar roles). (RF15, RF16, RF17)
	2. Ingresa tu correo para recibir instrucciones.	Notificaciones	Recibirás alertas en tiempo real cuando te asignen una tarea, te mencionen, una tarea cambie de estado o recibas una invitación a un proyecto. (RF9)
Interfaz: Pantalla Principal	Al iniciar sesión, verás el "Dashboard de Proyectos", que muestra una lista de todos tus proyectos. Desde aquí puedes acceder a ellos o crear uno nuevo.	Configuración de Perfil: Editar Datos	1. Ve a "Mi Perfil" desde el panel de navegación.
Interfaz: Panel de Navegación	Menú lateral para acceder a: Proyectos, Mi Perfil, Panel de Administración (si eres admin) y Cerrar Sesión.		2. Cambia tu nombre y foto de perfil.
Gestión de Proyectos: Crear	1. En el dashboard, haz clic en "Crear Nuevo Proyecto".	Configuración de Perfil: Cambiar Contraseña	1. Ve a "Mi Perfil", sección "Seguridad".
	2. Rellena el nombre, descripción y fecha de inicio.		2. Ingresa tu contraseña actual y la nueva para confirmar.
	3. Haz clic en "Crear Proyecto". (RF1)	¿Olvidé mi contraseña?:	Usa la opción de recuperación en la pantalla de login.
Gestión de Proyectos: Invitar Miembros	1. Dentro de un proyecto, ve a la sección "Miembros".	Preguntas Frecuentes (FAQ)	¿Cómo invito a alguien?: Desde la sección "Miembros" dentro de un proyecto.
	2. Ingresa el correo electrónico de tus compañeros para enviarles una invitación.		¿No veo el panel de admin?: Es solo para usuarios con rol de "Administrador".
Gestión de Proyectos: Editar / Eliminar	1. En el dashboard, busca el proyecto.	Consejos y Buenas Prácticas	Mantén el tablero Kanban actualizado, sé específico en las descripciones de las tareas y usa las menciones @ para una comunicación efectiva.
	2. Haz clic en el menú de opciones (...) y selecciona "Editar" o "Eliminar". (RF2, RF3)	Soporte y Reporte de Errores	Para dudas o reportar un error, contacta a <a href="mailto:75142209@continental.edu.pe">75142209@continental.edu.pe</a> . Si reportas un error, incluye una descripción detallada, los pasos para reproducirlo y una captura de pantalla.

### DESCARGAR MANUAL COMPLETO :

[https://docs.google.com/document/d/1vTgtF\\_C7skj\\_OVvWq6OuPl8AZlabGgh5DTjHFvZ9iDE/edit?usp=sharing](https://docs.google.com/document/d/1vTgtF_C7skj_OVvWq6OuPl8AZlabGgh5DTjHFvZ9iDE/edit?usp=sharing)

## Anexo 03. Pruebas

### Unitaria

```
PASS  tests/auth.controller.test.js
Auth Controller
  registerUser
    ✓ debería registrar un nuevo usuario (174 ms)
    ✓ debería fallar si el correo ya está registrado (18 ms)
    ✓ debería fallar si faltan campos (10 ms)
  loginUser
    ✓ debería hacer login correctamente (128 ms)
    ✓ debería fallar si el usuario no existe (73 ms)
    ✓ debería fallar si la contraseña es incorrecta (142 ms)

Test Suites: 1 passed, 1 total
Tests:       6 passed, 6 total
Snapshots:   0 total
Time:        2.48 s
Ran all test suites.
```

```
● PS C:\Users\yocju\Desktop\Nueva carpeta (6)\PGP-IA-TP2\backend> npm test

> backend@1.0.0 test
> cross-env NODE_ENV=test jest --detectOpenHandles --forceExit

PASS  tests/project.controller.test.js
POST /api/projects
  ✓ debería crear un proyecto nuevo (133 ms)
  ✓ debería fallar sin token de autenticación (10 ms)

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
Snapshots:   0 total
Time:        2.178 s, estimated 3 s
Ran all test suites.
```

```
> backend@1.0.0 test
> jest

PASS  tests/card.controller.test.js
Más tests para card.controller
  asignarMiembros
    ✓ debería retornar 400 si miembros no es array (4 ms)
    ✓ debería retornar 404 si no encuentra tarjeta
    ✓ debería retornar 400 si alguno no es miembro del proyecto
    ✓ debería asignar miembros correctamente (1 ms)
  agregarChecklistItem
    ✓ debería retornar 400 si no envian nombre
    ✓ debería retornar 404 si no encuentra tarjeta
    ✓ debería agregar item al checklist
  actualizarChecklistItem
    ✓ debería retornar 404 si no encuentra tarjeta (1 ms)
    ✓ debería retornar 400 si el índice del checklist no existe (1 ms)
    ✓ debería actualizar ítem del checklist
  eliminarChecklistItem
    ✓ debería retornar 404 si no encuentra tarjeta
    ✓ debería retornar 400 si índice no existe
    ✓ debería eliminar ítem del checklist

Test Suites: 1 passed, 1 total
Tests:       13 passed, 13 total
Snapshots:   0 total
Time:        1.203 s, estimated 2 s
Ran all test suites.
```

```
● PS C:\Users\yocju\Desktop\test-list.controller\PGP-IA-TP2\backend> npm test

> backend@1.0.0 test
> jest

PASS  tests/list.controller.test.js
List Controller
  clearLista
    ✓ debería devolver error 400 si no se proporciona nombre (3 ms)
    ✓ debería devolver error 404 si el proyecto no existe (1 ms)
    ✓ debería devolver error 403 si el usuario no es miembro del proyecto
    ✓ debería crear una lista exitosamente (1 ms)

PASS  tests/list.controller.test.js
PASS  tests/list.controller.test.js
List Controller
PASS  tests/list.controller.test.js
PASS  tests/list.controller.test.js
PASS  tests/list.controller.test.js
PASS  tests/list.controller.test.js
List Controller
```

```
● PS C:\Users\yocju\Desktop\test-list.controller\PGP-IA-TP2\backend> npm test

> backend@1.0.0 test
> jest

PASS tests/list.controller.test.js
List Controller
  clearLista
    ✓ debería devolver error 400 si no se proporciona nombre (3 ms)
    ✓ debería devolver error 404 si el proyecto no existe (1 ms)
    ✓ debería devolver error 403 si el usuario no es miembro del proyecto
    ✓ debería crear una lista exitosamente (1 ms)

PASS tests/list.controller.test.js
PASS tests/list.controller.test.js
List Controller
PASS tests/list.controller.test.js
List Controller
PASS tests/list.controller.test.js
PASS tests/list.controller.test.js
PASS tests/list.controller.test.js
PASS tests/list.controller.test.js
List Controller
PASS tests/list.controller.test.js
List Controller
```

```
Windows PowerShell
 34     } catch (error) {
 35         console.error(error);
 36         ^
 37         res.status(500).json({ msg: 'Error al guardar mensaje' });
 38     };
at error (controllers/message.controller.js:35:13)
at Object.<anonymous> (tests/message.controller.test.js:81:5)

PASS  tests/message.controller.test.js
  guardarMensaje
    ✓ debería retornar 400 si faltan datos (3 ms)
    ✓ debería retornar 404 si no existe el proyecto (1 ms)
    ✓ debería retornar 403 si el usuario no es miembro
    ✓ debería guardar el mensaje si todo está bien (1 ms)
    ✓ debería retornar 500 si hay un error interno (19 ms)

Test Suites: 1 passed, 1 total
Tests:       5 passed, 5 total
Snapshots:   0 total
Time:        1.083 s, estimated 2 s
Ran all test suites.
```

