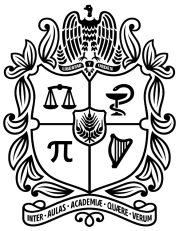


UNIVERSIDAD
NACIONAL
DE COLOMBIA

Universidad Nacional de Colombia - Sede Bogotá

Facultad de Ingeniería

Departamento de Ingeniería de Sistemas e Industrial

Asignatura:

Ingeniería de Software I

Proyecto - Taller Requerimientos y Clean Code

Natalia Carolina Bautista Sanchez
Daniel Santiago Cocinero Jimenez
Baruj Vladimir Ramirez Escalante
Juan Daniel Ramirez Mojica

Profesor:

Oscar Eduardo Alvarez Rodriguez

2024 - II

Bogotá D.C.

Índice

Selección del Proyecto a Desarrollar

Para el desarrollo del proyecto, se decidió trabajar en la propuesta “Club de Lectura Digital”, seleccionado mediante un proceso iterativo de evaluación y priorización.

A partir de una lluvia de ideas inicial, surgieron doce propuestas, las cuales nacieron tanto de las necesidades observadas en nuestras experiencias personales como de nuestros hobbys y del deseo de realizar algo diferente y aprender nuevas herramientas, o incluso de una combinación de ambas.

Para el proceso de consenso dentro del equipo, se adaptó una herramienta llamada **“Selección de cartas”** con el propósito de evaluar y priorizar las ideas. El procedimiento para utilizar esta herramienta es el siguiente:

1. Escriba cada una de las opciones en tarjetas individuales.
2. Cuente el número total de opciones, ya que este será el número máximo de puntuación.
3. Cada miembro del equipo selecciona la opción que menos le convenza, asignándole el número 1, y la opción que más le guste, asignándole el número máximo. Las demás opciones se clasificarán según esta misma regla de puntuación.
4. Pondere los puntajes de todos los miembros del equipo.
5. Realice una breve discusión con el equipo, donde se compartan puntos de vista y se clarifiquen las razones detrás de las preferencias y descartes.
6. Elimine las opciones con los puntajes más bajos.
7. Repita el proceso de manera iterativa hasta que quede una única opción.

De esta manera, cada integrante otorgó puntajes para destacar las opciones más relevantes. En cada iteración, las ideas se discutieron y redujeron progresivamente, concentrándose solo en aquellas con mayor potencial. Este enfoque permitió refinar las alternativas y alcanzar un consenso grupal, culminando en la elección del “Club de Lectura Digital" como la propuesta más adecuada. Puede consultar los resultados de este proceso en el adjunto número **uno** y número **dos**.

Levantamiento de Requerimientos

Contexto

El proyecto surge de la propuesta de un integrante del equipo, que observó una oportunidad de negocio en base a uno de sus hobbies, el objetivo de esta propuesta es crear una plataforma digital para fomentar la lectura colectiva y la discusión de libros. Esta iniciativa **busca unir a personas con diversos intereses en un espacio virtual donde puedan compartir experiencias lectoras, debatir ideas y conectarse a través de su amor por los libros.** Para esto se puede usar las plataformas digitales que tienen la capacidad de superar ciertas barreras tradicionales que enfrentan los clubes de lectura, tales como **la dificultad para coordinar horarios, la falta de espacios físicos accesibles para todos los interesados y las limitaciones para conectar a personas con intereses similares**, especialmente en comunidades geográficamente dispersas.

Resumen Inicial del Problema

Se enfrenta el desafío de diseñar y desarrollar una plataforma intuitiva y accesible que permita a los usuarios crear, gestionar y participar en clubes de lectura en línea. Además, se requiere un sistema que facilite las discusiones, el seguimiento del progreso de lectura y la toma de decisiones colectivas, como la selección de nuevos libros. La solución debe garantizar una experiencia fluida y atractiva para los usuarios, promoviendo la participación activa y el sentido de comunidad.

Los usuarios potenciales del sistema esperan poder compartir sus ideas y participar en discusiones organizadas por temas, así como contar con un sistema que les permita decidir el libro a leer (como lo podría ser un sistema de votación). También esperan una plataforma que permita realizar un seguimiento del progreso de lectura y se genere una experiencia participativa que fomente la interacción entre los miembros y fortalezca el sentido de comunidad.

Beneficios para cada integrante

Juan Daniel Ramírez Mojica: A través de este proyecto, espero aprender a utilizar nuevas herramientas que aún no he explorado en profundidad, como la creación de un chat y el uso de bases de datos no relacionales. Del mismo modo, me gustaría poner en práctica los conocimientos que he adquirido en distintas clases en un proyecto más aproximado a la realidad, especialmente en temas recientes como clean code, patrones de diseño y testing, ya que son conceptos relativamente nuevos para mí.

Natalia Carolina Bautista Sanchez: En este proyecto busco aprender a hacer un buen desarrollo frontend, aprendiendo a hacer interfaces que sean estéticas y al mismo tiempo fáciles de utilizar para los usuarios, dándoles una experiencia agradable e intuitiva. Quisiera aprender en el futuro acerca de desarrollo UI/Ux, este proyecto me proporciona unas buenas bases para comenzar a aprender del tema.

Daniel Santiago Cocinero Jimenez: Mi objetivo con este proyecto es aprender a utilizar Spring Boot, entender cómo implementar una base de datos no relacional y como es el desarrollo web, puesto que nunca he hecho un backend y me gustaria entender como se conecta con el front.

Baruj Vladimir Ramirez Escalante: Con este proyecto, buscó implementar técnicas para el diseño y desarrollo de software, principalmente el desarrollo con IA, pues parece ser que de aquí a futuro se verá cada vez más presente en el desarrollo de proyectos.

Restricciones y Alcance del Proyecto

- **Tiempo:** El plazo con el que se cuenta es de alrededor de un mes, ya que se debe entregar el día 6 de marzo del 2025.
- **Presupuesto: $16.132.744**. Esto se verá reflejado más adelante.
- **Recursos:** El equipo de desarrollo está compuesto por 4 desarrolladores full-stack.
- **Alcance:** Dado que el tiempo y los recursos son limitados, se priorizará la creación de un Producto Mínimo Viable (MVP), enfocándose únicamente en las funcionalidades esenciales que permitan lanzar una versión básica pero funcional.

Impacto en el Negocio

Este proyecto tiene el potencial de generar un impacto en diversos aspectos: En términos de beneficios, se espera incrementar el “engagement” de los usuarios, al ofrecer una experiencia interactiva y en cierto modo personalizada que fomente una participación activa.

Siguiendo con el punto anterior, desde un punto de vista más económico, se ve potencial adquiriendo posible financiación por parte de entidades públicas como el ministerio de cultura, con el que se fomenten hábitos de lectura; del mismo modo que se podrían encontrar oportunidades comerciales, como alianzas con editoriales.

Sin embargo, también se identifican riesgos asociados al comportamiento del mercado, la adopción por parte de los usuarios y la competencia de plataformas similares (cómo podría ser goodreads, aunque se podría realizar una alianza), estos aspectos se deben manejar de manera cuidadosa para garantizar el éxito de este proyecto.

Viabilidad Técnica

Este proyecto se considera técnicamente viable debido a la experiencia y habilidades del equipo de desarrollo, compuesto por cuatro desarrolladores full-stack. El equipo cuenta con conocimientos sólidos en Java, JavaScript, HTML, CSS, bases de datos relacionales y desarrollo backend, lo que garantiza una base técnica adecuada para el proyecto. Sin embargo, aspectos como la integración de bases de datos no relacionales y la implementación de ciertas funcionalidades específicas requerirán tiempo adicional de investigación y pruebas.

En términos de tecnología, se dispone de herramientas modernas que pueden facilitar el desarrollo, como Spring Boot para la construcción del backend, bases de datos no relacionales (MongoDB), así como APIs externas para ampliar la funcionalidad del sistema.

A pesar de esto, se deben considerar posibles obstáculos, como la curva de aprendizaje de nuevas tecnologías y los desafíos asociados con la integración de distintos servicios y herramientas dentro del proyecto.

Levantamiento de Requerimientos

Para llevar a cabo este proyecto, se planteó inicialmente realizar una entrevista como método de recopilación de información. Los miembros del grupo se reunieron a través de una videollamada por Google Meet, durante la cual discutieron las ideas que cada uno tenía sobre la aplicación. Durante la reunión, se logró un consenso general acerca de las características y funcionalidades principales de la aplicación, al tiempo que se generaron los mock ups iniciales de la aplicación, mientras la llamada era grabada para fines de documentación.

Posteriormente, la grabación fue procesada mediante el modelo de transcripción Whisper, lo que permitió generar un texto transcrito de la reunión. Esta transcripción se utilizó como entrada en ChatGPT, con el objetivo de extraer una lista de requerimientos basada en el consenso alcanzado por el grupo.

La transcripción de la videollamada como los mockups iniciales, se añaden como anexo **tres** y **cuatro respectivamente**.

Cómo resultado, se obtuvieron los siguientes requerimientos:

Requerimientos Funcionales

Registro e inicio de sesión: Registro de nuevos usuarios mediante formulario (nombre, correo electrónico, contraseña).

- Inicio de sesión para usuarios registrados.
- Recuperación de contraseña mediante correo electrónico.
- Almacenamiento seguro de contraseñas usando algoritmos de cifrado (ej: bcrypt).

Exploración de clubes: La plataforma debe permitir a los usuarios explorar y buscar clubes que les interesen y de los que quieran hacer parte.

- Listado de clubes destacados o populares en la página principal.
- Barra de búsqueda para encontrar clubes.
- Sugerencias de clubes basadas en las preferencias del usuario.
- Visualización de detalles del club antes de unirse (nombre, descripción, libro actual, imagen).
- Opción para unirse a clubes públicos o privados.

Gestión de clubes: La plataforma debe permitir que los usuarios puedan crear y gestionar clubes de acuerdo a las características que deseen.

- Crear un nuevo club.
- Configuración del libro actual a leer en el club.
- Visualización del historial de libros leídos.
- Edición de información del club (solo por el administrador del club).
- Eliminación del club (solo por el administrador del club).

Participación en clubes: La plataforma debe permitir la interacción entre miembros de un club. Los usuarios podrán debatir en secciones específicas, crear temas de discusión y votar por el próximo libro.

- Visualización de información del club (Nombre, Imagen, Libro actual, Descripción, Avisos importantes).
- Espacio de debates con secciones para discutir.

- Posibilidad de crear temas de discusión con aprobación del administrador.
- Comentarios y respuestas dentro de los temas.
- Sistema de votación para decidir el próximo libro a leer.
- Los usuarios pueden marcar si han terminado el libro que se está leyendo actualmente.
- Visualización del porcentaje de usuarios que han completado el libro.

Notificaciones: La plataforma debe informar a los usuarios sobre cambios en el libro, nuevos temas de discusión y resultados de votaciones.

- Notificaciones sobre cambios en el libro, nuevos temas de discusión y resultados de votaciones.
- Configuración de preferencias de notificaciones (activadas/desactivadas por usuario).
- Envío de notificaciones por correo electrónico.

Gestión de roles: La plataforma debe permitir la asignación de roles con diferentes permisos. Los administradores tendrán control sobre la gestión del club, mientras que los miembros regulares podrán participar en la visualización, comentarios y votaciones.

- Rol de administrador con capacidad de crear y editar temas de discusión, aprobar temas propuesto por miembros, editar información del club y moderar miembros.
- Roles básicos para miembros regulares (visualizar, comentar, votar).

Perfil de usuario: La plataforma debe permitir que el usuario gestione la información de su perfil que desea compartir a los demás.

- Visualización y edición del perfil personal
- Mostrar un historial de libros leídos o géneros favoritos.

Requerimientos No Funcionales

Seguridad y privacidad: La plataforma debe garantizar la seguridad de los datos mediante el cifrado de contraseñas y datos sensibles. Los clubes podrán configurarse como privados, con contraseñas seguras, y se deberán aplicar restricciones de acceso a funcionalidades avanzadas según el rol del usuario.

Diseño Intuitivo: : La plataforma debe contar con un diseño intuitivo que permita al usuario utilizar de manera fácil y eficiente; sin confundirlo ni sobrecargarlo con opciones innecesarias. Además, debe tener un diseño claro y accesible.

Rendimiento: La plataforma debe ser capaz de manejar múltiples solicitudes de usuarios de manera eficiente, garantizando tiempos de respuesta rápidos y un rendimiento óptimo.

Análisis de Requerimientos y Clasificación con MoSCoW

Preguntas que surgieron:

- ¿Cúal es el número máximo de miembros que puede tener un club?
- ¿Cómo se van a realizar las notificaciones: solo por correo electrónico o también en la página?
- ¿Qué criterios se utilizará para mostrar el listado de clubes destacados o populares en la página principal?
    - ¿Cómo se mide la popularidad de un club? ¿Con el número de miembros?
    - ¿Habrá un sistema de rotación de estos clubes?
- ¿Cómo se definirá el acceso a los clubes privados: solo por contraseña o también usando el modo de invitación por link?
- ¿El sistema de votación para decidir el próximo libro será anónimo o visible para todos los miembros del club?
- ¿Qué tanto podrán los usuarios personalizar su perfil? ¿Qué información podrán añadir?
- ¿Existirá solo un administrador por club o podrían haber varios?
- ¿Cómo podría abandonar un miembro un club si este es el administrador y no existe más de uno? ¿Debe asignar un nuevo administrador antes de abandonar el club?
- ¿Cómo se gestionará el espacio de debate para evitar contenido inapropiado o fuera de contexto?

Clasificación MoSCoW:

Recordando, el método MoSCoW clasifica los requerimientos en:

1. Must Have (Imprescindibles): Necesarios para que el sistema funcione.
2. Should Have (Importantes): Aportan valor significativo pero no son críticos.
3. Could Have (Deseables): Mejoran la experiencia pero no son esenciales.
4. Won't Have (No incluir ahora): No se implementarán en esta versión.

Requerimientos Clasificados:

Análisis de requerimientos							
Must Have		**Should Have**		**Could Have**		**Won't have**	
MH01	Registro de nuevos usuarios	SH01	Almacenamiento seguro de contraseñas (Cifrado)	CH01	Recuperación de contraseña mediante correo electrónico	WH01	Sugerencias de clubes basadas en las preferencias del usuario
MH02	Inicio de sesión para usuarios registrados	SH02	Visualización de detalles del club	CH02	Barra de búsqueda para encontrar clubes	WH02	Indicador de progreso de lectura
MH03	Listado de clubes populares en la página principal	SH03	Permitir a los usuarios crear un nuevo club	CH03	Visualización detallada de la información del club (después de unirse)	WH03	Aprobar temas propuestos por miembros (administrador del club)
MH04	Permitir a los usuarios unirse a clubes	SH04	Eliminación del club (solo por el administrador del club)	CH04	Sistema de votación para decidir el próximo libro a leer.	WH04	Visualización del historial de libros leídos en el club
MH05	Permitir a los usuarios abandonar clubes	SH05	Edición de información del club (solo por el administrador del club)	CH05	Notificaciones	WH05	Mostrar un historial de libros leídos o géneros favoritos en el perfil
MH06	Configuración del libro actual a leer en el club			CH06	Crear y editar temas de discusión (administrador del club)		
MH07	Espacios de debate			CH07	Visualización y edición del perfil personal		
MH08	División de roles, usuario y administrador			CH08	Configuración de clubes privados con contraseñas		

Tabla 1. Requerimientos Clasificados de Book Hunting.

Estimación

Para el proceso de estimación se utilizará la metodología "Planning Poker" basada en la sucesión de Fibonacci, haciendo uso de la herramienta https://planningpokeronline.com/.

Tras haber analizado los resultados obtenidos en el "Planning Poker" y haber comunicado las razones por las cuales habíamos dado cada una de las votaciones, este fue el consenso al que se llegó:

Must Have:

Must Have				
ID	**Requerimiento**	**Tiempo Estimado**	**Complejidad**	**Justificación**
MH01	Registro de nuevos usuarios	1 día	Baja	Es un proceso bastante usual
MH02	Inicio de sesión para usuarios registrados	1 día	Baja	También es un proceso bastante usual
MH03	Listado de clubes populares en la página principal	2 días	Media	Se requiere la adición de los clubes a la base de datos y una UI más definida
MH04	Permitir a los usuarios unirse a clubes	1 día	Baja	No requiere demasiada lógica en la base de datos
MH05	Permitir a los usuarios abandonar clubes	1 día	Baja	Tampoco requiere demasiada lógica en la base de datos
MH06	Configuración del libro actual a leer en el club	1 día	Baja	Es una modificación pequeña de la UI del club
MH07	Espacios de debate	3 días	Alta	Requiere implementar un sistema de chat tanto en front como en back
MH08	División de roles, usuario y administrador	2 días	Media	Se requiere bastante lógica de permisos y UI con opciones diferentes

Tabla 2. Estimación Requerimientos Must Have de Book Hunting.

Should Have:

Should Have				
ID	**Requerimiento**	**Tiempo Estimado**	**Complejidad**	**Justificación**
SH01	Almacenamiento seguro de contraseñas (Cifrado)	1 día	Baja	Se tiene un integrante con experiencia en el área
SH02	Visualización de detalles del club (antes de unirse)	1 día	Baja	No requiere demasiada lógica en la consulta a la base de datos
SH03	Permitir a los usuarios crear un nuevo club	2 días	Media	Se debe implementar un nuevo rol al usuario y hacer modificaciones en la base de datos
SH04	Eliminación del club (solo por el administrador del club)	1 día	Baja	No requiere demasiada lógica en la base de datos
SH05	Edición de información del club (solo por el administrador del club)	1 día	Baja	No es una modificación muy grande en la base de datos

Tabla 3. Estimación Requerimientos Should Have de Book Hunting.

Could Have:

Could Have				
ID	**Requerimiento**	**Tiempo Estimado**	**Complejidad**	**Justificación**
CH01	Recuperación de contraseña mediante correo electrónico	3 días	Alta	Requiere buscar un sistema para el soporte de emails
CH02	Barra de búsqueda para encontrar clubes	2 días	Media	Se debe crear varias consultas diferentes en la base de datos
CH03	Visualización detallada de la información del club (después de unirse)	1 día	Baja	No requiere demasiada lógica en la consulta a la base de datos
CH04	Sistema de votación para decidir el próximo libro a leer	2 días	Media	Es un elemento con el que varios usuarios interaccionan y es importante en la experiencia de estos
CH05	Notificaciones	3 días	Alta	Requiere notificaciones por email y no se tienen conocimientos técnicos sobre

				esto
CH06	Crear y editar temas de discusión (administrador del club)	3 días	Alta	Requiere implementar un sistema de chat más dinámico
CH07	Visualización y edición del perfil personal	1 día	Baja	No requiere demasiada lógica en la base de datos y en las consultas a esta
CH08	Configuración de clubes privados con contraseñas	1 día	Baja	No requiere mucha lógica de programación ni en la base de datos

Tabla 4. Estimación Requerimientos Could Have de Book Hunting.

Won’t Have:

Won't Have				
ID	**Requerimiento**	**Tiempo Estimado**	**Complejidad**	**Justificación**
WH01	Sugerencias de clubes basadas en las preferencias del usuario	4 días	Alta	Requiere algoritmos de recomendación y lógica difícil
WH02	Indicador de progreso de lectura	1 día	Baja	Solo es hacer una modificación en la base de datos y UI
WH03	Aprobar temas propuestos por miembros (administrador del club)	1 día	Baja	No es complejo de implementar en la base de datos, un poco más complejo en UI
WH04	Visualización del historial de libros leídos en el club	2 días	Media	Requiere implementar trazabilidad y consultas en la base de datos
WH05	Mostrar un historial de libros leídos o géneros favoritos en el perfil	2 días	Media	Parecido a un campo "sobre mí", no requiere demasiada complejidad de implementación

Tabla 5. Estimación Requerimientos Won’t Have de Book Hunting.

Análisis de Gestión de Software

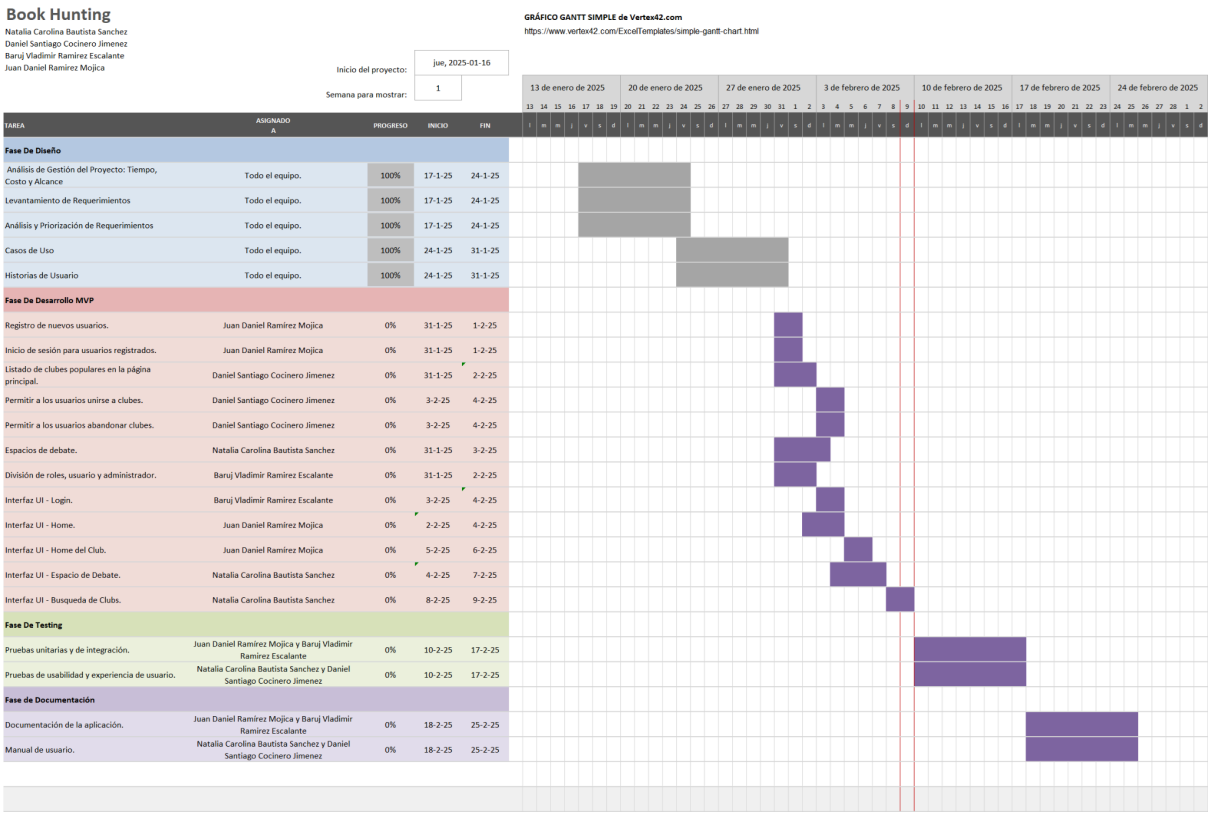
Tiempo

El desarrollo del proyecto se dividirá en las siguientes etapas:

- **Diseño (2 semanas)**
- **Desarrollo (1 semana)**
- **Testing (1 semanas)**
- **Documentación (1 semana)**

Total estimado: 4 semanas

Para organizar el tiempo de desarrollo, se utilizó un diagrama de Gantt en el que las tareas fueron distribuidas entre los integrantes y se asignaron intervalos de tiempo específicos para cada una. Esto se refleja en la siguiente figura. Para una mejor visualización, consulte el adjunto número [5] o acceda al siguiente enlace. [5. Diagrama De Gantt.pdf - Google Drive]



Book Hunting

Natalia Carolina Bautista Sanchez
Daniel Santiago Cocinero Jimenez
Baruj Vladimir Ramirez Escalante
Juan Daniel Ramirez Mojica

GRÁFICO GANTT SIMPLE de Vertex42.com
https://www.vertex42.com/ExcelTemplates/simple-gantt-chart.html

Inicio del proyecto: jue, 2025-01-16
Semana para mostrar: 1

TAREA	ASIGNADO A	PROGRESO	INICIO	FIN
Fase De Diseño				
Análisis de Gestión del Proyecto: Tiempo, Costo y Alcance	Todo el equipo.	100%	17-1-25	24-1-25
Levantamiento de Requerimientos	Todo el equipo.	100%	17-1-25	24-1-25
Análisis y Priorización de Requerimientos	Todo el equipo.	100%	17-1-25	24-1-25
Casos de Uso	Todo el equipo.	100%	24-1-25	31-1-25
Historias de Usuario	Todo el equipo.	100%	24-1-25	31-1-25
Fase De Desarrollo MVP				
Registro de nuevos usuarios.	Juan Daniel Ramirez Mojica	0%	31-1-25	1-2-25
Inicio de sesión para usuarios registrados.	Juan Daniel Ramirez Mojica	0%	31-1-25	1-2-25
Listado de clubes populares en la página principal.	Daniel Santiago Cocinero Jimenez	0%	31-1-25	2-2-25
Permitir a los usuarios unirse a clubes.	Daniel Santiago Cocinero Jimenez	0%	3-2-25	4-2-25
Permitir a los usuarios abandonar clubes.	Daniel Santiago Cocinero Jimenez	0%	3-2-25	4-2-25
Espacios de debate.	Natalia Carolina Bautista Sanchez	0%	31-1-25	3-2-25
División de roles, usuario y administrador.	Baruj Vladimir Ramirez Escalante	0%	31-1-25	2-2-25
Interfaz UI - Login.	Baruj Vladimir Ramirez Escalante	0%	3-2-25	4-2-25
Interfaz UI - Home.	Juan Daniel Ramirez Mojica	0%	2-2-25	4-2-25
Interfaz UI - Home del Club.	Juan Daniel Ramirez Mojica	0%	5-2-25	6-2-25
Interfaz UI - Espacio de Debate.	Natalia Carolina Bautista Sanchez	0%	4-2-25	7-2-25
Interfaz UI - Busqueda de Clubs.	Natalia Carolina Bautista Sanchez	0%	8-2-25	9-2-25
Fase De Testing				
Pruebas unitarias y de integración.	Juan Daniel Ramirez Mojica y Baruj Vladimir Ramirez Escalante	0%	10-2-25	17-2-25
Pruebas de usabilidad y experiencia de usuario.	Natalia Carolina Bautista Sanchez y Daniel Santiago Cocinero Jimenez	0%	10-2-25	17-2-25
Fase de Documentación				
Documentación de la aplicación.	Juan Daniel Ramirez Mojica y Baruj Vladimir Ramirez Escalante	0%	18-2-25	25-2-25
Manual de usuario.	Natalia Carolina Bautista Sanchez y Daniel Santiago Cocinero Jimenez	0%	18-2-25	25-2-25

13 de enero de 2025 | 20 de enero de 2025 | 27 de enero de 2025 | 3 de febrero de 2025 | 10 de febrero de 2025 | 17 de febrero de 2025 | 24 de febrero de 2025

Figura 1. Diagrama de Gantt del proyecto Book Hunting.

Costo

Para esta estimación, se consideraron los siguientes aspectos:

- **Jornada laboral:** Los empleados trabajarán a tiempo completo. Dado que el proyecto se finalizará antes del 15 de julio de 2025, se tomará en cuenta lo estipulado por la Ley 2101 de 2021, que establece una jornada laboral máxima legal de 46 horas semanales.
- **Modalidad de trabajo:** Dado que el trabajo será remoto, no se contempla el auxilio de transporte. Sin embargo, según el artículo 10 de la Ley 2088 de 2021, en su inciso segundo, se dispone el pago de un auxilio de conectividad.

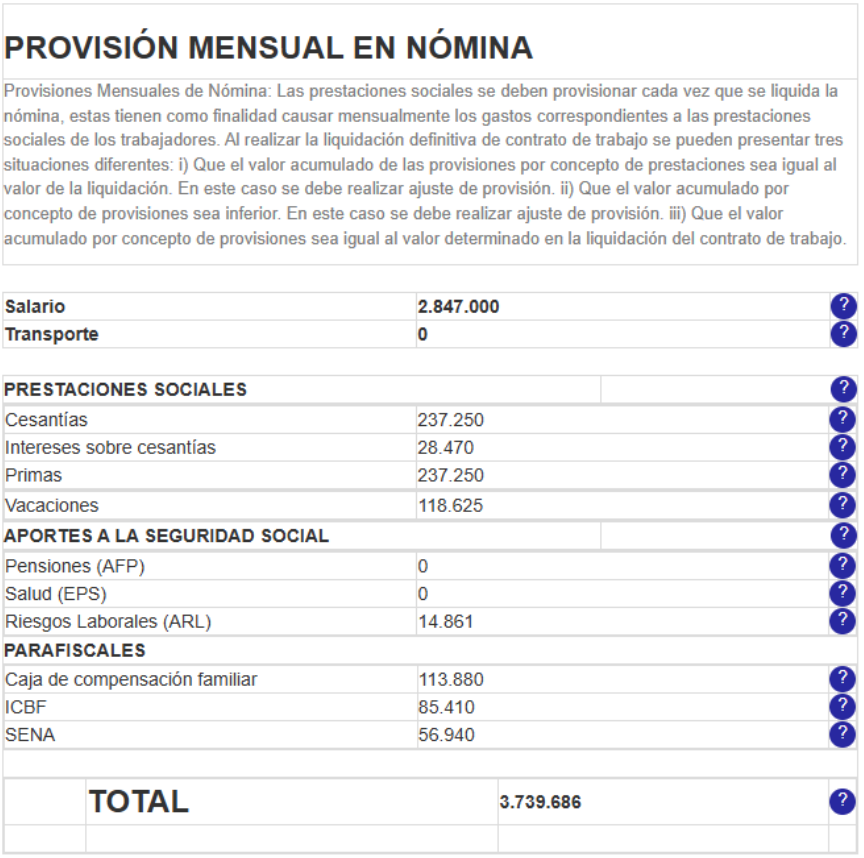
  Dicho artículo establece que:

  - Para los servidores públicos, el auxilio de conectividad se reconocerá en los términos y condiciones establecidos para el auxilio de transporte.
  - Para los trabajadores del sector privado, el valor establecido para el auxilio de transporte se reconocerá como auxilio de conectividad digital y tendrá los mismos efectos salariales.

  Por lo tanto, se adicionará un auxilio de conectividad de $200.000 pesos mensuales.
- **Salario base:**
  - Se utilizará el salario mínimo establecido para el año 2025, conforme al artículo 1 del Decreto 1572 de 2024, que establece un salario mínimo de **$1.423.500 pesos mensuales**, con vigencia a partir del 1 de enero de 2025. A continuación, se presenta el decreto completo: Decreto 1572 de 2024 - Gestor Normativo - Función Pública.
  - De acuerdo con la página “talent.com”, el salario medio para un desarrollador junior es de **$2.500.000** (dos millones quinientos mil pesos). Por lo tanto, se considerará que a cada desarrollador se le asignan 2 salarios mínimos legales vigentes, lo que resulta en un salario de **$2.847.000** (dos millones ochocientos cuarenta y siete mil pesos) por cada desarrollador. A continuación encuentra la página: Salario medio para Desarrollador Junior en Colombia 2025
- **Cálculo de costos laborales:** El cálculo del costo de los empleados se realizará utilizando la calculadora laboral oficial presentada por el Ministerio del Trabajo.

A continuación, se presenta el decreto completo: [MinTrabajo Calculadora Laboral]

- Se obtiene que para el empleador, el costo de contratar a un desarrollador por un mes, es de **$3.739.686.** Al cual se le debe adicionar el auxilio de conectividad, obteniendo un total de **$3.939.686**



PROVISIÓN MENSUAL EN NÓMINA

Provisiones Mensuales de Nómina: Las prestaciones sociales se deben provisionar cada vez que se liquida la nómina, estas tienen como finalidad causar mensualmente los gastos correspondientes a las prestaciones sociales de los trabajadores. Al realizar la liquidación definitiva de contrato de trabajo se pueden presentar tres situaciones diferentes: i) Que el valor acumulado de las provisiones por concepto de prestaciones sea igual al valor de la liquidación. En este caso se debe realizar ajuste de provisión. ii) Que el valor acumulado por concepto de provisiones sea inferior. En este caso se debe realizar ajuste de provisión. iii) Que el valor acumulado por concepto de provisiones sea igual al valor determinado en la liquidación del contrato de trabajo.

Salario	**2.847.000**
Transporte	**0**
PRESTACIONES SOCIALES	
Cesantías	237.250
Intereses sobre cesantías	28.470
Primas	237.250
Vacaciones	118.625
APORTES A LA SEGURIDAD SOCIAL	
Pensiones (AFP)	0
Salud (EPS)	0
Riesgos Laborales (ARL)	14.861
PARAFISCALES	
Caja de compensación familiar	113.880
ICBF	85.410
SENA	56.940
TOTAL	**3.739.686**

Figura 2. Cálculo costo mensual de nómina del proyecto Book Hunting.

Dado a que se cuenta con cuatro desarrolladores, se obtiene que el costo total por los 4 desarrolladores es de: **$15.758.744.**

Para el presupuesto de despliegue se tomará en cuenta el backend en Java con Spring Boot, el frontend en React.js, y bases de datos MongoDB. Esta estimación asume un uso moderado de recursos y tráfico.

Se hace el cálculo para el despliegue en AWS:

- **Costo Total Estimado: Aproximadamente $89 USD/mes**

  A conversión de hoy (31/01/2025): **$374.000** pesos colombianos

Estimación de Costo de Personal de Desarrollo					
Roles de Desarrollador	**Número de Desarrolladores**	**Costo por Desarrollador - COP/mes**		**Costo Mensual Total - COP**	
Desarrollador Junior	4	$3.939.686		$15.758.744	
Estimación de Costo de Servicios					
Servicio	**Tipo / Especificación**	**Uso**	**Almacenamiento**	**Transferencia de Datos**	**Costo Estimado - USD/mes**
Instancia de Computación (Servidor Backend)	Instancia t3.medium (2 vCPU, 4 GB RAM)	730 horas/mes (24/7)	Incluido en la instancia	N/A	$37
Instancia de Computación (Servidor Frontend)	Instancia t3.small (2 vCPU, 2 GB RAM)	730 horas/mes (24/7)	Incluido en la instancia	N/A	$18
Base de Datos (MongoDB o MySQL)	Amazon RDS db.t3.micro (1 vCPU, 1 GB RAM)	Facturación por horas según uso	20 GB	Según uso	$15
Almacenamiento de Objetos (para imágenes y otros archivos estáticos)	Amazon S3	N/A	50 GB	50 GB	$10
Transferencia de Datos Saliente	Asumiendo 100 GB de transferencia de datos saliente	N/A	N/A	100 GB	$9
Total Estimación Proyecto					
Total Estimación de Costo de Personal de Desarrollo			$15.758.744 COP/mes		
Total Estimación de Costo de Servicios			$89 USD/mes	$374.000 COP/mes	
Total de Estimación de Costos			$16.132.744 COP/mes		

Tabla 6. Cálculo costo total mensual del proyecto Book Hunting.

Lo que nos da un total de presupuesto para el proyecto de **$16.132.744** pesos colombianos.

Alcance

Dado el tiempo limitado para el desarrollo del proyecto, es fundamental enfocarnos en entregar un producto funcional y estable dentro del plazo establecido. Las

funcionalidades clasificadas como **Must Have** representan el núcleo esencial de la plataforma, permitiendo a los usuarios registrarse, unirse y participar en clubes de lectura, y a los administradores gestionar los clubes de manera básica.

Incluir funcionalidades adicionales de las categorías **Should Have** y **Could Have** implicaría un aumento en la complejidad del desarrollo, lo que podría comprometer la calidad, estabilidad y entrega del producto dentro del tiempo estipulado. Al centrarnos en el **Must Have**, garantizamos que la aplicación cumpla con su propósito principal y tenga una base sólida para futuras mejoras en iteraciones posteriores.

Diseño y Arquitectura

Arquitectura del sistema

En el desarrollo de software, la elección de una arquitectura adecuada es un factor clave para el éxito del proyecto. Para nuestra plataforma BookHunt, hemos optado por una arquitectura basada en cliente-servidor, donde el frontend y el backend están desacoplados para facilitar la escalabilidad y mantenibilidad del sistema.

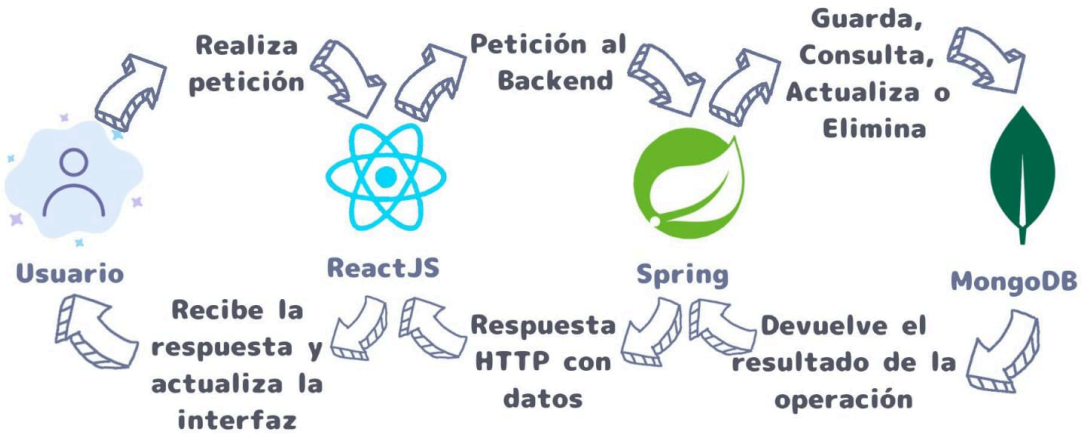


Figura 3. Arquitectura del Sistema.

Frontend:

El frontend utilizará principalmente React.js. Esta es una librería de JavaScript ampliamente utilizada para construir interfaces de usuario interactivas y dinámicas. Es ideal para este proyecto debido a su capacidad para manejar estados complejos y su eficiencia en la actualización de la interfaz de usuario. Además, React.js es compatible con la creación de componentes reutilizables, lo que facilitará la construcción de una plataforma modular y escalable.

Backend:

Para el backend se usará Spring Boot, el cual es un framework robusto y maduro para el desarrollo de aplicaciones backend en Java. Es especialmente adecuado para este proyecto debido a su capacidad para manejar operaciones complejas, como la gestión de usuarios, clubes de lectura, y la interacción con la base de datos. Además, Spring Boot ofrece una integración sencilla con bases de datos y servicios externos, lo que permitirá una implementación eficiente de las funcionalidades requeridas.

Base de datos:

MongoDB será la principal herramienta en este apartado. La herramienta se destaca por ser una base de datos NoSQL que ofrece flexibilidad para manejar datos semi-estructurados, lo que es ideal para una plataforma como BookHunt, donde los datos de los usuarios y clubes pueden variar en estructura. Además, MongoDB es escalable horizontalmente, lo que permitirá al proyecto crecer en el futuro sin problemas de rendimiento.

Comunicación Frontend-Backend:

Finalmente, para la comunicación entre el frontend (React.js) y el backend (Spring Boot) se realizará a través de una API REST. Este enfoque es estándar en el desarrollo de aplicaciones web modernas y permite una separación clara entre la lógica del frontend y el backend, facilitando el mantenimiento y la escalabilidad del proyecto.

Diseño de la base de datos:

Para garantizar un almacenamiento eficiente y escalable, es fundamental elegir una base de datos que se adapte a las necesidades de la plataforma. Dado que nuestra aplicación involucra interacciones dinámicas entre usuarios, gestión de contenido y actualización frecuente de datos, es crucial contar con un sistema flexible que permita manejar grandes volúmenes de información sin comprometer el rendimiento.

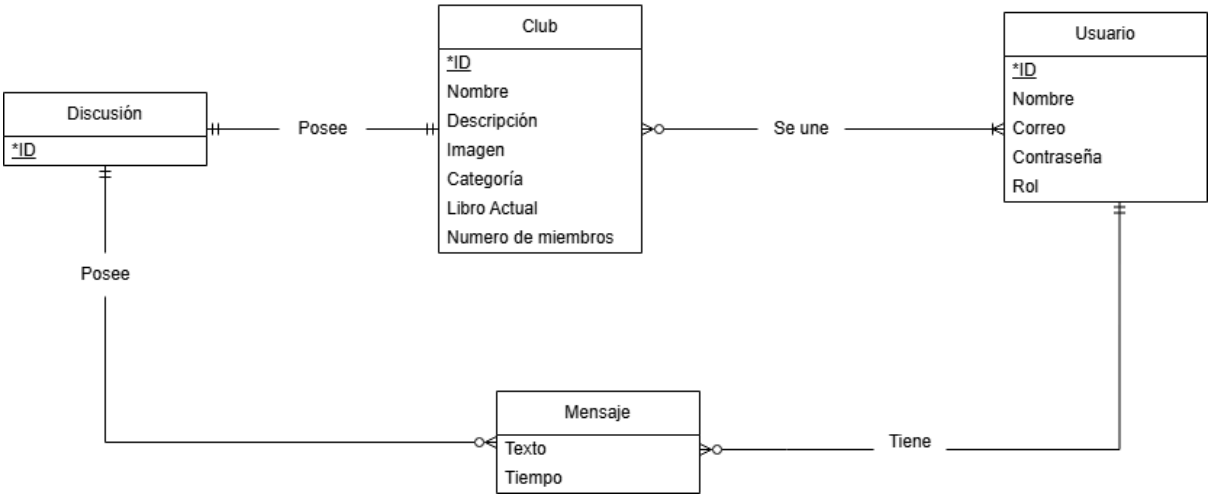


Figura 4. Diseño de la Base de Datos.

Para el almacenamiento de datos, optamos por utilizar MongoDB. Debido a su flexibilidad para gestionar datos semi-estructurados y la capacidad de escalar horizontalmente si la aplicación crece mucho en usuarios y contenido. En contextos tipo redes sociales se suele usar bases de datos no relacionales debido a que manejan datos que no suelen necesitar de la estructura compleja de una base de datos SQL.

El diagrama intenta representar la base de datos en un diagrama entidad relación con objetivo de representar la base de datos no relacional de forma gráfica.

En este se puede observar las siguientes entidades:

- Usuario: Se utiliza para almacenar los datos de logueo, y cualquier información personal del usuario, en este caso también se almacenan las contraseñas pero hasheadas.
- Club: Se utiliza para almacenar toda la información de los clubes así como sus miembros y la información de discusiones.
- Discusión: Se utiliza para almacenar información de la discusión y a su vez de los mensajes en esta.
- Mensaje: Guarda información de tiempo de envío, mensaje enviado, y usuario emisor del mensaje.

Patrones de diseño

MVC:

En el backend se implementó el patrón de diseño MVC debido a que por defecto Spring Boot organiza la aplicación en controladores para manejar las solicitudes, servicios para la lógica de negocio y repositorios/modelos para el acceso a datos. Aquí la vista es considerada la capa de presentación de datos. Nuestro sistema maneja múltiples entidades con relaciones complejas, por lo que separar la lógica en controladores, servicios y repositorios mejora la organización del código.

Problema que resuelve:

- Facilita la separación de responsabilidades, lo que mejora la mantenibilidad y escalabilidad del código.
- Permite que diferentes equipos trabajen en paralelo en las capas del sistema (backend, frontend y base de datos).
- Organiza el código de manera clara, haciendo que sea más fácil de probar y depurar.

DTO:

Por otro lado también se ha implementado el patrón de diseño DTO (Data transfer Object) debido a que Spring Boot promueve el uso de esto para evitar funciones con demasiados parámetros. En el proyecto se implementaron en la capa de controladores, donde las peticiones http del login se validan y se almacenan como un objeto en un solo parámetro, de esta forma reduciendo la cantidad de parámetros y de código.

Problema que resuelve:

- Evita la exposición directa de las entidades del modelo en las respuestas HTTP.
- Mejora el rendimiento al transferir sólo los datos necesarios en cada operación.
- Reduce el acoplamiento entre la capa de presentación y la capa de persistencia.

```java
@PostMapping("/register")  dcocinero
public ResponseEntity<AuthenticationResponse> registerUser(@Valid @RequestBody RegisterRequest request) {
    return ResponseEntity.ok(authenticationService.register(request));
}
```

Como se puede observar el único parámetro es request de tipo RegisterRequest que es la clase DTO.

```java
@Data  2 usages  dcocinero
@Builder
@AllArgsConstructor
@NoArgsConstructor
public class RegisterRequest {
    @NotBlank(message = "Username is required")
    @Size(min = 3, max = 20, message = "Username should be between 3 and 20 characters long")
    private String username;
    @NotBlank(message = "Email is required")
    @Email(message = "Email should be valid")
    private String email;
    @NotBlank(message = "Password is required")
    @Size(min = 6, message = "Password should be at least 6 characters long")
    private String password;
}
```

Que como se puede observar esta clase ya valida y almacena todos los campos de la petición.

Adjuntos

[1] Propuestas Realizadas: 1. Lluvia de Ideas Propuestas.pdf.
[2] Selección de Propuestas: 2. Selección Propuestas.
[3] Transcripción de la Videollamada: 3. Transcripción.txt.
[4] Mockups Iniciales: 4. Mockups Iniciales.png.
[5] Diagrama de Gantt: 5. Diagrama De Gantt.pdf