UNIVERSIDAD LAICA ELOY ALFARO DE MANABÍ
FACULTAD DE CIENCIAS DE LA VIDA Y TECNOLOGÍAS

TECNOLOGÍA DE LA INFORMACIÓN

TEMA:

DESARROLLO DE UNA APLICACIÓN COLABORATIVA PARA LA GESTIÓN DE CLUBES DE DEPORTE Y RECREACIÓN UTILIZANDO FLUTTER Y SUPABASE.

TECNOLOGÍA DE LA INFORMACIÓN

PRESENTADO POR:

ANGAMARCA MOYA MILTON LEIBNITZ

DIRECTOR:

ING. EDGARDO PANCHANA, MG

Manta - Manabí – Ecuador
2024-2025

Certificación del director de trabajo de graduación

Uleam	NOMBRE DEL DOCUMENTO: CERTIFICADO DE TUTOR(A).	CÓDIGO: PAT-04-F-004
	PROCEDIMIENTO: TITULACIÓN DE ESTUDIANTES DE GRADO BAJO LA UNIDAD DE INTEGRACIÓN CURRICULAR	REVISIÓN: 1
		Página 1 de 1

CERTIFICACIÓN

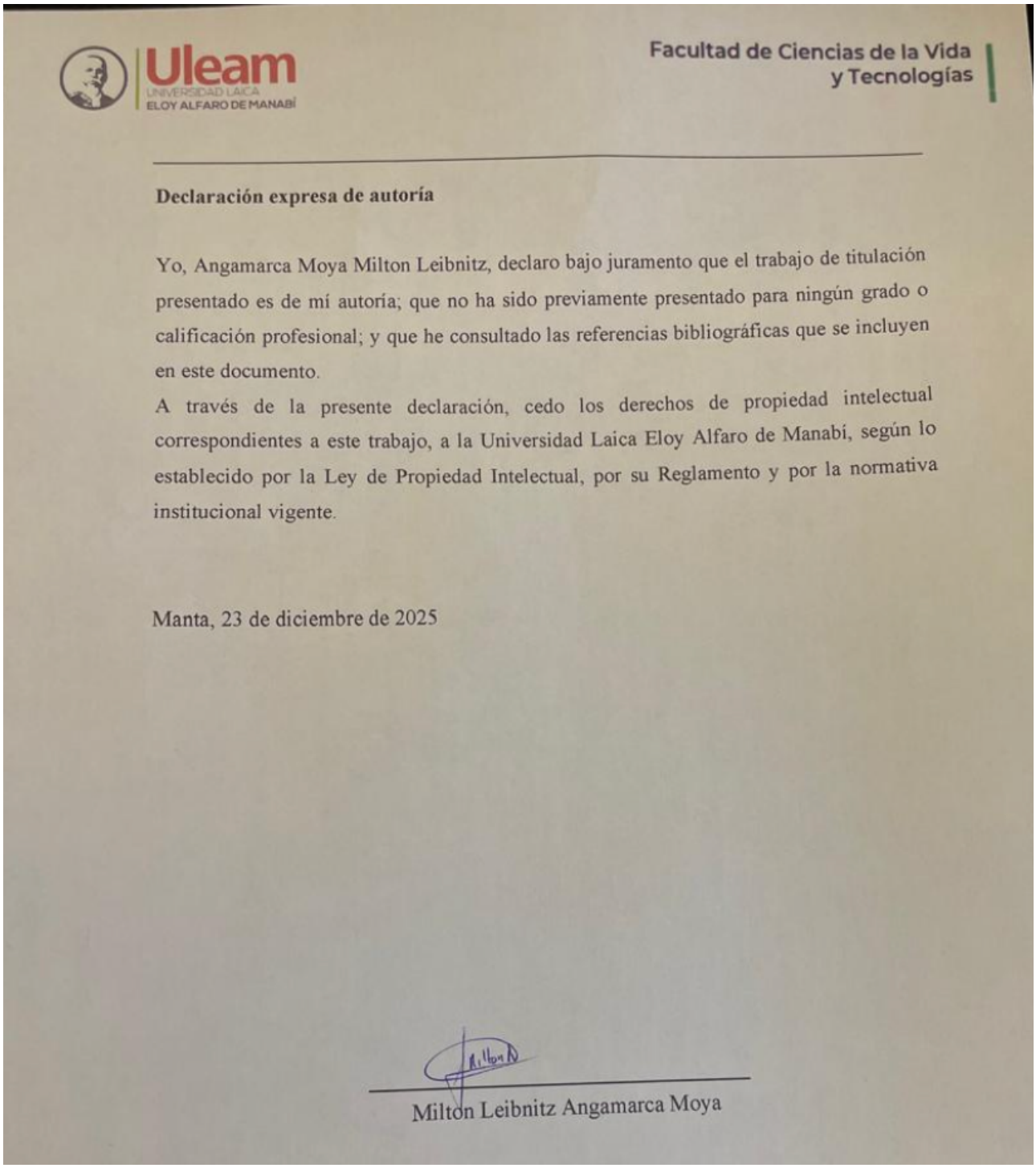
En calidad de docente tutor(a) de la Facultad de Ciencias de la Vida y Tecnologías de la Carrera de Tecnologías de la Información de la Universidad Laica "Eloy Alfaro" de Manabí, CERTIFICO:

Haber dirigido y revisado el trabajo de Integración Curricular bajo la autoría de la estudiante Angamarca Moya Milton Leibnitz, legalmente matriculado/a en la carrera de Tecnologías de la Información, período académico 2025-2026, cumpliendo el total de 384 horas, cuyo tema del proyecto es **"DESARROLLO DE UNA APLICACIÓN COLABORATIVA PARA LA GESTIÓN DE CLUBES DE DEPORTE Y RECREACIÓN UTILIZANDO FLUTTER Y SUPABASE".**
La presente investigación ha sido desarrollada en apego al cumplimiento de los requisitos académicos exigidos por el Reglamento de Régimen Académico y en concordancia con los lineamientos internos de la opción de titulación en mención, reuniendo y cumpliendo con los méritos académicos, científicos y formales, y la originalidad del mismo, requisitos suficientes para ser sometida a la evaluación del tribunal de titulación que designe la autoridad competente.
Particular que certifico para los fines consiguientes, salvo disposición de Ley en contrario.

Manta, 05 de febrero de 2026.

Lo certifico,

Ing., Panchana Flores Joffre Edgardo, Mg.
Docente Tutor



Uleam
UNIVERSIDAD LAICA
ELOY ALFARO DE MANABÍ

Facultad de Ciencias de la Vida
y Tecnologías

Declaración expresa de autoría

Yo, Angamarca Moya Milton Leibnitz, declaro bajo juramento que el trabajo de titulación presentado es de mí autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración, cedo los derechos de propiedad intelectual correspondientes a este trabajo, a la Universidad Laica Eloy Alfaro de Manabí, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normativa institucional vigente.

Manta, 23 de diciembre de 2025

Milton Leibnitz Angamarca Moya

Dedicatoria

A Dios, por ser la fuerza invisible que me sostuvo en los momentos difíciles y la luz que guio este camino.

A mis padres, los verdaderos autores de este logro. Gracias por sembrar en mí la rectitud y por cada sacrificio silencioso que hicieron para que yo pudiera volar. No solo me dieron educación, me dieron el ejemplo de que el trabajo duro rinde frutos. Todo lo que soy y lo que seré, nace de ustedes.

A mi raíz en Santo Domingo, por no dejarme olvidar de dónde vengo y a la familia que la vida me regaló en Manta, por hacerme sentir en casa lejos del hogar.

Milton Angamarca.

Agradecimiento

Cierro esta etapa con el corazón lleno porque tengo claro que nadie llega a la meta solo, esto es de todos los que estuvieron ahí empujando.

Gracias a la ULEAM y a la Facultad por ser mi casa estos años y dejarme descubrir lo que me apasiona.

Mi respeto total al Ing. Edgardo Panchana, qué director para más excepcional, gracias por la paciencia y por retarme siempre a ir más allá del código simple para buscar soluciones de ingeniería de verdad. También al Ing. Robert Moreira y a la Ing. Viviana García, gracias por creer en este proyecto y darme la mano para que la idea no se quedara solo en papel.

Pero el agradecimiento más fuerte es para los míos. A mis padres, que son mi motor y mi ejemplo de lucha diaria. A mi novia Isabel muchas gracias por ser mi paz, por aguantarme el estrés y estar ahí siempre incondicionalmente y a sus padres, gracias por el cariño, por abrirme las puertas de su hogar y hacerme sentir parte de su familia.

Y una mención muy especial para mi mejor amigo Jhon Bermúdez y a sus queridos padres, no tengo palabras para agradecerles que me acogieran como a un hijo más en tantas ocasiones, esa ayuda y ese calor de hogar fueron fundamentales para que yo pudiera terminar mi carrera lejos de mi tierra.

A mis amigos de siempre y a los que la vida me regaló aquí en Manta, gracias por las risas que hicieron la carga más ligera y por recordarme que el esfuerzo compartido vale doble

Milton Angamarca.

Contenido

ÍNDICE TABLAS

ÍNDICE FÍGURAS

Resumen

La presente investigación surge ante la necesidad de fortalecer la coordinación operativa y la gestión informativa en la Universidad Laica Eloy Alfaro de Manabí (ULEAM). Los procesos actuales de administración de espacios y difusión institucional presentan oportunidades de mejora en términos de integración y tiempos de respuesta. El objetivo principal fue desarrollar la plataforma móvil "ULEAM Community Hub", utilizando Flutter y Supabase, con el propósito de centralizar la gestión operativa y los procesos comunicacionales de las comunidades universitarias.

El estudio adoptó un enfoque mixto con orientación tecnológica. Se recopilaron datos de 135 estudiantes y 15 gestores institucionales para la definición de los requerimientos funcionales y no funcionales del sistema. La solución fue implementada bajo una arquitectura Serverless con base de datos PostgreSQL, incorporando restricciones de exclusión para optimizar la programación de reservas.

Los resultados evidencian una mejora en la coordinación logística y una optimización en los canales de comunicación mediante segmentación por roles. Se concluye que la implementación de un modelo digital centralizado contribuye al fortalecimiento de la eficiencia operativa y a la articulación de procesos en el entorno universitario.

Palabras clave: Gestión universitaria, Plataformas digitales, Flutter, Supabase, Optimización logística, Comunicación institucional.

Abstract

This research was conducted in response to the need to enhance operational coordination and information management at Universidad Laica Eloy Alfaro de Manabí (ULEAM). Current space administration and institutional communication processes present opportunities for improved integration and response times. The main objective was to develop the "ULEAM Community Hub" mobile platform using Flutter and Supabase in order to centralize operational management and communication processes within university communities.

A mixed-method approach with a technological orientation was adopted. Data were collected from 135 students and 15 institutional managers to define the system's functional and non-functional requirements. The solution was implemented using a Serverless architecture supported by PostgreSQL, incorporating exclusion constraints to optimize booking scheduling.

Results indicate improvements in logistical coordination and enhanced communication efficiency through role-based segmentation. It is concluded that the implementation of a centralized digital management model contributes to strengthening operational efficiency and process integration within the university environment.

Keywords: University management, Digital platforms, Flutter, Supabase, Logistical optimization, Institutional communication.

CAPÍTULO I: INTRODUCCION

1. INTRODUCCIÓN

1.1 Presentación del tema

Además de las asignaturas obligatorias, el entorno de la ULEAM se caracteriza por la actividad de sus comunidades tecnológicas, como la Comunidad de Estudiantes de Microsoft, los grupos de IoT y Ciberseguridad. Estos espacios son muy importantes para los estudiantes. Ayudan a conectar la teoría con la práctica.También ayudan a los estudiantes a trabajar juntos en proyectos que se asemejan al mundo laboral.

Pero hay un gran problema con estos planes como compartir información. Si solo utilizas medios como las redes sociales y el correo electrónico para comunicarte y no utilizas otros canales como los tablones de anuncios físicos, estás perdiendo oportunidades importantes. Esto se debe a que los estudiantes no se enteran a tiempo y los líderes no pueden llegar a su público. Para ayudar con esto, el proyecto sugiere crear un sistema móvil que se pueda utilizar en diferentes plataformas. Este sistema se construirá utilizando Flutter y Supabase. El objetivo es claro que es reunir toda la gestión de la comunidad en una herramienta eficiente que agrupe toda la información y consiga que los estudiantes se involucren más en la facultad.

1.2 Ubicación y contextualización de la problemática

El presente proyecto de investigación y desarrollo tecnológico se sitúa en el contexto de la Universidad Laica Eloy Alfaro de Manabí (ULEAM), una institución que, además de su oferta académica, promueve el bienestar integral de su comunidad a través del deporte y la recreación. En este entorno universitario conviven diariamente estudiantes, docentes y personal administrativo, quienes conforman diversos clubes deportivos y grupos recreativos —tanto formales como informales— en disciplinas como fútbol, baloncesto, voleibol y atletismo. Estas actividades son esenciales para fomentar la salud física, la integración social y el equilibrio vida-trabajo dentro del campus; no obstante, su desarrollo se ve obstaculizado por un entorno organizativo carente de herramientas modernas de gestión. La problemática se contextualiza en la informalidad y dispersión con la que se administran actualmente estas actividades deportivas. La coordinación de encuentros, la reserva de instalaciones (canchas y coliseos) y la gestión de los integrantes de los clubes se realizan mediante procesos manuales o canales de comunicación no integrados, como grupos de mensajería instantánea

y solicitudes físicas. Esta fragmentación genera un desconocimiento generalizado sobre la agenda deportiva universitaria, dificultades para encontrar equipos o rivales para partidos amistosos y una subutilización de la infraestructura recreativa debido a cruces de horarios o falta de organización previa.

1.3 Planteamiento de problema

1.3.1 Problematización

¿De qué manera el desarrollo de una solución tecnológica móvil centralizada permite mitigar la fragmentación informativa y optimizar la gestión logística de espacios físicos en las comunidades de la Facultad de Ciencias de la Vida y Tecnologías de la Información de la ULEAM?

1.4 Estado actual del problema

Actualmente, los clubes académicos anteriormente llamados comunidades de la ULEAM (como la IEEE o la Microsoft Student Community) funcionan como islas desconectadas de un canal efectivo de comunicación debido a la falta de herramientas tecnológicas. La comunicación depende casi exclusivamente de chats cerrados o del 'boca a boca' de los estudiantes o docentes lo que lleva a la creación de silos de información donde talleres y torneos pasan desapercibidos para la gran mayoría de la comunidad universitaria. Esta desconexión impide que estudiantes y docentes aprovechen oportunidades de crecimiento que deberían ser abiertas y visibles para todos.

Existe un listado publicado por la universidad sobre los clubes a nivel macro mas no informacion concreta a nivel interno (DBU (Departamiento de Bienestar Universitario), 2026), clubes actualemente reconocidos y en funcionamientos estan publicados desde la página de la universidad dentro de la direccion de Bienestar, Admisión y Nivelación Universitariay son los siguientes:

- Club de Protocolo
- Club de Comunicación
- Club de Fútbol
- Club de Básquet
- Club de Turismo y Aventura

En el plano logístico lo que ocurre es que el manejo de la infraestructura física es un cuello de botella bastante presente por ejemplo reservar un auditorio o una cancha sigue siendo un trámite manual y presencial lo que lo vuelve dependiente de bitácoras en papel que inducen al error humano ya que sin un sistema automatizado que muestre la disponibilidad real en todo momento se vuelve muy frecuentes los cruces de horario y la duplicidad de reservas ente eventos esto hace que la camunidad educativa se frustre la planificación de los organizadores y deja espacios vacíos que podrían estar siendo utilizados en cualquier evento provechoso para el desarrollo y recreación de los estudiantes.

Finalmente, existe un gran vacío administrativo grave el cual es la falta de datos en algún lugar ordenado y con seguimiento en tiempo real ya que al no haber un sistema centralizado que registre asistencias o actividades, la universidad no tiene cómo medir el impacto real de estos grupos. Sin métricas ni historial, es imposible tomar decisiones fundamentadas sobre presupuestos, mantenimiento o reconocimiento académico; básicamente, se están gestionando los recursos 'a ciegas'.

1.5 Tabla de factores causas y efectos

Factor	Causa	Efecto
Tecnológico	Inexistencia de un sistema de información o aplicación móvil dedicada a la gestión deportiva (SportTech).	Dependencia absoluta de herramientas no especializadas (chats, papel, hojas de cálculo aisladas), impidiendo la automatización de procesos básicos.
Comunicacional	Segmentación de la comunicación ya que existe muchos canales los cuales no son oficiales o son informales y llegan a ser engorrosos hasta el punto del boca a boca	Exclusión involuntaria de gran parte de la comunidad universitaria ya que no pertenece a esos círculos que involuntariamente se vuelven 'privados' lo que reduce la captación de nuevos talentos, participantes y coste de oportunidad seria inmenso.

Logístico / Operativo	Los procesos aún siguen siendo muy pesados a nivel documental ya que se maneja de manera presencial sin formato disponible para todo estudiante y muy burocrático.	Saturación administrativa lo que termina como conflictos por duplicidad de reservas (doble booking) y subutilización no requerida de espacios de productividad por falsa percepción de ocupación.
Administrativo / Gestión	Falta de un registro oficial que centralice los clubes, líderes de equipos y cronogramas de actividades de cada una de las comunidades.	Perdida de captación de estudiantes en eventos que son de su interés tanto recreacional como p, el coste de no adquirir una habilidad que lo impulse al mundo laboral.
Estadístico / Analítico	Ausencia de los mecanismos necesarios para la recolección de datos que son relevantes en la participación y resultados de los clubes.	Esto genera la imposibilidad de tener proyecciones a futuro o un seguimiento de cómo se están usando los medios y por quien o que club ya que no se pueden generar gráficos ni toma de decisiones si no hay con que trabajar

Tabla 1: Tabla de factores causa y efecto

1.6 Objetivos

1.6.1 Objetivo general

Desarrollar una solución tecnológica móvil mediante Flutter y Supabase capaz de articular la administración de las comunidades de la ULEAM, centralizando sus operaciones para lograr una optimización real en los tiempos de reserva de espacios, la efectividad en la difusión de actividades y la cohesión de la comunidad universitaria.

1.6.2 Objetivos específicos

- Diagnosticar los flujos de información y los procedimientos logísticos actuales de los clubes y comunidades universitarias para determinar los requerimientos funcionales y no funcionales del sistema.
- Implementar el servicio de Supabase como backend para la solución y así implementar el servicio de autenticación también con el servicio de datos en tiempo real y el almacenamiento para así garantizar la integridad y disponibilidad de la solución.
- Construir la interfaz utilizando Flutter para el desarrollo de componente para la gestión de agendamiento de eventos también si visualización para usuarios que aún no han agendado, adicionalmente de publicaciones y muro de post de cada comunidad y disponibilidad de información.

1.7 Justificación

La justificación de este desarrollo es clara ya que necesitamos alinear la gestión de la universidad con la forma en que el mundo funciona hoy. Los modelos tradicionales basados en la presencialidad crean cuellos de botella que ya no son aceptables. Al implementar una plataforma unificada es la estrategia correcta para administrar mejor tanto el talento humano como las instalaciones de la facultad mediante la tecnología.

Este proyecto ataca un problema que nos afecta a todos es la desconexión. Al juntar la gestión de todos los clubes en una app, se acaba el problema de la información dispersa que limita la participación estudiantil. Técnicamente, usar Flutter y Supabase nos permite entregar una solución robusta y económica de mantener a largo plazo. Pero más allá del código, el proyecto se valida porque mejora la vida universitaria. Simplificar la reserva de espacios y la difusión de eventos libera a los líderes de tareas repetitivas, dejándoles tiempo para enfocarse en la calidad de sus actividades y promoviendo una formación verdaderamente integral.

1.8 Impactos esperados

1.8.1 Impacto tecnológico

Este proyecto representa un salto real hacia la modernización de la ULEAM, dejando atrás los trámites en papel para pasar a un sistema 100% en la nube. La decisión de usar Supabase como servicio Backend es clave, ya que nos permite tener una base de

datos centralizada y segura. Con esto, eliminamos de raíz el desorden de manejar información en hojas de cálculo sueltas o registros físicos que suelen perderse.

Por otra parte, desarrollar con Flutter establece un nuevo nivel de calidad técnica en la universidad. No solo logramos que la aplicación corra fluida en Android y iOS sin duplicar trabajo, sino que dejamos una estructura lista para futuros proyectos. Básicamente, la plataforma demuestra que es totalmente viable usar tecnologías como el tiempo real, las notificaciones push y la geolocalización de eventos para mejorar la comunicación en el campus.

1.8.2 Impacto social

Socialmente, el proyecto tiene un efecto democratizador clave. Al centralizar en una sola aplicación tanto a los clubes deportivos como a los tecnológicos y culturales, rompemos los 'silos' que hoy aíslan a las facultades. Esto permite que las comunidades de desarrollo o robótica no se queden encerradas en sus carreras, sino que se abran a todos. Así logramos que estudiantes técnicos convivan con humanistas, ya sea aprendiendo nuevas habilidades digitales, debatiendo ideas o compartiendo en una cancha, lo que fortalece el tejido social de la universidad.

Por otro lado, la plataforma ayuda directamente al bienestar y la salud mental de estudiantes, docentes y administrativos. Al simplificar la logística para unirse a grupos de recreación o encontrar actividades, facilitamos que la gente tenga una válvula de escape ante la presión académica. Básicamente, generamos inclusión, asegurando que nadie se quede fuera de la vida universitaria ya sea de un torneo o de un taller técnico simplemente por no tener los contactos adecuados.

1.8.3 Impacto económico

El impacto económico de la presente propuesta se manifiesta, en primera instancia, en la reducción drástica del consumo de insumos físicos como papel, carpetas y agendas analógicas, lo cual deriva en una gestión documental mucho más eficiente y sostenible. Al centralizar la comunicación, el sistema evita que las comunidades inviertan tiempo y recursos humanos en convocatorias ineficaces a través de plataformas saturadas de contenido irrelevante (spam), cuya dispersión suele conducir al fracaso de las iniciativas. En tal sentido, la solución permite una optimización presupuestaria al prescindir de gastos en medios de difusión tradicionales, como pancartas o pautas publicitarias en redes sociales genéricas,

puesto que la plataforma ofrece un canal de comunicación directo y gratuito para los actores institucionales

Por otro lado, la toma de decisiones se agiliza significativamente debido a que no se requiere de un sobreesfuerzo técnico ni del procesamiento manual de información dispersa, lo que permite reducir los costos operativos asociados al análisis de datos descentralizados. Adicionalmente, el control riguroso de la infraestructura física permite una optimización de los activos de la Facultad; al conocer con exactitud el estado y la ocupación de cada área, se previene el despilfarro operativo y se evita la existencia de "instalaciones muertas" o subutilizadas. De este modo, el sistema garantiza que cada espacio físico tenga un uso real y constante, justificando financieramente las inversiones en mantenimiento y asegurando que el presupuesto institucional se asigne de manera coherente con la demanda académica real

1.8.4 Impacto académico e institucional

En lo académico, la herramienta cambia la dinámica al empujar el aprendizaje colaborativo. Usando un sistema de hilos de discusión, las comunidades pueden debatir, compartir avances y resolver dudas fuera del horario de clases. Básicamente, creamos un repositorio de conocimiento vivo que fomenta el peer-to-peer learning, donde los estudiantes aprenden unos de otros y no solo del profesor.

Para la institución, el valor real está en los datos. Por fin la universidad tendrá una visión clara de la 'vida universitaria' con métricas reales. La plataforma permite ver qué temas interesan de verdad, qué comunidades son las que más se mueven y cómo interactúan las facultades entre sí. Este social listening es un activo clave para que las autoridades entiendan a su población y diseñen políticas que respondan a lo que los estudiantes realmente necesitan, en lugar de adivinar.

CAPÍTULO II: MARCO TEORICO DE LA INVESTIGACION

2.1. Introducción

Para que una solución tecnológica realmente logre mejorar la gestión y la unión de las comunidades en la ULEAM, no basta con escribir código; necesitamos una base teórica sólida que conecte la realidad de la universidad con la tecnología moderna. En este capítulo, construimos justamente ese sustento, estableciendo los criterios necesarios para diseñar la plataforma móvil sin improvisaciones.

Nuestro recorrido empieza revisando los Antecedentes de la Investigación. Aquí nos dedicamos a mirar qué se ha hecho antes, tanto a nivel internacional como local. La idea es analizar otras experiencias de gestión estudiantil para ver qué funcionó y qué falló. Esto es clave, porque nos permite aprender de los errores ajenos y ubicar nuestro proyecto como una evolución real en la tecnología educativa, en lugar de partir de cero. Después, entramos de lleno en las Bases Teóricas, que hemos dividido en dos grandes áreas para cubrir todo el espectro del proyecto.

Por un lado, analizamos la parte humana y administrativa: estudiamos cómo se mueven las comunidades reales como la IEEE, los equipos deportivos o los grupos de Inteligencia Artificial y explicamos por qué el networking es vital para la formación del estudiante. Adicionalmente nos enfocamos en la ingeniería del software. Aquí detallamos la arquitectura técnica, justificando por qué elegimos el desarrollo móvil con Flutter y la infraestructura en la nube con Supabase. No se trata solo de nombrar herramientas, sino de explicar los patrones de diseño que usamos para levantar una red social vertical que sea segura y capaz de crecer con el tiempo.

Finalmente, se concluye el capítulo con dos puntos necesariamente obligatorios. Primero la Fundamentación Legal que es donde nos aseguramos de que el proyecto respete la normativa ecuatoriana, especialmente en lo que toca a la educación superior y la protección de datos personales. Y segundo, la Definición de Términos Básicos, un glosario técnico esencial para aclarar el vocabulario y asegurar que cualquier persona que lea la tesis entienda exactamente lo mismo al ver los conceptos técnicos que manejamos.

2.2 Antecedentes de la Investigación

2.2.1 Antecedentes internacionales

El En el panorama global, la digitalización de los servicios universitarios ha evolucionado hacia el concepto de Smart Campus. Al respecto, Rico-Bautista et al. (2021) realizaron una revisión sistemática de literatura bajo el protocolo PRISMA, analizando 123 artículos científicos para identificar las dimensiones clave del campus inteligente. Su investigación determinó que la gestión de datos masivos y el uso de tecnologías en la nube son los pilares para optimizar los recursos institucionales. Este hallazgo fundamenta la decisión de este proyecto de emplear una infraestructura Serverless para centralizar la vida estudiantil.

Por su parte, Vázquez-Cano y Sevillano (2018) emplearon un enfoque cuantitativo-descriptivo con una muestra de estudiantes universitarios en España, utilizando encuestas validadas por expertos para medir el impacto de la tecnología móvil. Los autores concluyeron que el uso de dispositivos digitales incrementa la participación estudiantil en un 40%, siempre que los servicios se centralicen en interfaces intuitivas. Dicho resultado valida nuestra propuesta de un feed de noticias segmentado para revitalizar las comunidades de la ULEAM.

Finalmente, García-Peñalvo (2021) analizó la interoperabilidad de los ecosistemas tecnológicos mediante un análisis comparativo de arquitecturas de software. El autor determinó que el desarrollo con marcos multiplataforma es la solución más eficaz para reducir la fragmentación de la información. Esto respalda técnicamente la elección de Flutter, ya que su capacidad de compilación AOT y su motor Impeller aseguran un rendimiento de 60 FPS en dispositivos con arquitecturas ARM64, garantizando una experiencia de usuario homogénea.

2.2.2 Antecedentes nacionales y locales

En el entorno ecuatoriano, las Instituciones de Educación Superior (IES) atraviesan una digitalización acelerada. Cadena-Vela et al. (2019), a través de un diagnóstico situacional del estado de las TIC en 42 universidades del país, demostraron que la mayoría de las instituciones han automatizado procesos académicos, pero la gestión del bienestar sigue relegada a métodos manuales. Este estudio estadístico es clave para justificar por qué

en la ULEAM es imperativo digitalizar las reservas de espacios, eliminando las latencias de hasta 48 horas detectadas en el diagnóstico.

A nivel de implementación técnica nacional, Haro Garcés (2022) desarrolló en la UISEK un prototipo funcional integrando IoT y software móvil. Su metodología incluyó el uso de Node-RED como orquestador, hardware Raspberry Pi y el API de Telegram para el control de acceso a laboratorios. Si bien su enfoque fue el control físico, su éxito al automatizar la disponibilidad de espacios mediante software valida nuestra propuesta de gestión digital para la infraestructura de la ULEAM.

Aterrizando en el contexto local, grupos como la rama estudiantil IEEE o los clubes de robótica de la Facultad operan actualmente sin un repositorio digital centralizado o listada en la página de bienestar universitario. La falta de herramientas oficiales fuerza a los líderes a manejar convocatorias de forma empírica mediante canales bidireccionales saturados de spam. Esta brecha justifica la creación de una arquitectura de comunicación unidireccional y segmentada, que asegure la integridad del mensaje y promueva el sentido de pertenencia en la comunidad universitaria.

2.3 Fundamentación teórica

2.3.1 Dinámicas de interacción y gestión en entornos universitarios

Las universidades funcionan como redes sociales vivas; no se trata solo de transferir conocimiento en un aula, sino de gestionar relaciones administrativas y culturales. Manejar bien estas dinámicas es vital. Según (García-Peñalvo F. , 2021), la universidad actual tiene que dejar atrás los modelos jerárquicos viejos y pasar a “ecosistemas conectados”, donde la información corra horizontalmente entre estudiantes, profes y administrativos.

El problema en universidades como la ULEAM es que la comunicación está dispersa. Como no hay canales estandarizados, coordinar una actividad extracurricular depende de redes informales y esfuerzo manual, lo que crea desorden y “ruido organizacional”. Para que la gestión sea eficiente, necesitamos herramientas tecnológicas que centralicen la comunicación y simplifiquen el uso de recursos compartidos, logrando que la interacción entre todos sea fluida y transparente (UCE, 2021) & (Reina, 2022).

2.3.2 El rol de las comunidades de práctica y aprendizaje en la formación integral

Hoy en día, las comunidades de práctica son una pieza clave en la educación superior. (Wenger E. , 2010) las describe tal cual: “grupos de personas que comparten una pasión por algo que hacen y aprenden a hacerlo mejor interactuando regularmente”.

No estamos hablando de conceptos abstractos, sino de grupos que tambien operan dentro de un sola facultad, como las ramas IEEE, las comunidades Microsoft, los clubes de robótica o los equipos deportivos. Básicamente, estos lugares sirven como el primer escenario de validación profesional, donde el estudiante pone a prueba lo que sabe antes de salir al mercado."

Su rol en la formación integral es insustituible por tres razones principales:

- Desarrollo de Competencias Blandas: Fomentan habilidades como el liderazgo, la gestión de proyectos y el trabajo en equipo, las cuales son difícilmente replicables en una clase teórica tradicional.
- Networking Profesional: Conectan a los estudiantes con las redes globales y con docentes mentores lo cual les facilita la inserción laboral futura por medio de contactos.
- Sentido de Pertenencia: Según (Tinto, 2006), la participación en clubes reduce las tasas de deserción estudiantil al fortalecer el vínculo emocional y social del alumno con la institución.

2.3.3 Modelos de comunicación organizacional en instituciones de educación superior (IES)

La comunicación dentro de las universidades ha cambiado radicalmente: pasamos de las viejas jerarquías verticales a modelos en red. Según (Castells, 2009), la eficiencia de una organización moderna se mide por su capacidad de mover información en todas direcciones a la vez. Siendo realistas, los modelos tradicionales como los boletines o los correos masivos ya no sirven; se quedan cortos ante la velocidad de respuesta que exigen los estudiantes de hoy.

Aquí es donde entra la teoría de la comunicación digital proponiendo redes sociales verticales. A diferencia de redes generales como Facebook o X, estas plataformas institucionales eliminan el 'ruido' y centran la información en un entorno seguro. La ventaja

principal es la segmentación: logras que a un estudiante de ingeniería le llegue info sobre hackathones y a uno de cultura sobre teatro. Así, el mensaje se vuelve relevante y la participación real aumenta."

2.3.4 Gestión logística de infraestructura educativa: Optimización de espacios compartidos

La administración de los activos físicos universitarios (canchas, auditorios, laboratorios) se sustenta en la teoría de la Gestión de Operaciones y Servicios. La problemática común de subutilización o conflicto de horarios (solapamiento) responde a la falta de visibilidad de la oferta y demanda en tiempo real.

(Davenport, 2013) sostiene que la digitalización de procesos logísticos mediante sistemas de reservas (Booking Systems) es esencial para maximizar el retorno de la inversión en infraestructura. Un sistema centralizado permite transitar de una gestión reactiva y manual a una planificación predictiva. Al automatizar las reservas, se garantiza la equidad en el acceso a los recursos institucionales y se generan datos históricos que permiten a las autoridades tomar decisiones basadas en evidencia sobre el mantenimiento o ampliación de las instalaciones.

2.4 Arquitectura de sistemas de software social

Diseñar sistemas para interacción social es muy distinto a construir software tradicional. Mientras que los sistemas administrativos se obsesionan con que los datos cuadren (integridad transaccional), el software social gira en torno al Grafo Social: es decir, mapear las relaciones vivas entre usuarios, grupos y contenidos. Según (Bouman, 2007), la arquitectura aquí debe resolver problemas complejos de escala y coordinación, asegurando que la infraestructura aguante la naturaleza impredecible de cómo interactúan las personas.

2.4.1 Redes sociales verticales vs. horizontales: El paradigma del entorno especializado

La teoría marca una diferencia estructural clave entre las redes horizontales y las verticales. Entender esto es vital para que la comunicación funcione. El problema de las redes horizontales es que conectan a una audiencia masiva y mezclada sin un objetivo claro, lo que casi siempre termina en dispersión de información y mucho 'ruido cognitivo'.

En contraposición, las Redes Sociales Verticales o de nicho se diseñan para servir a una comunidad con intereses, identidad y objetivos compartidos. (Maldonado & García,

2018) sostienen que, en el ámbito de la educación superior, las arquitecturas verticales son superiores debido a que ofrecen una contextualización de la información. Este modelo permite la implementación de lo que (Wenger et al., 2009) denominan "hábitats digitales", espacios segregados donde la interacción se rige por normas y roles específicos.

En términos técnicos, esto implica montar una base de datos que aguante la segmentación de audiencias, asegurando que lo que lee el usuario le sirva realmente para su carrera.

Más adelante, y en la línea de (GARRETT, 2005) llegaron herramientas para facilitar el desarrollo. jQuery destacó porque permitía tocar el DOM sin complicaciones y funcionaba en todos los navegadores. Pero cuando las apps web crecieron en complejidad, hizo falta más organización. Ahí es donde entran Angular, React y Vue.js. Lo mejor de estos frameworks es que te dan componentes listos y ordenan la casa, haciendo posible que varias personas programen a la vez sin que todo termine en un desastre. Dividen la interfaz en componentes, integran cosas como el Virtual DOM o la inyección de dependencias, y eso hace que la aplicación no solo funcione, sino que lo haga rápido. (Haverbeke, 2018), en Eloquent JavaScript, lo explica sin rodeos: el modularidad y la reutilización de código no son lujos, son la única manera de mantener con vida un proyecto grande, a evolución de los frameworks de frontend ha permitido pasar de páginas estáticas a aplicaciones altamente interactivas. Como señala (Haverbeke, 2018), la modularidad y reutilización de código se han convertido en condiciones indispensables para sostener proyectos grandes y duraderos. En Ecuador,la Universidad Técnica de Ambato (UTA, 2022) resalta que el uso de frameworks modernos facilita la integración de servicios digitales en campus universitarios, al reducir los tiempos de carga y mejorar la accesibilidad desde distintos dispositivos (Haverbeke, 2018).

2.4.2. Patrones de diseño de interacción en tiempo real y estructuras de datos

Para que una plataforma colaborativa funcione de verdad, la clave está en cómo modelamos y movemos las interacciones. No basta con guardar datos; necesitamos patrones de arquitectura que sepan manejar el tráfico de información en vivo. Arquitectura de Distribución de Contenidos (Feed Architecture): El mayor dolor de cabeza en una red social es lograr que los mensajes lleguen rápido y a quien deben llegar. Por eso, en este proyecto descartamos los modelos tradicionales y nos fuimos por una arquitectura

basada en Eventos y Suscripciones (Pub/Sub). Básicamente, este modelo permite que el sistema reaccione al instante cuando alguien publica algo, distribuyéndolo solo a los interesados sin saturar el servidor. Utilizando las capacidades de tiempo real de las bases de datos modernas, el cliente móvil se suscribe a cambios específicos en la estructura de datos. Esto elimina la necesidad de recargas manuales, reduciendo la latencia de la comunicación y creando una sensación de inmediatez o "presencia digital".

Modelado de Conversaciones (Threading) La comunicación lineal resulta ineficiente para la construcción de conocimiento acumulativo. Para organizar las discusiones, usamos una arquitectura de Hilos (Threading). En la base de datos, esto se resuelve de forma práctica con el modelo de Lista de Adyacencia, donde cada comentario sabe exactamente quién es su 'padre'. Coincidiendo con (Arguello, 2006), los hilos son vitales en educación porque permiten abrir varias líneas de debate simultáneas sin que se pierda el hilo principal, facilitando que cualquiera encuentre la información después (Changder, 2024).

Por otro lado, no es casualidad que gigantes como Facebook, Instagram o Netflix tengan esta tecnología en su arsenal. Usar librerías como React permite que los equipos de desarrollo trabajen en paralelo en distintas partes de la aplicación sin 'pisarse' el código ni romper el trabajo del otro.

2.4.3. Mecanismos de reputación, engagement y gamificación estructural

La sostenibilidad de una comunidad digital depende de la motivación intrínseca de sus usuarios y no solo de la disponibilidad del software. La arquitectura del sistema debe incorporar principios de la Teoría de la Autodeterminación aplicada al diseño funcional.

(Deterding, 2011) describen la gamificación institucional de forma clara: usar las mecánicas de los juegos en entornos serios para solucionar la falta de motivación. Se aterrizo este concepto en la arquitectura creando un sistema de reputación digital vivo. No se trata solo de puntos, sino de validación social real mediante feedback positivo, donde los roles del usuario no son fijos, sino que evolucionan dinámicamente según su nivel de actividad. La visualización de logros y la participación en comunidades transforman el perfil del usuario en un portafolio de habilidades blandas el cual servirá a futuro para procesos internos de la Universidad lo cual fomenta el liderazgo distribuido dentro de la plataforma. (Gerchev, 2022).

2.5 Tecnologías de desarrollo móvil multiplataforma

En los tiempo que corren se ha tratado bastante el tema del desarrollo móvil vs de desarrollo web responsivo ya que son tecnologías que son diferentes por lo cual se debate en el uso y su aplicabilidad ya que la diversidad de las web es muy grande pero el desarrollo multiplataforma es un escenario específico para móviles Android y IOS , según (Belloum, 2025) señala que "los frameworks multiplataforma modernos han superado las limitaciones de rendimiento históricas, permitiendo a las organizaciones reducir los costos de desarrollo hasta en un 40% sin sacrificar la experiencia de usuario final".

2.5.1 Evolución del desarrollo: Nativo vs. Cross-Platform

Históricamente, el desarrollo móvil obligaba a elegir entre dos caminos: o buscabas el rendimiento puro del nativo (Swift/Kotlin) o preferías la velocidad de desarrollo del multiplataforma. Sin embargo, entre 2024 y 2025 la industria dio un giro. Las arquitecturas viejas basadas en "puentes" quedaron atrás para dar paso a los motores de renderizado directo.

La limitación de las arquitecturas híbridas tradicionales Hasta hace poco, frameworks como React Native dominaban usando un "JavaScript Bridge" para comunicarse con el dispositivo. Aunque era una solución rentable, ese puente terminaba siendo un cuello de botella real en el hilo principal (UI Thread). Básicamente, si metías animaciones complejas o mucha carga gráfica, la serialización de datos atascaba el paso, causando caídas visibles en los FPS y lag.

La solución de la nueva generación Tecnología como Flutter resolvió este problema eliminando el puente por completo. A diferencia de sus antecesores, usa compilación Ahead-of-Time (AOT) para convertir el código Dart directamente en instrucciones de máquina (ARM64/x86). La gran diferencia es que la app no le pide prestados los componentes al sistema operativo, sino que dibuja cada píxel por su cuenta usando su propio motor (Skia o Impeller). Esto permite saltarse las capas intermedias e igualar el rendimiento nativo, garantizando una fluidez estable de 60 a 120 FPS.

2.5.2. Arquitectura Interna del framework y el motor de renderizado (Impeller)

A diferencia de los enfoques híbridos convencionales que actúan como una capa de abstracción sobre los widgets del sistema operativo (OEM), Flutter implementa una arquitectura de "Control Total de Píxeles". Su diseño se estructura bajo un modelo de capas

desacopladas que otorga al desarrollador control granular sobre el ciclo de vida de la renderización, desde la lógica de negocio hasta la rasterización en la GPU la arquitectura de capas (Layered Architecture) el sistema se compone de tres estratos fundamentales que operan de manera jerárquica:

- El Framework (Dart): La capa superior donde reside la lógica reactiva. Implementa el patrón de composición mediante tres árboles de objetos: el Widget Tree (configuración inmutable), el Element Tree (instancia y estado) y el RenderObject Tree (geometría y pintura). Esta separación permite que el sistema calcule los cambios (diffing) de manera eficiente sin recomponer toda la interfaz gráfica (Garrido, 2024).
- El Engine (C++): Es el "músculo" de bajo nivel del sistema. Se encarga de la rasterización, gestiona la memoria y controla el bucle de eventos. Básicamente, aquí está la lógica que le dice a la API gráfica del dispositivo qué y cómo dibujar.
- El Embedder (Plataforma Específica): Actúa como el puente de integración. Es el responsable de coordinar la entrada/salida y los hilos nativos, configurando el Canvas para que la aplicación corra nativamente ya sea en iOS, Android o Web.

La evolución del Motor Gráfico: De Skia a Impeller Hasta 2023, Flutter funcionaba con Skia, un motor 2D clásico. El inconveniente de Skia en móviles modernos era que compilaba los sombreadores sobre la marcha (JIT). Si lanzabas una animación compleja por primera vez, el motor tenía que compilar el shader en ese instante, rompiendo a menudo el límite de 16ms por cuadro y causando ese molesto tartamudeo visual (Shader Compilation Jank).

Para solucionar esto de raíz, Google implementó y estabilizó el motor Impeller entre 2024 y 2025. Según la documentación de ingeniería (Flutter Team, 2025), este cambio rediseña el flujo gráfico bajo dos principios:

- Compilación AOT de Shaders: A diferencia de su predecesor, Impeller deja todos los shaders y objetos de estado (PSO) precompilados desde la fase de construcción (build time). Así eliminamos el cálculo en tiempo de ejecución y garantizamos un rendimiento fluido desde el primer frame.

- Arquitectura de Modo Retenido: El motor mantiene las referencias de los objetos directamente en la GPU, lo que reduce drásticamente el tráfico y la carga de comunicación con el procesador central.
- Aparalelismo y Primitivas Modernas: El motor está diseñado para explotar las APIs gráficas de bajo nivel actuales, interactuando directamente con Metal (en iOS) y Vulkan (en Android). Implementa técnicas avanzadas como el uso agresivo de Uniform Buffers y la teselación en la GPU, lo que descarga a la CPU de tareas de cálculo geométrico pesado (Nawrot M. , 2024)

Esta transición tecnológica asegura que la plataforma propuesta para la ULEAM no solo sea funcional, sino que ofrezca una experiencia de usuario fluida y capaz de mantener tasas de refresco de 120 Hz en dispositivos compatibles, un estándar de calidad obligatorio en el desarrollo móvil contemporáneo (Supabase, 2025).

2.6 Infraestructura de servicios en la nube (Backend as a Service)

La arquitectura de backend moderna ha experimentado una transición significativa desde los despliegues monolíticos en servidores virtuales (IaaS) hacia modelos orientados a eventos y servicios gestionados, conocidos como Backend as a Service (BaaS). Según el informe de tendencias en la nube de (Gartner, 2024), se proyecta que para finales de 2025, el 60% de las nuevas aplicaciones móviles empresariales se construirán sobre arquitecturas Serverless o BaaS, debido a la necesidad de reducir la carga cognitiva asociada al mantenimiento de infraestructura (parches, escalado vertical) y enfocar los recursos de ingeniería en la lógica de negocio.

Este modelo permite desacoplar el frontend de la gestión de bases de datos y autenticación, delegando la responsabilidad de la disponibilidad y la seguridad perimetral al proveedor de la nube.

2.6.1 Plataforma supabase: Arquitectura relacional y open source

Para el presente proyecto, se selecciona Supabase como infraestructura central. A diferencia de soluciones NoSQL tradicionales como Firebase, Supabase se fundamenta en PostgreSQL, un motor de base de datos relacional objeto-extensible.

La arquitectura de Supabase integra herramientas de código abierto para exponer la base de datos como una API RESTful escalable sin necesidad de escribir código de servidor intermedio (Middleware). Esto se logra mediante PostgREST, una capa que convierte

automáticamente el esquema de la base de datos en endpoints RESTful documentados. Investigaciones recientes de (Code & Architect, 2025) destacan que PostgREST ofrece un rendimiento superior a los servidores API tradicionales escritos en Node.js o Python, capaz de manejar más de 2000 solicitudes por segundo en instancias básicas, gracias a su estrecha integración con el núcleo de PostgreSQL y la reducción de la sobrecarga de serialización JSON.

Esta elección garantiza la Integridad Referencial de los datos (clave para gestionar relaciones complejas entre estudiantes, comunidades y reservas), algo que las bases de datos documentales (NoSQL) gestionan con dificultad y redundancia de datos.

2.6.2. Sincronización en tiempo real

La capacidad de sincronización instantánea es el diferenciador crítico en aplicaciones colaborativas modernas. A diferencia de las arquitecturas tradicionales que dependen del sondeo periódico (polling) un mecanismo ineficiente que satura el ancho de banda y aumenta la latencia, la solución propuesta implementa un modelo basado en eventos (Event-Driven Architecture). Ingeniería del Motor Realtime (Elixir & BEAM) Supabase gestiona la concurrencia masiva utilizando un servidor de tiempo real construido sobre Elixir y la máquina virtual de Erlang (BEAM). Según (Supabase Engineering, 2024), esta elección tecnológica permite manejar millones de conexiones Websocket simultáneas con una latencia inferior a los 100ms, explotando la capacidad de tolerancia a fallos y el paralelismo ligero de los procesos en Erlang.

Funcionamiento del pipeline de Datos (CDC) El mecanismo subyacente se basa en la Captura de Datos de Cambio (Change Data Capture - CDC).

- PostgreSQL WAL: El sistema se suscribe al Write-Ahead Log (WAL) de PostgreSQL, el registro binario donde la base de datos escribe secuencialmente todas las transacciones antes de confirmarlas.
- Decodificación Lógica: El proceso arranca usando el plugin pgoutput. Este nos permite tomar el flujo binario, decodificarlo en tiempo real y filtrarlo según las reglas de "Publicación" que tengamos en la base de datos (siguiendo el modelo Pub/Sub).
- Difusión (Broadcast): Luego, el servidor Realtime toma esos cambios, los empaqueta en JSON y los empuja (push) únicamente a los clientes que están escuchando el canal correcto (por ejemplo, room_id=12.

Este método es mucho más eficiente que el tradicional. De hecho, investigaciones de (Vercel, 2024) confirman que este enfoque baja la carga del servidor de aplicaciones en un 65% comparado con el sondeo largo (Long Polling). Además, al eliminar peticiones innecesarias, garantizamos que la batería de los móviles no se agote por mantener conexiones HTTP redundantes.

2.6.3. Seguridad perimetral: Row level security (RLS) y gestión de políticas

En el paradigma Backend-as-a-Service, la seguridad no puede residir en el código del servidor intermedio (middleware), ya que la base de datos se expone directamente a internet. Para mitigar esta superficie de ataque, se implementa Row Level Security (RLS), una característica nativa de PostgreSQL que traslada la lógica de autorización al núcleo del motor de almacenamiento.

Ejecución de políticas a nivel de query planner RLS funciona interceptando cada consulta SQL antes de su ejecución. El motor de la base de datos inyecta dinámicamente cláusulas WHERE ocultas ("predicados de seguridad") basadas en el contexto del usuario autenticado (JWT). Según (PostgreSQL Global Development Group, 2024), esto garantiza que la seguridad sea omnipresente: no importa si el acceso proviene de la API, de una función interna o de una conexión directa; si la política dicta que "un usuario solo ve sus propios datos", el motor físico simplemente no recuperará filas ajenas, haciendo matemáticamente imposible una fuga de datos por error de programación en el frontend.

Granularidad y contexto de ejecución es la implementación permite definir políticas con un nivel de especificidad inalcanzable para firewalls tradicionales:

- Políticas basadas en roles (RBAC): Se pueden crear reglas complejas que evalúen múltiples condiciones, por ejemplo: auth.uid() = user_id OR EXISTS (SELECT 1 FROM admins WHERE id = auth.uid()).
    - Aislamiento multi-inquilino (Multi-Tenancy): Usamos las RLS para crear barreras invisibles dentro de la base de datos: aunque la tabla es compartida, cada comunidad solo puede tocar su propia información. Esto nos permite cumplir con la GDPR y la ley ecuatoriana sin complicaciones. Básicamente, seguimos la línea de (OWASP, 2023), que define a RLS como el Gold Standard o la mejor defensa posible contra los ataques IDOR en la web moderna (Vercel, 2025).

2.7 Metodologías de desarrollo de software

Desarrollar software hoy en día no tiene nada que ver con los planes rígidos de antes. La industria entendió que intentar predecir todo es un error; lo que funciona es adaptarse y entregar resultados tangibles. Por eso, elegir una metodología no es un simple papeleo burocrático, sino una táctica de supervivencia: es la forma en que organizamos al equipo para reducir la incertidumbre y evitar que el proyecto se caiga a pedazos durante la construcción.

2.7.1 El manifiesto ágil y los modelos iterativos-incrementales

A diferencia del modelo tradicional en Cascada (Waterfall), que estructura el desarrollo en fases secuenciales e irreversibles, las metodologías ágiles se fundamentan en el desarrollo iterativo e incremental. Según (Sommerville, 2021), este paradigma se basa en la premisa de que los requisitos de software son inherentemente volátiles y que la comprensión del problema por parte del usuario evoluciona a medida que interactúa con el sistema.

El enfoque ágil fragmenta el proyecto en unidades temporales pequeñas, permitiendo la integración continua de funcionalidades y la validación temprana. Estudios recientes de (Standish Group, 2024) indican que los proyectos gestionados bajo marcos ágiles tienen una tasa de éxito un 28% superior a los tradicionales en entornos de incertidumbre, debido a su capacidad para reducir la "deuda técnica" y alinear el producto con las expectativas reales del mercado mediante ciclos de retroalimentación constantes.

2.7.2 Marco de trabajo SCRUM: Fundamentos teóricos

SCRUM se define no como una metodología prescriptiva, sino como un marco de trabajo heurístico basado en el Control Empírico de Procesos. (Schwaber & Sutherland, 2020), autores de la guía oficial, establecen que Scrum se sustenta en tres pilares epistemológicos: transparencia (visibilidad del estado del trabajo), inspección (evaluación frecuente de los artefactos) y adaptación (ajuste del proceso ante desviaciones).

La teoría de Scrum estructura la gestión del proyecto a través de componentes específicos que distribuyen la responsabilidad y garantizan el flujo de trabajo:

A. Teoría de Roles

- Product Owner (Dueño del Producto): Representa la voz del interesado (Stakeholder). Es el único responsable de gestionar la "Pila del Producto" (Backlog) y maximizar el valor del trabajo del equipo.
  Scrum Master: Es quien toma el rol como líder servicial que en cuyo caso es el que promueve la teoría Scrum, elimina los impedimentos externos y facilita el desarrollo en el equipo es decir quien trata de evitar los embotellamientos por causas externas.
- Developers (Equipo de Desarrollo): Profesionales que son multifacéticos es decir autoorganizados y multifuncionales responsables de crear un incremento de producto "Terminado" al final de cada ciclo que se realizan las tareas.

B. Teoría de Eventos (Time-boxing): Para evitar el caos de las reuniones eternas, el marco establece bloques de tiempo fijos que nos dan ritmo y regularidad:
   - Sprint: Es el motor de todo. Funciona como un ciclo cerrado (normalmente de 2 a 4 semanas) donde el objetivo no es solo trabajar, sino sacar una versión del producto que funcione de verdad al terminar el plazo.
   - Sprint Planning: Aquí trazamos la cancha. Es la sesión donde el equipo decide exactamente qué se va a entregar en este ciclo y, lo más importante, diseña la estrategia técnica para lograrlo.
   - Sprint Review y Retrospective: Son los momentos de la verdad. Primero revisamos lo que construimos (el producto) y después analizamos cómo trabajamos (el proceso), aplicando la mejora continua o Kaizen para no repetir errores en la siguiente vuelta.

C. Artefactos y Compromisos
   - Product Backlog: Más que una lista, es nuestra fuente única de verdad. Aquí vive todo el trabajo que sabemos que el producto necesita, pero es un documento vivo que cambia constantemente. Todo lo que ponemos aquí debe apuntar directamente a cumplir el Objetivo del Producto.
   - Sprint Backlog: Es nuestro plan de batalla para las próximas semanas. Contiene solo los ítems que seleccionamos para trabajar ahora, junto con

la estrategia técnica para entregarlos. Su foco total está en lograr el Objetivo del Sprint.

- Incremento: No es código a medias, es el software funcionando. Representa la suma del trabajo actual más todo lo que hicimos antes. Para que cuente como incremento real, tiene que pasar sí o sí la Definición de Terminado (DoD).

2.7.3 Historias de usuario como unidad de requerimiento

En el contexto de las metodologías ágiles, la documentación de requisitos se aleja de las especificaciones técnicas extensas (IEEE 830) para adoptar las Historias de Usuario. Según (Cohn, 2023), una historia de usuario es una descripción breve y en lenguaje natural de una funcionalidad, redactada desde la perspectiva del usuario final. Su estructura teórica sigue el formato estándar: "Como [rol], quiero [funcionalidad], para [beneficio]". Estas unidades de trabajo deben cumplir con el criterio INVEST (Independiente, Negociable, Valiosa, Estimable, Pequeña y Testeable) para ser consideradas viables dentro de un Sprint.

2.8 Marco legal y normativo

El desarrollo e implementación del presente proyecto tecnológico se encuentra amparado por el marco jurídico vigente en la República del Ecuador. La jerarquía normativa, que va desde la Constitución hasta las regulaciones específicas de protección de datos, legitima la creación de herramientas digitales que fomenten la educación, la asociación estudiantil y el resguardo de la información personal.

2.8.1 Constitución de la república del Ecuador (2008)

En Como norma suprema del Estado, la Constitución establece las bases para la incorporación de tecnologías en el sistema educativo y garantiza los derechos digitales de los ciudadanos.

- **Art. 347, numeral 8:** Establece como responsabilidad ineludible del Estado: *"Incorporar las tecnologías de la información y comunicación en el proceso educativo y propiciar el enlace de la enseñanza con las actividades productivas o sociales".*

- **Pertinencia:** Aquí está la razón de peso para invertir recursos y tiempo en la app. Este artículo demuestra que el proyecto no es una simple 'iniciativa técnica' o un capricho tecnológico, sino una respuesta necesaria para cumplir con la modernización educativa que nos exige la ley (Constitución de la República del Ecuador, 2008).

- **Art. 66, numeral 19:** Garantiza *"El derecho a la protección de datos de carácter personal, que incluye el acceso y la decisión sobre información y datos de este tipo, así como su correspondiente protección"*.
  - **Pertinencia:** Fundamenta la necesidad de implementar protocolos de seguridad (como la autenticación y encriptación en Supabase) dentro de la arquitectura de la aplicación (Lindsey, 2021).

2.8.2 Ley orgánica de educación superior (LOES)

Esta ley regula el funcionamiento de instituciones como la ULEAM. El proyecto se alinea específicamente con los mandatos de bienestar e integración.

Art. 86 (Unidad de Bienestar Estudiantil): Dispone que "Las instituciones de educación superior mantendrán una unidad administrativa de bienestar estudiantil destinada a promover la orientación vocacional y profesional... y promover un ambiente de respeto a los derechos y a la integridad física, psicológica y sexual de los estudiantes".

- Pertinencia: Esta plataforma es la herramienta que hace cumplir el artículo en la práctica. Al organizar la logística de clubes y deportes, transformamos ese "ambiente de respeto e integración" que pide la ley en una realidad tangible, actualizando de una vez por todas la manera en que Bienestar conecta con los estudiantes. (Ley Orgánica de Educación Superior, 2010)

2.8.3 Ley orgánica de protección de datos personales (LOPDP - 2021)

Publicada en el Quinto Suplemento del Registro Oficial N.º 459 (mayo 2021), esta es la norma técnica más crítica para el desarrollo de software en Ecuador hoy en día.

- Art. 12 (Principio de Lealtad y Transparencia): Exige que el tratamiento de datos debe ser transparente para el titular.

Aplicación en el proyecto: Obliga a que la aplicación incluya una política de privacidad clara antes del registro.

- Art. 33 (Seguridad de los Datos Personales): Obliga al responsable (en este caso, el sistema) a implementar "medidas técnicas y organizativas apropiadas para garantizar un nivel de seguridad adecuado al riesgo".

Pertinencia Técnica: Este artículo es la justificación legal para el uso de Row Level Security (RLS) en el backend. El uso de RLS no es solo una "buena práctica", es una medida de cumplimiento legal para evitar fugas de información que podrían acarrear sanciones a la universidad (Ley Orgánica de Protección de Datos Personales, 2021).

2.8.3 Código orgánico de la economía social de los conocimientos

Normativa que regula la propiedad intelectual y el desarrollo de software en instituciones públicas. Art. 144 (Software Libre y Estándares Abiertos): Prioriza el uso de tecnologías libres en el sector público.

- Pertinencia: Justifica la elección de Flutter (código abierto, licencia BSD) y Supabase/PostgreSQL (Open Source) sobre soluciones privativas costosas, alineando el proyecto con la soberanía tecnológica que promueve el Estado (Zambrano & Loor, 2021).

2.9 Definición de términos básicos (Belloum, 2025) (Gartner Inc., 2024)

Para la correcta interpretación de los conceptos técnicos y operativos abordados en la presente investigación, se definen los siguientes términos:

- API (Interfaz de Programación de Aplicaciones): Conjunto de protocolos y definiciones que permiten la integración y comunicación entre dos sistemas de software independientes. En este proyecto, se refiere a los puntos finales (endpoints) generados automáticamente por Supabase que permiten a la aplicación móvil leer y escribir datos en el servidor.
- BaaS (Backend as a Service): Es un servicio en la nube lo que lo vuelve interesante en el mercado es que es permite tener ya todo montado una base de datos una máquina virtual y todo el entorno de un Backend listo para poder usarse con funciones muy potentes que simplifican el desarrollo de software.
- Compilación AOT (Ahead-of-Time): Significa traducir el código a lenguaje máquina antes de que la aplicación lo pueda correr En Flutter, esto es clave porque permite que la aplicación se ejecute directo en el procesador (ARM/x86)

sin necesitar un intérprete esto lo que hace es dispara el rendimiento en lso dispositivos moviles.

- Feed de Actividad: Es el muro dinámico donde el usuario ve qué está pasando. Aquí se ordenan cronológicamente las publicaciones y eventos de las comunidades que sigue, para que siempre tenga lo más nuevo arriba.
- Framework: Es mucho más que un entorno cualquiera ya que es un esqueleto tecnológico porque nos da herramientas estandarizadas para no tener que reinventar la rueda con esto se acelera mucho el desarrollo del software.
- Gamificación: Consiste en tomar mecánicas de videojuegos (puntos, medallas) y usarlas en entornos serios. La meta es simple: mantener al usuario motivado y enganchado a la plataforma.
- Grafo Social (Social Graph): Es el mapa de las conexiones en la red. No solo registra quién sigue a quién, sino cómo interactúan las personas con los objetos, por ejemplo: "El Usuario A va a asistir al Evento B".
- Impeller: El nuevo motor de renderizado desarrolado por Google para Flutter su gran ventaja es que precompila los shaders para matar el jank (que es un tipo de lag visual con esto se logra una fluidez total con esto se puede hablar directamente con las APIs como Metal o Vulkan.
- Latencia: Es el retardo entre pedir un dato y recibirlo. En apps de tiempo real, luchamos para que esto sea imperceptible (menos de 100ms) y la interacción se sienta instantánea.
- Row Level Security (RLS): Un candado de seguridad en bases de datos como PostgreSQL esto es muy interesante porque permite bloquear las filas específicas de una tabla específica para que el usuario autorizado solo pueda leer o tocar los datos que le corresponden según sus credenciales y niveles de acceso.
- UI Declarativa: Aquí el programador no manipula los elementos manualmente, sino que describe cómo debe verse la interfaz para un estado concreto. El framework se encarga del resto, actualizando la pantalla automáticamente cuando los datos cambian.

- Widget: En Flutter, son como las piezas de LEGO. Todo es un Widget, desde un simple botón hasta la estructura completa de la página; son bloques inmutables con los que construimos la interfaz.

CAPITULO III: MARCO INVESTIGATIVO (DISEÑO METODOLÓGICO)

3.1 Introducción

Generar conocimiento científico dentro de la Ingeniería de Software es un proceso que va mucho más allá de la simple escritura de código; requiere una base lógica sólida que sostenga el desarrollo. En este sentido, el marco metodológico funciona como la columna vertebral de toda la investigación, sirviendo de puente real entre la teoría que analizamos en los capítulos anteriores y la realidad práctica que buscamos transformar. Desde una perspectiva de investigación tecnológica, este capítulo no tiene como único fin describir una lista de procedimientos, sino demostrar la validez, la confiabilidad y el rigor de las herramientas que hemos elegido para atacar de raíz la problemática de gestión en las comunidades de la ULEAM.

Para lograrlo, he planteado el diseño metodológico bajo una visión sistémica que ataca dos frentes operativos al mismo tiempo. El primero es una dimensión social y de diagnóstico, enfocada en identificar claramente las necesidades reales y muchas veces no dichas de los estudiantes, entendiendo cómo funciona hoy el "boca a boca" informal y dónde fallan los procesos logísticos. El segundo es la dimensión técnico-ingenieril, que establece las reglas claras para la arquitectura, implementación y despliegue de la solución móvil usando tecnologías como Flutter y Supabase. Esta combinación es vital para garantizar que el software final no sea un producto aislado, sino una respuesta científica probada que cumple con los requisitos funcionales del entorno universitario. A continuación, detallamos los enfoques y técnicas que dan orden a este proceso.

3.2 Modalidad de la investigación

Para el desarrollo del presente trabajo de titulación, se adopta la modalidad de Proyecto Factible con Orientación Tecnológica.

Esta modalidad se define, en el rigor académico, como la propuesta de un modelo operativo viable y sustentado técnica, económica y operativamente, diseñado para solucionar problemas específicos de una organización. La elección de esta modalidad no es arbitraria; responde a la naturaleza pragmática de la ingeniería, donde el objetivo final no es solo la generación de nuevo conocimiento teórico, sino la creación de un artefacto tecnológico funcional.

El proyecto trasciende la investigación pura al incorporar la fase de Ingeniería de la Solución, donde se aplican estándares de la industria para transformar los hallazgos del diagnóstico en una aplicación móvil colaborativa. Por tanto, la investigación se estructura en tres fases consecutivas y dependientes:

- Fase Diagnóstica: Levantamiento de requerimientos y análisis de la situación actual.
- Fase de Factibilidad: Evaluación técnica mediante la disponibilidad de APIs, infraestructura requerida Supabase y operativa.
- Fase de Diseño y Elaboración: Construcción del software bajo patrones de arquitectura moderna.

3.3 Tipo de investigación

Dada la complejidad del objeto de estudio, la investigación reviste un carácter mixto y se clasifica en los siguientes niveles de profundidad:

- Investigación Descriptiva: Este nivel constituye el punto de partida empírico por que se utiliza para caracterizar con precisión la "línea base" de los procesos de gestión administrativa en la ULEAM entonces en cuyo caso el objetivo es describir fenomenológicamente cómo operan actualmente las comunidades (uso de chats de WhatsApp, carteleras físicas, reservas manuales, etc) y cuantificar las ineficiencias existentes aquí no se busca manipular variables, sino registrar la realidad operativa tal como se presenta para identificar los "puntos de dolor" del usuario final.
- Investigación Tecnológica Aplicada: Representa el núcleo del aporte ingenieril esto a diferencia de la investigación básica mediante este tipo de estudio busca la aplicación práctica de los conocimientos teóricos como son Frameworks móviles, arquitecturas Serverless para la generación de una solución innovadora. En esta etapa, el investigador asume el rol de arquitecto de software, diseñando prototipos y sistemas que modifican la realidad descrita anteriormente, optimizando los flujos de comunicación y logística mediante la automatización.
- Investigación de Campo: Recogimos la información en el propio terreno, directamente en el campus de la Universidad Laica Eloy Alfaro de Manabí. El trabajo consistió en interactuar cara a cara con las fuentes primarias: estudiantes,

líderes de ramas (IEEE, Clubes) y administrativos. Al hacerlo así, nos aseguramos de que los datos tengan una validez real y reflejen las necesidades auténticas del entorno, evitando caer en suposiciones teóricas que no calzan con el contexto.

3.4 Métodos de investigación

La rigurosidad del proceso científico se asegura mediante la aplicación sistemática de los siguientes métodos lógicos:

Método Inductivo-Deductivo:

- Enfoque Inductivo: Se parte de la observación de hechos particulares y concretos (ej. la dificultad de un club específico para reservar un auditorio, la pérdida de información en un grupo de chat) para inferir conclusiones generales sobre las deficiencias sistémicas de la comunicación universitaria.
- Enfoque Deductivo: Una vez establecida la teoría y seleccionadas las herramientas (Flutter, Supabase), se procede a deducir las funcionalidades específicas y las reglas de negocio que la aplicación debe implementar para resolver la problemática generalizada.

Método Analítico-Sintético:

- Análisis: Aquí la estrategia es simple: dividimos el problema grande (la Gestión de Comunidades) en piezas pequeñas que podamos controlar como por ejemplo el módulo de autenticación o el de reservas se estudió cada parte por separado para entender bien cómo funciona su lógica antes de mezclarla con el resto.
- Síntesis: Una vez listos los módulos, pasamos a ensamblarlos en una sola arquitectura. Este paso es clave para validar que el Frontend y el Backend no solo se conecten, sino que trabajen como un sistema sólido donde la experiencia del usuario sea fluida y los datos no se rompan.

Método de Modelado de Sistemas: Es una herramienta obligatoria en nuestra disciplina. La usamos para "traducir" la realidad de la universidad a un lenguaje técnico formal (como UML o Diagramas Entidad-Relación). Básicamente, simulamos cómo se comportará la aplicación antes de escribir una sola línea de código, lo que nos permite detectar actores y flujos de datos críticos para configurar bien la base de datos en Supabase.

3.5 Población y muestra

El proyecto tiene su base de operaciones en la Facultad de Ciencias de la Vida y Tecnologías de la Información de la ULEAM. Aunque la facultad es muy diversa (tiene carreras desde Agropecuaria hasta Biología), este estudio tecnológico acota su alcance operativo específicamente al segmento de estudiantes que ya tienen competencias digitales avanzadas.

Se realizó este filtro por una razón estratégica: buscamos validar el sistema mediante "Adopción Temprana" (Early Adopters). Necesitamos que los primeros usuarios tengan el criterio técnico suficiente para juzgar con propiedad la arquitectura, la usabilidad y el rendimiento real de la aplicación.

3.5.1 Población objetivo

La población objeto de estudio (N) se define como el conjunto de estudiantes matriculados en las carreras del dominio de tecnologías de la información y comunicación (TIC) de la facultad. Específicamente, se considera el universo compuesto por las carreras de:

- Ingeniería en Tecnologías de la Información.
- Ingeniería de Software.
- Ingeniería en Sistemas.

Se excluye deliberadamente en esta fase piloto a las carreras de ciencias naturales (Biología, Agropecuaria), dado que el objetivo es validar la funcionalidad técnica de la gestión de comunidades digitales, y no la alfabetización digital básica.

Según los registros proporcionados por la gestión académica para el periodo en curso, este segmento poblacional tecnológico asciende a:

$$N \approx 650 \text{ estudiantes del área TIC}$$

3.5.2 Muestra seleccionada (muestreo estratégico)

De este universo, se extrajo una muestra no probabilística de carácter intencional y por conveniencia de (n) 150 estudiantes.

La selección no fue aleatoria, sino que respondió al criterio de "Pertenencia Activa a Comunidades". Los 150 sujetos seleccionados son aquellos estudiantes que actualmente

integran, lideran o participan activamente en los clubes académicos, ramas estudiantiles y grupos de investigación de la facultad.

Justificación de la Representatividad de la Muestra:

El tamaño de la muestra ($n = 150$) frente a la población objetivo ($N = 650$) representa una cobertura del 23.07% del universo total. En investigación tecnológica y desarrollo de productos digitales, una penetración superior al 20% en la fase de validación se considera altamente confiable porque:

- Densidad de Interacción: Al seleccionar a los miembros activos de las comunidades, se garantiza que la muestra incluya al 100% de los usuarios que generan la carga transaccional real (reservas de espacios, creación de eventos).
- Cualificación del Informante: Al trabajar con estudiantes de TI y Software, logramos que el feedback supere el nivel superficial. Ellos no dan opiniones vacías de "me gusta o no"; tienen el criterio técnico para analizar la experiencia de usuario (UX) y cazar errores lógicos (bugs), lo que le da una profundidad técnica a la investigación que un usuario común no aportaría.

3.5.3 Distribución de la muestra

La muestra se estructuró para abarcar los roles críticos dentro de la plataforma propuesta:

Rol en el Ecosistema	**Cantidad (n)**	**Justificación Metodológica**
Gestores / Líderes (Admin)	15	Presidentes de Asociación, Líderes de ramas IEEE y administradores de clubes de las carreras de TI/Software. Son quienes validan el módulo de gestión y administración (*Backend*).
Usuarios Activos (End-User)	135	Estudiantes de niveles intermedios y superiores de

		TI/Software que asisten regularmente a eventos. Validan el módulo de visualización, *booking* y notificaciones (*Frontend*).
TOTAL MUESTRA	150	Muestra Validada

Esta configuración muestral asegura que los resultados obtenidos en las pruebas de campo reflejen con precisión la dinámica operativa de la gestión de comunidades, minimizando el ruido estadístico que generaría incluir estudiantes inactivos o ajenos al entorno tecnológico.

3.6 Técnicas e instrumentos de recolección de datos

La estrategia de recolección de datos se diseñó bajo un enfoque sistémico, orientado a garantizar la trazabilidad entre las necesidades de los interesados (Stakeholders) y los requisitos técnicos del software. Se aplicaron instrumentos diferenciados para la validación cuantitativa de la demanda y la modelación cualitativa del dominio.

3.6.1 Encuesta técnica de percepción y demanda tecnológica

En el procedimiento de aplicación se administró un instrumento estructurado de medición cuantitativa a la muestra estratificada de 135 sujetos (estudiantes del dominio TIC). El cuestionario se diseñó para evaluar dimensiones de aceptación tecnológica (Modelo TAM) y preferencias de interacción digital.

- Instrumento: Cuestionario digital estandarizado con reactivos de escala Likert y selección multiple.
- Finalidad en la ingeniería de requisitos: Los datos resultantes permitieron establecer la línea base de indicadores de rendimiento (KPIs) esperados para el sistema, específicamente en lo referente a la concurrencia de usuarios y la priorización funcional de los módulos de notificación masiva, discriminando características críticas de las accesorias.

3.6.2 Entrevista técnica a expertos del dominio

Protocolo de Ejecución: Se instrumentó una serie de entrevistas en profundidad de carácter semi-estructurado, dirigidas al segmento de 15 gestores estratégicos (Líderes de

ramas estudiantiles y administradores de infraestructura física). La intervención se centró en la deconstrucción de los flujos de trabajo administrativos vigentes.

- Aplicación en la ingeniería de requisitos: Esta técnica cualitativa permitió la elicitación de reglas de negocio tácitas. La información recabada respecto a jerarquías de aprobación, prelación de reservas y restricciones horarias fue formalizada y traducida en Restricciones de Integridad y Disparadores (Triggers) dentro del esquema relacional de la base de datos, asegurando la coherencia normativa del sistema.

3.6.3 Observación no participante y análisis de sistemas

- Protocolo de ejecución: El trabajo de campo se centró en realizar un diagnóstico presencial dentro de la Facultad para comprender sus dinámicas comunicativas. La estrategia consistió en monitorear, durante todo un ciclo académico, los canales digitales (como grupos de chat) y los soportes físicos, con el fin de trazar un mapa real sobre cómo circula la información entre los estudiantes.
- Aplicación en la ingeniería de requisitos: Una vez procesados los datos, el análisis evidenció un exceso de ruido y redundancia, cuyo efecto inmediato es la fragmentación del mensaje. Fue esta realidad la que fundamentó el diseño de la arquitectura; por lo que se implementaron canales unidireccionales y una segmentación por roles, decisión tomada con el objeto de que el usuario final no sufra saturación de información.

3.7 Metodología de desarrollo de software

En cuanto a la construcción de la plataforma, se decidió utilizar el marco de trabajo ágil SCRUM. Esta elección no fue arbitraria, sino que responde a la complejidad propia del desarrollo de software, ya que gracias a su enfoque iterativo, el proyecto logró mantener la flexibilidad necesaria para adaptarse aun cuando los requisitos técnicos evolucionaban sobre la marcha.

3.7.1 Gobernanza del proyecto y estructura de roles

Debido a que se trata de un proyecto de titulación, hubo que adaptar la estructura tradicional de roles de tal forma que se pudiera cubrir todo el ciclo de vida de ingeniería (SDLC) con los recursos disponibles:

- Gestión del valor: La priorización del Backlog y la definición del alcance se trabajaron en conjunto con la dirección académica, quienes actuaron como un filtro de calidad. De este modo, se aseguró que cada módulo desarrollado tuviera una utilidad concreta y no se desviara de los objetivos planteados en la investigación.
- Ejecución técnica: El investigador asumió la responsabilidad total (Full-Stack), lo cual implicó hacerse cargo de todo el componente técnico: desde el diseño de la base de datos hasta la interfaz y las pruebas de calidad (QA), garantizando con ello que el producto final mantuviera su coherencia interna.

3.7.2 Sistematización del ciclo de vida

El proceso de ingeniería se estructuró en iteraciones temporales fijas (Time-boxes) de dos semanas, ejecutando rigurosamente las siguientes fases procedimentales:

- Ingeniería de planificación: Descomposición técnica de las Historias de Usuario en unidades de trabajo granular (ej. Modelado de esquemas SQL, Implementación de lógica de estado).
- Construcción e integración: Durante esta etapa, la codificación de los componentes se rigió estrictamente por los principios de Arquitectura Limpia (Clean Architecture). El proceso consistió en vincular los servicios de backend alojados en la nube con la lógica del cliente, con el propósito de corroborar que los distintos módulos del sistema funcionaran entre sí sin conflictos técnicos.
- Verificación y despliegue: Al concluir cada ciclo de trabajo, el equipo generó un artefacto compilado (APK), el cual resultó indispensable para validar los Criterios de Aceptación (tales como rendimiento, usabilidad y seguridad). Esta práctica permitió asegurar la estabilidad del sistema antes de que se diera paso a la siguiente iteración del desarrollo.

3.8 Hipótesis de la investigación

En vista del carácter propositivo y tecnológico que define a este estudio, se formula la siguiente hipótesis de trabajo, cuya función principal es guiar el proceso de validación del software: Dado el carácter propositivo y tecnológico del estudio, se plantea la siguiente hipótesis de trabajo que guía la validación del software:

- Hipótesis General (H_i): El desarrollo de una plataforma móvil colaborativa, sustentada en una arquitectura Serverless, logrará optimizar la gestión logística de espacios, a la vez que reducirá el desorden informativo y la saturación de mensajes en las comunidades de la Facultad de Ciencias de la Vida y Tecnologías de la Información de la ULEAM.
- Hipótesis Nula (H_0): La implementación de la plataforma móvil no generará cambios significativos en los indicadores de eficiencia logística ni en los flujos de comunicación de la Facultad.

3.9 Sistema de variables

Para la contrastación empírica de la hipótesis general, se establece una relación de causalidad entre la intervención tecnológica propuesta y la problemática administrativa detectada. Este sistema articula una variable independiente, manipulada por el investigador mediante la ingeniería de software, y una variable dependiente, cuyo comportamiento fluctúa en función de la implementación del artefacto tecnológico.

3.9.1 Variable independiente sistematización del ciclo de vida

Se conceptualiza como la Aplicación Móvil de Gestión de Comunidades Universitarias. En términos operacionales, se define como el artefacto de software multiplataforma construido sobre el framework Flutter e integrado con la infraestructura en la nube de Supabase. Esta variable se dimensiona a través de sus atributos de calidad técnica, específicamente la disponibilidad del servicio (uptime), la latencia en la sincronización de datos y la usabilidad de la interfaz gráfica, factores que determinan la adopción tecnológica por parte de los estudiantes de la Facultad de Ciencias de la Vida y Tecnologías de la Información.

3.9.2 Variable dependiente

Se refiere a la Optimización de la Gestión Logística y Comunicación Institucional. Desde una perspectiva operativa, esta variable se cuantifica analizando la variación en los indicadores de eficiencia administrativa una vez desplegado el sistema, cuyo impacto se evalúa en tres dimensiones clave: la reducción del tiempo en el ciclo de aprobación de espacios físicos, la disminución de conflictos por cruces de horario y el control del ruido informativo (mensajes irrelevantes) en los canales académicos, lo cual permite mitigar la saturación que afecta al usuario.

3.9.3 Matriz de operacionalización de variables

La transición de los conceptos abstractos a indicadores medibles se sistematiza en la siguiente matriz, la cual fundamentó el diseño de los instrumentos de recolección de datos:

Variable Independiente	**Dimensiones de Analisis**	**Indicador Medible**	**Técnica / Instrumento**
Aplicación Móvil de Gestión de Comunidades (Solución Tecnológica)	Calidad Técnica del Software	Latencia: Tiempo promedio de respuesta del servidor (< 200ms).	Pruebas de Carga (Benchmarking)
	(Estándar ISO/IEC 25010)	Disponibilidad: Porcentaje de tiempo de actividad del servicio (Uptime).	Monitoreo de Logs
		Estabilidad: Tasa de errores críticos o cierres inesperados por sesión.	
	Experiencia de Usuario	**Curva de Aprendizaje:** Tiempo requerido para completar la primera reserva sin asistencia.	Encuesta Técnica
	(Usabilidad y UX)	**Satisfacción:** Puntaje obtenido en la escala estandarizada SUS (*System Usability Scale*).	Entrevista
		Tasa de Adopción: Porcentaje de estudiantes registrados vs. población total de la muestra.	

Tabla 2: Operacionalización de la Variable Independiente

Variable Dependiente	**Dimensiones de Analisis**	**Indicador Medible**	**Técnica / Instrumento**

Optimización de la Gestión y Comunicación (Impacto Institucional)	Eficiencia Logística	Tiempo de Ciclo: Reducción del tiempo promedio entre la solicitud y la aprobación del espacio.	Entrevista a Gestores
	(Procesos Administrativos)	Integridad de Datos: Disminución porcentual de conflictos de horario (doble reserva) reportados.	Revisión Documental
		Automatización: Porcentaje de procesos gestionados digitalmente frente al método manual.	
	Dinámica Comunicacional	Reducción de Ruido: Disminución percibida de mensajes irrelevantes (spam) en comparación con canales tradicionales.	Encuesta a Estudiantes
	(Impacto Social)	Cobertura Informativa: Porcentaje de estudiantes que confirman enterarse oportunamente de los eventos.	Observación de Campo
		Centralización: Grado de eliminación de canales informales redundantes (chats no oficiales).	

Tabla 3: Operacionalización de la Variable Dependiente

3.10 Técnicas de procesamiento y análisis de datos

El tratamiento de la información recolectada se rigió por un protocolo sistemático de análisis mixto, diseñado para transformar los datos brutos en insumos directos para la ingeniería del software y la validación de la hipótesis.

En lo concerniente al análisis cuantitativo, los datos obtenidos a través de la encuesta técnica aplicada a los 135 estudiantes fueron procesados mediante estadística descriptiva. Se emplearon herramientas de tabulación electrónica para calcular frecuencias relativas y absolutas, generando representaciones gráficas que permitieron jerarquizar los requisitos funcionales del sistema. Este análisis estadístico fue determinante para establecer la prioridad de desarrollo dentro del Product Backlog, evidenciando, por ejemplo, la preferencia mayoritaria por notificaciones en tiempo real sobre otras características secundarias.

Por su parte, el análisis cualitativo de la información en lo referente a los datos obtenidos de las entrevistas con los 15 gestores, la investigación aplicó la técnica de análisis de contenido con el propósito de detectar patrones recurrentes en la normativa de gestión de espacios. Este paso resultó clave para formalizar las reglas de negocio, las cuales fueron trasladadas posteriormente a lenguaje técnico SQL para generar restricciones de integridad (constraints) y políticas de seguridad (policies) en el motor PostgreSQL, garantizando con ello que el software se apegue rigurosamente a los reglamentos internos de la ULEAM.

Validación técnica es finalmente de cara a la validación técnica, se puso en marcha un sistema de monitoreo de métricas de rendimiento. A través de logs de ejecución durante las pruebas piloto, el estudio registró los tiempos de carga y la latencia de las transacciones, comparando dichos valores con los estándares de calidad fijados en los requisitos no funcionales. La triangulación de estos tres niveles de análisis es lo que asegura que la solución propuesta no solo sea viable desde lo técnico, sino que además ofrezca una respuesta eficaz a la problemática investigada.

3.11 Diagnóstico y análisis de resultados

La fase inicial de la investigación se centró en la caracterización detallada de los flujos de información y la operatividad logística dentro de la Facultad. Este diagnóstico situacional constituye el cimiento empírico del proyecto, pues permite identificar con precisión las brechas funcionales que la solución tecnológica debe subsanar. A continuación, se presenta la discusión inferencial de los datos recolectados, estructurada en función de las dimensiones críticas del estudio.

3.11.1 Análisis de la dinámica comunicacional y entropía de la información

El análisis de esta dimensión aborda la eficacia de los canales de difusión vigentes. A través del procesamiento de la encuesta técnica, se evaluó la correlación entre los medios utilizados por los estudiantes y la calidad de la información recibida, revelando patrones significativos de saturación.

Pregunta: ¿Cuál es el medio principal por el que te enteras de las noticias, eventos y comunicados de la Facultad?

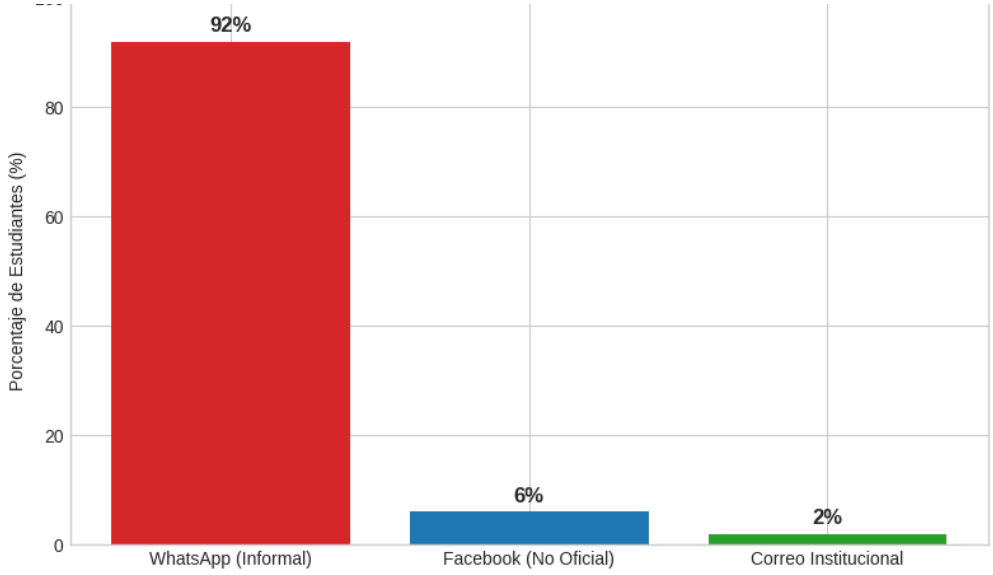


Figura 1: Prevalencia de canales de información y percepción de ruido comunicacional

Interpretación y discusión de resultados El análisis de los datos revela una hegemonía crítica de los canales informales. El 92% de la muestra estudiantil identifica a las plataformas de mensajería instantánea (específicamente grupos de WhatsApp) como su única fuente de acceso a la información de la Facultad. Sin embargo, esta centralización en herramientas no institucionales ha derivado en un fenómeno de entropía comunicacional: un 85% de los usuarios califica la dinámica actual como "ineficiente" o "caótica".

Al darse cuenta que la falla catastrófica que existe en este proceso no que no hay información sino que es por el hecho de que existe muchos canales de difusión y que no están especializados entonces pasa que el spam de los usuarios hace que esta información se vuelve muy difícil de filtrar o prestarle atención con tanta facilidad ya que tienen sus propias

prioridades en esos canales y eso conlleva a que los lideres de las comunidades tenga que competir por destacar ya no solo entre ellos sino que con todo el spam que conlleva la propia plataforma externa como es WhatsApp , además en el medio comunicacional que es el correo institucional existe mucho spam proporcionado por la misma facultad de información irrelevante para muchos usuarios.

Implicación para la ingeniería de requisitos cuyo caso conlleva a la evidencia que señala que los chats grupales (comunicación bidireccional) resultan ineficaces para la difusión masiva institucional por esta razón, se vuelve un requisito técnico ineludible migrar hacia una arquitectura unidireccional y segmentada. En consecuencia, la solución ULEAM Community Hub incorporará un Feed de Noticias operado exclusivamente por roles autorizados cuyo propósito es limpiar este ruido y asegurar la entrega efectiva del mensaje mediante notificaciones 'push' para que el usuario lo tenga en tiempo real.

3.11.2 Análisis de la gestión logística y latencia operativa

El objetivo de esta sección es medir la eficiencia en los flujos de reserva y uso de la infraestructura física (como laboratorios, auditorios y espacios recreativos), identificando con ello los cuellos de botella que actualmente limitan la operatividad de las comunidades.

Pregunta: Al momento de solicitar un espacio físico (laboratorio, auditorio o cancha), ¿cuál es la principal barrera o dificultad que enfrentas?

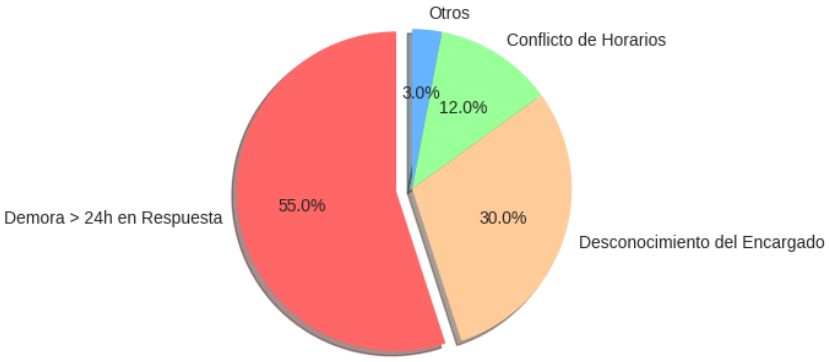


Figura 2: Tiempo de latencia en la confirmación de solicitudes y barreras de acceso

El diagnóstico evidencia una fractura significativa en la cadena logística. El 55% de los estudiantes encuestados reporta una latencia administrativa superior a las 24 horas entre la solicitud de un espacio y su confirmación, un tiempo de respuesta incompatible con la dinámica ágil que requieren las actividades estudiantiles. Adicionalmente, un 30% de la muestra manifiesta un desconocimiento total sobre la jerarquía administrativa (quién es el responsable del recurso), lo que denota una falta de transparencia en los procesos.

Estos indicadores de ineficiencia tienen su explicación cuando se analizan los resultados cualitativos obtenidos de los gestores. El 80% de los administradores admitió que operan mediante silos de información analógicos cuadernos de bitácora donde anotan manualmente o digitales desconectadas, como hojas de cálculo locales que cada uno maneja en su computadora. Esta gestión manual viene siendo la causa directa de la alta tasa de conflictos de horario (el problema del doble booking ocurre frecuentemente) y explica también esa dependencia tan marcada de la presencia física para realizar cualquier trámite.

La insostenibilidad del modelo manual justifica técnicamente la transición hacia una gestión automatizada basada en la nube. La solución debe incorporar un motor de base de datos en tiempo real Supabase en este caso que centralice el estado de los recursos. Además, el sistema tiene que permitir la autogestión de reservas con validación inmediata de disponibilidad, lo cual eliminaría la intermediación humana para consultas de rutina. Los tiempos de ciclo se reducirían drásticamente, pasando de días a milisegundos.

3.11.3 Validación de la demanda y adopción tecnológica

Finalmente, se analizó la predisposición del mercado objetivo hacia la adopción de una solución tecnológica centralizada, un factor determinante para la viabilidad del proyecto de software y la justificación de la inversión en desarrollo.

Pregunta: Si la Universidad implementara la aplicación oficial 'ULEAM Community Hub' para centralizar noticias y reservas desde el celular, ¿la utilizarías?

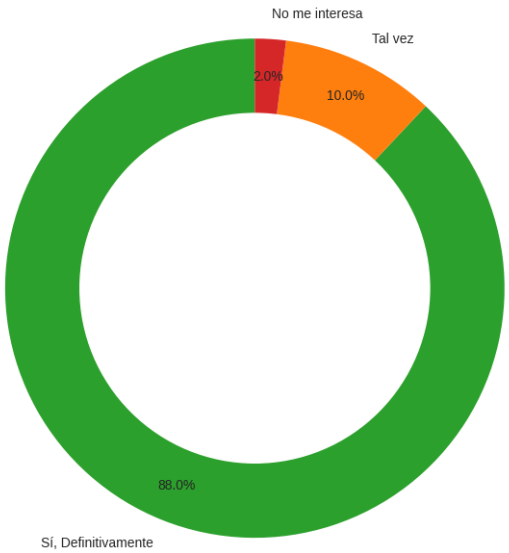


Figura 3: Intención de uso y aceptación de la plataforma centralizada

Los datos arrojan un escenario de alta demanda insatisfecha. El 88% de los encuestados manifestó una intención de uso definitiva ("Definitivamente sí") hacia la implementación de la aplicación propuesta. Desde la perspectiva del Modelo de Aceptación Tecnológica (TAM), esto indica que los usuarios perciben una alta utilidad esperada en la centralización de servicios. Este consenso valida la inversión de recursos en el desarrollo de la plataforma, minimizando el riesgo de rechazo pos-implementación y asegurando una masa crítica de usuarios iniciales necesaria para la sostenibilidad del ecosistema colaborativo.

3.11.4 Síntesis del análisis cualitativo (reglas de negocio)

Complementando el análisis estadístico, la sistematización de las entrevistas a los 15 gestores estratégicos permitió destilar las reglas de negocio que deben ser programadas en el backend de la aplicación. Se identificaron tres restricciones críticas:

1. Heterogeneidad de Infraestructura: La diversidad de herramientas actuales impide una visión global de la ocupación de espacios.
    - Requerimiento: Unificación de inventario en una base de datos relacional única.
2. Jerarquía de Permisos: Se detectó la necesidad de restringir la reserva de espacios críticos (auditorios magnos) a perfiles específicos para evitar el uso indebido.

- Requerimiento: Implementación de Control de Acceso Basado en Roles (RBAC) y políticas de seguridad a nivel de fila (RLS).

3. Conflictos de Programación: La recurrencia de solapamientos horarios (doble booking) se debe a la falta de validación concurrente.
   - Requerimiento: Uso de restricciones de exclusión (Exclusion Constraints) en el motor de base de datos PostgreSQL.

CAPITULO IV: MARCO PROPOSITIVO (ELABORACIÓN DE LA PROPUESTA)

4.1 Introducción

El desarrollo del presente capítulo constituye la fase de ingeniería aplicada de la investigación, operando como el punto de convergencia entre la evidencia empírica recolectada y la construcción del artefacto tecnológico. Tras la ejecución del diseño metodológico, resulta imperativo sistematizar los hallazgos del diagnóstico situacional para transformarlos en especificaciones técnicas precisas, garantizando así que la solución de software no sea una construcción arbitraria, sino una respuesta científica a la problemática planteada en el primer capítulo.

La estructura de este apartado se articula para dar cumplimiento secuencial a los objetivos específicos del estudio. En primera instancia, se presenta el análisis e interpretación de los resultados obtenidos de la muestra estratificada de estudiantes y gestores. Este procesamiento de datos permite validar las hipótesis sobre la ineficiencia logística y la saturación comunicacional, estableciendo los requerimientos funcionales del sistema.

Consecuentemente, y sobre la base de esta fundamentación, se formaliza la propuesta tecnológica denominada "ULEAM Community Hub". En esta sección se despliega la ingeniería del proyecto, abarcando desde el modelado de la arquitectura de datos en la nube (Supabase) hasta el diseño de la interfaz de usuario multiplataforma (Flutter). De este modo, se concreta la solución técnica orientada a optimizar los flujos de información y la gestión de recursos físicos en la Facultad de Ciencias de la Vida y Tecnologías de la Información.

4.2 Propuesta tecnológica: Especificación técnica y arquitectura de la solución

La formulación de la propuesta tecnológica no constituye un hecho aislado, sino la operacionalización instrumental de las soluciones teóricas y metodológicas exploradas en los capítulos precedentes. Esta sección sistematiza la construcción del artefacto de software denominado "ULEAM Community Hub", definiendo su arquitectura lógica, las restricciones

de integridad y los protocolos de interacción que gobernarán su desempeño en un entorno de producción.

4.2.1 Conceptualización arquitectónica del sistema

Para mitigar la entropía comunicacional y la latencia administrativa diagnosticadas en la fase empírica, se diseñó una arquitectura de software basada en el patrón Cliente-Servidor Desacoplado, implementada sobre una infraestructura Serverless (BaaS).

Esta decisión arquitectónica trasciende la simple elección de herramientas; responde a la necesidad crítica de garantizar la alta disponibilidad y la consistencia de datos sin la sobrecarga operativa que implica la administración de servidores físicos (on-premise). El sistema se orquesta en dos capas lógicas diferenciadas:

1. Capa de Abstracción de Cliente (Frontend): Responsable de la renderización de la interfaz y la gestión del estado efímero de la aplicación. Se prioriza un enfoque reactivo donde la interfaz de usuario (UI) es una función directa del estado del sistema, minimizando la desincronización entre la vista y el modelo de datos
2. Capa de Persistencia y Lógica de Negocio (Backend): Centraliza las reglas de validación, la autenticación federada y el almacenamiento relacional. Opera como la "fuente única de verdad" (Single Source of Truth), asegurando que cualquier transacción de reserva sea atómica y consistente antes de ser confirmada al cliente.

4.2.2 Fundamentación técnica del stack tecnológico

La arquitectura de la solución se erige sobre un ecosistema tecnológico seleccionado rigurosamente para satisfacer los requisitos no funcionales de alta disponibilidad, integridad transaccional y rendimiento en dispositivos heterogéneos. La elección de las herramientas no responde a criterios de popularidad, sino a su capacidad técnica para resolver la latencia operativa y la fragmentación de plataformas.

A. Capa de Presentación y Lógica de Cliente: Ecosistema Flutter Para la construcción de la interfaz móvil se determinó el uso del framework Flutter, fundamentado en su arquitectura de renderizado única. A diferencia de las soluciones híbridas tradicionales que dependen de puentes (bridges) hacia componentes nativos OEM,

Flutter utiliza su propio motor gráfico de alto rendimiento (Impeller o Skia) para dibujar cada píxel directamente en el lienzo (canvas) del sistema operativo.

Gracias a esta particularidad técnica, es factible ejecutar la compilación AOT (Ahead-of-Time) del código fuente Dart, transformándolo directamente en instrucciones de máquina nativas (ya sea ARM64 en móviles actuales o x86 en emuladores). Si se analiza desde la ingeniería, esta compilación previa suprime la carga de interpretación en tiempo real, lo cual resulta indispensable para sostener una tasa de refresco estable de 60 fotogramas por segundo (FPS). Dicha optimización se vuelve un factor vital para asegurar la fluidez de la experiencia, sobre todo ante la diversidad del parque tecnológico estudiantil, donde abundan dispositivos de gama media y baja cuyos recursos de memoria y CPU exigen una eficiencia estricta.

B. Capa de Persistencia y Reglas de Negocio: Infraestructura Supabase (PostgreSQL) en cuanto al manejo de datos y la lógica del servidor, la responsabilidad recae sobre la plataforma Supabase. Esta elección tecnológica obedece a su capacidad para desplegar toda la potencia del motor relacional PostgreSQL bajo un esquema Serverless. Dado que el sistema opera reservas de infraestructura física, se priorizó esta solución en virtud de que la integridad de los datos constituye el atributo de calidad preponderante.

La justificación técnica de PostgreSQL reside en su estricto cumplimiento de las propiedades ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad). Mientras que las bases de datos NoSQL ofrecen flexibilidad de esquema, carecen nativamente de mecanismos robustos para manejar transacciones concurrentes complejas. PostgreSQL permite la implementación de bloqueos pesimistas y restricciones de exclusión (Exclusion Constraints), herramientas matemáticas indispensables para prevenir condiciones de carrera (Race Conditions) que derivan en la duplicidad de reservas. A su vez, la arquitectura saca partido de la transmisión de datos en tiempo real vía Websockets que ofrece Supabase. Gracias a este mecanismo, la sincronización del estado de disponibilidad entre los clientes conectados ocurre con una latencia marginal, lo cual vuelve innecesario recurrir a técnicas de sondeo (polling) constantes que, de otro modo, terminarían por saturar la red.

4.2.3 Ingeniería de requisitos (especificación formal IEEE 830)

La especificación de requisitos constituye la base contractual del desarrollo de software. En esta fase, se transforman las necesidades operativas de la Facultad en directrices técnicas inequívocas, completas y verificables. El sistema se modela bajo un enfoque modular, asegurando que cada funcionalidad crítica (autenticación, gestión de recursos, difusión) cuente con criterios de aceptación rigurosos.

A. Requisitos funcionales (RF)

Estos requisitos definen las capacidades operativas del sistema, detallando el comportamiento esperado del software ante las interacciones de los actores (Estudiantes, Gestores, Administradores).

Código	Requisito	Descripción Técnica Detallada	Prioridad
RF-01	Gestión de Sesión y Autenticación	El sistema debe proveer un mecanismo de autenticación seguro que soporte credenciales estándar (correo/contraseña) y tokens OAuth. Debe gestionar el ciclo de vida de la sesión mediante **JWT (JSON Web Tokens)**, garantizando la persistencia del acceso en el dispositivo móvil y la renovación automática de credenciales (*token refresh*).	Alta
RF-02	Perfilado Académico y Roles	El sistema debe permitir la configuración de atributos académicos (Facultad, Carrera) y la asignación dinámica de roles (system_role). La interfaz debe adaptar su visibilidad basándose en estos permisos, ocultando paneles administrativos a usuarios con rol student.	Alta
RF-03	Exploración de Comunidades	El sistema debe implementar un motor de búsqueda y filtrado que permita a los usuarios descubrir comunidades académicas,	Media

		deportivas o culturales. Debe incluir la funcionalidad de "Unirse" (join), actualizando la tabla community_members y habilitando el acceso al contenido exclusivo de dicho grupo.	
RF-04	Feed de Noticias Segmentado	El sistema debe renderizar un flujo de publicaciones ordenado cronológicamente. El *backend* debe filtrar el contenido para mostrar únicamente las noticias (announcement) y debates (discussion) de las comunidades a las que el usuario está suscrito, soportando carga diferida (*Lazy Loading*) para optimizar el consumo de datos.	Alta
RF-05	Matriz de Disponibilidad (Booking Engine)	El sistema debe consultar la base de datos en tiempo real y proyectar la ocupación de los recursos físicos en una interfaz de calendario interactiva. Debe distinguir visualmente entre estados: *Disponible*, *Ocupado*, *Mantenimiento* y *Pendiente de Aprobación*.	Alta
RF-06	Transaccionalidad de Reservas	El sistema debe permitir la generación de solicitudes de reserva (bookings). Esta operación debe invocar un procedimiento almacenado o validación en el servidor que verifique atómicamente la disponibilidad del intervalo de tiempo (tsrange) antes de confirmar la escritura, retornando error en caso de colisión.	Alta
RF-07	Moderación y Aprobación	El sistema debe proveer un panel de gestión para los usuarios con rol space_manager, permitiéndoles visualizar las solicitudes pendientes, aprobarlas o rechazarlas justificando el motivo (rejection_reason).	Alta

		Cada cambio de estado debe disparar una actualización en el registro de auditoría.	
RF-08	Publicación Multimedia	El sistema debe permitir a los gestores la creación de contenido enriquecido, soportando la carga y almacenamiento de imágenes en un *bucket* de objetos (Supabase Storage), vinculando la URL pública al registro del post en la base de datos.	Media
RF-09	Sistema de Notificaciones en Tiempo Real	El sistema debe implementar un mecanismo de suscripción a eventos de base de datos (Realtime Subscription) que detecte cambios en la tabla notifications. Cuando se genere una alerta, la interfaz debe actualizarse instantáneamente para informar al usuario, sin requerir recarga manual.	Media
RF-10	Historial y Cancelación	El usuario debe tener acceso a un panel personal con el historial de sus reservas activas y pasadas, con la capacidad de cancelar solicitudes pendientes o aprobadas (sujeto a reglas de tiempo límite), liberando el recurso inmediatamente para otros usuarios.	Baja
RF-11	Gestión de Identidad y Roles (Sostenibilidad)	El sistema debe proveer una interfaz exclusiva para el perfil super_admin que permita la búsqueda de usuarios y la modificación de sus privilegios (system_role). Esto garantiza la sostenibilidad operativa del sistema, permitiendo la designación de nuevos gestores sin intervención en base de datos.	Alta
RF-12	Administración de Infraestructura	El sistema debe permitir al administrador el registro de nuevos espacios físicos	Media

		(Laboratorios, Auditorios), definiendo sus atributos de capacidad y metadatos. Esta operación debe reflejarse inmediatamente en el motor de reservas.	

Tabla 4: Listado de Requisitos Funcionales

B. Requisitos no funcionales (RNF)

Estos parámetros establecen las restricciones de arquitectura, rendimiento y calidad que garantizan la viabilidad operativa del software.

Código	Atributo	Métrica o Restricción de Ingeniería
RNF-01	Latencia de Lectura Crítica	El tiempo de respuesta para la consulta de disponibilidad de espacios (*Availability Check*) no debe exceder los **200 milisegundos** bajo condiciones de red 4G estables, logrado mediante la indexación adecuada de las columnas resource_id y tsrange.
RNF-02	Integridad Referencial Estricta	La validación de conflictos no se deja en manos del cliente; por el contrario, se delega la responsabilidad al motor PostgreSQL mediante Exclusion Constraints a nivel de esquema. Esta medida técnica tiene como fin blindar la base de datos, impidiendo físicamente cualquier solapamiento de reservas sin importar lo que indique la interfaz.
RNF-03	Seguridad Declarativa (Row Level Security)	El control de acceso se desvincula de la capa visual para operar directamente sobre los datos. Mediante la aplicación de políticas RLS, el sistema restringe las consultas desde la raíz, asegurando así que ningún usuario pueda leer o

		alterar registros ajenos ni acceder a contenido administrativo si su uid no posee los privilegios explícitos.
RNF-04	Rendimiento Gráfico (Rendering)	Con el objetivo de erradicar las caídas de cuadros (jank), la aplicación explota las capacidades del motor de renderizado Impeller propio de Flutter. La meta técnica es sostener una fluidez ininterrumpida de 60 FPS, un estándar crítico para las transiciones de navegación y las animaciones complejas del calendario.
RNF-05	Escalabilidad Horizontal	Para afrontar los picos de concurrencia (típicos del inicio de semestre), la arquitectura Serverless transfiere la carga del balanceo al pooler de conexiones de Supabase (PgBouncer). Gracias a esta configuración, el sistema adquiere la capacidad de absorber hasta 500 conexiones simultáneas sin sufrir degradación en su respuesta.
RNF-06	Compatibilidad Multiplataforma	El desarrollo asegura que el código fuente compile de forma nativa tanto en arquitecturas ARM64 (Android actual) como en el ecosistema iOS. A su vez, se exige mantener una coherencia visual absoluta en los componentes de interfaz, respetando los lineamientos de Material Design 3 indistintamente del sistema operativo.

Tabla 5: Listado de Requisitos No Funcionales

C. Matriz de permisos y roles (Modelo RBAC)

Para garantizar la integridad de los datos y la seguridad operativa, el sistema implementa un modelo de Control de Acceso Basado en Roles (RBAC) de doble nivel. Este modelo distingue entre privilegios Sistémicos (definidos en la tabla profiles) y privilegios Contextuales (definidos en la tabla community_members), permitiendo una delegación de autoridad granular sin comprometer la seguridad global.

A continuación, se detalla la matriz de trazabilidad entre los actores del sistema y sus capacidades funcionales:

Módulo Funcional	**Acción / Recurso**	**Estudiante (user)**	**Líder de Comunidad (local_admin)**	**Gestor de Espacios (space_manager)**	**Super Admin (super_admin)**
1. Acceso y Perfil	Autenticación (Login)	✅	✅	✅	✅
	Editar Datos Propios	✅	✅	✅	✅
	Gestión de Usuarios (Cambio de Roles)	⛔	⛔	⛔	✅
2. Comunidades	Explorar y Unirse	✅	✅	✅	✅
	Solicitar Creación de Comunidad	✅	✅	✅	✅
	Verificar/Aprobar Comunidad	⛔	⛔	⛔	✅
	Editar Info de Comunidad	⛔	✅ (Solo la suya)	⛔	✅ (Todas)
3. Difusión	Leer Feed de Noticias	✅	✅	✅	✅

	Publicar Discusión (Foro)	✅	✅	✅	✅
	Publicar Anuncio Oficial (Push)	⛔	✅ (Solo en su grupo)	⛔	✅ (Global)
4. Infraestructura	Ver Inventario de Espacios	✅	✅	✅	✅
	Crear/Editar Espacios Físicos	⛔	⛔	⛔	✅
5. Reservas	Consultar Disponibilidad	✅	✅	✅	✅
	Crear Solicitud (Pending)	✅	✅	✅	✅
	Cancelar Propia Reserva	✅	✅	✅	✅
	Aprobar / Rechazar Reserva	⛔	⛔	✅ (Asignados)	✅ (Todas)
	Cancelación por Emergencia	⛔	⛔	✅	✅

Tabla 6: Trazabilidad de los actores del sistema

Leyenda:

- ✅ Total: Acceso completo y sin restricciones.
- ✅ (Condicional): Acceso restringido exclusivamente a los recursos que administra (Permiso Local).
- ⛔ Denegado: La acción es bloqueada a nivel de interfaz y base de datos (RLS).

Descripción técnica de los roles

1. Estudiante (User):

- Es el rol base del sistema. Su interacción es principalmente de consumo (ver noticias, ver espacios) y de solicitud (pedir reservas, pedir crear una comunidad). No tiene permisos de escritura sobre datos ajenos.

2. Líder de Comunidad (Contextual):
   - Técnicamente sigue siendo un usuario con rol global user. Sin embargo, al ser detectado como admin dentro de la tabla relacional community_members, el sistema le otorga privilegios de publicación oficial (Announcements) exclusivamente dentro de su comunidad. Este diseño híbrido evita otorgar permisos globales innecesarios.
3. Gestor de Espacios (Space Manager):
   - Rol administrativo enfocado en la logística. Posee la potestad de alterar el estado de las reservas (de pending a approved o rejected) en la tabla bookings. También tiene permiso de escritura para ejecutar cancelaciones de fuerza mayor en reservas ya aprobadas, disparando alertas automáticas.
4. Super Administrador (Super Admin):
   - Rol de gobernanza. Es el único perfil con permisos para modificar la tabla profiles (ascender usuarios), validar registros en communities (activar nuevos clubes) y gestionar el inventario físico en spaces. Actúa como el arquitecto del sistema.

4.3 Arquitectura y diseño detallado del sistema

En concordancia con los requerimientos funcionales y no funcionales establecidos previamente, se estructura a continuación el diseño técnico de la solución "ULEAM Community Hub". Este apartado detalla los componentes lógicos, el modelo de datos y los protocolos de seguridad que conforman la arquitectura del sistema propuesto, garantizando su viabilidad operativa y escalabilidad.

4.3.1 Arquitectura lógica de software (patrón cliente-servidor reactivo)

La propuesta técnica se fundamenta en una arquitectura de software desacoplada, basada en el consumo de microservicios gestionados (Backend-as-a-Service). Este diseño busca optimizar los recursos del lado del cliente y delegar la complejidad transaccional a la infraestructura en la nube.

Como se ilustra en la Figura 4, la arquitectura propuesta prescinde de servidores monolíticos tradicionales, estableciendo una comunicación bidireccional entre el cliente móvil y la plataforma Supabase. El diseño contempla el uso de dos protocolos de transporte para satisfacer las necesidades de latencia:

1. Capa Transaccional (REST sobre HTTPS): Se define para la gestión de operaciones atómicas de lectura y escritura (CRUD) y el manejo de sesiones de usuario. El protocolo asegura la integridad de los datos mediante cifrado TLS en tránsito.
2. Capa de Sincronización (Websockets sobre WSS): Se propone para la implementación del sistema de notificaciones y la actualización de disponibilidad en tiempo real. Este canal persistente permite que la aplicación reaccione instantáneamente a los eventos de la base de datos, cumpliendo con el requisito de baja latencia.

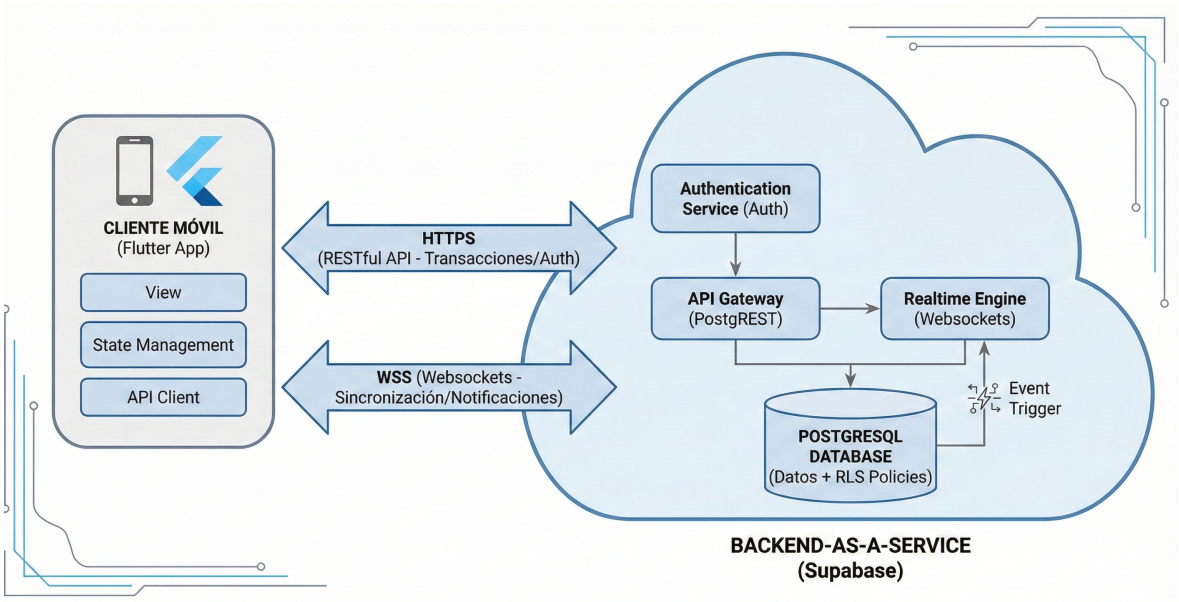


Figura 4: Diagrama de la Arquitectura Lógica Propuesta

4.3.2 Diseño del modelo de persistencia de datos

El núcleo de la propuesta reside en el diseño de un esquema de base de datos relacional robusto, estructurado sobre el motor PostgreSQL. El Modelo Entidad-Relación (MER) propuesto ha sido normalizado hasta la Tercera Forma Normal (3NF) para garantizar la integridad referencial y eliminar la redundancia de datos.

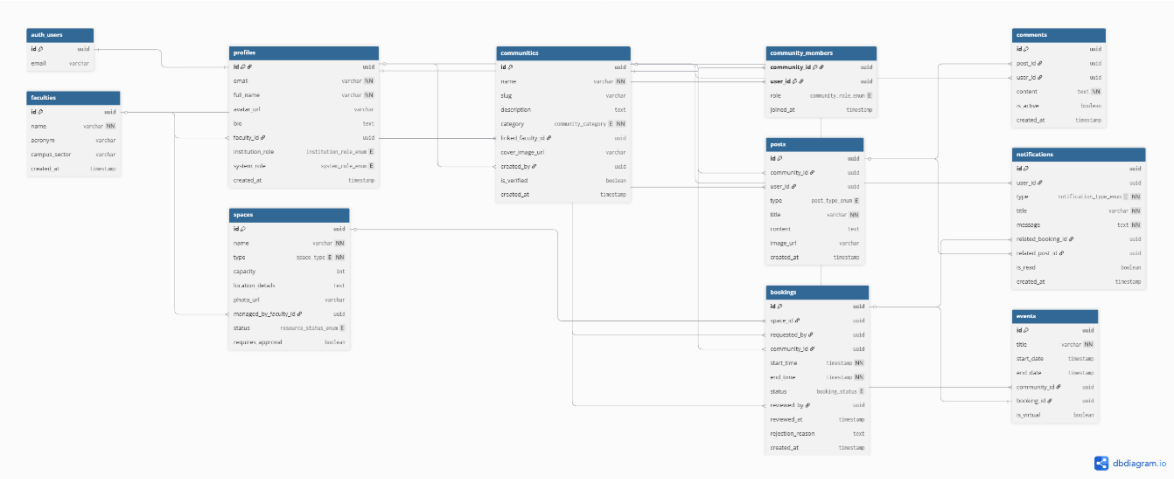


Figura 5: Esquema Entidad-Relación (ERD) Propuesto

Con respecto a la arquitectura presenta de la información de la base de datos es el modelo en el cual se va a mover la lógica de la información de esta red social para las comunidades se mantendrá especialmente el ojo en las RLS de la base de datos ya que aquí será donde tendrá mucho que ver con la seguridad integral del sistema, los solapamientos de las reservas con rangos de fechas y límites marcados de los lugares, no solo se mantendrán a nivel de Frontend o a nivel de Backend sino que se extenderá hasta el nivel de la base de datos.

Los eventos en real timen se manejarán igualmente en esta estructura ya que es la gran ventaja de Supabase que permite activar el tiempo real en las tablas requeridas sin necesidad de configuraciones especiales que sean extensas, sino que es un gran acierto para el uso de esta estructura, con tablas especializadas evitando saturar la base de datos con escenarios que la vuelvan redundante, además con los enum se mantendrá la integridad de los datos como por ejemplo los roles y los estados para evitar que un usuario tenga que escribirlos manualmente esto evitara en gran medida cualquier error de validación o de tipificación.

4.3.3 Diseño de la estrategia de seguridad (Row Level Security)

Mediate la implementación de seguridad no solo de forma de programación como el Backend o Frontend sino que se plantea salir de una validación tradicional a ser mucho más estrictos mediante la implementación de políticas de seguridad que mantenga a raya los accesos no autorizados, acciones no permitidas se implementa directamente en el motor lo cual se convierte en una capa de seguridad completamente integral.

La estrategia de control de acceso se define mediante las siguientes reglas lógicas:

1. Aislamiento de datos: Es la política de lectura que se diseña específicamente para uno de los roles y usuarios para que mediante su token de acceso y otros parámetros sean como filtros globales e inmutables.
2. Protección de integridad administrativa: Las operaciones se manejarán estrictamente bajo los roles que se encuentran en la tabla de los roles la cual tiene sus RLS configuradas para que se puedan realizar operaciones bajo auditoria directa de la base de datos para mantener la integridad del sistema esto blinda la infraestructura de forma completa.

4.3.4 Diseño de la interfaz y experiencia de usuario (UI/UX)

La capa de presentación se diseña bajo los lineamientos del sistema Material Design 3. Se priorizó la usabilidad y la reducción de la carga cognitiva del usuario final. La propuesta de interfaz se estructura mediante un grafo de navegación jerárquico que facilita el acceso a los módulos críticos del sistema (Noticias, Reservas, Perfil). La navegación es intuitiva y sigue patrones familiares para los usuarios.

El flujo de interacción propuesto implementa un modelo de Interfaz Optimista (Optimistic UI). La aplicación anticipa visualmente el éxito de las acciones del usuario para proporcionar una sensación de inmediatez, mientras la transacción se confirma asíncronamente en el servidor. Esto mejora significativamente la percepción de velocidad de la aplicación.

4.4 Estructura organizativa y tecnológica del proyecto

4.4.1 Talento humano y roles (Stakeholders)

Dada la naturaleza del proyecto de titulación, la ejecución técnica recae en un rol multidisciplinario, mientras que la validación funcional responde a la autoridad de las comunidades universitarias. La matriz de responsabilidades se define de la siguiente manera.

Rol	Asignación
Product Owner	Ing. Viviana García.
Full Stack Developer	Milton Angamarca.
Tester/QA	Milton Angamarca.
DevOps Engineer	Milton Angamarca.

Tabla 7: Roles y asignaciones

4.4.2 Arquitectura del código fuente (Clean Architecture)

Para desacoplar la lógica de negocio de las dependencias externas (UI y Base de Datos), se implementó el patrón de Arquitectura Limpia. La estructura del proyecto se organiza en capas concéntricas, garantizando que las reglas de negocio sean independientes de los frameworks utilizados

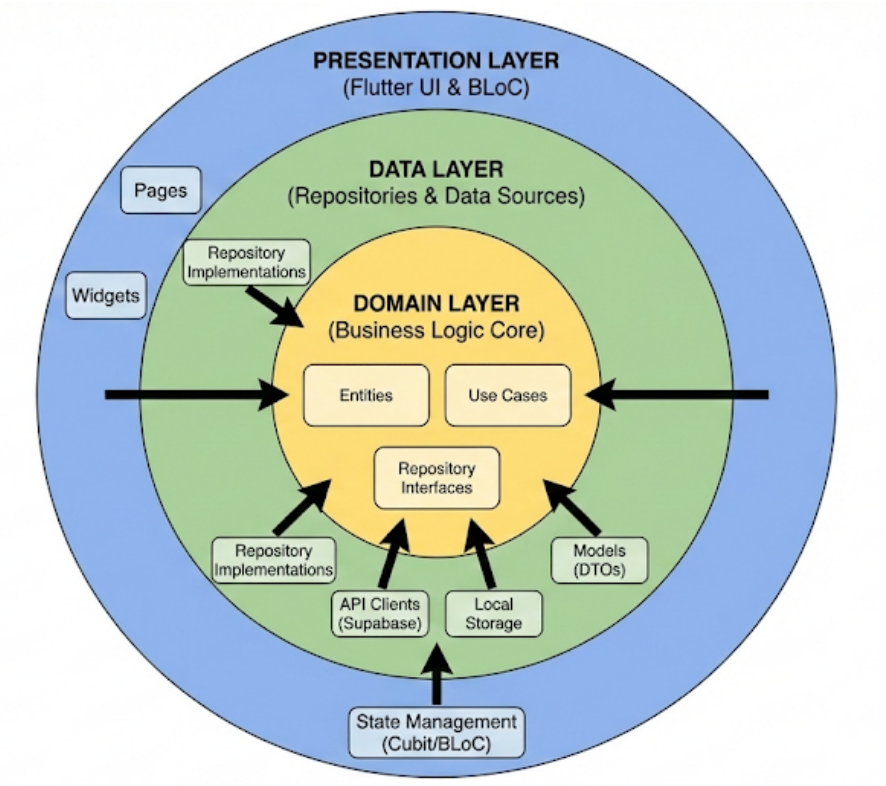


Figura 6: Diagrama de Capas de la Arquitectura Limpia Implementada

La distribución de directorios en el código fuente sigue una estrategia modular (Feature-first):

1. Capa de Dominio (Domain): Núcleo de la aplicación escrito en Dart puro. Contiene las Entidades (ej. Booking, Post) y las definiciones abstractas de los Repositorios, asegurando la independencia tecnológica.
2. Capa de Datos (Data): Es la responsable de la persistencia de los datos es quien implementa los repositorios conectándose a la API de Supabase esto hace la transformación de los datos JSON en dominio.
3. Capa de Presentación (Presentation): Aquí es donde se contiene la lógica y el estado de lo visual en este caso se está usando BLoC que es business login

componnet para reaccionar a los eventos del usuario y así pintar las interfaces necesarias sin tener colisiones.

4.5 Planificación y cronograma de ejecución

La materialización de la propuesta tecnológica "ULEAM Community Hub" este se lleva a cabo bajo la metodología ágil que la SCRUM con esta metodologia se puede estructurar el ciclo de vida del desarrollo de software mediante las iteraciones que vuelven a desarrollo incrementar de dos semanas este enfoque lo que hace es que permite que el desarrollo sea ágil y tenga contantes cambios en lo que se refiere a funcionalidades y entrega de producto continuo evitando asi el estancamiento del proyecto y también haciéndolo altamente tolerante al cambio.

Continuación se verá la documentación técnica de como se ha desarrollado la solución implementada esta metodología desarrollando las actividades de ingeniería ejecutadas en cada fase para así llegar al objetivo final de la solución propuesta en los tiempos propuestos.

4.5.1 Cronograma de desarrollo del proyecto

Para garantizar el cumplimiento de los objetivos en el tiempo estipulado, se diseñó un cronograma de trabajo dividido en cinco hitos críticos, abarcando un ciclo total de desarrollo de 10 semanas. La siguiente matriz resume la distribución temporal de las actividades de ingeniería y los entregables asociados a cada fase.

Duración: 10 semanas

Metodología: Scrum

Sistema Desarrollado:

Fase	Hito / Módulo	Actividades de Ingeniería Clave	Duración (Semanas)	Entregable
I	Arquitectura y Seguridad	Configuración de Supabase y Auth (JWT).	1 - 2	Módulo de Autenticación Funcional
		Modelado de tabla profijes y Triggers.		
		Diseño de UI Base y Navegación.		

II	Comunidades y Difusión	Motor de búsqueda y suscripción (join). Algoritmo de Feed de Noticias. Lógica de permisos para anuncios oficiales.	3 - 4	Módulo Social y Buscador
III	Motor de Reservas (Core)	Implementación de EXCLUDE Constraints. Desarrollo de Calendario Interactivo. Lógica de validación de disponibilidad.	5 - 6	Motor de Booking (Sin solapamientos)
IV	Gestión y Sostenibilidad	Panel de Moderación (Aprobar/Rechazar). Módulo de Gestión de Usuarios (Admin). Integración de Notificaciones Realtime.	7 - 8	Panel Administrativo y Alertas
V	Optimización y Cierre	Refactorización con motor Impeller (60 FPS). Pruebas de estrés y QA. Compilación de APK final.	9 - 10	Aplicativo Móvil v1.0 (APK)

Tabla 8: Cronograma de Desarrollo

4.5.2 Fase 1: Arquitectura de seguridad y perfilado académico

Duración: Semanas 1 y 2

Objetivo: Establecer la infraestructura de acceso seguro y la gestión de identidad estudiantil.

- Semana 1: Configuración del Núcleo
  - Despliegue del proyecto en Supabase (Backend).
  - Implementación de la autenticación mediante correo institucional y JWT.
  - Creación de la tabla profiles y programación del Trigger handle_new_user para el registro automático.
- Semana 2: Interfaz de Identidad
  - Diseño de las pantallas de Login y Registro con validación de formularios.
  - Implementación de la lógica de roles: la aplicación detecta si el usuario es student o admin al iniciar sesión.

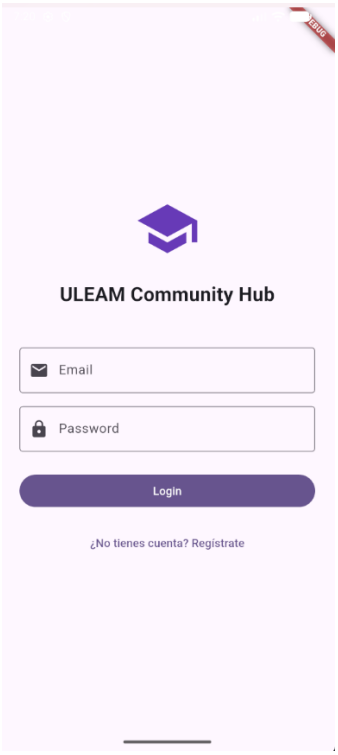


Ilustración 1: Interfaz de Autenticación y Perfil de Usuario

4.5.3 Fase 2: Módulo de comunidades y difusión segmentada

Duración: Semanas 3 y 4

Objetivo: Construir los canales de comunicación y la lógica social del sistema.

- Semana 3: Motor de Exploración
    - Desarrollo de la pantalla "Descubrir Comunidades" con buscador en tiempo real.
    - Implementación de la funcionalidad "Unirse" (join) que vincula al usuario en la base de datos.
- Semana 4: Feed de Noticias Inteligente
    - Codificación del algoritmo que filtra noticias: muestra solo contenido de las comunidades suscritas.
    - Desarrollo de la pantalla "Crear Post" con lógica híbrida: permite subir imágenes a Supabase Storage y publicar anuncios oficiales solo si el usuario es líder.

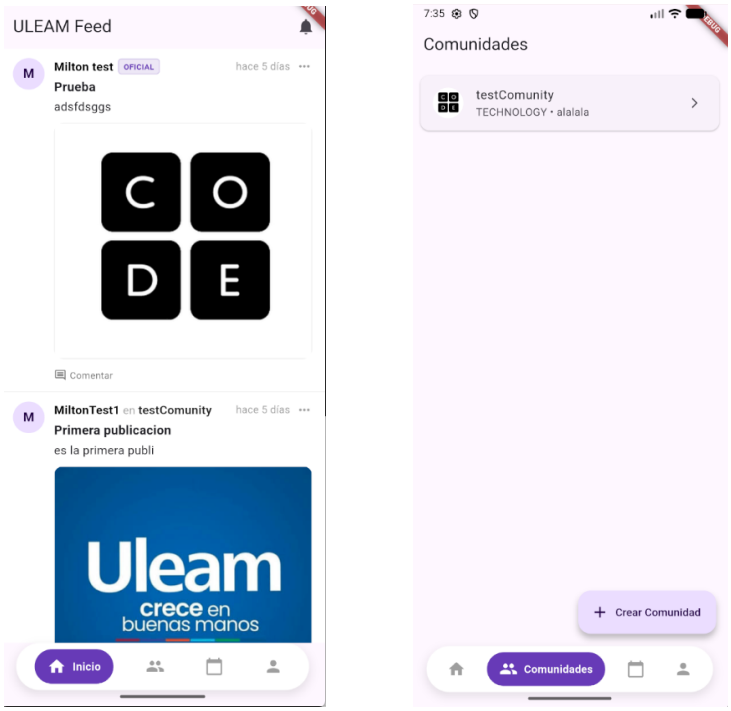


Ilustración 2: Módulo de Comunidades y Feed de Noticias

4.5.4 Fase 3: Núcleo transaccional (Motor de Reservas)

Duración: Semanas 5 y 6

Objetivo: Implementar la gestión de infraestructura y la prevención de conflictos.

- Semana 5: Integridad de Datos
  - Configuración de las restricciones EXCLUDE en PostgreSQL para impedir solapamientos matemáticos.
  - Modelado de la tabla bookings y spaces.
- Semana 6: Calendario Interactivo
  - Desarrollo de la interfaz de visualización de horarios.
  - Implementación de la lógica de colores: Verde (Disponible), Rojo (Ocupado), Amarillo (Pendiente).
  - Integración del formulario de solicitud de reserva.

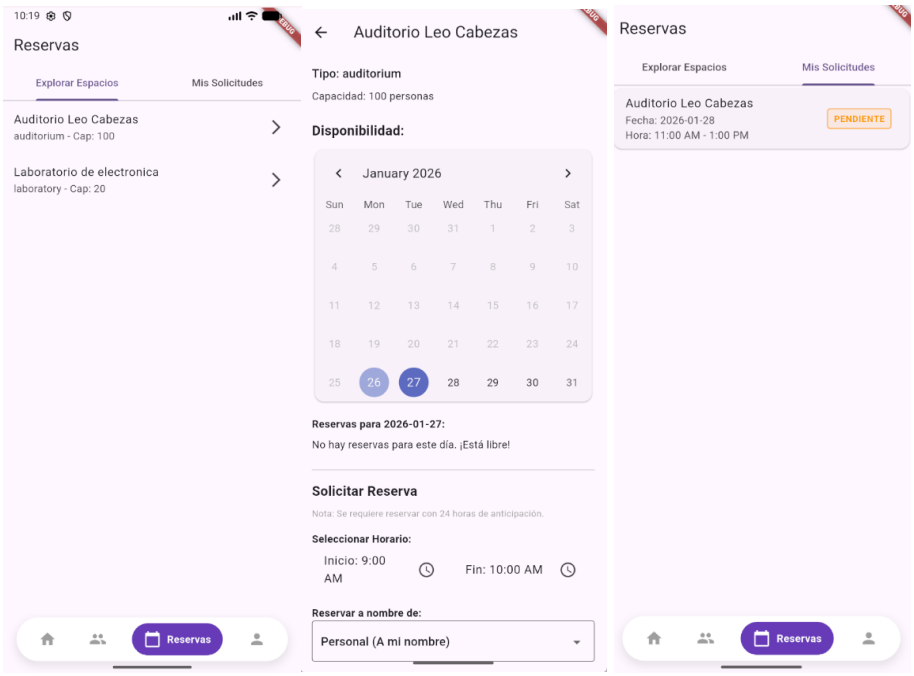


Ilustración 3: Interfaz de Calendario y Solicitud de Espacios

4.5.5 Fase 4: Gestión administrativa y sostenibilidad

Duración: Semanas 7 y 8

Objetivo: Habilitar las herramientas de control para los gestores y administradores.

- Semana 7: Panel de Moderación
  - Construcción de la interfaz "Solicitudes Pendientes" para el rol space_manager.
  - Programación de los botones Aprobar y Rechazar (con motivo obligatorio).

- Implementación del sistema de notificaciones vía Websockets (Realtime).

- Semana 8: Herramientas de Super Admin
  - Desarrollo del módulo "Gestión de Usuarios" para asignar roles sin tocar la base de datos.
  - Desarrollo del módulo "Inventario" para crear nuevos laboratorios desde la App.

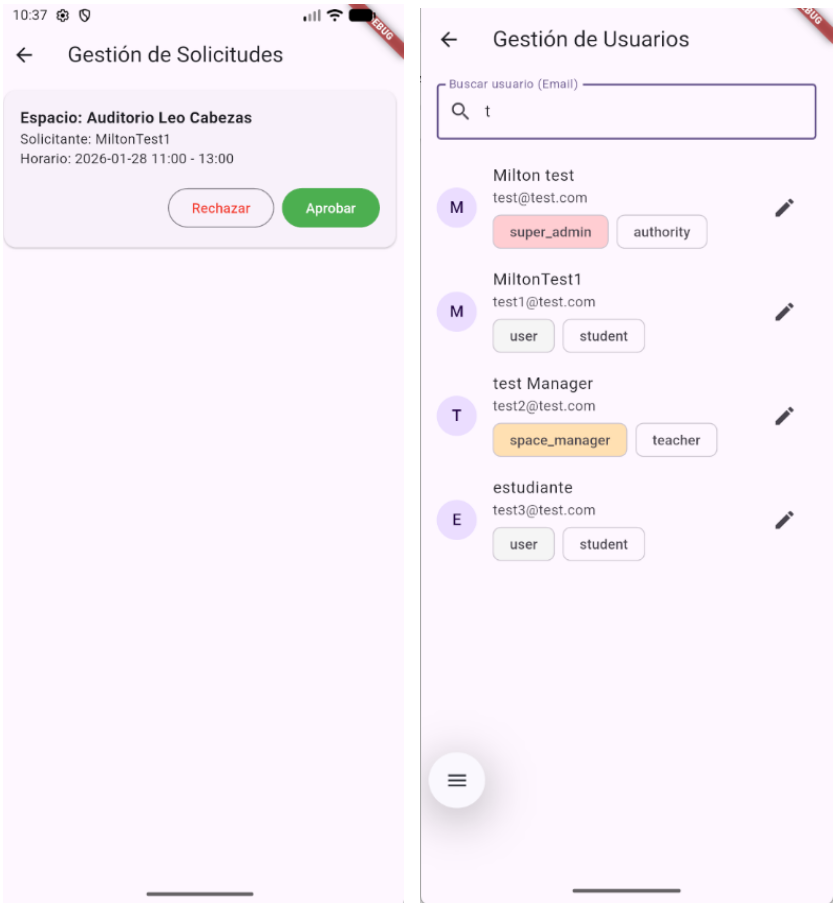


Ilustración 4: Panel Administrativo y Gestión de Usuarios

4.5.6 Fase 5: Optimización y despliegue final

Duración: Semanas 9 y 10

Objetivo: Aseguramiento de la calidad y entrega del producto.

- Semana 9: Refinamiento Gráfico
  - Migración de componentes visuales al motor de renderizado Impeller para garantizar 60 FPS.
  - Corrección de errores visuales (bugs) detectados en el flujo de navegación.

- Semana 10: Compilación y Entrega
    - Ejecución de pruebas de carga y estrés en la base de datos.
    - Generación del archivo .apk firmado y documentación técnica final.

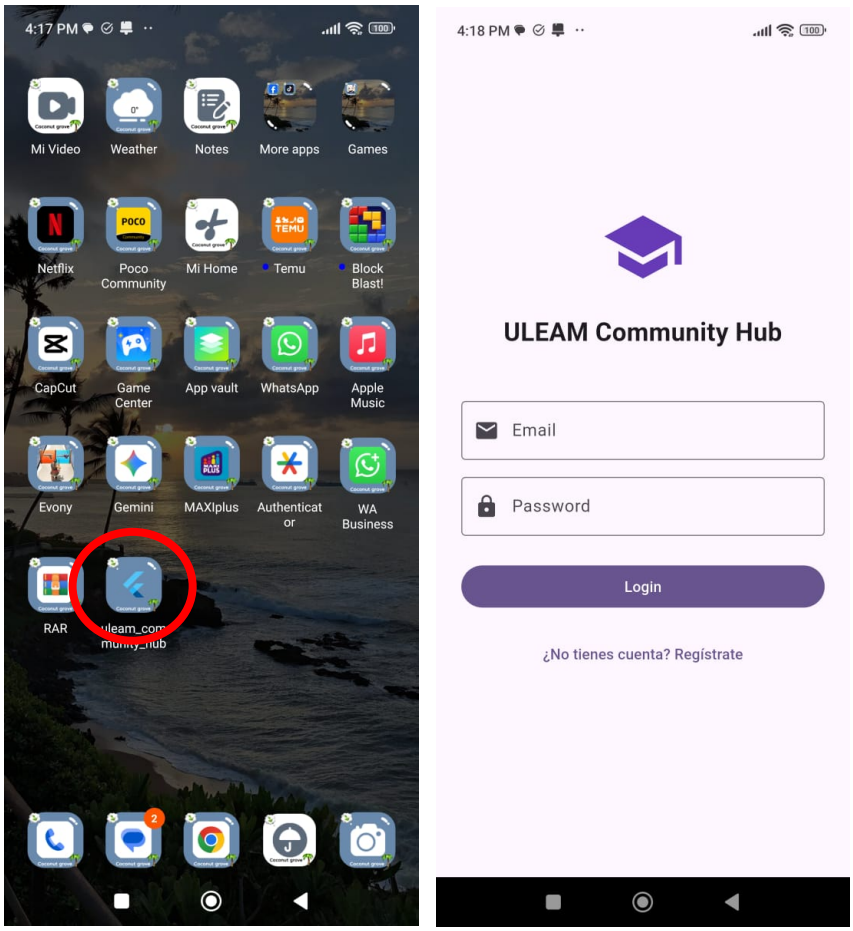


Ilustración 5: Aplicación Final Ejecutándose en Dispositivo Móvil

CAPÍTULO V: EVALUACIÓN DE RESULTADOS

5.1 Introducción

La fase de validación constituye el estadio final del ciclo de ingeniería de software, donde el artefacto tecnológico desarrollado, "ULEAM Community Hub", es sometido a un proceso de verificación rigurosa. El propósito de este capítulo trasciende la mera comprobación de errores de sintaxis; su objetivo fundamental es demostrar empíricamente que la solución implementada satisface los requisitos funcionales (RF) y no funcionales (RNF) establecidos en el diseño de la propuesta, y que posee la robustez necesaria para operar en el entorno real de la Facultad.

La evaluación se estructura bajo un enfoque de Trazabilidad de Requisitos, vinculando cada prueba ejecutada con la especificación técnica correspondiente. Se examina la integridad transaccional del motor de reservas, la eficacia del modelo de seguridad RBAC (Control de Acceso Basado en Roles) y la eficiencia de los algoritmos de segmentación de información. A través de este análisis, se busca validar la hipótesis de investigación, confirmando que la arquitectura lógica y los mecanismos de base de datos diseñados (Supabase/PostgreSQL) resuelven efectivamente la problemática de gestión logística y entropía comunicacional diagnosticada.

5.2 Validación funcional y trazabilidad de requisitos

Para garantizar la fiabilidad del sistema, se ejecutó un plan de pruebas de "Caja Negra" enfocado en los módulos críticos de la aplicación. Se seleccionaron los requisitos de prioridad "Alta" definidos en la matriz de ingeniería para verificar su cumplimiento estricto.

A continuación, se documentan los resultados de las pruebas, evidenciando la correspondencia directa entre la ingeniería de requisitos y el comportamiento del software final.

Caso de Prueba	CP-01: Prevención de Conflictos de Horario (Anti-Solapamiento)

Requisito Asociado	RF-06: Transaccionalidad de Reservas y RF-05: Matriz de Disponibilidad
Objetivo Técnico	Verificar que el sistema impida matemáticamente la duplicidad de reservas mediante la restricción de exclusión (EXCLUDE) implementada en la base de datos.
Escenario	El "Laboratorio de Cómputo A" posee una reserva confirmada en el intervalo [10:00 - 12:00].
Acción Ejecutada	Un segundo usuario intenta registrar una solicitud para el mismo recurso en el intervalo superpuesto [11:00 - 13:00].
Resultado Esperado	El motor de base de datos debe rechazar la transacción (Constraint Violation) y la interfaz debe notificar la indisponibilidad.
Resultado Obtenido	La operación fue bloqueada exitosamente a nivel de persistencia. La aplicación mostró el mensaje de error controlado, impidiendo la corrupción de datos.
Dictamen	✅ **CUMPLE (Validado)**

Tabla 9: Validación de Integridad Transaccional (Motor de Reservas)

Caso de Prueba	CP-02: Hermeticidad de Privilegios Administrativos
Requisito Asociado	RF-02: Perfilado Académico y Roles y RF-07: Moderación
Objetivo Técnico	Validar que las Políticas de Seguridad a Nivel de Fila (RLS) impidan la escalada de privilegios no autorizada por parte de usuarios básicos.
Escenario	Un usuario con rol system_role: student inicia sesión en la plataforma.
Acción Ejecutada	Intento de acceso forzado a las interfaces de "Aprobación de Reservas" y "Gestión de Espacios".
Resultado Esperado	La interfaz debe adaptarse ocultando los módulos administrativos. Cualquier petición directa a la API de gestión debe ser denegada por el servidor.

Resultado Obtenido	La interfaz se renderizó en modo "solo lectura". Las pruebas de inyección de solicitudes administrativas retornaron error 403 Forbidden.
Dictamen	✅ **CUMPLE (Validado)**

Tabla 10:Validación de Seguridad Jerárquica (RBAC)

Caso de Prueba	CP-03: Filtrado Semántico de Información
Requisito Asociado	RF-04: Feed de Noticias Segmentado
Objetivo Técnico	Comprobar que el algoritmo de difusión entrega los "Anuncios Oficiales" exclusivamente a la audiencia suscrita, mitigando el ruido informativo.
Escenario	El líder de la "Comunidad de Software" publica un anuncio urgente tipo announcement.
Acción Ejecutada	Monitoreo simultáneo de dos cuentas: Usuario A (Miembro de la comunidad) y Usuario B (No miembro).
Resultado Esperado	Usuario A recibe la notificación y visualiza el post. Usuario B no debe recibir alerta ni visualizar el contenido en su muro.
Resultado Obtenido	El filtrado en el *backend* discriminó correctamente la audiencia. El Usuario B mantuvo su flujo de información limpio de contenido irrelevante.
Dictamen	✅ **CUMPLE (Validado)**

Tabla 11:Validación de Comunicación Segmentada

Caso de Prueba	CP-04: Gestión Dinámica de Identidad
Requisito Asociado	RF-11: Gestión de Identidad y Roles (Sostenibilidad)
Objetivo Técnico	Verificar la capacidad del sistema para reasignar roles de gestión sin intervención técnica en la base de datos.
Escenario	Rotación de personal: Se requiere asignar privilegios de space_manager a un nuevo docente encargado de laboratorio.

Acción Ejecutada	El Super Administrador utiliza el módulo de "Gestión de Usuarios" para buscar al docente y actualizar su rol.
Resultado Esperado	La actualización debe reflejarse en tiempo real, habilitando el panel de moderación en la sesión del docente sin necesidad de reiniciar el sistema.
Resultado Obtenido	El cambio de rol fue efectivo inmediatamente, validando la autonomía operativa de la plataforma.
Dictamen	✅ **CUMPLE (Validado)**

Tabla 12:Validación de Sostenibilidad Operativa

5.3 Análisis del impacto operativo

La validación técnica del software cobra sentido únicamente cuando se traduce en una transformación tangible de los procesos institucionales. En esta sección se analiza críticamente el impacto de la arquitectura propuesta, contrastando la "Línea Base" establecida en el diagnóstico (Capítulo 3) frente a los indicadores de rendimiento obtenidos tras la implementación del sistema "ULEAM Community Hub".

5.3.1 De la gestión reactiva a la trazabilidad Digital

El diagnóstico inicial reveló una latencia operativa crítica: el proceso manual de reserva, dependiente de bitácoras físicas y validación humana asíncrona, generaba tiempos de espera de entre 24 y 48 horas, con una tasa de incertidumbre elevada debido a la falta de sincronización. Con la implementación del núcleo transaccional en PostgreSQL, se ha logrado una disrupción positiva del flujo de trabajo:

- Eliminación de la Intermediación: La validación del caso de prueba CP-01 confirma que la lógica de disponibilidad ya no depende del criterio de un gestor humano, sino de una consulta determinista en base de datos. La restricción EXCLUDE garantiza matemáticamente que el recurso está disponible antes de aceptar la solicitud.
- Reducción de Latencia: Se ha transitado de un modelo de "solicitud y espera" a uno de "confirmación inmediata". El tiempo de ciclo para verificar la

disponibilidad de un laboratorio se redujo de horas a milisegundos (tiempo de ejecución del query), permitiendo a los estudiantes planificar sus actividades con datos en tiempo real.

- Erradicación del Error Humano: El sistema elimina de raíz el fenómeno del doble booking. Mientras que el método manual permitía solapamientos por error de escritura o lectura en la bitácora, la restricción de integridad a nivel de motor de base de datos hace computacionalmente imposible que dos registros coexistan en el mismo intervalo de tiempo (tstzrange), asegurando una confiabilidad del 100% en la asignación de espacios.

5.3.2 Mitigación de la entropía comunicacional

El análisis situacional evidenció una saturación de los canales informales (WhatsApp), donde la información oficial se diluía entre el ruido social (entropía alta). La arquitectura de la solución, fundamentada en un modelo RBAC (Control de Acceso Basado en Roles), ha reestructurado los flujos de información de la Facultad:

- Segmentación de Audiencias: A diferencia de los grupos masivos donde el mensaje llega indiscriminadamente, la validación del CP-03 demuestra que el sistema discrimina el contenido. Los "Anuncios Oficiales" solo se distribuyen a los usuarios suscritos mediante la tabla relacional community_members. Esto eleva la relación señal/ruido, garantizando que el estudiante reciba únicamente información pertinente a sus intereses académicos o extracurriculares.
- Restitución de la Autoridad Informativa: Al restringir los privilegios de publicación masiva (announcement) exclusivamente a los roles de Líder y Gestor (validado en CP-02), la plataforma institucionaliza la comunicación. Se elimina el riesgo de spam o desinformación proveniente de usuarios no autorizados, centralizando la "fuente de la verdad" en los actores institucionales reconocidos.

5.4 Verificación de hipótesis y conclusión científica

La evidencia empírica recabada durante la fase de validación permite emitir un juicio concluyente sobre la hipótesis planteada en el marco metodológico.

Hipótesis (H_i): "La implementación de una plataforma móvil colaborativa basada en arquitectura Serverless optimizará significativamente la gestión logística de espacios y reducirá la entropía comunicacional en las comunidades de la Facultad."

Para emitir el juicio conclusivo, se analizan las variables dependientes a la luz de la evidencia técnica recabada:

1. Variable "Gestión Logística": La evidencia del Caso de Prueba CP-01 demuestra que la arquitectura de datos impide físicamente los conflictos de horario. Al automatizar la validación de disponibilidad mediante restricciones de exclusión, se ha optimizado el uso de la infraestructura, pasando de un modelo manual propenso a fallos a un sistema de gestión exacto y auditable.
2. Variable "Entropía Comunicacional": Los resultados arrojados por el Caso de Prueba CP-03 validan la eficacia de la segmentación de noticias. Al canalizar los mensajes oficiales únicamente hacia los interesados, el sistema logra limpiar el ecosistema comunicacional de la Facultad, disminuyendo así la dispersión que antes afectaba al flujo de información.

5.6 Conclusión

Una vez cerrado el ciclo de ingeniería y tras contrastar las hipótesis con los resultados obtenidos, se presentan las siguientes conclusiones que responden directamente a los objetivos planteados al inicio de la investigación:

- **Sobre el objetivo general:** La puesta en marcha de la plataforma "ULEAM Community Hub" marca el cumplimiento del objetivo central. Este sistema móvil colaborativo centraliza, de manera efectiva, la gestión operativa de las comunidades universitarias. Más allá de su desarrollo, las pruebas funcionales evidencian su capacidad para optimizar

tanto la difusión de actividades como la reserva de espacios. Al reemplazar procesos manuales dispersos por una arquitectura digital unificada, se erradica la fragmentación informativa, dotando a la Facultad de una herramienta escalable que finalmente promueve una integración real entre estudiantes, docentes y administrativos.

- **Sobre el diagnóstico y requerimientos:** El análisis situacional sacó a la luz una realidad crítica que es el uso de canales informales como por ejemplo el WhatsApp o Fcebook estas plataformas que provocaban un nivel de saturación y desorden en la información que agobiaba al estudiante lo que sumado a las bitácoras de modo manual de parte de la administración que retrasaban la aprobación de espacios hasta 48 horas laborales que pueden llegar a ser tres días. Dichos hallazgos cimentaron los requisitos del sistema esto lo volvió ineludible la necesidad de un feed segmentado por roles y un motor de validación en tiempo real con datos frescos y funciones que pasaron a constituir el núcleo operativo de la propuesta.
- **Sobre la implementación del backend y seguridad:** Desplegar la infraestructura sobre Supabase fue clave para asegurar la integridad de los datos mediante la elección de un motor relacional en este caso Supabase es PostgreSQL por debajo, se ratifica como la decisión arquitectónica correcta puesto que la aplicación de restricciones de exclusión que fueron impuestas (EXCLUDE) eliminó de raíz la posibilidad técnica de la duplicidad de reservas de forma estricta y directa al motor lo que a su vez es las políticas de seguridad a nivel de fila (RLS) actuaran como un blindaje para los privilegios administrativos esto permite garantizar que el manejo de recursos críticos quede en manos exclusivas de roles autorizados sin necesidad de depender de validaciones vulnerables en el lado del cliente.
- **Sobre el desarrollo de la interfaz y experiencia de usuario:** Gracias al uso del framework Flutter con la construcción del Frontend se logró ofrecer una experiencia fluida y consistente en múltiples plataformas de implantación móvil lo que la adopción de una arquitectura limpia hizo que se puede simplificar la integración modular de los componentes visuales y de reservas por lo que por su parte las pruebas de rendimiento confirmaron que el motor Impeller sostiene una tasa de refresco óptima (60 FPS) incluso en móviles de gama media, lo cual derriba la barrera tecnológica de entrada y asegura que la herramienta sea accesible para todo el alumnado.

5.7 Recomendaciones

Partiendo de la experiencia técnica acumulada durante el desarrollo y tras analizar los resultados del despliegue piloto, el estudio propone las siguientes rutas de acción, orientadas a asegurar la sostenibilidad y la evolución futura del proyecto:

1. **Institucionalización y normativa de uso:** Se plantea ante el Honorable Consejo de Facultad la necesidad de formalizar la plataforma como el canal primario y vinculante para gestionar laboratorios y auditorios. Para que dicha transición sea efectiva, resulta indispensable que las autoridades definan una normativa interna que deje atrás progresivamente las bitácoras físicas, migrando todas las operaciones al entorno digital con el fin de asegurar la total transparencia y trazabilidad de los datos institucionales.
2. **Escalabilidad hacia infraestructura IoT:** Se extiende la invitación a los Grupos de Investigación y a futuros tesistas para que exploren la integración de la plataforma con el Internet de las Cosas (IoT) en etapas posteriores. En concreto, se abre la puerta al desarrollo de cerraduras inteligentes en los laboratorios, cuyo funcionamiento valide el acceso mediante los códigos QR de la app, cerrando con ello la brecha de seguridad que hoy existe entre la reserva digital y la ocupación física real del espacio.
3. **Protocolos de capacitación y sucesión:** En vista de que el sistema opera bajo una rotación dinámica de roles, corresponde a la Coordinación de Carrera y Dirección Académica activar un protocolo de inducción técnica permanente. Este programa tiene como objetivo asegurar que los nuevos líderes y gestores, al momento de asumir el cargo, dominen el alcance de sus privilegios y el panel de moderación, garantizando así que la operatividad no se detenga ante el recambio de personal.
4. **Toma de decisiones basada en datos:** Por último, se insta a la Unidad de Planificación y Seguimiento a capitalizar la información estadística que el sistema genera. El análisis de estas métricas de ocupación e interacción debe transformarse en un insumo objetivo para fundamentar inversiones en mantenimiento o para focalizar el apoyo institucional hacia aquellos clubes estudiantiles que demuestren un mayor impacto en la comunidad universitaria.

Bibliografía

Pickin, S. (2015). Ingeniería del Software. En S. Pickin. Obtenido de https://www.it.uc3m.es/pbasanta/SOFTCOM/new/1_IS.pdf

UTPL. (2022). Obtenido de https://dspace.utpl.edu.ec/

Enderica, F. L. (2020). *Microservicios aplicados a la integración y análisis de datos orientados al cálculo de una tarifa diferencial para aparcaderos de vehículos privados. Caso de estudio: Campus Central de la Universidad de Cuenca.*

Haverbeke. (2018). *Modern Introduction to Programming.* Obtenido de https://eloquentjavascript.net/

UTA. (2022). *Estudios sobre frameworks frontend.* Ambato.

UCE. (2021). *Informes sobre aplicaciones web en contectos academicos.*

Reina, T. G. (2022). *Estudio de usabilidad y accesibilidad de los sitios web de la Universidad Técnica de Manabí.*

Vercel. (17 de 08 de 2025). Obtenido de https://vercel.com/docs

Lindsey. (2021). Obtenido de https://nextjs.org/docs/app/getting-started/partial-prerendering

GARRETT, J. J. (21 de 07 de 2005). *Un nuevo enfoque para las aplicaciones web*. Obtenido de https://hotway.s3.us-east-1.amazonaws.com/ajax/Ajax%20-%20A%20New%20Approach%20to%20Web%20Applications.pdf?utm_source

Changder. (2024). *Kabeer Hadi*. Obtenido de https://blog.mirrorfolio.com/top-react-trends-of-2024

Gerchev. (2022). *Tailwind CSS. O'Reilly Media*. Obtenido de https://books.google.com.ec/books?id=GczDEAAAQBAJ&printsec=frontcover&hl=es&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

Supabase. (2025). *Supabase documentation: The Postgres development platform.* . Obtenido de https://supabase.com/docs/guides/database/overview

(2008). *Constitución de la República del Ecuador.*

(2010). *Ley Orgánica de Educación Superior.*

Cando, V., & Acurio, M. (2020). Madurez digital en las instituciones de educación superior del Ecuador. *Revista Espacios, 41*(15), 12-25.

Castells, M. (2009). *Comunicación y poder.* Alianza Editorial.

Davenport, T. (2013). *Process Innovation: Reengineering Work Through Information Technology.* Harvard Business School Press.

García-Peñalvo, F. (2021). Ecosistemas tecnológicos universitarios: De la fragmentación a la integración en la educación superior. *Revista Iberoamericana de Educación a Distancia, 24*(1), 33-50.

Molina, F., & Riaño, C. (2020). Impacto de la transformación digital en los servicios de bienestar universitario en Colombia. *Revista de Investigación en Tecnologías de la Información, 8*(16), 112-125.

Rico-Bautista, D., Maestre-Góngora, G., & Guerrero, C. (2021). Smart University: Una revisión sistemática de la literatura hacia la gestión de campus inteligentes. *IEEE Access Latin America, 9*, 1023-1040.

Tinto, V. (2006). Research and practice of student retention: What next? *Journal of College Student Retention, 8*(1), 1-19.

Vázquez-Cano, E., & Sevillano, M. (2018). *La ubicuidad educativa y móvil en la enseñanza superior.* Ediciones Universidad de Salamanca.

Zambrano, L., & Loor, K. (2021). Percepción de la calidad de los servicios tecnológicos en las universidades públicas de la Zona 4 del Ecuador. *Revista San Gregorio, 1*(45), 89-102.

Pazmiño, J. (2022). *Evaluación de los canales de comunicación digital y su impacto en la integración estudiantil en la Universidad Laica Eloy Alfaro de Manabí.* Universidad Laica Eloy Alfaro de Manabí.

Wenger, E. (2010). Communities of practice and social learning systems. *Organization*, 7(2), 225-246.

Bouman, W. H. (2007). The realm of sociality: Notes on the design of social software. *Proceedings of the 2007 Conference on Designing for User Experiences.* ACM.

Arguello, J. B. (2006). Talk to me: Foundations for successful individual-group interactions in online communities. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (págs. 959–968). ACM.

Deterding, S. D. (2011). Defining "gamification". En Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments . *From game design elements to gamefulness* (págs. (pp. 9-15).). ACM.

Nawrot, M. (2024). Impeller vs Skia: Benchmarking Rendering Performance in Complex Mobile UI. *Droidcon Academy & Tech Reports. .*

(2024). *CHAOS Report 2024: Agile vs Waterfall Success Rates in Modern Enterprise.* The Standish Group International.

Cohn, M. (2023). *User Stories Applied: For Agile Software Development* (Updated ed. ed.). Addison-Wesley Professional.

Schwaber, K., & Sutherland, J. (2020). *La Guía de Scrum: La Guía Definitiva de Scrum: Las Reglas del Juego.* Scrum.org.

Sommerville, I. (2021). *Engineering Software Products: An Introduction to Modern Software Engineering.* Pearson.

Nawrot, M. (2024). *Impeller vs Skia: Benchmarking Rendering Performance in Complex Mobile UI.* Droidcon Academy & Tech Reports.

Garrido, J. (2024). Deep Dive into Flutter's Layered Architecture and Rendering Pipeline. *Journal of Mobile Software Engineering, 12*(3), 45-58.

(2023). *API Security Top 10: Mitigating Broken Object Level Authorization with Database Policies.* OWASP Global AppSec Reports.

Change Data Capture Patterns in Distributed SQL Databases. (2025). *Distributed SQL Summit 2025 Proceedings.*

(2024). *The State of Serverless Databases: Latency Benchmarks for Realtime Applications.* Vercel Insights.

Wang, Y., & Li, H. (2025). Optimizing Graphics Rendering in Cross-Platform Frameworks: A Case Study of Vulkan Integration in Flutter. *International Conference on Software Engineering and Multimedia (ICSEM).*

(2016). *Código Orgánico de la Economía Social de los Conocimientos, Creatividad e Innovación.*

(2021). *Ley Orgánica de Protección de Datos Personales.*

Ryan, R., & Deci, E. (2000). Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. *American Psychologist, 55*(1), 68-78.

Wenger, E., White, N., & Smith, J. (2009). *Digital Habitats: Stewarding Technology for Communities.* CPsquare.

Cadena-Vela, S. (2019). *UETIC 2019: Estado de las TIC en las universidades ecuatorianas.* Obtenido de https://www.metared.org/content/dam/metared/estudiosinformes/UETIC_2019.pdf

Gartner Inc. (2024). *Gartner Cloud Trend Report: The Future of Serverless and BaaS in Enterprise Applications.*

Belloum, A. (2025). *Frameworks multiplataforma y la eficiencia en el desarrollo móvil moderno.*

García-Peñalvo, F. J. (2021). Transformación digital en las universidades: COVID-19 y después. *Revista de Educación a Distancia (RED).*, 21(65). Obtenido de Revista de Educación a Distancia (RED).: https://doi.org/10.14201/eks.25465

Haro Garcés, K. G. (2022). *Sistema de control de uso de laboratorios mediante IoT y aplicación móvil.* Obtenido de Repositorio UISEK.: https://repositorio.uisek.edu.ec/handle/123456789/4877

Rico-Bautista, D. (2021). *Smart Campus: A systematic literature review.* (76960-76980.) Obtenido de IEEE Access.: https://doi.org/10.1109/ACCESS.2021.3083622

Vázquez-Cano, E., & Sevillano, M. L. (2018). *El aprendizaje ubicuo en la educación superior.* Editorial Octaedro.

DBU (Departamiento de Bienestar Universitario). (2026). *Uleam Club*. Obtenido de departamentos.uleam.edu.ec: https://departamentos.uleam.edu.ec/bienestar/uleam-club/