# UNIVERSIDAD TÉCNICA DEL NORTE


FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE SOFTWARE

TEMA:

**IMPLEMENTACIÓN DEL SERVICIO "RED SOCIAL DE MENSAJERÍA" EN LA PLATAFORMA MÓVIL DE LA UNIVERSIDAD TÉCNICA DEL NORTE UTILIZANDO EL MARCO DE TRABAJO SCRUM Y LA NORMA ISO/IEC 25022**

Trabajo de grado previo a la obtención del título de Ingeniero de Software presentado ante la ilustre Universidad Técnica del Norte

**AUTOR**:

Jorge Rafael Rosero Cuaspud

**DIRECTOR**:

Ing. José Antonio Quiña Mera, Ph.D.

**Ibarra - Ecuador**

2025

# UNIVERSIDAD TÉCNICA DEL NORTE

## BIBLIOTECA UNIVERSITARIA

### AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

## 1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

| DATOS DE CONTACTO | | | |
|---|---|---|---|
| **CÉDULA DE IDENTIDAD:** | 1003962824 | | |
| **APELLIDOS Y NOMBRES:** | Rosero Cuaspud Jorge Rafael | | |
| **DIRECCIÓN:** | Mira, Carchi. Panamericana Norte y Av. García Moreno | | |
| **EMAIL:** | jrroseroc@utn.edu.ec / jorgerafaelrosero@gmail.com | | |
| **TELÉFONO FIJO:** | | **TELÉFONO MÓVIL:** | 0961971500 |

| DATOS DE LA OBRA | |
|---|---|
| **TÍTULO:** | Implementación del servicio “Red Social de Mensajería” en la plataforma móvil de la Universidad Técnica del Norte utilizando el marco de trabajo SCRUM y la norma ISO/IEC 25022 |
| **AUTOR:** | Jorge Rafael Rosero Cuaspud |
| **FECHA DE APROBACIÓN: DD/MM/AAAA** | 24 de octubre de 2025 |
| **PROGRAMA:** | ■ **PREGRADO** ☐ **POSGRADO** |
| **TÍTULO POR EL QUE OPTA:** | Ingeniero en Software |
| **ASESOR /DIRECTOR:** | Ing. José Anotnio Quiña Mera, Ph.D. |

## 2. CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrolló sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 24 días del mes de octubre de 2025

**EL AUTOR:**

Firmado electrónicamente por:
JORGE RAFAEL ROSERO CUASPUD
Validar únicamente con FirmaEC

.....

Jorge Rafael Rosero Cuaspud

CI: 1003962824

# CERTIFICACIÓN DEL DIRECTOR

Certifico que el trabajo de grado “Implementación del servicio ‘Red Social de Mensajería’ en la plataforma móvil de la Universidad Técnica del Norte utilizando el marco de trabajo SCRUM y la norma ISO/IEC 25022”, ha desarrollado en su totalidad por el señor: Jorge Rafael Rosero Cuaspud, portador de la cédula de identidad número 1003962824.

JOSE ANTONIO QUIÑA MERA

Firmado digitalmente por JOSE ANTONIO QUIÑA MERA
Nombre de reconocimiento (DN): c=EC, sn=QUIÑA MERA, givenName=JOSE ANTONIO, serialNumber=IDCEC-1002322384, cn=JOSE ANTONIO QUIÑA MERA, 2.5.4.97=TINEC-1002322384001

……

Ing. José Antonio Quiña Mera, Ph.D.

Director de Trabajo de Grado

# DEDICATORIA

A mi madre, quien con su apoyo incondicional me supo guiar a lo largo de mi vida, enseñándome que el amor y la perseverancia son las herramientas más poderosas para alcanzar las metas. Gracias por cada sacrificio, por cada palabra en los momentos de duda, y por creer mí.

A mi padre, por ser ejemplo de trabajo. Por demostrarme que los obstáculos no son límites, sino oportunidades para crecer.

A mis hermanos, quienes con su compañía y apoyo incondicional hicieron más ligero cada paso de este recorrido. Gracias por celebrar cada pequeño logro como si fuera suyo.

A mis amigos, esos compañeros de batalla que caminaron junto a mí en este proceso, quienes me tendieron la mano, celebraron mis triunfos y no me hicieron sentir de menos. Que hicieron este camino más ameno con cada risa, chiste, sin importar en la situación donde nos encontrábamos.

A todos ustedes, que son parte fundamental de este logro, este trabajo lleva un pedacito de cada uno de ustedes.

# AGRADECIMIENTO

A mis padres, por su apoyo incondicional a lo largo de toda mi vida y por estar presentes en cada uno de mis objetivos. Su sacrificio constante y amor me han brindado la fortaleza necesaria para perseguir y alcanzar cada meta propuesta. Les agradezco infinitamente por enseñarme que cumplir un sueño implicar aprender a caer y, sobre todo, a levantarse con determinación, manteniéndome siempre fiel a mis principios, valores y con humildad ante los logros conseguidos.

A mis hermanos, por acompañarme y sostenerme en los momentos difíciles, por enseñarme el significado de la familia y por estar siempre presentes cuando más los necesitaba. Gracias por ser pilares fundamentales en este nuevo capítulo de mi vida.

A mis tutores, Ing. Antonio Quiña y Ing. Mauricio Rea, por confiar en mí y brindarme la oportunidad de demostrar mis capacidades en este proyecto tan importante, gracias por su guía, dedicación, conocimientos y compromisos compartidos.

A mis amigos y compañeros, por hacer de este camino una experiencia memorable, por cada momento compartido, por su apoyo mutuo y por recordarme que los verdaderos logros se construyen en conjunto.

A la Universidad Técnica del Norte, por bridarme una formación integral que va más allá de los conocimientos técnicos, sino también los valores y principios fundamentales para convertirme en un profesional competente, ético y comprometido con la sociedad.

# Tabla de Contenidos


INTRODUCCIÓN ..... 3
Planteamiento del Problema ..... 3
Metodología ..... 5
Justificación ..... 6
Justificación tecnológica ..... 6
Justificación social ..... 6
Antecedentes ..... 6
CAPÍTULO II ..... 9
MARCO TEÓRICO ..... 9
2.1 Las aplicaciones móviles en el contexto educativo ..... 9
2.1.1 Concepto de aplicaciones móviles ..... 9
2.1.2 Transformación digital en la educación superior ..... 9
2.1.3 Aplicaciones móviles académicas en Ecuador ..... 10
2.1.4 Beneficios de las aplicaciones móviles en universidades ..... 10
2.1.5 Limitaciones y desafíos en el uso de aplicaciones móviles ..... 10
2.1.6 Impacto de las aplicaciones móviles en el aprendizaje ..... 10
2.1.7 Ejemplos de aplicaciones móviles usadas en instituciones educativas ..... 10
2.2 Mensajería en tiempo real ..... 11
2.2.1 Concepto y evolución de los sistemas de mensajería ..... 11
2.2.2 Componentes principales de los sistemas de mensajería ..... 11
2.2.3 Tecnologías para la mensajería en tiempo real ..... 12
2.2.3.1 Concepto de WebSocket ..... 12
2.2.3.2 Handshake en WebSockets ..... 12
2.2.4 Aplicaciones de mensajería en el contexto académico ..... 13
2.2.4.1 Ventajas y desventajas para estudiantes y docentes ..... 13
2.2.4.2 Casos de éxito en universidades ..... 14
2.2.5 Seguridad en sistemas de mensajería ..... 14
2.3 Metodología Scrum y Norma ISO/IEC 25022 ..... 15
2.3.1 Metodología Scrum en el desarrollo de software ..... 15
2.3.1.1 Concepto de Scrum ..... 15
2.3.1.2 Roles y estructura de Scrum ..... 15
2.3.1.3 Integración de Scrum con la norma ISO/IEC 25022 ..... 15
2.3.2 Comparativa con otras metodologías ágiles ..... 16

2.3.3 Introducción a las normas ISO en el desarrollo de software ..... 17
2.3.3.1 Importancia de los estándares internacionales ..... 17
2.3.3.2 Diferencias entre ISO/IEC 25010, 25022 y otras normas de calidad . 17
2.3.4 Propósito y alcance de la ISO/IEC 25022 ..... 18
2.3.4.1 Definición de calidad en uso ..... 18
2.3.5 Aplicaciones en proyectos tecnológicos ..... 18
2.3.6 Métricas de calidad según ISO/IEC 25022 ..... 18
2.3.6.1 Eficacia: definición e indicadores ..... 18
2.3.6.2 Eficiencia: definición e indicadores ..... 18
2.3.6.3 Satisfacción: definición e indicadores ..... 18
2.3.7 Casos de estudio y aplicaciones prácticas ..... 19
2.3.7.1 Ejemplo en plataforma educativa ..... 19
2.4 Tecnologías utilizadas en el proyecto ..... 19
2.4.1 Flutter ..... 19
2.4.1.1 Concepto de Flutter ..... 19
2.4.1.2 Comparativa entre Flutter y otras tecnologías multiplataforma ..... 20
2.4.2 Node.js y arquitectura de microservicios ..... 21
2.4.3 Bases de datos y tecnologías para datos en tiempo real ..... 21
2.4.3.1 Concepto de base de datos ..... 21
2.4.3.2 Bases de datos relacionales vs. no relacionales ..... 22
2.4.4 Herramientas como Firebase y Oracle ..... 22
2.4.4.1 Concepto de Firebase ..... 22
2.4.4.2 Concepto de Oracle ..... 22
2.4.4.3 Concepto de Oracle APEX ..... 22
2.4.5 Node.js ..... 22
2.4.5.1 Concepto de Node.js ..... 22
2.4.5.2 Concepto de NestJs ..... 23
2.5 Trabajos relacionados ..... 23
CAPÍTULO III ..... 26
DESARROLLO DEL PROYECTO ..... 26
3.1 Análisis ..... 26
3.1.1 Equipo Scrum ..... 26
3.1.2 Definición de requisitos ..... 27
3.1.3 Product Backlog ..... 33
3.2 Diseño – Sprint 0 ..... 34

3.2.1 Resumen del Sprint 0 ..... 34
3.2.2 Arquitectura Tecnológica ..... 34
3.2.3 Esquema Base de Datos Inicial ..... 39
3.2.4 Diagrama de Proceso ..... 40
3.3 Desarrollo del Servicio “Red Social de Mensajería” ..... 40
3.3.1 Sprint 1 ..... 40
3.3.2 Sprint 2 ..... 57
3.3.3 Sprint 3 ..... 68
3.3.4 Sprint 4 ..... 79
3.4 Validación y cierre del desarrollo ..... 96
3.4.1 Pruebas de aceptación de historias de usuario ..... 96
3.4.2 Reunión de revisión y ajustes adicionales ..... 97
3.4.3 Incorporación de nuevas historias de usuario ..... 97
CAPÍTULO IV ..... 106
VALIDACIÓN DE RESULTADOS ..... 106
4.1 Modelo de calidad en uso ..... 106
4.1.1 Definición del modelo ISO/IEC 25022 ..... 106
4.1.2 Métricas evaluadas ..... 107
4.1.3 Relación con los instrumentos utilizados ..... 108
4.2 Metodología de evaluación ..... 109
4.2.1 Diseño del taller práctico ..... 109
4.2.2 Diseño de la encuesta SUS ..... 111
4.3 Fiabilidad de los instrumentos ..... 112
4.3.1 Fiabilidad del taller práctico ..... 112
4.3.2 Fiabilidad de la encuesta SUS ..... 113
4.4 Resultados ..... 114
4.4.1 Resultados del taller práctico ..... 114
4.4.2 Resultados de la encuesta SUS ..... 115
4.5 Análisis e interpretación ..... 116
4.5.1 Evaluación de la eficacia ..... 116
4.5.2 Evaluación de la eficiencia ..... 117
4.5.3 Evaluación de la satisfacción ..... 119
4.5.4 Cambios posteriores a la evaluación ..... 121
4.5.5 Evaluación general de la calidad en uso ..... 122
4.6 Conclusiones de la validación ..... 125

CONCLUSIONES ..... 127
RECOMENDACIONES ..... 128
BIBLIOGRAFÍA ..... 129
ANEXOS ..... 135
Anexo A. Fotografías de reuniones en el DDTI ..... 135
Anexo B. Fotografías de aplicación del taller práctico y encuesta SUS ..... 136
Anexo C. Taller práctico para evaluar el módulo de la aplicación móvil ..... 137
Anexo D. Tabulación del taller práctico ..... 138
Anexo E. Tabulación de la encuesta SUS ..... 144
Anexo F. Acta de Entrega de Recepción ..... 147

## Índice de Figuras

Figura 1. Árbol de problemas ..... 3
Figura 2. Arquitectura por desarrollar ..... 4
Figura 3. Componentes principales de los sistemas de mensajería ..... 12
Figura 4. Arquitectura de WebSocket ..... 13
Figura 5. Metodología Scrum ..... 15
Figura 6. Arquitectura de Flutter ..... 20
Figura 7. Arquitectura de capas de NestJs ..... 23
Figura 8. Ciclo de Sprints ..... 26
Figura 9. Arquitectura compartida de microservicios ..... 35
Figura 10. Estructura de carpetas y archivos de la plantilla ..... 36
Figura 11. Estructura de carpetas del proyecto UTN Móvil ..... 37
Figura 12. Esquema base de datos entidad-relación inicial ..... 39
Figura 13. Diagrama de proceso ..... 40
Figura 14. Modelo entidad-relación ajustado ..... 42
Figura 15. Estructura de carpetas actual ..... 43
Figura 16. Repository de mensajes individuales ..... 45
Figura 17. Estructura de archivos en la carpeta socket actual ..... 47
Figura 18. Configuración WebSocket Gateway ..... 48
Figura 19. Configuración de manejo de conexiones Socket ..... 50
Figura 20. Configuración de eventos ..... 51
Figura 21. Estructura de carpetas en Flutter ..... 52
Figura 22. Registro de dependencias App Red Social ..... 52
Figura 23. Vista de Carrera – versión 1 ..... 54
Figura 24. Vista de buscador de personas (delegate) ..... 55
Figura 25. Vista de conversaciones individuales – versión 1 ..... 56
Figura 26. Modelo entidad-relación de base de datos local ..... 58
Figura 27. Collection que representa la tabla de mensaje individual ..... 61
Figura 28. Vista de conversación individual con mensajes de respuesta - versión 1 ..... 62
Figura 29. Vista de conversación individual con mensajes de respuesta – versión 2 ..... 63
Figura 30. Arquitectura de eventos para mensajes entre emisor y receptor ..... 64
Figura 31. Configuración para unir a rooms ..... 65
Figura 32. Mockup de vista de conversacion grupal ..... 66

Figura 33. Implementación de mockup de conversación grupal ..... 67
Figura 34. Bug de mensajes de respuesta seguidos en modo offline ..... 69
Figura 35. Arquitectura híbrida para enviar, subir y descargar archivos ..... 70
Figura 36. Vista de selección de fotos desde la galería ..... 71
Figura 37. Vista de selección de fotos desde un álbum ..... 71
Figura 38. Vista de tomar una fotografía desde la cámara ..... 72
Figura 39. Vista de selección de un archivo ..... 72
Figura 40. Vista antes de enviar un pdf ..... 73
Figura 41. Vista antes de enviar diferentes tipos de archivos ..... 73
Figura 42. Vista antes de enviar una imagen ..... 74
Figura 43. Vista de archivo como mensaje para abrir – versión 1 ..... 74
Figura 44. Vista de archivo como mensaje ajeno – versión 2 ..... 75
Figura 45. Vista de Imagen como mensaje – versión 1 ..... 75
Figura 46. Vista de archivo como mensaje propio ..... 76
Figura 47. Temas con diferentes paletas de colores – versión 1 ..... 76
Figura 48. Temas con diferentes paletas de colores – versión 2 ..... 77
Figura 49. Mockup con la paleta de colores seleccionado ..... 78
Figura 50. Vista de miembros de conversación grupal de asignatura ..... 81
Figura 51. Vista de miembros de conversación grupal de docentes ..... 81
Figura 52. Vista de chats individuales con foto de perfil ..... 82
Figura 53. Vista de conversación individual con foto de perfil ..... 83
Figura 54. Vista de información del contacto con foto de perfil ..... 83
Figura 55. Vista de conversaciones individuales con badges ..... 84
Figura 56. Vista de Institución - Todas ..... 85
Figura 57. Vista de Institución - Generales ..... 85
Figura 58. Vista de institución – Docentes - versión 1 ..... 86
Figura 59. Vista de institución - Docentes - versión 2 ..... 87
Figura 60. Implementación de notificaciones ..... 88
Figura 61. Implementación de funcionalidades de mensajes ..... 89
Figura 62. Tabla Red Social Ciclos Académicos ..... 90
Figura 63. Vista de aplicación en mantenimiento ..... 91
Figura 64. Vista de aplicación en mantenimiento con mensaje personalizado ..... 91
Figura 65. Editor de páginas del módulo Red Social en APEX ..... 92
Figura 66. Vista de ciclos académicos para la aplicación ..... 92

Figura 67. Formulario para seleccionar un ciclo académico ..... 93
Figura 68. Formulario para agregar un nuevo ciclo académico a la aplicación ..... 94
Figura 69. Lista de ciclos académicos para editar ..... 94
Figura 70. Vista del formulario de edición de un ciclo académico ..... 95
Figura 71. Arquitectura de eventos entre emisor y receptor - versión 2 ..... 102
Figura 72. Vista de un mensaje con la opción eliminar..... 102
Figura 73. Vista de conversación grupal con mensajes eliminados ..... 103
Figura 74. Lógica para eliminar la asignatura local ..... 104
Figura 75. Vista del badge con mensajes pendientes - versión 1 ..... 105
Figura 76. Vista del badge con mensajes pendientes - versión 2 ..... 105
Figura 77. Pruebas de normalidad al taller práctico ..... 112
Figura 78. Matriz de correlaciones del taller práctico ..... 113
Figura 79. Alfa de Cronbach a la encuesta SUS..... 113
Figura 80. Resultado de SUS..... 120
Figura 81. Resultado de calidad en uso ..... 125

## Índice de Tablas

Tabla 1. Antecedentes ..... 6
Tabla 2. Ventajas y desventajas del uso de aplicaciones de mensajería en el contexto académico ..... 13
Tabla 3. Tabla comparativa de metodologías ágiles ..... 16
Tabla 4. Tabla comparativa: Flutter vs React Native ..... 20
Tabla 5. Equipo Scrum ..... 26
Tabla 6. Historia de Usuario 01 ..... 27
Tabla 7. Historia de Usuario 02 ..... 27
Tabla 8. Historia de Usuario 03 ..... 28
Tabla 9. Historia de Usuario 04 ..... 28
Tabla 10. Historia de Usuario 05 ..... 29
Tabla 11. Historia de Usuario 06 ..... 29
Tabla 12. Historia de Usuario 07 ..... 30
Tabla 13. Historia de Usuario 08 ..... 30
Tabla 14. Historia de Usuario 09 ..... 30
Tabla 15. Historia de Usuario 10 ..... 31
Tabla 16. Historia de Usuario 11 ..... 31
Tabla 17. Historia de Usuario 12 ..... 31
Tabla 18. Historia de Usuario 13 ..... 32
Tabla 19. Historia de Usuario 14 ..... 32
Tabla 20. Historia de Usuario 15 ..... 32
Tabla 21. Historia de Usuario 16 ..... 33
Tabla 22. Product Backlog ..... 33
Tabla 23. Sprint Backlog - Sprint 1 ..... 41
Tabla 24. Sprint Backlog - Sprint 2 ..... 57
Tabla 25. Sprint Backlog - Sprint 3 ..... 68
Tabla 26. Sprint Backlog - Sprint 4 ..... 79
Tabla 27. Resultados de pruebas de aceptación ..... 96
Tabla 28. Historia de Usuario 17 ..... 98
Tabla 29. Historia de Usuario 18 ..... 98
Tabla 30. Historia de Usuario 19 ..... 98

Tabla 31. Historia de Usuario 20 ..... 99
Tabla 32. Product Backlog Completo ..... 99
Tabla 33. Tipos de mensajes ..... 101
Tabla 34. Modelo de calidad en uso ..... 106
Tabla 35. Relación de métricas con los instrumentos de recolección ..... 108
Tabla 36. Taller práctico para evaluar el módulo de la aplicación móvil ..... 109
Tabla 37. Encuesta SUS ..... 111
Tabla 38. Resultados del taller práctico ..... 114
Tabla 39. Correspondencia entre valores numéricos y respuestas en la escala de Likert ..... 115
Tabla 40. Resultados de la encuesta SUS ..... 115
Tabla 41. Tiempo de un experto por tarea ..... 118
Tabla 42. Resultados de Calidad en Uso ..... 124

# RESUMEN

El presente trabajo describe el desarrollo e implementación del servicio de mensajería en tiempo real para la plataforma móvil de la Universidad Técnica del Norte. Ante la falta de un canal oficial de comunicación entre docentes y estudiantes, y la dependencia de aplicaciones externas como WhatsApp que generan dispersión de información, se diseñó una solución centralizada que mejora la comunicación académica institucional.

El sistema fue desarrollado bajo una arquitectura de microservicios en NestJS para el backend y Flutter con arquitectura limpia para la aplicación móvil, garantizando modularidad, escalabilidad y mantenibilidad. Se implementaron funcionalidades clave como mensajería individual y grupal en tiempo real mediante WebSocket, envío de archivos e imágenes, estados de entrega de mensajes (pendiente, enviado, recibido, leído), cifrado de contenido para proteger la privacidad, notificaciones push mediante Firebase Cloud Messaging, sincronización offline con base de datos local Isar, y un panel administrativo en Oracle APEX para gestionar configuraciones del sistema.

El desarrollo siguió la metodología ágil SCRUM organizada en cuatro sprints de cuatro semanas cada uno, permitiendo entregas incrementales y adaptación continua a los requisitos. La evaluación de calidad en uso se realizó aplicando la norma ISO/IEC 25022 con 46 estudiantes de Ingeniería en Software, midiendo tres características principales: eficacia, eficiencia y satisfacción. Los resultados demuestran un desempeño sobresaliente con 100% de tareas completadas, 69.67% de eficiencia temporal respecto a un usuario experto, y un puntaje SUS de 81.03 puntos que refleja alta satisfacción percibida. La calificación global de calidad en uso alcanzó 82.52%, validando que el sistema cumple con estándares técnicos y de experiencia de usuario, siendo apto para su uso institucional.

**Palabras clave:** aplicación móvil, mensajería en tiempo real, Flutter, NestJS, WebSocket, Scrum, ISO/IEC 25022, calidad en uso, educación superior, microservicios.

# ABSTRACT

This work describes the development and implementation of a real-time messaging service for the mobile platform of Universidad Técnica del Norte. Given the lack of an official communication channel between teachers and students, and the dependence on external applications such as WhatsApp that cause information dispersion, a centralized solution was designed to improve institutional academic communication.

The system was developed under a microservices architecture in NestJS for the backend and Flutter with clean architecture for the mobile application, ensuring modularity, scalability, and maintainability. Key functionalities were implemented such as individual and group real-time messaging via WebSocket, file and image sharing, message delivery statuses (pending, sent, received, read), content encryption to protect privacy, push notifications through Firebase Cloud Messaging, offline synchronization with Isar local database, and an administrative panel in Oracle APEX to manage system configurations.

The development followed the SCRUM agile methodology organized into four four-week sprints, allowing for incremental deliveries and continuous adaptation to requirements. Quality in use evaluation was conducted by applying the ISO/IEC 25022 standard with 46 Software Engineering students, measuring three main characteristics: effectiveness, efficiency, and satisfaction. The results demonstrate outstanding performance with 100% task completion, 69.67% time efficiency compared to an expert user, and a SUS score of 81.03 points reflecting high perceived satisfaction. The overall quality in use rating reached 82.52%, validating that the system meets technical and user experience standards, making it suitable for institutional use.



**Keywords:** mobile application, real-time messaging, Flutter, NestJS, WebSocket, Scrum, ISO/IEC 25022, quality in use, higher education, microservices.

# INTRODUCCIÓN

## Planteamiento del Problema

Actualmente, las aplicaciones móviles instaladas en los teléfonos inteligentes se han convertido en herramientas valiosas para facilitar el aprendizaje [1]. Este fenómeno ha sido posible debido al continuo avance de las tecnologías de la información y comunicación, que han permitido el desarrollo de plataformas que facilitan nuevas formas de interacción como los entornos virtuales de aprendizaje [2]. En estos entornos, los docentes pueden mejorar la comunicación con los estudiantes al facilitar el acceso a materiales didácticos y al emitir avisos importantes ya sea de profesores a alumnos o viceversa [3].

En este contexto, la escasez de una plataforma oficial de mensajería entre docentes y estudiantes ha llevado a la adopción de herramientas externas como WhatsApp para mantener la comunicación [3]. Si bien WhatsApp puede ser utilizado para construir espacios de comunicación y fomentar el compromiso académico, su uso en un contexto no formal puede llevar a que la comunicación se perciba como poco profesional. Además, la dependencia de esta y otras aplicaciones no centralizadas puede provocar una dispersión de información, dificultando la organización del conocimiento en el ámbito académico [4]. La Figura 1 presenta el árbol de problemas relacionado.

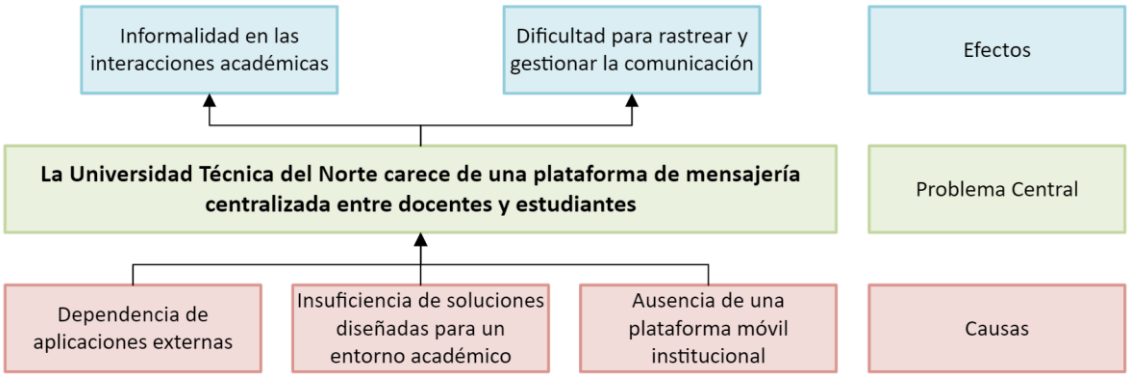



*Figura 1. Árbol de problemas*

Asimismo, la Figura 2 representa la arquitectura compartida que se desarrollará.

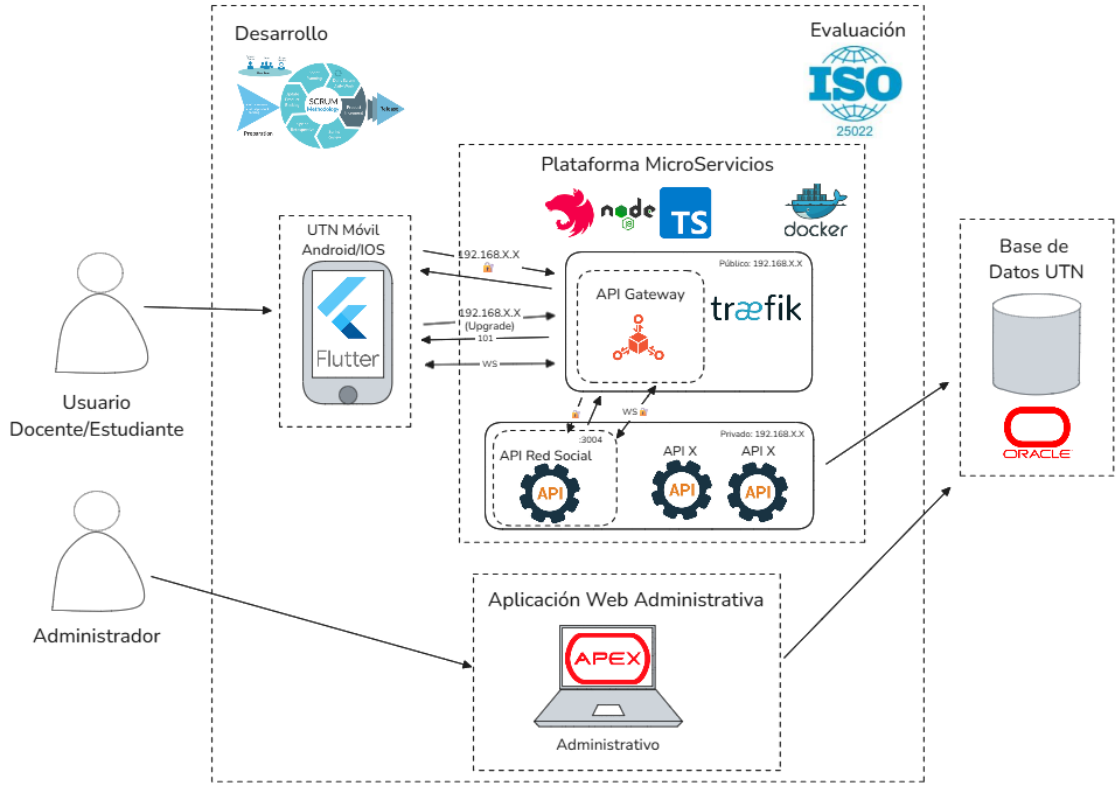



*Figura 2. Arquitectura por desarrollar*

El servicio “Red Social de Mensajería” implementará las siguientes funcionalidades clave, orientadas a mejorar la interacción académica entre docentes y estudiantes:

Mensajería individual y grupal en tiempo real: los usuarios podrán enviar y recibir mensajes de texto instantáneos, centralizando la comunicación académica dentro de un entorno formal y accesible desde cualquier dispositivo móvil.
Notificaciones de mensajes nuevos: Los estudiantes y docentes recibirán notificaciones en tiempo real sobre nuevos mensajes o interacciones importantes relacionados con actividades académicas.

Historial de conversaciones: Los usuarios podrán acceder al historial completo de sus conversaciones, organizadas por personas, asignaturas o conversaciones institucionales, facilitando el seguimiento de temas discutidos previamente.

Envío de archivos: se permitirá a los usuarios adjuntar y enviar archivos como documentos y recursos relevantes para las asignaturas, facilitando el intercambio de material académico.

Aplicación web en APEX: el personal administrativo tendrá acceso a una aplicación web para administrar configuraciones de la plataforma móvil.

El proyecto se llevará a cabo utilizando el marco de trabajo SCRUM, lo que facilitará iteraciones ágiles, la evaluación constante y la capacidad de ajustarse a los cambios [5]. Asimismo, se evaluará la calidad de uso de la aplicación móvil según la norma ISO/IEC 25022, utilizando métricas como eficacia, eficiencia y satisfacción, de acuerdo con los parámetros establecidos por la norma [6].

**Metodología**

El presente proyecto de investigación es de tipo aplicada, ya que busca implementar el servicio "Red Social de mensajería" en la plataforma móvil de la Universidad Técnica del Norte. Las actividades para cumplir con los objetivos establecidos son las siguientes:

1. En el primer objetivo se llevará a cabo una investigación documental con el propósito de recopilar información sobre el desarrollo de aplicaciones móviles y sistemas de mensajería en tiempo real utilizando el framework Flutter y una arquitectura basada en microservicios. Esta investigación se apoyará en las bases de datos científicas a las que tiene acceso la Universidad Técnica del Norte y se utilizará un gestor bibliográfico para gestionar las referencias encontradas.
2. En el segundo objetivo se seguirá la metodología Scrum, organizadas en tres fases clave: planificación y definición, ejecución del desarrollo y revisión. A través de un ciclo de sprints, se desarrollarán las funcionalidades específicas del servicio de mensajería en tiempo real, lo que permitirá iterar sobre el producto para su mejora continua y adaptación fácil a los cambios y a las necesidades de los usuarios [5], [6].
3. Tras haber finalizado el desarrollo se producirá a realizar una evaluación de la calidad en uso del servicio utilizando la norma ISO/IEC 25022 a través de una encuesta de usabilidad con los usuarios del sistema [6].

## Justificación

### Justificación tecnológica

El desarrollo del servicio "Red Social de Mensajería" para la plataforma móvil de la Universidad Técnica del Norte es una respuesta innovadora a las crecientes demandas de comunicación en el entorno académico moderno. Este proyecto refleja un entendimiento claro de cómo las herramientas tecnológicas están transformando las formas en que docentes y estudiantes se comunican y colaboran. Al integrar un servicio de mensajería oficial y centralizado, la universidad no solo mejora la accesibilidad de la información, sino que también se ajusta a los patrones digitales actuales, proporcionando un canal de comunicación eficiente y disponible.

### Justificación social

Este proyecto está alineado con el Objetivo de Desarrollo sostenible número cuatro, "Educación de calidad" [7], demostrando el compromiso de la Universidad Técnica del Norte en mejorar los procesos de interacción académica. Los principales beneficiarios, tanto docentes como estudiantes, se verán favorecidos al contar con un servicio de mensajería eficiente que facilita la comunicación directiva, promoviendo un ambiente colaborativo y enriqueciendo la experiencia educativa en su conjunto.

## Antecedentes

*Tabla 1. Antecedentes*

| Investigación | Aporte |
|---|---|
| **Contexto: Local**<br>**Desarrollo de una aplicación móvil utilizando el framework Flutter para fomentar el área turística del GAD de Pedro Moncayo** [8]**.** | Este estudio demuestra cómo la utilización de Flutter y metodologías ágiles como Scrum pueden contribuir eficazmente al desarrollo de aplicaciones móviles, resultando en soluciones tecnológicas bien aceptadas por los usuarios. La presente propuesta aborda el desarrollo de un servicio de gestión de mensajería con el framework Flutter. |
| **Contexto: Local**<br>**Comparación de la eficacia entre tres metodologías ágiles en el desarrollo de software** | El trabajo de grado proporciona una base teórica sobre la norma ISO/IEC 25022 y su aplicación práctica en la mejora de la calidad del software. El presente trabajo, sin embargo, se centra en el uso de la norma ISO/IEC 25022 para evaluar |

| | |
|---|---|
| **basada en el estándar de la ISO/IEC 25022** [6]**.** | específicamente la calidad de uso de una plataforma de comunicación académica. |
| **Contexto: Nacional**<br>**Las plataformas digitales como herramientas comunicacionales entre los docentes y estudiantes de la unidad educativa Luis Augusto Mendoza Moreira del cantón La Libertad** [9]**.** | El trabajo demuestra cómo las herramientas digitales pueden convertirse en un soporte fundamental para la interacción académica, tanto en términos de educación como de comunicación. La propuesta actual tiene como objetivo el desarrollo de un servicio de mensajería en tiempo real dentro de una plataforma universitaria. |
| **Contexto: Nacional**<br>**Desarrollo de una red social académica, que optimice la comunicación y difusión de información entre estudiantes, docentes y autoridades, de la Carrera de Ingeniería en Software de la Universidad de las Fuerzas Armadas - ESPE sede Latacunga** [10]**.** | Este estudio detalla el desarrollo de una red social académica utilizando metodologías ágiles como Scrum. El presente trabajo se enfoca en el desarrollo de un servicio de mensajería en tiempo real dentro de una plataforma de comunicación académica, centrado específicamente en la Universidad Técnica del Norte |
| **Contexto: Internacional**<br>**Educación y redes sociales: aliados en tiempos de pandemia** [11]**.** | Este artículo refuerza la idea de que las redes sociales y la educación virtual complementan la enseñanza, pero no sustituyen la interacción presencial. El presente trabajo busca el desarrollo de una red social de mensajería en la educación superior, específicamente dentro de una plataforma universitaria. |
| **Contexto: Internacional**<br>**Websocket to Support Real Time Smart Home Applications** [12]**.** | El estudio compara los métodos de polling y websocket para la transmisión de datos en tiempo real, concluyendo que websocket es más eficiente en términos de uso de ancho de banda y consumo de memoria. El aporte del trabajo será el uso de websocket para para desarrollar el servicio de |

| | |
|---|---|
| | mensajería en tiempo real en una plataforma académica. |

# CAPÍTULO II

# MARCO TEÓRICO

En este capítulo, se estableció una revisión bibliográfica para fundamentar los elementos teóricos necesarios que soportan la implementación del servicio "Red Social de Mensajería". Esta investigación se apoya en el marco de trabajo SCRUM para gestionar el desarrollo ágil del proyecto y en el estándar ISO/IEC 25022 para evaluar la calidad en uso del servicio, garantizando su funcionalidad, eficacia y satisfacción en el contexto académico.

## 2.1 Las aplicaciones móviles en el contexto educativo

Como resultado de los avances tecnológicos actuales, el incremento en el uso de dispositivos electrónicos, plataformas y herramientas para la comunicación ofrece a las personas un lugar ilimitado de posibilidades para interactuar y comunicarse. Además, fomenta el aprendizaje mediante dispositivos, recursos abiertos y herramientas digitales, expandiendo las oportunidades educativas más allá del entorno tradicional del aula [13].

### 2.1.1 Concepto de aplicaciones móviles

Las aplicaciones móviles consisten en programas diseñados para funcionar en dispositivos portátiles como teléfonos inteligentes, tabletas u otros equipos similares, permitiendo su uso en cualquier lugar y momento [14].

### 2.1.2 Transformación digital en la educación superior

En la actualidad, la educación superior está una transformación digital única. La incorporación de herramientas tecnológicas en el entorno educativo ha transformado las formas tradicionales de educación [15].

La crisis generada por la pandemia de COVID-19 impulsó con mayor urgencia la transición de las universidades hacia enfoques educativos híbridos o completamente virtuales. Este cambio evidenció no solo la importancia de implementar estrategias tecnológicas adecuadas, sino también la necesidad de que el personal docente desarrolle habilidades digitales para afrontar los desafíos de un entorno educativo digitalizado y superar las barreras al cambio inherentes a este proceso [16].

### 2.1.3 Aplicaciones móviles académicas en Ecuador

En el contexto local, todas las personas cuentan con un dispositivo móvil, y la mayoría de las personas lo ha usado para fines académicos [17]. Esto provoca que el uso de aplicaciones móviles sea frecuente, permitiendo a estudiantes y docentes acceder a recursos educativos, gestionar tareas y comunicarse en tiempo real desde cualquier lugar.

### 2.1.4 Beneficios de las aplicaciones móviles en universidades

De acuerdo con Cornejo et al., (2021), los principales beneficios del uso de aplicaciones móviles y tics incluyen el impulso de la alfabetización digital y audiovisual, el fortalecimiento de la autonomía en los estudiantes, la promoción del trabajo colaborativo en equipo y la mejora en la comunicación entre la comunidad educativa. Además, estos instrumentos facilitan a los alumnos acceder a materiales educativos y ser partícipes de actividades de aprendizaje desde cualquier lugar, promoviendo una educación más flexible y personalizada.

### 2.1.5 Limitaciones y desafíos en el uso de aplicaciones móviles

Como lo destaca [19], los principales desafíos en el uso de plataformas móviles como herramientas de aprendizaje incluyen garantizar la calidad del aprendizaje, seleccionar y utilizar adecuadamente las tecnologías disponibles y evaluar su impacto en los resultados educativos. Por su parte [20], destacan que la conectividad desigual y las limitaciones en el acceso al entorno digital representan barreras significativas para su implementación efectiva en el ámbito académico.

### 2.1.6 Impacto de las aplicaciones móviles en el aprendizaje

Como lo describe [21], la influencia beneficiosa sobre aplicaciones móviles en el aprendizaje de matemáticas, destacan beneficios como la mejora del rendimiento académico, el aumento de la motivación y la promoción de actitudes positivas hacia el aprendizaje. Estas aplicaciones facilitan el proceso de enseñanza-aprendizaje al permitir a los alumnos mantener autonomía y recibir apoyo del profesor de ser necesario.

### 2.1.7 Ejemplos de aplicaciones móviles usadas en instituciones educativas

En el ámbito educativo, las aplicaciones móviles han sido ampliamente utilizadas para el aprendizaje de idiomas, destacándose por su carácter interactivo y accesible.

Herramientas como Duolingo ofrecen a los estudiantes la oportunidad de practicar vocabulario, habilidades auditivas y gramática de forma flexible, adaptándose a diversos contextos y horarios. Estas plataformas son valoradas por su facilidad de uso y la presentación atractiva de los contenidos, lo que incrementa el interés de los usuarios en el proceso de aprendizaje [22].

## 2.2 Mensajería en tiempo real

Según [23], una aplicación en tiempo real es un software que facilita una comunicación bidireccional, dinámica y continua entre un servidor y los usuarios conectados a este. Con este contexto la mensajería en tiempo real se presenta como un concepto que permite a los usuarios, mediante conexiones establecidas, enviar y recibir mensajes de manera instantánea.

### 2.2.1 Concepto y evolución de los sistemas de mensajería

En referencia a lo que menciona [24], los sistemas de mensajería son plataformas que facilitan la comunicación en tiempo real entre usuarios a través de dispositivos conectados a Internet. Estas herramientas operan mediante programas especializados y protocolos, permitiendo el intercambio de mensajes y, en algunos casos, apoyándose en servidores para gestionar las conexiones y garantizar el flujo eficiente de datos.

Los sistemas de mensajería han evolucionado significativamente, pasando del uso predominante del correo electrónico, que carecía de inmediatez y confirmación de recepción, a herramientas más dinámicas como HipChat, Skype Empresarial y Slack [25].

### 2.2.2 Componentes principales de los sistemas de mensajería

Según lo descrito por [26], los sistemas de mensajería se componen principalmente de cuatro elementos clave: el cliente y el servidor de mensajería, el protocolo de comunicación y la conexión de red. Además, es fundamental considerar la autenticación como un componente esencial, ya que garantiza que los usuarios puedan establecer conexiones seguras y proteger la integridad de los datos intercambiados.

La Figura 3 destaca como los cuatro elementos clave que interaccionan entre sí.

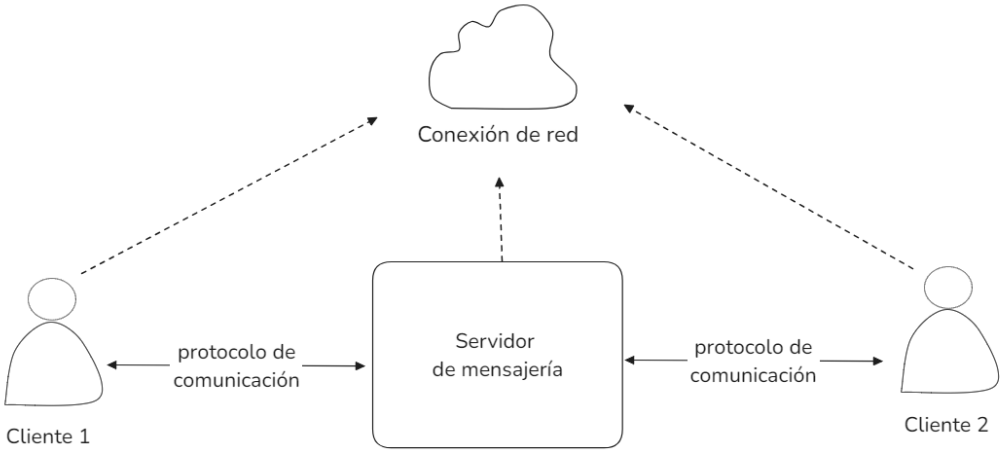



*Figura 3. Componentes principales de los sistemas de mensajería*

### 2.2.3 Tecnologías para la mensajería en tiempo real

#### 2.2.3.1 Concepto de WebSocket

De acuerdo con [27], los WebSockets son una tecnología esencial en los sistemas de mensajería en tiempo real, ya que permiten una comunicación continua y bidireccional entre el cliente y el servidor. Esta característica resulta indispensable para garantizar el intercambio inmediato de información en aplicaciones de mensajería.

#### 2.2.3.2 Handshake en WebSockets

El handshake en WebSockets es el procedimiento inicial donde el cliente y el servidor establecen una conexión. Este proceso comienza con una solicitud HTTP del cliente al servidor, solicitando transición del protocolo HTTP al protocolo WebSocket. Una vez que el servidor lo aprueba, la conexión se actualiza para utilizar el formato de datos binario propio de WebSocket, permitiendo la comunicación en tiempo real [28].

La Figura 4 muestra la arquitectura de un WebSocket.

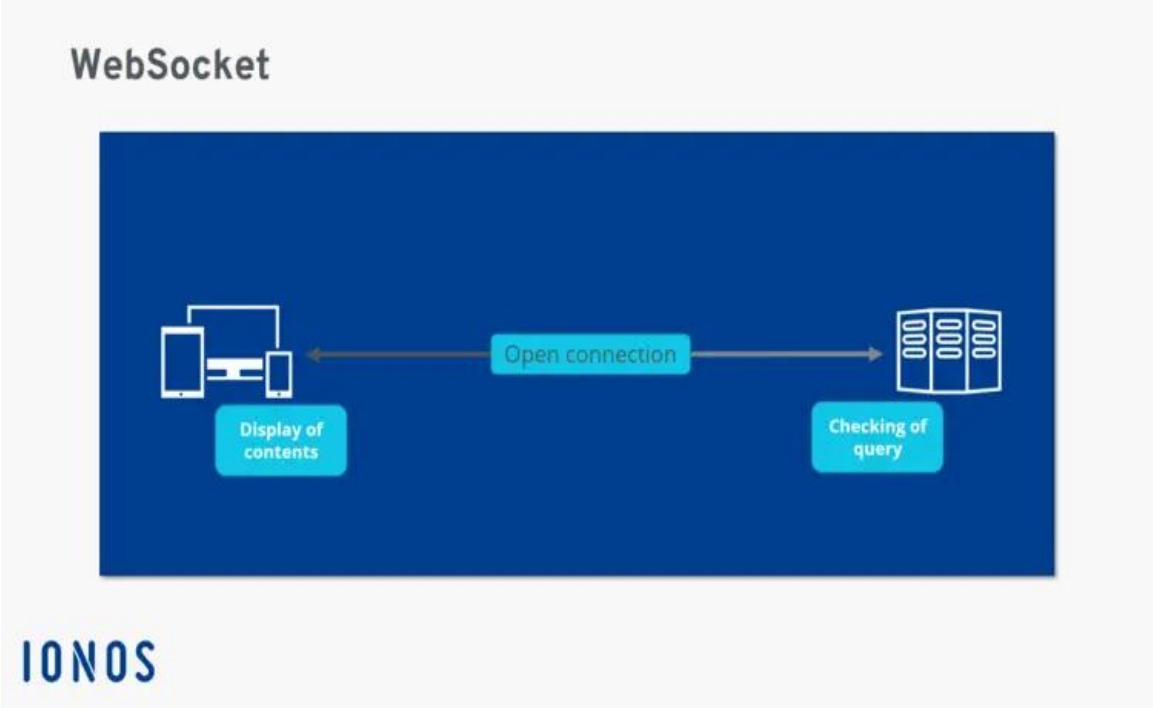



*Figura 4. Arquitectura de WebSocket*

Fuente: [29]

### 2.2.4 Aplicaciones de mensajería en el contexto académico

Las aplicaciones de mensajería han demostrado ser herramientas clave en el contexto académico, estas, como WhatsApp, facilitaron el intercambio de materiales educativos, la comunicación entre estudiantes y docentes, y el aprendizaje colaborativo [20].

#### 2.2.4.1 Ventajas y desventajas para estudiantes y docentes

Las aplicaciones de mensajería en el ámbito educativo destacan por facilitar la comunicación instantánea, promover la colaboración y permitir el acceso desde dispositivos móviles. Sin embargo, enfrentan limitaciones como distracciones, desigualdad en el acceso a la tecnología y la informalidad en las interacciones, que pueden afectar la calidad del aprendizaje [20].

La Tabla 2 describe las ventajas y desventajas sobre el uso de aplicaciones de mensajería en el contexto académico.

*Tabla 2. Ventajas y desventajas del uso de aplicaciones de mensajería en el contexto académico*

| Ventajas | Desventajas |
|---|---|
| **Facilitan el acceso rápido a la información y la comunicación grupal.** | Incrementan la brecha tecnológica entre estudiantes con diferente acceso a dispositivos y conectividad. |
| **Fomentan la colaboración mediante actividades grupales e intercambio de ideas.** | Pueden generar desigualdades sociales, especialmente en poblaciones vulnerables, como zonas rurales o con bajos recursos económicos. |
| **Permiten el aprendizaje asíncrono, mejorando la autonomía estudiantil.** | La falta de formación en el manejo de tecnologías afecta tanto a docentes como a estudiantes, limitando su efectividad educativa. |
| **Ofrecen una plataforma accesible para compartir materiales educativos y didácticos.** | Riesgo de uso indebido o distracciones al utilizar estas herramientas con fines no educativos. |

Fuente: adaptado de [20]

#### 2.2.4.2 Casos de éxito en universidades

Durante la crisis sanitario del COVID-19, la Universidad Nacional Autónoma de Honduras (UNAH) sobresalió por integrar herramientas tecnológicas, incluidas las aplicaciones de mensajería instantánea, como estrategias clave para la educación en línea. Un 54.72% de los docentes empleó WhatsApp para la comunicación y seguimiento de estudiantes, demostrando su efectividad para adaptarse al entorno digital. Además, el éxito en la transformación de prácticas educativas refleja la capacidad de los profesores para integrar estas tecnologías de manera adecuada en la enseñanza, contribuyendo a la continuada y calidad del aprendizaje en tiempos de crisis [30].

### 2.2.5 Seguridad en sistemas de mensajería

Según lo planteado por [25], una consideración clave al utilizar aplicaciones de mensajería es garantizar que cuenten con cifrado de datos, preferiblemente de extremo a extremo. Esta medida es esencial para proteger tanto la confidencialidad como la integridad de las comunicaciones, asegurando que los datos estén protegidos contra accesos no autorizados o alteraciones.

## 2.3 Metodología Scrum y Norma ISO/IEC 25022

Este apartado describe la metodología Scrum y su integración con la norma ISO/IEC 25022 para garantizar la calidad en el desarrollo del sistema de mensajería propuesto.

### 2.3.1 Metodología Scrum en el desarrollo de software

#### 2.3.1.1 Concepto de Scrum

Scrum es un marco de trabajo ágil que estructura el desarrollo de software mediante ciclos incrementales y repetitivos conocidos como sprints. Este enfoque promueve la colaboración del equipo, la adaptabilidad y la entrega continua de valor [31].

La Figura 5 muestra los elementos de la metodología Scrum.

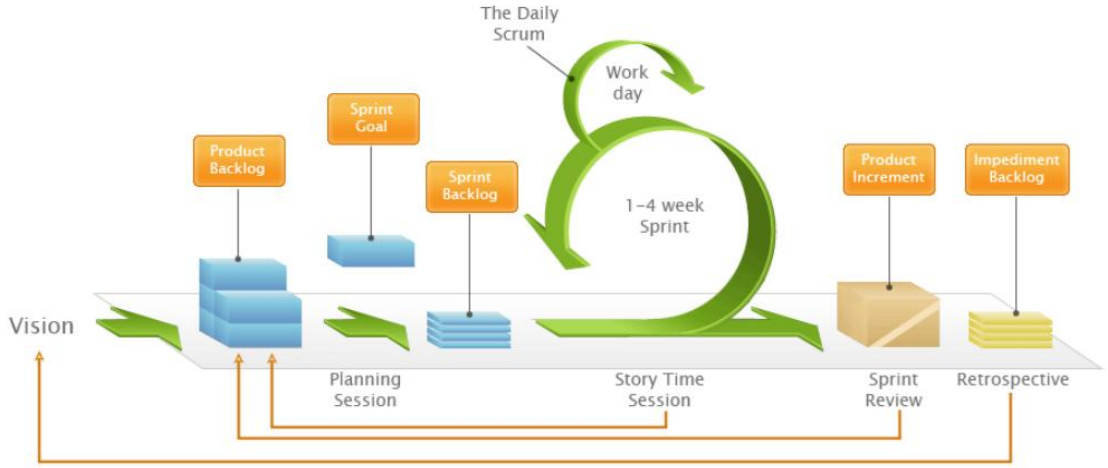



*Figura 5. Metodología Scrum*

Fuente: [32]

#### 2.3.1.2 Roles y estructura de Scrum

Se incluyen tres roles principales: el Scrum Master (facilitador del equipo), el Product Owner (gestor de requerimientos y prioridades) y el equipo de desarrollo (encargado de construir el producto) [33].

#### 2.3.1.3 Integración de Scrum con la norma ISO/IEC 25022

La combinación de Scrum con la norma ISO/IEC 25022 se implementa en las fases de desarrollo y evaluación de proyectos. En este contexto, Scrum contribuye a una gestión eficaz de proyectos a través de entregas incrementales y sistemáticas del producto,

facilitando así la mejora y adaptación constante del proceso de desarrollo. Simultáneamente, se utiliza el estándar ISO/IEC 25022 para comprobar los niveles de satisfacción del usuario[34].

### 2.3.2 Comparativa con otras metodologías ágiles

Scrum, Kanban, Extreme Programming (XP) y Dynamic Systems Development Method (DSDM) son metodologías ágiles diseñadas para mejorar la gestión de proyectos, cada una con enfoques y beneficios únicos. Scrum sobresale por su estructura basada en roles, ciclos iterativos llamados sprints y énfasis en la colaboración. Kanban, en cambio, se centra en la flexibilidad y la visualización continua del trabajo sin ciclos fijos, es fácil de adoptar y fomenta la colaboración, pero carece de un método sólido para estimar tareas.

Por otro lado, XP prioriza la calidad del software mediante prácticas como la programación en pareja y TDD, aunque demanda alto compromiso del equipo. DSDM busca entregar productos con rapidez y flexibilidad, involucrando activamente al cliente, pero exige experiencia y un equipo muy colaborativo [35].

*Tabla 3. Tabla comparativa de metodologías ágiles*

| Metodología | Características | Ventajas | Desventajas |
|---|---|---|---|
| **Scrum** | Roles definidos (Scrum Master, Product Owner, equipo de desarrollo).<br>Uso de sprints para interacciones.<br>Enfoque en colaboración y transparencia. | Alta adaptabilidad a cambios.<br>Entregas frecuentes que mejoran la satisfacción del cliente. | Complejos al inicio.<br>Dependencia del Scrum Master.<br>Resistencia al cambio en equipos tradicionales. |
| **Kanban** | Visualización del flujo de trabajo en tableros.<br>Sin roles definidos.<br>Modificaciones continuas sin ciclos fijos. | Alta flexibilidad.<br>Enfoque en mejora continua y calidad.<br>Colaboración fluida. | Falta de estructura para estimaciones.<br>Requiere gestión activa del flujo de trabajo. |
| **Extreme Programming (XP)** | Calidad del código como prioridad. | Mejora continua de la calidad del software. | Exigente para equipos no familiarizados. |

| | | | |
|---|---|---|---|
| | Prácticas como TDD, programación en pareja e integración continua. | Adaptabilidad a cambios en requisitos. | Alto compromiso necesario. |
| **DSDM** | Entrega rápida y flexible. Participación del cliente. | Alta colaboración y comunicación. Flexibilidad para los cambios en los requisitos | Complejidad en proyectos grandes. Requiere experiencia y compromiso del cliente. |

Fuente: adaptado de [35]

### 2.3.3 Introducción a las normas ISO en el desarrollo de software

#### 2.3.3.1 Importancia de los estándares internacionales

Los estándares internaciones, como los desarrolladores por la ISO, son fundamentales para respaldar la calidad y consistencia en el desarrollo de software. Estos lineamientos proporcionan criterios claros para cumplir con requisitos funcionales, desempeño y de calidad, minimizando errores y asegurando productos confiables y competitivos [36].

#### 2.3.3.2 Diferencias entre ISO/IEC 25010, 25022 y otras normas de calidad

Las normas ISO/IEC 25010 y 25022 se complementan al abordar distintos aspectos de la calidad del software. ISO/IEC 25010 define un modelo de calidad que incluye tanto la calidad del producto como la calidad en uso. Por su parte, ISO/IEC 25022 se enfoca en métricas específicas para evaluar la calidad en uso en escenarios reales, proporcionando un enfoque cuantitativo para medir el desempeño del software. En contraste, normas anteriores como ISO/IEC 9126 centradas en calidad interna y externa del producto, aunque no consideraban de forma integral el contexto de uso o la experiencia del usuario, aspectos fundamentales en las normas actuales [37].

### 2.3.4 Propósito y alcance de la ISO/IEC 25022

#### 2.3.4.1 Definición de calidad en uso

La calidad en uso se refiere a la capacidad de un producto software para permitir que los usuarios alcancen sus objetivos de manera efectiva, eficiente, segura y satisfactoria dentro de un contexto específico [37].

### 2.3.5 Aplicaciones en proyectos tecnológicos

La norma ISO/IEC 25022 se utiliza para especificar las métricas relacionadas con la calidad en uso, las cuales son esenciales para valorar la efectividad, eficiencia satisfacción en un sistema. En el contexto de proyectos tecnológicos, estas métricas permiten cuantificar la usabilidad y calidad del software desde la perspectiva del usuario [37].

### 2.3.6 Métricas de calidad según ISO/IEC 25022

#### 2.3.6.1 Eficacia: definición e indicadores

La eficiencia se enfoca en evaluar si el sistema permite a los usuarios lograr sus metas con exactitud y completez después de realizar una acción. Algunos indicadores asociados incluyen el número de tareas completadas, los objetivos alcanzados y la cantidad de errores en las tareas realizadas[38].

#### 2.3.6.2 Eficiencia: definición e indicadores

La eficiencia mide los recursos utilizados en función de la precisión y completitud con la que los usuarios logran sus metas. Los principales indicadores incluyen el tiempo requerido para completar tareas, la eficiencia temporal y la rentabilidad del proceso [38].

#### 2.3.6.3 Satisfacción: definición e indicadores

La satisfacción analiza el grado en que los usuarios se sienten satisfechos con el logro de sus objetivos, teniendo en cuenta los resultados y las consecuencias del uso del sistema. Entre los indicadores más relevantes se encuentran la utilidad perecida, la confianza del usuario en el sistema y su comodidad durante la interacción [38].

### 2.3.7 Casos de estudio y aplicaciones prácticas

#### 2.3.7.1 Ejemplo en plataforma educativa

Según [39], se empleó la norma ISO/IEC 25022 para analizar la calidad en uso del Sistema Académico Galileo Asistente de la Universidad Nacional del Centro del Perú. Ahí, se diseñó para modelo de calidad basado en el estándar ISO/IEC 25010 y se elaboró un instrumento apoyado en las directrices de la norma ISO/IEC 25022. Este procedimiento tuvo como objetivo evaluar la percepción de los usuarios administrativos respecto a la eficacia y eficiencia de las tareas realizadas a través del sistema.

## 2.4 Tecnologías utilizadas en el proyecto

En este apartado se detallan las tecnologías que se usarán para este proyecto como lo es Flutter, NodeJs y APEX.

### 2.4.1 Flutter

#### 2.4.1.1 Concepto de Flutter

Flutter es un kit de herramientas de desarrollo móvil de código abierto, creado por Google, que facilita la creación de aplicaciones móviles de forma eficiente y sencilla. Una de sus mayores fortalezas es la posibilidad de desarrollar un único código y desplegarlo en múltiples plataformas, como Android, iOS y ChromeOs. Además, Flutter proporciona un entorno completo para el desarrollo de aplicaciones, lo que permiten a los desarrolladores centrarse en las funcionalidades de sus aplicaciones sin preocuparse por detalles técnicos adicionales [40].

La Figura 6 detalla el diseño modular de Flutter.

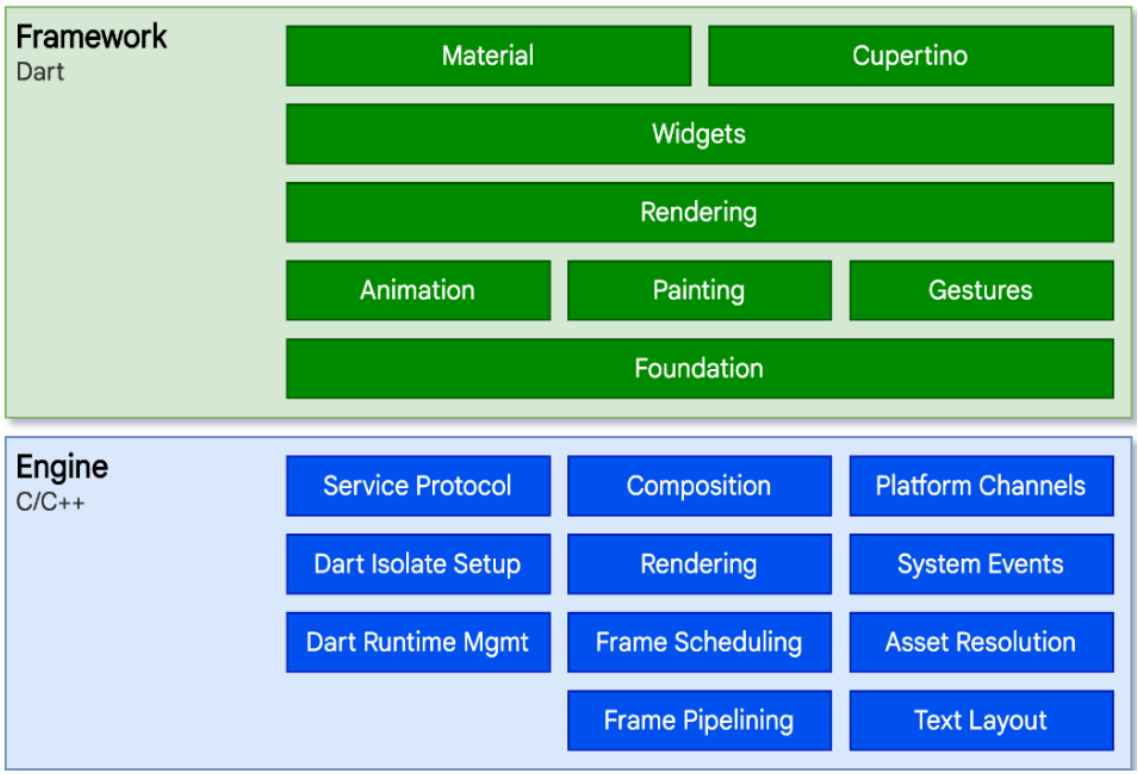



*Figura 6. Arquitectura de Flutter*

Fuente: [41]

#### 2.4.1.2 Comparativa entre Flutter y otras tecnologías multiplataforma

React Native es una herramienta desarrollada por Facebook que permite crear aplicaciones móviles para iOS y Android utilizando JavaScript y React. Su enfoque en la reutilización de código y componentes lo convierte en una opción eficiente [42].

*Tabla 4. Tabla comparativa: Flutter vs React Native*

| Categoría | Flutter | React Native |
|---|---|---|
| **Lenguaje de programación** | Utiliza Dart, diseñado por Google, ideal para desarrolladores familiarizados con lenguajes de orientados a objetos. | Basado en JavaScript, ampliamente adoptado y conocido, lo que facilita su aprendizaje. |
| **Rendimiento** | Ofrece alto rendimiento inicial y eficiencia gracias a su sistema de pruebas. | Depende de bibliotecas externas y módulos nativos, lo que puede impactar el rendimiento en algunos casos. |

| | | |
|---|---|---|
| **Hot Reload** | Permite visualizar cambios en tiempo real durante el desarrollo, mejorando la productividad. | También soporta "Hot Reload", brindando una experiencia de desarrollo ágil. |
| **Flexibilidad en desarrollo** | Diseñado para aplicaciones con interfaces complejas y animaciones avanzadas. | Ofrece gran flexibilidad gracias a su extensa gama de bibliotecas de terceros. |
| **Tamaño de la aplicación** | Genera aplicaciones con un tamaño inicial menor, alrededor de 4.7 MB para proyectos básicos. | Las aplicaciones básicas tienen un tamaño inicial de 7 MB. |
| **Comunidad y soporte** | Aunque está en constante crecimiento, su comunidad es más pequeña en comparación. | Cuenta con una comunidad más grande y activa, lo que facilita encontrar soporte y recursos. |
| **Facilidad de aprendizaje** | La curva de aprendizaje puede ser mayor debido a Dart, aunque es sencillo para desarrolladores con experiencia previa. | Más accesible gracias a la familiaridad con JavaScript y React. |

Fuente: adaptado de [42]

### 2.4.2 Node.js y arquitectura de microservicios

La implementación de Node.js como backend escalable facilita tanto el manejo de transacciones en tiempo real como el manejo eficiente de información. Asimismo, la incorporación de una arquitectura basada en microservicios junto con API RESTful optimiza la funcionalidad del sistema y se establece como un modelo replicable para diversas aplicaciones tecnológicas [43].

### 2.4.3 Bases de datos y tecnologías para datos en tiempo real

#### 2.4.3.1 Concepto de base de datos

Una base de datos se define como un sistema estructurado diseñado para guardar, administrar y acceder a la información de forma eficaz [44].

#### 2.4.3.2 Bases de datos relacionales vs. no relacionales

Los sistemas bases de datos relacionales estructuran la información en tablas interconectadas que permiten operaciones complejas y mantienen la integridad de los datos mediante reglas ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad). En contraste las bases de datos no relacionales (NoSQL) admiten estructuras de datos más flexibles como clave-valor, documentos, grafos y columnas [45].

### 2.4.4 Herramientas como Firebase y Oracle

#### 2.4.4.1 Concepto de Firebase

Firebase se define como un conjunto integral de herramientas diseñadas para facilitar el desarrollo de aplicaciones de alta calidad, promover el crecimiento de usuario y generar ingresos. La plataforma se presenta como una colección de soluciones que simplifican el proceso de creación de aplicaciones. Además, Firebase es compatible con diversos entornos de desarrollo, incluidos Android, iOS, JavaScript, Node.js, Python, C++ y Unity [46].

#### 2.4.4.2 Concepto de Oracle

Oracle, como se autodefine es una empresa que proporciona una solución integral compuesta por servicios completamente integrados de aplicaciones y plataformas en la nube [47].

#### 2.4.4.3 Concepto de Oracle APEX

Oracle APEX es una herramienta de desarrollo basada en el modelo de datos subyacente, lo que requiere migrar previamente las tablas SQL y sus datos al servidor correspondiente [48].

### 2.4.5 Node.js

#### 2.4.5.1 Concepto de Node.js

Node.js es una plataforma que facilita la ejecución de JavaScript en el servidor, ideal para aplicaciones de alto rendimiento como APIs y servicios en tiempo real. Esto se logra gracias a su diseño basado en eventos y modelo de entrada/salida no bloqueante. Express.js es uno de sus frameworks más populares por su simplicidad y flexibilidad, ofreciendo herramientas para estructurar aplicaciones web [49], [50].

#### 2.4.5.2 Concepto de NestJs

NestJs se basa en Node.js y Express, pero adopta un enfoque más estructurado inspirado en Angular. Usa TypeScript y organiza el código en módulos, facilitando la escalabilidad y mantenimiento en proyectos complejos. Además, incluye características como inyección de dependencias y middleware robusto, siendo ideal para arquitecturas grandes y bien definidas [49]. Esta arquitectura es ilustrada en la Figura 7.

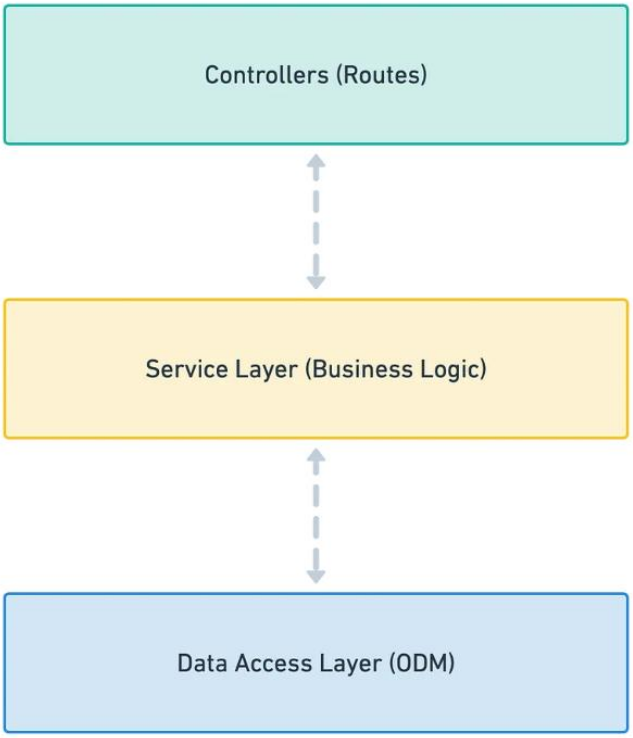



*Figura 7. Arquitectura de capas de NestJs*

Fuente: [51]

## 2.5 Trabajos relacionados

El estudio tuvo como objetivo diseñar y desarrollar una aplicación web y móvil en tiempo real que facilitara la gestión de usuarios y productos, destacando la evolución y el potencial de las tecnologías modernas. Para ello se implementó la metodología Scrum, permitiendo iteraciones incrementales para garantizar la satisfacción del cliente. Se desarrolló una solución funcional y sincronizada utilizando AngularJS para la aplicación web, Ionic para la versión móvil y Firebase con Cloud Firestore para el almacenamiento y sincronización de datos en tiempo real, lo que permitió gestionar usuarios y productos

de manera eficiente en ambas plataformas. Sin embargo, se identificaron desafíos en términos de escalabilidad y seguridad, especialmente vinculados al uso de Firebase.

Como mejoras futuras, se propone implementar medidas avanzadas de protección de datos, integrar servicios adicionales para ampliar funcionalidades y realizar pruebas de usabilidad con usuarios finales para optimizar la experiencia general [23].

El presente estudio se enfocó en la creación de una red social académica denominada "Sociademy", diseñada para optimizar la comunicación y el intercambio de datos entre alumnos, profesores y autoridades de la carrera de Ingeniería en Software en la Universidad de las Fuerzas Armadas – ESPE, Sede Latacunga. Para su implementación se empleó la metodología Scrum, que permitió trabajar de manera iterativa e incremental, respondiendo de forma ágil a las necesidades cambiantes del proyecto y gestionando eficazmente los riesgos. Las herramientas utilizadas incluyeron Laravel para construir la API Rest y gestionar la lógica del sistema, React Native para desarrollar una interfaz multiplataforma accesible desde web y dispositivos móviles, y MySQL para la administración de datos [10].

Este artículo detalla la creación de un sistema de información (S.I.T.A) diseñado para mejorar los procesos de transporte y mensajería en la empresa Audifarma S.A., empleando un enfoque metodológico similar al que usted propone. Se enfoca en las etapas de investigación y análisis de necesidades, el diseño y desarrollo del sistema, su implementación, y la divulgación del proyecto [52].

El estudio analizó el uso de WhatsApp como herramienta pedagógica para mejorar la interacción entre estudiantes y docentes en un curso virtual, integrándolo con Microsoft TEAMS mediante un guion educativo. La metodología incluyó actividades planificadas y encuestas para evaluar su efectividad. Los resultados mostraron que WhatsApp incrementó la participación y el intercambio de ideas, complementando las plataformas tradicionales. Sin embargo, se identificaron limitaciones como la dependencia de la conectividad y la saturación de mensajes. Se sugiere explorar herramientas adicionales y realizar pruebas en escenarios más amplios para mejorar la experiencia y el impacto en el aprendizaje [53].

El estudio analizó el uso de WhatsApp como herramienta educativa en zonas rurales para evaluar su efectividad en el desarrollo de habilidades lectoras. Se utilizó un enfoque metodológico mixto, combinando encuestas y entrevistas a docentes de una unidad educativa en Manabí, Ecuador, y se recopilaron datos sobre metodologías activas y el uso de TIC en el proceso de enseñanza. Los resultados mostraron que el 92% de los docentes valoran positivamente WhatsApp para fomentar la participación y mejorar la comprensión lectora, gracias a su accesibilidad para compartir recursos educativos. Sin embargo, se identificaron desafíos como el acceso desigual a internet y herramientas tecnológicas. Como trabajo futuro, se propone explorar la integración de otras plataformas digitales y estrategias para superar las barreras tecnológicas en la educación rural [54].

El estudio se centró en analizar el uso práctico y la efectividad de WebSockets y tecnologías en tiempo real en comparación con métodos como el polling HTTP, para lo cual se emplearon técnicas de monitoreo de llamadas a APIs de navegador a través del protocolo Chrome DevTools, recopilando datos detallados de su implementación en diversos sitios web. Los resultados destacaron la ventaja de los WebSockets en la reducción de sobrecarga y la comunicación bidireccional, aunque su adopción está limitada y los desarrolladores priorizan la simplicidad sobre la eficiencia al transmitir mayormente mensajes en texto. Entre las restricciones del trabajo, se incluyen dificultades para evaluar conexiones autenticadas y la baja atención a prácticas de seguridad, sugiriéndose investigar más sobre su uso en entornos autenticados y fortalecer la documentación para su implementación segura [55].

# CAPÍTULO III

# DESARROLLO DEL PROYECTO

Este capítulo detallará las fases de desarrollo del servicio “Red Social de Mensajería” para la plataforma móvil de la Universidad Técnica del Norte, aplicando la metodología Scrum.

En la Figura 8 se detalla el proceso de cada sprint que formará parte del desarrollo del proyecto.

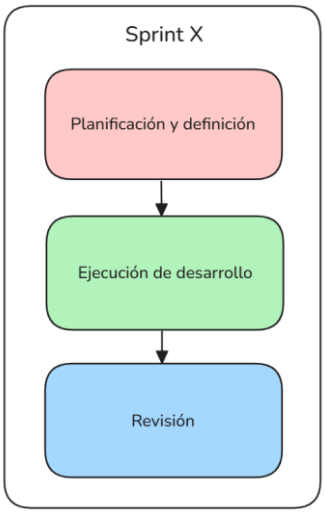



*Figura 8. Ciclo de Sprints*

## 3.1 Análisis

### 3.1.1 Equipo Scrum

En esta sección se especifica el equipo Scrum que hará parte en el transcurso del desarrollo del servicio “Red Social de Mensajería” con los roles y cargos que tendrán cada uno.

*Tabla 5. Equipo Scrum*

| Nombre | Rol | Cargo |
|---|---|---|
| **Antonio Quiña** | Product Owner. | Director del trabajo de grado<br>Director de UTN Móvil |
| **Jorge Rosero** | Scrum Master.<br>Equipo de Desarrollo. | Tesista |

| | | |
|---|---|---|
| **Dirección de Desarrollo Tecnológico e Informático** | Stakerholders. | Departamento de TI de la Institución |

### 3.1.2 Definición de requisitos

De acuerdo con la metodología Scrum, los requisitos deben obtenerse mediante historias de usuario, las cuales se adaptó el formato a partir de [56]. Para ello, se realizó una reunión con la Dirección de Desarrollo Tecnológico e Informático de la Universidad Técnica del Norte. Durante este encuentro, se definió el alcance del proyecto y se ofrecieron sugerencias sobre las funcionalidades del servicio que estarán disponibles en la aplicación móvil.

A continuación, se presentan las historias de usuario formuladas a partir de esta reunión:

*Tabla 6. Historia de Usuario 01*

| | | |
|---|---|---|
| **Código: HU-01** | **Título: Gestión de conversaciones individuales** | **Prioridad: Alta** |
| **Usuario: Docente/Estudiante** | Dependencia: N/A | Estimación: 36 h |
| **Descripción:** | Como estudiante o docente quiero visualizar las conversaciones de persona a persona. | |
| **Pruebas de aceptación:** | Al ingresar al sistema me deben mostrar los “Chats” que reflejan las conversaciones individuales. | |
| | Al estar en el sistema me debe aparecer nuevas conversaciones en caso de recibir mensajes. | |
| | Si se envía un mensaje al regresar a la vista de conversaciones individuales me debe aparecer la nueva conversación. | |

*Tabla 7. Historia de Usuario 02*

| | | |
|---|---|---|
| **Código: HU-02** | **Título: Gestión de conversaciones de asignaturas** | **Prioridad: Alta** |

| | | |
|---|---|---|
| **Usuario: Docente/Estudiante** | Dependencia: N/A | Estimación: 36 h |
| **Descripción:** | Como estudiante o docente quiero visualizar las conversaciones de asignaturas actuales. | |
| **Pruebas de aceptación:** | Al ingresar al sistema me deben mostrar las asignaturas en forma de conversaciones grupales. | |
| | Al estar en el sistema me deben aparecer las conversaciones ordenadas en forma descendente de la fecha del último mensaje. | |

*Tabla 8. Historia de Usuario 03*

| | | |
|---|---|---|
| **Código: HU-03** | **Título: Gestión de conversaciones de carrera y facultad** | **Prioridad: Alta** |
| **Usuario: Docente/Estudiante** | Dependencia: N/A | Estimación: 36 h |
| **Descripción:** | Como estudiante o docente quiero visualizar las conversaciones de carrera y facultad actuales. | |
| **Pruebas de aceptación:** | Al ingresar al sistema me deben mostrar la carrera y facultad en forma de conversaciones grupales. | |
| | Al estar en el sistema me deben aparecer las conversaciones ordenadas en forma descendente de la fecha del último mensaje. | |

*Tabla 9. Historia de Usuario 04*

| | | |
|---|---|---|
| **Código: HU-04** | **Título: Gestión de conversaciones de carrera y facultad de docentes** | **Prioridad: Media** |
| **Usuario: Docente** | Dependencia: N/A | Estimación: 24 h |
| **Descripción:** | Como docente necesito visualizar las conversaciones de la carrera y facultad dónde sólo existan docentes. | |
| **Pruebas de aceptación:** | Al ingresar al sistema me deben mostrar la carrera y facultad en forma de conversaciones grupales, donde los miembros sean sólo docentes. | |

| | |
|---|---|
| | Al estar en el sistema me deben aparecer las conversaciones ordenadas en forma descendente de la fecha del último mensaje. |

*Tabla 10. Historia de Usuario 05*

| | | |
|---|---|---|
| **Código: HU-05** | **Título: Gestión de visualización de mensajes** | **Prioridad: Alta** |
| **Usuario: Docente/Estudiante** | **Dependencia:** HU-01, HU-02, HU-03, HU-04 | Estimación: 40 h |
| **Descripción:** | Como docente o estudiante quiero visualizar los mensajes de cada conversación. | |
| **Pruebas de aceptación:** | Al ingresar a cualquier conversación me debe mostrar los mensajes de esa conversación ordenados descendentemente por fecha. | |
| | Al estar en la conversación me deben aparecer nuevos mensajes. | |
| | Al estar en la conversación me debe aparecer el nuevo mensaje en caso de enviar uno. | |

*Tabla 11. Historia de Usuario 06*

| | | |
|---|---|---|
| **Código: HU-06** | **Título: Gestión de envío de mensajes individuales** | **Prioridad: Alta** |
| **Usuario: Docente/Estudiante** | Dependencia: HU-01 | Estimación: 40 h |
| **Descripción:** | Como docente o estudiante quiero enviar mensajes a una persona específica. | |
| **Pruebas de aceptación:** | Al ingresar al sistema se debe tener una forma de identificar a la persona que quiero enviar el mensaje. | |
| | Al estar en la conversación de la persona que quiero enviar el mensaje, se debe poder enviar mensajes. | |

*Tabla 12. Historia de Usuario 07*

| | | |
|---|---|---|
| **Código: HU-07** | **Título: Gestión de envío de mensajes en las asignaturas** | **Prioridad: Alta** |
| **Usuario: Docente/Estudiante** | Dependencia: HU-02 | Estimación: 30 h |
| **Descripción:** | Como docente o estudiante quiero enviar mensajes a una asignatura específica. | |
| **Pruebas de aceptación:** | Al ingresar al sistema se debe tener una forma de identificar a la asignatura que quiero enviar el mensaje. | |
| | Al estar en la conversación de la asignatura que quiero enviar el mensaje, se debe poder enviar mensajes. | |

*Tabla 13. Historia de Usuario 08*

| | | |
|---|---|---|
| **Código: HU-08** | **Título: Gestión de envío de mensajes en la carrera y facultad** | **Prioridad: Alta** |
| **Usuario: Docente/Estudiante** | Dependencia: HU-03 | Estimación: 30 h |
| **Descripción:** | Como docente o estudiante quiero enviar mensajes a una asignatura específica. | |
| **Pruebas de aceptación:** | Al ingresar al sistema se debe tener una forma de identificar a la carrera o facultad que quiero enviar el mensaje. | |
| | Al estar en la conversación de la carrera o facultad que quiero enviar el mensaje, se debe poder enviar mensajes. | |

*Tabla 14. Historia de Usuario 09*

| | | |
|---|---|---|
| **Código: HU-09** | **Título: Gestión de envío de archivos como mensajes** | **Prioridad: Alta** |
| **Usuario: Docente/Estudiante** | **Dependencia:** HU-06, HU-07, HU-08 | Estimación: 50 h |
| **Descripción:** | Como docente o estudiante quiero enviar archivos como mensajes en las conversaciones. | |

| | |
|---|---|
| **Pruebas de aceptación:** | Al ingresar a una conversación, el sistema debería permitir enviar un archivo en forma de mensaje. |
| | Al ingresar a una conversación, el sistema debería permitir enviar una imagen en forma de mensaje. |

*Tabla 15. Historia de Usuario 10*

| | | |
|---|---|---|
| **Código: HU-10** | **Título: Gestión de estados de entrega de mensajes** | **Prioridad: Alta** |
| **Usuario: Docente/Estudiante** | **Dependencia:** HU-06, HU-07, HU-08 | Estimación: 50 h |
| **Descripción:** | Como docente o estudiante quiero saber en qué estado está el mensaje enviado. | |
| **Pruebas de aceptación:** | Al enviar un mensaje, el sistema debería mostrar una forma de identificar en qué estado se encuentra el mensaje. | |
| | Al enviar el mensaje, el sistema debería mostrar si el mensaje fue enviado. | |

*Tabla 16. Historia de Usuario 11*

| | | |
|---|---|---|
| **Código: HU-11** | **Título: Gestión de búsqueda de usuarios** | **Prioridad: Media** |
| **Usuario: Docente/Estudiante** | Dependencia: N/A | Estimación: 40 h |
| **Descripción:** | Como docente o estudiante quiero una forma de identificar la persona a quién voy a enviar el mensaje. | |
| **Pruebas de aceptación:** | Al ingresar al sistema, me debe permitir una forma de identificar a la persona quién quiero enviar un mensaje. | |

*Tabla 17. Historia de Usuario 12*

| | | |
|---|---|---|
| **Código: HU-12** | **Título: Gestión de privacidad de mensajes** | **Prioridad: Alta** |

| | | |
|---|---|---|
| **Usuario: Docente/Estudiante** | Dependencia: N/A | Estimación: 40 h |
| **Descripción:** | Como docente o estudiante quiero que los mensajes de una conversación solo los puedan leer quienes pertenecen a esa conversación. | |
| **Pruebas de aceptación:** | Al enviar un mensaje se debe cifrar el mensaje para que en la base de datos no se lo pueda leer. | |

*Tabla 18. Historia de Usuario 13*

| **Código: HU-13** | **Título: Gestión de la aplicación** | **Prioridad: Media** |
|---|---|---|
| **Usuario: Administrador** | Dependencia: N/A | Estimación: 32 h |
| **Descripción:** | Como administrativo quiero un panel de administración, que permita seleccionar el ciclo académico actual para la aplicación. | |
| **Pruebas de aceptación:** | Al entrar al sistema me debe permitir dar seguimiento a los mensajes que necesite. | |

*Tabla 19. Historia de Usuario 14*

| **Código: HU-14** | **Título: Visualización de miembros de una conversación** | **Prioridad: Baja** |
|---|---|---|
| **Usuario: Docente** | **Dependencia:** HU-02, HU-03 | Estimación: 20 h |
| **Descripción:** | Como docente quiero saber los miembros de una conversación grupal. | |
| **Pruebas de aceptación:** | Al entrar al sistema me debe permitir visualizar los miembros de una conversación grupal. | |

*Tabla 20. Historia de Usuario 15*

| **Código: HU-15** | **Título: Notificación individual** | **Prioridad: Alta** |
|---|---|---|
| **Usuario: Docente** | **Dependencia:** HU-01, HU-06 | Estimación: 20 h |

| | |
|---|---|
| **Descripción:** | Como docente y estudiante quiero recibir una notificación cuando se envía un nuevo mensaje de uno a uno. |
| **Pruebas de aceptación:** | Cuando se envíe un nuevo mensaje individual, me debe aparecer el nombre de la persona que envía el mensaje y el contenido. |

*Tabla 21. Historia de Usuario 16*

| **Código: HU-16** | **Título: Notificación grupal** | **Prioridad: Alta** |
|---|---|---|
| **Usuario: Docente** | **Dependencia:** HU-01, HU-06 | Estimación: 20 h |
| **Descripción:** | Como docente y estudiante quiero recibir una notificación cuando se envía un nuevo mensaje grupal. | |
| **Pruebas de aceptación:** | Cuando se envíe un nuevo mensaje grupal, me debe aparecer el título de la conversación, el nombre de la persona que envía el mensaje y el contenido. | |

### 3.1.3 Product Backlog

Tras la definición de las historias de usuario, el Product Owner se encarga de asignarles una prioridad y un orden para su desarrollo. Esta organización se puede ver en la Tabla 22.

*Tabla 22. Product Backlog*

| Orden | Código | Título | Estimación (Horas) |
|---|---|---|---|
| **1** | HU-02 | Gestión de conversaciones de asignaturas | 36 |
| **2** | HU-03 | Gestión de conversaciones de carrera y facultad | 36 |
| **3** | HU-01 | Gestión de conversaciones individuales | 36 |
| **4** | HU-06 | Gestión de envío de mensajes individuales | 40 |
| **5** | HU-07 | Gestión de envío de mensajes en las asignaturas | 30 |
| **6** | HU-08 | Gestión de envío de mensajes en la carrera y facultad | 30 |
| **7** | HU-05 | Gestión de visualización de mensajes | 40 |
| **8** | HU-09 | Gestión de envío de archivos como mensajes | 50 |
| **9** | HU-10 | Gestión de estados de entrega como mensajes | 50 |

| | | | |
|---|---|---|---|
| **10** | HU-04 | Gestión de conversaciones de carrera y facultad de docentes | 24 |
| **11** | HU-12 | Gestión de privacidad de mensajes | 40 |
| **12** | HU-11 | Gestión de búsqueda de usuarios | 40 |
| **13** | HU-13 | Gestión de la aplicación | 32 |
| **14** | HU-14 | Visualización de miembros de una conversación | 20 |
| **15** | HU-15 | Notificación individual | 32 |
| **16** | HU-16 | Notificación grupal | 32 |

## 3.2 Diseño – Sprint 0

Según [57], el Sprint 0 corresponde a una fase inicial en la que se establecen los cimientos técnicos y organizativos necesarios para el desarrollo, lo cual coincide con lo implementado en este proyecto. Durante este sprint, se realizó una transferencia tecnológica entre los miembros del equipo, quienes ya contaban con los microservicios y la arquitectura preliminar. La fase inicial se centró en establecer los principios arquitectónicos que guiarán el desarrollo del proyecto. En particular, se comenzó con la explicación detallada de la arquitectura del backend, donde se utilizarán microservicios basados en NestJS.

### 3.2.1 Resumen del Sprint 0

En el Sprint 0, se definió la arquitectura tecnológica del proyecto y se establecieron las bases para el desarrollo del software. Se realizó una transferencia de conocimientos entre los miembros del equipo sobre la arquitectura y los microservicios previamente desarrollados. El foco principal de esta fase fue establecer la estructura que guiará el desarrollo, incluyendo la arquitectura del backend y los microservicios en NestJS, la arquitectura y librerías más importantes en la aplicación móvil, y arquitectura en general para el despliegue con Docker.

### 3.2.2 Arquitectura Tecnológica

En esta parte de describe cómo se estructuró la arquitectura del microservicio, incluyendo la arquitectura tanto del backend como de la aplicación móvil separada por módulos.

En la transferencia tecnológica se compartió una plantilla para acelerar el proceso de desarrollo del microservicio donde ya contaba con la lógica suficiente para la conexión a la base de datos y también los componentes necesarios para continuar con el desarrollo. En la Figura 9 se muestra la arquitectura de los microservicios la cual se divide en dos servidores, uno público que es para el Gateway, y otro privado, donde están los microservicios de todas las aplicaciones, cada aplicación tiene la capacidad de conectarse a la base de datos y cumplir con su respectiva función.

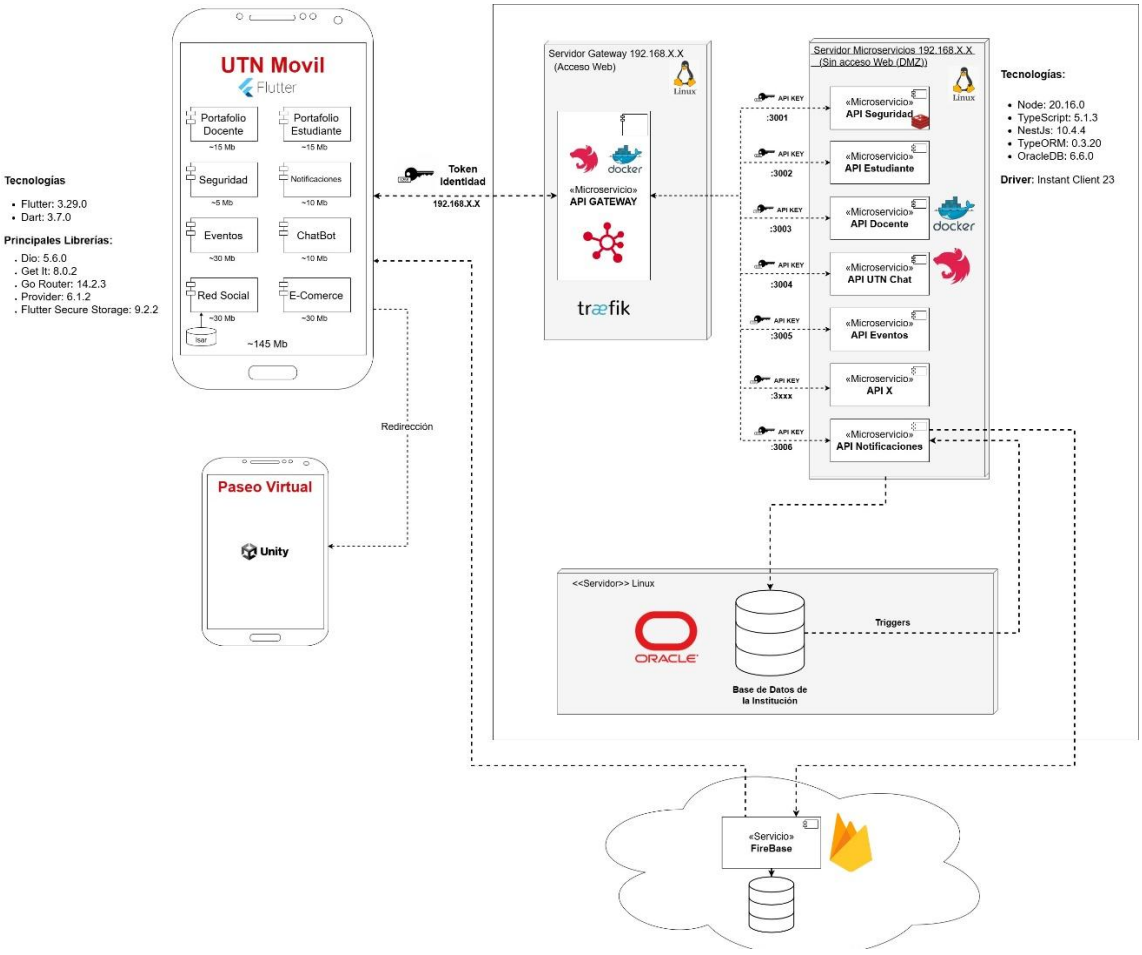



*Figura 9. Arquitectura compartida de microservicios*

La arquitectura del sistema está diseñada de manera que cada solicitud requiere un token de identidad, el cual es generado por el API de Seguridad. Posteriormente, el API Gateway se encarga de redirigir la solicitud al microservicio correspondiente. Para ello, el Gateway debe contar con el API Key registrado de cada microservicio. Además, cada microservicio establece su conexión a la base de datos mediante la librería TypeORM, la cual facilita el mapeo de las entidades a tablas y la transformación de las

consultas, simplificando el proceso de programación y mejorando la eficiencia en la interacción con la base de datos [58].

Además, aunque la arquitectura por defecto de NestJS sugiere el uso de dos capas principales “Controller” y “Service”, en este proyecto se adoptó una arquitectura adaptada que incluye una tercera capa denominada “Repository”. En esta estructura, el “Repository” es responsable de manejar directamente las consultas y la interacción con la base de datos, promoviendo una mejor separación de responsabilidades. La capa “Service” se encarga de la lógica de negocio, utilizando el “Repository” para acceder a los datos necesarios. Finalmente, la capa “Controller” gestiona las solicitudes HTTP, procesando elementos como parámetros “params”, parámetros de consulta “query params” y el cuerpo de la solicitud “body”, y delega la lógica en los servicios correspondientes. Esta adaptación permite un código más modular, mantenible y alineado con los principios de arquitectura limpia.

La Figura 10 representa la plantilla alojada en un repositorio de GitHub para acelerar el proceso, el cual ya cuenta con las configuraciones necesarias, para mantener un mismo estándar y tener un mismo código base.

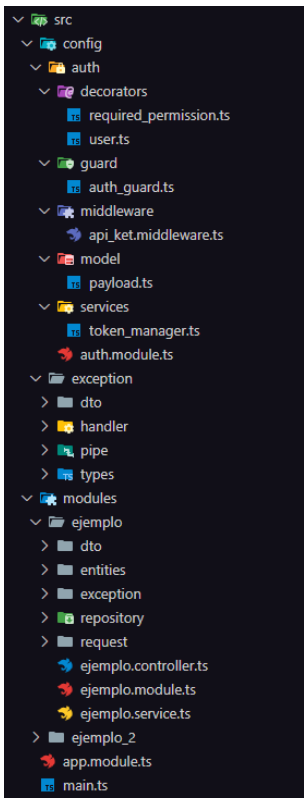



*Figura 10. Estructura de carpetas y archivos de la plantilla*

Durante la socialización, se compartió el repositorio de GitHub donde se aloja el proyecto principal de UTN Móvil, que incluye todas las aplicaciones en Flutter organizadas de manera modular y con un código limpio, garantizando su fácil mantenimiento y escalabilidad.

Se explicó la estructura de las ramas del repositorio: en la rama *Main* se encuentra el proyecto principal. La rama Desarrollo es donde se integran los nuevos proyectos una vez que están finalizados, para luego ser fusionados con la rama *Main* y puestos en uso. Cada proyecto puede contar con su propia rama de desarrollo, pero debe seguir una convención de nomenclatura que incluya un prefijo y el nombre del proyecto, para identificar claramente qué aplicación se está trabajando. Por ejemplo, para el proyecto de la Red Social de Mensajería, se creó la rama "*dev_red_social*", en la cual se genera una carpeta con el proyecto independiente, ubicada junto a los demás proyectos dentro de la carpeta Apps, tal como se ilustra en la Figura 11.

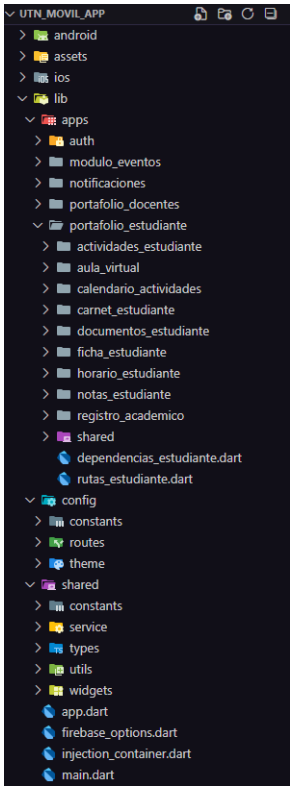



*Figura 11. Estructura de carpetas del proyecto UTN Móvil*

Dentro de la estructura del proyecto, se detalló que en la carpeta "*apps*" debe ubicarse cada aplicación individual, independiente y asociada al microservicio que se esté

desarrollando. Dentro de esta carpeta, las funcionalidades deben organizarse en subcarpetas basadas en casos de uso específicos. Siguiendo el principio de arquitectura limpia [59], las aplicaciones se estructuran en tres capas principales:

- Capa de dominio: Se encarga de la lógica de negocio, incluyendo las reglas y entidades fundamentales del sistema.
- Capa de infraestructura: Maneja las interacciones con sistemas externos, como bases de datos, APIs o servicios de almacenamiento.
- Capa de presentación: Se ocupa de la interfaz de usuario y la gestión de las interacciones con el usuario final, permitiendo que la aplicación sea accesible y usable.

Además, cada aplicación cuenta con un archivo de dependencias que registra los repositorios, servicios y proveedores necesarios para su funcionamiento. Estas dependencias se registran en el archivo "*injection_container.dart*", lo que permite gestionar la inyección de dependencias en la aplicación de manera eficiente.

La configuración de esta arquitectura se facilita mediante la librería GetIt, que es un contenedor de dependencias para Dart y Flutter [60]. GetIt permite gestionar y acceder a las dependencias de forma sencilla, promoviendo un código más limpio y desacoplado.

Además, se hace uso de la librería Provider, la cual se encarga de gestionar el estado de la aplicación y facilita la inyección de dependencias a nivel de widgets. Provider permite que los widgets puedan acceder de manera eficiente a los objetos que necesitan, sin necesidad de pasar explícitamente los datos entre ellos. Esta librería es especialmente útil para manejar el estado en aplicaciones Flutter y garantiza que los cambios en el estado de la aplicación se reflejen en la UI de manera reactiva [61].

Para ejecutar las solicitudes HTTP, se utiliza la librería Dio, que es un cliente HTTP para Dart que permite realizar peticiones de manera eficiente y gestionar aspectos como la interceptación, configuración global de solicitudes, y manejo de errores [62].

Para la gestión de la navegación dentro de la aplicación, se utiliza la librería *go_router*, la cual permite definir rutas de forma declarativa y estructurada en proyectos Flutter. Cada aplicación individual define sus propias rutas internas registrándolas

localmente, y luego estas son integradas globalmente en el archivo *config/routes/app_router.dart*. Este enfoque centralizado facilita el mantenimiento y escalabilidad del sistema, asegurando que todas las rutas de navegación estén correctamente organizadas y disponibles desde un solo punto de entrada. Además, *go_router* ofrece soporte para navegación con parámetros, redirecciones, rutas anidadas y manejo de autenticación, lo que la convierte en una solución robusta para aplicaciones con múltiples módulos [63].

### 3.2.3 Esquema Base de Datos Inicial

Para establecer una estructura coherente y escalable en el sistema, se elaboró la primera versión del esquema de base de datos utilizando una herramienta de modelado entidad-relación. Este esquema refleja las entidades fundamentales necesarias para el funcionamiento inicial del microservicio de mensajería, incluyendo estados de entrega, conversaciones y mensajes. La Figura 12 presenta el diagrama correspondiente, el cual servirá como base para futuras extensiones y refinamientos conforme se incorporen nuevas funcionalidades al sistema.

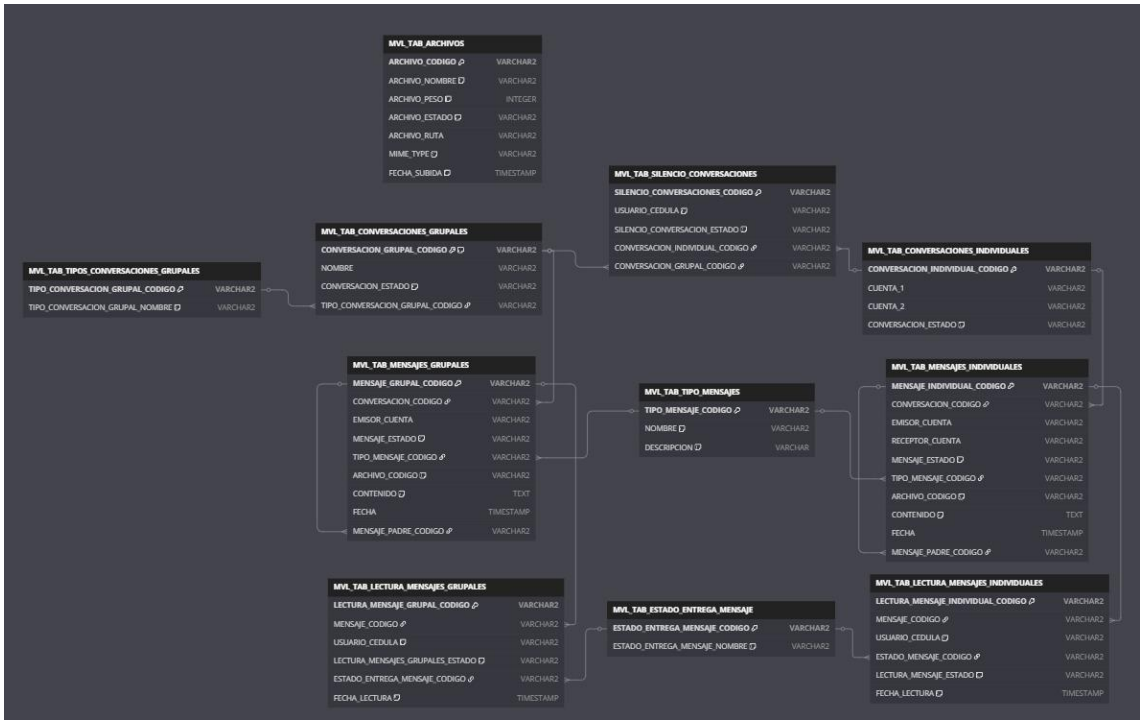



*Figura 12. Esquema base de datos entidad-relación inicial*

### 3.2.4 Diagrama de Proceso

De acuerdo con los requisitos funcionales y el esquema de base de datos inicial, se diseñó un diagrama de proceso que contempla los principales roles involucrados. En este flujo, tanto el estudiante como el docente pueden actuar como emisores o receptores de mensajes, mientras que la plataforma de mensajería, compuesta por la aplicación móvil y la API desarrollada, cumple un rol intermedio encargado de gestionar la interacción entre ambas partes. Adicionalmente, se incorpora un subproceso destinado al rol administrativo, cuya única funcionalidad es iniciar sesión y dar seguimiento a un mensaje o conversación específica en caso de requerirse. La Figura 13 representa este flujo general y la interacción entre los distintos roles.

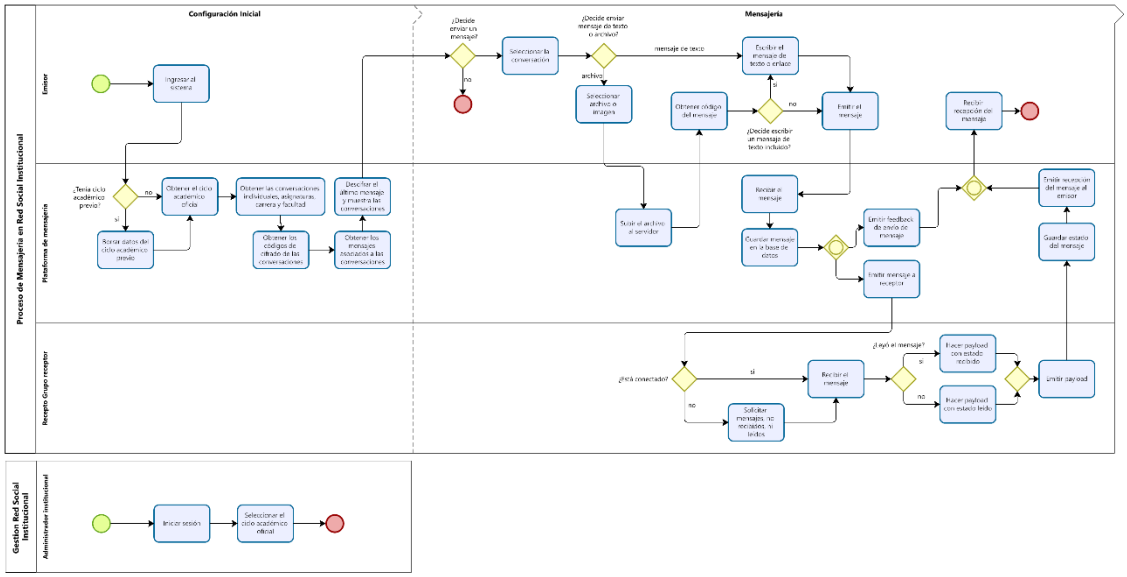



*Figura 13. Diagrama de proceso*

## 3.3 Desarrollo del Servicio “Red Social de Mensajería”

### 3.3.1 Sprint 1

**Planificación del Sprint 1**

**Objetivo del sprint:**

Desarrollar los primeros componentes con datos del sistema, estableciendo la arquitectura inicial y la lógica para los primeros servicios.

**Duración:**

El sprint se planificó para 4 semanas (del 03/03/2025 al 30/03/2025) donde se trabajará 30 horas por semana aproximadamente.

**Resultado:** Se definió el Sprint Backlog que se detalla a continuación.

*Tabla 23. Sprint Backlog - Sprint 1*

| Código Historia de Usuario | Título | Tarea | Horas |
|---|---|---|---|
| **N/A** | Configuración del entorno | Preparar herramientas, entorno local, control de versiones y base del proyecto | 2 |
| **HU-02** | Gestión de conversaciones de asignaturas | Implementar endpoint para obtener las asignaturas del docente | 6 |
| | | Implementar endpoint para obtener las asignaturas del estudiante | 6 |
| | | Implementar funcionalidad de obtener asignaturas como conversaciones grupales en aplicación móvil | 12 |
| **HU-03** | Gestión de conversaciones de carrera y facultad | Implementar endpoint para obtener la(s) carrera(s) y facultad del docente | 6 |
| | | Implementar endpoint para obtener la carrera y facultad del estudiante | 6 |
| | | Implementar funcionalidad de obtener la carrera y facultad como conversaciones grupales en aplicación móvil | 12 |
| **HU-11** | Gestión de búsqueda de usuarios | Implementar endpoint para obtener los usuarios por nombre | 8 |
| **HU-01** | Gestión de conversaciones individuales | Implementar funcionalidad para mostrar a los usuarios encontrados como conversaciones individuales en aplicación móvil | 8 |
| | | Implementar servicio para crear las conversaciones individuales | 8 |
| **N/A** | Diseño de base de datos para el backend | Mejorar el diseño de la base de datos de acuerdo con las normas manejadas en el DDTI | 2 |
| **HU-02, HU-03** | Gestión de conversaciones grupales | Implementar el servicio para crear las conversaciones grupales | 8 |
| **N/A** | Implementación de base de datos local | Diseñar una base de datos orientada en el usuario que está iniciado sesión | 3 |
| | | Implementar los datasources locales para crud de conversaciones y mensajes | 24 |
| | | **Total** | 113 |

Dado que se disponen de 30 horas por semana, el Sprint 1 contempla un total estimado de 120 horas distribuidas en las 4 semanas. Sin embargo, se ha reservado un margen de 7 horas para cubrir posibles imprevistos o emergencias, por lo que el total efectivo planificado en el Sprint Backlog es de 113 horas.

**Ejecución del desarrollo**

Para la ejecución de la tarea "Configuración del entorno", que incluye la preparación de herramientas, entorno local, control de versiones y base del proyecto, se comenzó configurando la plantilla compartida dentro de la organización del proyecto en GitHub. A partir de esta base, se realizaron diversos ajustes iniciales.

Se llevó a cabo una reunión con un funcionario del DDTI para revisar y afinar el modelo entidad-relación de la base de datos inicial. Una de las recomendaciones fue establecer una convención de nombres para los objetos de base de datos: las tablas deben comenzar con el prefijo MVL_TAB_, las vistas con MVL_VIEW_, y los procedimientos almacenados con MVL_PROC_, seguido del nombre descriptivo de su contenido o funcionalidad. También se sugirió incluir una columna ESTADO de tipo VARCHAR2(1) en las tablas que almacenen grandes volúmenes de datos o que no sean consideradas estáticas, utilizando los valores 'A' para indicar registros activos y 'I' para registros inactivos. Esta práctica permite realizar una limpieza lógica de datos mediante actualizaciones, o una limpieza física posterior utilizando sentencias DELETE.

El modelo entidad-relación ajustado conforme a estas directrices se muestra en la Figura 14, y sirvió como base para iniciar el desarrollo del sistema.

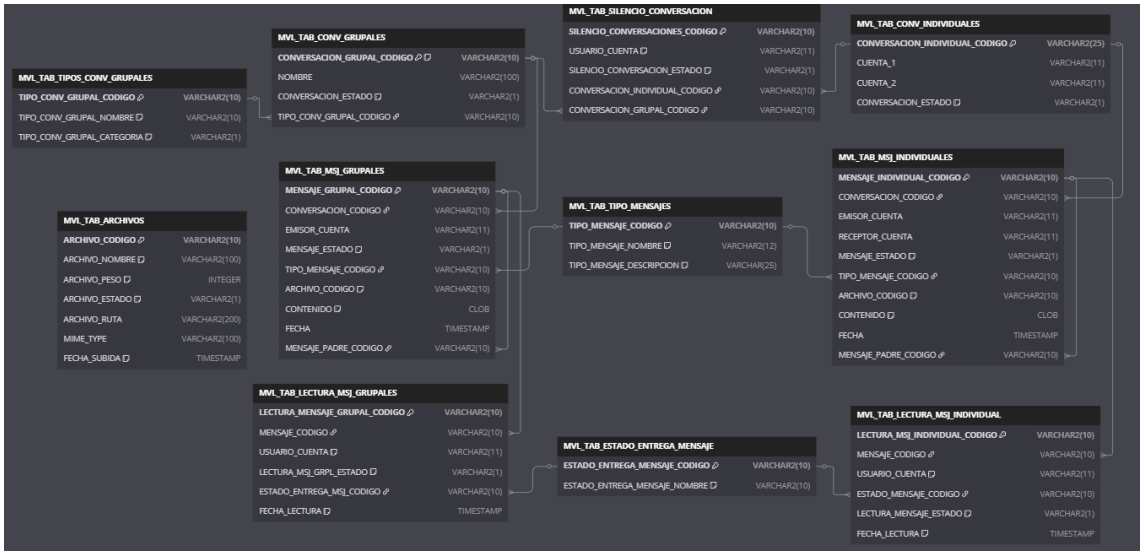



*Figura 14. Modelo entidad-relación ajustado*

Uno de los principales cambios aplicados durante el Sprint 1 fue la modificación en la estructura de carpetas del proyecto. Originalmente, los casos de uso estaban

organizados bajo la ruta *src/modules/caso_uso_#*; sin embargo, con el objetivo de mejorar la escalabilidad y evitar ambigüedades entre funcionalidades y endpoints, se adoptó una nueva estructura jerárquica basada en submódulos: *src/modules/submodule_#/caso_uso#*. Esta reorganización facilita la identificación y mantenimiento de funcionalidades específicas por contexto o responsabilidad. La Figura 15 ilustra esta nueva organización.

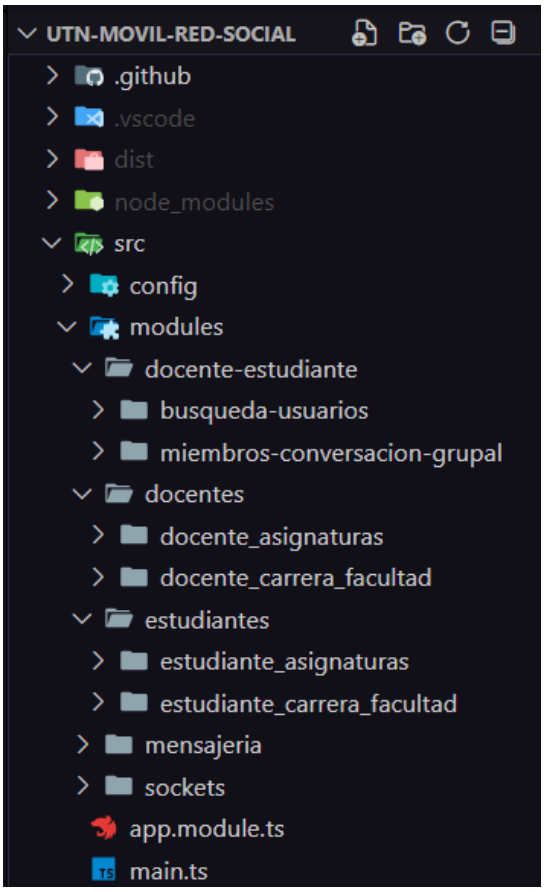



*Figura 15. Estructura de carpetas actual*

En la Figura 15 se observa que los submódulos han sido segmentados de acuerdo con los roles previstos dentro del sistema: docente y estudiante. Adicionalmente, se creó un submódulo específico para el manejo de Sockets. Para los roles mencionados, se agruparon las funcionalidades correspondientes —por ejemplo, asignaturas, carrera y facultad— asociadas a las historias de usuario HU-02 y HU-03.

También se añadió un submódulo denominado docente-estudiante, destinado a agrupar funcionalidades compartidas entre ambos roles. Estas funcionalidades no dependen del rol del usuario, como es el caso de la búsqueda de usuarios y la visualización

de miembros de una conversación grupal, correspondientes a las historias de usuario HU-11 y HU-14, respectivamente.

Cada submódulo mantiene una estructura interna compuesta por controller, service y repository, alineada con una arquitectura basada en REST API. En el caso de la arquitectura orientada a Sockets, se implementaron listeners, encargados de reaccionar ante eventos específicos. Estos listeners hacen uso de los services, los cuales, a su vez, se apoyan en los repositories para interactuar como la capa de datos.

Durante el desarrollo de los CRUDs de mensajes individuales, se identificó que resulta más conveniente utilizar entities con TypeORM mediante la inyección del decorador @InjectRepository, especialmente para operaciones como inserciones o actualizaciones que afectan pocos campos o un único registro. Una entity en TypeORM es una clase de TypeScript que representa una tabla en la base de datos; cada instancia de esta clase corresponde a una fila, y sus propiedades reflejan las columnas de dicha tabla.

Utilizar entities permite aprovechar métodos como: *.save()*, que no solo realizan la operación en la base de datos, sino que también retornan directamente el objeto insertado o actualizado, incluyendo los campos generados automáticamente como claves primarias o fechas de creación [58]. Esto evita la necesidad de realizar una consulta adicional para obtener dichos valores, ya que se devuelven listos para ser utilizados como un objeto JSON. En la Figura 16 se puede observar una implementación que combina ambos enfoques: el uso de entities para insertar datos, y consultas personalizadas para lecturas más complejas.

```typescript
@Injectable()
export class MensajesIndividualesRepository {
    constructor(
        @InjectRepository(MensajeIndividual)
        private readonly mensajesIndividualesRepository: Repository<MensajeIndividual>,

        @InjectDataSource() private readonly datasource: DataSource,
    ) { }

    public async createMensaje(createMensajeIndividualDto: CreateMensajeIndividualDto): Promise<MensajeIndividual> {
        try {
            return await this.mensajesIndividualesRepository.save(createMensajeIndividualDto);
        } catch (error) {
            console.log('[Error] al guardar mensaje individual: ', error);
        }
    }

    public async getMensajeIndividualByCodigo(codigoMensaje: string): Promise<MensajeIndividual> {
        return await this.mensajesIndividualesRepository.findOne({ where: { mensajeIndividualCodigo: codigoMensaje } });
    }

    public async getMensajesIndividualesByCuenta(cuenta: string): Promise<MensajeIndividualDto[]> {
        const consulta = OBTENER_MENSAJES_INDIVIDUALES_POR_CUENTA_CONSULTA;

        const result: Object[] = await this.datasource.query(consulta, [cuenta]);
        return plainToInstance(MensajeIndividualDto, result, { excludeExtraneousValues: true })
    }

    public async getMensajesIndividualesByConversacion(conversacionCodigo: string): Promise<MensajeIndividualDto[]> {
        const consulta = OBTENER_MENSAJES_INDIVIDUALES_POR_CONVERSACION_CONSULTA;

        const result: Object[] = await this.datasource.query(consulta, [conversacionCodigo]);
        return plainToInstance(MensajeIndividualDto, result, { excludeExtraneousValues: true })
    }
}
```

*Figura 16. Repository de mensajes individuales*

Para los casos en los que se requiere leer datos que no se ajustan a la estructura definida por una entity, o cuando las consultas son más complejas, es preferible utilizar la inyección del decorador @InjectDataSource, que permite acceder directamente a la instancia de la base de datos. Esto se combina con el uso de consultas SQL definidas como strings usando *backticks* (`), que es un tipo de comillas invertidas propio de JavaScript y TypeScript que permite declarar cadenas multilínea. Esta técnica facilita encapsular consultas largas de forma legible y mantenible dentro del código, permitiendo su ejecución directa mediante *dataSource.query()*. Además de resaltar que el uso de esta segunda (*@InjectDataSource*) forma hace que las consultas sean más rápidas comparadas con la primera forma (*@InjectRepository*).

Cabe destacar que, en caso de que un service requiera lógica definida en otro submódulo, se reutiliza directamente el service correspondiente, con el fin de evitar duplicación de lógica y respetar el principio de responsabilidad única. Así, un listener se convierte en el punto de entrada de una operación reactiva, que delega la ejecución al service especializado.

Para la arquitectura basada en Sockets, se planteó una estructura modular de eventos con el fin de evitar ambigüedades, facilitar la escalabilidad y prevenir conflictos en los nombres de eventos. Se definió una convención clara para nombrarlos utilizando la sintaxis:
*<caso_uso>.<acción>*

Un caso de uso corresponde a una funcionalidad específica dentro de los submódulos del sistema. Por ejemplo, en el submódulo de mensajería, existe el caso de uso mensajes individuales, por lo tanto, un evento asociado podría tener el nombre "mensaje-individual.enviar".

Si el caso de uso contiene dos o más palabras, se utiliza un guion (-) para separar cada término. Para distinguir la acción del caso de uso, se emplea un punto (.). Esta convención permite una gestión ordenada de los eventos y facilita su mantenimiento, ya que cada evento se ubica de forma lógica dentro de su correspondiente submódulo.

La definición de estos eventos se realiza en archivos tipo listener, cuya ruta sigue la siguiente estructura:
*sockets/listeners/<caso_uso>.listener.ts*

Cada archivo "*.listener.ts*" es una clase donde se inyectan las dependencias necesarias, y su única responsabilidad es registrar los eventos bajo la sintaxis establecida.

En la Figura 17 se observa cómo se agrupan los archivos listener en una carpeta específica del submódulo sockets, manteniendo la coherencia estructural con el resto del proyecto.

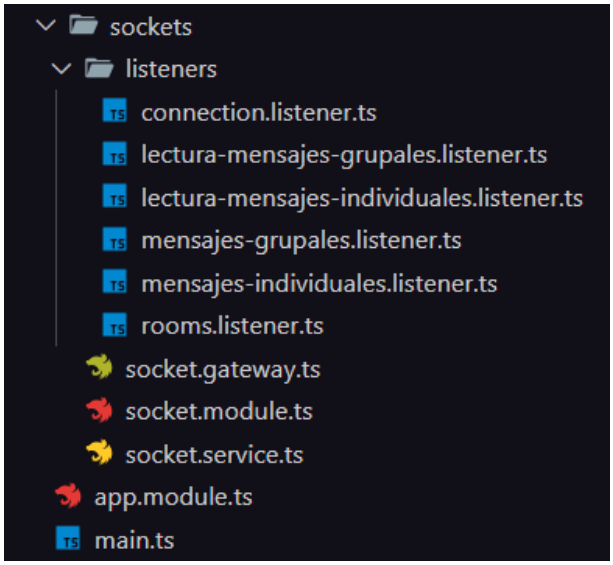



*Figura 17. Estructura de archivos en la carpeta socket actual*

Además, se ajustó el comportamiento del *guard* incluido en la plantilla inicial. Este guard es responsable de autenticar las conexiones y extraer la información de identificación (como la cédula o la cuenta) del token JWT enviado por los headers.

Anteriormente, se manejaban tres parámetros (cedula, cuentaEstudiante y cuentaDocente), lo cual generaba lógica repetitiva para determinar cuál estaba presente. Ahora, se simplificó este proceso dejando únicamente los parámetros cedula y cuenta, los cuales son definidos en el *token_manager.ts* mediante la decodificación del token, eliminando así ambigüedades y mejorando la claridad del flujo de autenticación.

Como en la aplicación ya existía un *auth_guard.ts* encargado de proteger las rutas de las solicitudes REST, se complementó esta lógica con un nuevo archivo llamado *ws_auth_guard.ts*, el cual replica la validación, pero aplicada específicamente a conexiones WebSocket (WS). Este guardia permite estandarizar los mecanismos de autenticación tanto en arquitectura REST como en arquitectura WS, logrando así una mayor homogeneidad en la gestión de seguridad para todas las solicitudes entrantes.

Esto es particularmente relevante en el archivo *socket.gateway.ts*, donde se configura el comportamiento del gateway de socket. Allí, se registra un listener que gestiona los eventos de conexión y desconexión, utilizando el *ws_auth_guard.ts* como punto de validación.

En la Figura 18, se observa la definición de un WebSocket Gateway, que actúa como punto de entrada principal para todas las conexiones WebSocket del sistema. Esta clase, decorada con *@WebSocketGateway*, permite interceptar y gestionar eventos como la conexión y desconexión de clientes.

Dentro del SocketGateway, se inyecta el *ConnectionListener*, que centraliza la lógica de conexión y limpieza de recursos, y el SocketService, que facilita el acceso compartido al servidor de sockets (*this.server*) en otros componentes del sistema. Esta arquitectura modular permite una mejor organización de la lógica reactiva del sistema y prepara el entorno para registrar múltiples listeners especializados por caso de uso.

```
@WebSocketGateway({
    // namespace: '/red_social',
    namespace: '/socket.io',
    cors: {
        origin: '*',
        allowedHeaders: ['Authorization']
    }
})
export class SocketGateway implements OnGatewayConnection, OnGatewayDisconnect {
    @WebSocketServer()
    server: Server;

    constructor(
        private readonly connectionListener: ConnectionListener,
        private readonly socketService: SocketService
    ) { }

    afterInit(server: Server) {
        this.socketService.setServer(server);
    }

    async handleConnection(client: Socket) {
        await this.connectionListener.handleConnection(client, this.server);
    }

    async handleDisconnect(client: Socket) {
        await this.connectionListener.handleDisconnect(client);
    }
}
```

*Figura 18. Configuración WebSocket Gateway*

Cuando un cliente establece una conexión mediante WebSocket, se ejecuta el método *handleConnection()* del archivo *connection.listener.ts*. Esta función inicia con la validación del token JWT a través del guardia WsAuthGuard, garantizando que únicamente los usuarios autenticados puedan establecer conexión. En caso de que la validación falle, el cliente es desconectado de forma inmediata.

Si la autenticación es exitosa, se extrae la información del usuario (como su cuenta o cédula) desde el token, se imprime un mensaje en consola confirmando la conexión, y el cliente es asignado a una sala utilizando su cuenta como identificador (*client.join(user.cuenta)*). A continuación, se registran de forma dinámica los listeners que gestionan los eventos del sistema. Estos archivos se encuentran en la carpeta listeners/, y están organizados por casos de uso específicos, como se muestra a continuación:

- rooms.listener.ts
- mensajes-individuales.listener.ts
- lectura-mensajes-individuales.listener.ts
- mensajes-grupales.listener.ts

Cada listener contiene los eventos que serán manejados por el cliente WebSocket, permitiendo encapsular la lógica por funcionalidad y asegurar una arquitectura modular, clara y escalable. De este modo, cada cliente solo queda suscrito a los eventos que le corresponden según el contexto de uso.

El método *handleDisconnect()* se encarga de gestionar la desconexión de los usuarios, removiéndolos de la sala correspondiente y registrando este evento en consola. Este proceso garantiza una salida limpia del sistema y evita mantener referencias activas innecesarias.

En la Figura 19 se visualiza el flujo completo de autenticación, asignación a salas y registro de listeners, así como el manejo de desconexión de clientes.

```
async handleConnection(client: Socket, server: Server) {
    const wsAuthGuard = new WsAuthGuard(this.tokenManager);
    try {
        const isAuthorized = await wsAuthGuard.canActivate({
            switchToWs: () => ({ getClient: () => client })
        } as any);

        if (!isAuthorized) {
            client.disconnect();
            return;
        }

        const user = client.data.user;
        console.log(`Usuario conectado: ${user.cuenta}`);
        client.join(user.cuenta);
        // console.log('Rooms actuales:', client.rooms);


        // Registrar los eventos que manejará el cliente
        this.roomsListener.register(client);
        this.mensajesIndividualesListener.register(client);
        this.lecturaMensajesIndividualesListener.register(client);

        this.MensajesGrupalesListener.register(client);

    } catch (error) {
        console.warn('Autenticasión fallida en WebSocket', error.message);
        client.disconnect();
    }
}
```

*Figura 19. Configuración de manejo de conexiones Socket*

A un usuario generalmente se le una a su propia room o sala para poder enviarle un mensaje, y a rooms o salas compartidas para emisión y escucha de eventos compartidos como enviar un mensaje a un asignatura, carrera o facultad. Como se observa en la Figura 17, al conectarse cualquier usuario se une a la sala con su cuenta y será fácil identificarlo ya que se alinea con la base de datos y el *payload* del mensaje para saber el emisor y receptor.

Para la emisión de estos eventos se configuró un SocketService (Servicio de Socket), que gestione la emisión de eventos a un usuario específico y a una room específica, pero para este último excluyendo al usuario emisor, ya que si no se agrega esta opción al emisor le llegará un nuevo mensaje enviado por sí mismo.

En la Figura 20 se observa los métodos para emitir a un usuario y room específicas, también un método de emitir a todos los usuarios conectados.

```
@Injectable()
export class SocketService {
    private server: Server;

    setServer(server: Server) {
        this.server = server;
    }

    emitToUser(usuarioCuenta: string, event: string, data: any) {
        if (this.server) {
            this.server.to(usuarioCuenta).emit(event, data);
        }
    }

    emitToRoom(room: string, event: string, data: any, excludeSelf: string) {
        if (this.server) {
            this.server.to(room).except(excludeSelf).emit(event, data);
        }
    }

    emitToAll(event: string, data: any) {
        if (this.server) {
            this.server.emit(event, data);
        }
    }
}
```

*Figura 20. Configuración de eventos*

Para la configuración del proyecto en Flutter de UTN Móvil, se mantuvo el mismo patrón de agrupación de archivos, se tengo dividido en *red_social/<módulo>/<submódulo>/<caso_uso>*, para un mejor entendimiento se visualiza la Figura 21 donde se tiene agrupado por funcionalidades académicas, de mensajería y la base de datos local.

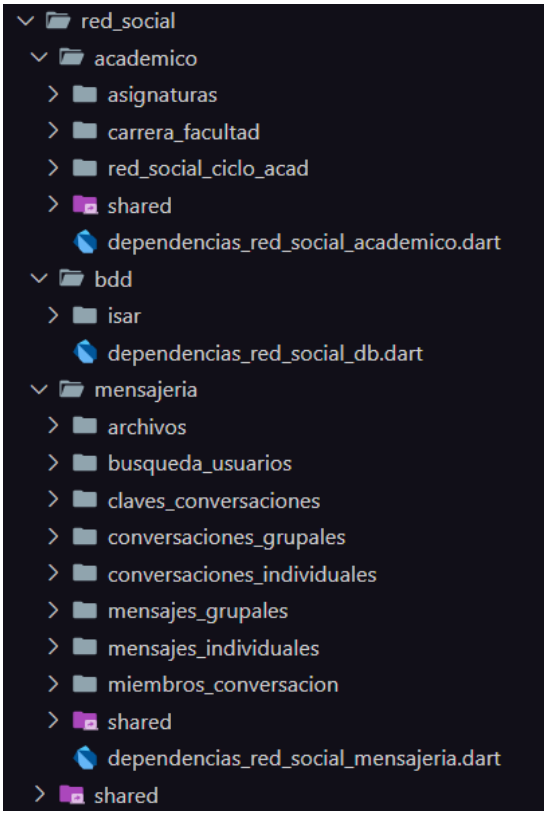



*Figura 21. Estructura de carpetas en Flutter*

En la Figura 21, se repite el patrón de tener las dependencias de cada submódulo por separado, esto para dar mantenimiento y que sea escalable, también ayuda para la inyección de dependencias y estas registrarlas en el archivo principal de dependencias de la aplicación, la Figura 22 representa el registro modula de dependencias en la App Red Social.

```dart
initDependenciasAppRedSocial(GetIt getIt) {
  if (AppConstants.environment != Environment.prod) {
    initDependenciasRedSocialDB(getIt);
    initDependenciasRedSocialAcademico(getIt);
    initDependenciasRedSocialMensajeria(getIt);
    initDependenciasRedSocialInfrastructure(getIt);
  } else {
    initDependenciasRedSocialDB(getIt);
    initDependenciasRedSocialAcademico(getIt);
    initDependenciasRedSocialMensajeria(getIt);
    initDependenciasRedSocialInfrastructure(getIt);
  }
}
```

*Figura 22. Registro de dependencias App Red Social*

Se aprecia que no tiene ningún cambio el uso del if, pero se deja para tener en caso de que se cambie el modo de trabajo, por ejemplo, si para cuando se desarrolla en modo mock para diseño no se quiere cambiar las dependencias que solo se necesiten en modo desarrollo o en modo producción, como puede ser la base de datos local.

En estas dependencias, se utiliza el patrón de diseño Singleton al momento de registrarlas en GetIt, especialmente para aquellas clases que deben tener una única instancia en toda la aplicación, como los controladores de mensajes o conversaciones. Aplicar este patrón permite que, al navegar entre vistas, no se creen nuevas instancias, lo cual es clave para mantener el estado y ejecutar métodos como recargar, sin inicializaciones innecesarias. Gracias a ello, es posible actualizar la interfaz de usuario de forma reactiva cuando llega un nuevo mensaje o se crea una nueva conversación, ya que todas las operaciones se realizan sobre la misma instancia compartida.

Las pantallas funcionales que se obtuvieron relacionadas a las historias de usuario: HU-01, HU-02, HU-03, HU-11, estas pantallas pueden cambiar para mejorar el diseño y experiencia de usuario.

La Figura 23 relacionada con la historia de usuario HU-03 que representa las conversaciones de carrera y facultad, en esta primera versión solo se plantea las conversaciones donde se encuentren los estudiantes y docentes, es decir, una conversación general para todos los miembros sin exclusión alguna.


*Figura 23. Vista de Carrera – versión 1*

La Figura 24 corresponde a la historia de usuario HU-11, relacionada con la funcionalidad de búsqueda de usuarios. Para implementarla en Flutter, se utilizó un delegate, que en este contexto es una interfaz especial que muestra una pantalla de búsqueda de forma independiente, como una ventana emergente o modal. Esta pantalla permite al usuario escribir un nombre y ver resultados filtrados en tiempo real.

En esta búsqueda se incluyen tanto docentes como estudiantes. Sin embargo, en el caso de los estudiantes, únicamente se muestran aquellos que están actualmente matriculados. Esto se debe a que la información se obtiene desde las tablas académicas activas de la base de datos, por lo que no es posible encontrar a estudiantes que ya no estén registrados en el periodo académico vigente.


*Figura 24. Vista de buscador de personas (delegate)*

La Figura 25 representa un mockup de las conversaciones individuales donde se visualiza el nombre de la persona, el último mensaje, el número de mensajes no leídos por el usuario que usa la aplicación y todas estas conversaciones ordenadas por fecha donde se ubican los más recientes primero.


*Figura 25. Vista de conversaciones individuales – versión 1*

**Revisión del Sprint 1**

Durante la revisión del Sprint 1 se evaluaron las actividades completadas, registrándose muy pocas observaciones, ya que la mayor parte del trabajo se centró en el desarrollo del backend y la lógica de negocio necesarias para completar las funcionalidades clave.

Se sugirieron únicamente algunos ajustes visuales menores, como la incorporación de estados de carga en las vistas y la unificación del diseño del buscador de usuarios con el de los chats individuales.

También se recomendó eliminar la descripción del ciclo académico en las conversaciones de carrera, trasladando dicha información a una sección adicional que se muestre al navegar hacia ellas.

### 3.3.2 Sprint 2

**Planificación del Sprint 2**

**Objetivo del sprint:**

Desarrollar la funcionalidad de envío de mensajes por socket y planteamiento de modo offline con base de datos local.

**Duración:**

El sprint se planificó para 4 semanas (del 01/04/2025 al 30/04/2025) donde se trabajará 30 horas por semana aproximadamente.

**Resultado:** Se definió el Sprint Backlog que se detalla a continuación.

*Tabla 24. Sprint Backlog - Sprint 2*

| Código Historia de Usuario | Título | Tarea | Horas |
|---|---|---|---|
| **N/A** | Diseño de base de datos local | Diseñar las entidades que representarán las tablas enfocadas en el usuario que tiene iniciado sesión. | 2 |
| **N/A** | Conexión de Socket entre el API y la aplicación móvil | Implementar endpoint para establecer la conexión de WebSocket | 6 |
| | | Implementar un socket service modular para usar en toda la aplicación | 6 |
| | | Implementar los sockets services específicos para cada caso de uso | 12 |
| **N/A** | Implementación de los datasources locales | Implementar el datasource local para conversaciones individuales | 10 |
| | | Implementar el datasource local para mensajes individuales | 10 |
| | | Implementar el datasource local para conversaciones grupales | 12 |
| | | Implementar el datasource local para mensajes grupales | 8 |
| **HU-06** | Gestión de mensajes individuales | Implementar la lógica para emitir mensajes individuales por socket | 20 |
| | | Implementar la lógica para escuchar mensajes individuales por socket | 20 |
| **HU-07, HU-08** | Gestión de envío de mensajes grupales | Implementar lógica unir a las romos y empezar con el envío de mensajes grupales | 10 |
| | | **Total** | 116 |

Dado que se disponen de 30 horas por semana, el Sprint 1 contempla un total estimado de 120 horas distribuidas en las 4 semanas. Sin embargo, se ha reservado un

margen de 4 horas para cubrir posibles imprevistos o emergencias, por lo que el total efectivo planificado en el Sprint Backlog es de 116 horas.

Ejecución del desarrollo

Para la ejecución de la tarea "Diseño de base de datos local", se utilizó una herramienta web de modelado entidad-relación. A partir del esquema de la base de datos del backend, se diseñó una nueva versión adaptada al contexto de uso local, en la cual las tablas se reestructuraron con un enfoque centrado exclusivamente en el usuario que ha iniciado sesión en la aplicación, en lugar de contemplar múltiples usuarios como en el backend.

La Figura 26 muestra las tablas o entidades diagramadas para la base de datos local.

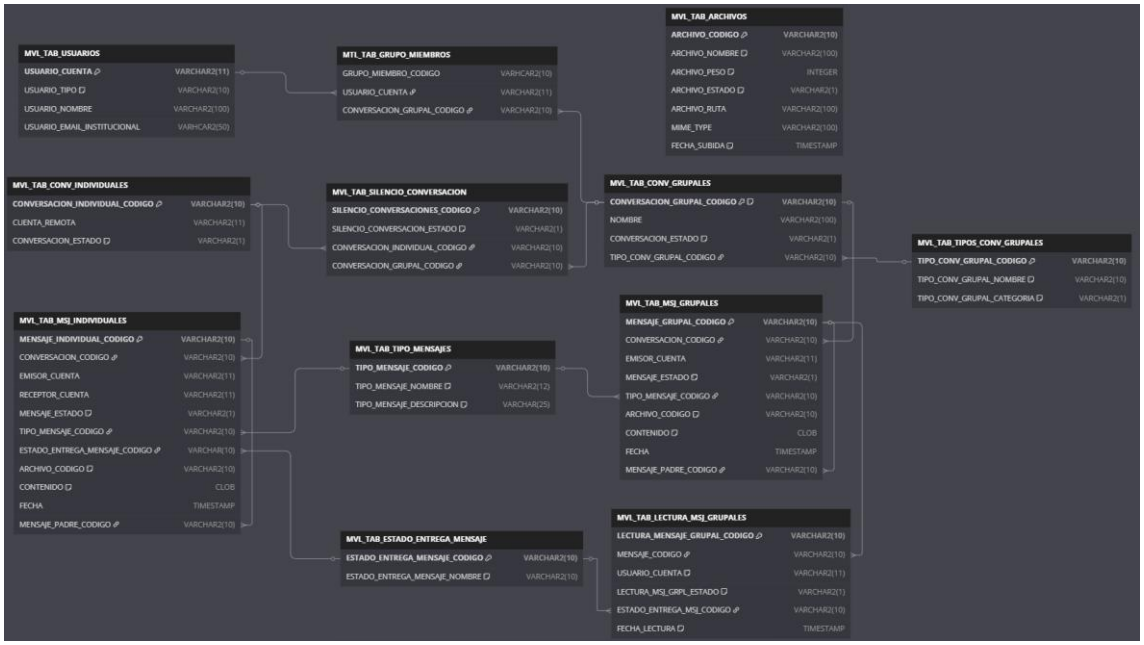



*Figura 26. Modelo entidad-relación de base de datos local*

Con la base de datos ya diagramada, inicialmente se había acordado utilizar SQLite (un sistema de gestión de bases de datos relacional liviano, ampliamente utilizado en dispositivos móviles por su simplicidad y su integración directa con el sistema de archivos local). Para facilitar esta integración en Flutter, se optó por usar primero Drift, que es una librería para flutter que permite trabajar con SQLite usando consultas SQL escritas de forma segura en Dart, con generación automática de código como DAOs y adaptadores [64]. Luego se intentó implementar directamente con sqlite3, que es la

librería nativa para manipular SQLite sin herramientas de abstracción, lo que exige mayor configuración manual [65].

Se buscaba implementar una funcionalidad representativa que permitiera validar la integración y el rendimiento de la base de datos local. Esta funcionalidad consistía en obtener las conversaciones individuales desde el backend, junto con sus respectivos mensajes y usuarios, con el fin de mostrar información como el nombre, tipo (estudiante o docente) y correo electrónico del interlocutor.

Para este punto, los endpoints del backend ya estaban funcionales, por lo que la tarea consistía en desarrollar los DAOs necesarios para que las conversaciones individuales se presentaran ordenadas según la fecha del último mensaje (de modo que los chats más recientes aparecieran primero). Además, se debía incluir un contador visual con la cantidad de mensajes no leídos por el usuario, una funcionalidad compleja que ponía a prueba la eficiencia del diseño local.

Esta implementación inicial requirió cerca de dos semanas de trabajo, principalmente debido a la complejidad de manejar múltiples DAOs, entidades, adaptadores y archivos de configuración. Según los principios de la arquitectura limpia, se definieron Entities (estructuras de datos utilizadas a nivel de dominio o capa de presentación) y Models (estructuras utilizadas para mapear la información desde o hacia una fuente de datos). Sin embargo, se evidenció la necesidad de usar un nuevo tipo de estructura denominada ViewEntity (una entidad enriquecida con datos combinados de varias fuentes, lista para mostrarse directamente en la UI), ya que los datos requeridos no provenían directamente de una tabla, sino de varias combinaciones y transformaciones.

Al intentar mapear los datos del backend, pasando primero por guardar los datos en la base de datos local y luego desde allí mapearlos a estas ViewEntities, surgieron múltiples errores relacionados con el tipado de datos, incompatibilidad entre los modelos y dificultades para manejar estructuras anidadas. Además, el uso de Drift o SQLite3 implicó una carga de configuración significativa, generando más de 70 archivos entre DAOs, entidades y adaptadores, migraciones y queries personalizadas.

Ante estas dificultades, se decidió investigar otras bases de datos locales populares en el ecosistema de Flutter, encontrando así Isar, la cual es una base de datos local NoSQL de alto rendimiento optimizada para Flutter y Dart, que permite trabajar con colecciones de objetos persistentes sin necesidad de escribir SQL, y con soporta para consultas reactivas, sincronización offline y relaciones entre colecciones [66].

Durante su investigación, se descubrió que el flujo de trabajo de Isar es significativamente más ágil: se define las colecciones (clases anotadas con @collection) y, mediante un comando de generación "*flutter pub run build_runner build*", se crean automáticamente los archivos necesarios que reemplazan a los DAOs. Isar proporciona métodos listos para usar como: *get()*, *put()*, *delete()*, *filter()*, *sortBy()*, etc.

Gracias a estas herramientas, el tiempo de desarrollo se redujo drásticamente. La misma funcionalidad descrita anteriormente, que con SQLite había tomado dos semanas, fue implementada en un solo día usando Isar.

Finalmente, únicamente fue necesario configurar los datasources locales con la lógica específica de inserción, actualización y lectura de datos. Además se añadió una capa de sincronización, que se encargue de obtener los nuevos mensajes o conversaciones que no se tengan registrados de forma local, esto gracias a las lecturas de mensajes, donde los mensajes que no hayan sido recibidos, ni leídos van a ser los que esté solicitando el usuario, una vez que los tenga de forma local, se emite la lectura que han sido recibidos, para que en una próxima solicitud no se encuentren en los datos que provienen del backend, así ahorrando de manera eficiente datos duplicados, y reducir llamadas innecesarias a los endpoints.

Esta experiencia permitió evidenciar claramente las ventajas de Isar en términos de facilidad de uso, velocidad de desarrollo, menor cantidad de configuración y mejor adaptabilidad al patrón arquitectónico utilizado en la aplicación.

En la Figura 27, se evidencia la forma de construir una colección de mensajes individuales que además tengan indexados datos o incluso haga uso de un índice compuesto lo que acelera las consultas.

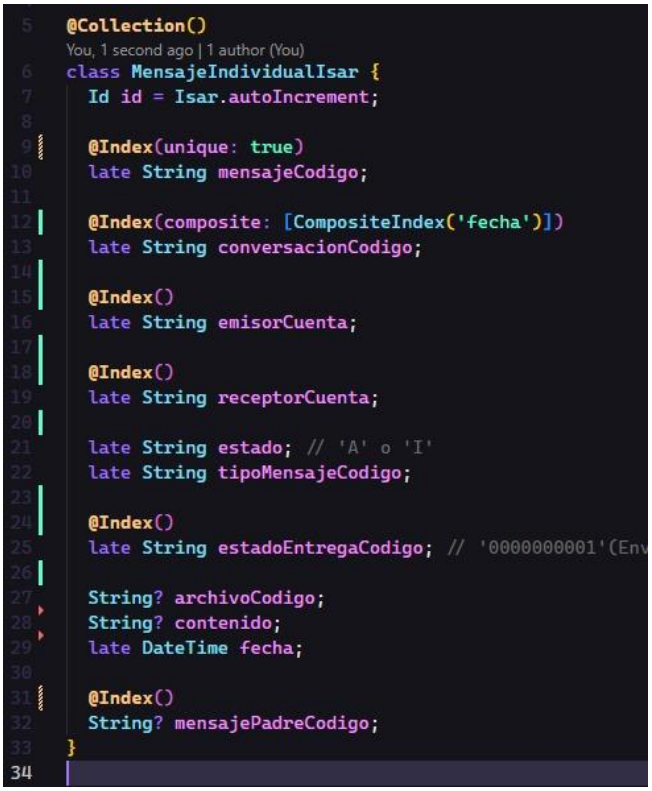



*Figura 27. Collection que representa la tabla de mensaje individual*

Para el diseño de las pantallas de mensajes individuales, se han configurado un *AppBar* en la parte superior que muestre el nombre y el correo electrónico del usuario con quien se mantiene la conversación, junto con acciones como regresar o acceder a la información detallada de la conversación. Tanto el *AppBar* como los mensajes propios utilizan el color primario institucional para mantener coherencia visual con la identidad de la aplicación, como se muestra en la Figura 28. Además, se integraron los widgets necesarios para mostrar, de forma clara y contextual, el contenido del mensaje al que se está respondiendo, en caso de tratarse de una respuesta.


*Figura 28. Vista de conversación individual con mensajes de respuesta - versión 1*

En esta sección, el diseño del ancho de los mensajes fue cuidadosamente ajustado para adaptarse de manera óptima al espacio disponible, incluso cuando se incluyen varios widgets anidados, como el contenedor del mensaje al que se está respondiendo. Para lograrlo, se implementó una estructura en la que el contenedor principal utiliza el ancho máximo permitido, pero mantiene flexibilidad para que sus elementos internos se adapten dinámicamente al contenido. De esta forma, cada subcomponente utiliza solo el espacio necesario, evitando desperdicio visual y optimizando la presentación de los mensajes. Este enfoque mejora la legibilidad y el equilibrio de la interfaz, como se observa en la Figura 29.


*Figura 29. Vista de conversación individual con mensajes de respuesta – versión 2*

En el diseño mostrado en la Figura 29, se visualizan los estados de entrega de los mensajes, inspirados en el modelo utilizado por WhatsApp. En el backend se definieron tres estados principales: enviado, recibido y leído. Sin embargo, en la aplicación también se presenta un cuarto estado denominado pendiente, el cual corresponde exclusivamente a la base de datos local del dispositivo. Este estado indica que el mensaje aún no ha sido enviado exitosamente al servidor.

En relación con lo anterior, se diseñó una arquitectura de eventos que modela lo que se ha denominado el "ciclo de vida de un mensaje", entendiendo que dicho ciclo culmina una vez que el mensaje alcanza el estado de leído. Esta arquitectura permite emitir eventos asociados a cada transición de estado y se divide en dos variantes: una para mensajes individuales y otra para mensajes grupales. La Figura 30 ilustra detalladamente esta estructura de eventos, destacando cómo se gestionan las transiciones desde el envío hasta la confirmación de lectura.

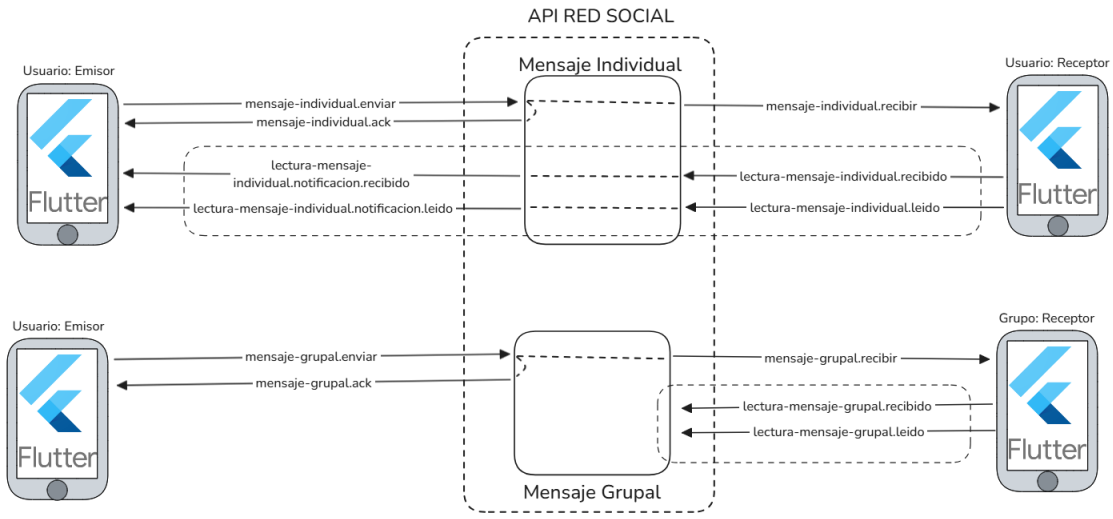



*Figura 30. Arquitectura de eventos para mensajes entre emisor y receptor*

Para implementar la comunicación en tiempo real, se utilizó el framework Socket.IO, una biblioteca que permite establecer conexiones WebSocket de manera sencilla y confiable tanto en el servidor como en el cliente. Socket.IO facilita el envío y recepción de eventos personalizados, permitiendo así construir sistemas interactivos como chats en tiempo real. En el lado del cliente se empleó socket.io-client, su contraparte para aplicaciones frontend [67].

Inicialmente, se intentó utilizar callbacks para que el cliente emitiera el evento de envío de mensaje y esperar una confirmación directa indicando que el mensaje fue correctamente procesado, lo cual permitiría cambiar su estado de pendiente a enviado. Sin embargo, se identificó que el cliente de Socket.IO no permite el uso directo de callbacks para todos los eventos personalizados en ciertas circunstancias. Antes esta limitación, se diseñó un nuevo evento llamado <*mensaje*>.*ack*, donde "ack" proviene de *acknowledgment* (reconocimiento en español). Este evento es emitido por el servidor y escuchado únicamente por el emisor del mensaje, proporcionando así una solución robusta para detectar cuándo un mensaje ha sido correctamente entregado y cambiar su estado de manera precisa.

En cuando al desarrollo de las historias de usuario relacionadas con la gestión de mensajes grupales (HU-07, HU-08), se adoptó un nuevo enfoque basado en la creación de un listener que permita al cliente unirse a múltiples roos (salas virtuales). Para ello, se

configuró el envío de un arreglo de strings (cadena de caracteres), donde cada string representa una sala a la que el usuario debe unirse. La Figura 31 muestra esta implementación. Gracias a esta estrategia, cuando se emite un mensaje a un grupo específico, todos los clientes conectados a la sala correspondiente recibirán el mensaje en tiempo real.

```
@Injectable()
export class RoomsListener {
    constructor() { }

    register(client: Socket) {
        client.on('rooms.join', async (payload: {rooms: string[]}) => {
            const user = client.data.user;
            console.log(`Usuario ${user.cuenta} se unió a las salas: ${payload.rooms.join(', ')}`);
            payload.rooms.forEach((room) => {
                client.join(room);
            });
        });
    }
}
```

*Figura 31. Configuración para unir a rooms*

Siguiendo la misma línea de diseño aplicada en la pantalla de conversación individual, se desarrolló la pantalla de conversación grupal y, como descripción secundaria, el ciclo académico actual. Se conservaron los colores institucionales y el estilo visual de los mensajes individuales, manteniendo así una experiencia de usuario coherente. No obstante, se introdujo una diferencia clave: en cada mensaje grupal, se muestra el nombre del emisor encima del contenido del mensaje, para facilitar la identificación de los participantes. La Figura 32 ilustra el diseño inicial propuesto para estas conversaciones grupales.


*Figura 32. Mockup de vista de conversacion grupal*

Se implementó los listeners necesarios para manejar los eventos definidos en la arquitectura de eventos para mensajes entre emisor y receptor, una vez que los usuarios se han unido correctamente a las rooms correspondientes. Gracias a esta configuración, fue posible replicar la misma funcionalidad de envío y recepción de mensajes ya implementada en las conversaciones individuales, ahora adaptada al contexto de los mensajes grupales. La Figura 33 muestra el diseño resultante de la vista de conversación grupal con esta funcionalidad aplicada.


*Figura 33. Implementación de mockup de conversación grupal*

**Revisión del Sprint 2**

Durante la retroalimentación correspondiente al Sprint 2, se recibieron las observaciones por parte del director del trabajo de grado, principalmente enfocadas en aspectos visuales y de funcionalidad pendientes entre las recomendaciones se surgió ajustar la paleta de colores, ya que el color primario utilizado resultaba visualmente muy saturado al aplicarse de forma predominante en la interfaz. Además, se señaló la necesidad de completar la implementación de las funcionalidades de envío de imágenes y archivos como mensajes, las cuales aún no estaban disponibles en ese momento.

Finalmente, se propuso un cambio sutil pero significativo en el diseño: aumentar el ancho máximo de los mensajes, pasando del 70% al 85% del ancho de la pantalla, con el objetivo de mejorar el aprovechamiento del espacio y la legibilidad de los contenidos.

### 3.3.3 Sprint 3

**Planificación del Sprint 3**

**Objetivo del sprint:**

Desarrollar los primeros componentes con datos del sistema, estableciendo la arquitectura inicial y la lógica para los primeros servicios.

**Duración:**

El sprint se planificó para 4 semanas (del 02/05/2025 al 31/05/2025) donde se trabajará 30 horas por semana aproximadamente.

**Resultado:** Se definió el Sprint Backlog que se detalla a continuación.

*Tabla 25. Sprint Backlog - Sprint 3*

| Código Historia de Usuario | Título | Tarea | Horas |
|---|---|---|---|
| **HU-09** | Gestión de envío de archivos como mensajes | Investigar sobre subida de archivos al servidor | 4 |
| | | Implementar un procedimiento que guarde el archivo y devuelva el archivo código | 16 |
| | | Implementar un endpoint para la subida de archivos | 8 |
| | | Implementar un endpoint para la descarga de archivos | 4 |
| | | Implementar funcionalidad de envío de archivos como mensajes | 15 |
| | | Implementar funcionalidad de envío de imágenes como mensajes | 10 |
| **HU-05** | Gestión de visualización de mensajes | Implementar funcionalidad de visualización de imagen como mensaje | 12 |
| | | Implementar funcionalidad de visualización de pdf como mensaje | 15 |
| | | Implementar funcionalidad de visualización de archivos como mensajes | 20 |
| **N/A** | Diseño y paleta de colores | Cambiar la paleta de colores | 10 |
| | | **Total** | 114 |

Dado que se disponen de 30 horas por semana, el Sprint 3 contempla un total estimado de 120 horas distribuidas en las 4 semanas. Sin embargo, se ha reservado un

margen de 6 horas para cubrir posibles imprevistos o emergencias, por lo que el total efectivo planificado en el Sprint Backlog es de 114 horas.

**Ejecución del desarrollo**

En la ejecución del desarrollo surgió un bug donde los mensajes y los mensajes de respuesta son pendientes, es decir aún no se ha confirmado el envío, podría suceder un error en el backend, ya que se está intentando insertar un mensaje, cuyo "mensaje padre" aún no ha sido insertado, por lo que el foreign key no existe, se diseñó un sistema eficiente donde se combina el socket y la recursividad, para que en el envío de mensajes pendientes, si el mensaje pendiente enviado actualmente, fue respondido por otro mensaje pendiente, simplemente se actualizaría la foreign key para que no surta errores, e igualmente para informarle al receptor que se está respondiendo los mensajes. El ejemplo visual se puede observar en la Figura 34.

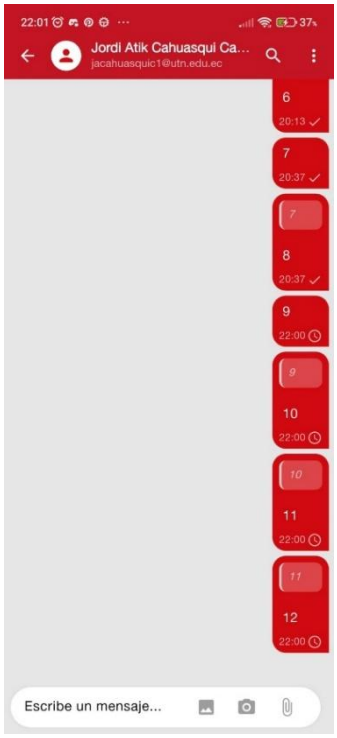



*Figura 34. Bug de mensajes de respuesta seguidos en modo offline*

Para el envío de archivos o mensajes, se emplea una estrategia común. Inicialmente, el archivo se envía a través de un endpoint de la API REST, el cual responde con un código único que identifica dicho archivo en la base de datos. Este código se incluye en el *payload* del mensaje que será emitido, y el tipo de mensaje cambio según el contexto (texto, imagen o archivo).

De forma similar al proceso de subida, la descarga del archivo se realiza utilizando el código incluido en el *payload.* Una vez descargado, el archivo se almacena en una carpeta local, y se registra en la base de datos local la ruta exacta donde se ha guardado, lo que permite acceder fácilmente a él en el futuro. La Figura 35 (Arquitectura REST API para subir y descargar archivos) muestra cómo se utiliza un endpoint con los eventos de *request* y *response* para la transferencia de archivos, junto con el uso de WebSocket como canal bidireccional para el envío y recepción de mensajes.

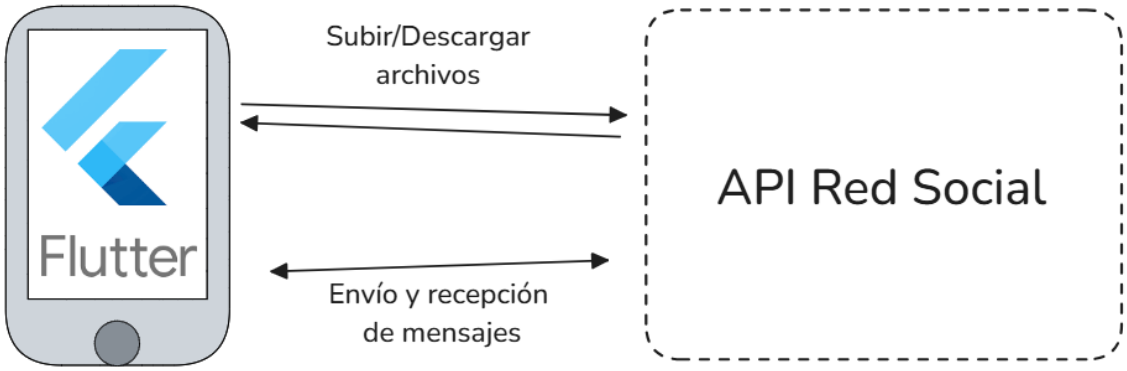



*Figura 35. Arquitectura híbrida para enviar, subir y descargar archivos*

Para implementar la funcionalidad de selección de imágenes desde el dispositivo o la toma de fotografías, se utilizó la librería *image_picker,* la cual permite acceder a la cámara o a la galería del dispositivo móvil para obtener imágenes de forma sencilla [68].

Para la selección de archivos de distintos tipos, se empleó la librería *file_picker,* que facilita la exploración del sistema de archivos del dispositivo y la selección de documentos, imágenes u otros archivos compatibles [68]. En este caso, se configuraron manualmente las extensiones permitidas con el objetivo de restringir el envío de videos u otros formatos no autorizados. A continuación, se presentan las Figuras: Figura 36, que ilustran las distintas funcionalidades desarrolladas.

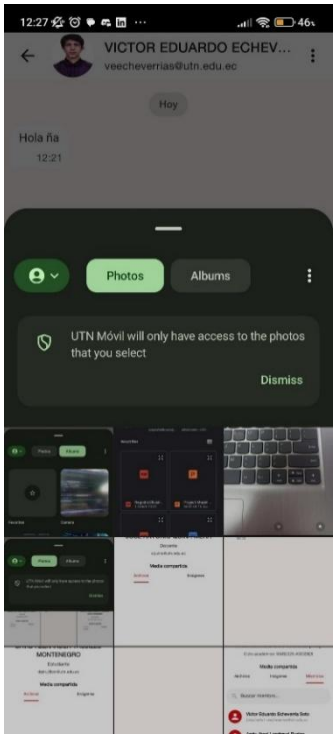



*Figura 36. Vista de selección de fotos desde la galería*

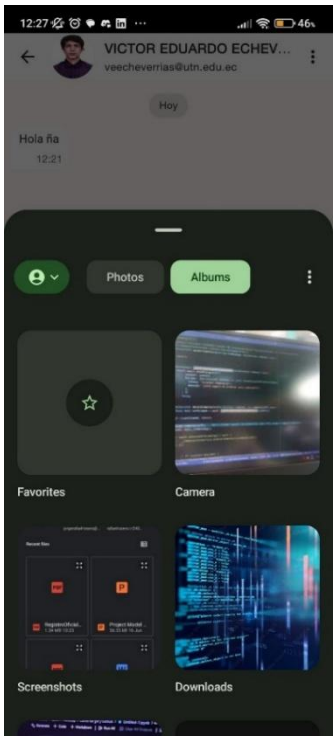



*Figura 37. Vista de selección de fotos desde un álbum*

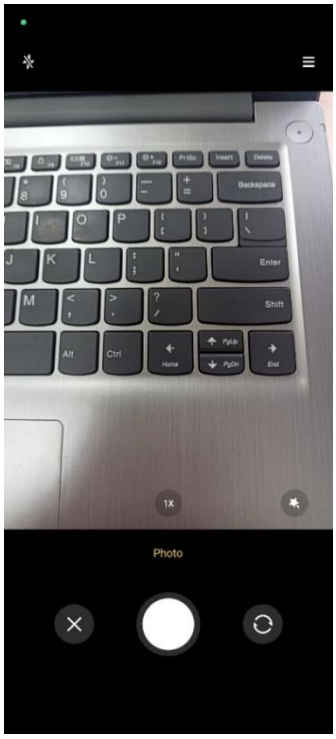



*Figura 38. Vista de tomar una fotografía desde la cámara*

La Librería *file_picker* permite adaptar qué extensiones están permitidas para seleccionar los archivos, entonces lo que se hizo fue programar y dividir en grupos

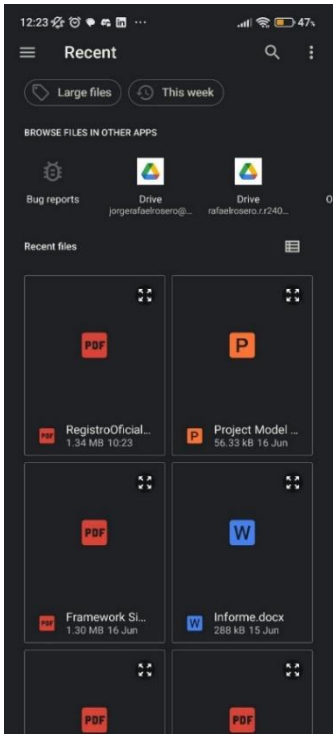



*Figura 39. Vista de selección de un archivo*

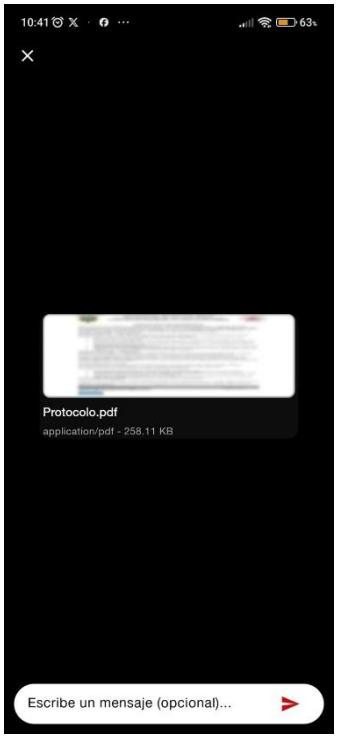



*Figura 40. Vista antes de enviar un pdf*

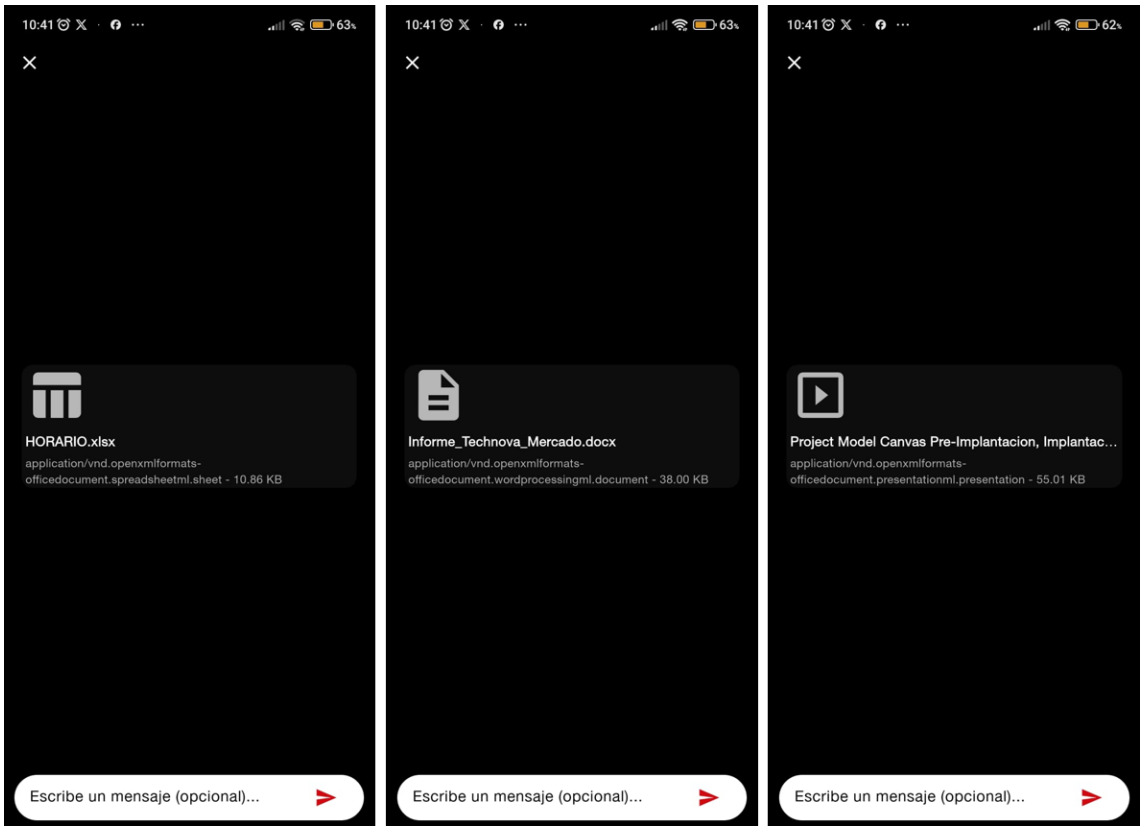



*Figura 41. Vista antes de enviar diferentes tipos de archivos*


*Figura 42. Vista antes de enviar una imagen*

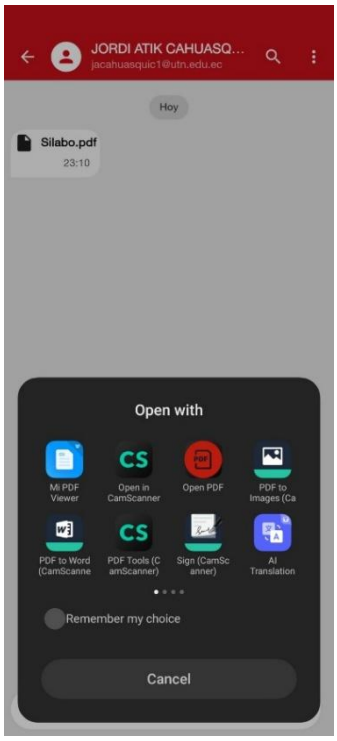



*Figura 43. Vista de archivo como mensaje para abrir – versión 1*

Para la visualización de mensajes que contienen archivos PDF, se implementó una estrategia específica: se genera una imagen de vista previa a partir de la primera página del PDF y esta se difumina. Además, con el fin de optimizar el rendimiento y evitar

problemas relacionados con el uso excesivo de memoria, esta imagen se almacena en caché local. La Figura 44 muestra cómo se representa visualmente este tipo de mensaje en la interfaz.


*Figura 44. Vista de archivo como mensaje ajeno – versión 2*

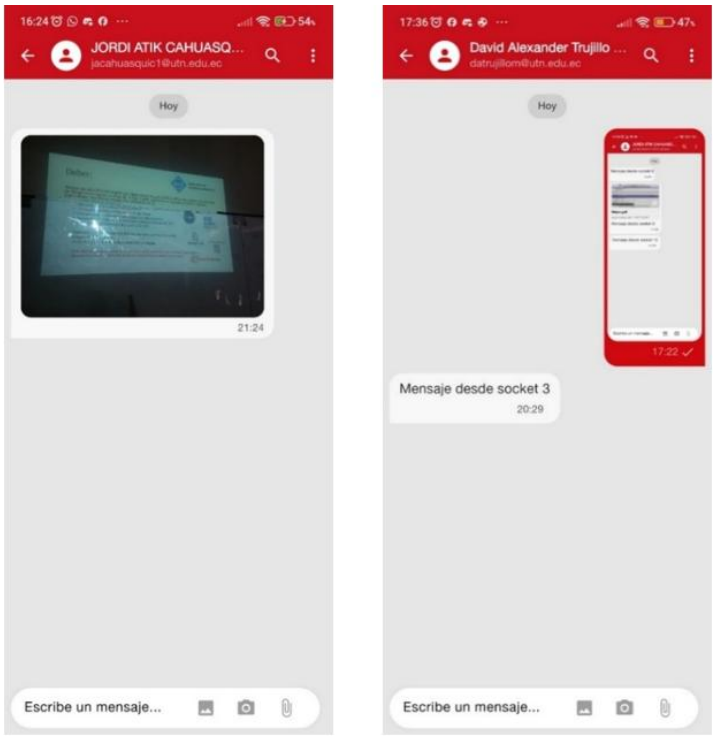



*Figura 45. Vista de Imagen como mensaje – versión 1*


*Figura 46. Vista de archivo como mensaje propio*

Como se mencionó en la revisión del sprint anterior, el color primario presentaba una saturación excesiva, lo que motivó un ajuste en el diseño visual. Para ello, se exploraron distintas combinaciones de colores que permitieran generar nuevas paletas más equilibradas y agradables a la vista. Las propuestas resultantes se presentan en la Figura 47, donde se comparan las distintas alternativas evaluadas.

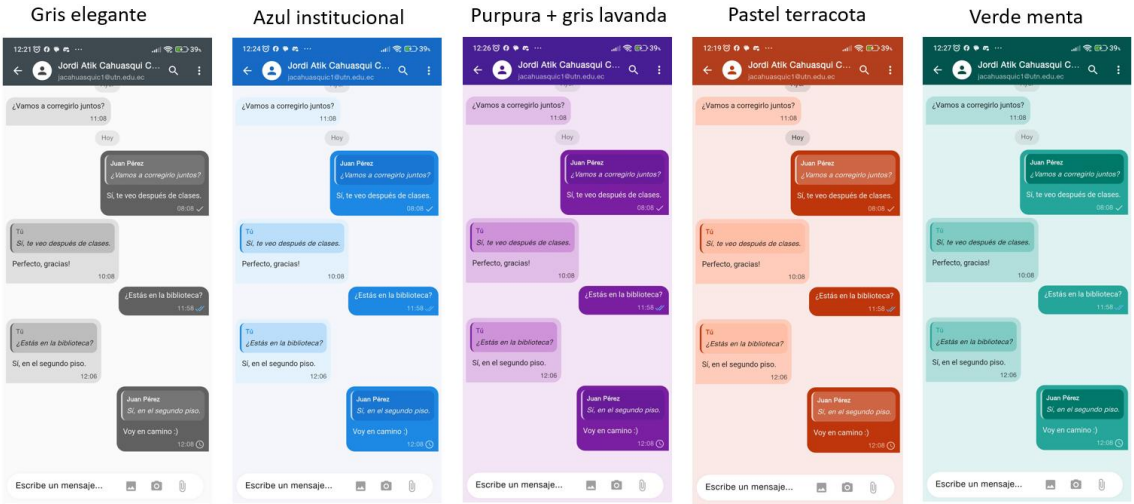



*Figura 47. Temas con diferentes paletas de colores – versión 1*

Durante el proceso de diseño, se identificó que uno de los elementos que generaba mayor saturación visual era el color utilizando en el *AppBar*. Por ello, se optó por reemplazarlo con un tono neutro y proponer nuevas paletas de colores para el resto de la interfaz. Este diseño es presentado en la Figura 48.

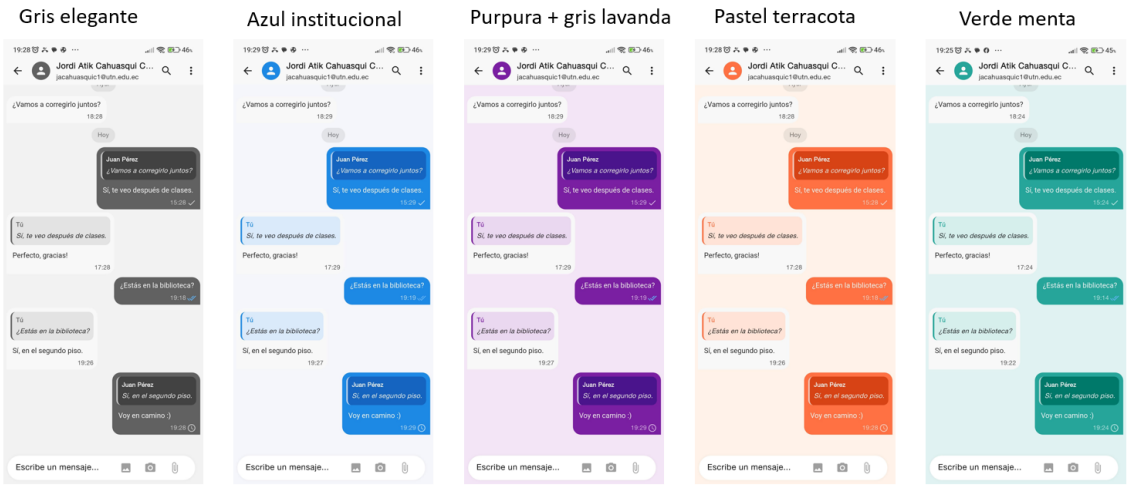



*Figura 48. Temas con diferentes paletas de colores – versión 2*

Sin embargo, tras evaluar las opciones presentadas, ninguna logró convencer completamente, por lo que se decidió conservar el color primario, aunque utilizándolo en menor proporción que antes.

Inspirado en la interfaz de WhatsApp, se incorporó este color únicamente en elementos puntuales, como los bordes de los mensajes respondidos. Adicionalmente, cuando el mensaje es propio, se añadió un borde en el lado derecho, con el fin de facilitar su identificación visual. La Figura 49 muestra el diseño final propuesto con estos ajustes.

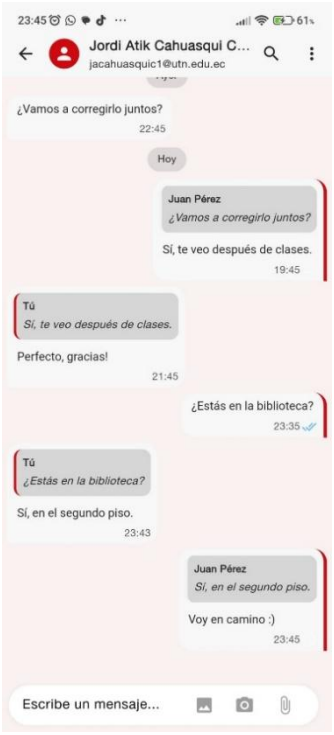



*Figura 49. Mockup con la paleta de colores seleccionado*

**Revisión del Sprint 3**

Durante la revisión del trabajo, se discutió que, si bien ya se contaba con las funcionalidades esenciales de la aplicación, aún eran necesarias algunas funcionalidades secundarias, pero igualmente importantes. Entre estas se identificaron: la incorporación de notificaciones, la posibilidad de visualizar los miembros de las conversaciones grupales, y la opción de ver fotografías en las conversaciones individuales. Además, se señaló la necesidad de implementar un sistema de filtrado en la sección *Institución,* especialmente para los docentes, ya que actualmente las conversaciones aparecen duplicadas. Este filtro permitiría separar las conversaciones exclusivas entre docentes de aquellas en las que también participan estudiantes.

Asimismo, se consideró relevante integrar el cifrado de mensajes, con el objetivo de reforzar la privacidad y seguridad de las comunicaciones dentro de la plataforma. Todos estos requerimientos fueron alineados con historias de usuario específicas, permitiendo planificar su desarrollo en sprints posteriores.

### 3.3.4 Sprint 4

**Planificación del Sprint 4**

**Objetivo del sprint:**

Desarrollar los primeros componentes con datos del sistema, estableciendo la arquitectura inicial y la lógica para los primeros servicios.

**Duración:**

El sprint se planificó para 4 semanas (del 02/06/2025 al 31/06/2025) donde se trabajará 30 horas por semana aproximadamente.

**Resultado:** Se definió el Sprint Backlog que se detalla a continuación.

*Tabla 26. Sprint Backlog - Sprint 4*

| Código Historia de Usuario | Título | Tarea | Horas |
|---|---|---|---|
| **HU-14** | Visualización de miembros de una conversación | Implementar visualización de miembros en grupos de asignaturas (rol estudiante) | 6 |
| | | Implementar visualización de miembros en grupos de asignaturas, carrera y facultad (rol docente) | 6 |
| **HU-01** | Gestión de conversaciones de carrera y facultad | Implementar visualización de fotografía de perfil en lista de conversaciones individuales | 5 |
| | | Implementar visualización de fotografía de perfil en la conversación individual | 5 |
| | | Implementar badges de notificación sobre íconos de navegación inferior | 5 |
| | | Agregar botón flotante para búsqueda y conectar con delegate de búsqueda existente | 5 |
| **HU-11** | Gestión de búsqueda de usuarios | Separar la lógica de búsqueda: búsqueda local en barra superior y búsqueda global en botón flotante | 8 |
| **N/A** | Conversaciones docentes (exclusivas) | Implementar filtro en vista de institución: todas, generales, solo docentes. | 8 |
| **HU-15** | Notificación individual | Implementar lógica para envío de notificaciones FCM de mensajes individuales | 10 |
| **HU-16** | Notificación grupal | Implementar lógica para envío de notificaciones FCM de mensajes grupales | 10 |

| | | | |
|---|---|---|---|
| **HU-15, HU-16** | Notificación individual y grupal | Implementar la cancelación de suscripciones automáticas de tokens y topics al cerrar sesión | 6 |
| **HU-13** | Gestión de la aplicación | Crear tabla de configuración para ciclos académicos y estado de la aplicación | 4 |
| | | Implementar formulario de APEX para ciclo activo, modo mantenimiento y mensaje a usuarios | 8 |
| | | Implementar validación del estado de la aplicación y versión BDD local desde frontend móvil | 5 |
| | | **Total** | 91 |

Dado que se disponen de 30 horas por semana, el Sprint 4 contempla un total estimado de 120 horas distribuidas en las 4 semanas. Sin embargo, para este sprint al realizarse antes de la evaluación se ha reservado un total de 29 horas para validar funcionalidades integradas, realizar pruebas de integración con el backend, y preparar la documentación para la evaluación de la aplicación.

**Ejecución del desarrollo**

Se desarrolló la funcionalidad para visualizar los miembros de una conversación grupal, permitiendo que el estudiante únicamente pueda ver los integrantes de los grupos correspondientes al nivel de asignaturas, en los que está inscrito. Esta restricción garantiza que los estudiantes no accedan a conversaciones o miembros de niveles organizativos superiores. La Figura 51 ilustra su funcionalidad.

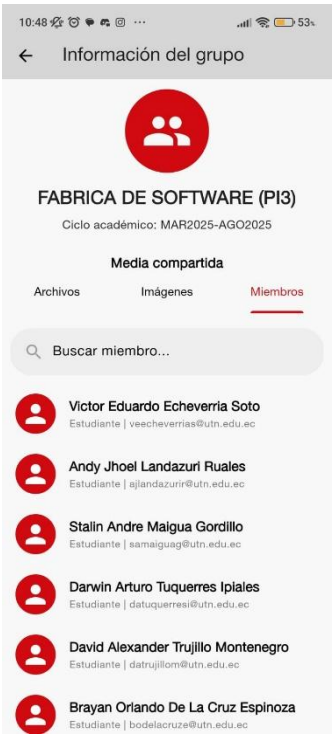



*Figura 50. Vista de miembros de conversación grupal de asignatura*

Por su parte, el docente tiene la posibilidad de ver los miembros de todas las conversaciones grupales en las que participa, abarcando los niveles de asignatura, carrera y facultad. Esto se alinea con el rol institucional del docente. La Figura 51 representa esta funcionalidad extendida.

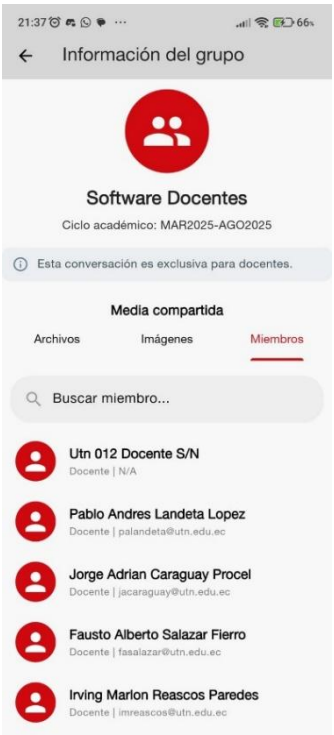



*Figura 51. Vista de miembros de conversación grupal de docentes*

Adicionalmente, se implementó la opción de visualizar la fotografía de perfil de los usuarios con los que se mantiene una conversación individual, Esta funcionalidad mejora la experiencia del usuario, ya que permite identificar fácilmente a la persona con la que se está interactuando. La Figura 52 muestra esta mejora aplicada a la interfaz.

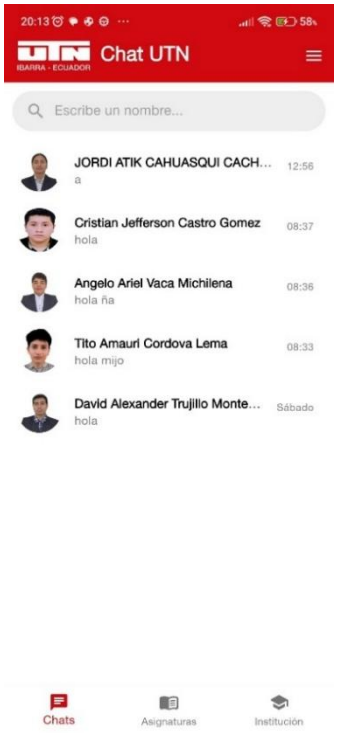



*Figura 52. Vista de chats individuales con foto de perfil*

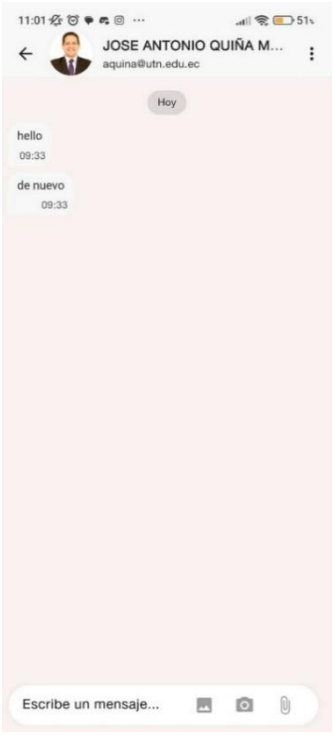



*Figura 53. Vista de conversación individual con foto de perfil*


*Figura 54. Vista de información del contacto con foto de perfil*

Por último, se integró la funcionalidad de *badges:* pequeños indicadores visuales en forma de burbujas que se superponen a los íconos de la barra de navegación inferior (*bottom Navigation Bar*), Estos *badges* aparecen automáticamente cuando existen

mensajes sin leer en cualquiera de las secciones, lo que permite al usuario identificar rápidamente nuevas notificaciones sin necesidad de ingresar a cada vista. La Figura 55 evidencia cómo se representa visualmente esta característica.


*Figura 55. Vista de conversaciones individuales con badges*

En la misma Figura 55, se observa también la incorporación de un *Floating Action Button* (botón flotante de acción rápida) en la parte inferior derecha. Este botón mantiene su funcionalidad previa: desplegar el *delegate* para la búsqueda de nuevos usuarios. Además, se modificó la barra de búsqueda superior para que ahora filtre únicamente por nombre entre las conversaciones ya cargadas en la vista. La búsqueda de usuarios nuevos se gestiona exclusivamente desde el botón flotante ubicado abajo a la derecha.

El docente mencionó que le es difícil diferenciar entre conversaciones generales y conversaciones en las que solo existan docentes, entonces se agregó una especie de filtro que ayude a filtrar o a tener pestañas en las que sólo se muestren conversaciones de un tipo. Las Figuras: Figura 56, Figura 57, Figura 58 muestran los tres tipos de filtros implementados.


*Figura 56. Vista de Institución - Todas*

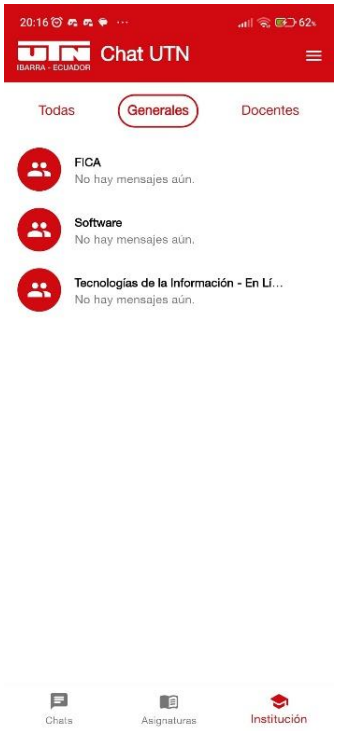



*Figura 57. Vista de Institución - Generales*

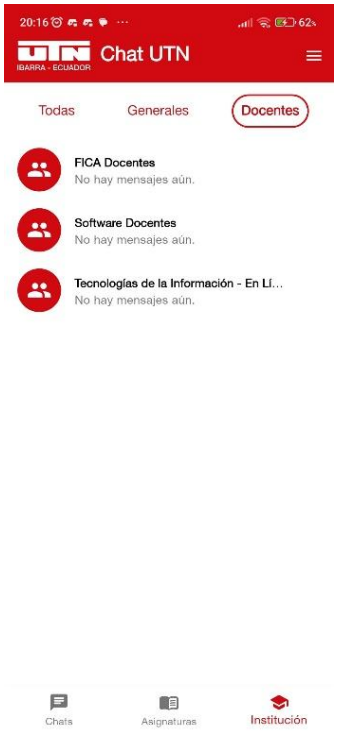



*Figura 58. Vista de institución – Docentes - versión 1*

Después conversar con el director del trabajo de grado, se sugirió cambiar el texto y el orden de los filtros, ya que "Generales" resulta un término demasiado abstracto y poco descriptivo para la funcionalidad. En su lugar, se decidió utilizar "Estudiantes y Docentes", porque transmite de manera más clara el alcance del filtro. Además, no se optó por "Docentes y Estudiantes", ya que ambos términos iniciarían con "Docentes" y no existiría una diferencia perceptible para el usuario. Tras este análisis, el diseño final puede observarse en la Figura 59.

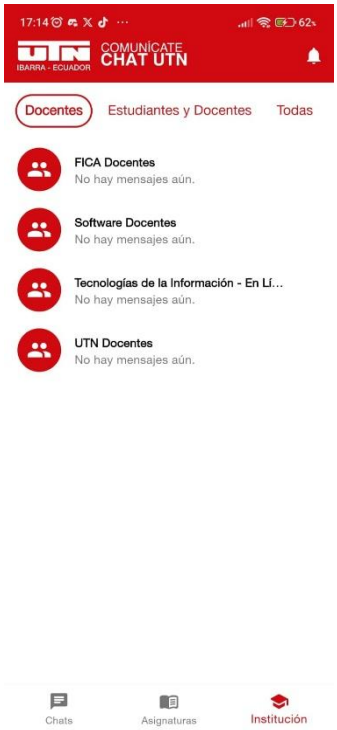



*Figura 59. Vista de institución - Docentes - versión 2*

La implementación del sistema de notificaciones se basó en una referencia tomada de un trabajo de titulación, elaborado por otro miembro del equipo cuyo enfoque principal era precisamente el manejo de notificaciones. A partir de dicho trabajo, se reutilizó la hoja de cálculo en Excel con la configuración inicial del servicio, lo que sirvió como base para estructurar el servicio en NestJs.

Para el caso de los mensajes individuales, se realiza una consulta a la base de datos que permiten obtener el token FCM (Firebase Cloud Messaging). Este token es un identificador único generado por Firebase que permite enviar notificaciones directamente a un dispositivo específico. Una vez recuperado el token, se procede a descifrar el contenido del mensaje, ya que los mensajes están cifrados para proteger la privacidad del usuario, y con esa información se construye el cuerpo de la notificación.

La construcción del mensaje de notificación se realiza utilizando un patrón de diseño *Factory*, que permite generar múltiples variantes de notificaciones de forma flexible. Gracias a esta estructura, fue posible implementar hasta 10 tipos diferentes de notificaciones, diferenciando entre mensajes individuales y grupales, así como otros posibles escenarios de uso.

En el caso de las notificaciones grupales, se reutiliza la lógica que permite unir a los usuarios a las conversaciones grupales, ya sea por asignatura, carrera o facultad. Esta misma lógica se aplica para suscribirse dinámicamente a los *topics* de *Firebase,* que funcionan como canales temáticos: al unirse a un topic, todos los usuarios suscritos recibirán la misma notificación. Esta estrategia permite enviar una única notificación que llegue a múltiples dispositivos de forma eficiente. Asimismo, se implementó la funcionalidad de cancelación de suscripción automática: al cerrar sesión o eliminar datos locales, el usuario también se desvincula de los topics de Firebase, evitando así recibir notificaciones no deseadas.

La Figura 60 muestra visualmente el funcionamiento integrado en la aplicación.

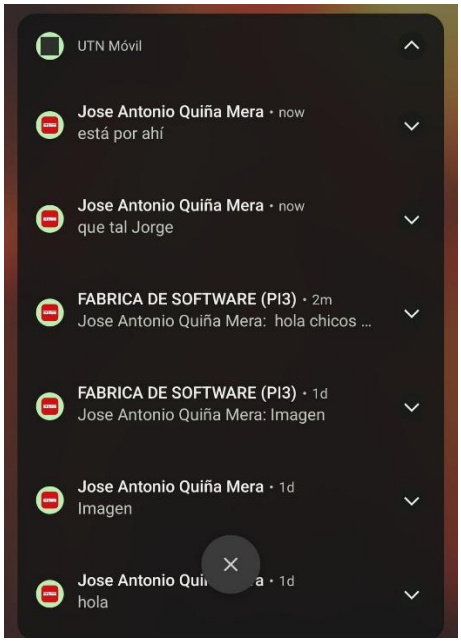



*Figura 60. Implementación de notificaciones*

Para finalizar, se implementaron funcionalidades que permiten interactuar con los mensajes de forma más completa: compartir contenido multimedia (archivos o imágenes), descargar imágenes a la galería, o, en el caso de mensajes de texto, copiarlos directamente al portapapeles. Estas acciones se habilitan al mantener presionado un mensaje, lo que resalta el contenido en pantalla y despliega un menú con las opciones disponibles de manera específica. Dichas funcionalidades se ilustran en la Figura 61.

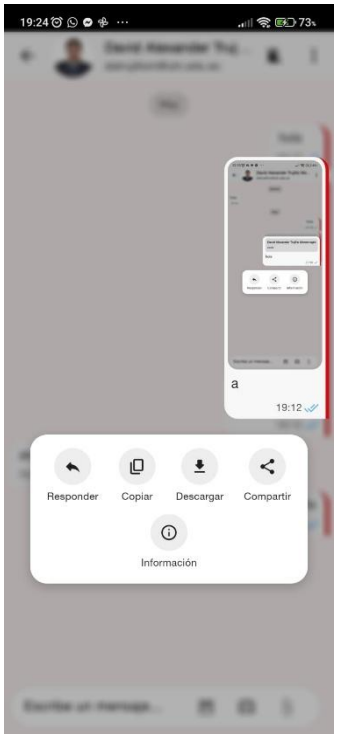



*Figura 61. Implementación de funcionalidades de mensajes*

**Panel de Administrador en APEX**

Para implementar la historia de usuario relacionada con el panel administrativo en APEX, detallada en la Tabla 18 *Gestión de la aplicación*, se creó una nueva tabla en la base de datos. Esta tabla permite:

- asignar el ciclo académico actual con el que trabajará la aplicación,
- poner la aplicación en modo mantenimiento,
- mostrar un mensaje en la pantalla principal,
- reiniciar de forma remota los datos internos del dispositivo en caso de problemas.

La estructura de esta tabla se presenta en la Figura 62.

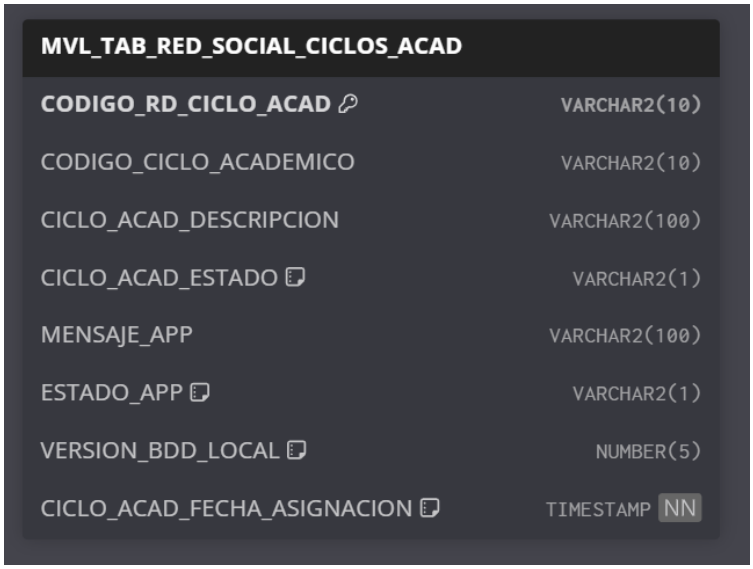



*Figura 62. Tabla Red Social Ciclos Académicos*

Cada columna cumple un rol específico:

- CODIGO_CICLO_ACADEMICO y CICLO_ACAD_DESCRIPCION: identifican y describen el ciclo académico actual asignado a la aplicación.
- CICLO_ACAD_ESTADO: define el estado del ciclo académico. La aplicación selecciona siempre el ciclo académico con estado activo (A) y más reciente según la columna CICLO_ACAD_FECHA_ASIGNACION.
- MENSAJE_APP: almacena el mensaje que se muestra al usuario cuando la aplicación está en estado inactivo (I). Si no se define el mensaje se presenta uno por defecto (ver Figura 63).
- ESTADO_APP: determina si la aplicación de la Red Social está en estado activo (A) o inactivo (I). Este valor no afecta al resto de módulos de la aplicación institucional UTN Móvil.
- VERSION_BDD_LOCAL: refleja la versión de la base de datos local en los dispositivos móviles. Permite reiniciar o eliminar datos de forma remota al modificar su valor, ya que la aplicación valida continuamente esta versión. Si no se asigna un número, el valor por defecto es 1.
- CICLO_ACAD_FECHA_ASIGNACION: registra la fecha en la que se insertó el ciclo académico correspondiente.

En la Figura 63 se observa la pantalla de la aplicación cuando es encuentra con estado inactivo, junto con el mensaje que se mostrará por defecto.


*Figura 63. Vista de aplicación en mantenimiento*

En la Figura 64 se muestra la vista de la aplicación cuando el registro académico actual tiene un valor definido en la columna de MENSAJE_APP. En este caso, dicho mensaje personalizado se presenta en pantalla al usuario.


*Figura 64. Vista de aplicación en mantenimiento con mensaje personalizado*

Para el desarrollo en Oracle APEX, se solicitó al DDTI el acceso con credenciales al entorno de desarrollo. A partir de allí, se procedió a implementar la lista de ciclos académicos y dos formularios asociados a la misma tabla descrita anteriormente, como se evidencia en la Figura 65.

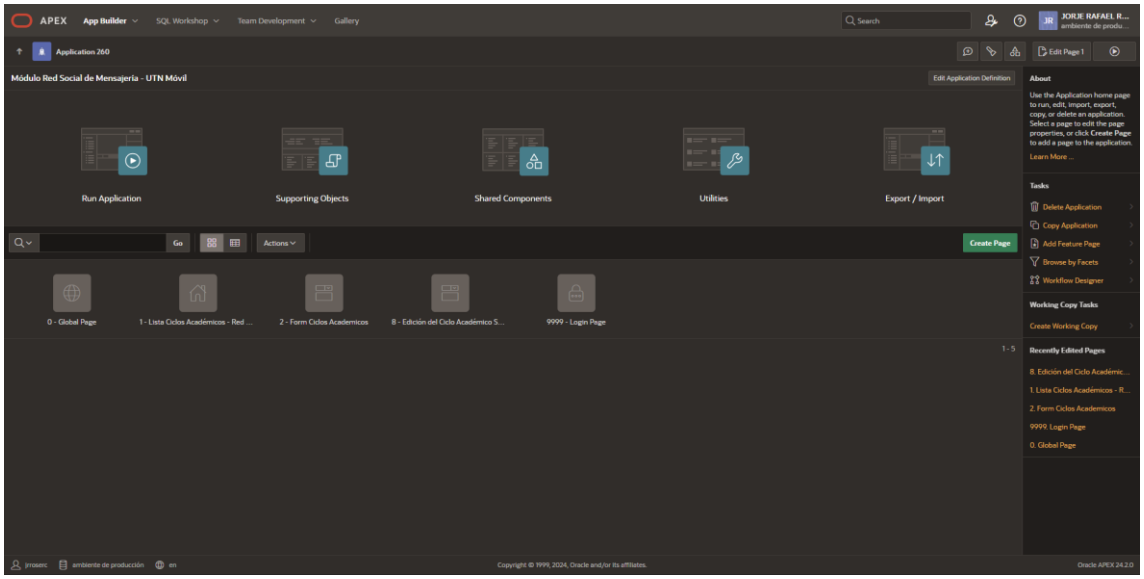



*Figura 65. Editor de páginas del módulo Red Social en APEX*

En esta sección se visualizan los registros correspondientes a la tabla mencionada, tal como se ilustra en la Figura 66. Desde aquí es posible crear un nuevo ciclo académico, seleccionar uno existente o editar cualquier registro.

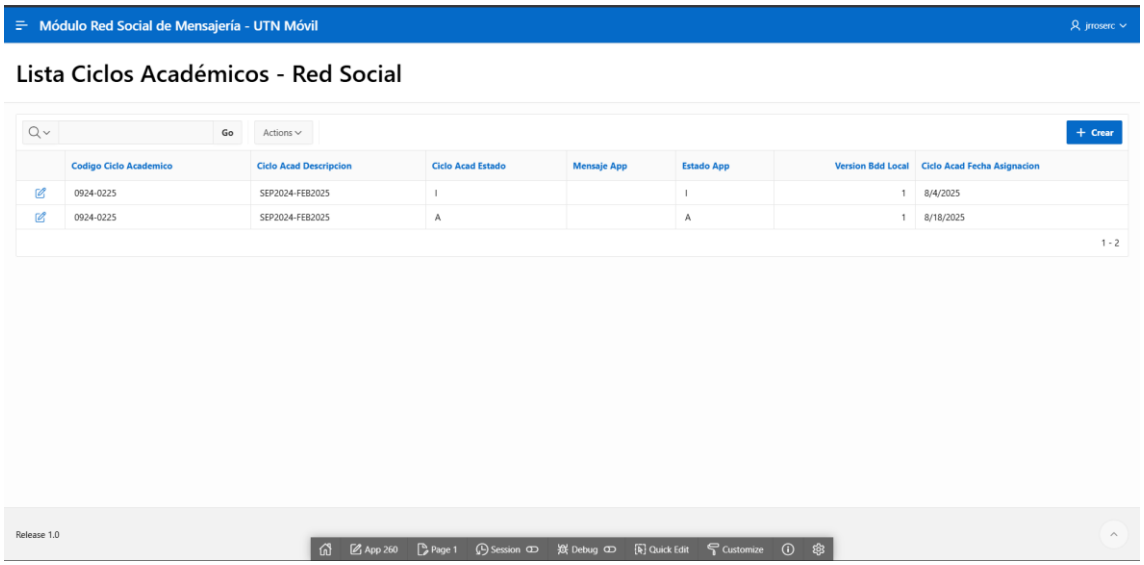



*Figura 66. Vista de ciclos académicos para la aplicación*

En el formulario de creación, se listan los últimos 10 ciclos académicos disponibles para permitir la selección del ciclo académico actual. Esta funcionalidad se implementó en respuesta a una observación realizada en la última reunión, donde se indicó que no todo debía dejarse de forma automática, especialmente en lo relativo a la gestión de ciclos académicos. Esta acción se refleja en la Figura 67.

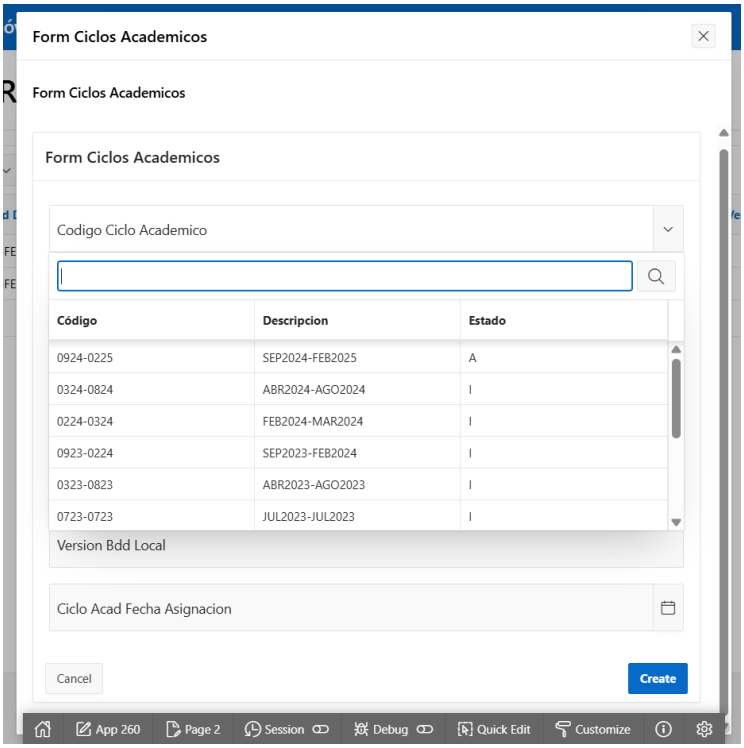



*Figura 67. Formulario para seleccionar un ciclo académico*

Una vez seleccionado un ciclo académico de la lista, el formulario se completa automáticamente con tres valores, como se observa en la Figura 68. A partir de ahí, se puede llenar las demás columnas manualmente.

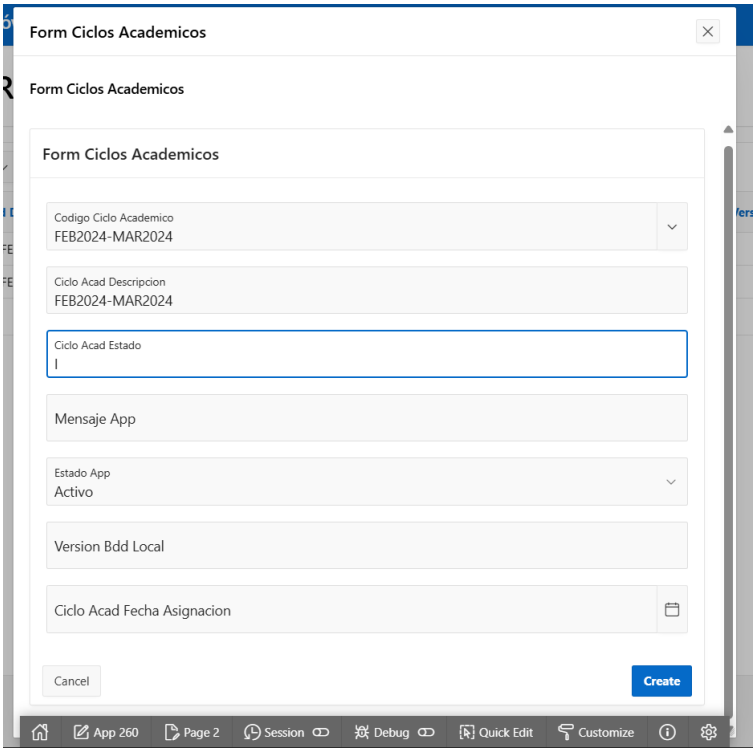



*Figura 68. Formulario para agregar un nuevo ciclo académico a la aplicación*

Finalmente, si no se ingresan valores en las últimas filas del formulario, estos se asignarán automáticamente con valores por defecto. Para editar cualquier registro, se puede hacer clic en el botón ubicado a la izquierda de cada fila, como se señala en la Figura 69.

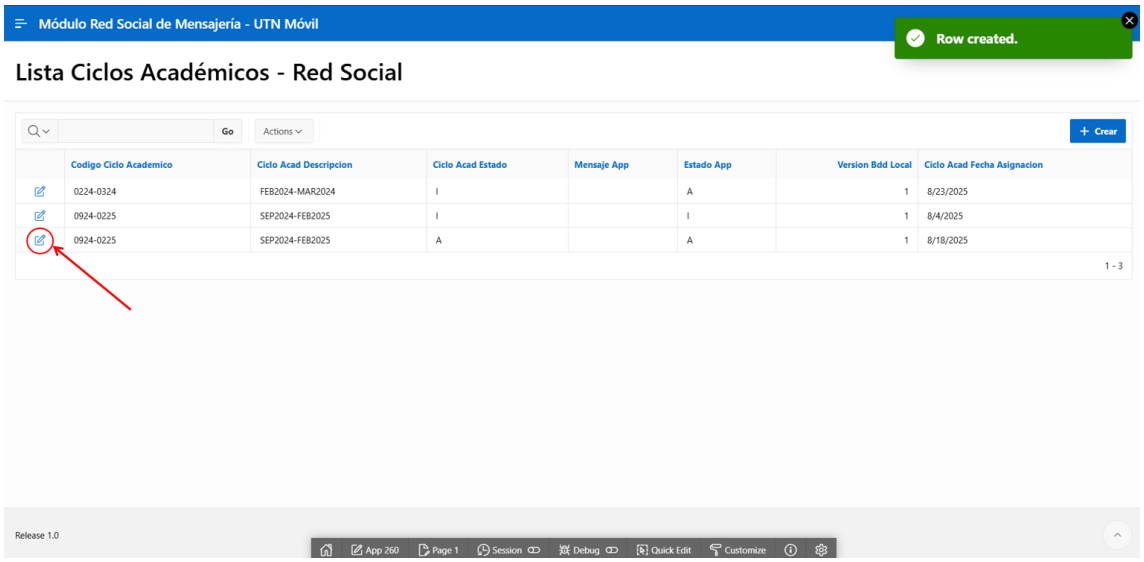



*Figura 69. Lista de ciclos académicos para editar*

Al hacerlo, se despliega un formulario modal que permite editar o eliminar el registro. No obstante, se recomienda evitar su eliminación definitiva y, en su lugar, cambiar su estado a inactivo (I), tal y como se muestra en la Figura 70.

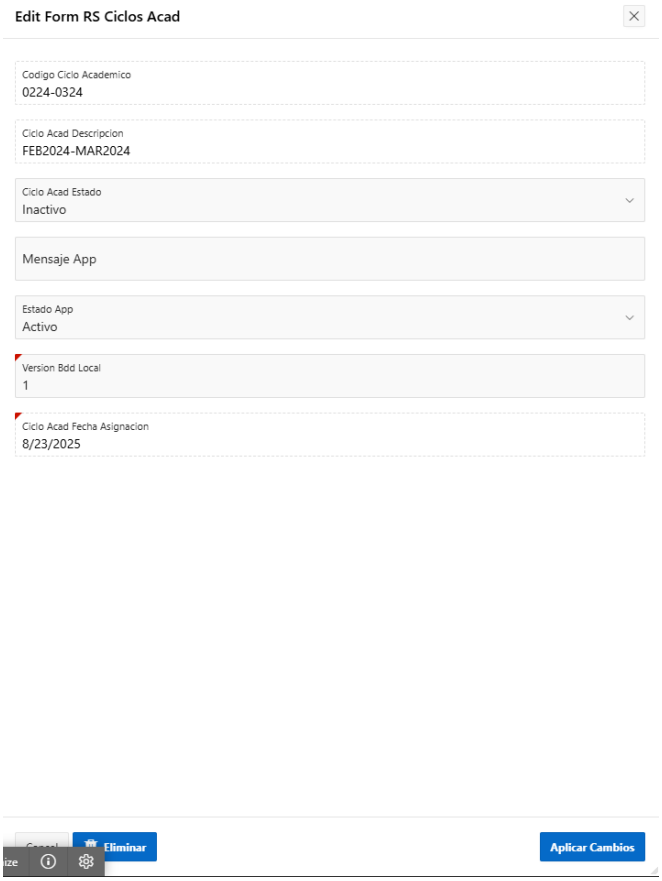



*Figura 70. Vista del formulario de edición de un ciclo académico*

Desde este formulario se pueden realizar los respectivos cambios descritos al inicio de la sección.

**Revisión del Sprint 4**

No se realizó una revisión formal con sugerencias de cambio durante este sprint. Por tanto, la versión desarrollada hasta ese momento fue considerada como la versión definitiva para efectos de la evaluación detallada de este documento, de acuerdo con los entregables establecidos en el cronograma del proyecto.

## 3.4 Validación y cierre del desarrollo

### 3.4.1 Pruebas de aceptación de historias de usuario

Las pruebas de aceptación se realizaron sobre cada historia de usuario definida en el Product Backlog detallado en la Tabla 22. Estas pruebas consistieron en verificar, junto con el Product Owner, que los criterios previamente establecidos se cumplieron en la versión final entregada al cierre de los sprints.

La Tabla 27 resume los resultados obtenidos:

*Tabla 27. Resultados de pruebas de aceptación*

| Código | Título | Resultado de prueba | Observación |
|---|---|---|---|
| **HU-02** | Gestión de conversaciones de asignaturas | Cumplida | Ninguna |
| **HU-03** | Gestión de conversaciones de carrera y facultad | Cumplida | Ninguna |
| **HU-01** | Gestión de conversaciones individuales | Cumplida | Ninguna |
| **HU-06** | Gestión de envío de mensajes individuales | Cumplida | Ninguna |
| **HU-07** | Gestión de envío de mensajes en las asignaturas | Cumplida | Ninguna |
| **HU-08** | Gestión de envío de mensajes en la carrera y facultad | Cumplida | Ninguna |
| **HU-05** | Gestión de visualización de mensajes | Cumplida | Ninguna |
| **HU-09** | Gestión de envío de archivos como mensajes | Cumplida | Ninguna |
| **HU-10** | Gestión de estados de entrega como mensajes | Cumplida | Ninguna |
| **HU-04** | Gestión de conversaciones de carrera y facultad de docentes | Cumplida | Ninguna |
| **HU-12** | Gestión de privacidad de mensajes | Cumplida | Ninguna |

| | | | |
|---|---|---|---|
| **HU-11** | Gestión de búsqueda de usuarios | Cumplida | Ninguna |
| **HU-13** | Gestión de la aplicación | Cumplida | Ninguna |
| **HU-14** | Visualización de miembros de una conversación | Cumplida | Ninguna |
| **HU-15** | Notificación individual | Cumplida | Ninguna |
| **HU-16** | Notificación grupal | Cumplida | Ninguna |

### 3.4.2 Reunión de revisión y ajustes adicionales

Se llevó a cabo una reunión de revisión con el objetivo de verificar si la aplicación cumplía con los requisitos establecidos por el DDTI. Durante esta sesión se identificaron algunas funcionalidades relevantes que no habían sido contempladas inicialmente, pero que resultan esenciales para el correcto funcionamiento del sistema.

La primera de ellas corresponde a la eliminación de mensajes, tanto en conversaciones individuales como grupales, ya que constituye una característica indispensable para otorgar mayor control a los usuarios sobre su comunicación.

La segunda observación se relaciona con la gestión de asignaturas: en el caso de que un estudiante anule la matrícula de alguna materia, ésta ya no debería mostrar en la vista de *Asignaturas* ni en las conversaciones correspondientes almacenadas en el dispositivo móvil.

A partir de estos hallazgos, se definieron cuatro nuevas historias de usuario, las cuales se presentan en la siguiente sección. Cabe destacar que estas incorporaciones se realizaron después de la evaluación general descrita en este documento.

### 3.4.3 Incorporación de nuevas historias de usuario

Como resultado de la reunión de revisión, se procedió a la incorporación de nuevas historias de usuario que reflejan los ajustes mencionados. Estas historias complementan el Product Backlog original y aseguran que la aplicación cumpla con las expectativas funcionales y de usabilidad planteadas por los usuarios finales.

*Tabla 28. Historia de Usuario 17*

| **Código: HU-17** | **Título: Eliminación de mensaje individual** | **Prioridad: Alta** |
|---|---|---|
| **Usuario: Docente/Estudiante** | **Dependencia:** HU-06 | Estimación: 12 h |
| **Descripción:** | Como docente y estudiante quiero tener la posibilidad de eliminar un mensaje individual. | |
| **Pruebas de aceptación:** | Cuando se seleccione la opción de eliminar un mensaje, en la conversación debe mostrarse como "mensaje eliminado". | |
| | En caso de que el receptor no esté conectado al momento de la eliminación, al volver a conectarse también deberá visualizar dicho mensaje como "eliminado". | |

*Tabla 29. Historia de Usuario 18*

| **Código: HU-18** | **Título: Eliminación de mensaje grupal** | **Prioridad: Alta** |
|---|---|---|
| **Usuario: Docente/Estudiante** | **Dependencia:** HU-07, HU-08 | Estimación: 12 h |
| **Descripción:** | Como docente y estudiante quiero tener la posibilidad de eliminar un mensaje grupal. | |
| **Pruebas de aceptación:** | Cuando un usuario elimine un mensaje grupal, todos los miembros de la conversación deberán visualizarlo como "mensaje eliminado". | |
| | Si uno o varios miembros no se encuentran conectados al momento de la eliminación, al iniciar sesión posteriormente deberán visualizar asimismo dicho mensaje como "eliminado". | |

*Tabla 30. Historia de Usuario 19*

| **Código: HU-19** | **Título: Eliminación de conversaciones de asignaturas anuladas** | **Prioridad: Alta** |
|---|---|---|
| **Usuario: Estudiante** | **Dependencia:** HU-02, HU-03 | Estimación: 16 h |
| **Descripción:** | Como estudiante quiero que al anular matrícula de una asignatura esta desaparezca automáticamente de la vista de | |

| | |
|---|---|
| | *Asignaturas* y de las conversaciones locales, para evitar acceder a materias en las que ya no estoy matriculado |
| **Pruebas de aceptación:** | Si un estudiante anula la matrícula de una asignatura, esta no debe mostrarse en la lista de *Asignaturas.* |
| | Las conversaciones y mensajes asociados deben eliminarse de la base de datos local del dispositivo. |

*Tabla 31. Historia de Usuario 20*

| **Código: HU-20** | **Título: Visualización de badge de mensajes pendientes** | **Prioridad: Baja** |
|---|---|---|
| **Usuario: Docente/Estudiante** | **Dependencia:** HU-05 | Estimación: 12 h |
| **Descripción:** | Como docente y estudiante quiero visualizar en la pantalla principal un badge con el número de mensajes pendientes, para identificar de forma rápida si tengo nuevos mensajes que aún no he recibido. | |
| **Pruebas de aceptación:** | El badge debe mostrar el número exacto de mensajes pendientes. | |
| | Si la cantidad de mensajes supera los 99, el badge debe mostrar "+99". | |
| | Al entrar a la aplicación de Red Social se debe actualizar el badge ya los mensajes ya están recibidos. | |
| | Si no existen mensajes pendientes, el badge no debe mostrarse. | |

A continuación, se presenta el nuevo Product Backlog donde se incorporan las nuevas historias de usuario (HU-17 a HU-20), junto con las historias de usuario iniciales para tener el Product Backlog completo que se refleja en la Tabla 32.

*Tabla 32. Product Backlog Completo*

| Orden | Código | Título | Estimación (Horas) |
|---|---|---|---|
| **1** | HU-02 | Gestión de conversaciones de asignaturas | 36 |

| | | | |
|---|---|---|---|
| **2** | HU-03 | Gestión de conversaciones de carrera y facultad | 36 |
| **3** | HU-01 | Gestión de conversaciones individuales | 36 |
| **4** | HU-06 | Gestión de envío de mensajes individuales | 40 |
| **5** | HU-07 | Gestión de envío de mensajes en las asignaturas | 30 |
| **6** | HU-08 | Gestión de envío de mensajes en la carrera y facultad | 30 |
| **7** | HU-05 | Gestión de visualización de mensajes | 40 |
| **8** | HU-09 | Gestión de envío de archivos como mensajes | 50 |
| **9** | HU-10 | Gestión de estados de entrega como mensajes | 50 |
| **10** | HU-04 | Gestión de conversaciones de carrera y facultad de docentes | 24 |
| **11** | HU-12 | Gestión de privacidad de mensajes | 40 |
| **12** | HU-11 | Gestión de búsqueda de usuarios | 40 |
| **13** | HU-13 | Gestión de la aplicación | 32 |
| **14** | HU-14 | Visualización de miembros de una conversación | 20 |
| **15** | HU-15 | Notificación individual | 32 |
| **16** | HU-16 | Notificación grupal | 32 |
| **17** | HU-17 | Eliminación de mensaje individual | 12 |
| **18** | HU-18 | Eliminación de mensaje grupal | 12 |
| **19** | HU-19 | Eliminación de conversaciones de asignaturas anuladas | 16 |
| **20** | HU-20 | Visualización de badge de mensajes pendientes | 10 |

Posteriormente, se procedió a documentar el desarrollo de las historias de usuario incorporadas en esta etapa (HU-17 a HU-20). Para cada una de ellas se describió la definición, pruebas de aceptación y a continuación la forma en que fueron implementadas dentro de la aplicación.

Para la implementación de una nueva funcionalidad en el sistema de mensajería, como lo es la eliminación de mensajes, se diseñó primeramente la integración por sincronización, en lugar de implementar directamente los eventos por socket. Esto se decidió porque, tras haber desarrollado varios eventos previamente, se presentaron problemas de sincronización una vez implementados los eventos en tiempo real. En consecuencia, se concluyó que, para nuevas funcionalidades y con el fin de no afectar el

diseño que ya estaba en funcionamiento, primero debe implementarse la sincronización y posteriormente los eventos en tiempo real por socket.

Es decir, primero se debe garantizar que el emisor elimine el mensaje y que, al ingresar nuevamente, el receptor visualice dicho mensaje como “eliminado”.
Además, en lugar de agregar un nuevo estado de entrega a la lista existente (“Enviado”, “Recibido”, “Leído” y “Pendiente”), en caso de cambios locales en los dispositivos, se optó por añadir un nuevo tipo de mensaje.

De esta forma, ahora existen cuatro tipos distintos de mensajes, tal como se muestra en la Tabla 33.

*Tabla 33. Tipos de mensajes*

| Código | Tipo |
|---|---|
| **0000000001** | Texto |
| **0000000002** | Imagen |
| **0000000003** | Archivo |
| **0000000004** | Eliminado |

La lógica implementada consiste en actualizar el mensaje de su tipo original a “Eliminado”.

Una vez finalizada la sincronización correctamente, se procedió a implementar los nuevos eventos por socket, los cuales complementan la arquitectura de eventos ya definida (Figura 30). En la Figura 71 se visualiza la arquitectura de eventos entre emisor y receptor actualizada con los eventos para eliminar mensajes.

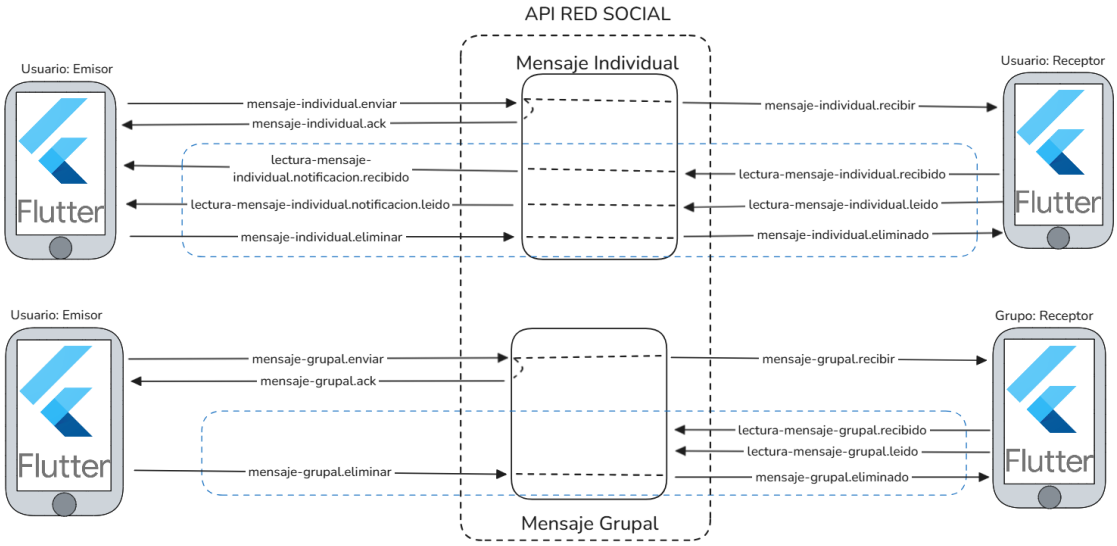



*Figura 71. Arquitectura de eventos entre emisor y receptor - versión 2*

Es importante señalar que los elementos dentro de los recuadros punteados azules son opcionales, y únicamente emisor y receptor pueden emitir estos eventos mediante socket al usar la aplicación. Además, únicamente los emisores de los mensajes tienen la opción de eliminarlos. En la Figura 72 se muestra la nueva opción disponible al mantener presionado un mensaje propio.

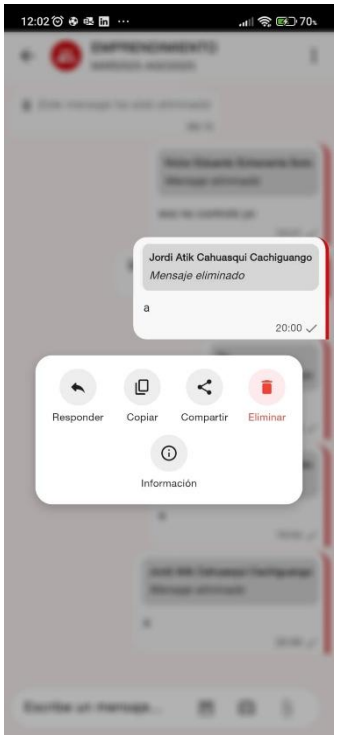



*Figura 72. Vista de un mensaje con la opción eliminar*

Cuando un mensaje cambia su tipo a “Eliminado”, se presenta con diseño mostrado en la Figura 73.

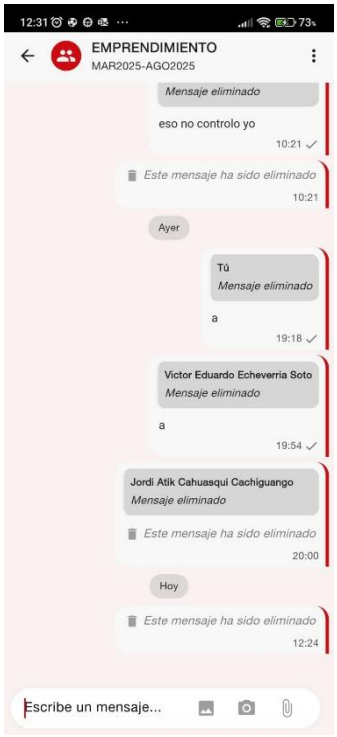



*Figura 73. Vista de conversación grupal con mensajes eliminados*

Cabe destacar que un mensaje eliminado no puede ser respondido, copiado ni compartido, ya que estas opciones estarán deshabilitadas para dicho tipo de mensajes.

En cuanto al desarrollo de la Historia de Usuario 19, referente a la eliminación de conversaciones de asignaturas anuladas, se implementó una comprobación que se ejecuta cada vez que el usuario ingresa a la página principal de la Red Social. En esta comprobación se consultan las asignaturas en las que el usuario está matriculado y se contrastan con las almacenadas localmente. En caso de existir alguna asignatura en la base de datos local que ya no figure en la matrícula activa, se procede a:

Cancelar la suscripción al *topic* de Firebase correspondiente, para evitar que se reciban notificaciones de dicha asignatura.
Eliminar todos los mensajes de texto, archivos e imágenes asociados a la conversación de esa asignatura.

La Figura 74 ilustra la lógica aplicada para eliminar la asignatura local en caso de que ya no exista en la base de datos o que el usuario haya anulado la matrícula.

```dart
Future<void> eliminarAsinaturaLocalEnCasoDeQueYaNoExistaEnBackend(List<ConversacionGrupalViewEntity> asignaturasFromBackend) async {
  final asignaturasFromLocal = await _asignaturasRepository.getAsignaturasFromLocal();
  final codigosFromBackend = asignaturasFromBackend.map((a) => a.conversacionGrupalCodigo).toSet();
  final codigosFromLocal = asignaturasFromLocal.map((a) => a.conversacionGrupalCodigo).toSet();
  final codigosToDelete = codigosFromLocal.difference(codigosFromBackend);

  if (codigosToDelete.isNotEmpty) {
    await _asignaturasRepository.eliminarAsignaturasLocalesByCodigos(codigosToDelete.toList());
  }
}
```

*Figura 74. Lógica para eliminar la asignatura local*

Respecto a la Historia de Usuario 20, detallada en la Tabla 31, correspondiente a la visualización de un badge con mensajes pendientes, se implementó un indicador en la pantalla principal que muestra el número de mensajes aún no recibidos, ya sean individuales o grupales. Para ello, se construyeron dos consultas en el backend:

- Una para obtener los mensajes pendientes individuales en los que el receptor es el usuario.
- Otra para obtener los mensajes pendientes en los grupos en los cuales el usuario se encuentra inscrito.

Finalmente, estos valores se muestran en la pantalla principal mediante un badge. La primera versión se muestra en la Figura 75 y la segunda versión, en la cual se suman los mensajes pendientes individuales y grupales en un solo badge, se presenta en la Figura 76.


*Figura 75. Vista del badge con mensajes pendientes - versión 1*

Tabla 31. Historia de Usuario 20


*Figura 76. Vista del badge con mensajes pendientes - versión 2*

# CAPÍTULO IV

# VALIDACIÓN DE RESULTADOS

## 4.1 Modelo de calidad en uso

### 4.1.1 Definición del modelo ISO/IEC 25022

El modelo ISO/IEC 25022 se emplea para evaluar la calidad en uso de un sistema, considerando tres características principales: Eficacia, Eficiencia y Satisfacción.

Cada característica se mide a través de un conjunto de métricas específicas, expresadas en distintas unidades y con pesos relativos que reflejan su importancia en el cálculo global [69].

En la Tabla 34 se presentan las características, métricas, unidades de medición y pesos adoptados en este estudio, basados en la estructura de la ISO/IEC 25022.

*Tabla 34. Modelo de calidad en uso*

| **Modelo de Calidad en Uso** | | | | |
|---|---|---|---|---|
| **Característica** | Métrica | Unidad / escala | Peso de métrica | Peso de Característica |
| **Eficacia** | Tareas completadas | Proporción 0-1 | 40 % | 40 % |
| | Objetivos logrados | % 0-100 | 30 % | |
| | Tareas con errores | Nro. Errores | 20 % | |
| | Intensidad de error | Proporción 0-1 | 10 % | |
| **Eficiencia** | Tiempo de tarea | Seg/min | 60 % | 30 % |
| | Eficiencia de tiempo | Obj./min | 40 % | |
| **Satisfacción** | Satisfacción global | Pts 0-100 | 40 % | 30 % |
| | Utilidad percibida | Likert 1-5 | 30 % | |
| | Confianza | Likert 1-5 / % | 20 % | |
| | Comodidad | Likert 1-5 / % | 10 % | |

### 4.1.2 Métricas evaluadas

A continuación, se detalla las funciones de medición de cada una de las métricas descritas en el modelo de calidad en uso.

**Eficacia**

Tareas completadas

X = A / B

- A = Número de tareas únicas completadas
- B = Número total de tareas únicas intentadas

Objetivos logrados

X = 1 - ∑Ai

- Ai = valor proporcional de cada objetivo omitido o incorrecto (0-1)

Tareas con errores

X = A / B

- A = Número de tareas con errores.
- B = Número total de tareas.

Intensidad de error

X = A / B

- A = número de usuarios que cometieron 1 error o más.

**Eficiencia**

Tiempo de tarea

X = T

- T = tiempo medio para completar la tarea

Eficiencia de tiempo

X = A / T

- A = número de objetivos logrados (o tareas completadas)
- T = tiempo total empleado

**Satisfacción**

Satisfacción global

Puntos del cuestionario SUS

- SUS = [(P1−1) + (5−P2) + (P3−1) + (5−P4) + (P5−1) + (5−P6) + (P7−1) + (5−P8) + (P9−1) + (5−P10)] × 2.5

Utilidad percibida

Promedio de las puntuaciones de los ítems del SUS relacionados con utilidad y valor percibido:

X = (P1 + P3 + P5) / 3

(ítems en escala de Likert de 1 a 5, ya invertidos si fueran negativos).

Confianza

Puntuación otorgada al ítem P9 ("Me siento seguro al usar el sistema") en la escala de Likert de 1 a 5.

X = A

- A = puntuación del cuestionario Likert (1 - 5) otorgado al ítem 9

Comodidad

Promedio de los ítems negativos relacionados con dificultad de uso, donde se invierte los pares antes de calcular el promedio.

X = (P2' + P4' + P6' + P8' + P10') / 5

- Pn' = valor invertido al ítem negativo (5=1, 4=2, 3=3, 2=4, 1=5).

### 4.1.3 Relación con los instrumentos utilizados

*Tabla 35. Relación de métricas con los instrumentos de recolección*

| Métrica | Se obtiene en | Cómo |
|---|---|---|
| **Tareas completadas, Objetivos logrados, Errores** | Taller práctico | Check-list, Excel |
| **Tiempo de tarea, Eficiencia de tiempo (productividad)** | Taller práctico | Registro de tiempo en Excel |

| | | |
|---|---|---|
| **Utilidad percibida** | Encuesta SUS | Ítems positivos: P1 (uso frecuente), P3 (facilidad de uso), P5 (facilidad de aprendizaje). Se promedia y se invierten ítems negativos si los hubiera. |
| **Confianza** | Encuesta SUS | P9 (seguridad al usar la aplicación) |
| **Comodidad** | Encuesta SUS | Ítems negativos invertidos: P2 (complejidad), P4 (necesidad de ayuda experta), P6 (inconsistencias), P8 (complicado), P10 (requiere conocimientos técnicos). |

## 4.2 Metodología de evaluación

### 4.2.1 Diseño del taller práctico

La muestra se obtuvo mediante un muestreo no probabilístico por conveniencia, seleccionando a 46 estudiantes de los últimos semestres de la carrera de Ingeniería en Software de la Universidad Técnica del Norte. La elección de este tipo de muestro responde a la dificultad de acceder a toda la población estudiantil debido a la carta académica y actividades propias de la etapa final de la carrera, lo que limita la disponibilidad de los participantes. Se priorizó a los estudiantes con mayor conocimiento técnico y experiencia académica, ya que su perfil resulta más pertinente para evaluar la propuesta planteada.

Escenario: el usuario debe tener la aplicación instalada, es decir el *app-release.apk* compartido, con la versión beta que se encuentra en desarrollo. Para el taller se tomará en cuenta que el usuario tiene una cuenta institucional y ya tiene iniciado sesión con las notificaciones activadas en la aplicación.

Objetivos y tareas: se diseñó la Tabla 36 que detalla los objetivos y el número de tareas que se debe desarrollar por cada objetivo.

*Tabla 36. Taller práctico para evaluar el módulo de la aplicación móvil*

| **Objetivo** | **Tarea** |
|---|---|

| | | |
|---|---|---|
| **1** | Enviar un mensaje de texto a una persona específica | En la vista de ***Chats*****,** buscar y seleccionar a una persona específica |
| | | Escribir y enviar un mensaje de texto (saludo, "Hola, ¿Cómo estás?", etc.) |
| **2** | Responder a un mensaje de texto | Seleccionar el mensaje a responder, deslizar hacía la derecha |
| | | Escribir y enviar el mensaje de respuesta |
| **3** | Enviar una imagen como mensaje | En la conversación abierta, presionar el ícono de fotografía.<br>Seleccionar la imagen y verificar en la vista previa de envío |
| | | Descartar la imagen en caso de que se seleccionó otra imagen por equivocación o redactar un mensaje opcional, finalmente enviar la imagen |
| **4** | Enviar un archivo adjunto como mensaje | En la conversación, presionar el ícono de archivo adjunto.<br>Seleccionar el pdf o archivo que se desee compartir y verificar los datos de archivo como el nombre, extensión o peso en la vista previa de envío |
| | | Descartar el archivo en caso de que se seleccionó otro archivo por equivocación o redactar un mensaje opcional, finalmente enviar el archivo |
| **5** | Enviar un mensaje grupal | Navegar a la vista de ***Asignaturas.*** En la conversación de la asignatura seleccionada, enviar un mensaje de texto (saludo, "Hola compañeros", etc.). |
| **6** | Silenciar notificaciones | Seleccionar y navegar a una conversación. En la parte superior derecha, en el ícono de menú (tres puntos verticales), seleccionar la opción ***Silenciar Notificaciones.*** |

Instrumentación: se diseñó una tabla en Excel que contienen los objetivos y tareas a cumplir con el tiempo de inicio y tiempo final para calcular el tiempo total por cada tarea.

### 4.2.2 Diseño de la encuesta SUS

La System Usability Scale (SUS) es un instrumento ampliamente utilizado para evaluar la usabilidad percibido de un sistema. Está compuesto por 10 ítems con formato de afirmaciones, que se responden mediante una escala de Likert de 5 puntos (1 = Totalmente en desacuerdo, 5 = Totalmente de acuerdo) [70].

En este estudio, los ítems de la encuesta se agrupan en tres factores de interés:

- Utilidad percibida: ítems P1, P3 y P5.
- Confianza: ítem P9.
- Comodidad: ítems P2, P4, P6, P8 y P10 (ítems negativos, por lo que se invierten en la escala previo al cálculo del promedio).

El cálculo del puntaje SUS global se realizó aplicando la fórmula establecida por el estándar:

$$SUS = [(P1-1) + (5-P2) + (P3-1) + (5-P4) + (P5-1) + (5-P6) + (P7-1) + (5-P8) + (P9-1) + (5-P10)] \times 2.5$$

El valor resultante se interpreta según el rango propuesto por Bangor, Kortum y Miller (2009), donde 68 puntos se considera el umbral de usabilidad aceptable. Valores superiores indican un mejor desempeño percibido [71].

En la Tabla 37 se muestran las preguntadas utilizadas en la encuesta SUS.

*Tabla 37. Encuesta SUS*

| Nro. | Pregunta |
|---|---|
| **P1** | ¿Considera que utilizaría esta aplicación con frecuencia? |
| **P2** | ¿Considera que la aplicación es compleja? |
| **P3** | ¿Considera que la aplicación es fácil de utilizar? |
| **P4** | ¿Considera que necesitaría apoyo de un experto para poder utilizar esta aplicación? |

| | |
|---|---|
| **P5** | ¿Considera que esta aplicación es fácil de aprender para la mayoría de las personas? |
| **P6** | ¿Considera que la aplicación presenta inconsistencias? |
| **P7** | ¿Considera que la mayoría de las personas aprenderían a utilizar este sistema rápidamente? |
| **P8** | ¿Encontré el sistema muy complicado de usar? |
| **P9** | ¿Se sintió seguro/a al utilizar la aplicación? |
| **P10** | ¿Considera que se requieren conocimientos técnicos previos para utilizar correctamente la aplicación? |

## 4.3 Fiabilidad de los instrumentos

### 4.3.1 Fiabilidad del taller práctico

En la Figura 77 se presentan los resultados obtenidos tras aplicar la prueba de normalidad de Shapiro Wilk a las variables "Errores", "Objetivo Logrados", "Tareas Logradas" y "Observaciones", con el fin de determinar el tipo de análisis estadístico para evaluar la fiabilidad del taller práctico.

Descriptives

| | Errores | Objetivos_logrados | Tareas_logradas | Observaciones |
|---|---|---|---|---|
| N | 46 | 46 | 46 | 46 |
| Missing | 0 | 0 | 0 | 0 |
| Mean | 0.304 | 6.00 | 10.0 | 0.413 |
| Median | 0.00 | 6.00 | 10.0 | 0.00 |
| Standard deviation | 0.553 | 0.00 | 0.00 | 0.580 |
| Minimum | 0 | 6 | 10 | 0 |
| Maximum | 2 | 6 | 10 | 2 |
| Shapiro-Wilk W | 0.586 | NaN | NaN | 0.673 |
| Shapiro-Wilk p | < .001 | NaN | NaN | < .001 |

*Figura 77. Pruebas de normalidad al taller práctico*

Los resultados evidencian que las variables "Errores" ($W=0.586$, $p<0.001$) y "Observaciones" ($W=0.673$, $p<0.001$) no siguen una distribución normal. Por otro lado, las variables "Objetivos Logrados" y "Tareas Logradas" presentaron valores constantes para todos los participantes (6 y 10 respectivamente). Esta ausencia de variabilidad imposibilita el cálculo de correlaciones para estas variables

Dado que las variables con variabilidad no cumplen el supuesto de normalidad, se procedió a emplear métodos estadísticos no paramétricos. Específicamente, se utilizó coeficiente de correlación de Spearman para examinar la relación entre las variables "Errores" y "Observaciones", cuyos resultados se muestran en la Figura 78.

Correlation Matrix

| | | Errores | Observaciones |
|---|---|---|---|
| Errores | Spearman's rho | — | |
| | df | — | |
| | p-value | — | |
| Observaciones | Spearman's rho | 0.855 | — |
| | df | 44 | — |
| | p-value | < .001 | — |

*Figura 78. Matriz de correlaciones del taller práctico*

Se encontró una correlación positiva muy fuerte y estadísticamente significativa (rho=0.855, p<0.001) entre las variables "Errores" y "Observaciones". Este hallazgo indica que a medida que aumenta el número de errores cometidos durante la ejecución del taller, también se incrementa la cantidad de observaciones registradas. Esta relación respalda la coherencia interna del taller práctico.

### 4.3.2 Fiabilidad de la encuesta SUS

Asimismo, con el objetivo de verificar la confiabilidad de la encuesta SUS, se aplicó el alfa de Cronbach a las respuestas recopiladas. Antes del cálculo, se invirtieron las puntuaciones de los ítems P2, P4, P6, P8 y P10, dado que estos presentan una formulación negativa en el cuestionario.

Scale Reliability Statistics

| | Cronbach's α |
|---|---|
| scale | 0.826 |

*Figura 79. Alfa de Cronbach a la encuesta SUS*

El resultado obtenido fue de 0.826, como se observa en Figura 79 , lo que indica una buena consistencia interna entre los ítems evaluados. Este valor confirma que el cuestionario SUS es un instrumento fiable para medir la percepción de usabilidad.

## 4.4 Resultados

### 4.4.1 Resultados del taller práctico

La Tabla 38 presenta el porcentaje de usuarios que completaron cada tarea correspondiente a los objetivos evaluados, así como el tiempo promedio empleado en su ejecución.

Las tareas han sido resumidas a partir de la información detallada en el **Anexo C**, el cual contiene la descripción completa de cada una de ellas.

*Tabla 38. Resultados del taller práctico*

| Nro. | Tarea | Usuarios que completaron (%) | Tiempo promedio (s) |
|---|---|---|---|
| **A1.1** | Buscar y seleccionar una persona específica | 100% | 23.22 |
| **A1.2** | Escribir y enviar mensaje de texto | 100% | 11.57 |
| **A2.1** | Seleccionar mensaje a responder, deslizar hacía la derecha | 100% | 7.76 |
| **A2.2** | Escribir y enviar mensaje de respuesta | 100% | 14.87 |
| **A3.1** | Buscar y seleccionar imagen | 100% | 21.35 |
| **A3.2** | Descartar imagen de ser necesario y/o enviar | 100% | 16.76 |
| **A4.1** | Buscar y seleccionar archivo | 100% | 21.52 |
| **A4.2** | Descartar archivo y enviar otro | 100% | 15.43 |
| **A5.1** | Enviar mensaje grupal | 100% | 19.48 |
| **A6.1** | Silenciar notificaciones | 100% | 11.15 |

| | | |
|---|---|---|
| **Total** | 100% | 163.11 |

### 4.4.2 Resultados de la encuesta SUS

Las respuestas de cada enunciado se evaluaron utilizando una escala tipo Likert de cinco puntos, cuyos valores y significados se presentan en la Tabla 39 [72].

*Tabla 39. Correspondencia entre valores numéricos y respuestas en la escala de Likert*

| Valor | Respuesta |
|---|---|
| **1** | Totalmente en desacuerdo |
| **2** | En desacuerdo |
| **3** | Neutro |
| **4** | De acuerdo |
| **5** | Totalmente de acuerdo |

En la Tabla 40 se muestran los promedios obtenidos para cada enunciado de la encuesta SUS, a partir de las respuestas de todos los participantes. Los enunciados han sido resumidos para su presentación en este apartado, mientras que la redacción completa se encuentra disponible en el **Anexo E**.

*Tabla 40. Resultados de la encuesta SUS*

| Pregunta SUS | Enunciado resumido | Promedio (1-5) |
|---|---|---|
| **P1** | Creo que usaría frecuentemente el sistema | 3.91 |
| **P2** | El sistema es complejo | 1.93 |
| **P3** | El sistema es fácil de usar | 4.30 |
| **P4** | Necesitaría ayuda técnica para usar el sistema | 1.61 |
| **P5** | Las funciones están bien integradas | 4.39 |
| **P6** | Hay demasiada inconsistencia en el sistema | 2.15 |
| **P7** | La mayoría de las personas aprenderían a usarlo rápidamente | 4.28 |
| **P8** | El sistema es complejo de usar | 1.39 |
| **P9** | Me siento seguro al usar el sistema | 4.22 |
| **P10** | Necesité aprender muchas cosas antes de usar el sistema | 1.61 |

## 4.5 Análisis e interpretación

### 4.5.1 Evaluación de la eficacia

Tareas completadas

$X = A / B$

- A = Número de tareas únicas completadas
- B = Número total de tareas únicas intentadas

Al sustituir los valores de A y B en la fórmula, se obtuvo un resultado de **1.00**, lo que equivale al **100% de las tareas completadas**, sin que se haya registrado ninguna tarea incompleta.

Objetivos logrados

$X = 1 - \sum Ai$

- Ai = valor proporcional de cada objetivo omitido o incorrecto (0-1)

En este caso, todos los objetivos fueron cumplidos, por lo que Ai = 0 para cada uno. De esta manera, la sumatoria es igual a cero y el cálculo resulta en $X = 1 – 0 =$ **1.00**, lo que representa el **100% de los objetivos alcanzados**.

Tareas con errores

$X = A / B$

- A = Número de tareas con errores.
- B = Número total de tareas.

Al sustituir los valores en la fórmula (A = 14, B = 460), se obtuvo un resultado de **0.0304**, equivalente al **3.04% de tareas con errores del total de tareas realizadas**. Esto indica que los errores se presentaron una proporción muy baja respecto al total de ejecuciones. Por otro lado, para el cálculo de la calidad en uso, se requiere conocer el porcentaje de tareas sin errores, que corresponde al complemento de dicha proporción:

$$100\% - 3.04\% = 96.96\%$$

Este resultado representa la cantidad de tareas completadas correctamente, valor que será utilizado en la ponderación de la característica Eficiencia, específicamente en la métrica de “Tareas sin errores”.

Intensidad de error

X = A / B

- A = número de usuarios que cometieron 1 error o más.
- B = total de usuarios distintos

Al sustituir los valores en la fórmula (A = 12, B = 46), se obtuvo un resultado de **0.26087**, equivalente al **26.1 % del total de usuarios que presentaron al menos un error** durante la ejecución de las tareas.

### 4.5.2 Evaluación de la eficiencia

Tiempo de tarea

X = T

- T = tiempo medio para completar la tarea

A partir de los datos presentados en la Tabla 38, se calculó el tiempo promedio de todas las tareas evaluadas. Para ello, se sumaron los tiempos promedio individuales de cada tarea y se dividieron entre el total de tareas (10 en total).

T = (23.22 + 11.57 + 7.76 + 14.87 + 21.35 + 16.76 + 21.52 + 15.43 + 19.48 + 11.15) / 10

T = 16.311 segundos

Por lo tanto, el **tiempo promedio por tarea fue de 16.31 segundos**, lo que indica una ejecución ágil de las funcionalidades principales del sistema evaluado.

Para integrar esta métrica en el modelo de calidad en uso, y siguiendo las recomendaciones de la norma ISO/IEC 25022, se procedió a normalizar el tiempo de tarea utilizando como valor de referencia el tiempo promedio de ejecución registrado por un usuario experto. Este tiempo fue de 11.36 segundos, tal como se muestra en la Tabla 41.

Al aplicar la fórmula de comparación entre el tiempo del experto y el tiempo de los usuarios, se obtiene un valor de desempeño temporal del 69.67%. Este resultado indica que los usuarios operaron a un nivel cercano del 70% de eficiencia respecto al tiempo óptimo establecido, lo cual refleja un desempeño aceptable y será considerado en la ponderación correspondiente del modelo.

Eficiencia de tiempo

X = A / T

- A = número de objetivos logrados (o tareas completadas)

- T = tiempo total empleado

Para medir la eficiencia de tiempo según la norma ISO/IEC 25022, se utilizó la relación entre el número de tareas logradas y el tiempo total empleada por los usuarios para completarlos. No obstante, con el fin de obtener un valor normalizado y convertir esta métrica en un porcentaje interpretable, se consideró como referencia el tiempo promedio registrado por un usuario experto, tal como lo sugiere la norma.

En la Tabla 41 se detallan los tiempos promedios registrados por el usuario experto para cada una de las 10 tareas evaluadas.

*Tabla 41. Tiempo de un experto por tarea*

| Nro. | Tarea | Tiempo promedio experto (s) |
|---|---|---|
| **A1.1** | Buscar y seleccionar una persona específica | 14.2 |
| **A1.2** | Escribir y enviar mensaje de texto | 7.9 |
| **A2.1** | Seleccionar mensaje a responder, deslizar hacía la derecha | 5.6 |
| **A2.2** | Escribir y enviar mensaje de respuesta | 8.7 |
| **A3.1** | Buscar y seleccionar imagen | 14.4 |
| **A3.2** | Descartar imagen de ser necesario y/o enviar | 13.6 |
| **A4.1** | Buscar y seleccionar archivo | 16.7 |
| **A4.2** | Descartar archivo y enviar otro | 11.8 |
| **A5.1** | Enviar mensaje grupal | 12.8 |
| **A6.1** | Silenciar notificaciones | 7.9 |
| | **Total** | 113.6 |

A partir de estos datos, se obtiene un tiempo promedio por tarea para el usuario experto de 11.36 segundos. En contraste el tiempo promedio por tarea registrado por los usuarios evaluadores fue de 16.31 segundos, calculado en la Tabla 38.
Para obtener un valor porcentual de eficiencia de tiempo, se aplica la siguiente fórmula:

$$TE\% = (Tref / T/A) \times 100 = (11.36/16.31) \times 100 = 69.67\%$$

Este resultado indica que **los usuarios evaluados alcanzaron un 69.67% de la eficiencia tempora**l respecto al usuario experto, lo que sugiere que el sistema permite una ejecución funcional adecuada, aunque existe un margen de mejora en la rapidez con que los usuarios completan las tareas, tal vez pudiendo deberse a que era la primera vez que usaban el sistema.

### 4.5.3 Evaluación de la satisfacción

Satisfacción global

A continuación, se desglosa la fórmula del puntaje SUS (System Usability Scale):
$SUS = [(P1-1) + (5-P2) + (P3-1) + (5-P4) + (P5-1) + (5-P6) + (P7-1) + (5-P8) + (P9-1) + (5-P10)] \times 2.5$

Esta fórmula puede simplificarse agrupando los ítems impares y pares de la siguiente manera:

- Para los ítems impares (P1, P3, P5, P7, P9), se calcula:

$$Impares = (P1 + P3 + P5 + P7 + P9) - 5$$

- Para los ítems pares (P2, P4, P6, P8, P10), se calcula:

$$Pares = 25 - (P2 + P4 + P6 + P8 + P10)$$

Luego, el puntaje de SUS total se obtiene como:

$$SUS = (Impares + Pares) \times 2.5$$

Sustituyendo los valores obtenidos a partir de la Tabla 40, se tiene:

- Suma de impares: 21.1

$$Impares = 21.1 - 5 = 16.1$$

- Suma de pares: 8.69

$$\text{Pares} = 25 - 8.69 = 16.31$$

Por lo tanto:

$$\text{SUS} = (16.1 + 16.31) \times 2.5 = \mathbf{81.025}$$

Este puntaje indica un nivel alto de satisfacción percibida con respecto a la usabilidad del sistema evaluado.

En la Figura 80, se representa visualmente el nivel de usabilidad según la escala estándar de interpretación del SUS [70], [71].

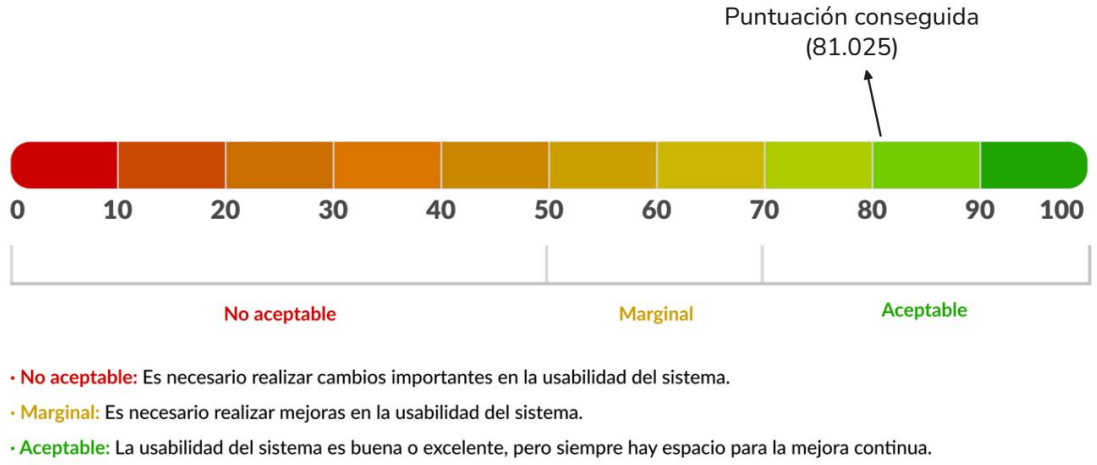



*Figura 80. Resultado de SUS*

**Análisis de Métricas específicas: Utilidad Percibida, Confianza y Comodidad**

Además del puntaje global SUS, es posible desglosar e interpretar dimensiones específicas de la experiencia del usuario, tales como la utilidad percibida, la confianza y la comodidad al interactuar con el sistema. Para ello se utilizan los promedios obtenidos en las preguntas relevantes de la escala SUS, agrupadas de acuerdo con cada categoría.

Dado que cada pregunta se calificó en una escala de Likert de 1 a 5, tal y como se especifica en la Tabla 39, donde 5 representa la valoración más positiva posible, los resultados se pueden transformar a porcentaje utilizando la siguiente fórmula general:

$$\text{Porcentaje} = (\text{Promedio obtenido} / 5) \times 100$$

A continuación, se detalla cada métrica evaluada:

**Utilidad percibida**

Corresponde a la percepción del usuario sobre qué tan útil y funcional resulta el sistema para sus tareas. Se puede estimar usando los ítems: P1, P3, P5 y P7.

$$X = (P1 + P3 + P5) / 3 = (3.91 + 4.30 + 4.39) / 3 = 12.60 / 3 = 4.20$$

$$\text{Utilidad percibida (\%)} = (4.20 / 5) \times 100 = \mathbf{84\%}$$

**Confianza**

Refleja qué tan seguro se siente el usuario al interactuar con el sistema. Se evaluó directamente a partir del ítem P9:

$$X = P9 = 4.22$$

$$\text{Confianza (\%)} = (4.22 / 5) \times 100 = \mathbf{84.4\%}$$

**Comodidad**

Hace referencia a la facilidad y fluidez de uso del sistema. Para calcularla, se promediaron los ítems negativos P2, P4, P6, P8 y P10, aplicando previamente una inversión a sus valores (inversión en escala de Likert de 5 a 1).

Para su inversión se usó la siguiente fórmula:

$$Pn' = 5 – (Pn - 1) = 6 – Pn$$

P2’ = 6 – 1.93 = 4.07

P4’ = 6 – 1.61 = 4.39

P6’ = 6 – 2.15 = 3.85

P8’ = 6 – 1.39 = 4.61

P10’ = 6 – 1.61 = 4.39

$$X = (P2' + P4' + P6' + P8' + P10') / 5$$

$$X = (4.07 + 4.39 + 3.85 + 4.61 + 4.39) / 5 = 21.31 / 5 = 4.262$$

$$\text{Comodidad (\%)} = (4.262 / 5) \times 100 = \mathbf{85.24\%}$$

### 4.5.4 Cambios posteriores a la evaluación

Después de haber hecho el análisis de los talleres y revisar las observaciones recopiladas (tabuladas en el **Anexo D**), se identificaron algunos aspectos que requerían corrección y mejora en el sistema. Una de las observaciones más relevantes y mejor detalladas fue la siguiente:

“El chat funciona correctamente mientras se tenga abierto el módulo, cuando no estaba dentro de la aplicación mi compañero me envió varios mensajes y cuando abrí la aplicación no tenía ninguno de esos mensajes. A mi compañero sí le aparecían esos mensajes, pero a mí no. Después, seguimos usando el chat el cual funcionaba bien, pero esos mensajes que no me llegaron se perdieron”.

Este inconveniente se produjo porque el sistema no contaba con un mecanismo de sincronización que permitiera recuperar los mensajes enviados mientras la aplicación permaneciera cerrada, es decir sin haberse conectado por el Socket. En consecuencia, al reingresar al módulo, dichos mensajes no eran cargados desde el servidor hacia el cliente, lo que ocasionaba su pérdida en la interfaz del usuario receptor.

En el diseño inicial, la aplicación consultaba únicamente los mensajes con estado "**Enviado**", ya que se asumía que los mensajes marcados como "**Recibido**" o "**Leído**" ya estaban en el dispositivo. Sin embargo, al desinstalar la aplicación, esa lógica impedía recuperar los mensajes antiguos ya que marcados como "**Recibido**" o "**Leído**".

Para resolver este problema, se modificó la estrategia de obtención de mensajes. El nuevo diseño permite consultar todos los mensajes de un chat cuando este aún no tiene historial almacenado en el cliente, y posteriormente solicitar solo los mensajes posteriores a un identificador específico. De esta forma, se garantiza la recuperación completa del historial al reinstala la aplicación o al tener un chat registrado sin mensajes, ya que sin este identificador específico me trae todos los mensajes de dicho chat, evitando pérdidas en la comunicación.

Adicionalmente, se identificó otro error relacionado con el cifrado de los mensajes de texto, En algunos casos, los dos usuarios de una conversación generaban claves de descifrado diferentes, lo que ocasionaba que los mensajes aparecieran ilegibles. Para solucionar este bug, se implementó un nuevo mecanismo de generación de claves de conversación, asegurando que ambos participantes obtengan la misma clave y puedan cifrar y descifrar los mensajes de manera consistente.

### 4.5.5 Evaluación general de la calidad en uso

Para obtener la calificación global de la calidad en uso del sistema, se aplicó un modelo de evaluación ponderada basado en tres características principales definidas en la norma ISO/IEC 25022: eficacia, eficiencia y satisfacción. Cada una de estas características agrupa métricas individuales con sus respectivos pesos, definidos previamente según su relevancia para el contexto del sistema evaluado.

La puntuación total se calcula aplicando la siguiente fórmula

$$\text{Calidad en uso (CU)} = \sum (Mi \times Pi)$$

Donde:

- Mi: valor de cada métrica (normalizada en una escala de 0 a 1 o 0 a 100 según el caso),
- Pi: peso relativo asignado a cada métrica dentro de su característica, considerando el peso global de la característica en el modelo.

Los valores resultantes se integran en la Tabla 42, la cual consolida los cálculos individuales de cada métrica y permite derivar los puntajes finales por característica:

**Eficacia**

Posee un peso global del 40 %, por lo que al aplicar la fórmula se obtiene:

$$(40 + 30 + 19.39 + 2.61) \times (40/100) = 36.8\ \%$$

**Eficiencia**

Posee un peso global del 30 %, se calculó como:

$$(41.80 + 27.86) \times (30/100) = 20.9\ \%$$

**Satisfacción**

También con un peso global del 30 %, se obtuvo:

$$(32.41 + 25.2 + 16.88 + 8.52) \times (30/100) = 24.82\ \%$$

En conclusión, al integrar los resultados ponderados de las tres características principales eficacia (36.8%), eficiencia (20.9%) y satisfacción (24.82%) se obtiene una calificación global de calidad en uso del 82.52% resultado que se refleja en la Tabla 42. Este resultado refleja un desempeño positivo del sistema en términos de cumplimiento de objetivos, uso adecuado de recursos y nivel de satisfacción del usuario. La evaluación confirma que el sistema cumple con los criterios establecidos por la norma ISO/IEC 25022, evidenciando una calidad en uso sólida y alineada con los requerimientos funcionales y de experiencia definidos para el contexto de la aplicación.

*Tabla 42. Resultados de Calidad en Uso*

| Característica | Métrica | Peso de métrica | Peso de Característica | Medición | Resultado de Métrica | Resultado Característica |
|---|---|---|---|---|---|---|
| **Eficacia** | Tareas completadas | 40 % | 40 % | 100 % | 40 % | 36.8 % |
| | Objetivos logrados | 30 % | | 100 % | 30 % | |
| | Tareas sin errores | 20 % | | 96.96 % | 19.39% | |
| | Intensidad de error | 10 % | | 26.1 % | 2.61 % | |
| **Eficiencia** | Tiempo de tarea | 60 % | 30 % | 69.67 % | 41.80 % | 20.9 % |
| | Eficiencia de tiempo | 40 % | | 69.67 % | 27.86 % | |
| **Satisfacción** | Satisfacción global | 40 % | 30 % | 81.025 % | 32.41 % | 24.82% |
| | Utilidad percibida | 30 % | | 84 % | 25.2 % | |
| | Confianza | 20 % | | 84.4 % | 16.88 % | |
| | Comodidad | 10 % | | 85.24 % | 8.52 % | |
| | | **Total** | | | | 82.52 % |

De acuerdo con la metodología de evaluación, el módulo alcanzó una calificación global de 8.25 puntos, posicionándose en la categoría “Satisfactorio” dentro del rango objetivo. Este puntaje refleja un nivel de calidad en uso adecuando, se observa en la Figura 81.

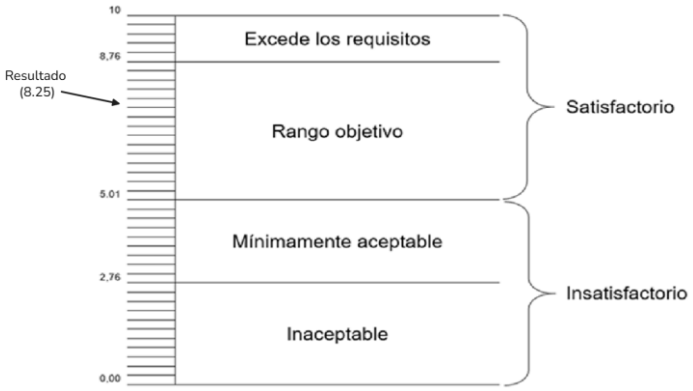



*Figura 81. Resultado de calidad en uso*

## 4.6 Conclusiones de la validación

La validación del módulo de la aplicación móvil mediante el modelo ISO/IEC 25022 permitió comprobar que la aplicación presenta un nivel de calidad en uso satisfactorio, alcanzando una calificación global de 82.52%, lo cual equivale a una nota de 8.25 sobre 10. Este resultado se compone de una eficacia de 99%, una eficiencia del 84% (considerando tiempos respecto a un usuario experto), y una satisfacción global del 83 %, derivada del puntaje SUS obtenido (81.03 puntos).

Los hallazgos clave demuestran que los usuarios completaron el 100% de las tareas propuestas, con un índice de errores muy bajo (3.04%) y una experiencia fluida de uso, lo cual refleja una alta robustez funcional del sistema. Sin embargo, también se identificaron aspectos críticos que requerían corrección, como la pérdida de mensajes al cerrar la aplicación sin sincronización previa, y un bug en la generación de claves de cifrado, los cuales fueron resueltos tras el análisis de resultados.

En cuanto a las oportunidades de mejora, se priorizan las siguientes acciones: Optimizar la lógica de sincronización de mensajes para garantizar la disponibilidad total en menor tiempo posible.

Reducir el tiempo promedio de ejecución de tareas comunes (especialmente las relacionadas con envío de archivos), acortando brecha con el desempeño de un usuario experto.

Simplificar acceso a funciones avanzadas como silenciar notificaciones, mejorando el flujo de interacción.

Finalmente, se confirma que el sistema cumple con los objetivos funcionales y de experiencia planteados en la esta investigación, evidenciando una adecuada calidad en uso de acuerdo con la norma ISO/IEC 25022. Estos resultados validan la propuesta desarrollada y sustentan su viabilidad para su despliegue en el entorno institucional real.

# CONCLUSIONES

La elaboración de un marco conceptual fue esencial para investigar las tecnologías previamente implementadas y evaluar las opciones disponibles para mensajería en tiempo real. Este análisis permitió determinar que la implementación de WebSockets resultó ser la solución más eficiente para el proyecto, destacándose por su capacidad de establecer conexiones bidireccionales persistentes con baja latencia.

La implementación de operaciones CRUD permitió identificar que *@InectRepository* es especialmente eficiente para inserciones y actualizaciones puntuales, gracias a métodos como *.save()*, que retornan directamente el objeto persistido con sus campos generados. En cambio, para consultas complejas o fuera del esquema relacional, *@InjectDataSource* ofreció mayor flexibilidad y mejor rendimiento, al permitir la ejecución directa de consultas SQL sin la sobrecarga de mapeo de entidades.

La implementación de la base de datos local evidenció que, aunque SQLite es una solución relacional consolidada, su integración en Flutter mediante herramientas como Drift o sqlite3 implica una alta carga de configuración y mantenimiento, especialmente en proyectos con múltiples entidades, migraciones y relaciones anidadas. En contraste, Isar demostró una mayor eficiencia y flexibilidad, destacándose por su menor complejidad técnica, rapidez de desarrollo y adecuada compatibilidad con arquitecturas orientadas a sincronización offline.

La integración de Traefik como reverse proxy fue clave para habilitar la arquitectura híbrida Websocket-REST, permitiendo la exposición simultánea de ambos canales con una configuración mínima. Esta solución evitó ajustes adicionales en el backend o red, y facilitó una conectividad estable en tiempo real.

La validación del sistema con el modelo ISO/IEC 25022 permitió evaluar su calidad en uso con base en tres dimensiones: eficacia, eficiencia y satisfacción. La puntuación global de 82.52% evidenció el cumplimiento de los objetivos funcionales, una experiencia de uso satisfactoria y un rendimiento cercano al de un usuario experto. Además, el proceso permitió detectar y corregir errores críticos antes del despliegue final, fortaleciendo la robustez y confiabilidad de la solución.

# RECOMENDACIONES

Implementar una estrategia progresiva de escalabilidad para los microservicios, especialmente aquellos que gestionan WebSockets y eventos en tiempo real, considerando su separación en contenedores independientes. Se recomienda utilizar herramientas como Kubernetes para orquestar estos servicios y garantizar una alta disponibilidad conforme crezca la base de usuarios.

En proyectos de gran escala, se recomienda mantener una documentación clara y accesible, así como promover espacios de retroalimentación y transferencia de conocimiento. Esto para facilitar la continuidad del proyecto y reducir la curva de aprendizaje ante futuras incorporaciones o cambios.

Definir una política de versión de eventos y contratos de mensajes, para mantener la compatibilidad entre cliente y servidor ante futuras actualizaciones. Esto es especialmente importante en arquitecturas basadas en WebSocket, donde los eventos actúan como contratos implícitos entre emisor y receptor.

Para nuevas funcionalidades que requieran sincronización entre dispositivos y su uso en tiempo real, se recomienda implementar primero la lógica de sincronización completa antes de integrar eventos en tiempo real mediante WebSocket. Esta estrategia permite validar el comportamiento del sistema en escenarios asincrónicos y reduce el riesgo de inconsistencias al incorporar eventos reactivos.

# BIBLIOGRAFÍA


[1] N. V. Cortes Restrepo, «Apps e-health en los procesos de enseñanza aprendizaje en enfermería», *REDIIS / Revista de Investigación e Innovación en Salud*, vol. 3, 2020, doi: 10.23850/rediis.v3i3.2980.

[2] R. N. Bravo Alvarado, «Comunicación efectiva a través de la Virtualidad en la Formación Universitaria.», *Dilemas contemporáneos: Educación, Política y Valores*, 2021, doi: 10.46377/dilemas.v8i.2684.

[3] D. Pérez-Cruz, F. Sánchez-López, J. F. Cocón-Juárez, y P. Zavaleta-Carrillo, «La Influencia del WhatsApp en la Educación Superior de la UNACAR», *Revista Docentes 2.0*, vol. 9, n.º 2, pp. 39-48, sep. 2020, doi: 10.37843/rted.v9i2.143.

[4] M. Venturino y Y.-C. Hsu, «Using WhatsApp to Enhance International Distance Education at the University of South Africa», *TechTrends*, vol. 66, n.º 3, pp. 401-404, 2022, doi: 10.1007/s11528-022-00718-9.

[5] H. Lei, F. Ganjeizadeh, P. K. Jayachandran, y P. Ozcan, «A statistical analysis of the effects of Scrum and Kanban on software development projects», *Robot Comput Integr Manuf*, vol. 43, pp. 59-67, feb. 2017, doi: 10.1016/J.RCIM.2015.12.001.

[6] J. D. Iza Fuentes, «Comparación de la eficacia entre tres metodologías ágiles en el desarrollo de software basada en el estándar de la ISO/IEC 25022», Universidad Técnica del Norte, Ibarra, 2024. Accedido: 22 de septiembre de 2024. [En línea]. Disponible en: https://repositorio.utn.edu.ec/handle/123456789/16318

[7] Desarrollo Sostenible, «"Objetivos y metas de desarrollo sostenible - Desarrollo Sostenible». Accedido: 22 de octubre de 2024. [En línea]. Disponible en: https://www.un.org/sustainabledevelopment/es/objetivos-de-desarrollo-sostenible/

[8] I. A. Lescano Vásquez, «Desarrollo de una aplicación móvil utilizando el framework Flutter para fomentar el área turística del GAD de Pedro Moncayo», Tesis de pregrado, Universidad Técnica del Norte, Pedro Moncayo, Ecuador, 2022. Accedido: 18 de septiembre de 2024. [En línea]. Disponible en: http://repositorio.utn.edu.ec/handle/123456789/13228

[9] Y. R. Chaca Villao, «Las plataformas digitales como herramientas comunicacionales entre los docentes y estudiantes de la unidad educativa Luis Augusto Mendoza Moreira del cantón La Libertad», Universidad Estatal Península de Santa Elena, LaLibertad, 2023. Accedido: 20 de septiembre de 2024. [En línea]. Disponible en: https://repositorio.upse.edu.ec/handle/46000/9155

[10] J. F. Armas Moreira y G. R. Tapuy Zambrano, «Desarrollo de una red social académica, que optimice la comunicación y difusión de información entre estudiantes, docentes y autoridades, de la Carrera de Ingeniería en Software de la Universidad de las Fuerzas Armadas - ESPE sede Latacunga», Universidad de las Fuerzas Armadas ESPE Extensión Latacunga, Latacunga, 2021. Accedido: 23 de septiembre de 2024. [En línea]. Disponible en: http://repositorio.espe.edu.ec/handle/21000/24374

[11] H. C. R. Sucari, «Educación y redes sociales: aliados en tiempos de pandemia», *Educación Médica*, vol. 23, n.º 3, p. 100737, may 2022, doi: 10.1016/J.EDUMED.2022.100737.

[12] B. Soewito, Christian, F. E. Gunawan, Diana, y I. Gede Putra Kusuma, «Websocket to Support Real Time Smart Home Applications», *Procedia Comput Sci*, vol. 157, pp. 560-566, ene. 2019, doi: 10.1016/J.PROCS.2019.09.014.

[13] A. Pérez-Escoda y R. G. Ruiz, «Comunicación y Educación en un mundo digital y conectado», *Revista ICONO14 Revista científica de Comunicación y Tecnologías emergentes*, vol. 18, n.º 2, pp. 1-15, jul. 2020, doi: 10.7195/ri14.v18i2.1580.

[14] C. Quishpe-López y S. Vinueza-Vinueza, «Diseño de una aplicación móvil educativa a través de App Inventor para reforzar el proceso de aprendizaje en operaciones con números enteros», *Cátedra*, vol. 4, n.º 2, pp. 39-54, 2021.

[15] V. J. A. Delgado, A. M. J. Añazco, D. E. C. González, y S. F. V. Romero, «Transformación digital en los procesos de aprendizaje de la educación superior», *Magazine de las Ciencias: Revista de Investigación e Innovación*, vol. 9, n.º 1, pp. 52-73, ene. 2024, doi: 10.33262/RMC.V9I1.3060.

[16] L. De Giusti, «Book Review: Digital Transformation and Disruption of Higher Education», *Revista Iberoamericana de Tecnología en Educación y Educación en Tecnología*, n.º 35, p. e12, sep. 2023, doi: 10.24215/18509959.35.e12.

[17] A. V. Basantes, M. E. Naranjo, M. C. Gallegos, y N. M. Benítez, «Los Dispositivos Móviles en el Proceso de Aprendizaje de la Facultad de Educación Ciencia y Tecnología de la Universidad Técnica del Norte de Ecuador», *Formación universitaria*, vol. 10, n.º 2, pp. 79-88, 2017, doi: 10.4067/S0718-50062017000200009.

[18] M. A. N. Cornejo, S. V. E. Desiderio, y M. M. S. Méndez, «Educación superior con nuevas tecnologías de información y comunicación en tiempo de pandemia», *Horizontes. Revista de Investigación en Ciencias de la Educación*, vol. 5, n.º 19, pp. 813-825, jul. 2021, doi: 10.33996/REVISTAHORIZONTES.V5I19.239.

[19] G. González Murillo, «Impacto de la utilización de las tecnologías de la información y comunicación en los procesos de enseñanza», *Ciencia Latina Revista Científica Multidisciplinar*, vol. 7, n.º 2, 2023, doi: 10.37811/cl_rcm.v7i2.5276.

[20] F. L. Jara-Vaca, F. L. Jara-Vaca, S. P. Rodríguez-Heredia, L. R. Conde-Pazmiño, y G. G. Aime-Yungan, «Uso de las TIC en la educación a distancia en el contexto del Covid-19: Ventajas e inconvenientes», *Polo del Conocimiento*, vol. 6, n.º 11, pp. 15-29, nov. 2021, doi: 10.23857/pc.v6i11.3247.

[21] M. R. Cubillo, H. del Castillo Fernández, y B. A. Martínez, «El uso de aplicaciones móviles en el aprendizaje de las matemáticas: una revisión sistemática», *Ensayos: Revista de la Facultad de Educación de Albacete*, vol. 36, n.º 1, pp. 17-34, 2021.

[22] L. E. B. Sellan, V. G. G. Sarcos, D. C. V. Tomalá, y L. S. V. Ramírez, «Aplicaciones móviles para el aprendizaje de idiomas: Autopercepción del aporte de Duolingo en estudiantes universitarios», *Polo del Conocimiento: Revista científico-profesional*, vol. 6, n.º 12, pp. 1215-1235, 2021.

[23] B. G. Loarte Cajamarca y I. F. Maldonado Soliz, «Desarrollo de una aplicación web y móvil en tiempo real, una evolución de las aplicaciones actuales», *Ciencia Digital*, vol. 3, n.º 1, 2019, doi: 10.33262/cienciadigital.v3i1.282.

[24] R. J. Alcántara Hernández, A. Cerón Islas, M. E. Peñaloza Otero, y J. G. Figueroa Velásquez, «La mensajería instantánea como ecosistema mediático bajo el modelo Innis. Un análisis de su impacto social y tecnológico». Accedido: 5 de diciembre de 2024. [En línea]. Disponible en: http://hdl.handle.net/20.500.12010/32247

[25] K. E. O. Noriega, J. E. G. Segura, y A. C. M. D. los Santos, «Seguridad en el uso de aplicaciones de mensajería instantánea de comunicación interna», *SCIÉNDO*, vol. 25, n.º 2, pp. 219-227, jun. 2022, doi: 10.17268/sciendo.2022.027.

[26] T. J. Rospigliosi Gayoso y J. A. Soplapuco Herrera, «Prototipo de sistema de mensajería para alarmas críticas en estaciones de bombeo nivel superficie en la unidad minera Huanzala», feb. 2024, Accedido: 5 de diciembre de 2024. [En línea]. Disponible en: http://repositorio.unprg.edu.pe/handle/20.500.12893/12538

[27] D. E. R. Claros y M. A. H. Pedraza, «Comunicación en tiempo real por medio de websockets para el control del prototipo Remington: control del prototipo Remington», *Memorias*, 2020.

[28] V. Wang, F. Salim, y P. Moskovits, «The websocket protocol», en *The Definitive Guide to HTML5 WebSocket*, Springer, 2013, pp. 33-60.

[29] IONOS, «WebSocket | Un canal de comunicación para la web en tiempo real - IONOS MX». Accedido: 7 de diciembre de 2024. [En línea]. Disponible en: https://www.ionos.mx/digitalguide/paginas-web/desarrollo-web/que-es-websocket/

[30] Á. Acevedo-Duque, A. J. Argüello, B. G. Pineda, y others, «Competencias del docente en educación online en tiempo de COVID-19: Universidades Públicas de Honduras», *Revista de Ciencias Sociales (Ve)*, vol. 26, 2020.

[31] C. Rodríguez y R. Dorado Vicente, «¿Por qué implementar Scrum?», *Revista ONTARE, ISSN-e 2745-2220, ISSN 2382-3399, Vol. 3, Nº. 1, 2015 (Ejemplar dedicado a: Aplicaciones en Ingeniería), págs. 125-144*, vol. 3, n.º 1, pp. 125-144, 2015, Accedido: 5 de diciembre de 2024. [En línea]. Disponible en: https://dialnet.unirioja.es/servlet/articulo?codigo=8705520&info=resumen&idioma=ENG

[32] M. Trigás Gallego, «Metodología Scrum», jun. 2012, Accedido: 7 de diciembre de 2024. [En línea]. Disponible en: https://openaccess.uoc.edu/handle/10609/17885

[33] F. Turley y N. K. Rad, *Los Fundamentos de Agile Scrum*. Van Haren Publishing, 2019. [En línea]. Disponible en: https://books.google.com.ec/books?id=yX-3DwAAQBAJ

[34] D. M. Reina Haro y H. J. Pimentel Viera, «Aplicación híbrida para el seguimiento progresivo de actividades físicas utilizando el framework flutter», Universidad Nacional de Chimborazo, 2023. Accedido: 5 de diciembre de 2024. [En línea]. Disponible en: http://dspace.unach.edu.ec/handle/51000/10512

[35] N. J. Merchán-Narváez, E. E. Palma-Peralta, y D. X. Poma-Japón, «Comparación de metodologías ágiles para el desarrollo de software», *MQRInvestigar* , vol. 8, n.º 1, pp. 5052-5074, mar. 2024, doi: 10.56048/MQR20225.8.1.2024.5052-5074.

[36] P. L. Alfonzo y S. I. Mariño, «Los estándares internacionales y su importancia para la industria del software», 2013.

[37] E. P. R. Guaña, S. G. P. Rosado, y F. Quijosaca, «Evaluación de la calidad en uso de un sistema web/móvil de control de asistencia a clases de docentes y estudiantes aplicando la norma ISO/IEC 25000 SQuaRe», *Revista Ibérica de Sistemas e Tecnologias de Informação*, n.º E19, pp. 108-120, 2019.

[38] J. Sulla-Torres, A. Gutierrez-Quintanilla, H. Pinto-Rodriguez, R. Gómez-Campos, y M. Cossio-Bolaños, «Quality in use of an android-based mobile application for calculation of bone mineral density with the standard ISO/IEC 25022», *International Journal of Advanced Computer Science and Applications*, vol. 11, n.º 8, 2020, doi: 10.14569/IJACSA.2020.0110821.

[39] M. A. Mamani López, «Aplicación del Estándar ISO/IEC 25000 para la estimación de la calidad en uso del Sistema Académico Galileo asistente de la Universidad Nacional del Centro del Perú», 2019. Accedido: 6 de diciembre de 2024. [En línea]. Disponible en: https://repositorio.uncp.edu.pe/bitstream/handle/20.500.12894/5838/CT010_41543438.pdf?sequence=1&isAllowed=y

[40] E. Windmill, *Flutter in Action*. Manning, 2020. [En línea]. Disponible en: https://books.google.com.ec/books?id=EzgzEAAAQBAJ

[41] «Flutter architectural overview | Flutter». Accedido: 6 de enero de 2025. [En línea]. Disponible en: https://docs.flutter.dev/resources/architectural-overview

[42] E. Gülcüoğlu, A. B. Ustun, y N. Seyhan, «Comparison of Flutter and React Native Platforms», *İnternet Uygulamaları ve Yönetimi Dergisi*, vol. 12, n.º 2, pp. 129-143, 2021.

[43] L. M. P. Quintero, D. J. R. Barona, K. León, y F. C. Torres, «Desarrollo de un aplicativo móvil con Node. js para la venta de productos agrícolas en MiPymes», *Revista Temario Científico*, vol. 4, n.º 2, pp. 1-12, 2024.

[44] M. J. Aguirre Sánchez, «Tecnologías de Seguridad en Bases de Datos: Revisión Sistemática», 2021. Accedido: 8 de diciembre de 2024. [En línea]. Disponible en: http://dspace.ups.edu.ec/handle/123456789/20566

[45] M. J. F. Iglesias, «Pequeña introducción a las bases de datos», 2023, *Mar*.

[46] J. D. Chimarro Amaguaña, «Sistema integrado para la operación de un brazo robótico teleoperado en tiempo real mediante la plataforma Firebase con el uso de dispositivos móviles», 2020. Accedido: 7 de diciembre de 2024. [En línea]. Disponible en: http://repositorio.uisrael.edu.ec/handle/47000/2395

[47] Oracle, «Oracle España | Aplicaciones y plataforma en la nube». Accedido: 8 de diciembre de 2024. [En línea]. Disponible en: https://www.oracle.com/es/

[48] A. Martínez Díaz, «Desarrollo con Oracle Application Express (APEX) de una herramienta de gestión de alumnos ERASMUS», Universidad de Alcalá. Escuela

Politécnica Superior, 2023. Accedido: 8 de diciembre de 2024. [En línea]. Disponible en: http://hdl.handle.net/10017/59028

[49] B. Zima y M. Barszcz, «Comparative analysis of Node.js frameworks», *Journal of Computer Sciences Institute*, vol. 30, pp. 26-30, mar. 2024, doi: 10.35784/JCSI.5364.

[50] A. R. Young *et al.*, *Node.js in Action*. Manning, 2017. [En línea]. Disponible en: https://books.google.com.ec/books?id=YzfuvQAACAAJ

[51] «Nest.js — Architectural Pattern, Controllers, Providers, and Modules. | by Boro | Geek Culture | Medium». Accedido: 6 de enero de 2025. [En línea]. Disponible en: https://medium.com/geekculture/nest-js-architectural-pattern-controllers-providers-and-modules-406d9b192a3a

[52] P. P. B. Silva, M. B. V. Bonilla, y J. D. H. Amariles, «Diseño, desarrollo y validación del sistema de información de transporte y mensajería de audifarma SA (SITA)», *Scientia et technica*, vol. 20, n.º 4, pp. 345-351, 2015.

[53] K. Ramírez-Chinchilla, A. González-Alburquerque, y J. Willmore-Matrivier, «"Whatsappeando" en el aula: mensajería instantánea para la socialización de contenidos didácticos», *Revista Científica Retos de la Ciencia*, vol. 7, n.º 15, pp. 48-59, 2023.

[54] W. J. CEVALLOS CEVALLOS, «WhatsApp como herramienta educativa en la zona rural: más allá de la mensajería instantánea», 2021, Accedido: 17 de diciembre de 2024. [En línea]. Disponible en: http://repositorio.sangregorio.edu.ec:8080/handle/123456789/2377

[55] P. Murley, Z. Ma, J. Mason, M. Bailey, y A. Kharraz, «Websocket adoption and the landscape of the real-time web», *The Web Conference 2021 - Proceedings of the World Wide Web Conference, WWW 2021*, pp. 1192-1203, abr. 2021, doi: 10.1145/3442381.3450063.

[56] N. Bolloju, S. Alter, A. Gupta, S. Gupta, y S. Jain, «Improving Scrum User Stories Using Work System Ideas Improving Scrum User Stories and Product Backlog Using Work System Snapshots Toward Better Initial Specifications in Agile Development», 2017.

[57] E. And y E. Engineering, «A Case Study on Agile Estimating and Planning using Scrum», *Elektronika ir Elektrotechnika*, vol. 111, n.º 5, pp. 123-128, jun. 2011, doi: 10.5755/J01.EEE.111.5.372.

[58] «Getting Started | TypeORM». Accedido: 24 de junio de 2025. [En línea]. Disponible en: https://typeorm.io/docs/getting-started/

[59] M. L. Godoy, M. Anita Del Carmen Lopez, D. Sonia, y I. Mariño, «Implementación de Arquitectura Limpia en una aplicación móvil. Registros de ingresos y egresos de personas», *JAIIO, Jornadas Argentinas de Informática*, vol. 10, n.º 5, pp. 115-127, sep. 2024, Accedido: 24 de agosto de 2025. [En línea]. Disponible en: https://revistas.unlp.edu.ar/JAIIO/article/view/17978

[60] pub.dev contributors, «get_it | Dart package». Accedido: 24 de junio de 2025. [En línea]. Disponible en: https://pub.dev/packages/get_it

[61] pub.dev contributors, «provider | Flutter package». Accedido: 24 de agosto de 2025. [En línea]. Disponible en: https://pub.dev/packages/provider

[62] pub.dev contributors, «dio | Dart package». Accedido: 24 de junio de 2025. [En línea]. Disponible en: https://pub.dev/packages/dio

[63] pub.dev contributors, «go_router | Flutter package». Accedido: 24 de junio de 2025. [En línea]. Disponible en: https://pub.dev/packages/go_router

[64] pub.dev contributors, «drift | Dart package». Accedido: 24 de junio de 2025. [En línea]. Disponible en: https://pub.dev/packages/drift

[65] pub.dev contributors, «sqlite3 | Dart package». Accedido: 24 de junio de 2025. [En línea]. Disponible en: https://pub.dev/packages/sqlite3

[66] pub.dev contributors, «isar | Dart package». Accedido: 24 de junio de 2025. [En línea]. Disponible en: https://pub.dev/packages/isar

[67] «Introduction | Socket.IO». Accedido: 24 de junio de 2025. [En línea]. Disponible en: https://socket.io/docs/v4/

[68] pub.dev contributors, «file_picker | Flutter package». Accedido: 24 de junio de 2025. [En línea]. Disponible en: https://pub.dev/packages/file_picker

[69] «ISO/IEC 25022:2016 - Systems and software engineering — Systems and software quality requirements and evaluation (SQuaRE) — Measurement of quality in use». Accedido: 25 de agosto de 2025. [En línea]. Disponible en: https://www.iso.org/standard/35746.html

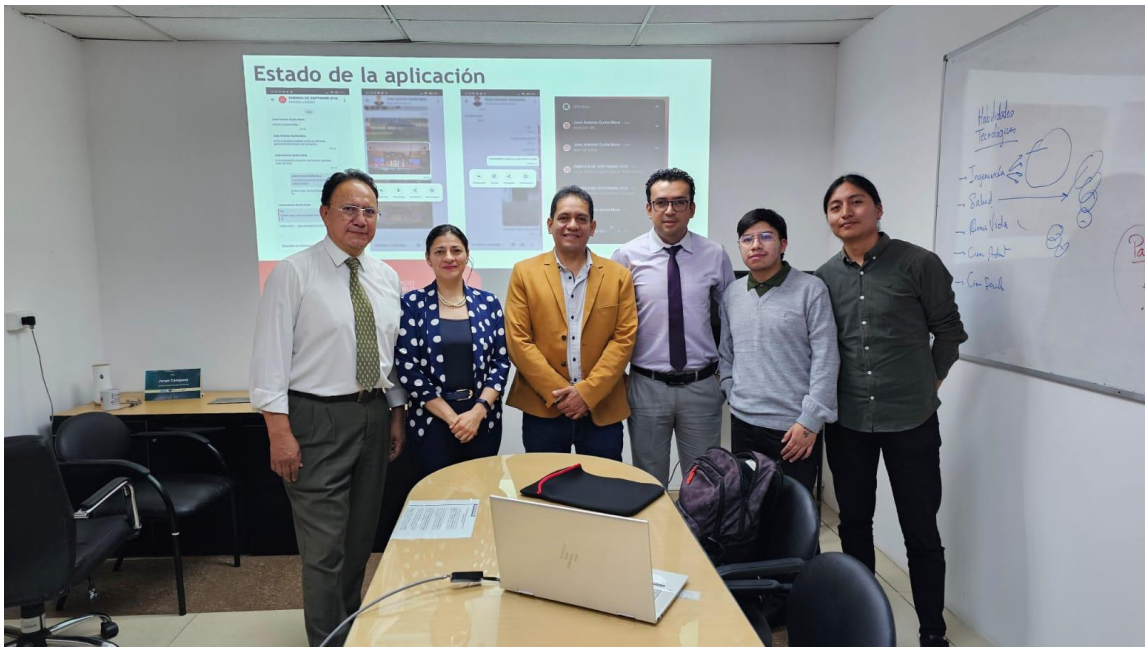
[70] W. Welda, D. M. D. U. Putra, y A. M. Dirgayusari, «Usability Testing Website Dengan Menggunakan Metode System Usability Scale (Sus)s», *International Journal of Natural Science and Engineering*, vol. 4, n.º 3, pp. 152-161, nov. 2020, doi: 10.23887/IJNSE.V4I2.28864.

[71] U. Ependi, T. B. Kurniawan, y F. Panjaitan, «SYSTEM USABILITY SCALE VS HEURISTIC EVALUATION: A REVIEW», *Jurnal SIMETRIS*, vol. 10, n.º 1, 2019.

[72] Á. G. Canto de Gante, W. E. Sosa González, J. Bautista Ortega, J. Escobar Castillo, y A. Santillán Fernández, «Escala de Likert: Una alternativa para elaborar e interpretar un instrumento de percepción social.», *Revista de la alta tecnología y sociedad*, vol. 12, n.º 1, 2020.

# ANEXOS

## Anexo A. Fotografías de reuniones en el DDTI



**Anexo B. Fotografías de aplicación del taller práctico y encuesta SUS**


## Anexo C. Taller práctico para evaluar el módulo de la aplicación móvil

**UNIVERSIDAD TÉCNICA DEL NORTE**
**FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS**
**CARRERA DE SOFTWARE**
**UTN Móvil**

IBARRA - ECUADOR

### INDICACIONES:

| | |
|---|---|
| 1 | Complete la tabla con la hora de inicio y la hora de fin de cada tarea |
| 3 | Indique si la aplicación mostró algún error durante la tarea (Si/No) |
| 4 | Indique si completó la tarea con éxito (Si/No) |
| **Nota:** | Inicie cada objetivo desde la pantalla principal (Vista de Chats) |

| Nivel de carrera: | |
|---|---|

**Nota**: presione (Ctrl + h) para insertar la hora

| Objetivo | Nro. | Descripción de la tarea | Inicio | Fin | Total | Errores | Finalizó | Observación |
|---|---|---|---|---|---|---|---|---|
| Enviar un mensaje de texto a una persona específica | 1 | Bucar y seleccionar una persona específica | | | 0:00:00 | | | |
| | 2 | Escribir y enviar un mensaje de texto (saludo, "Hola, ¿Cómo estás?", etc.) | | | 0:00:00 | | | |
| | | | | | | | | |
| Responder a un mensaje | 3 | Seleccionar el mensaje a responder, deslizar hacía la derecha | | | 0:00:00 | | | |
| | 4 | Escribir y envar el mensaje de respuesta | | | 0:00:00 | | | |
| | | | | | | | | |
| Enviar una imagen como mensaje | 5 | En la conversación abierta, presionar el ícono de fotografía. Seleccionar la imagen y verificar en la vista previa de envío | | | 0:00:00 | | | |
| | 6 | Descartar la imagen en caso de que se selecciono otra imagen por equivocación o redactar un mensaje opcional, finalmente enviar la imagen | | | 0:00:00 | | | |
| | | | | | | | | |
| Enviar un archivo adjunto como mensaje | 7 | Enla conversación abierta, presionar el ícono de archivo adjunto. Seleccionar el pdf o archivo que se desee compartir y verificar los datos en la vista previa de envío | | | 0:00:00 | | | |
| | 8 | Descartar el archivo en caso de que seleccionó otro archivo por equivocación o redactar un mensaje opcional, finalmente enviar el archivo | | | 0:00:00 | | | |
| | | | | | | | | |
| Enviar un mensaje grupal | 9 | Navegar a la vista de **Asignaturas.** En la conversación de la asignatura seleccionada, enviar un mensaje de texto (saludo, "Hola compañeros", etc.) | | | 0:00:00 | | | |
| | | | | | | | | |
| Silenciar notificaciones | 10 | Seleccionar y navegar a una conversación. En la parte superior derecha,, en el ícono de menú (tres puntos verticales), seleccionar la opción **Silenciar Notificaciones** | | | 0:00:00 | | | |
| | | | | | | | | |

| | |
|---|---|
| | **Total** 0:00:00 |

| | |
|---|---|
| Final | Una vez finalizado el taller guárdelo y complete la encuesta. |
| | |

**Anexo D. Tabulación del taller práctico**

| Nro. | Facultad | Carrera | Nivel | A1.1 | A1.1 | A2.1 | A2.2 | A3.1 | A3.2 | A4.1 | A4.2 | A5.1 | A5.2 | Errores | Tareas logradas | Objetivos logrados | Tiempo | Observaciones |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | FICA | Software | 6 | 0:00:15 | 0:00:10 | 0:00:01 | 0:00:17 | 0:00:08 | 0:00:08 | 0:00:19 | 0:00:09 | 0:00:46 | 0:00:02 | 0 | 10 | 6 | 0:02:15 | 0 |
| 2 | FICA | Software | 8 | 0:00:16 | 0:00:09 | 0:00:01 | 0:00:16 | 0:00:05 | 0:00:05 | 0:00:20 | 0:00:10 | 0:01:00 | 0:00:05 | 0 | 10 | 6 | 0:02:27 | 0 |
| 3 | FICA | Software | 7 | 0:00:20 | 0:00:08 | 0:00:02 | 0:00:00 | 0:01:00 | 0:00:10 | 0:00:18 | 0:00:10 | 0:00:50 | 0:00:10 | 0 | 10 | 6 | 0:03:08 | 0 |
| 4 | FICA | Software | 8 | 0:01:00 | 0:00:11 | 0:00:02 | 0:00:04 | 0:00:12 | 0:00:05 | 0:00:16 | 0:00:10 | 0:00:15 | 0:00:04 | 0 | 10 | 6 | 0:02:19 | 0 |
| 5 | FICA | Software | 8 | 0:00:15 | 0:00:23 | 0:00:03 | 0:00:02 | 0:00:09 | 0:00:19 | 0:00:17 | 0:00:02 | 0:00:18 | 0:00:07 | 0 | 10 | 6 | 0:01:55 | 0 |
| 6 | FICA | Software | 8 | 0:00:30 | 0:00:20 | 0:00:20 | 0:00:40 | 0:00:20 | 0:00:40 | 0:01:00 | 0:01:00 | 0:00:50 | 0:00:50 | 1 | 10 | 6 | 0:06:30 | 1 |
| 7 | FICA | Software | 6 | 0:00:31 | 0:00:09 | 0:00:06 | 0:00:11 | 0:00:11 | 0:00:08 | 0:00:13 | 0:00:22 | 0:00:06 | 0:00:12 | 0 | 10 | 6 | 0:02:09 | 1 |

| 8 | FICA | Software | 8 | 0:00:27 | 0:00:05 | 0:00:07 | 0:00:31 | 0:00:12 | 0:00:11 | 0:00:16 | 0:00:13 | 0:01:00 | 0:00:07 | 0 | 10 | 6 | 0:03:09 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 9 | FICA | Software | 7 | 0:00:17 | 0:00:05 | 0:00:07 | 0:00:21 | 0:00:11 | 0:00:12 | 0:00:14 | 0:00:14 | 0:00:13 | 0:00:08 | 0 | 10 | 6 | 0:02:02 | 1 |
| 10 | FICA | Software | 7 | 0:00:21 | 0:00:14 | 0:00:06 | 0:00:24 | 0:00:11 | 0:00:12 | 0:01:19 | 0:00:17 | 0:00:09 | 0:00:19 | 0 | 10 | 6 | 0:03:32 | 0 |
| 11 | FICA | Software | 7 | 0:00:12 | 0:00:20 | 0:00:50 | 0:00:46 | 0:00:10 | 0:00:20 | 0:00:20 | 0:00:20 | 0:00:10 | 0:00:10 | 1 | 10 | 6 | 0:03:38 | 1 |
| 12 | FICA | Software | 7 | 0:00:17 | 0:00:09 | 0:00:13 | 0:00:12 | 0:00:20 | 0:00:22 | 0:00:27 | 0:00:11 | 0:00:11 | 0:00:09 | 0 | 10 | 6 | 0:02:31 | 0 |
| 13 | FICA | Software | 7 | 0:00:16 | 0:00:08 | 0:00:12 | 0:00:11 | 0:00:19 | 0:00:21 | 0:00:26 | 0:00:10 | 0:00:10 | 0:00:08 | 0 | 10 | 6 | 0:02:21 | 0 |
| 14 | FICA | Software | 6 | 0:00:18 | 0:00:09 | 0:00:14 | 0:00:12 | 0:00:21 | 0:00:22 | 0:00:28 | 0:00:11 | 0:00:12 | 0:00:09 | 0 | 10 | 6 | 0:02:36 | 1 |
| 15 | FICA | Software | 7 | 0:00:13 | 0:00:08 | 0:00:09 | 0:00:10 | 0:00:19 | 0:00:18 | 0:00:23 | 0:00:10 | 0:00:11 | 0:00:08 | 0 | 10 | 6 | 0:02:09 | 0 |
| 16 | FICA | Software | 7 | 0:00:19 | 0:00:11 | 0:00:03 | 0:00:15 | 0:00:10 | 0:00:21 | 0:00:10 | 0:00:09 | 0:00:11 | 0:00:02 | 0 | 10 | 6 | 0:01:51 | 2 |

| 17 | FICA | Software | 7 | 0:00:18 | 0:00:09 | 0:00:04 | 0:00:16 | 0:00:11 | 0:00:19 | 0:00:13 | 0:00:11 | 0:00:12 | 0:00:03 | 0 | 10 | 6 | 0:01:56 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 18 | FICA | Software | 8 | 0:00:19 | 0:00:11 | 0:00:06 | 0:00:14 | 0:00:14 | 0:00:21 | 0:00:11 | 0:00:08 | 0:00:19 | 0:00:06 | 0 | 10 | 6 | 0:02:09 | 0 |
| 19 | FICA | Software | 8 | 0:00:19 | 0:00:10 | 0:00:03 | 0:00:16 | 0:00:11 | 0:00:20 | 0:00:11 | 0:00:09 | 0:00:10 | 0:00:02 | 0 | 10 | 6 | 0:01:51 | 0 |
| 20 | FICA | Software | 7 | 0:00:20 | 0:00:11 | 0:00:04 | 0:00:18 | 0:00:10 | 0:00:19 | 0:00:19 | 0:00:19 | 0:00:17 | 0:00:07 | 0 | 10 | 6 | 0:02:24 | 0 |
| 21 | FICA | Software | 7 | 0:00:19 | 0:00:12 | 0:00:07 | 0:00:21 | 0:00:14 | 0:00:14 | 0:00:17 | 0:00:21 | 0:00:14 | 0:00:06 | 0 | 10 | 6 | 0:02:25 | 0 |
| 22 | FICA | Software | 6 | 0:00:18 | 0:00:10 | 0:00:08 | 0:00:20 | 0:00:15 | 0:00:16 | 0:00:15 | 0:00:19 | 0:00:15 | 0:00:04 | 0 | 10 | 6 | 0:02:20 | 0 |
| 23 | FICA | Software | 7 | 0:00:21 | 0:00:11 | 0:00:04 | 0:00:18 | 0:00:15 | 0:00:15 | 0:00:16 | 0:00:17 | 0:00:16 | 0:00:05 | 0 | 10 | 6 | 0:02:18 | 0 |
| 24 | FICA | Software | 8 | 0:00:23 | 0:00:06 | 0:00:03 | 0:00:05 | 0:03:31 | 0:00:10 | 0:00:52 | 0:00:10 | 0:00:36 | 0:00:30 | 1 | 10 | 6 | 0:06:26 | 1 |
| 25 | FICA | Software | 8 | 0:00:18 | 0:00:11 | 0:00:03 | 0:00:15 | 0:00:12 | 0:00:21 | 0:00:12 | 0:00:11 | 0:00:13 | 0:00:02 | 0 | 10 | 6 | 0:01:58 | 0 |

| 26 | FICA | Software | 7 | 0:00:19 | 0:00:19 | 0:00:04 | 0:00:16 | 0:00:10 | 0:00:19 | 0:00:14 | 0:00:13 | 0:00:11 | 0:00:04 | 0 | 10 | 6 | 0:02:09 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | FICA | Software | 7 | 0:00:21 | 0:00:14 | 0:00:03 | 0:00:17 | 0:00:12 | 0:00:18 | 0:00:13 | 0:00:16 | 0:00:09 | 0:00:03 | 0 | 10 | 6 | 0:02:06 | 0 |
| 28 | FICA | Software | 6 | 0:00:22 | 0:00:13 | 0:00:05 | 0:00:15 | 0:00:11 | 0:00:17 | 0:00:15 | 0:00:14 | 0:00:10 | 0:00:04 | 0 | 10 | 6 | 0:02:06 | 0 |
| 29 | FICA | Software | 6 | 0:00:17 | 0:00:10 | 0:00:03 | 0:00:15 | 0:00:12 | 0:00:19 | 0:00:11 | 0:00:10 | 0:00:12 | 0:00:03 | 0 | 10 | 6 | 0:02:02 | 0 |
| 30 | FICA | Software | 6 | 0:00:16 | 0:00:11 | 0:00:03 | 0:00:14 | 0:00:11 | 0:00:17 | 0:00:12 | 0:00:11 | 0:00:11 | 0:00:05 | 1 | 10 | 6 | 0:01:51 | 1 |
| 31 | FICA | Software | 7 | 0:00:18 | 0:00:13 | 0:00:04 | 0:00:16 | 0:00:10 | 0:00:15 | 0:00:10 | 0:00:09 | 0:00:08 | 0:00:05 | 1 | 10 | 6 | 0:01:48 | 1 |
| 32 | FICA | Software | 6 | 0:00:30 | 0:00:20 | 0:00:20 | 0:00:40 | 0:00:20 | 0:00:40 | 0:01:00 | 0:01:00 | 0:00:50 | 0:00:50 | 1 | 10 | 6 | 0:06:30 | 1 |
| 33 | FICA | Software | 6 | 0:00:19 | 0:00:14 | 0:00:06 | 0:00:18 | 0:00:13 | 0:00:13 | 0:00:07 | 0:00:11 | 0:00:14 | 0:00:06 | 0 | 10 | 6 | 0:02:01 | 0 |
| 34 | FICA | Software | 6 | 0:00:15 | 0:00:11 | 0:00:10 | 0:00:05 | 0:00:16 | 0:00:06 | 0:00:12 | 0:00:04 | 0:00:06 | 0:00:06 | 0 | 10 | 6 | 0:01:31 | 0 |

| 35 | FICA | Software | 7 | 0:00:08 | 0:00:08 | 0:00:05 | 0:00:07 | 0:00:05 | 0:00:06 | 0:00:31 | 0:00:08 | 0:00:29 | 0:00:07 | 0 | 10 | 6 | 0:01:54 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 36 | FICA | Software | 7 | 0:00:15 | 0:00:10 | 0:00:10 | 0:00:05 | 0:00:15 | 0:00:05 | 0:00:10 | 0:00:05 | 0:00:08 | 0:00:08 | 0 | 10 | 6 | 0:01:31 | 0 |
| 37 | FICA | Software | 6 | 0:00:30 | 0:00:12 | 0:00:30 | 0:00:05 | 0:00:27 | 0:01:00 | 0:00:48 | 0:00:31 | 0:00:23 | 0:00:14 | 1 | 10 | 6 | 0:04:40 | 1 |
| 38 | FICA | Software | 8 | 0:01:12 | 0:00:06 | 0:00:03 | 0:00:03 | 0:00:15 | 0:00:20 | 0:00:07 | 0:00:20 | 0:00:20 | 0:00:08 | 0 | 10 | 6 | 0:02:54 | 0 |
| 39 | FICA | Software | 7 | 0:00:40 | 0:00:11 | 0:00:08 | 0:00:06 | 0:01:00 | 0:00:03 | 0:00:20 | 0:00:15 | 0:00:34 | 0:00:27 | 0 | 10 | 6 | 0:03:44 | 0 |
| 40 | FICA | Software | 7 | 0:00:45 | 0:00:26 | 0:00:18 | 0:00:18 | 0:00:38 | 0:00:20 | 0:00:28 | 0:00:30 | 0:00:10 | 0:01:00 | 2 | 10 | 6 | 0:04:53 | 2 |
| 41 | FICA | Software | 6 | 0:00:19 | 0:00:10 | 0:00:02 | 0:00:15 | 0:00:10 | 0:00:05 | 0:00:10 | 0:00:05 | 0:00:10 | 0:00:02 | 1 | 10 | 6 | 0:01:28 | 1 |
| 42 | FICA | Software | 7 | 0:00:40 | 0:00:11 | 0:00:08 | 0:00:06 | 0:01:00 | 0:00:03 | 0:00:20 | 0:00:15 | 0:00:34 | 0:00:27 | 1 | 10 | 6 | 0:03:44 | 1 |
| 43 | FICA | Software | 6 | 0:00:21 | 0:00:11 | 0:00:03 | 0:00:16 | 0:00:12 | 0:00:06 | 0:00:09 | 0:00:09 | 0:00:11 | 0:00:03 | 1 | 10 | 6 | 0:01:41 | 1 |

| | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 44 | FICA | Software | 8 | 0:00:19 | 0:00:12 | 0:00:04 | 0:00:17 | 0:00:14 | 0:00:05 | 0:00:16 | 0:00:11 | 0:00:12 | 0:00:05 | 0 | 10 | 6 | 0:01:55 | 0 |
| 45 | FICA | Software | 7 | 0:00:10 | 0:00:05 | 0:00:05 | 0:00:05 | 0:00:10 | 0:00:05 | 0:00:15 | 0:00:10 | 0:00:10 | 0:00:06 | 2 | 10 | 6 | 0:01:21 | 1 |
| 46 | FICA | Software | 8 | 0:00:50 | 0:00:15 | 0:00:05 | 0:00:10 | 0:00:30 | 0:01:00 | 0:00:50 | 0:00:40 | 0:00:20 | 0:00:25 | 0 | 10 | 6 | 0:05:05 | 1 |

**Anexo E. Tabulación de la encuesta SUS**

| Nro. | Facultad | Carrera | Nivel | P1 | P2 | P3 | P4 | P5 | P6 | P7 | P8 | P9 | P10 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | FICA | Software | 6 | 4 | 4 | 4 | 5 | 2 | 2 | 5 | 2 | 5 | 5 |
| 2 | FICA | Software | 8 | 4 | 1 | 5 | 1 | 5 | 1 | 5 | 1 | 4 | 1 |
| 3 | FICA | Software | 7 | 5 | 2 | 5 | 2 | 5 | 2 | 1 | 2 | 5 | 2 |
| 4 | FICA | Software | 8 | 5 | 1 | 5 | 1 | 5 | 1 | 5 | 1 | 5 | 1 |
| 5 | FICA | Software | 8 | 3 | 2 | 4 | 1 | 4 | 2 | 3 | 2 | 1 | 1 |
| 6 | FICA | Software | 8 | 4 | 2 | 4 | 2 | 4 | 2 | 4 | 1 | 4 | 2 |
| 7 | FICA | Software | 6 | 3 | 1 | 4 | 1 | 5 | 1 | 5 | 1 | 5 | 1 |
| 8 | FICA | Software | 8 | 3 | 1 | 4 | 1 | 5 | 1 | 5 | 1 | 5 | 1 |
| 9 | FICA | Software | 7 | 4 | 2 | 5 | 4 | 4 | 1 | 1 | 1 | 1 | 1 |
| 10 | FICA | Software | 7 | 5 | 2 | 4 | 2 | 5 | 2 | 4 | 2 | 5 | 2 |
| 11 | FICA | Software | 7 | 5 | 1 | 5 | 1 | 4 | 1 | 5 | 1 | 5 | 4 |
| 12 | FICA | Software | 7 | 5 | 1 | 5 | 2 | 5 | 1 | 5 | 1 | 5 | 1 |
| 13 | FICA | Software | 7 | 5 | 2 | 4 | 5 | 5 | 2 | 4 | 2 | 5 | 2 |
| 14 | FICA | Software | 6 | 2 | 1 | 5 | 2 | 4 | 2 | 5 | 1 | 5 | 1 |
| 15 | FICA | Software | 7 | 3 | 4 | 4 | 1 | 5 | 5 | 5 | 1 | 5 | 1 |
| 16 | FICA | Software | 7 | 3 | 3 | 4 | 3 | 4 | 2 | 3 | 2 | 5 | 3 |
| 17 | FICA | Software | 7 | 3 | 4 | 4 | 1 | 5 | 5 | 5 | 1 | 5 | 1 |
| 18 | FICA | Software | 8 | 2 | 1 | 5 | 1 | 5 | 1 | 5 | 1 | 5 | 3 |
| 19 | FICA | Software | 8 | 4 | 2 | 4 | 1 | 5 | 4 | 4 | 2 | 4 | 1 |
| 20 | FICA | Software | 7 | 3 | 4 | 4 | 1 | 5 | 5 | 5 | 1 | 5 | 1 |

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 21 | FICA | Software | 7 | 5 | 1 | 5 | 1 | 5 | 1 | 5 | 1 | 5 | 1 |
| 22 | FICA | Software | 6 | 4 | 1 | 5 | 1 | 5 | 3 | 5 | 1 | 5 | 1 |
| 23 | FICA | Software | 7 | 5 | 1 | 5 | 5 | 5 | 1 | 5 | 1 | 5 | 1 |
| 24 | FICA | Software | 8 | 4 | 1 | 5 | 1 | 5 | 1 | 5 | 1 | 5 | 1 |
| 25 | FICA | Software | 8 | 5 | 1 | 5 | 3 | 5 | 2 | 5 | 1 | 5 | 4 |
| 26 | FICA | Software | 7 | 5 | 1 | 5 | 1 | 5 | 1 | 5 | 1 | 5 | 1 |
| 27 | FICA | Software | 7 | 4 | 1 | 4 | 1 | 5 | 2 | 5 | 1 | 2 | 1 |
| 28 | FICA | Software | 6 | 4 | 4 | 5 | 1 | 5 | 3 | 4 | 1 | 3 | 1 |
| 29 | FICA | Software | 6 | 4 | 1 | 4 | 1 | 2 | 1 | 5 | 1 | 2 | 1 |
| 30 | FICA | Software | 6 | 1 | 3 | 1 | 1 | 3 | 1 | 4 | 2 | 3 | 2 |
| 31 | FICA | Software | 7 | 5 | 1 | 4 | 1 | 5 | 3 | 5 | 1 | 5 | 1 |
| 32 | FICA | Software | 6 | 4 | 3 | 3 | 2 | 3 | 4 | 5 | 1 | 1 | 3 |
| 33 | FICA | Software | 6 | 4 | 2 | 4 | 2 | 4 | 2 | 2 | 2 | 4 | 2 |
| 34 | FICA | Software | 6 | 5 | 2 | 4 | 2 | 4 | 2 | 1 | 2 | 5 | 1 |
| 35 | FICA | Software | 7 | 3 | 5 | 5 | 1 | 5 | 2 | 5 | 1 | 5 | 1 |
| 36 | FICA | Software | 7 | 5 | 3 | 4 | 2 | 4 | 2 | 4 | 2 | 4 | 2 |
| 37 | FICA | Software | 6 | 3 | 1 | 4 | 1 | 4 | 3 | 5 | 1 | 5 | 1 |
| 38 | FICA | Software | 8 | 4 | 3 | 5 | 1 | 5 | 1 | 5 | 1 | 4 | 1 |
| 39 | FICA | Software | 7 | 5 | 2 | 4 | 1 | 5 | 2 | 5 | 2 | 4 | 1 |
| 40 | FICA | Software | 7 | 2 | 2 | 4 | 1 | 4 | 3 | 4 | 2 | 4 | 2 |
| 41 | FICA | Software | 6 | 4 | 2 | 4 | 1 | 5 | 4 | 4 | 2 | 4 | 1 |

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 42 | FICA | Software | 7 | 4 | 1 | 4 | 1 | 5 | 1 | 5 | 1 | 4 | 1 |
| 43 | FICA | Software | 6 | 5 | 1 | 4 | 1 | 5 | 4 | 5 | 1 | 4 | 1 |
| 44 | FICA | Software | 8 | 4 | 1 | 5 | 1 | 5 | 3 | 5 | 1 | 5 | 1 |
| 45 | FICA | Software | 7 | 4 | 1 | 4 | 1 | 1 | 1 | 1 | 3 | 3 | 4 |
| 46 | FICA | Software | 8 | 3 | 3 | 4 | 1 | 2 | 3 | 4 | 3 | 4 | 2 |

## Anexo F. Acta de Entrega de Recepción

**UNIVERSIDAD TÉCNICA DEL NORTE**

Acreditada Resolución Nro. 173-SE-33-CACES-2020

**DIRECCIÓN DE DESARROLLO TECNOLÓGICO E INFORMÁTICO**

**ACTA DE ENTREGA RECEPCIÓN**
**MÓDULO CHAT UTN**

Ibarra, 08 de octubre de 2025

Ingeniero
Juan Carlos García
SUBDIRECTOR DE DESARROLLO TECNOLÓGICO E INFORMÁTICO
Presente.-

Reciba un cordial saludo.
En las instalaciones de la Dirección de Desarrollo Tecnológico e Informático de la Universidad Técnica del Norte, el día **07 de octubre de 2025**, comparecen las partes que suscriben la presente acta para formalizar la entrega-recepción integral y transferencia tecnológica del proyecto de titulación denominado **Red Social de Mensajería (Chat UTN)**, desarrollado como parte del ecosistema UTN Móvil.

### 1. PARTES INTERVINIENTES

| ROL | NOMBRE | CARGO/CALIDAD | DEPENDENCIA |
|---|---|---|---|
| Entrega | Sr. Jorge Rafael Rosero Cuaspud | Estudiante - Ingeniería en Software | FICA |
| Director del Proyecto | Ing. Antonio Quiña | Director de Tesis | FICA |
| Recibe | Ing. Franklin Enríquez | Analista de Sistemas | DDTI |
| Aprobación | Ing. Juan Carlos García | Subdirector | DDTI |

### 2. ANTECEDENTES Y CONTEXTO DEL PROYECTO

El **Módulo Chat UTN** constituye un componente del ecosistema **UTN Móvil**, desarrollado por el estudiante Jorge Rafael Rosero Cuaspud como requisito para la obtención del título de Ingeniero en Software y bajo la dirección del Ing. Antonio Quiña.

El sistema ha sido implementado y se encuentra actualmente en producción en el **Servidor 2** de la arquitectura institucional, proporcionando servicios de mensajería instantánea a la comunidad universitaria.

Dado que el estudiante se encuentra en la fase final de su proceso de titulación, resulta necesaria la transferencia formal y completa del proyecto a la Dirección de Desarrollo Tecnológico e Informático (DDTI) para garantizar la continuidad operativa, el soporte técnico, el mantenimiento y la evolución futura del módulo.

### 3. OBJETO DE LA ENTREGA-RECEPCIÓN

El presente acto administrativo tiene como objeto formalizar la **entrega-recepción integral** del proyecto, abarcando los siguientes componentes:

- ✓ Código fuente completo del sistema.
- ✓ Base de datos (scripts, modelos, diccionarios).
- ✓ Documentación técnica y funcional completa.
- ✓ Manuales de usuario y administración.

- ✓ Transferencia de conocimiento tecnológico (Know-How).
- ✓ Credenciales de acceso, configuraciones y variables de entorno.

## 4. INFORMACIÓN GENERAL DEL PROYECTO

### 4.1. Identificación del Sistema

| ATRIBUTO | DESCRIPCIÓN |
|---|---|
| Nombre del Proyecto | Red Social de Mensajería (Chat UTN) |
| Módulo | Chat UTN |
| Versión | V1.0 |
| Estado | En Producción - Beta |
| Fecha de Despliegue | 02/10/2025 |
| Ubicación | Servidor 2 - Arquitectura Institucional |
| Ambiente | Producción – Alma Linux |
| Propósito | Proveer un sistema de mensajería instantánea institucional para la comunidad universitaria. |

### 4.2. Stack Tecnológico

| COMPONENTE | TECNOLOGÍA | VERSIÓN |
|---|---|---|
| **Backend - Lenguaje** | Node.js | v20.16.0 + |
| **Backend - Framework** | NestJS | ^10.0.0 |
| **Backend - ORM** | TypeORM | ^0.3.20 |
| **Frontend - Framework** | Flutter | 3.29.0 |
| **Frontend - Lenguaje** | Dart | 3.7.0 |
| **Base de Datos Driver** | Oracle Database | ^6.6.0 |
| **Base de Datos (Local Móvil)** | Isar | 6.0.0 |

| | | |
|---|---|---|
| **Sistema Operativo - Servidores** | AlmaLinux | 9.x |
| **Contenedor** | Docker | 28.2.2 |
| **Notificaciones Push** | Firebase Cloud Messaging | ^15.1.1 |
| **Herramienta Administrativa** | Oracle APEX Desarrollo* | Aplicación 206 |

## 5. ALCANCE DE LA ENTREGA

### 5.1. Código Fuente

| COMPONENTE | DESCRIPCIÓN | REPOSITORIO | ESTADO DE ENTREGA |
|---|---|---|---|
| **Backend (Microservicio)** | Código fuente del microservicio de Red Social de Mensajería desarrollado en NestJS | https://github.com/UTN-MOVIL/utn-movil-red-social.git | ✓ Entregado |
| **API Gateway** | Código fuente del API Gateway con Traefik | https://github.com/UTN-MOVIL/utn-movil-api-gateway | ✓ Entregado |
| **Aplicación Móvil** | Código fuente (módulo Chat UTN) de la aplicación móvil desarrollada en Flutter | https://github.com/UTN-MOVIL/utn_movil_app.git | ✓ Entregado |
| **Base de Datos** | Scripts SQL, diagramas y modelos de datos (Oracle e Isar) | https://github.com/DevJorgeRafael/base-de-datos-chat-utn.git | ✓ Entregado |

### 5.2. Base de Datos

El sistema utiliza dos modelos de base de datos complementarios: **Oracle Database** para el backend y **Isar** (base de datos NoSQL local) para la aplicación móvil Flutter.

A. Base de Datos Oracle (Backend)

Información General:

| ATRIBUTO | DESCRIPCIÓN |
|---|---|
| Motor | Oracle Database |
| Esquema | MOVIL_UTN |
| Driver/Cliente | Oracle Instant Client 23 |
| Convención de Nombres | Prefijo MVL_ para todas las entidades |
| Tipos de Objetos | Tablas (MVL_TAB_), Vistas (MVL_VIEW_), Procedimientos (MVL_PROC_) |

B. Base de Datos Isar (Aplicación Móvil - Flutter)

**Información General:**

| ATRIBUTO | DESCRIPCIÓN |
|---|---|
| Motor | Isar Database (NoSQL) |
| Versión | 6.0.0 |
| Tipo | Base de datos local embebida |
| Plataforma | Flutter (multiplataforma) |
| Formato | Colecciones (equivalente a tablas) |

### 5.3. Documentación Técnica

La documentación del proyecto del estudiantes consta la siguiente lista de documentos:

- ✓ Manual de Usuario - Red Social de Mensajería (Chat UTN)
- ✓ Manual Técnico - Red Social de Mensajería (Chat UTN)
- ✓ Diagramas y Scripts de Base de Datos - Red Social de Mensajería (Chat UTN)
- ✓ Arquitectura y Configuración de Servidores - Red Social de Mensajería (Chat UTN)
- ✓ Manual de Instalación y Despliegue - Red Social de Mensajería (Chat UTN)
- ✓ Plantilla de Requisitos de Software - Red Social de Mensajería (Chat UTN)

## 6. PROCESO DE TRANSFERENCIA TECNOLÓGICA

### 6.1. Sesión de Transferencia

La transferencia tecnológica se llevó a cabo el día **07 de octubre de 2025** en las oficinas de la DDTI, mediante una reunión técnica que cubrió las siguientes fases:

- **Fase 1: Presentación General del Sistema:** Contexto, objetivos, arquitectura general y funcionalidades principales.
- **Fase 2: Revisión Técnica Detallada:** Estructura del código fuente (backend y frontend), modelo de datos, integraciones y patrones de diseño.
- **Fase 3: Aspectos Operativos:** Configuración en Servidor 2, procedimientos de despliegue, gestión de logs y respaldos.
- **Fase 4: Entrega de Materiales:** Entrega digital de toda la documentación, transferencia de credenciales y verificación de la completitud de los materiales.

### 6.2. Nivel de Transferencia Alcanzado

La transferencia tecnológica se ha completado de manera **INTEGRAL Y SATISFACTORIA**, logrando una comprensión completa de la arquitectura, los componentes y los procedimientos operativos del sistema, quedando habilitado para su gestión y mantenimiento.

## 7. COMPROMISOS DE LAS PARTES

### 7.1. Compromisos del Estudiante (Quien Entrega)

El Sr. Jorge Rafael Rosero Cuaspud se compromete a:

1. Haber entregado la totalidad del código fuente, documentación y artefactos del proyecto.
2. Garantizar que el código entregado corresponde a la última versión estable en producción.
3. Estar disponible para consultas técnicas puntuales durante un período de **30 días calendario** posterior a la firma del acta.
4. Proporcionar aclaraciones adicionales sobre aspectos específicos del desarrollo si son requeridas.

### 7.2. Compromisos de DDTI

La Dirección de Desarrollo Tecnológico e Informático, representada por el Ing. Franklin Enríquez, se compromete a:

1. Resguardar con la debida confidencialidad toda la documentación, código fuente y credenciales entregadas.
2. Asumir la responsabilidad de la continuidad operativa y el ciclo de vida del Módulo Chat UTN.
3. Realizar el soporte técnico y mantenimiento evolutivo del sistema.
4. Utilizar los materiales entregados exclusivamente para los fines institucionales de la Universidad Técnica del Norte.

## 8. OBSERVACIONES Y RECOMENDACIONES

- **Observaciones:** El Módulo Chat UTN se recibe en estado operativo y en producción en el Servidor 2, sin incidencias críticas pendientes.

- **Recomendaciones:** Se recomienda planificar auditorías de seguridad periódicas, mantener actualizadas las dependencias tecnológicas y establecer una política de respaldos automatizados.

## 9. DECLARACIÓN

Las partes que suscriben la presente acta declaran que la entrega-recepción del proyecto **Módulo Chat UTN** se ha realizado de manera **COMPLETA, CONFORME Y SATISFACTORIA**. No existen pendientes que impidan la formalización de este acto.

## 10. CONSTANCIA Y FIRMAS

En constancia de aceptación y conformidad con todo lo establecido, suscriben los comparecientes en cuatro ejemplares de igual tenor y valor legal.

| | | |
|---|---|---|
| **ENTREGA DEL PROYECTO** | **Jorge Rafael Rosero Cuaspud**<br>*Estudiante* | Firmado electrónicamente por:<br>JORGE RAFAEL ROSERO CUASPUD<br>Validar únicamente con FirmaEC |
| **DIRECCIÓN DEL PROYECTO** | **Antonio Quiña**<br>Director del Proyecto de Titulación | JOSE ANTONIO QUIÑA MERA<br>Firmado digitalmente por JOSE ANTONIO QUIÑA MERA<br>Nombre de reconocimiento (DN): c=EC, sn=QUIÑA MERA, givenName=JOSE ANTONIO, serialNumber=IDCEC-1002322384, cn=JOSE ANTONIO QUIÑA MERA, 2.5.4.97=TINEC-1002322384001 |
| **RECEPCIÓN TÉCNICA** | **Franklin Enríquez**<br>Analista de Sistemas DDTI | Firmado electrónicamente por:<br>FRANKLIN LEANDRO ENRIQUEZ ROSERO<br>Validar únicamente con FirmaEC |
| **APROBACIÓN** | **Juan Carlo García**<br>Subdirector DDTI | Firmado electrónicamente por:<br>JUAN CARLOS GARCIA PINCHAO<br>Validar únicamente con FirmaEC |