

[Integrantes](#)

[Redacción del levantamiento de Requerimientos](#)

[Objetivos del proyecto](#)

[Funcionalidades identificadas](#)

[Gestión de cursos \(para profesores\)](#)

[Acceso y registro \(para estudiantes\)](#)

[Mensajería en tiempo real](#)

[Roles y permisos](#)

[Seguridad y privacidad](#)

[Adicionales](#)

[Punto 3: Análisis de requerimientos con el método MoSCoW](#)

[Punto 4: Análisis gestión de software](#)

[Punto 8: Diseño y arquitectura](#)

[Punto 9: Patrones de diseño](#)

[Patrón Singleton](#)

[Patrón Observer](#)

[Patrón Repository](#)

Integrantes

- Daniel Alejandro Duitama Correa (dduitama@unal.edu.co)
- Edwin Felipe Pinilla Peralta (epinillap@unal.edu.co)
- Miguel Angel Martinez Fernandez (miamartinezfe@unal.edu.co)
- Juan Sebastián Umaña Camacho (juumanac@unal.edu.co)

Redacción del levantamiento de Requerimientos

Redacción del levantamiento de requerimientos

La idea del sistema de mensajería instantánea "**UNión**" surge de la necesidad de mejorar la comunicación académica, **creando cursos para organizar la interacción entre estudiantes y profesores** dentro del ámbito universitario. Actualmente, la comunicación académica se encuentra dispersa en múltiples plataformas como correos electrónicos, redes sociales y Classroom, lo que dificulta la organización y el acceso a la información relevante. Además, los estudiantes enfrentan problemas para colaborar de manera efectiva con sus compañeros y resolver dudas con sus profesores en un entorno formal.

Para abordar estas dificultades, se propuso una plataforma que permita **crear, editar y eliminar cursos** según las necesidades académicas, además de **agregar estudiantes a los cursos mediante invitaciones o códigos de acceso**. Esto garantizará que cada usuario tenga acceso solo a los cursos en los que está inscrito y que los profesores puedan **gestionar la comunicación con sus estudiantes de manera eficiente**.

El sistema también contará con **mensajería en tiempo real**, permitiendo a los usuarios **buscar a otros estudiantes o profesores mediante un buscador con filtros por nombre o correo institucional**. Para iniciar una conversación con otro usuario, será necesario **enviar una solicitud de amistad**, la cual podrá ser aceptada o rechazada. Una vez conectados, podrán **enviar mensajes dentro del curso, crear hilos de discusión organizados por temas y recibir notificaciones por correo de mensajes nuevos y anuncios importantes**.

Además, se ofrecerán herramientas para mejorar la accesibilidad y la organización de la información, como un **panel centralizado donde se puedan visualizar todas las conversaciones** y un **filtro para encontrar rápidamente chats individuales o grupales**. También se incluirá la opción de **crear encuestas con fecha límite** para fomentar la participación y la toma de decisiones en el entorno académico.

Para garantizar una experiencia segura y ordenada, el sistema establecerá **roles y permisos específicos para profesores y estudiantes**. Los profesores podrán **gestionar los cursos y moderar los mensajes**, mientras que los estudiantes tendrán acceso restringido únicamente a sus cursos y conversaciones correspondientes. Asimismo, se implementarán medidas de **seguridad y privacidad** para asegurar que solo los usuarios autorizados puedan acceder a la información del curso, protegiendo los datos mediante protocolos de autenticación segura.

Finalmente, se incluirá un **módulo para subir archivos relevantes al curso**, permitiendo tanto a estudiantes como a profesores compartir material de apoyo de manera sencilla.

Objetivos del proyecto

El equipo espera que este desarrollo le permita adquirir experiencia en el diseño y construcción de sistemas seguros y escalables, aplicando conceptos avanzados como **WebSockets, clean code y patrones de diseño**. Además, busca fortalecer su perfil profesional al enfrentarse a la **resolución de problemáticas reales en entornos académicos**.

Para los usuarios buscamos cubrir estas expectativas

- Profesores:
 - Un sistema fácil de usar para gestionar cursos y estudiantes.
 - Garantía de que solo los estudiantes autorizados accedan a la comunicación del curso.
 - Herramientas para mantener la privacidad y la organización de la mensajería.
- Estudiantes:
 - Acceso rápido a los cursos y sus comunicaciones asociadas.
 - Un entorno seguro para interactuar con compañeros y profesores.
 - Funcionalidades como notificaciones en tiempo real y búsqueda de mensajes.

Funcionalidades identificadas

Gestión de cursos (para profesores)

1. Crear un curso para organizar la comunicación de sus estudiantes. **(GC-01)**
2. Editar la información del curso, como nombre y descripción. **(GC-02)**
3. Eliminar cursos obsoletos o que ya no son necesarios. **(GC-03)**
4. Agregar estudiantes al curso mediante invitaciones o códigos de acceso. **(GC-04)**
5. Un módulo para subir archivos pertinentes al curso. **(GC-05)**

Acceso y registro (para estudiantes)

6. Unirse a un curso al que fueron invitados utilizando un código único. **(AR-01)**
7. Acceder sólo a los cursos en los que están inscritos. **(AR-02)**

Mensajería en tiempo real

8. Los usuarios pueden buscar a otros estudiantes o profesores a través de un buscador que permite filtrar por nombre o correo institucional. Si desean iniciar un chat con una persona específica, deben enviarle una solicitud de amistad. La otra persona puede aceptar o rechazar la solicitud. **(MT-01)**
9. Enviar mensajes entre estudiantes y profesores dentro del curso. **(MT-02)**
10. Recibir notificaciones de mensajes nuevos y anuncios importantes. **(MT-03)**
11. Crear hilos de discusión organizados por temas dentro del curso. **(MT-04)**
12. Creación de encuestas con fecha límite. **(MT-05)**
13. Panel en donde se vean todas mis conversaciones, y filtro para encontrar rápidamente conversaciones individuales o grupales **(MT-06)**
14. Los usuarios pueden decidir qué tipo de notificaciones desean recibir. **(MT-07)**.

15. Búsqueda avanzada (por ejemplo, por palabras clave o fechas).(MT-08)

Roles y permisos

16. Definir roles específicos para profesores y estudiantes, asegurando que cada uno tenga acceso únicamente a las funcionalidades necesarias. **(RP-01)**.
- a. Profesores: gestión de cursos y moderación de mensajes.
 - b. Estudiantes: acceso a cursos y comunicación.

Seguridad y privacidad

17. Garantizar que solo usuarios autorizados puedan acceder a la información del curso. **(SP-01)**
 18. Proteger los datos de los usuarios mediante protocolos de autenticación segura. **(SP-02)**
 19. Autenticación biométrica (huella digital o reconocimiento facial). **(SP-03)**

Adicionales

20. Módulo para coordinar transporte compartido entre usuarios del sistema. **(AD-01)**
 21. Integración con sistemas externos (por ejemplo, plataformas de gestión educativa como Moodle). **(AD-02)**

Punto 3: Análisis de requerimientos con el método MoSCoW

Requerimiento	Importancia	Estimación de esfuerzo	Justificación
Enviar mensajes entre estudiantes y profesores dentro del curso. (MT-02)	Must Have	5	La comunicación directa entre estudiantes y profesores es fundamental para el aprendizaje y la resolución de dudas. Implementar esta funcionalidad requiere integración de sistemas de mensajería en tiempo real, lo que puede ser complejo pero es esencial para la experiencia del usuario.
Recibir notificaciones de mensajes nuevos y anuncios importantes. (MT-03)	Must Have	3	Las notificaciones aseguran que los usuarios estén al tanto de mensajes y anuncios críticos. Es crucial para mantener la comunicación efectiva y la participación en el curso.
Crear hilos de discusión organizados por temas dentro del curso. (MT-04)	Must Have	3	Los hilos de discusión organizados facilitan la participación y el seguimiento de conversaciones específicas. Esta funcionalidad mejora la organización del curso y la experiencia del usuario.

Panel en donde se vean todas mis conversaciones, y filtro para encontrar rápidamente conversaciones individuales o grupales. (MT-06)	Must Have	3	Un panel centralizado de conversaciones con filtros mejora la usabilidad y la eficiencia en la gestión de mensajes. Además, ayuda a que la aplicación esté más organizada.
Definir roles específicos para profesores y estudiantes, asegurando que cada uno tenga acceso únicamente a las funcionalidades necesarias. (RP-01)	Must Have	5	La definición de roles es crucial para la seguridad y la funcionalidad del sistema. Aunque es complejo de implementar, garantiza que cada usuario tenga acceso adecuado y protege la integridad del curso.
Garantizar que solo usuarios autorizados puedan acceder a la información del curso. (SP-01)	Must Have	5	La seguridad de la información es fundamental. Implementar autenticación y autorización robustas es esencial para proteger los datos del curso y cumplir con estándares de privacidad.
Crear un curso para organizar la comunicación de sus estudiantes. (GC-01)	Must Have	8	La creación de cursos es la base de la plataforma. Aunque es complejo debido a la necesidad de integración con múltiples funcionalidades, es esencial para la organización y gestión efectiva de los estudiantes.
Proteger los datos de los usuarios mediante protocolos de autenticación segura. (SP-02)	Must Have	3	La protección de datos es esencial para la confianza del usuario. Implementar protocolos de autenticación segura, es crucial para la seguridad y privacidad de los usuarios.
Editar la información del curso, como nombre y descripción. (GC-02)	Could Have	3	Permitir la edición de la información del curso ofrece flexibilidad a los profesores. Aunque no es crítico, mejora la usabilidad y la personalización del curso.
Agregar estudiantes al curso mediante códigos de acceso. (GC-04)	Could Have	5	La inclusión de estudiantes mediante códigos de acceso facilita la gestión del curso. Aunque no es crítico, ya que los estudiantes se pueden unir vía invitaciones por correo, este requerimiento haría que la app fuera más fácil de usar en general.

Creación de encuestas con fecha límite. (MT-05)	Could Have	2	Las encuestas con fecha límite son útiles para recopilar feedback. Su implementación es relativamente sencilla y añade valor a la interacción dentro del curso.
Acceder sólo a los cursos en los que están inscritos. (AR-02)	Could Have	3	Restringir el acceso a cursos específicos mejora la seguridad y la organización. Es importante para mantener la privacidad de los cursos.
Los usuarios pueden buscar a otros a través de un buscador. Si desean iniciar un chat con una persona específica, deben enviarle una solicitud de amistad. (MT-01)	Could Have	5	La búsqueda de usuarios y la solicitud de amistad facilita la interacción entre usuarios. Aunque puede ser algo complejo de implementar, mejora la conectividad y la experiencia social dentro de la plataforma.
Eliminar cursos obsoletos o que ya no son necesarios. (GC-03)	Should Have	2	La eliminación de cursos obsoletos ayuda a mantener la plataforma organizada. No es una funcionalidad crítica pero es relativamente sencilla de implementar.
Los usuarios pueden decidir qué tipo de notificaciones desean recibir. (MT-07)	Wont Have	5	Permitir que los usuarios personalicen las notificaciones que reciben mejora la experiencia del usuario al reducir la fatiga de notificaciones no deseadas. Es importante para la satisfacción del usuario y la usabilidad de la aplicación pero no es esencial.
Módulo para coordinar transporte compartido entre usuarios de la aplicación. (AD-01)	Won't Have	8	Aunque la coordinación de transporte compartido podría ser una característica interesante, su implementación es compleja y requiere integración con varios sistemas adicionales. Dado que no es esencial para los objetivos principales de la aplicación, se decide no incluirlo en esta fase.
Integración con sistemas externos (por ejemplo, plataformas de gestión educativa como Moodle). (AD-02)	Won't Have	5	Aunque la integración con plataformas externas como Moodle podría mejorar la interoperabilidad, su implementación es compleja y requiere recursos significativos para garantizar la compatibilidad con diferentes APIs. Dado que no es esencial para los objetivos

			principales de la plataforma, se decide no incluirlo en esta fase..
Autenticación biométrica (huella digital o reconocimiento facial). (SP-03)	Won't Have	3	La autenticación biométrica ofrece un nivel adicional de seguridad y conveniencia, dado que no es una necesidad crítica para la mayoría de los usuarios y añadiría complejidad innecesaria, se decide no incluirla en esta versión.
Búsqueda avanzada (por ejemplo, por palabras clave o fechas). (MT-08)	Won't Have	3	Aunque la búsqueda avanzada podría mejorar la usabilidad, su implementación requiere un diseño de base de datos optimizado y algoritmos de búsqueda eficientes, lo que añadiría complejidad al sistema. Dado que las necesidades actuales de búsqueda son cubiertas por funcionalidades más simples, se decide no priorizar esta característica en esta fase.

Punto 4: Análisis gestión de software

1. Tiempo estimado

La estimación del tiempo se ha realizado tomando en cuenta la fecha máxima de entrega del proyecto y el número de desarrolladores participantes (4).

Módulo	Funcionalidades	Tiempo Estimado (Días)	Objetivos Principales
Gestión de Cursos	<ul style="list-style-type: none"> GC-01 GC-02 GC-03 GC-04 GC-05 AR-02 	10 -15	<ul style="list-style-type: none"> Crear, editar y eliminar cursos. Invitar estudiantes mediante códigos de acceso.
Comunicación y Mensajería	<ul style="list-style-type: none"> MT-02 MT-03 MT-04 MT-05 	8 -10	<ul style="list-style-type: none"> Mensajería en tiempo real entre estudiantes y profesores. Notificaciones de mensajes y anuncios. Creación de encuestas con fecha límite.
Gestión de Usuarios y Seguridad	<ul style="list-style-type: none"> SP-01 SP-02 RP-01 	8 -10	<ul style="list-style-type: none"> Roles de usuario (profesor y estudiante). Autenticación segura (por ejemplo, OAuth)
Módulo Búsqueda y Organización de Conversaciones	<ul style="list-style-type: none"> MT-01 MT-06 	2-3	<ul style="list-style-type: none"> Búsqueda de usuarios y conversaciones. Panel de conversaciones con filtros.

2. Costo Estimado

La estimación del costo se realizó tomando como referencia los sueldos y valores en Colombia y con pesos colombianos (COP).

Fuentes:

- <https://co.computrabajo.com/salarios/programador-junior>

- <https://platzi.com/tutoriales/3208-programacion-basica/24348-descubre-cuanto-gana-un-programador-en-mexico-colombia-argentina-y-ee-uu-en-2023/>
- <https://co.talent.com/salary?job=desarrollador+junior>

2a.Sueldos

Rol	Salario Mensual (COP - Aprox) (Promedio)
Desarrollador Frontend Junior	\$2.000.000
Desarrollador Frontend Senior	\$4.000.000
Desarrollador Backend Junior	\$2.500.000
Desarrollador Backend Senior	\$5.500.000
Desarrollador Full-Stack Senior	\$7.300.000
Diseñador UX/UI	\$3.000.000

2b.Servicios en la nube

La estimación de los servicios en la nube se realiza en base a unos 4000 usuarios con un máximo de 25000 y con las funcionalidades esenciales que requiere la aplicación.

Fuentes:

- <https://calculator.aws/#/>
- <https://vercel.com/docs/pricing>
- <https://auth0.com/pricing>

Servicio	Costo Mensual (COP - Aprox) (Promedio)	Costo Mensual (USD)(Promedio)
Amazon Web Services (AWS)	\$250.000 - \$750.000	\$60-\$180
Vercel	\$270.000 - \$830.000	\$65-\$200
Auth0	\$0 (Free Tier - 25000 usuarios activos)	\$0

2c.Herramientas Externas

Las únicas herramientas externas que se tiene previsto poder emplear son IDEs para los lenguajes de programación TypeScript y Java.

Fuentes:

- <https://www.jetbrains.com/idea/buy/?section=commercial&billing=monthly>

IDE	Costo Mensual (COP - Aprox)	Costo Mensual (USD)
Visual Studio Code	0\$ (Gratuita)	0\$
IntelliJ IDEA Ultimate (Organizaciones)	\$250.000	60\$

Teniendo las estimaciones realizadas en cuenta podemos estimar el costo de desarrollo en relación al tiempo, a continuación se reflejan los resultados para este proyecto:

Estimación total del proyecto			
Estimación de Sueldos			
Rol	Costo Mensual (COP-Aprox)	Número de desarrolladores	Total
Desarrollador Junior Frontend	\$2.000.000	2	\$4.000.000
Desarrollador Junior Backend	\$2.500.000	2	\$5.000.000
Estimación de servicios en la nube			
Servicio	Costo Mensual (COP-Aprox)	Número de servicios a solicitar	Total
Amazon Web Services (AWS)	\$250.000	1	\$250.000
Auth0	\$0 (Free Tier)	1	\$0
Estimación de herramientas Externas			
Servicio	Costo Mensual (COP-Aprox)	Número de licencias a solicitar	Total
Visual Studio Code	0\$ (Gratuita)	2	0\$
IntelliJ IDEA	\$250.000	2	\$500.000

Ultimate (Organizaciones)			
Total Mensual			
9.750.000			

3.Alcance

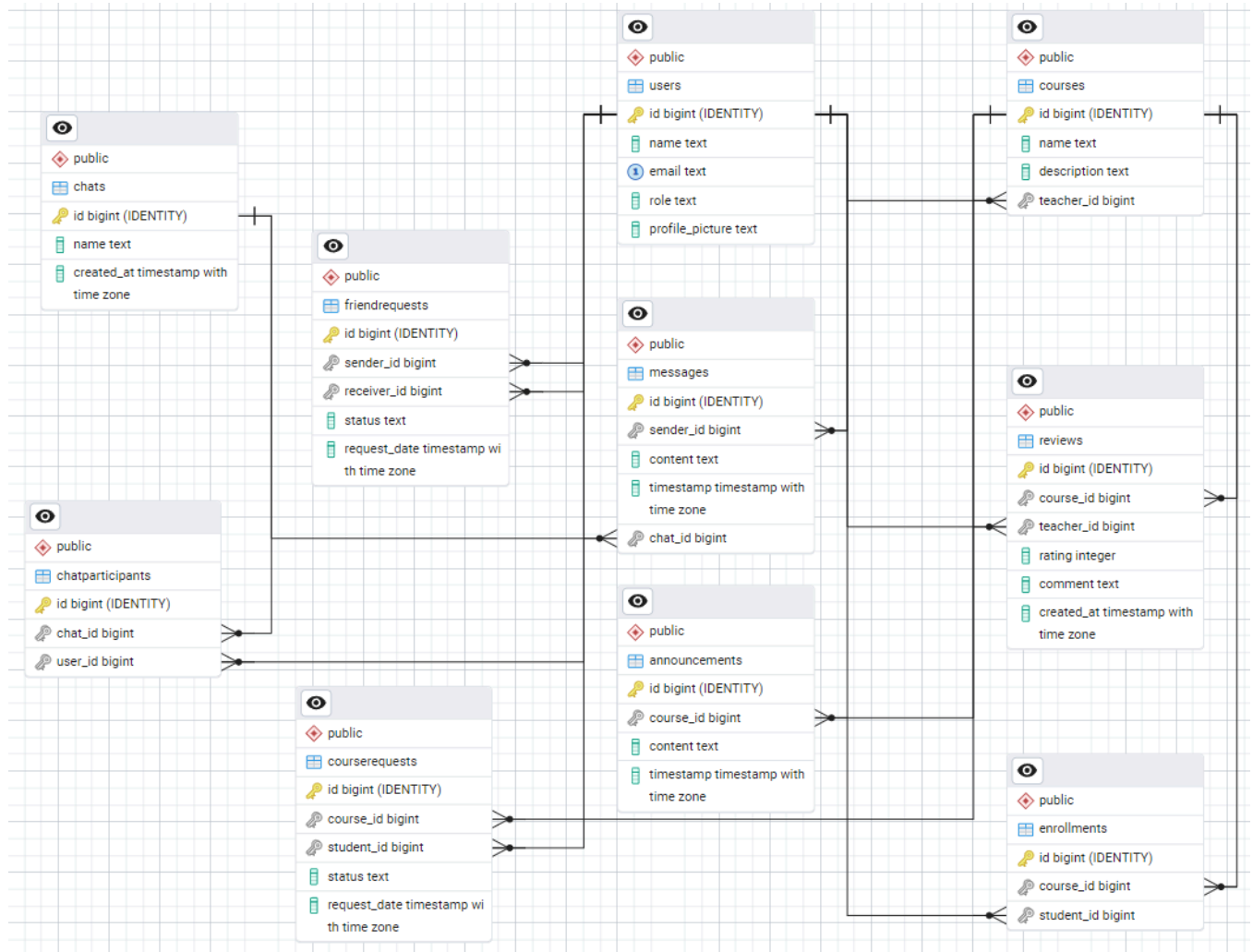
Funcionalidades incluidas en el MVP	
Módulo	Funcionalidad
Gestión de Cursos	<ul style="list-style-type: none"> • Crear, editar y eliminar cursos. (GC-01) (GC-02) (GC-03) • Invitar estudiantes mediante códigos de acceso. (GC-04) • Acceso restringido a cursos inscritos. (AR-02)
Comunicación y Mensajería	<ul style="list-style-type: none"> • Mensajería en tiempo real entre estudiantes y profesores. (MT-02) • Notificaciones de mensajes y anuncios. (MT-03) • Hilos de discusión organizados por temas. (MT-04) • Creación de encuestas con fecha límite. (MT-05)
Gestión de Usuarios y Seguridad	<ul style="list-style-type: none"> • Roles de usuario (profesor y estudiante). (RP-01) • Autenticación segura (por ejemplo, OAuth) (SP-02) • Control de acceso a la información del curso. (SP-01)
Módulo Búsqueda y Organización de Conversaciones	<ul style="list-style-type: none"> • Búsqueda de usuarios y conversaciones. (MT-01) • Panel de conversaciones con filtros. (MT-06)

Funcionalidades excluidas en el MVP
--

Módulo	Funcionalidad
Gestión de Cursos	<ul style="list-style-type: none">Integración con sistemas externos (por ejemplo, plataformas de gestión educativa como Moodle).(AD-02)
Comunicación y Mensajería	<ul style="list-style-type: none">Los usuarios pueden decidir qué tipo de notificaciones desean recibir.(MT-07)
Gestión de Usuarios y Seguridad	<ul style="list-style-type: none">Autenticación biométrica (huella digital o reconocimiento facial).(SP-03)
Módulo Búsqueda y Organización de Conversaciones	<ul style="list-style-type: none">Búsqueda avanzada (por ejemplo, por palabras clave o fechas).(MT-08)

Punto 8: Diseño y arquitectura

Diagrama ER base de datos:



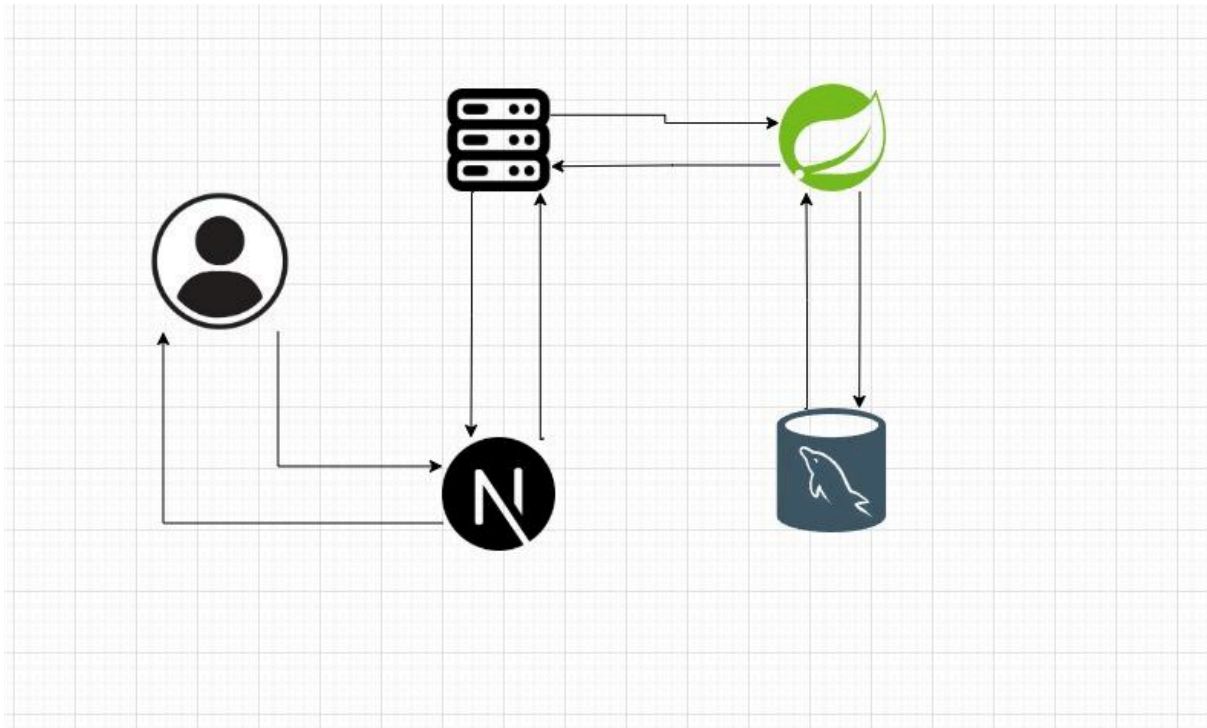
Justificación Diagrama ER base de datos:

La entidad “Enrollments” fue creada para manejar la cardinalidad muchos - muchos entre las entidades courses y announcements. Lo mismo con “chatparticipants” para las tablas chats y users.

El modelo sigue la **tercera forma normal (3NF)**, lo cual es ideal para evitar redundancias y garantizar la integridad de los datos. Por ejemplo: No se almacena información del profesor dentro de la tabla de cursos, sino que se usa teacher_id para referenciar a users.

Cada tabla tiene una clave primaria (id bigint (IDENTITY)) y referencias a otras tablas mediante claves foráneas.

Aunque las bases de datos no relacionales son ideales para manejo de datos en grandes volúmenes, y consultas rápidas para aplicaciones tipo mensajería, se decidió usar una base de datos relacional porque no se va a usar grandes cantidades de datos, además de que todo el equipo ya cuenta con la experiencia de manejar una.



Justificación Diagrama arquitectura de nube:

Se planea un acceso desde la web al usuario, usaremos la tecnología de Next.JS pues la mayoría del equipo cuenta con experiencia en el framework, es fácilmente escalable, es nuevo y es parecido a una tecnología muy popular, React.js.

El servidor será desplegado en AWS, específicamente en su capa gratuita pues cumple con nuestras necesidades. Este servidor contendrá el backend creado en la tecnología de Spring Boot, pues java es bastante sólido, fácilmente escalable y tiene una alta demanda y oferta en el mercado laboral, lo que podría permitir incluso una rápida ampliación del equipo y nos da una experiencia laboral muy importante.

La base de datos será en MySQL pues es fácilmente escalable, conocemos la tecnología y la hemos trabajado desde antes en backend.

Punto 9: Patrones de diseño

Patrón Singleton

Para WebSocket Handler y servicios críticos.

Se utilizarán **WebSockets** en la aplicación porque permiten una comunicación **bidireccional y en tiempo real** entre el servidor y los clientes. Esto es clave en funcionalidades como:

Mensajería en tiempo real:

- Permite que los mensajes enviados en los chats aparezcan **instantáneamente** sin necesidad de recargar la página o hacer peticiones constantes al servidor.
- Un enfoque tradicional con HTTP (polling) sería ineficiente, generando más carga en el servidor.


Notificaciones en vivo:

- Se pueden enviar notificaciones inmediatas a los usuarios cuando reciben una nueva solicitud de amistad, una inscripción en un curso o un mensaje en un chat.

El patrón **Singleton** asegura que una clase tenga una única instancia en toda la aplicación y proporciona un punto de acceso global a esa instancia. En este caso, evita la creación de múltiples manejadores de WebSockets o instancias duplicadas de servicios críticos, lo que podría generar conflictos o un uso ineficiente de recursos, asegurando que todas las conexiones sean gestionadas de manera centralizada y eficiente.

- Asegura que servicios críticos, como la **gestión de sesiones**, mantengan un único punto de acceso y no haya inconsistencias entre diferentes instancias.
- Mejora la escalabilidad, ya que una sola instancia puede gestionar múltiples conexiones sin sobrecargar el sistema.

Implementación:

 <i>WebSocketHandler</i>
- instance: WebSocketHandler
- WebSocketHandler() + getInstance(): WebSocketHandler + handleMessage(message: String)

El constructor es privado para evitar la creación de múltiples instancias.

El método getInstance() garantiza que solo exista una instancia del manejador de WebSockets.

El método handleMessage() gestiona los mensajes recibidos a través de WebSocket.

Patrón Observer

Para notificar eventos en tiempo real, para que los clientes (estudiantes/profesores) sean notificados automáticamente cuando hay nuevos mensajes o cambios en el estado de los chats.

Permite que múltiples partes del sistema sean notificadas automáticamente cuando ocurre un evento sin necesidad de consulta constante. Es ideal para la actualización en tiempo real de los chats cuando un usuario recibe un nuevo mensaje o cuando hay un cambio en el estado del chat (por ejemplo, que un usuario esté escribiendo).

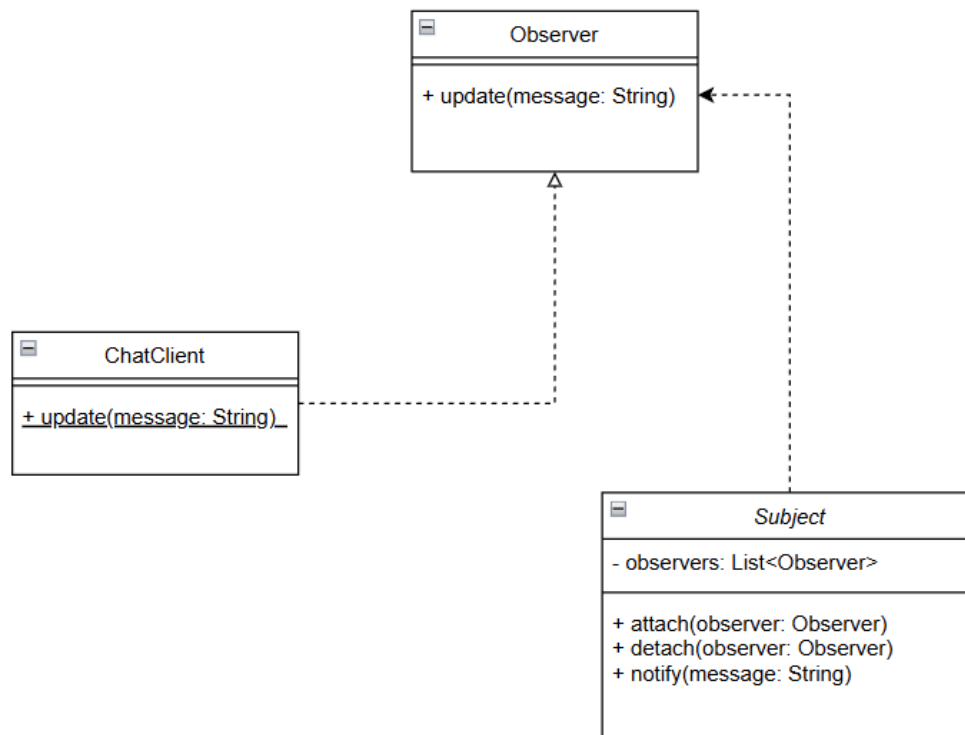
- Queremos que los estudiantes y profesores sean notificados en tiempo real cuando reciben un nuevo mensaje o cuando hay actualizaciones en los chats.
- Evita que los usuarios tengan que hacer peticiones constantes al servidor para verificar cambios, lo que optimiza el rendimiento.
- Facilita la integración con tecnologías como **WebSockets**..

Implementación:

“Subject” es la clase que mantiene una lista de observadores (observers) y notifica cambios.

“Observer” es una interfaz que define el método update().

“ChatClient” es una implementación concreta de Observer que recibe notificaciones cuando hay nuevos mensajes.



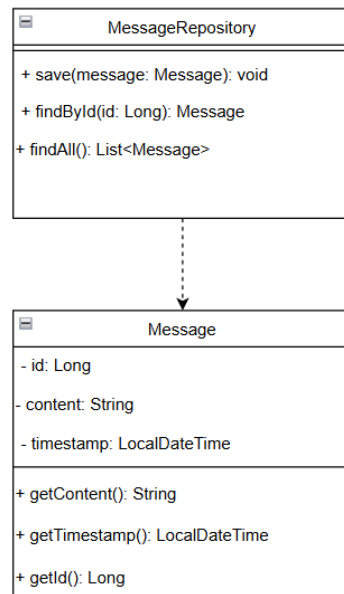
Patrón Repository

Para la capa de persistencia.

Separa la lógica de acceso a la base de datos del resto de la aplicación, lo que facilita el mantenimiento y la prueba del código. Evita que los controladores y servicios accedan directamente a la base de datos.

- Nos permite manejar la persistencia de los mensajes y usuarios sin acoplar el código a una base de datos específica.
- Facilita el cambio de la base de datos en el futuro sin modificar la lógica de negocio.
- Mejora la organización del código y permite realizar pruebas unitarias sin necesidad de acceder a la base de datos real.

Implementación:



“MessageRepository” es una interfaz que define los métodos para acceder a la base de datos.

“Message” es una entidad que representa un mensaje en el sistema.