Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

buscar nuevamente.

Flujo Alternativo 2: El usuario ya ha enviado una solicitud a esta persona

5a. Si el usuario ya ha enviado previamente una solicitud de amistad a la misma persona y esta aún no ha sido aceptada o rechazada, el sistema muestra un mensaje informativo indicando que la solicitud está pendiente.

Flujo Alternativo 3: El destinatario rechaza la solicitud

7a. Si el destinatario rechaza la solicitud, el sistema notifica al solicitante y no habilita el chat.

Historias de Usuario

Integrantes

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

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

Levantamiento de Requerimientos

1. ¿De dónde surge la idea?

La idea del sistema de mensajería instantánea "UNión" surge de la necesidad de mejorar la comunicación académica entre estudiantes y profesores dentro de los cursos universitarios. Durante el desarrollo del proyecto, se identificó que las plataformas actuales carecen de una integración enfocada en la facilidad para comunicarse y organización específica por curso. La propuesta se basó en entrevistas con estudiantes, así como observación directa de los problemas que enfrentan en el ámbito académico, como la dispersión de información en múltiples canales y la falta de orden.

2. ¿Cómo se pusieron de acuerdo para elegir la idea?

El equipo realizó una lluvia de ideas en la que cada miembro propuso un problema académico relevante (transporte, comunicación). Tras discutir las propuestas, se optó por desarrollar un sistema de mensajería por ser una solución práctica y de impacto inmediato donde se pueden utilizar los conocimientos adquiridos durante el curso.

3. ¿Cuáles son las problemáticas que buscan resolver?

- Falta de un canal sencillo y centralizado: Actualmente, la comunicación académica suele dispersarse entre plataformas como correos electrónicos, redes sociales y classroom, lo que dificulta la organización y privacidad, por eso se busca una aplicación web centralizada donde se trabajen estas problemáticas.
- Dificultades en la colaboración académica: Los estudiantes tienen problemas con canales de comunicación poco efectivos para interactuar y resolver dudas directamente con sus compañeros o profesores en un entorno formal, por lo qué una plataforma virtual donde cada estudiante esté inscrito automáticamente facilita esta y se promueva la participación con reviews.

4. ¿Qué expectativas tienen los usuarios potenciales del sistema?
Los usuarios esperan:

- 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.

5. ¿Qué beneficios esperan ustedes al desarrollar este proyecto? El equipo espera adquirir experiencia práctica en:

- Poner en práctica el diseño y desarrollo de sistemas seguros y escalables.
- Aplicación de conceptos avanzados como websockets, clean code y patrones de diseño.
- Resolución de problemáticas reales en entornos académicos, fortaleciendo nuestro perfil profesional.

Funcionalidades identificadas

Gestión de cursos (para profesores)

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

Acceso y registro (para estudiantes)

6. Unirse a un curso al que fueron invitados utilizando un código único.
7. Acceder sólo a los cursos en los que están inscritos.
8. Se debe poder decidir qué tipo de notificaciones se quiere recibir.

Mensajería en tiempo real

9. 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.
10. Enviar mensajes entre estudiantes y profesores dentro del curso.
11. Recibir notificaciones de mensajes nuevos y anuncios importantes.
12. Crear hilos de discusión organizados por temas dentro del curso.
13. Creación de encuestas con fecha límite.
14. Panel en donde se vean todas mis conversaciones, y filtro para encontrar rápidamente conversaciones individuales o grupales

Roles y permisos

15. Definir roles específicos para profesores y estudiantes, asegurando que cada uno tenga acceso únicamente a las funcionalidades necesarias.
    - Profesores: gestión de cursos y moderación de mensajes.
    - Estudiantes: acceso a cursos y comunicación.

Seguridad y privacidad

16. Garantizar que solo usuarios autorizados puedan acceder a la información del curso.

17. Proteger los datos de los usuarios mediante protocolos de autenticación segura.

Adicionales

18. Módulo para coordinar transporte compartido entre usuarios del sistema.

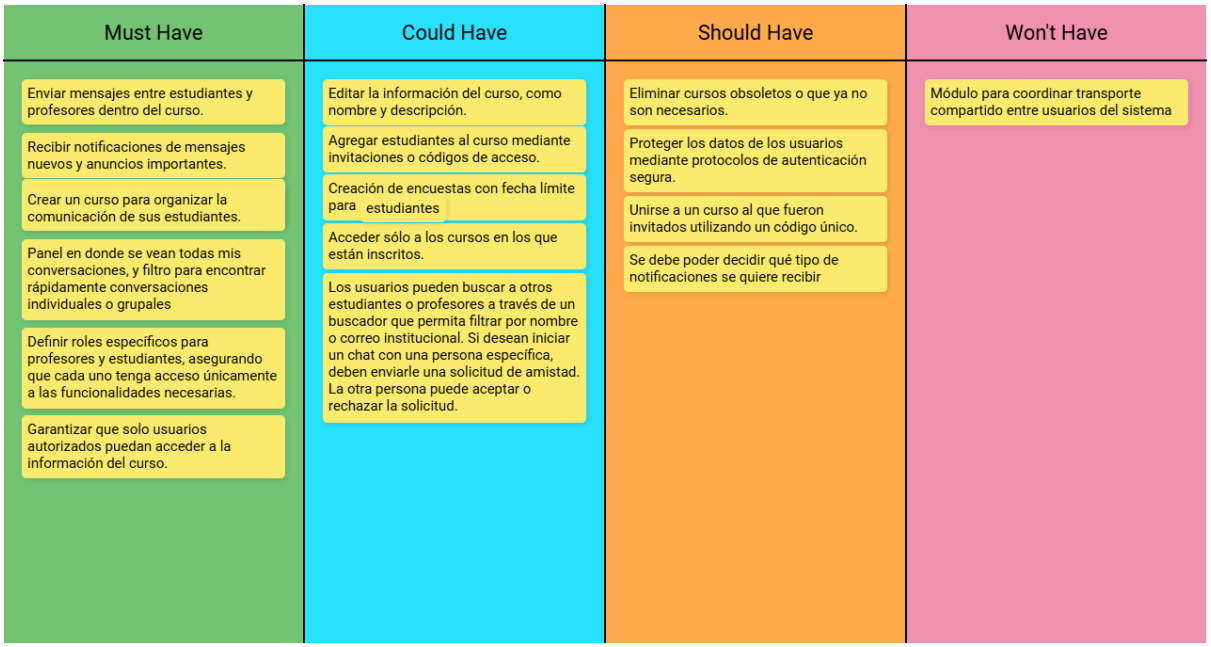
Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

Análisis de requerimientos

MoSCoW

Para el análisis de requerimientos se utilizó el análisis MoSCoW, para interpretar la siguiente tabla se siguieron los siguientes criterios:

- **Must Have**: requisitos críticos para la versión dada para que se considere un éxito.
- **Should Have**: requisitos de alta prioridad que deben incluirse en la versión determinada si es posible.
- **Could Have:** requisitos deseables, pero no necesarios para el éxito del lanzamiento determinado.
- **Won't Have:** los requisitos que las partes interesadas hayan acordado no se implementarán en la versión, pero pueden considerarse en el futuro.



Estimación Fibonacci

Tarea	Complejidad Técnica	Recursos Disponibles (1-13)	Impacto en la experiencia del Usuario	Relación con los Objetivos del Proyecto	Esfuerzo Total

	(1-13)		(1-13)	(1-13)	
Enviar mensajes entre estudiantes y profesores dentro del curso.	5	3	8	8	5
Recibir notificaciones de mensajes nuevos y anuncios importantes.	3	2	5	5	3
Crear un curso para organizar la comunicación de sus estudiantes.	8	5	8	8	8
-Panel en donde se vean todas mis conversaciones, y filtro para encontrar rápidamente conversaciones individuales o grupales	5	3	8	8	5
Definir roles específicos para profesores y estudiantes, asegurando que cada uno tenga acceso únicamente a las funcionalidades necesarias.	8	5	8	8	8
-Garantizar que solo usuarios autorizados puedan acceder a la información del curso.	5	3	8	8	5
Editar la información del curso, como nombre y descripción.	3	2	5	5	3

Agregar estudiantes al curso mediante invitaciones o códigos de acceso.	5	3	8	8	5
Creación de encuestas con fecha límite dentro de una conversación o curso.	3	2	5	5	3
Acceder sólo a los cursos en los que están inscritos.	3	2	5	5	3
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.	5	3	8	8	5
Proteger los datos de los usuarios mediante protocolos de autenticación segura.	2	5	3	3	3

El cálculo del esfuerzo total se hizo promediando todas las columnas y luego redondeando al número de la secuencia fibonacci más cercano.

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

Gestión del software

Tarea	Tiempo (Semanas)	Costo	Alcance
Módulo Gestión de Cursos ● Crear un curso para organizar la comunicación de sus estudiantes. ● Editar la información del curso, como nombre y descripción ● Eliminar cursos obsoletos o que ya no son necesarios ● Agregar estudiantes al curso mediante invitaciones o códigos de acceso ● Unirse a un curso al que fueron invitados utilizando un código único ● Acceder sólo a los cursos en los que están inscritos.	4	● Desarrollador Junior (4 semanas) $450 USD ● Licencias de herramientas (IDE, Git, etc.) $100 USD ● Servicios en la nube (AWS, Google Cloud) $200 USD ● **Total $750 USD**	Incluido ● Crear, editar y eliminar cursos. ● Invitar estudiantes mediante códigos de acceso. ● Unirse a cursos con códigos únicos. ● Acceso restringido a cursos inscritos. Excluido: ● Integración con sistemas externos (por ejemplo, plataformas de gestión educativa como Moodle). ● Funcionalidades avanzadas de personalización de cursos (por ejemplo, temas o diseños personalizados).
Módulo Comunicación y Mensajería ● Enviar mensajes entre estudiantes y profesores dentro del curso. ● Recibir notificaciones de mensajes nuevos y anuncios importantes. ● Crear hilos de discusión organizados por temas dentro del curso. ● Creación de encuestas con fecha límite.	3	● Desarrollador Junior (3 semanas) $337,5 USD ● Servicios en la nube (almacenamiento y mensajería en tiempo real) $150 USD ● **Total $487,5 USD**	Incluido ● Mensajería en tiempo real entre estudiantes y profesores. ● Notificaciones de mensajes y anuncios. ● Hilos de discusión organizados por temas. ● Creación de encuestas con fecha límite. Excluido: ● Mensajería de voz o video.

Módulo Gestión de Usuarios y Seguridad ● Definir roles específicos para profesores y estudiantes. ● Garantizar que solo usuarios autorizados puedan acceder a la información del curso. ● Proteger los datos de los usuarios mediante protocolos de autenticación segura.	2	● Desarrollador Junior (2 semanas) $225 USD ● Servicios en la nube (autenticación y almacenamiento seguro) $150 USD ● **Total $375 USD**	Incluido ● Roles de usuario (profesor y estudiante). ● Autenticación segura (por ejemplo, OAuth) ● Control de acceso a la información del curso. Excluido: ● Autenticación biométrica (huella digital o reconocimiento facial).
Módulo Búsqueda y Organización de Conversaciones ● Los usuarios pueden buscar a otros a través de un buscador. ● Panel donde se vean todas las conversaciones con un filtro para encontrar rápidamente chats individuales o grupales.	2	● Desarrollador Junior (2 semanas) $225 USD ● **Total $225 USD**	Incluido ● Búsqueda de usuarios y conversaciones. ● Panel de conversaciones con filtros. Excluido: ● Búsqueda avanzada (por ejemplo, por palabras clave o fechas).

Casos de uso

El número que acompaña al título del caso de uso, es según el orden de la lista de requerimientos funcionales.

(Daniel)

10. Enviar mensajes entre estudiantes y profesores dentro del curso.
Breve descripción Cuadro para escribir texto que al seleccionarlo el usuario ingrese por teclado el contenido del mensaje, al presionar el botón de enviar, se enviará el mensaje y llegará al panel de chat del destinatario.
Actor Estudiante y profesor
Precondiciones Tener una conversación activa (un chat) entre el remitente y el destinatario.
Flujo principal 1. El usuario (estudiante o profesor) abre el chat dentro del curso. 2. Escribe el mensaje en el cuadro de texto. 3. Presiona el botón de "Enviar". 4. El sistema verifica que el mensaje no esté vacío. 4a. Si el mensaje está vacío, el sistema muestra una alerta indicando que no se puede enviar un mensaje en blanco. 5. El sistema guarda el mensaje en la base de datos. 6. El mensaje se muestra en el panel del chat del remitente. 7. El mensaje se envía al destinatario. 8. El sistema muestra el mensaje en el panel del chat del destinatario en tiempo real.
Poscondiciones Mensaje mostrado en el panel del chat tanto en remitente como en destinatario.

(Daniel)

11. Recibir notificaciones de mensajes nuevos y anuncios importantes.
Breve descripción Cuando un usuario recibe un nuevo mensaje en un chat individual, grupal o un anuncio de un profesor, el sistema genera una notificación en tiempo real para informarle.
Actor Estudiantes, profesor

Precondiciones

El usuario debe estar registrado y autenticado en la aplicación.
El usuario debe tener notificaciones activadas en la aplicación.
Debe existir al menos un chat o curso activo donde se envíen mensajes o anuncios.

Flujo principal

1. Un usuario (profesor o estudiante) envía un mensaje o anuncio en un chat o curso.
2. El sistema recibe el mensaje/anuncio y lo guarda en la base de datos.
3. El sistema detecta que hay destinatarios activos en la conversación o curso.
4. El sistema genera una notificación con el contenido del mensaje o anuncio.
5. Si el destinatario está en la aplicación, la notificación se muestra en tiempo real.
6. Si el destinatario no está en la aplicación, el sistema envía una notificación push (si está habilitada) o un correo electrónico.

Flujo Alternativo 1: Usuario sin conexión a Internet

4a. Si el destinatario no tiene conexión a Internet, el sistema guarda la notificación en la base de datos.
4b. Cuando el usuario se reconecta, el sistema le envía las notificaciones pendientes.

Flujo Alternativo 2: Usuario con notificaciones desactivadas

4a. Si el usuario ha desactivado las notificaciones en la configuración, el sistema no envía alertas en tiempo real.
4b. Sin embargo, el mensaje/anuncio seguirá visible cuando el usuario ingrese a la aplicación.

Poscondiciones
El usuario recibe la notificación del mensaje o anuncio en su dispositivo.
El mensaje o anuncio queda registrado en el chat o curso correspondiente.

(Daniel)

1. Crear un curso para organizar la comunicación de sus estudiantes.

Breve descripción
El profesor puede crear un curso dentro de la aplicación, donde podrá agregar estudiantes como participantes, permitiendo la comunicación y el envío de anuncios dentro del grupo.

Actor
Profesor

Precondiciones
El profesor debe estar registrado y autenticado en la aplicación.
El profesor debe tener permisos para crear cursos.

Flujo principal

1. El profesor accede a la opción de "Crear curso".
2. El sistema solicita al profesor ingresar los datos del curso (nombre del curso, descripción, código del curso opcional).
3. El profesor agrega a los estudiantes ingresando sus correos institucionales o seleccionándolos de una lista.
4. El sistema verifica que los correos ingresados correspondan a estudiantes registrados.
5. El profesor confirma la creación del curso.
6. El sistema guarda la información del curso en la base de datos.
7. El sistema genera una solicitud de invitación a los estudiantes agregados.
8. Los estudiantes pueden aceptar o rechazar la invitación.
9. Si aceptan, el curso aparece en su lista de cursos dentro de la aplicación.

Flujo Alternativo 1: El profesor ingresa un correo inválido

4a. Si el profesor ingresa un correo que no está registrado en la plataforma, el sistema muestra un mensaje de error.
4b. El profesor puede corregir la lista y volver a intentarlo.

Flujo Alternativo 2: El estudiante rechaza la invitación

8a. Si un estudiante rechaza la invitación, el sistema registra la acción y el estudiante no se une al curso.

Poscondiciones
El curso queda registrado en la plataforma con su lista de participantes.
Los estudiantes aceptados pueden acceder al curso y participar en los chats.
El profesor puede gestionar los miembros y enviar anuncios dentro del curso.

(sebastian)

17. Proteger los datos de los usuarios mediante protocolos de autenticación segura.

Breve descripción
Permite que los usuarios de la institución académica accedan a la aplicación UNión de manera segura.

Actor
Usuario (Profesor o Estudiante)

Precondiciones
- El usuario debe tener una cuenta institucional de Google con un correo terminado en (unal.edu.co) para poder iniciar sesión en la aplicación.

Flujo principal
1. En la página principal de la aplicación el usuario hace clic en el botón “Iniciar sesión”.
2. El sistema redirecciona al usuario a la interfaz de inicio de sesión.
3. El usuario hace clic en el botón “Iniciar sesión con Google”.
4. El sistema verifica las cuentas de Google activas en el dispositivo y le permite al usuario seleccionar una para iniciar sesión.
5. El sistema recupera la información del usuario desde Google y verifica que el correo termine en (unal.edu.co)
6. Si el correo es válido, se genera un token de autenticación seguro.
7. El usuario es redirigido a la interfaz principal de la aplicación.

Flujo Alternativo 1: El usuario no tiene una cuenta de Google activa en el dispositivo

4a.Si el usuario no tiene una cuenta de Google activa en el dispositivo, Google solicitará ingresar credenciales (correo y contraseña de Google).

4b.El usuario introduce sus credenciales de Google.

4c.Google verifica las credenciales y, si son correctas, procede con el flujo principal en el paso 5.

4c.Si las credenciales son incorrectas, Google muestra un mensaje de error y solicita reintentar.

Flujo Alternativo 2: El correo no cumple con el dominio requerido

5a.Si el correo no cumple con el dominio requerido, el sistema muestra un mensaje de error, no permite el acceso y redirecciona al usuario al paso 4.

Poscondiciones
- El usuario autenticado puede acceder a las funcionalidades de la aplicación.
- Se ha generado y almacenado un token de autenticación válido para la sesión activa.

(sebastian)

9. Búsqueda y solicitud de amistad entre usuarios

Breve descripción
Permite que los usuarios de la aplicación busquen a otros usuarios mediante un buscador y les envíen solicitudes de amistad para iniciar una conversación.

Actor
Usuario (Profesor o Estudiante)

Precondiciones
- Los usuarios deben estar registrados en la aplicación.

Flujo principal
1. En la pantalla principal de la aplicación, el usuario accede al buscador.
2. El usuario ingresa el nombre o correo de la persona que desea encontrar.
3. El sistema muestra una lista de usuarios que coinciden con la búsqueda.
4. El usuario selecciona a la persona con la que desea iniciar un chat.
5. El usuario envía una solicitud de amistad.
6. El sistema notifica al destinatario sobre la solicitud de amistad.
7. Si el destinatario acepta la solicitud, el sistema habilita el chat entre ambos usuarios.

Flujo Alternativo 1: No hay coincidencias en la búsqueda

3a. Si no se encuentran coincidencias en la búsqueda, el sistema muestra un mensaje indicando que no se encontraron usuarios con esos criterios y sugiere buscar nuevamente.

Flujo Alternativo 2: El usuario ya ha enviado una solicitud a esta persona

5a. Si el usuario ya ha enviado previamente una solicitud de amistad a la misma persona y esta aún no ha sido aceptada o rechazada, el sistema muestra un mensaje informativo indicando que la solicitud está pendiente.

Flujo Alternativo 3: El destinatario rechaza la solicitud

7a. Si el destinatario rechaza la solicitud, el sistema notifica al solicitante y no habilita el chat.

Poscondiciones
- Si la solicitud de amistad es aceptada, ambos usuarios pueden iniciar una conversación en la aplicación.
- La relación de amistad queda guardada en la base de datos, lo que permite establecer nuevas conversaciones siempre y cuando esta se mantenga.

(sebastian)

13. Creación de encuestas con fecha límite dentro de una conversación o curso

Breve descripción
Permite que los usuarios creen encuestas dentro de una conversación o curso, estableciendo una fecha límite para la votación.

Actor
Usuario (Profesor o Estudiante)

Precondiciones
- El usuario debe haber iniciado sesión en la aplicación.
- El usuario debe formar parte de la conversación o curso donde se creará la encuesta.

Flujo principal
1. En una conversación o curso, el usuario accede a la opción "Crear encuesta".
2. El usuario ingresa la pregunta de la encuesta y define las opciones de respuesta.
3. El usuario establece una fecha y hora límite para la votación.
4. El usuario confirma la creación de la encuesta.
5. El sistema guarda la encuesta y la pública en la conversación o curso.
6. Los participantes pueden visualizar y responder la encuesta hasta la fecha límite.
7. Al llegar la fecha límite, el sistema cierra la encuesta y muestra los resultados.

Flujo Alternativo 1: El usuario no completa todos los campos obligatorios

3a. Si el usuario no ha ingresado la pregunta, opciones de respuesta o fecha límite, el sistema muestra un mensaje de error y no permite continuar hasta completar todos los campos.

Flujo Alternativo 2: La encuesta es eliminada antes de la fecha límite

6a. Si el creador de la encuesta decide eliminarla antes de la fecha límite, el sistema elimina la encuesta y notifica a los participantes que ya no está disponible.

Flujo Alternativo 3: La fecha límite ya pasó antes de la votación

6b. Si la fecha límite ingresada por el usuario es anterior a la fecha y hora actual, el sistema muestra un mensaje de error y solicita una nueva fecha válida.

Poscondiciones
- La encuesta queda registrada en la conversación o curso hasta que llegue la fecha límite o sea eliminada.
- Los resultados de la encuesta están disponibles para su consulta una vez finalizada la votación.

(Miguel)

2. Editar información del curso

Breve descripción

Permitir al profesor modificar detalles del curso (nombre, descripción).
Actor Profesor
Precondiciones Curso existente, permisos de edición
Flujo principal 1. El Profesor selecciona "Editar curso". 2. Modifica campos deseados. 3. Confirma cambios. 4. Sistema actualiza la información en la base de datos. **Flujo principal** 1. Campos vacíos → sistema muestra error.
Poscondiciones

(Miguel)

4. Agregar estudiantes mediante invitaciones
Breve descripción Invitar estudiantes a un curso mediante correo o código
Actor Profesor
Precondiciones Curso creado
Flujo principal 1. El profesor selecciona "Agregar estudiantes". 2. Ingresa correos o genera código de acceso. 3. El sistema envía invitaciones. 4. Estudiantes aceptan/rechazan **Flujo alternativo** 1. Correo inválido → sistema notifica error
Poscondiciones

(Miguel)

6. Acceder solo a cursos inscritos

Breve descripción Restringir acceso a cursos donde el estudiante está inscrito
Actor Estudiante
Precondiciones Estudiante autenticado
Flujo principal 1. Estudiante inicia sesión. 2. El sistema muestra solo cursos inscritos. 3. Intento de acceso a curso no autorizado → sistema bloquea.
Poscondiciones

15. Definir roles específicos (profesor/estudiante)
Breve descripción Asignar roles con permisos diferenciados
Actor Administrador
Precondiciones Usuario administrador
Flujo principal 1. Administrador asigna rol en panel de control. 2. Sistema actualiza permisos (ej: profesor puede crear cursos). **Flujo alternativo** 1. Rol inválido → sistema rechaza asignación
Poscondiciones

16. Solo los usuarios autorizados puedan acceder a la información del curso
Breve descripción Asignar roles con permisos diferenciados
Actor

Administrador
Precondiciones Usuario administrador
Flujo principal 3. Administrador asigna rol en panel de control. 4. Sistema actualiza permisos (ej: profesor puede crear cursos). **Flujo alternativo** 2. Rol inválido → sistema rechaza asignación
Poscondiciones

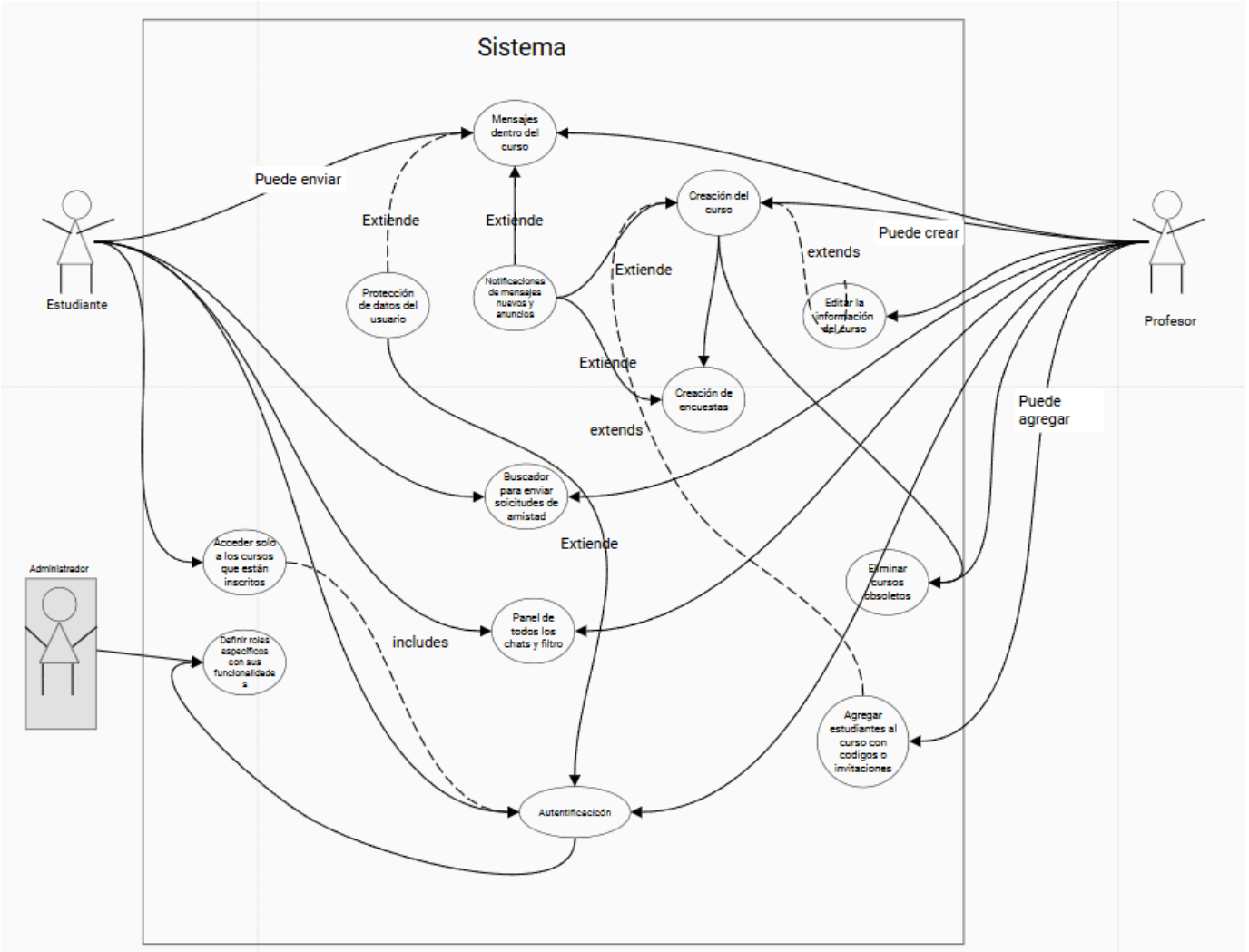
14. Panel de todas las conversaciones y filtro para búsqueda.
Breve descripción El usuario podrá visualizar un panel con todas sus conversaciones activas, ya sean individuales o grupales. Además, podrá filtrar las conversaciones por tipo (grupo o individual) y buscar mensajes dentro de un chat específico.
Actor Estudiante o Profesor
Precondiciones El usuario debe haber iniciado sesión en la aplicación. El usuario debe tener al menos una conversación activa.
Flujo principal 1. El usuario accede al panel de conversaciones. 2. El sistema muestra la lista de conversaciones activas del usuario. 3. El usuario puede seleccionar un filtro (individual o grupal). 4. El sistema actualiza la vista mostrando solo los chats que coinciden con el filtro. 5. (Opcional) El usuario ingresa un término de búsqueda para encontrar un chat o mensaje específico. 6. El sistema muestra los resultados que coincidan con la búsqueda. **Flujo alternativo:** ● **F1:** Si el usuario no tiene conversaciones activas, el sistema muestra un mensaje indicando que no hay chats disponibles. ● **F2:** Si no hay resultados en la búsqueda, el sistema muestra un mensaje indicando que no se encontraron coincidencias.
Poscondiciones

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

Se muestra la lista de conversaciones filtradas o la conversación buscada.
El usuario puede acceder a cualquier chat desde el panel.

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

Diagrama Casos de uso



Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

UNIVERSIDAD NACIONAL DE COLOMBIA

Historias de Usuario

Historia de Usuario # 1

Módulo	Autenticación
Descripción de la(s) funcionalidad(es) requerida(s):	Caso de uso 17: Proteger los datos de los usuarios mediante protocolos de autenticación segura.

URL	**Método**	**Código html**
api/auth/login	POST	200,400

Caso de uso técnico

Se realiza un un 'POST' para autenticar al usuario mediante Google, si todo está bien retorna 200 y un token de autenticación, si algo está mal retorna 400 y un mensaje de error.

Datos de entrada	**Datos de salida**

200:
{
"email": "usuario@unal.edu.co",
"token": "token_de_google"
}
	200:
	{
	"status": "success",
	"data":{
	"auth_token":"token_autenticacion"
	},

	“message”:”The user was successfully authenticated” }

Historia de Usuario # 2

Módulo	Comunicación y Mensajería
Descripción de la(s) funcionalidad(es) requerida(s):	Caso de uso 9: Búsqueda y solicitud de amistad entre usuarios.

URL	**Método**	**Código html**
/api/friends/request	POST	201,400

Caso de uso técnico

El usuario busca a otro usuario y envía una solicitud de amistad. Si todo está bien, se retoma 201, en caso de error se retoma 400.

Datos de entrada	**Datos de salida**

200:
 	
{
 	
"searcher_id": 12345,
 	
"target_id": 67890
 	
}
	200:
	
	{
	
	"status": "success",
	
	“data”:{
	
	"request_id": 54321
	
	},

	“message”:"Friend request sent successfully" }

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

UNIVERSIDAD NACIONAL DE COLOMBIA

Historia de Usuario # 3

Módulo	Comunicación y Mensajería
Descripción de la(s) funcionalidad(es) requerida(s):	Caso de uso 13: Creación de encuestas con fecha límite dentro de una conversación o curso

URL	**Método**	**Código html**
/api/polls/create	POST	201,400

Caso de uso técnico

El usuario crea una encuesta dentro de una conversación o curso, proporcionando una pregunta, opciones de respuesta y una fecha límite. Si todo está bien, se retoma 201, en caso de error se retoma 400.

Datos de entrada	**Datos de salida**

200:
{
"conversation_id": 12345,
"question": "¿Cuál es tu lenguaje de programación favorito?",
"options": ["Python", "JavaScript", "Java", "C++"],
"deadline": "2023-12-31T23:59:59Z",
}
	200:
	{
	"status": "success",
	“data”:{
	"poll_id": 67890
	},
	“message”:"Poll created successfully"
	}

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

Historia de Usuario # 4

Módulo	Gestión de Usuarios y Permisos
Descripción de la(s) funcionalidad(es) requerida(s):	Caso de uso 15: Permitir al administrador asignar roles (profesor/estudiante) a los usuarios para controlar sus permisos en la plataforma.

Backend

URL	**Método**	**Código http**
api/users/{id}/role	PATCH	200, 400

Caso de uso técnico

- Actualizar el rol de un usuario. Si el ID es válido y el rol es correcto (profesor o estudiante), retorna 200.
- Si el rol no es válido o el usuario no existe, retorna 400.

Datos de entrada	**Datos de salida**

N/A	{
	"status": "success",
	"data":{
	"id": 1,
	"nombre": "Juan Pérez",
	"role": "Profesor"
	},
	"message": "El rol fue asignado correctamente"
	}

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

Historia de Usuario # 5

Módulo	Gestión de Cursos
Descripción de la(s) funcionalidad(es) requerida(s):	Caso de uso 6: Garantizar que los estudiantes solo puedan ver y acceder a los cursos en los que están inscritos.

Backend

URL	**Método**	**Código http**
api/courses	GET	200, 403

Caso de uso técnico

- Retorna lista de cursos inscritos si el usuario está autenticado.
- Si no tiene permisos, retorna 403 (forbidden).

Datos de entrada	**Datos de salida**

N/A	{
	"status": "success",
	"data": [
	{
	"id": 1,
	"name": "Curso 1"
	}
	],
	"message": "Los cursos se han listado correctamente"
	}

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

Historia de Usuario # 6

Módulo	Gestión de Cursos
Descripción de la(s) funcionalidad(es) requerida(s):	4. Permitir al profesor agregar estudiantes a un curso mediante invitaciones por correo o código de acceso.

Backend

URL	**Método**	**Código http**
/courses/{id}/invite	POST	201, 400

Caso de uso técnico

- Envía invitaciones por correo o genera un código. Si los datos son válidos, retorna 201.
- Si el curso no existe o los correos son inválidos, retorna 400.

Datos de entrada

```
{
 "emails": [
   "estudiante@unal.edu.co"
 ],
 "generateCode": true
}
```

Datos de salida

```
{
 "status": "success",
 "data": {
   "code": "ABC123"
 },
 "message": "Se han invitado a los usuarios correctamente"
}
```

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

Historia de Usuario # 7

Módulo	Gestión de Cursos
Descripción de la(s) funcionalidad(es) requerida(s):	Caso de uso 10: Enviar mensajes entre estudiantes y profesores dentro del curso.

URL	**Método**	**Código http**
/api/courses/{course_id}/messages	POST	201, 400

Caso de uso técnico

El usuario (estudiante o profesor) envía un mensaje en el chat de un curso. Si la solicitud es válida, el mensaje se almacena y se devuelve una respuesta 201 Created. De lo contrario, se devuelve una respuesta 400 Bad Request.

Datos de entrada	**Datos de salida**

{
"course_id": 67890,
"sender_id": 12345,
"content": "Hola a todos, el formato de entrega del taller aplicado al proyecto ha sido modificado"
}
	{
	"status": "success",
	"data": {
	"message_id": 98765
	},
	"message": "Message sent successfully"
	}

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

Historia de Usuario # 8

Módulo	Gestión de Cursos
Descripción de la(s) funcionalidad(es) requerida(s):	Caso de uso 1: Crear un curso para organizar la comunicación de sus estudiantes.

URL	**Método**	**Código http**
/api/courses/create	POST	201, 400

Caso de uso técnico

El profesor crea un curso proporcionando un nombre y una descripción. Si todo está bien, se retorna 201 Created, en caso de error se retorna 400 Bad Request.

Datos de entrada	**Datos de salida**

{
"name": "Ingenieria de Software",
"description": "Grupo 1. Para dejar actividades y formatos de entrega.",
"professor_id": 12345
}
	{
	"status": "success",
	"data": {
	"course_id": 67890
	},
	"message": "Course created successfully"
	}

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

Historia de Usuario # 9

Módulo	Notificaciones
Descripción de la(s) funcionalidad(es) requerida(s):	Caso de uso 11: Recibir notificaciones de mensajes nuevos y anuncios importantes.

URL	**Método**	**Código http**
/api/notifications	GET	200,404

Caso de uso técnico

Un usuario (estudiante o profesor) solicita notificaciones. Si existen notificaciones, se recuperan con una respuesta 200 OK. Si no existen notificaciones, se devuelve una respuesta 404 No encontrado.

Datos de entrada


```
{
"user_id": 12345
}
```


Datos de salida


```
{
"status": "success",
"data": [
{
"notification_id": 1,
"type": "message",
"content": "Nuevo mensaje de Daniel."
},
{
"notification_id": 2,
"type": "announcement",
"content": "Nueva asignación en Ingeniería de Software."
}
],
"message": "Notifications retrieved successfully"
}
```

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

Historia de Usuario # 10

Módulo	Notificaciones
Descripción de la(s) funcionalidad(es) requerida(s):	Caso de uso 14: Panel donde se vean todas mis conversaciones y filtro para encontrar rápidamente conversaciones individuales o grupales

URL	**Método**	**Código http**
/api/chats	GET	200,404

Caso de uso técnico

Un Usuario (Estudiante o Profesor) solicita la lista de todas sus conversaciones activas. Si existen, se devuelven con un 200 OK. Si no hay conversaciones, se devuelve un 404 Not Found.

Datos de entrada


```
{
"user_id": 12345,
"filter": "group | individual",
"search_query": "John"
}
```


Datos de salida


```
{
"status": "success",
"data": [
{
"chat_id": 1,
"type": "group",
"name": "Ingeniería de Software",
"last_message": "Nueva asignación"
},
{
"chat_id": 2,
"type": "individual",
"name": "John",
"last_message": "Adios!"
}
],
"message": "Chats retrieved successfully"
}
```

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

Historia de Usuario # 11

Módulo	Notificaciones
Descripción de la(s) funcionalidad(es) requerida(s):	Caso de uso 16: Garantizar que solo usuarios autorizados puedan acceder a la información del curso

URL	**Método**	**Código http**
/api/courses/{course_id}/access	GET	200,403

Caso de uso técnico

Un Usuario (Estudiante o Profesor) intenta acceder a la información de un curso. Si está autorizado, se devuelve un 200 OK con los detalles del curso. Si no tiene permiso, se devuelve un 403 Forbidden.

Datos de entrada	**Datos de salida**

{
"course_id": 67890,
"user_id": 12345
}
 	
}
	{
	"status": "success",
	"data": {
	"course_id": 67890,
	"name": "Ingeniería de Software",
	"description": "Grupo 1.",
	"professor": "Oscar"
	},
	"message": "Access granted"
	}
	
	Si el usuario no tiene acceso:
	{
	"status": "error",
	"data": null,
	"message": "Access denied"
	}

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

Taller 2

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

Patrones de diseño

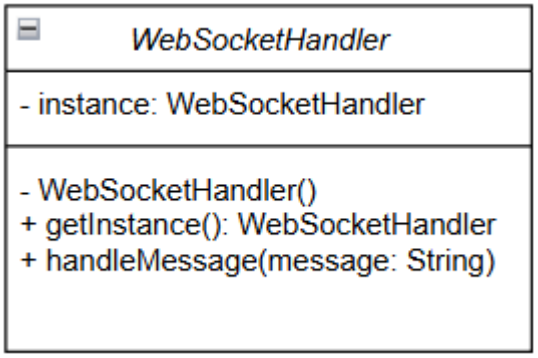
Patrón Singleton

Para WebSocket Handler y servicios críticos

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:



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.

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

Patrón Observer

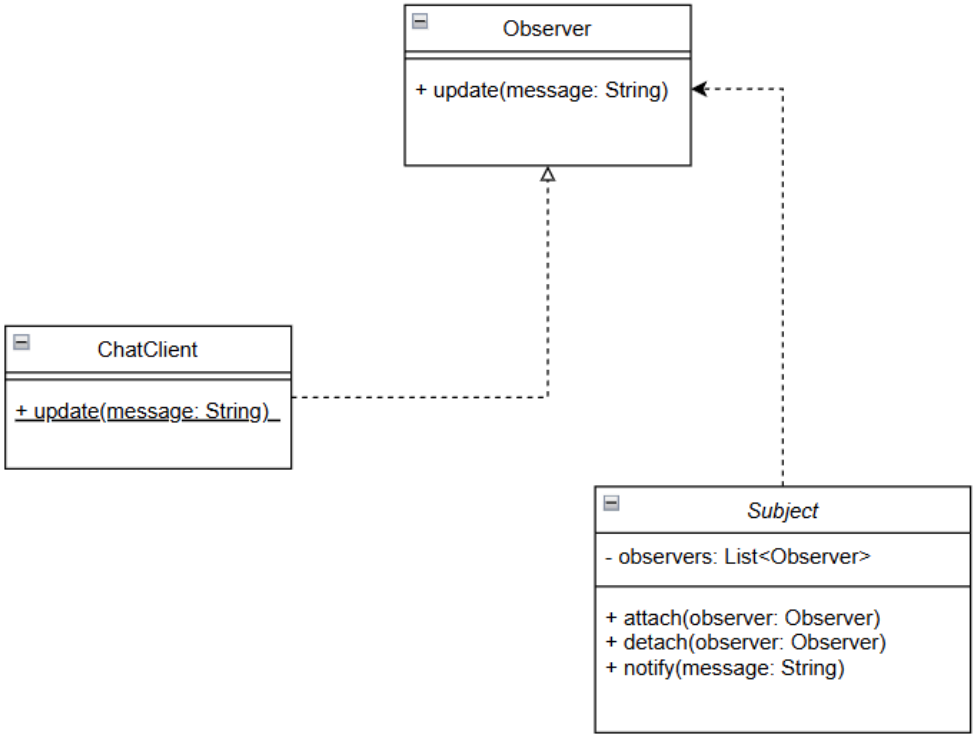
Para notificar eventos en tiempo real, para que los clientes (estudiantes/profesores) sean notificados automáticamente cuando hay nuevos mensajes o cambios en el estado de los chats.

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.



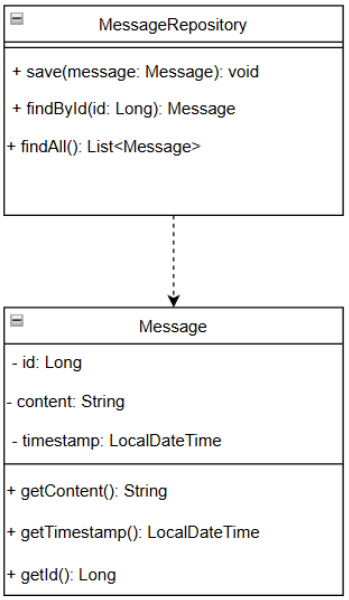
Patrón Repository

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.