Documento diseño e implementación proyecto 1

Natalia Erazo 202213680

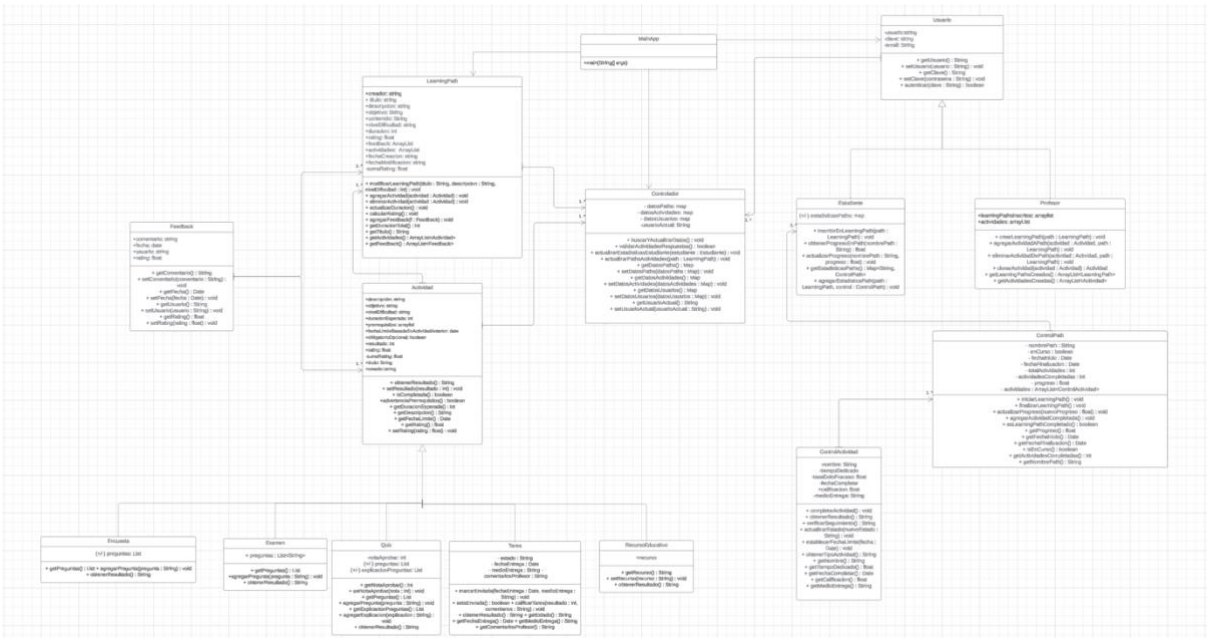
Sofía Arias 202310260

Anaís Castro 202322450

Samuel Diez 202311264

UML DE ALTO NIVEL

https://lucid.app/lucidchart/36b10575-9197-414b-bc6f-79f01ef6f9c7/edit?viewport_loc=-1600%2C-417%2C4033%2C1980%2C0_0&invitationId=inv_f41f64a3-7acb-4388-b5d6-6ce466c68521 .



A grandes rasgos, se propone un modelo de dominio centrado en una clase Controlador, que comunicará a la clase Usuarios (Profesor o Estudiante) con la clase LearningPath, compuesta de objetos cuyo tipo de clase es Actividades. Adicionalmente se definen dos clases secundarias ControlPath, con un atributo compuesto de objetos de la clase ControlActividad, para llevar un recuento de las estadísticas de cada actividad realizada por cada estudiante.

La clase controlador posee cuatro atributos:

- datosPaths: mapa donde se guardan todos los LearningPaths del sistema.
- datosActividades: mapa donde se guardan todos las Actividades del sistema.
- datosUsuarios: mapa donde se guardan todos los Usuarios (Profesores o Estuadiantes) del sistema.
- usuarioActual: el nombre de usuario que está actualmente utilizando el sistema.

Y, además de guardar los archivos, se encarga de que los profesores puedan crear, copiar o editar los Paths que deseen, de que los estudiantes se puedan inscribir a los Paths de interés y de que puedan realizar las actividades que deseen. Todo por medio de la consulta de datos, o la edición de los datos.

Se decide organizar todos los datos en mapas, pues de esta manera resulta más sencillo acceder a los Learning Paths, a las actividades y a los usuarios únicamente conociendo el valor de la llave (complejidad O (1)). Esto agrega una restricción adicional a nuestro programa: no pueden existir Learning Paths, ni actividades, ni usuarios con mismo nombre. Todos deben ser únicos para que no se generen errores de reescritura de los datos.

datosActividades guardará todas las actividades creadas por los Profesores. La clase Actividad se define según:

- descripcion: descripción de la actividad.
- objetivo: objetivo de la actividad.
- nivelDificultad: nivel de dificultad de la actividad.
- duracionEsperada: duración esperada de la actividad.
- prerrequisitos: lista de actividades prerrequisito de la actividad.
- fechaLimite…: fecha límite para realizar la actividad basada en una actividad anterior.
- obligatorioOpcional: si la actividad es obligatoria o no.
- resultado: resultado de la de la actividad.
- rating: rating de la actividad.
- sumaRating: sumatoria de todos los ratings dados a la actividad. Es un atributo auxiliar que permite calcular el rating general de la actividad sin tener que recorrer todos los ratings dados.
- titulo: título de la actividad.
- creador: profesor creador de la actividad.

Y se divide en cinco tipos de actividades distintas que heredan de actividad, pero poseen atributos específicos, tal que:

- Encuesta
    - preguntas: lista de preguntas de la encuesta.
- Examen
    - Preguntas: lista de preguntas del examen.
- Quiz
    - notaAprobar: nota para aprobar el quiz.
    - preguntas: lista de preguntas del quiz.
    - explicacionPreguntas: explicación de la solución de cada pregunta del quiz.
- Tarea
    - estado: estado de la tarea.
    - fechaDeEntrega: tipo Date, fecha de entrega de la tarea.
    - medioEntrega: medio de entrega de la tarea.
    - comentariosProfesor: comentarios del profesor sobre la tarea entregada.

- RecursoEducativo:
    - recurso: recurso educativo que el estudiante debe revisar.

En datosPaths se guardarán todos los Learning Paths, que se definen por:

- creador: creador del path.
- titulo: título del path.
- descripcion: descripción del path.
- objetivo: objetivo del path.
- contenido: contenido del path.
- nivelDificultad: nivel de dificultad del path.
- duracion: duración del path.
- rating: rating del path.
- feedback: lista de reseñas del path (clase Feedback).
- actividades: lista de actividades del path, compuesta por objetos de clase Actividad.
- fechaCreacion: fecha de creación del path.
- fechaModificacion: fecha de modificación del path.
- sumaRating: sumatoria de todos los ratings dados al path. Es un atributo auxiliar que permite calcular el rating general del path sin tener que recorrer todos los ratings dados.

Se define una pequeña clase Feedback para crear reseñas con estructura, definida por:

- comentario: comentario de la reseña.
- fecha: tipo Date, es la fecha en la que se realizó la reseña.
- usuario: usuario que hizo la reseña.
- rating: rating de la reseña.

datosUsuario es un mapa que se compone de todos los usuarios del sistema, donde encontramos dos tipos: Profesor y Estudiante. La clase Usuario se define como:

- usuario: nombre de usuario.
- email: email del usuario.
- clave: clave del usuario.

La clase Profesor hereda todos los atributos de usuario y además tiene:

- learningPaths: lista del nombre de sus LearningPaths.
- actividades: lista del nombre de sus Actividades.

  Los profesores tienen la capacidad de crear nuevas actividades y Learning Paths, de editarlos si son los creadores, o de clonarlos.

La clase Estudiante también hereda los atributos de Usuario y se complementa con:

- estadísticasPaths: un mapa donde para cada LearningPath guarda las estadísticas de cada actividad realizada. Esta se compone de elementos de ControlPath que a su vez contienen objetos de ControlActividad.

Los estudiantes tienen la capacidad de inscribirse a nuevos LearningPaths y de realizar las distintas actividades disponibles.

La clase ControlPath se propone con el objetivo de entender el progreso de un estudiante en una ruta de aprendizaje y a partir de los datos guardados, verificar qué puede hacer dentro del sistema. ControlPath tiene como atributos:

- nombrePath: nombre del path.
- enCurso: booleano que indica si hay alguna actividad en curso dentro del path.
- fechaInicio: fecha de inicio del path.
- fechaFinalizacion: fecha de finalización del path.
- totalActividades: total de actividades realizadas.
- actividadesCompletadas: total de actividades completadas.
- progreso: progreso del estudiante dentro del path.
- actividades: HashMap que contiene todas las estadísticas de las actividades realizadas por el estudiante. Los indicadores de cada actividad se guardan en objetos de tipo ControlActividad.

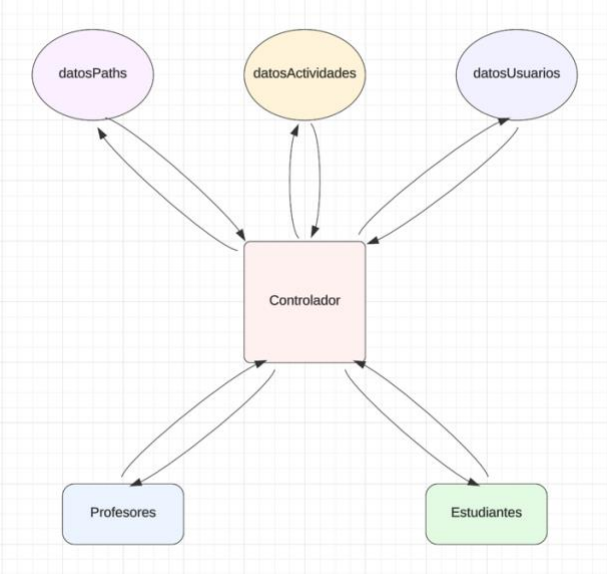
La clase ControlActividad tiene como atributos:

- nombre: nombre de la actividad.
- tiempoDedicado: tiempo dedicado a la actividad.
- tasaExitoFracaso: tasa de éxito y fracaso de la actividad.
- fechaCompletar: fecha en la que se completó la actividad.
- calificacion: calificación de la actividad.
- medioDeEntrega: medio de entrega de la actividad (si aplica).

Finalmente, para inicializar el programa, se emplea la clase Main.

DIAGRAMA DE MÉTODOS

https://lucid.app/lucidchart/1b0ed2d0-39a0-448a-89a9-60850e8e6f6c/edit?viewport_loc=-169%2C78%2C1737%2C1042%2C.Q4MUjXso07N&invitationId=inv_1db47c7f-1130-434d-9e3b-a2be17a7027e



El diagrama de métodos muestra la interacción entre las clases principales del proyecto. En este caso el Controlador funciona como el núcleo central del programa, teniendo la capacidad de comunicarse con las demás clases. El Controlador responde a las demandas de los usuarios, sean estudiantes o profesores, y puede entrar a los datos para verificarlos o modificarlos, según se requiera. Luego envía las respuestas obtenidas a los usuarios. No existe la posibilidad de que los usuarios puedan acceder a los datos por su cuenta, siempre se usa al Controlador como un intermediario. A través de estas interacciones bidireccionales, el controlador garantiza que todos los componentes del sistema se comuniquen de manera eficiente, facilitando una gestión centralizada y coordinada de la información y las operaciones.

PERSISTENCIA

Todo el diseño del programa se elabora teniendo en cuenta la persistencia de los datos. Por esto se propone un gran Controlador con acceso a todos los datos (Learning Paths existentes, Actividades existentes y Usuarios del sistema), y dos tipos de usuarios (Profesores y Estudiantes) que se pueden comunicar con el Controlador para suplir sus necesidades.

Los datos se almacenan en tres archivos JSON distintos, uno para los paths, otro para las actividades y otro para los usuarios del sistema. Como se propone organizarlos en forma de mapas (para facilitar el acceso a los datos mediante las llaves únicas), entonces nos valdremos de la librería externa Jackson, que interpreta de mejor manera los tipos de datos guardados en un JSON y permite que la creación del mapa sea mucho más sencilla. Para interpretar los tipos de datos, Jackson posee la herramienta ObjectMapper que realiza la clasificación de manera interna. El Controlador se encarga de leer estos JSON y convertirlos en los tres mapas.

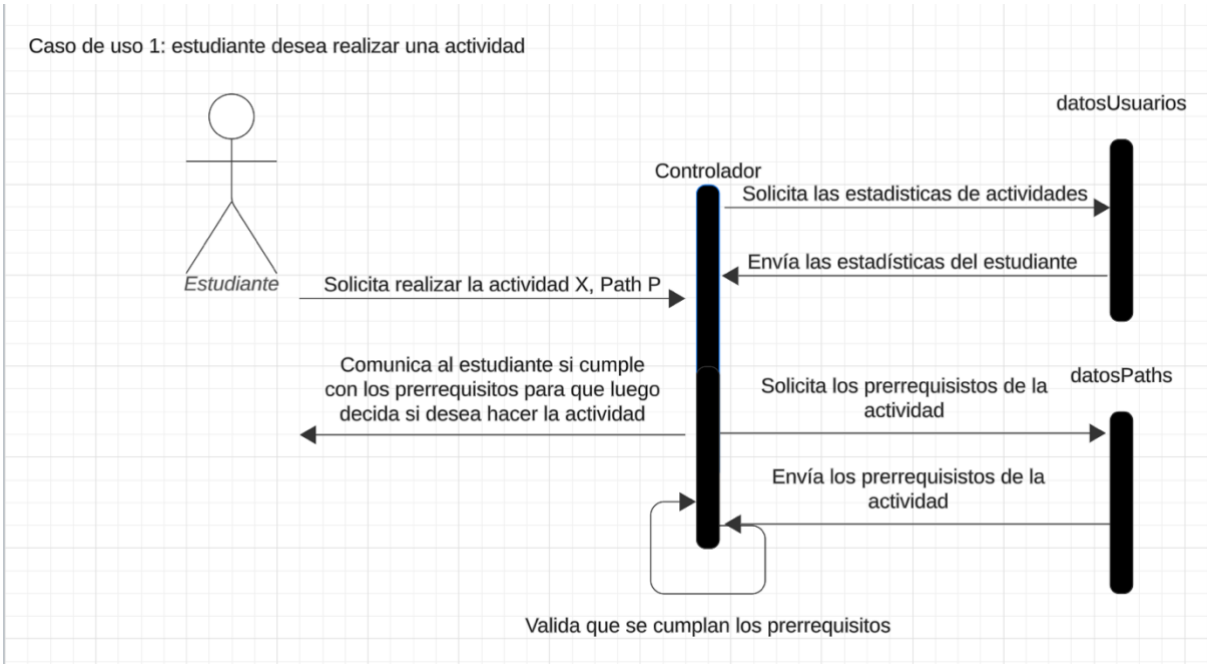
Cuando un usuario desea realizar alguna acción, se debe comunicar con el Controlador, este realizará las verificaciones (si es el caso) y modificará (agregar, eliminar o editar) simultáneamente la información guardada en los mapas y en los JSON. De esta manera, mientras el programa se está ejecutando la información no se perderá pues el mapa se está actualizando, y cuando se quiera acceder nuevamente al programa, los mapas desde JSON se crearán con la información actualizada. Así los usuarios no manipulan directamente la información, ni guardan información de los atributos (descripción, objetivo, etc.) de los Learning Paths o Actividades tomadas o creadas, únicamente guardan los resultados obtenidos al realizar los trabajos propuestos. Cuando se actualice una ruta de aprendizaje no deberían existir conflictos entre las actividades realizadas y la nueva información, dado que los estudiantes no están duplicando los Learning Paths, únicamente llevan un conteo de sus resultados obtenidos para hacer comparaciones con la información de cada ruta.

Caso 1: Estudiante desea realizar una actividad.

El estudiante solicita realizar la actividad X, del Path P. El Controlador se dirige a la actividad X y recupera la lista de prerrequisitos. El Controlador entra a comparar las estadísticas de las actividades ya realizadas por el estudiante con los prerrequisitos específicos de cada actividad. Si esta comparación es exitosa (el estudiante cumple con los prerrequisitos), entonces procede a mostrar al estudiante la actividad a realizar. Compara las respuestas dadas con las esperadas y brinda una nota. Se crea entonces un nuevo ControlActividad de la actividad X ligada a ControlPath, para el Path P, en el mapa del estudiante. Esta actualización de los datos del mapa datosUsuarios también se realiza en el archivo

JSON usuarios.json. En caso de que los prerrequisitos no se cumplan se le pregunta al Estudiante si desea o no realizar la actividad, si la realiza se ejecuta el proceso anterior.

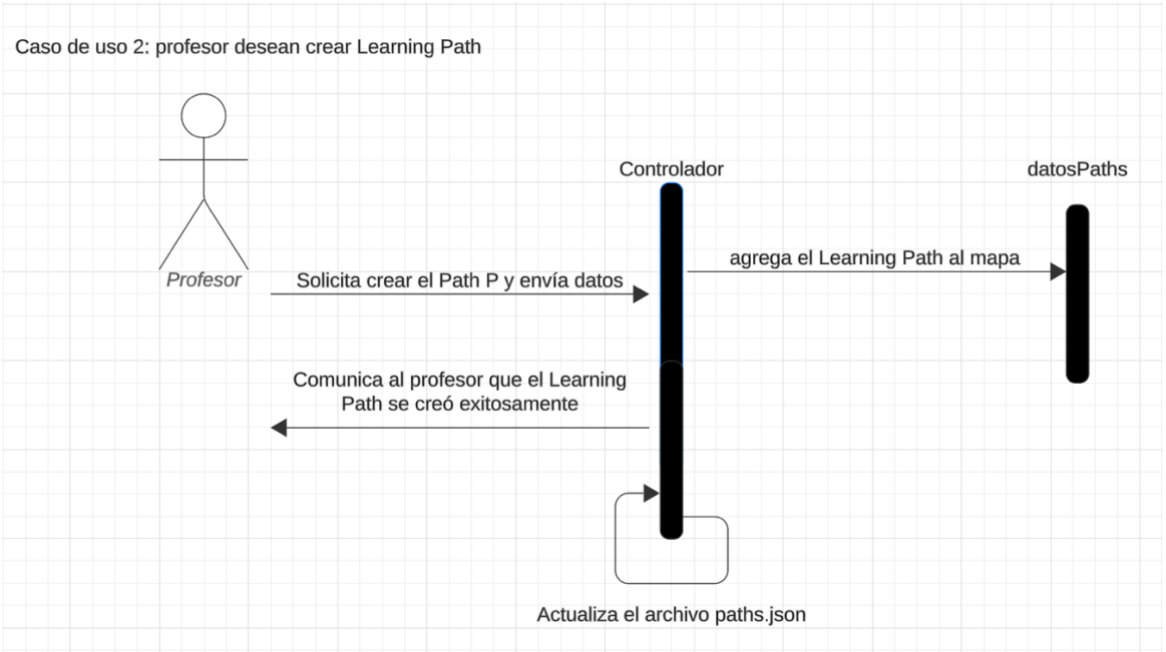
Diagrama de flujo Caso 1.



Caso 2: Profesor desea crear un Learning Path.

El profesor solicita la creación de un nuevo Learning Path P. Con todos los datos brindados para la creación de este Learning Path, el Controlador agregar el path P al mapa datosPaths y actualiza el JSON paths.json. Cabe agregar que el nuevo Learning Path solo se podrá crear si tiene un nombre único, que no se encuentre ya en el mapa datosPaths. Así ya se ha creado el Learning Path P, y está disponible para todos los que lo quieran tomar o clonar.

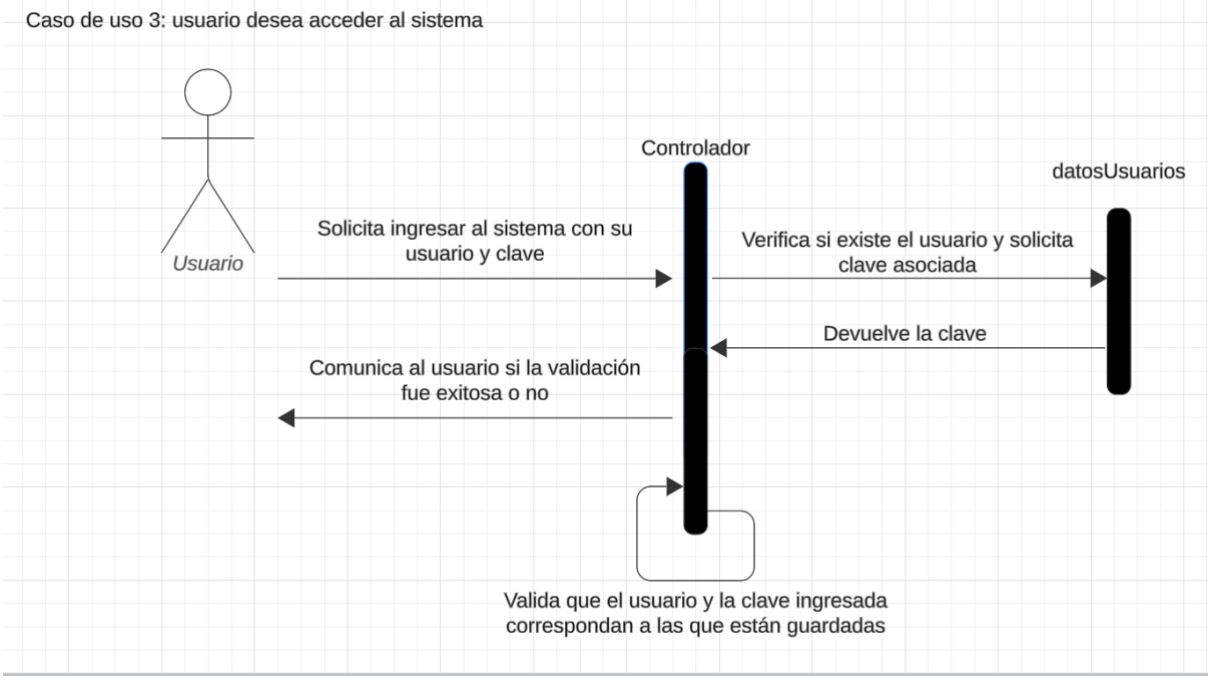
Diagrama de flujo Caso 2.



Caso 3: Un usuario desea acceder al sistema.

Un usuario (profesor o estudiante) desea acceder al sistema. Ingresa su usuario (único en todo el sistema, pues no hay nombres de usuario repetidos) y su clave. El Controlador recibe estos parámetros y entra a revisar el mapa datosUsuarios. Primero verifica que el usuario exista en el mapa, de ser así revisa que la clave guardada corresponda a la clave ingresada por el usuario. Si ambas verificaciones son exitosas, entonces el usuario puede acceder al sistema y realizar lo que requiera.

Diagrama de flujo Caso 3.



HISTORIAS DE USUARIO

- Profesor

Como profesor quiero una plataforma, que sea fácil de navegar, que me permite estar directamente conectado con el aprendizaje de mis estudiantes, para poder ver como progresan en su proceso de aprendizaje y hacer recomendaciones que los ayudaran a mejorar su conocimiento.

Acceptance criteria:

- Crear learning paths
- Editar los learning paths creados
- Crear actividades
- Revisar progreso de learning path de sus estudiantes
- Feedback sobre las actividades que completan los estudiantes

- Estudiante

Como estudiante quiero una plataforma de estudios para poder tener un aprendizaje complementario a una clase y de esta manera obtener planes de estudio que me acerquen a mis metas de aprendizaje.

Acceptance criteria:

- Inscribirse en un learning path
- Revisar progreso de learning path
- Revisar estado de actividad
- Completar actividades
- Dejar comentarios de retroalimentación y ratings en las actividades completadas

USER PERSONA

- Profesor

- Estudiante

REQUERIMIENTOS FUNCIONALES

Gestión de Usuarios

- El sistema debe permitir el registro y autenticación de usuarios (estudiantes y profesores).
- Los estudiantes deben poder ver Learning Paths recomendados según sus intereses y progreso.
- Los profesores deben poder crear y gestionar Learning Paths personalizados.
- Los profesores deben poder clonar y editar Learning Paths existentes.

Gestión de Learning Paths

- Los profesores deben poder crear Learning Paths ingresando información básica (título, descripción, duración, nivel de dificultad, objetivos).

- Los profesores deben poder agregar, modificar o eliminar actividades dentro de un Learning Path.
- El sistema debe calcular la duración estimada del Learning Path sumando las duraciones de las actividades.
- Los estudiantes deben poder inscribirse en Learning Paths.
- Los estudiantes deben poder marcar un Learning Path como completado al finalizar las actividades obligatorias.

Gestión de Actividades

- El sistema debe permitir la creación de actividades como:
  - Revisar un recurso educativo (PDF, video, sitio web, etc.).
  - Completar una tarea revisada por un profesor.
  - Realizar un quiz de opción múltiple.
  - Participar en encuestas.
- Las actividades deben poder ser marcadas como obligatorias u opcionales.
- El sistema debe mostrar el progreso de las actividades, incluyendo fechas límite.
- El sistema debe notificar a los estudiantes sobre actividades pendientes y fechas de entrega.

Feedback y Rating

- Los estudiantes deben poder dejar feedback y ratings en las actividades completadas.
- Los profesores deben poder revisar el feedback para mejorar los Learning Paths.
- El sistema debe mostrar el promedio de ratings de cada actividad.

Gestión de Inscripción y Progreso

- Los estudiantes deben poder inscribirse en varios Learning Paths.
- El sistema debe registrar el estado de la inscripción (en progreso, completado, abandonado).
- El sistema debe permitir a los estudiantes ver su progreso en cada Learning Path.
- El sistema debe mostrar el porcentaje de actividades obligatorias completadas en un Learning Path.
- Los estudiantes deben poder reiniciar un Learning Path.

Sistema de Recomendaciones

- El sistema debe recomendar Learning Paths basados en los intereses y progreso del estudiante.
- El sistema debe enviar notificaciones automáticas sobre actividades pendientes.
- Los estudiantes deben poder filtrar y buscar Learning Paths por criterios como duración, nivel de dificultad y rating.

Integración con LMS (Learning Management Systems)

- El sistema debe integrarse con plataformas LMS (como Bloque Neón) para registrar automáticamente la finalización de tareas y quices.
- El sistema debe notificar a los LMS sobre el progreso del estudiante en las actividades.

Gestión de Evaluaciones y Resultados

- Los profesores deben poder configurar quices con preguntas de opción múltiple.
- El sistema debe corregir automáticamente los quices y asignar una calificación.
- Los profesores deben poder revisar y calificar tareas entregadas por los estudiantes.
- Los estudiantes deben poder ver sus resultados de evaluaciones en tiempo real.

Reportes y Análisis

- El sistema debe generar reportes sobre el progreso de los estudiantes en los Learning Paths.
- El sistema debe analizar datos de progreso y rendimiento para generar estadísticas, como diferencias de éxito en quices realizados en distintos momentos del día.
- El sistema debe mostrar tasas de finalización y éxito de actividades.

Control de versiones de Learning Paths

- El sistema debe permitir crear nuevas versiones de Learning Paths sin afectar las inscripciones activas.
- El sistema debe mantener un historial de versiones, con fecha de modificación y cambios realizados.

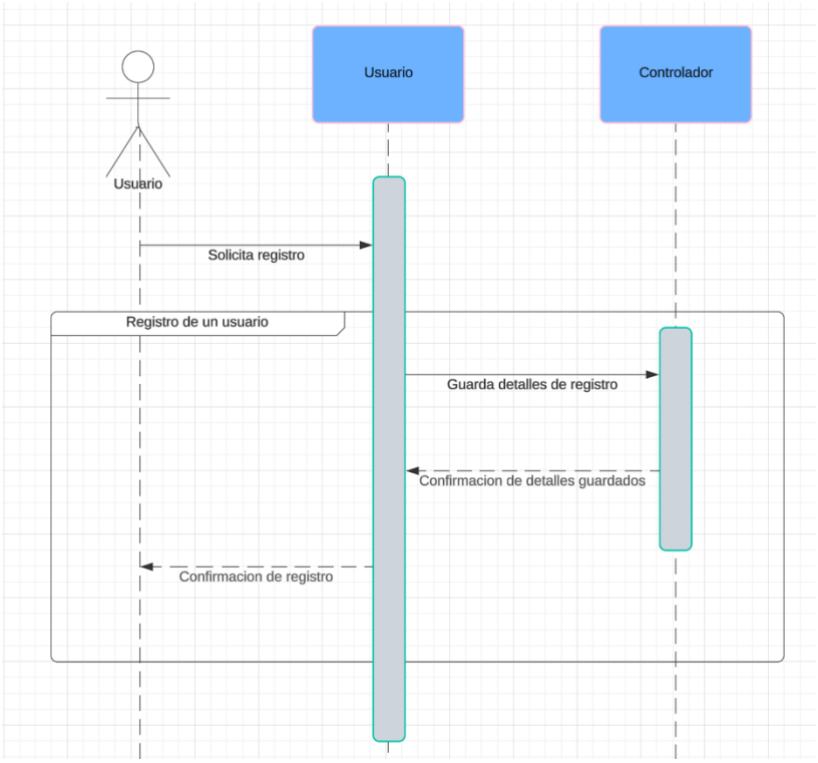
REQUERIMIENTOS NO FUNCIONALES

- Rendimiento:
  El sistema debe cargar de manera rápida y eficiente en condiciones normales de uso para funciones clave, como el registro de usuarios, la búsqueda de Learning Paths y la visualización del progreso.
- Usabilidad:
  La interfaz debe seguir principios de diseño centrado en el usuario, utilizando un lenguaje claro y accesible para estudiantes y profesores, evitando jerga técnica innecesaria.
- Seguridad:
  Los datos de los usuarios deben estar protegidos.
- Disponibilidad:
  El sistema debe estar disponible la mayoría del tiempo, garantizando un acceso continuo para estudiantes y profesores.

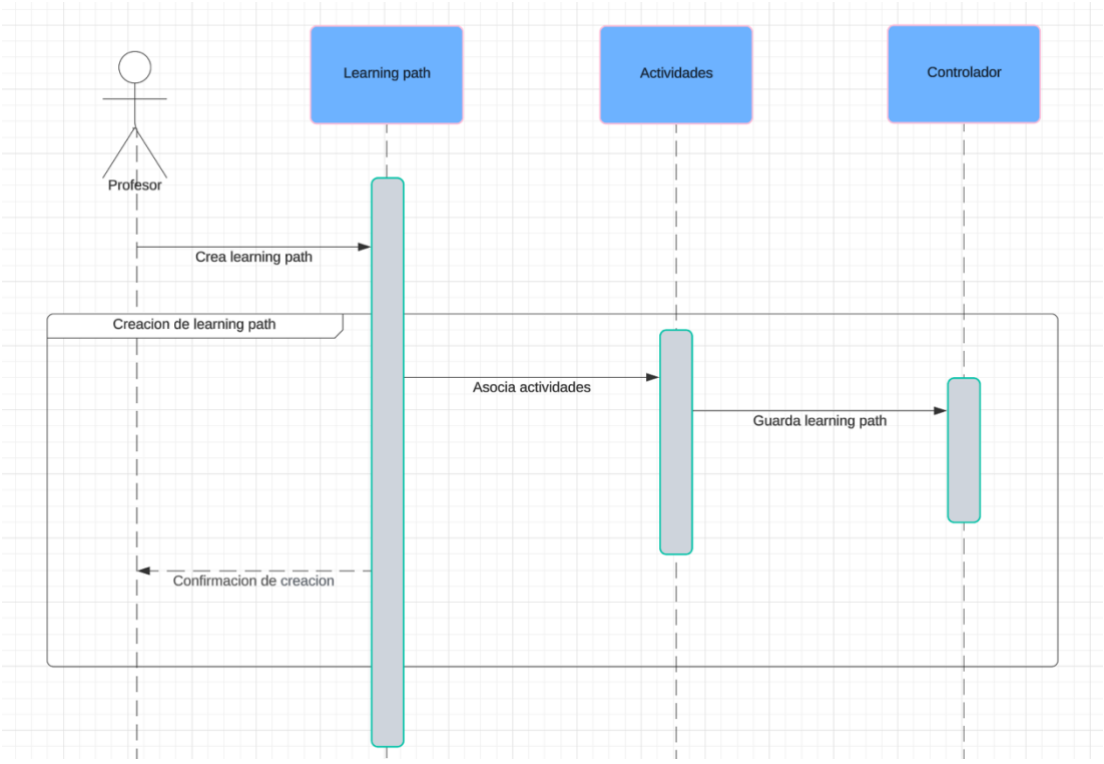
- Compatibilidad:
  La aplicación debe ser compatible con los principales navegadores web (Chrome, Firefox, Safari y Edge) en sus versiones más recientes, asegurando una experiencia de usuario consistente.
- Mantenibilidad:
  El código del sistema debe seguir buenas prácticas de programación, permitiendo que nuevos desarrolladores entiendan y modifiquen el sistema fácilmente. Además, debe incluir una documentación clara que detalle la arquitectura del sistema y las funcionalidades.

DIAGRAMAS DE SECUENCIA

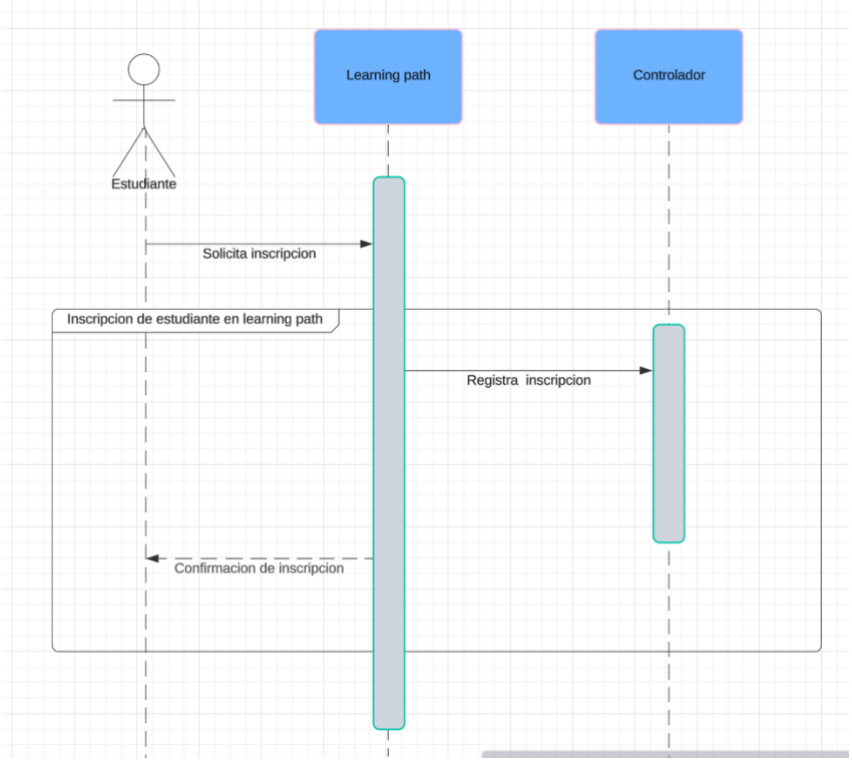
- Diagrama de secuencia de registro y autenticación de usuarios



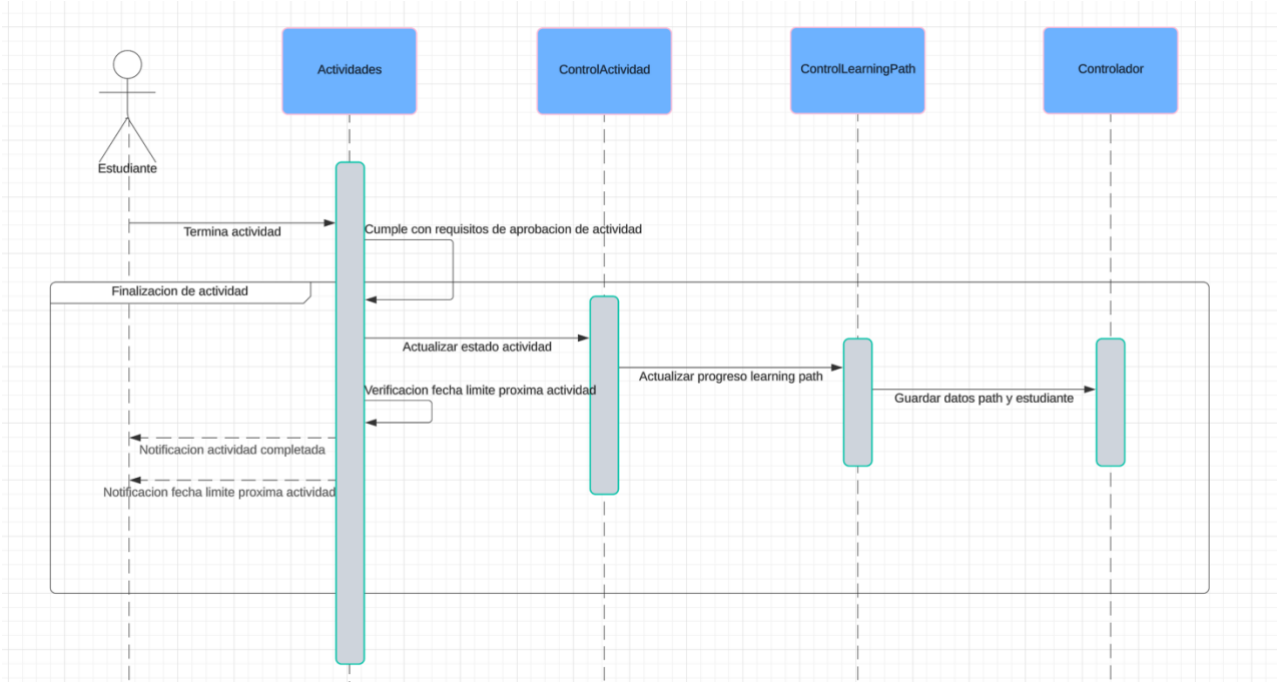
- Diagrama de secuencia para la creación de un Learning Path



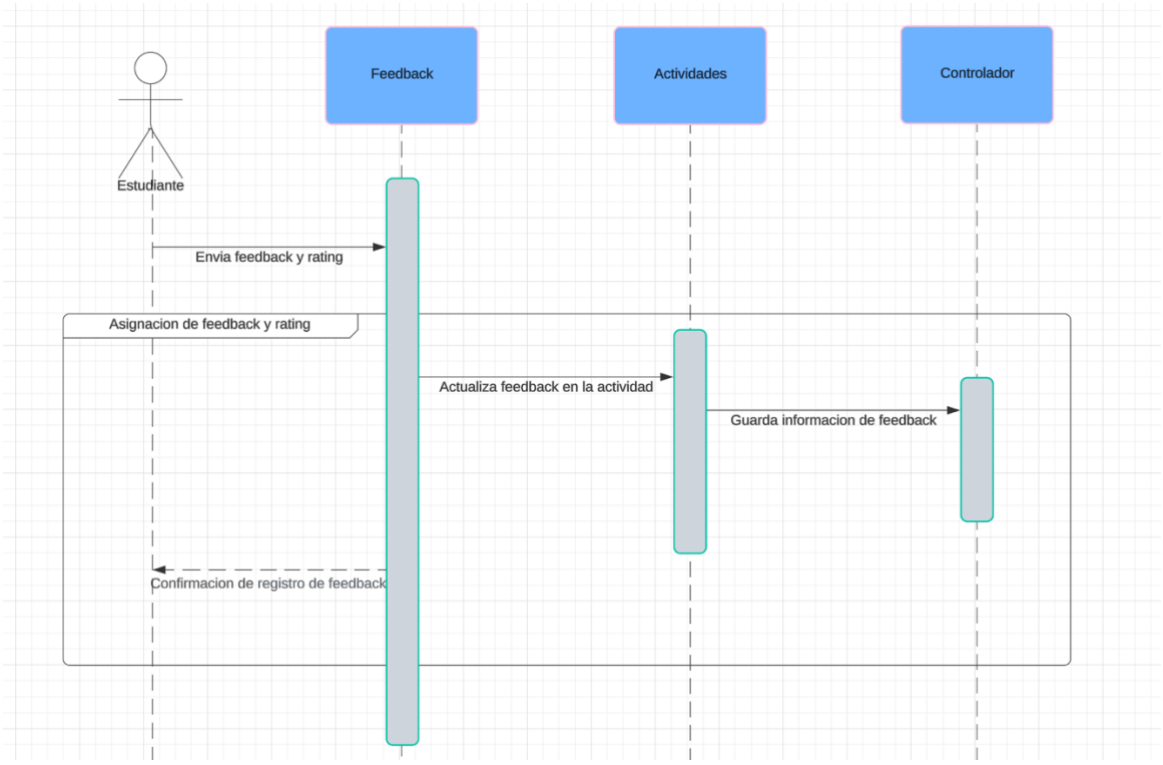
- Diagrama de secuencia para la inscripción de un estudiante en un Learning Path



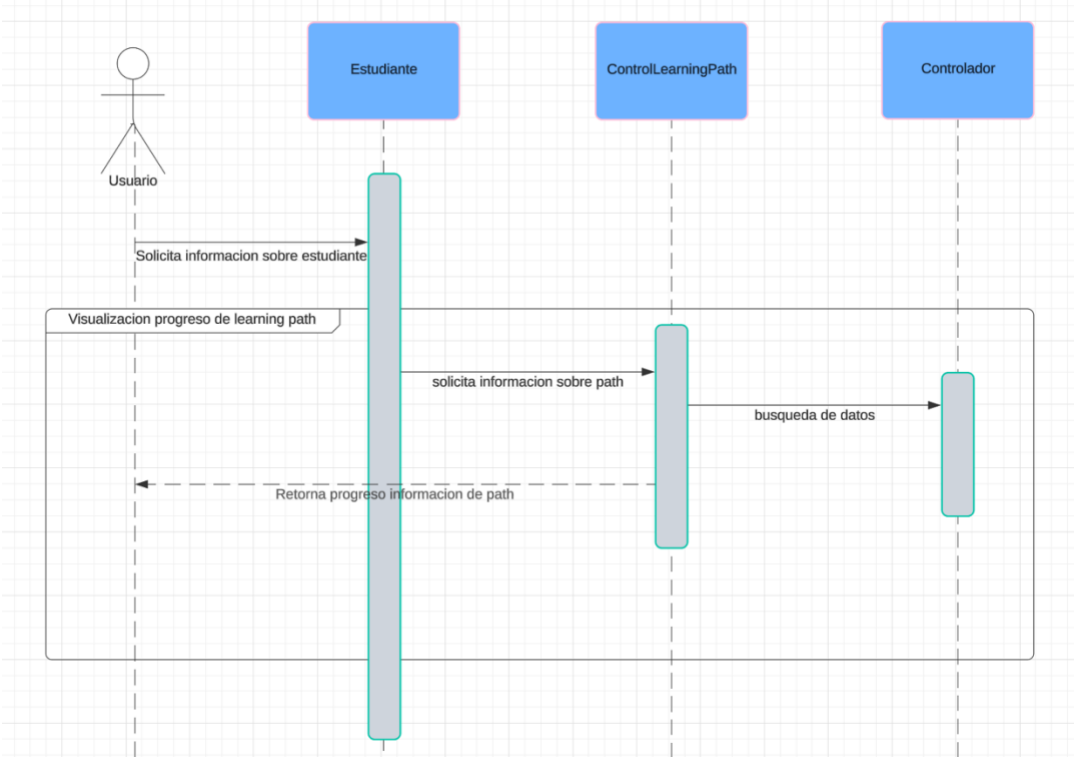
- Diagrama de secuencia de finalización de una actividad (con prerrequisitos)



- Diagrama de secuencia de asignación de feedback y rating



- Diagrama de secuencia para la visualización de progreso de un learning path



- Diagrama de secuencia para la modificación de un learning path

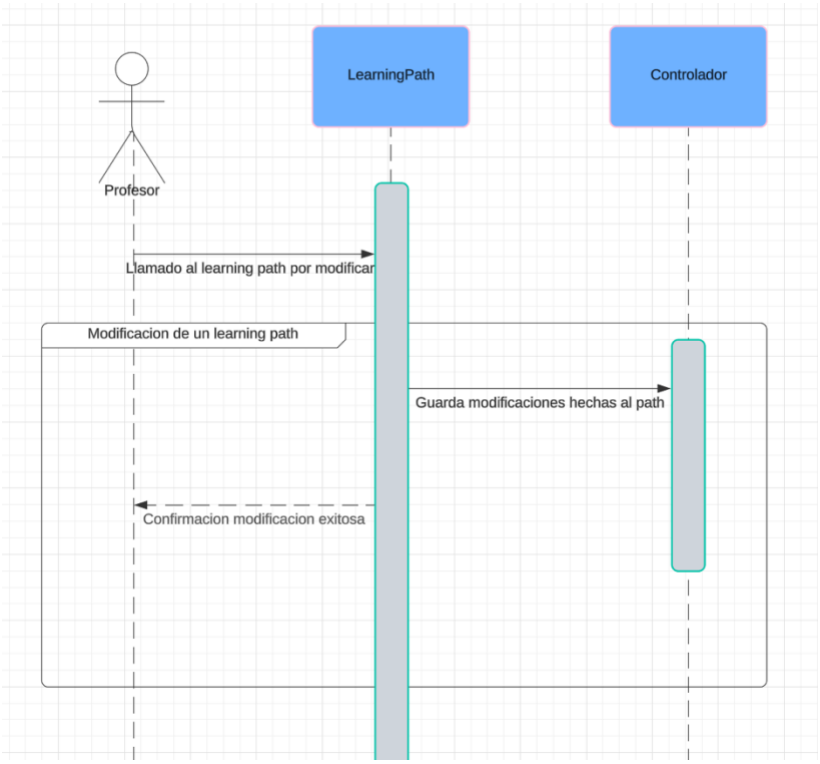
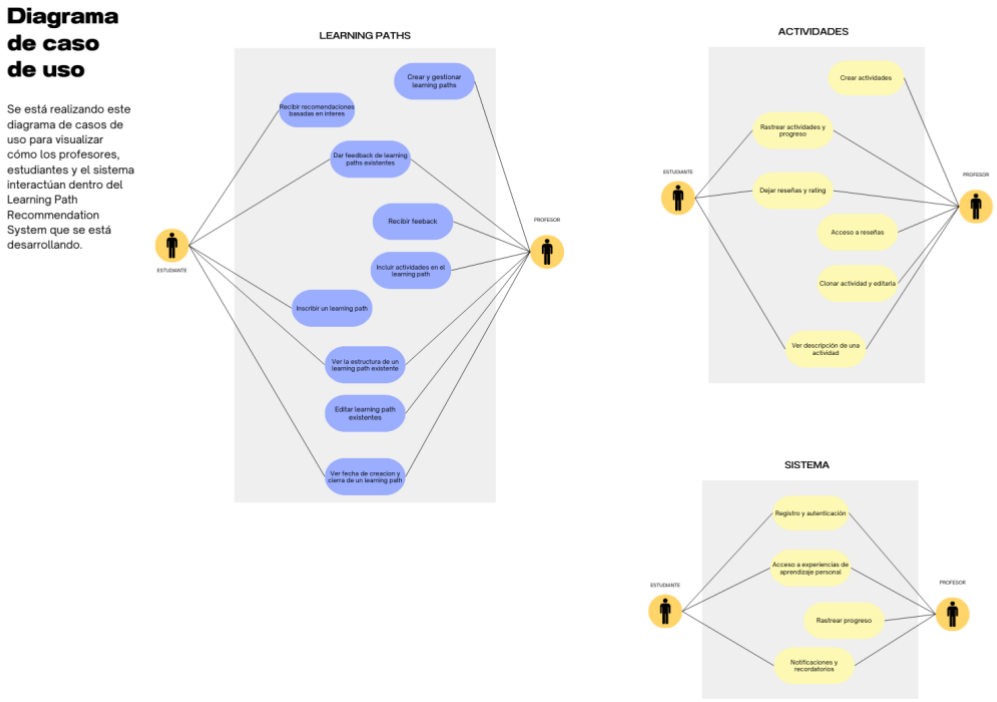


DIAGRAMA DE CASOS DE USO (interacciones entre los actores, usuarios u otros sistemas) y el sistema, mostrando qué funciones (casos de uso) pueden realizar los actores.)



link diagrama:
https://www.canva.com/design/DAGTr_390yI/pRMWO7Oxar7_AksnxrDzuw/view?utm_content=DAGTr_390yI&utm_campaign=designshare&utm_medium=link&utm_source=editor

RESTRICCIONES

1. Restricciones de Almacenamiento: El sistema debe almacenar toda la información en archivos, que pueden volverse pesados dependiendo de la cantidad de actividades y recursos incluidos. Esto puede afectar el rendimiento del sistema, por lo que es importante optimizar el espacio de almacenamiento y limitar el tamaño de los archivos subidos (máximo 100 MB). Además, se debe implementar una política de retención de datos que archive o elimine la información tras un tiempo determinado (por ejemplo, 2 años). Adicionalmente, a la hora de almacenar los datos, se debe verificar que ya no exista el nombre en el sistema, pues solo se puede tener nombres únicos.
2. Restricciones de Rendimiento: El sistema debe garantizar tiempos de carga rápidos, incluso para archivos grandes. Las operaciones críticas, como el registro de actividades y la calificación automática, deben realizarse en tiempo real sin retrasos. Asimismo, el sistema debe ser capaz de manejar múltiples usuarios concurrentes accediendo a actividades y archivos sin disminuir el rendimiento.
3. Restricciones de Seguridad: Toda la información personal y de progreso debe estar cifrada tanto en tránsito como en reposo. Los usuarios deben autenticarse antes de acceder al sistema, y los archivos subidos deben ser protegidos para evitar contenido malicioso. Además, solo los usuarios autorizados pueden acceder a áreas específicas del sistema según su rol (estudiante o profesor).

4. Restricciones de Usabilidad: El sistema debe ser accesible desde diferentes dispositivos (computadoras, tabletas, teléfonos) y navegadores. Además, los archivos pesados deben estar optimizados para facilitar su descarga en dispositivos móviles sin afectar la experiencia del usuario. El sistema también debe cumplir con normas de accesibilidad para usuarios con discapacidades.
5. Restricciones Legales y de Privacidad: El sistema debe cumplir con regulaciones como el GDPR para proteger los datos personales de los usuarios. Esto incluye garantizar que los usuarios puedan solicitar la eliminación de sus datos. Los profesores deben asegurarse de que los recursos educativos subidos respeten los derechos de autor y las leyes de propiedad intelectual.
6. Restricciones Técnicas: El sistema debe ser capaz de integrarse con plataformas LMS externas, aunque los archivos transferidos entre sistemas deben estar limitados en tamaño para evitar retrasos. Además, debe ser escalable para manejar un número creciente de usuarios y archivos sin afectar el rendimiento.
7. Restricciones de Mantenimiento: Las actualizaciones del sistema deben realizarse sin interrumpir el uso por parte de los usuarios ni comprometer la integridad de los datos. Deben existir respaldos periódicos de los archivos y la información importante para prevenir pérdidas de datos en caso de fallos técnicos.