Proyecto 1 – Diseño e Implementación

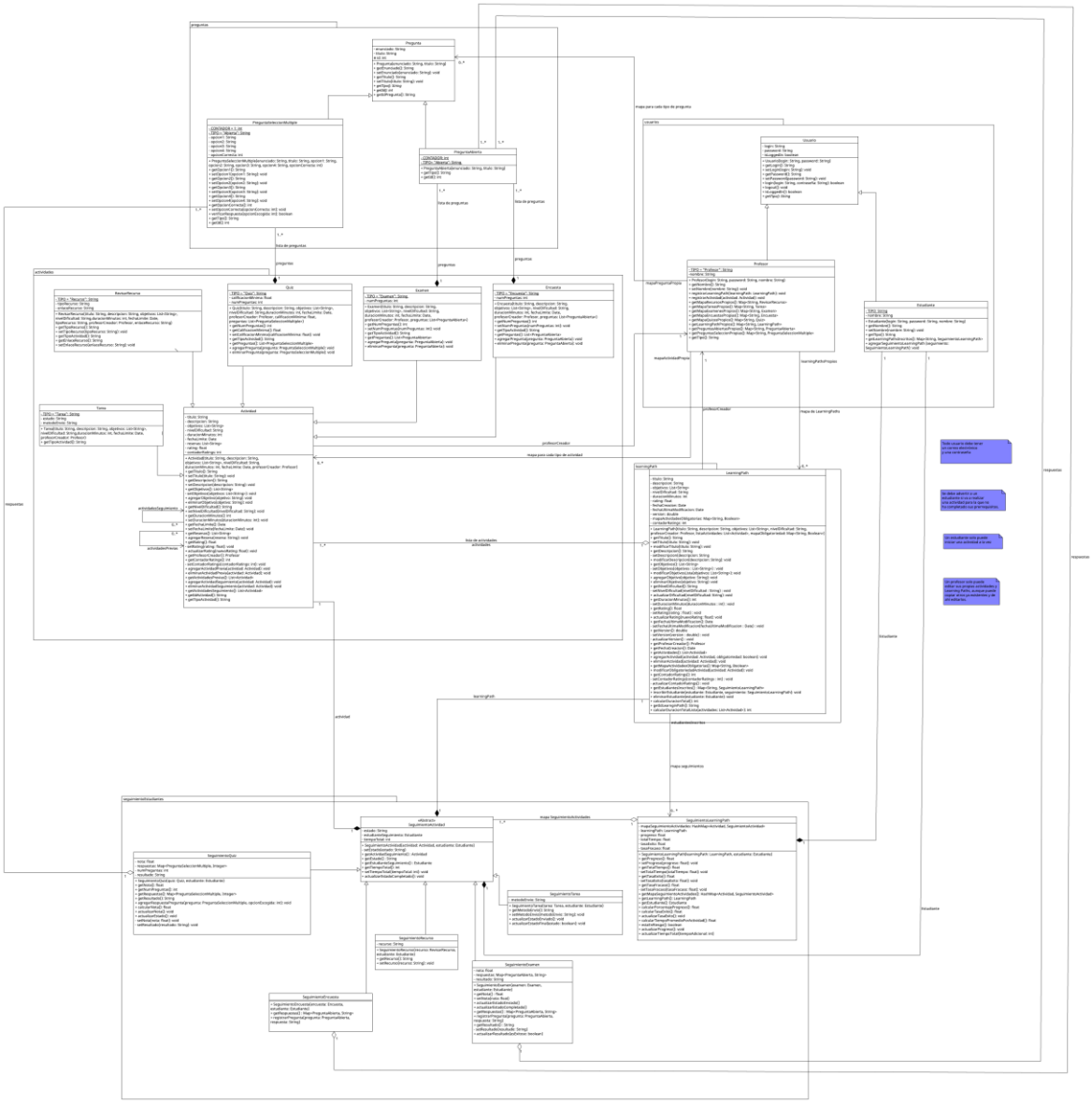
David Velásquez - 202321492

Cesar Espinosa – 202220692

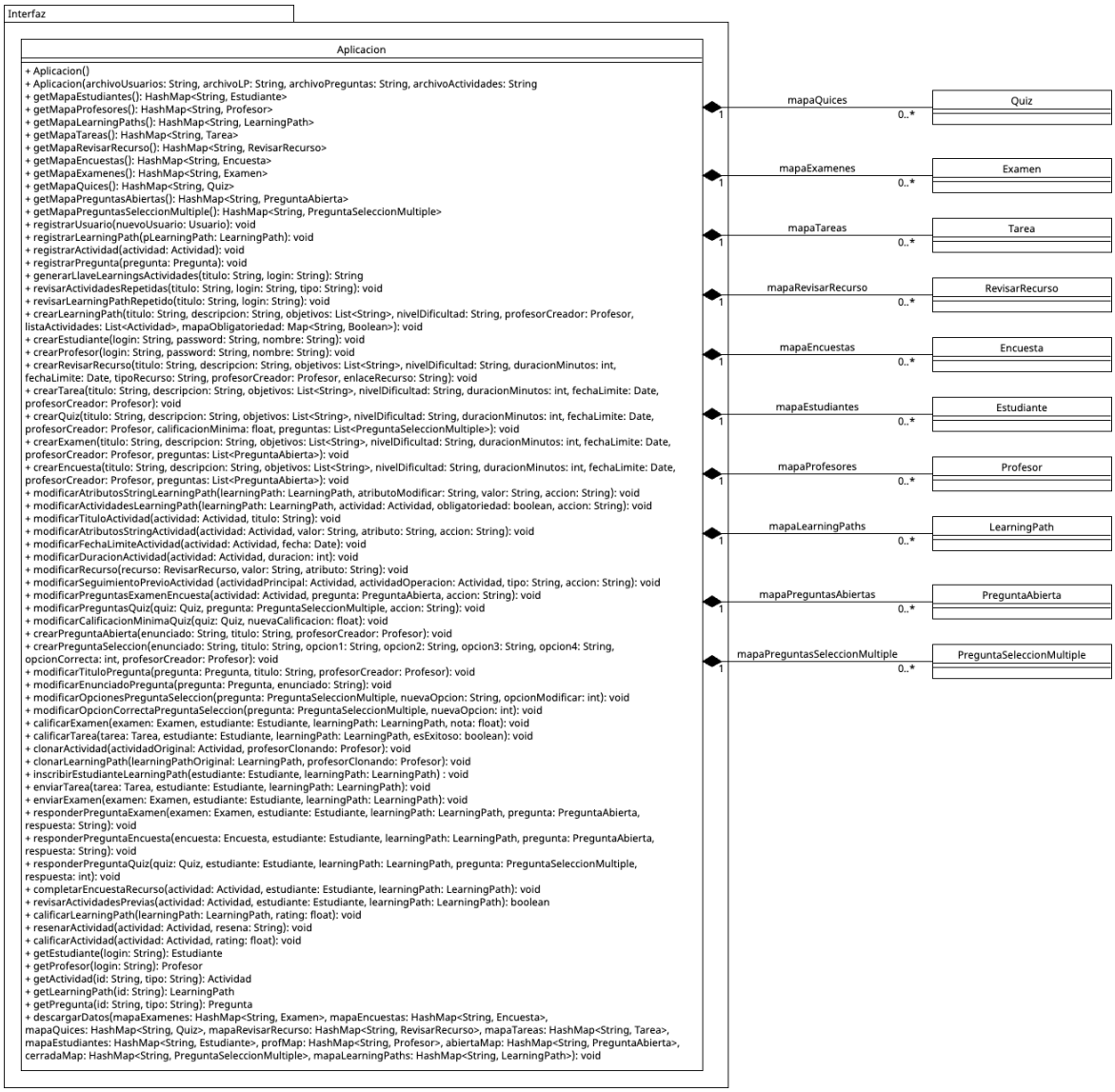
Santiago Muñoz - 202221167

Diagramas de clases

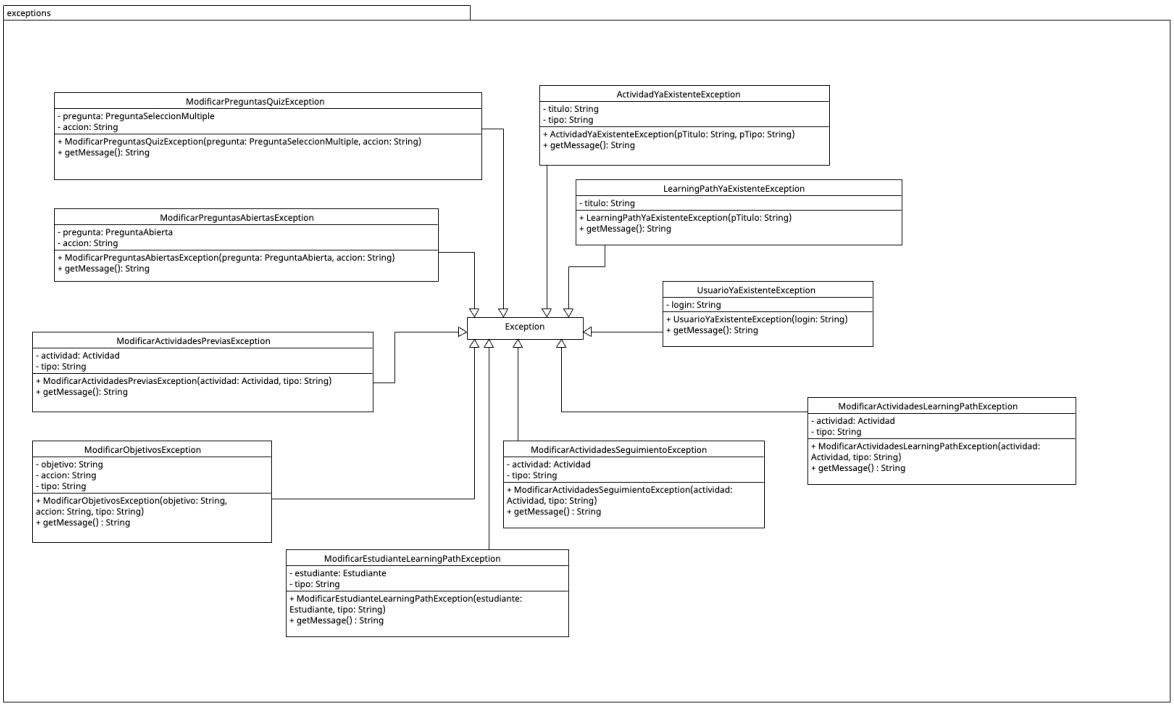
General



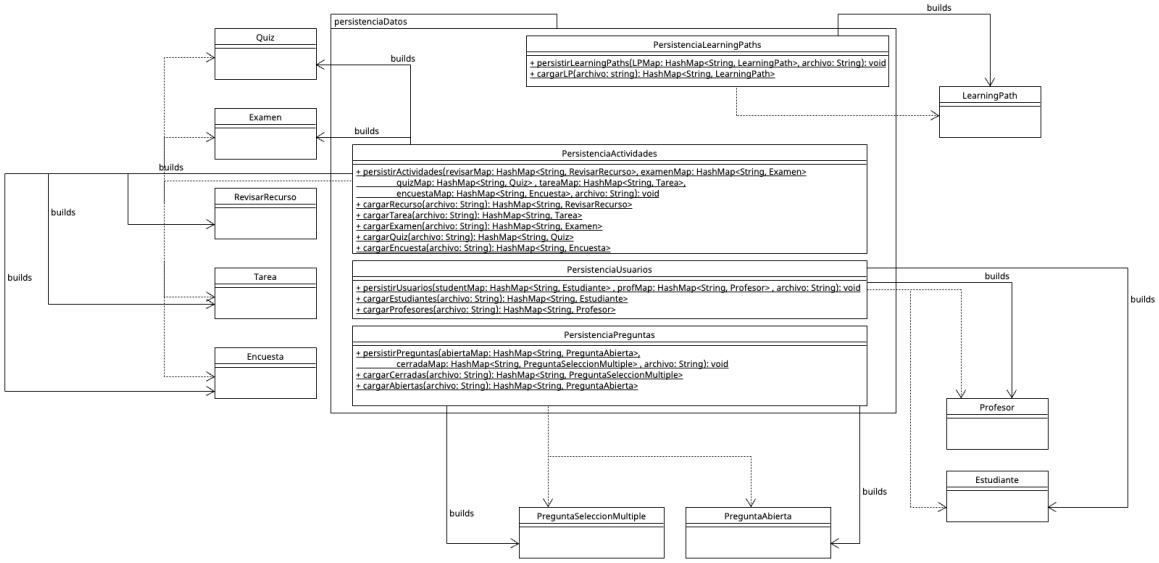
Aplicacion



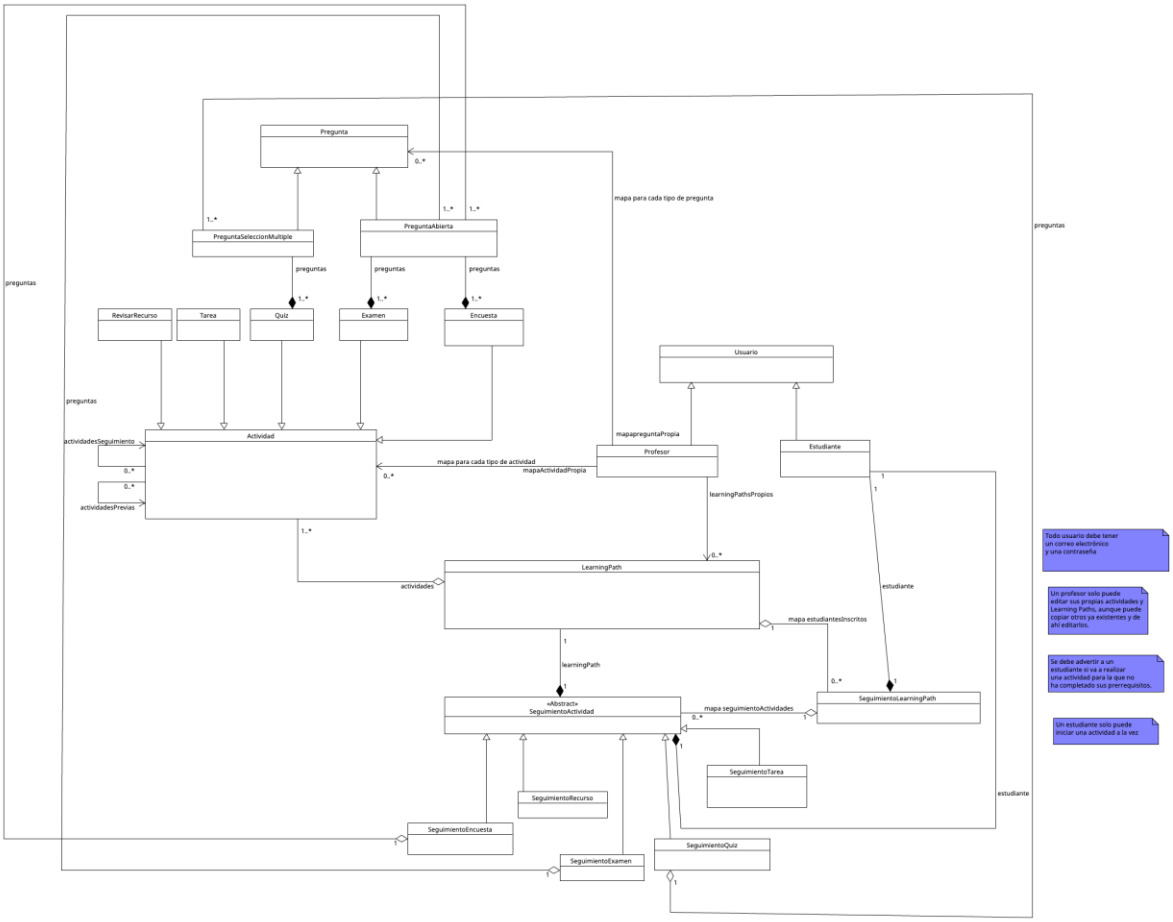
Excepciones



Persistencia

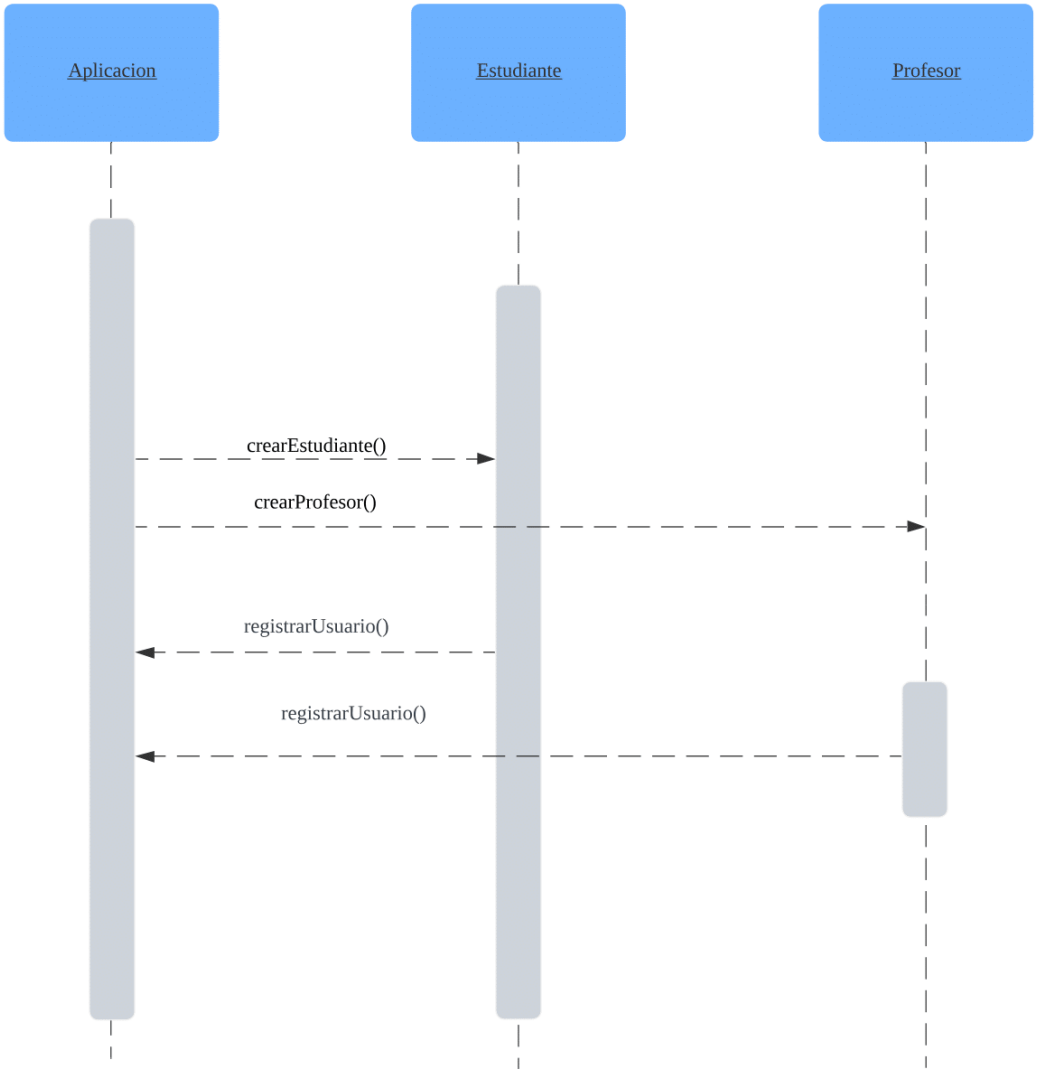


Alto Nivel

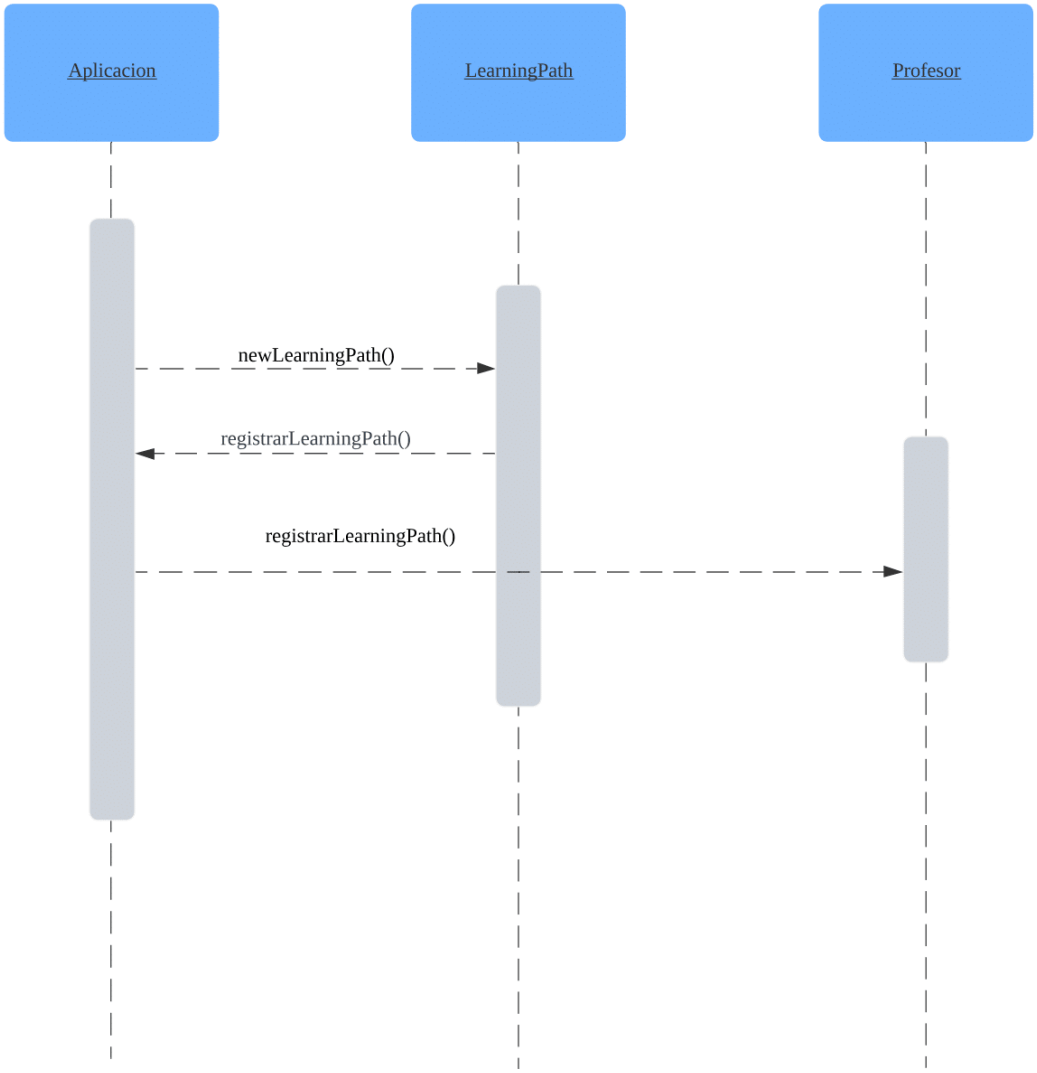


Diagramas de Secuencia

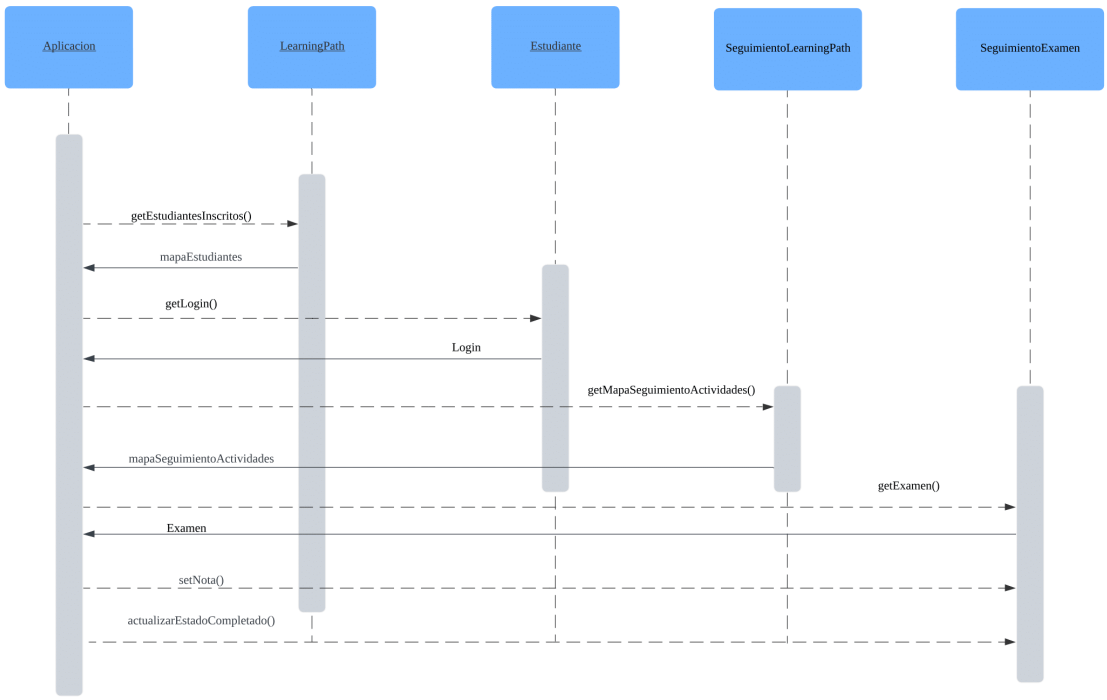
Crear Usuario



Crear LearningPath



Crear Examen



Asignación de Responsabilidades

package: user

Contiene todas las clases relacionadas al manejo de los usuarios dentro del sistema de aprendizaje.

1) Clase Usuario (Abstracta)

Responsabilidad general: Es la clase base de la que heredan tanto Estudiante como Profesor. Gestiona atributos y comportamientos básicos que todo usuario debe tener, como autenticación y manejo de credenciales.

Responsabilidades específicas:

- Autenticación de los usuarios mediante el login,
- Proveer una interfaz para obtener los datos básicos del usuario (métodos getLogin(), getPassword()),

Justificación: Se utiliza para evitar duplicar código en las subclases Estudiante y Profesor, asegurando que ambos tipos de usuarios compartan comportamientos comunes como autenticación y manejo de datos básicos.

1.1) Clase Estudiante

Responsabilidad general: Representa a los estudiantes que participan en los Learning Paths, y su responsabilidad principal es interactuar con estos caminos de aprendizaje y completar actividades.

Responsabilidades específicas:

- Inscribirse en los Learning Paths,
- Realizar actividades, como quizzes y tareas, dentro de un Learning Path,
- Seguir el progreso de su participación en los caminos de aprendizaje mediante la clase SeguimientoLearningPath.

Justificación: La clase gestiona las interacciones del estudiante con el sistema, permitiendo que se sigan y actualicen los avances de su aprendizaje.

1.2) Clase Profesor

Responsabilidad general: Los profesores son los creadores y gestores de los Learning Paths y actividades. Su función es proporcionar contenido educativo y evaluar el progreso de los estudiantes.

Responsabilidades específicas:

- Crear y gestionar Learning Paths,
- Crear, editar y gestionar actividades dentro de los caminos de aprendizaje,
- Evaluar el progreso de los estudiantes en actividades como tareas y exámenes.

Justificación: Esta clase facilita el control total del contenido educativo, permitiendo que los profesores manejen tanto las rutas de aprendizaje como las actividades y evaluaciones.

package: learningPath

Contiene todas las clases relativas al Learning Path, junto con la construcción de los métodos respectivos para manejarlo.

2) Clase LearningPath

Responsabilidad general: Representa un camino de aprendizaje que contiene una secuencia estructurada de actividades para los estudiantes.

Responsabilidades específicas:

- Gestionar las actividades que componen el camino de aprendizaje, como agregar o eliminar actividades,
- Controlar el progreso de los estudiantes inscritos,

- Proveer metadatos del Learning Path como su título, descripción, nivel de dificultad, y duración estimada.

  Justificación: El LearningPath es el núcleo del sistema, permitiendo estructurar y secuenciar actividades de forma que guíen el aprendizaje de los estudiantes.

package: actividades

Contiene todas las clases relativas a las actividades: encuestas, quices, exámenes, tareas, revisión de recursos. Implementa además los métodos para manejarlas.

3) Clase Actividad (Abstracta)

Responsabilidad general: Es la clase base para todos los tipos de actividades dentro de un Learning Path, como quizzes, tareas, revisiones de recursos, etc.

Responsabilidades específicas:

- Definir los atributos y métodos comunes a todas las actividades como título, descripción, objetivos, duración, y rating,
- Implementar el comportamiento básico de una actividad, como añadir reseñas o actualizar el rating.

Justificación: Se define como abstracta para permitir la extensión de diferentes tipos de actividades sin duplicar la lógica compartida entre ellas.

4) Subclases de Actividad (Quiz, Tarea, RevisarRecurso, Examen, Encuesta)

Responsabilidad general: Representan tipos específicos de actividades que los estudiantes deben realizar. Cada una tiene comportamientos y atributos únicos, pero heredan las funcionalidades básicas de la clase Actividad.

Responsabilidades específicas:

- Quiz: Almacenar preguntas de selección múltiple y calcular el resultado de la evaluación.
- Tarea: Permitir el envío de tareas y registro de la calificación por parte del profesor.
- RevisarRecurso: Proveer un enlace a un recurso educativo que el estudiante debe revisar.
- Examen: Almacenar preguntas abiertas y permitir su evaluación.
- Encuesta: Registrar respuestas a preguntas abiertas.

Justificación: Cada clase extiende la clase base Actividad, agregando comportamientos específicos para su tipo, lo que garantiza flexibilidad y claridad en el manejo de las actividades dentro del Learning Path.

package: seguimientoEstudiantes

Contiene todas las clases construidas para el manejo del progreso de los estudiantes dentro de las actividades y rutas de aprendizaje dadas.

5) Clase SeguimientoActividad (Abstracta)

Responsabilidad general: Es la clase base para la implementación del 'seguimiento' de las actividades. Es decir, se encarga de generar un perfil del avance de un usuario con una actividad en concreto.

Subclases:

- SeguimientoEncuesta
- SeguimientoExamen
- SeguimientoQuiz
- SeguimientoRecurso
- SeguimientoTarea

Responsabilidades específicas:

- Almacenar, actualizar y entregar el estado de la actividad,
- Almacenar el tiempo total dedicado a la actividad.

Justificación: Es fundamental para gestionar el seguimiento de los estudiantes, permitiendo medir su progreso y éxito a lo largo del Learning Path.

6) Clase SeguimientoLearningPath

Responsabilidad general: Rastrear el progreso de los estudiantes en un Learning Path, almacenando información sobre las actividades completadas, tiempos de ejecución, y tasas de éxito.

- **Responsabilidades específicas**:
- Calcular y actualizar el progreso del estudiante en el camino de aprendizaje.
- Almacenar el tiempo total dedicado y la tasa de éxito y fracaso de las actividades.

Justificación: Es fundamental para gestionar el seguimiento de los estudiantes, permitiendo medir su progreso y éxito a lo largo del Learning Path.

Justificaciones

1) Uso de Herencia en Actividad

- **Justificación**: Se implementa la herencia para centralizar la funcionalidad común a todas las actividades (título, descripción, duración, objetivos, reseñas). Esto permite que los tipos de actividades (como Quiz, Tarea, Examen) extiendan la funcionalidad

base y agreguen sus comportamientos únicos sin duplicar código. Esto también hace que el sistema sea fácilmente extensible, permitiendo añadir nuevas actividades en el futuro sin alterar el diseño existente.

2) Composición en LearningPath y Actividad

- **Justificación**: Se utiliza composición para modelar la relación entre un Learning Path y sus Actividades. Un Learning Path está compuesto por múltiples actividades, y estas pueden existir sin un Learning Path. La composición permite gestionar mejor la relación de contención entre estas entidades, donde el Learning Path agrega o elimina actividades a lo largo del tiempo.

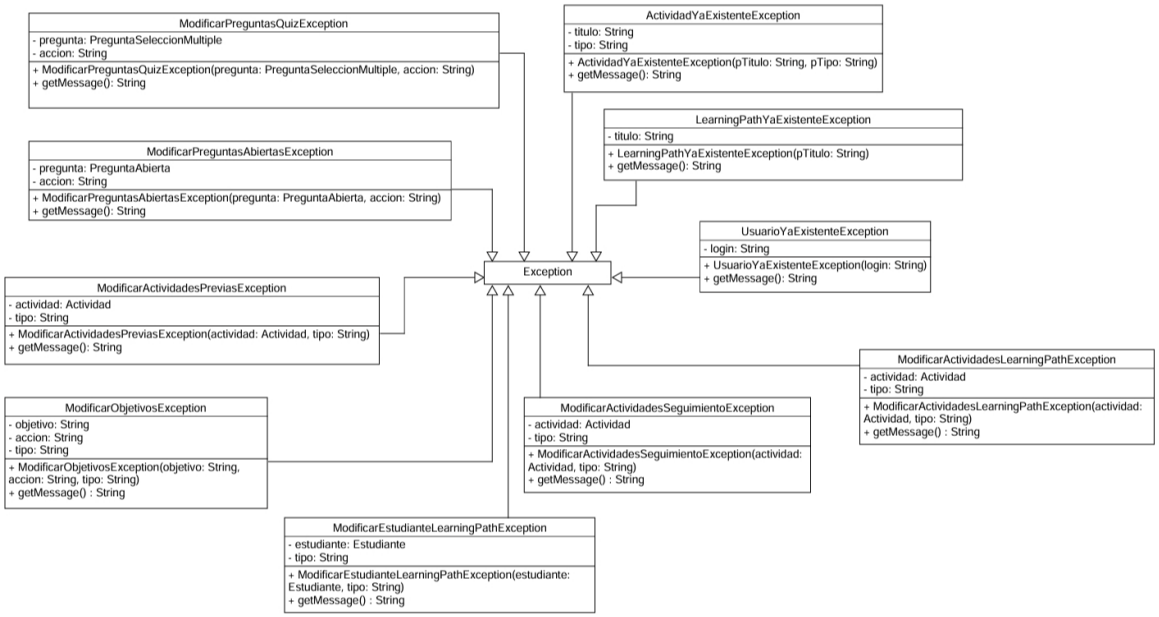
3) Manejo de Persistencia en Archivos

- **Justificación**: El diseño incluye una arquitectura de persistencia basada en archivos. Cada entidad principal (como LearningPath, Actividad, Usuario) se guarda en archivos separados. Esto se hace para mantener la modularidad y facilitar la recuperación de datos sin cargar toda la aplicación en memoria. Además, este enfoque permite expandir fácilmente la persistencia a bases de datos en el futuro sin grandes cambios estructurales en el código.

4) Flexibilidad para Futuras Expansiones

- **Justificación**: El diseño asegura que el sistema puede expandirse en el futuro. Al definir la clase Actividad como abstracta, se pueden crear fácilmente nuevos tipos de actividades sin modificar las clases ya existentes. De manera similar, la clase LearningPath puede incluir nuevas características, como la integración con sistemas externos (LMS), sin afectar las funcionalidades centrales del sistema actual.

Excepciones:



El archivo contiene varias excepciones personalizadas relacionadas con la modificación de componentes que se explican a continuación:

- **ModificarActividadesLearningPathException** es una excepción que se lanza cuando ocurre un problema al intentar modificar una actividad dentro de una ruta de aprendizaje.
- **ModificarActividadesSeguimientoException** se utiliza para manejar errores relacionados con la modificación de actividades de seguimiento.
- **ModificarEstudianteLearningPathException** es lanzada cuando hay un problema al modificar la información de un estudiante dentro de la ruta de aprendizaje.
- **ModificarObjetivosException** ocurre cuando se intenta modificar un objetivo dentro de la ruta de aprendizaje y hay un error.
- **ModificarActividadesPreviasException** gestiona los errores que ocurren cuando se intenta modificar actividades previas en la ruta de aprendizaje.
- **LearningPathYaExistenteException** es lanzada cuando se intenta crear una ruta de aprendizaje con un título que ya existe.
- **UsuarioYaExistenteException** se utiliza cuando se intenta registrar un usuario con un nombre de usuario o login que ya está en uso.
- **ModificarPreguntasAbiertasException** ocurre cuando hay un error al intentar modificar preguntas abiertas en el sistema.

- **ActividadYaExistenteException** se lanza cuando se intenta agregar una actividad que ya existe en el sistema.
- **ModificarPreguntasQuizException** maneja los errores que ocurren cuando se intenta modificar preguntas de selección múltiple dentro de un quiz.

Estas excepciones permiten manejar de manera específica los errores que pueden ocurrir al interactuar con los distintos componentes del sistema de rutas de aprendizaje.

Persistencia:

La persistencia del proyecto está constituida por cuatro clases

- PersistenciaActividades
- PersistenciaLearningPaths
- PersistenciaPreguntas
- PersistenciaUsuarios

Cada una se encarga de la persistencia de la respectiva clase por la cual recibe el nombre. Cada clase utiliza la librería GSON para implementar la persistencia. Para esto existe en cada uno de ellos dos tipos de funciones. Unas de descarga y otras de Carga.

Las funciones de descarga realizan los recorridos en las debidas estructuras (mapas de Hash) que se reciben por parámetro, genera un Objeto JSON principal y otros objetos JSON (uno por cada mapa). En cada Objeto JSON diferente al principal se añaden los objetos de una subclase específica. Finalmente, los objetos JSON secundarios se añaden al principal, donde la llave lleva el nombre correspondiente a su categoría. Así, todas las instancias de una misma superclase se añaden a un único archivo. De tal manera, surgen cuatro archivos JSON, uno por cada clase contenedora. Estos son: “actividades.json”, “lp.json”, “preguntas.json”, “usuarios.json”.

Las funciones de carga realizan la lectura de los archivos correspondientes y extraen el objeto JSON que necesiten de acuerdo con la subclase que deseen cargar. Posterior a ello, convierten el valor de la llave correspondiente de un JSON object a un objeto de la subclase determinada. Con base en eso cargan los respectivos elementos al mapa determinado para la subclase y hace el respectivo retorno del mapa correspondiente.