Camilo Molina
Samuel Rozen

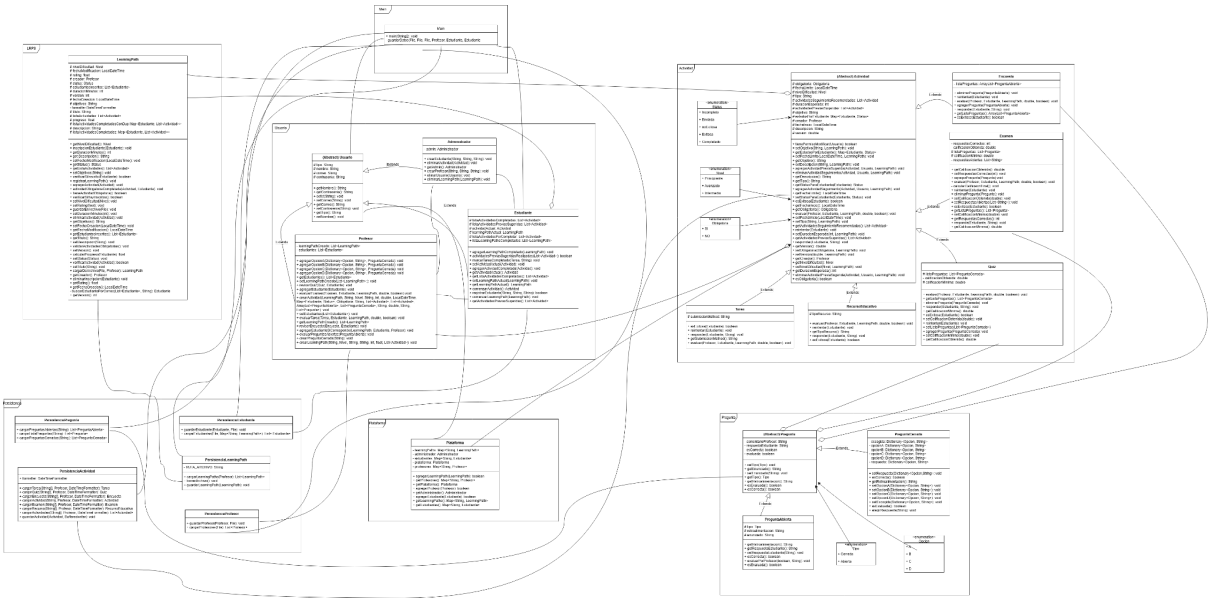
Proyecto 1 Entrega 2 - Diseño e Implementación

Introducción

El proyecto Learning Path Recommendation System tiene como objetivo ofrecer una plataforma estructurada para gestionar caminos de aprendizaje, o Learning Paths, que permitan a los estudiantes progresar a través de actividades educativas asignadas por profesores. En esta segunda entrega, nos enfocamos en implementar y probar funcionalidades clave del sistema, como la gestión de actividades, la inscripción de estudiantes, el cálculo del progreso en los Learning Paths y la persistencia de datos en archivos de texto.

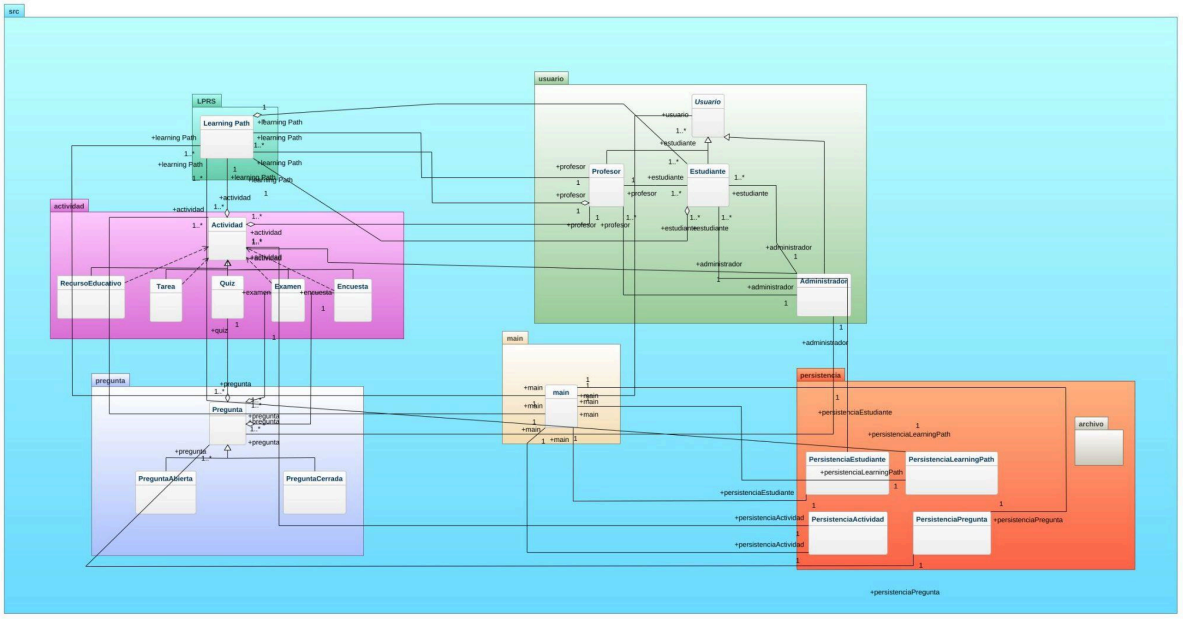
El propósito de esta fase fue estructurar la aplicación de tal manera que permitiera una interacción clara entre profesores y estudiantes. Para lograr esto, implementamos clases específicas que representan actividades, un sistema de estado que permite a los estudiantes completar o reintentar actividades, y un modelo de persistencia que guarda el estado actual de los Learning Paths, profesores y estudiantes en archivos. De este modo, cada ejecución del sistema puede mantener un historial y facilitar el seguimiento del progreso de cada estudiante en sus respectivos caminos de aprendizaje.

Con esta entrega, nos aseguramos de que el sistema no solo permita a los profesores crear y asignar actividades a los estudiantes, sino que también evalúe y registre el estado de cada actividad y camino de aprendizaje. Además, verificamos la integridad de los datos al guardar y cargar los Learning Paths y los perfiles de estudiantes y profesores, lo que garantiza una experiencia de usuario continua y confiable en cada sesión del sistema.



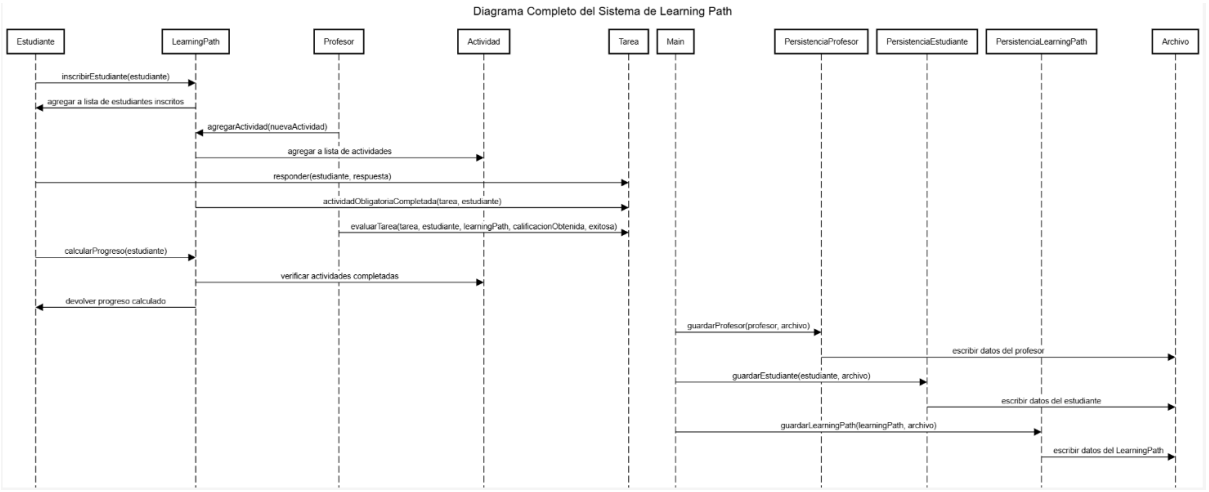
Anexo 1. Diagrama de clases

Este es un diagrama de clases que sigue la notación de UML y se basa en el modelo que habíamos desarrollado anteriormente. En esta versión, hemos incorporado nuevas relaciones reflejando los cambios realizados en la implementación, además de haber añadido los métodos específicos a las clases para detallar su comportamiento. Este diagrama proporciona una visión más precisa de la estructura del sistema y las interacciones entre sus componentes.



Anexo 2. Diagrama de clases de alto nivel

Este diagrama de clases de alto nivel representa la estructura y organización general del sistema de Learning Path, mostrando las clases principales, sus relaciones y responsabilidades clave. Su función es ilustrar cómo se distribuyen los roles principales (como profesor, estudiante y administrador), cómo se gestionan los caminos de aprendizaje y actividades, y cómo se maneja la persistencia de datos en el sistema. El diagrama proporciona una vista simplificada de la arquitectura del sistema, resaltando las interacciones esenciales sin entrar en los detalles de implementación específicos.



Anexo 3. Diagrama de secuencia de funcionalidades cruciales

Este diagrama representa el flujo principal del sistema, cubriendo las siguientes funcionalidades críticas:

1. **Inscripción de un Estudiante en un Learning Path**
    - **Descripción**: Este es el primer paso para que un estudiante participe en el sistema.
    - **Proceso**:
        - Estudiante envía una solicitud de inscripción al LearningPath.
        - LearningPath recibe la solicitud y agrega al Estudiante a su lista de estudiantes inscritos.
    - **Funcionalidad destacada**: Este proceso asegura que el estudiante esté registrado en el learning path, permitiéndole acceder a las actividades asignadas.
2. **Creación y Gestión de Actividades por el Profesor**
    - **Descripción**: Un Profesor crea y gestiona las actividades que forman parte del learning path, estructurando el contenido para los estudiantes.
    - **Proceso**:
        - Profesor crea una nueva actividad (nuevaActividad) y la agrega al LearningPath.
        - LearningPath recibe la actividad y la incorpora a su lista de actividades.
    - **Funcionalidad destacada**: Esto permite al Profesor diseñar un learning path personalizado, con actividades que guiarán el aprendizaje de los estudiantes.
3. **Responder y Evaluar una Actividad**
    - **Descripción**: Una vez inscrito, el Estudiante puede comenzar a completar las actividades, y el Profesor tiene la responsabilidad de evaluarlas.
    - **Proceso**:
        - Estudiante responde a una Tarea, enviando su respuesta.

        - LearningPath recibe la confirmación de que la Tarea fue completada como actividad obligatoria.
        - Profesor evalúa la Tarea, asignando una calificación y un estado de éxito o fracaso, según la calidad de la respuesta.
    - **Funcionalidad destacada**: Esta sección permite tanto la interacción directa del estudiante con las actividades del learning path como la retroalimentación del profesor, cerrando el ciclo de aprendizaje para cada actividad.
4. **Cálculo de Progreso del Estudiante en el Learning Path**
    - **Descripción**: Una vez que el estudiante completa actividades, el sistema debe calcular su avance para medir su progreso en el learning path.
    - **Proceso**:
        - Estudiante solicita al LearningPath calcular su progreso.
        - LearningPath revisa todas las actividades completadas (usando Actividad para verificar el estado) y calcula el porcentaje de avance.
        - LearningPath devuelve el progreso calculado al Estudiante.
    - **Funcionalidad destacada**: Esto proporciona una visión clara del avance del estudiante en el learning path, manteniendo la motivación y el control sobre el aprendizaje.
5. **Guardado y Carga de Datos en Archivos de Persistencia**
    - **Descripción**: Para asegurar la continuidad entre sesiones, el sistema debe guardar los datos de profesores, estudiantes, learning paths y actividades en archivos de persistencia.
    - **Proceso**:
        - Main solicita a cada clase de persistencia (PersistenciaProfesor, PersistenciaEstudiante, PersistenciaLearningPath) que guarde los datos actuales en un archivo.
        - Cada clase de persistencia, como PersistenciaProfesor, PersistenciaEstudiante, y PersistenciaLearningPath, escribe los datos de sus respectivos objetos (Profesor, Estudiante, LearningPath, etc.) en el archivo correspondiente.
    - **Funcionalidad destacada**: La persistencia de datos permite que el sistema retenga el estado de profesores, estudiantes, learning paths, y actividades, evitando la pérdida de información y proporcionando una experiencia continua en el sistema.

Diseño del Sistema

El diseño del sistema Learning Path Recommendation System está centrado en ofrecer una estructura que permite a los usuarios, específicamente profesores y estudiantes, interactuar de manera efectiva con los caminos de aprendizaje y sus actividades. Para ello, se optó por una arquitectura basada en objetos, donde cada entidad principal, como Profesor, Estudiante, Learning Path, y Actividad, se modela como una clase. Esto nos permite encapsular las funcionalidades de cada entidad de manera modular y facilita la extensión y el mantenimiento del código en futuras entregas.

1. Modularización de Actividades

La clase abstracta Actividad sirve como una superclase de la que heredan tipos específicos de actividades: Tarea, Quiz, RecursoEducativo, Examen, y Encuesta. Esta jerarquía se diseñó para que cada tipo de actividad tenga atributos y comportamientos específicos, como el método de entrega en Tarea o la lista de preguntas en Quiz. La herencia en este diseño nos permite gestionar fácilmente distintas reglas de negocio para cada tipo de actividad.

2. Estados y Control de Progreso

Implementamos un sistema de estados para las actividades, utilizando una enumeración Status que incluye valores como Incompleto, Completado, y NoExitosa. Cada actividad tiene un estado asociado por estudiante, gestionado a través de un HashMap que permite almacenar y consultar el estado de cada actividad para cada estudiante de manera independiente.

3. Persistencia de Datos

La persistencia de datos es clave en el sistema, ya que permite guardar y restaurar el estado de los Learning Paths, actividades y usuarios entre sesiones. Para ello, se decidió utilizar archivos de texto que almacenan datos en un formato específico, en el que cada entidad se guarda de manera estructurada. Cada vez que se realiza una acción relevante, como completar una actividad o inscribir un estudiante en un Learning Path, el sistema actualiza el archivo de persistencia.

4. Relaciones entre Clases y Cumplimiento de Restricciones de Dominio

Cada Learning Path puede tener múltiples actividades y debe cumplir con la restricción de contener al menos una actividad obligatoria. Para cumplir con esta regla, implementamos un método de validación en LearningPath que verifica la presencia de actividades obligatorias antes de registrar el Learning Path. Además, se asegura que cada estudiante esté inscrito en un solo Learning Path a la vez, lo que también está controlado mediante validaciones al momento de la inscripción.

5. Calificación y Retroalimentación en Actividades Evaluativas

Para actividades evaluativas, como Quiz y Examen, implementamos métodos de calificación específicos y mecanismos de retroalimentación que permiten al estudiante saber si su respuesta es correcta o incorrecta. Además, los quizzes cuentan con una calificación mínima requerida para ser considerados exitosos, lo cual se evalúa cada vez que un estudiante responde al quiz.

6. Cálculo del Progreso de Estudiantes

La clase LearningPath contiene un método calcularProgreso que determina el porcentaje de actividades obligatorias completadas por cada estudiante. Este método asegura que el progreso se calcula de manera individual y en función de las actividades completadas exitosamente, reflejando con precisión el avance del estudiante.

7. Flexibilidad para Reintentos y Reinscripción

El sistema permite que el estudiante reintente ciertas actividades (como Tareas) bajo condiciones específicas. Para ello, el método reintentar() se incluye en las actividades que lo permiten, junto con validaciones que aseguran que solo se pueda reintentar una actividad si no ha sido completada con éxito.

Implementación

En esta sección se describe cómo se implementó el sistema Learning Path, detallando las clases principales, el manejo de persistencia de datos, y la metodología de pruebas utilizadas para asegurar el correcto funcionamiento del sistema.

1. Descripción de Clases Principales

Clase LearningPath

La clase LearningPath representa un camino de aprendizaje estructurado que consiste en una serie de actividades que los estudiantes deben completar para alcanzar objetivos específicos. Su propósito principal es organizar y gestionar las actividades de aprendizaje, facilitando tanto el seguimiento del progreso de los estudiantes como la administración por parte del profesor.

Principales Atributos:

- **titulo, nivelDificultad, descripcion, objetivos, duracionMinutos**: Definen las características básicas del LearningPath, incluyendo su título, nivel de dificultad, descripción, objetivos y la duración esperada en minutos.
- **fechaCreacion, fechaModificacion, version**: Rastrean la creación y las modificaciones realizadas al LearningPath, junto con la versión actual.
- **status**: Indica el estado del LearningPath (por ejemplo, completo o incompleto).
- **creador**: Un objeto Profesor que representa al creador del LearningPath, quien tiene permisos para modificarlo.
- **rating**: Una calificación general del LearningPath, evaluada por los estudiantes.

- **estudiantesInscritos**: Una lista de estudiantes inscritos en el LearningPath.
- **progreso**: Representa el progreso del LearningPath para cada estudiante.
- **listaActividades**: Contiene todas las actividades que conforman el LearningPath.
- **listaActividadesCompletadasConDup y listaActividadesCompletadas**: Mapas que registran las actividades completadas por cada estudiante, permitiendo diferenciar entre actividades con duplicados y sin duplicados.

Principales Métodos:

- **Constructor**: Inicializa los atributos del LearningPath con la información básica proporcionada.
- **agregarActividad** y **eliminarActividad**: Métodos para añadir o quitar actividades en el LearningPath. Las actividades obligatorias no pueden eliminarse si no hay otras que sean obligatorias.
- **inscripcionEstudiante** y **eliminarInscripcion**: Métodos para gestionar la inscripción de estudiantes. Un estudiante solo puede inscribirse si no está inscrito en otro LearningPath.
- **actividadObligatoriaCompletada**: Marca una actividad obligatoria como completada para un estudiante específico y actualiza el estado.
- **calcularProgreso**: Calcula el progreso de un estudiante en función de las actividades obligatorias completadas. Si el progreso alcanza el 100%, se marca el LearningPath como completado.
- **guardarEnArchivo** y **cargarDeArchivo**: Métodos de persistencia para guardar y cargar el estado del LearningPath en un archivo de texto. Estos métodos incluyen el registro de las actividades completadas (con y sin duplicados) para cada estudiante.

Clase Actividad y Subclases

1. **Clase Actividad:**
    - **Propósito:** Actividad es una clase abstracta que define la estructura básica de las actividades dentro de un LearningPath, proporcionando atributos y métodos generales que las subclases especializadas implementan. Es el punto de partida para clases específicas como Examen, Encuesta, Quiz, Tarea, y RecursoEducativo.
    - **Atributos principales:**
        - Información descriptiva como descripcion, nivelDificultad, objetivo, y duracionEsperada.
        - Estados asociados a cada estudiante, gestionados en estadosPorEstudiante.
        - Atributos específicos de configuración: fechaLimite, version, obligatoria (indica si es obligatoria para completar el LearningPath).
        - Actividades previas y de seguimiento recomendadas, facilitando la creación de secuencias de aprendizaje.
    - **Métodos abstractos:**

        - **responder(Estudiante, String):** permite a los estudiantes responder la actividad. Cada subclase implementa este método para gestionar respuestas según su tipo.
        - **esExitosa(Estudiante):** determina si la actividad fue completada exitosamente para un estudiante específico.
        - **evaluar(Profesor, Estudiante, LearningPath, double, boolean):** aplica solo a actividades evaluables, permitiendo al profesor asignar una calificación y éxito.
        - **reintentar(Estudiante):** define la lógica para reintentar una actividad en caso de fracaso. Cada subclase implementa restricciones específicas de reintento.
    - **Métodos de modificación y control:** incluye métodos para cambiar atributos clave (como descripcion, nivelDificultad, objetivo, etc.) y verificaciones de permisos (tienePermisoModificar), asegurando que solo profesores o administradores puedan hacer modificaciones significativas.
2. **Clase Examen (subclase de Actividad):**
    - **Propósito:** Representa un examen compuesto de preguntas abiertas y cerradas. Se requiere una evaluación más formal, generalmente aplicada por el profesor.
    - **Atributos adicionales:**
        - **listaPreguntas:** almacena las preguntas del examen.
        - **calificacionMinima:** calificación mínima para aprobar el examen.
        - **respuestasCorrectas y calificacionObtenida:** valores para gestionar **la evaluación del estudiante.**
    - **Implementación de métodos abstractos:**
        - **responder:** procesa respuestas tanto para preguntas cerradas (que se evalúan automáticamente) como abiertas (cuyas respuestas se almacenan para evaluación manual).
        - **esExitosa:** verifica si el estudiante ha aprobado el examen basado en la calificacionObtenida.
        - **evaluar:** permite al profesor evaluar preguntas abiertas y determinar si el examen fue exitoso. Calcula la calificacionObtenida en función de respuestas correctas.
        - **reintentar:** permite que un estudiante reintente el examen si no lo ha aprobado aún.
3. **Clase Encuesta (subclase de Actividad):**
    - **Propósito:** Representa una encuesta con preguntas abiertas, diseñada para recopilar opiniones o conocimientos sin evaluación de éxito o fracaso.
    - **Atributos adicionales:**
        - **listaPreguntas:** almacena las preguntas de la encuesta, generalmente de tipo abierto.
    - **Implementación de métodos abstractos:**
        - **responder:** permite al estudiante contestar todas las preguntas de la encuesta y marca el estado como Completado.

        - **esExitosa:** solo marca la encuesta como exitosa si ha sido completada, ya que no requiere evaluación formal.
        - **evaluar y reintentar:** no aplican, pues las encuestas no son evaluables ni repetibles.
4. **Clase Quiz (subclase de Actividad):**
    - **Propósito:** Representa un cuestionario de preguntas cerradas con una calificación mínima para aprobar. Su evaluación es automática.
    - **Atributos adicionales:**
        - **listaPreguntas:** contiene las preguntas cerradas del quiz.
        - **calificacionMinima y calificacionObtenida:** se usan para definir el umbral de éxito y calcular la nota obtenida.
    - **Implementación de métodos abstractos:**
        - **responder:** procesa y evalúa automáticamente las respuestas, comparándolas con la calificacionMinima.
        - **esExitosa:** verifica si el estudiante ha alcanzado la calificación mínima y marca el quiz como exitoso.
        - **reintentar:** permite al estudiante intentar de nuevo si no ha aprobado.
        - **evaluar:** no aplica, ya que el quiz se evalúa automáticamente sin intervención del profesor.
5. **Clase RecursoEducativo (subclase de Actividad):**
    - **Propósito:** Representa recursos como videos, documentos, etc., que los estudiantes deben revisar para completar el LearningPath.
    - **Atributos adicionales:**
        - **tipoRecurso:** define el tipo de recurso (ej., video, documento).
    - **Implementación de métodos abstractos:**
        - **responder:** permite que el estudiante marque el recurso como revisado mediante la respuesta "visto", cambiando el estado a Completado.
        - **esExitosa:** verifica si el estudiante ha marcado el recurso como revisado.
        - **evaluar y reintentar:** no aplican, pues los recursos educativos no requieren evaluación ni reintento.
6. **Clase Tarea (subclase de Actividad):**
    - **Propósito:** Representa una tarea que el estudiante debe enviar, y que el profesor debe evaluar manualmente.
    - **Atributos adicionales:**
        - **submissionMethod:** método de entrega de la tarea (ej., LMS, correo).
    - **Implementación de métodos abstractos:**
        - **responder:** permite al estudiante enviar la tarea, indicando el método de entrega.
        - **esExitosa:** verifica si el profesor ha evaluado la tarea como exitosa.
        - **evaluar:** permite que el profesor califique la tarea y determine su éxito.
        - **reintentar:** permite al estudiante reintentar enviar la tarea si no ha sido aprobada.

Clase Usuario (Clase Abstracta)

Propósito: Usuario es una clase abstracta que define la estructura base de los usuarios del sistema, proporcionando atributos y métodos comunes que heredan todas sus subclases. No puede ser instanciada directamente, solo a través de sus subclases como Profesor, Estudiante y Administrador**.**

Atributos principales:

- **nombre:** Nombre del usuario.
- **contrasenia:** Contraseña para autenticación.
- **tipo:** Tipo de usuario en el sistema (por ejemplo, "profesor", "estudiante", "admin").
- **correo:** Correo electrónico para comunicación y notificaciones.

Métodos principales:

- **getNombre(), getContrasenia(), getTipo(), getCorreo():** Métodos para obtener los atributos del usuario.
- **setNombre(String nombre), setContrasenia(String contrasenia), setTipo(String tipo), setCorreo(String correo):** Métodos para modificar los atributos del usuario.

Clase Profesor (Subclase de Usuario)

Propósito: Profesor representa a los usuarios encargados de crear, gestionar y evaluar los LearningPaths y sus actividades. Esta clase permite la creación de actividades y su evaluación cuando es necesario.

Atributos principales:

- **learningPathCreado:** Lista de LearningPaths creados por el profesor.
- **estudiantes:** Lista de Estudiantes con quienes el profesor interactúa a través de los learning paths que ha creado.

Métodos principales:

- **evaluarTarea(Tarea tarea, Estudiante estudiante, LearningPath learningPath, double calificacionObtenida, boolean exitosa):** Permite evaluar una tarea asignada a un estudiante.
- **evaluarExamen(Examen examen, Estudiante estudiante, LearningPath learningPath, double calificacionObtenida, boolean exitosa):** Permite evaluar un examen del estudiante, determinando si fue exitoso.
- **crearLearningPath(String titulo, Nivel nivelDificultad, String descripcion, String objetivos, int duracionMinutos, float rating, List<Actividad> listaActividades):** Permite la creación de un LearningPath.
- **crearActividad(...):** Método para crear y agregar una actividad específica a un learning path.

- **agregarEstudianteSiCorresponde(LearningPath learningPath, Estudiante estudiante, Profesor profesor):** Agrega a un estudiante a la lista del profesor si está inscrito en uno de sus learning paths.

Clase Estudiante (Subclase de Usuario)

Propósito: Estudiante representa a los usuarios que se inscriben en los learning paths y avanzan a través de las actividades asignadas. Esta clase controla el progreso del estudiante dentro de un learning path específico.

Atributos principales:

- **actividadActual:** La actividad en la que el estudiante está trabajando en el momento.
- **learningPathActual:** Learning path en el que el estudiante está inscrito actualmente.
- **listaLearningPathsCompletados:** Lista de learning paths que el estudiante ha completado.
- **listaActividadesCompletadas:** Actividades completadas por el estudiante en su learning path actual.

Métodos principales:

- **comenzarLearningPath(LearningPath learningPath):** Permite al estudiante comenzar un learning path si no tiene otro en curso.
- **marcarTareaCompletada(Tarea tarea, String submissionMethod):** Marca una tarea como completada y enviada al profesor.
- **agregarLearningPathCompletado(LearningPath learningPathActual):** Añade un learning path a la lista de completados una vez que el estudiante termina todas las actividades requeridas.
- **comenzarActividad():** Comienza una actividad en curso y verifica que se cumplan los prerrequisitos.

Clase Administrador (Subclase de Usuario)

Propósito: Administrador es un usuario con permisos especiales para gestionar todas las entidades del sistema, pero no interactúa directamente con las actividades académicas ni el progreso de los estudiantes.

Atributos principales:

- **admin:** Instancia única de Administrador, asegurada mediante un patrón Singleton.

Métodos principales:

- **crearProfesor(String nombre, String contrasenia, String correo):** Crea un nuevo usuario de tipo Profesor.

- **crearEstudiante(String nombre, String contrasenia, String correo):** Crea un nuevo usuario de tipo Estudiante.
- **eliminarUsuario(Usuario usuario):** Elimina un usuario del sistema.
- eliminarLearningPath(LearningPath learningPath): Permite la eliminación de un LearningPath.
- **eliminarActividad(Actividad actividad):** Permite eliminar una actividad del sistema.

2. Persistencia de Datos

La persistencia de datos se maneja mediante archivos de texto que almacenan información estructurada sobre los profesores, estudiantes y *Learning Paths*. Cada entidad tiene un formato específico para su almacenamiento en el archivo, permitiendo que el sistema pueda guardar y cargar su estado de manera eficiente.

- **Formato de Archivo y Estructura**
    - **Profesor:** Se guarda el nombre, contraseña, correo electrónico, y los *Learning Paths* que ha creado.
    - **Estudiante:** Se guarda el nombre, contraseña, correo electrónico, y el *Learning Path* actual (si lo hay), junto con la lista de *Learning Paths* completados.
    - **LearningPath:** Se almacena el título, nivel de dificultad, descripción, objetivos, duración, y las actividades que contiene, cada una con su propio estado para cada estudiante inscrito.

Este formato asegura que cada entidad puede ser cargada de manera eficiente, y que el progreso de cada estudiante en cada actividad está claramente identificado.

3. Metodología de Pruebas

Para asegurar el correcto funcionamiento del sistema, se implementaron pruebas en el método main que simulan la interacción entre profesores, estudiantes y *Learning Paths*. Estas pruebas cubren las funcionalidades clave del sistema.

- **Pruebas en el Main**
  En el archivo Main, se realizan pruebas que validan los siguientes aspectos:
    - **Inscripción de estudiantes en un Learning Path:** Se verifica que los estudiantes puedan inscribirse en un *Learning Path* creado por un profesor y que el sistema respete la restricción de inscripción única.
    - **Ejecución y evaluación de actividades:** Se prueba que los estudiantes puedan responder a cada tipo de actividad, que las respuestas se registren correctamente, y que las actividades evaluativas, como Quiz y Tarea, se califiquen y retroalimenten correctamente.

    - **Cálculo de progreso:** Se calcula el progreso del estudiante en el *Learning Path* y se verifica que este porcentaje sea correcto según las actividades obligatorias completadas.
    - **Persistencia de datos:** Se guarda el estado del sistema en archivos de texto y se carga nuevamente para validar que la información se almacena y restaura de forma precisa.

Estas pruebas comprueban cada aspecto clave del sistema, asegurando que todas las interacciones entre las clases funcionan como se espera.

4. Instrucciones para Ejecutar el Main en Eclipse

- **Paso 1: Abrir el Proyecto**
    - En Eclipse, selecciona File > Open Projects from File System…
    - Selecciona la carpeta raíz del proyecto en tu sistema de archivos y haz clic en Finish para importar el proyecto.
- **Paso 2: Configurar el Archivo Main**
    - En la vista de Package Explorer, encuentra el archivo Main.java en el paquete main.
    - Haz clic derecho sobre Main.java y selecciona Run As > Java Application.
- **Paso 3: Ejecutar el Programa**
    - Eclipse ejecutará el archivo Main.java, y deberías ver los resultados en la consola de Eclipse.
    - La salida de la consola incluirá mensajes de registro que muestran el estado de inscripción, respuestas de actividades, y los resultados de las pruebas de persistencia.
- **Paso 4: Revisar los Archivos de Persistencia**
    - Después de ejecutar el programa, abre los archivos de persistencia generados (profesores.txt, estudiantes.txt, learningPaths.txt) en la carpeta src/persistencia/archivo. Si no le aparece en Eclipse busquelos desde el explorador de archivos en esta misma carpeta.
    - Verifica que los datos de los profesores, estudiantes, y *Learning Paths* se hayan guardado de acuerdo con la estructura de archivo descrita anteriormente.

Nota: Todos los anexos, para mejor visualización, se encuentran también adjuntados en la carpeta donde se encuentra este documento.