Análisis – Proyecto 1:

Juan José Cortés Villamil – 202325148

Alejandro Franco - 202225577

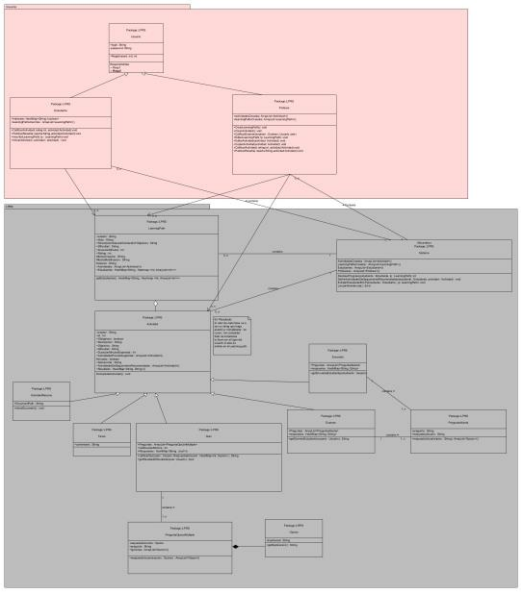
Amelia Serrano Arango- 202221556

Lerning Path Recommendation System:

La aplicación que se nos pide desarrollar busca crear un entorno educativo en el que estudiantes puedan buscar Learning Paths que sean de su interés. Estos Learning Paths, creados por los profesores, consisten en una serie de actividades que el estudiante deberá realizar, o no, depende de cómo haya sido creada la actividad. Las actividades pueden ser sencillamente recursos que consumir, tareas, quices, exámenes o encuestas.

Ahora, adicional a lo anterior debe haber una entidad de sistema que se encargara de rastrear lo siguiente: el progreso de los estudiantes en un learning path, cuando se inició un learning path, cuando se completó, tiempo dedicado por actividad para cada usuario, tasas de éxito y de fracaso por actividad y cuando fueron completadas cada una de las actividades por cada uno de los usuarios. La entidad sistema se encarga de vigilar que todo actúe de acuerdo con lo esperado. Es el sistema el que va a notificarle al estudiante que, si bien puede hacer alguna actividad avanzada y saltarse actividades, que lo mejor sería hacerlas en orden. Es el sistema el que también va a notificar siempre que se dé un error en la creación de learning paths y de actividades.

Modelo de dominio (Pueden encontrar la imagen con mejor calidad en la carpeta del repositorio):



Descripción del modelo de dominio:

- *Clase: LearningPath:*

Atributos:

- título: String → El título que identifica al Learning Path.
- `creador: String` → El nombre del profesor que creó el Learning Path.
- descripcionGeneralContenidoYObjectos: String → Una descripción general sobre el contenido y los objetivos del Learning Path.
- nivelDificultad: String → Una palabra que indica la dificultad (por ejemplo: básico, intermedio, avanzado).
- Rating: int -> La cantidad de estrellas que tiene el LearningPath dentro de la comunidad.
- duracion: int → La duración estimada en minutos para completar todo el Learning Path.
- fechaCreacion: string → La fecha en la que se creó el Learning Path.
- fechaModificacion: string → La fecha de la última modificación realizada al Learning Path.
- version: int → El número de versión del Learning Path.
- progreso: float → El porcentaje de actividades obligatorias completadas por el estudiante.
- actividades: List<Actividad> → Lista de actividades que componen el Learning Path
- `estudiantes: HashMap<Estudiante, Progreso>` → Mapa que relaciona estudiantes con su progreso en el Learning Path.

Métodos:

- `getEstudiantes(): HashMap<Estudiante, Progreso>` → Devuelve el mapa de estudiantes inscritos y su progreso en el Learning Path.

- Clase: Actividad (abstracta)

- **Atributos:**
  - id: int → Identificador único de la actividad.
  - título: String → El título que identifica la actividad.
  - descripcion: String → Descripción de lo que trata la actividad.
  - objetivo: String → El objetivo específico que el estudiante debe alcanzar al completar la actividad.
  - nivelDificultad: String→ Nivel de dificultad de la actividad.
  - duracion: int → Tiempo estimado en minutos para completar la actividad.
  - actividadPreviaSugerida: Actividad → Actividad previa que el estudiante debería completar antes.
  - obligatoria: boolean → Define si la actividad es obligatoria o no dentro del Learning Path.
  - `creador: String` → El nombre del profesor que creó la actividad.
  - `fechaLimite: String` → Fecha límite para completar la actividad.
  - ActividadesDeSeguimientoRecomendadas: List → Lista de actividades para recomendadas.
  - **`resultado: HashMap<Estudiante, String>`** → Un mapa que asocia a cada estudiante con el resultado de la actividad (por ejemplo, "completado", "pendiente", "fallido").

- **Métodos**:
  - **`completarActividad(): void`** → Permite al sistema marcar una actividad como completada por un estudiante y actualizar su resultado en el `HashMap`.

- Clase: Tarea (extiende de Actividad)

- **Atributos**:
  - **`comentario: String`** → Comentario del profesor acerca de la tarea entregada por el estudiante.

- Clase: Quiz (extiende de Actividad)

- **Atributos**:

- preguntas: List<PreguntaOpcionMultiple> → Lista de preguntas de opción múltiple que conforman el quiz.
- calificacionMinima: int → Calificación mínima necesaria para aprobar el quiz.
- **`respuestas: HashMap<Estudiante, List<Opcion>>`** → Un mapa que asocia a cada estudiante con sus respuestas seleccionadas en el quiz.

- **Métodos**:
  - **`calificar()`** → Calcula la calificación del estudiante con base en las respuestas proporcionadas.
  - **`getResultadoEstudiante(Estudiante): boolean`** → Retorna `true` si el estudiante aprobó el quiz, y `false` si no lo aprobó.

- Clase: PreguntaOpciónMultiple

- **Atributos**:
  - **`pregunta: String`** → El texto de la pregunta de opción múltiple.
  - **`opciones: ArrayList<Opcion>`** → Lista de opciones disponibles para la pregunta.
  - **`respuestaCorrecta: Opcion`** → La opción que es correcta.

- **Métodos**:
  - **`respuestaUsuario(): ArrayList<Opcion>`** → Retorna la lista de opciones seleccionadas por el estudiante como respuesta a la pregunta.

- Clase: Opcion

- **Atributos**:

  - **`explicacion: String`** → Explicación de por qué la opción es correcta o incorrecta.

- **Métodos**:

  - **`getExplicacion(): String`** → Retorna la explicación de la opción.

- Clase: ActividadRecurso (extiende de Actividad)

- **Atributos**:

    - **`documentPath: String`** → Ruta del archivo o recurso asociado a la actividad (por ejemplo, un PDF, video, o sitio web).
- **Métodos**:
    - **`showDocument(): void`** → Muestra o abre el documento asociado al recurso para que el estudiante lo pueda visualizar.

- Clase: Examen (extiende de Actividad)

- **Atributos**:
    - **`preguntas: ArrayList<PreguntaAbierta>`** → Lista de preguntas abiertas que conforman el examen.
    - **`respuestas: HashMap<Estudiante, String>`** → Un mapa que asocia a cada estudiante con su respuesta en el examen.
- **Métodos**:
    - **`getExamenEstudiante(Estudiante): String`** → Retorna la respuesta del estudiante para las preguntas del examen.

- Clase: PreguntaAbierta

- **Atributos**:
    - **`pregunta: String`** → El enunciado de la pregunta abierta.
    - **`respuestaUsuario: String`** → La respuesta proporcionada por el estudiante a la pregunta.

- **Métodos**:
    - **`respuestaUsuarioTexto(): ArrayList<String>`** → Retorna la respuesta proporcionada por el estudiante como un texto.

- Clase: Encuesta (extiende de Actividad)

- **Atributos**:
    - **`preguntas: ArrayList<PreguntaAbierta>`** → Lista de preguntas abiertas que forman la encuesta.
    - **`respuestas: HashMap<Estudiante, List<String>>`** → Un mapa que asocia a cada estudiante con sus respuestas a las preguntas de la encuesta.

- **Métodos**:
    - **`getEncuestaEstudiante(Estudiante): List<String>`** → Retorna la lista de respuestas proporcionadas por el estudiante para la encuesta.

- Clase: Sistema

- **Atributos**:
    - **`actividadesCreadas: ArrayList<Actividad>`** → Lista de actividades que han sido creadas en el sistema.
    - **`learningPathsCreacion: ArrayList<LearningPath>`** → Lista de Learning Paths que se están creando o han sido creados en el sistema.
    - **`estudiantes: ArrayList<Estudiante>`** → Lista de estudiantes registrados en el sistema.
    - **`profesores: ArrayList<Profesor>`** → Lista de profesores registrados en el sistema

- **Métodos**:
    - **`rastrearProgreso(Estudiante, LearningPath): void`** → Rastrear el progreso de un estudiante en un Learning Path específico.
    - **`definirActividadesDeSeguimientoRecomendadasEstudiante(): void`** → Define las actividades recomendadas para el seguimiento de estudiantes según su rendimiento.
    - **`enlistarEstudiantesEnLP(LearningPath): void`** → Inscribe a los estudiantes en un Learning Path específico.
    - **`lanzarAdvertencia(String mensaje): void`** → Lanza una advertencia al sistema con un mensaje específico (por ejemplo, cuando hay un error en la creación de actividades).

- Clase: Usuario (abstracta)

- **Atributos**:
    - **`login: String`** → Nombre de usuario para autenticarse en el sistema.
    - **`password: String`** → Contraseña asociada al nombre de usuario

- **Métodos**:

    - **`registrarse(): void`** → Permite que un nuevo usuario se registre en el sistema.

- Clase: Estudiante (extiende de Usuario)

- **Atributos**:
  - **`intereses: HashMap<String, String>`** → Un mapa que almacena los intereses del estudiante, donde la clave es el área de interés y el valor es una descripción o categoría relacionada.
  - **`learningPathsInscritos: ArrayList<LearningPath>`** → Lista de Learning Paths en los que el estudiante está inscrito.

- **Métodos**:
  - **`calificarActividad(Actividad): void`** → Permite al estudiante calificar una actividad completada.
  - **`publicarReseña(String reseña): void`** → Permite al estudiante publicar una reseña sobre una actividad o Learning Path.
  - **`inscribirLearningPath(LearningPath): void`** → Inscribe al estudiante en un Learning Path específico.
  - **`iniciarActividad(Actividad): void`** → Permite al estudiante iniciar una actividad dentro de un Learning Path.

- Clase: Profesor (extiende de Usuario)

- **Atributos**:
  - **`learningPathsCreados: List<LearningPath>`** → Lista de Learning Paths creados por el profesor.
  - **`ActividadesCreadas: ArrayList<Actividad>()`** → Lista de actividades creados por el profesor.

- **Métodos**:
  - **`crearLearningPath()`** → Permite que el profesor cree un nuevo Learning Path.
  - **`crearActividad()`** →Permite que el profesor cree una nueva actividad
  - **`calificarExamen(Examen examen, Usuario user)`** → Permite que el profesor califique el examen y cambie el estado de la actividad para el estudiante
  - **`editarActividad(Actividad actividad)`** → Permite que el profesor modifique una actividad de su propio Learning Path.

- **`EditarLearningPath(Ip learningParth)`** → Permite que el profesor modifique su propio Learning Path.
- **`CopiarActividad(Actividad actividad)`** → Duplica una actividad
- **`CalificarActividad(Actividad actividad)`** → Permite que el profesor califique la actividad y cambie el estado de la actividad para el estudiante.
- **`PublicarReseña(Actividad actividad)`** → Permite al profesor agregar una reseña a la actividad.

Restricciones del proyecto:

- Solo se puede iniciar una actividad a la vez.
- Un learning path debe tener al menos una actividad.
- Los únicos tipos de usuarios son Estudiante y Profesor.
- No pueden haber ni más ni menos de 4 opciones en una pregunta de opción múltiple.
- En una pregunta de opción múltiple, solo una de las respuestas puede ser correcta.
- En una pregunta de opción múltiple, las opciones no pueden repetirse.
- Las opciones de las preguntas de una opción múltiple tienen que contar con una explicación de porque son correctas o incorrectas.
- Una actividad o learning path solo puede ser modificada por su creador.
- Todo usuario tiene un login y una contraseña.
- Un estudiante no debe poder crear Learning Paths ni actividades.
- Dos actividades no pueden tener el mismo id.
- Dos learning paths no pueden tener el mismo nombre.
- Todas las calificaciones deben estar dentro de un rango definido (Por ejemplo: 0-5)
- La fecha límite de cada actividad no puede ser anterior a la fecha de creación del Learning Path o la actividad.
- Dentro de un mismo Learning Path, dos actividades no pueden tener el mismo título.
- Las encuestas deben tener al menos una pregunta antes de poder ser asignadas a un estudiante
- Una actividad debe estar marcada como "en curso" para que los estudiantes puedan acceder a ella
- Los estudiantes solo pueden dejar un comentario y un rating en una actividad si la han completado.
- Lo estudiantes no pueden ver el mismo Learning path 2 veces (Solo se puede repetir en caso de que se haya reprobado).

- Los estudiantes no pueden ver más de un learning path al mismo tiempo.
- Los estudiantes pueden cancelar un learning path.
- Todos los estudiantes y profesores deben tener un login único.

Programas de prueba:

1. La primera prueba tiene que verificar que un mismo usuario no puede iniciar dos actividades al mismo tiempo.

2. La segunda prueba tiene que verificar que no se pueden crear learning paths vacíos.

3. La tercera prueba va a verificar de una lista de usuarios creados que sean Estudiante o Profesor, pero no algo diferente.

4. La cuarta prueba, un usuario Profesor va a crear un quiz, y va a intentar crear una de las preguntas con tres opciones, no debería dejarlo seguir.

5. En la quinta prueba se hará algo parecido a la cuarta prueba, se va a crear una pregunta con dos opciones correcta, debe notificarse el error pues solo una opción correcta se permite.

6. En la sexta prueba nos compete probar que cuando hay dos opciones repetidas en una pregunta de opción múltiple el programa falla y notifica el error.

7. En la séptima prueba se debe evaluar que sucede si se adjunta una opción sin justificación a una pregunta de opción múltiple, la aplicación debería notificar que hay algo anormal en la creación de la opción que presenta carencias.

8. Esta es más bien un conjunto de pruebas en que un Usuario de tipo Profesor va a editar una cantidad X de actividades y de learning paths, algunos creados por él otros no. Las pruebas deberían recibir una alerta en los casos en que lo que se está editando no es creación de quien edita.

9. En la novena prueba hay que evaluar que un Usuario de tipo Estudiante no pueda crear LearningPaths ni actividades, es decir que al efectuar las pruebas se debería recibir alertas del sistema notificando aquella imposibilidad.

10. En la décima prueba hay que descartar la posibilidad de que haya id's repetidos para diferentes actividades, entonces hay que probar que a medida que se van creando actividades por un usuario Profesor, estas tienen id's diferentes a las anteriores. Una secuencia de pruebas siguiendo esta linea, en que dos profesores creen una cantidad de actividades, los id's de *todas* las actividades deberían ser diferentes todos.

11. La onceava prueba debe acertar sobre que no puede haber dos learning paths con un mismo nombre pues en el sistema se guardan en un mapa de hash, por lo que dos nombres iguales implicarían sobrescribir la información de una instancia anterior de un learning path. Esta prueba va a consistir en la creación de una serie de learning paths, algunos van a ser creados dos veces, la segunda vez debería ser rechazada.

12. En la décimo segunda prueba se debe intentar crear una actividad con una fecha límite anterior a la fecha de creación y verificar que el sistema rechace la creación con un mensaje de error indicando inconsistencia de fechas.

13. En la décimo tercera prueba se debe intentar calificar una actividad con una nota fuera del rango preestablecido, para que le sistema indique un error.

14. En la décimo cuarta prueba debemos intentar crear 2 actividades en un mismo learning path con el mismo nombre. Este intento debe marcar y enviar un error.

15. En la décimo quinta prueba se intentará realizar 2 Learning Paths al mismo tiempo.