**Ejercicios Análisis Requerimientos Funcionales**

**Ejercicio 1**

-**Objetivo:** Gestión académica en línea que permitirá a estudiantes y profesores gestionar cursos, tareas y exámenes de forma eficiente.

**Requerimientos Funcionales:**

**Inscripción en cursos:**

Los estudiantes se deben poder inscribir en cursos a través del sistema.

Acceder a materiales de estudio: Los estudiantes deben poder acceder a los materiales de estudio.

**Gestión de los materiales de estudio:**

 Los profesores deben poder gestionar los materiales de estudio.

**Envío y gestión de tareas:**

 Los estudiantes deben poder enviar y recibir tareas y poner comentarios en las que envían. Los profesores deben poder asignar y calificar las tareas.

**Publicación de calificaciones:**

 Los profesores publican calificaciones y los estudiantes reciben calificaciones. Incluir apartado de comentarios a modo de feedback

.

**Publicación de resultados finales:**

 Los profesores deben poder publicar los resultados finales y los estudiantes deben poder recibir dichos resultados.

**Foro de discusión:**

 Los profesores y estudiantes deben poder interactuar con el foro. La interacción debe incluir: Publicar, Eliminar, Modificar publicaciones y Responder.

**Integración con sistema externos:**

 El sistema debe ser capaz de integrarse con plataformas externas y bases de datos.

**Creación de una cuenta:**

El sistema debe permitir crear una cuenta en el propio sistema.

**Cursos:**

 Los cursos deben poder contener materiales, calificaciones, tareas, exámenes. Y dado el caso, permitir clases en streaming (confirmar).

**Requerimientos no funcionales:**

**Usabilidad:**

Interfaz intuitiva, fácil de usar y un diseño responsivo adaptado a móviles, tablets y computadores. Y accesible desde distintos navegadores.

**Seguridad:**

Autenticación de dos factores para todos los usuarios y encriptación de datos sensibles.

**Rendimiento:**

 Tiempos de respuesta del sistema iguales o inferiores a 3 segundos.

**Disponibilidad:**

 Accesibilidad continua con un tiempo de actividad 99.9%

**Escalabilidad:**

Capacidad de manejar un aumento en el número de usuarios y cursos eficientemente.

**Ambigüedades/Recomendaciones:**

* En el foro de discusión se puede filtrar por tipos de temas, carreras, tópicos, etc.
* El foro debe tener una moderación para palabras malsonantes o temas inapropiados?
* En el foro los profesores serán admins?
* ¿Existen otros roles a parte de profesores y estudiantes? Y un estudiante también puede ser profesor a la vez? Debe existir algún administrador del sistema o por áreas?
* Cómo se gestiona la integración con plataformas y bases de datos externas.
* ¿Qué plataforma de e-learning y bases de datos se van a integrar? (Puede que no se sepan todas desde un inicio, pero las que sepan que nos las digan).
* Se debe incluir un buscador por áreas, libros, cursos, etc?
* Se quiere incluir un chat para facilitar la comunicación estudiante-profesor?
* Se quiere incluir un chat entre estudiantes?
* Qué información se requiere de cada usuario.
* Cómo se gestionan las bajas, excedencias, etc.
* ¿Cómo se gestionan las sustituciones? Una asignatura puede tener más de un profesor?
* ¿Quieres que el usuario cree una cuenta? O que una vez inscrito en la universidad se cree automáticamente?
* Formatos de Materiales y Exámenes?
* ¿Cómo se realizan las entregas? Con links, con updates, archivos, audio o video?
* ¿La plataforma debe permitir clases en streaming?
* Recomendación no funcional: Contratación de servidores externos.
* La asistencia a clase es obligatoria? ¿Se debe controlar el acceso a clase?

**Clases / Entidades:**

**Clase Usuario:**

Atributos: +nombre: String, +apellido: String, +correo: String, +telefono: Int, -contraseña: String, +rol: String, +autenticacionDosFactores: Boolean

Métodos: -iniciarSesion(String correo, String contraseña): Boolean, +crearCuenta(): Boolean, +get y set para todos los atributos.

**Subclase Estudiante hereda de Usuario:**

Atributos: +cursosInscritos: List<Curso>, +tareasEnviadas: List<Tarea>, +calificacionesRecibidas: List<Calificacion>, +foroPublicaciones: List<PublicacionForo>

Métodos: +enviarTarea(Tarea tarea): Boolean, +recibirCalificacion(Calificacion calificacion): Boolean, +publicarEnForo(PublicacionForo publicacion): Boolean

**Subclase Profesor hereda de Usuario:**

Atributos: +cursosImpartidos: List<Curso>, +tareasAsignadas: List<Tarea>, +calificacionesPublicadas: List<Calificacion>, +foroPublicaciones: List<PublicacionForo>

Métodos: +asignarTarea(Tarea tarea, List<Estudiante> estudiantes): Boolean, +publicarCalificacion(Calificacion calificacion, List<Estudiante> estudiantes): Boolean, +publicarEnForo(PublicacionForo publicacion): Boolean

**Clase Curso:**

Atributos: +nombre: String, +codigo: String, +materialesEstudio: List<MaterialEstudio>, +tareas: List<Tarea>, +examenes: List<Examen>, +clasesStreaming: List<ClaseStreaming>

Métodos: +añadirMaterialEstudio(MaterialEstudio material): Boolean, +añadirTarea(Tarea tarea): Boolean, +añadirExamen(Examen examen): Boolean, +añadirClaseStreaming(ClaseStreaming clase): Boolean

**Clase Material Estudio:**

Atributos: +nombre: String, +tipo: String, +url: String

Métodos: +get y set para todos los atributos.

**Clase Foro:**

Atributos: +autor: Usuario, +contenido: String, +fechaPublicacion: Date, +comentarios: List<Comentario>

Metodos: +añadirComentario(Comentario comentario): Boolean

**Relación:**

Un estudiante se inscribe en cero o más cursos. Cardinalidad: 0..\*

Un profesor imparte uno o más cursos. Cardinalidad: 1..\*

Un profesor puede gestionar múltiples materiales de estudio. Cardinalidad: 1..\*

Un curso puede tener un foro. Cardinalidad: 1

Un foro puede tener de uno a más comentarios. Cardinalidad: 1..\*

Un curso puede tener de uno a más materiales de estudio. Cardinalidad: 1..\*

**Interfaz de usuario gráfica, Aplicación, Teams

Descripción generada automáticamente**