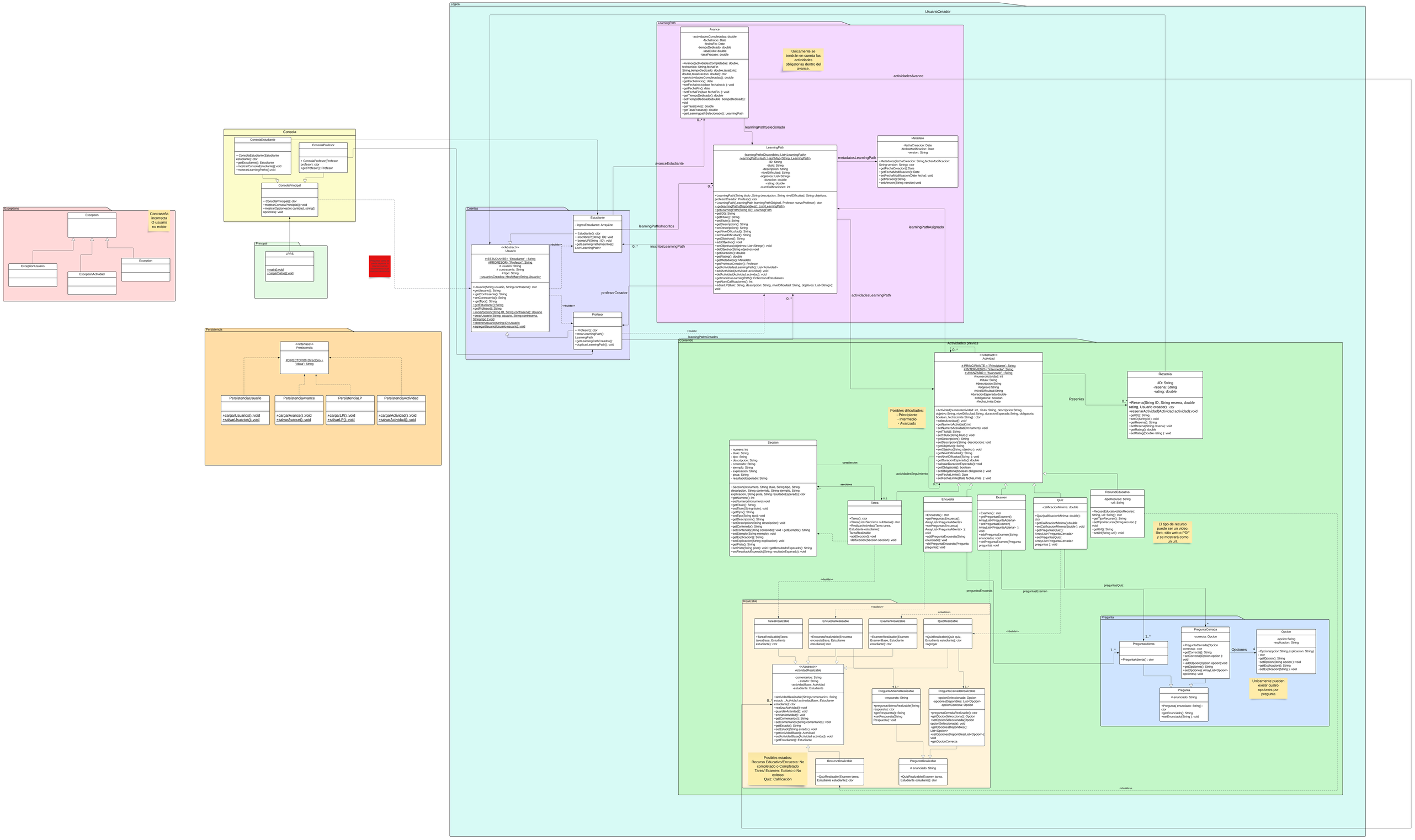
Logica
UsuarioCreador
LearningPath
Avance
-actividadesCompletadas: double
-fechainicio: Date
-fechaFin: Date
-tiempoDedicado: double
-tasaExito: double
-tasaFracaso: double
+Avance(actividadesCompletadas: double, fechainicio: String,fechaFin String,tiempoDedicado: double,tasaExito: double,tasaFracaso: double): ctor
+getActividadesCompletadas(): double
+getFechainicio(): date
+setFechainicio(date fechainicio): void
+getFechaFin(): date
+setFechaFin(date fechaFin): void
+getTiempoDedicado(): double
+setTiempoDedicado(double tiempoDedicado): void
+getTasaExito(): double
+getTasaFracaso(): double
+getLearningpathSeleccionado(): LearningPath
Unicamente se tendrán en cuenta las actividades obligatorias dentro del avance.
actividadesAvance
learningPathSeleccionado
avanceEstudiante
0..*
LearningPath
-learningPathsDisponibles: List<LearningPath>
-learningPathsHash: HashMap<String, LearningPath>
-ID: String
-titulo: String
-descripcion: String
-nivelDificultad: String
-objetivos: List<String>
-duracion: double
-rating: double
-numCalificaciones: int
+LearningPath(String titulo, String descripcion, String nivelDificultad, String objetivos, profesorCreador: Profesor): ctor
+LearningPath(LearningPath learningPathOriginal, Profesor nuevoProfesor): ctor
+ getlearningPathsDisponibles(): List<LearningPath>
+getLearningPath(String ID): LearningPath
+getID(): String
+getTitulo(): String
+setTitulo(): String
+getDescripcion(): String
+setDescripcion(): String
+getNivelDificultad(): String
+setNivelDificultad(): String
+getObjetivos(): String
+addObjetivo(): void
+setObjetivos(objetivos: List<String>): void
+delObjetivo(String objetivo):void
+getDuracion(): double
+getRating(): double
+getMetadatos(): Metadato
+getProfesorCreador(): Profesor
+getActividadesLearningPath(): List<Actividad>
+addActividad(Actividad actividad): void
+delActividad(Actividad actividad): void
+getinscritosLearningPath(): Collection<Estudiante>
+getNumCalificaciones(): Int
+editarLP(titulo: String, descripcion: String, nivelDificultad: String, objetivos: List<String>): void
metadatosLearningPath
Metadato
-fechaCreacion: Date
-fechaModificacion: Date
-version: String
+Metadatos(fechaCreacion: String,fechaModificacion: String,version: String): ctor
+getFechaCreacion(): Date
+getFechaModificacion(): Date
+setFechaModificacion(Date fecha): void
+getVersion(): String
+setVersion(String version):void
learningPathAsignado
actividadesLearningPath
Consola
ConsolaEstudiante
+ ConsolaEstudiante(Estudiante estudiante): ctor
+getEstudiante(): Estudiante
+mostrarConsolaEstudiante():void
+mostrarLearningPath():void
ConsolaProfesor
+ ConsolaProfesor(Profesor profesor): ctor
+getProfesor(): Profesor
ConsolaPrincipal
+ ConsolaPrincipal(): ctor
+mostrarConsolaPrincipal(): void
+mostrarOpciones(int cantidad, string[] opciones): void
Exceptions
Exception
Contraseña incorrecta
O usuario no existe
ExceptionUsuario
ExceptionActividad
Exception
Principal
LRIS
+main():void
+cargarDatos():void
Cuentas
Estudiante
-logrosEstudiante: ArrayList
+ Estudiante(): ctor
+ inscribirLP(String: ID): void
+ borrarLP(String: ID): void
+getLearningPathsInscritos(): List<LearningPath>
learningPathsInscritos
0..*
inscritosLearningPath
<<Abstract>>
Usuario
ESTUDIANTE= "Estudiante" : String
#PROFESOR= "Profesor" :String
usuario: String
contraseña: String
tipo: String
- usuariosCreados: HashMap<String,Usuario>
+Usuario(String usuario, String contraseña): ctor
+getUsuario(): String
+ getContraseña(): String
+setContraseña(): String
+ getTipo(): String
+getEstudiante():String
+getProfesor(): String
+iniciarSesion(String ID, String contraseña): Usuario
+crearUsuario(String usuario, String contraseña, String tipo):void
+obtenerUsuario(String ID):Usuario
+agregarUsuario(Usuario usuario): void
<<build>>
profesorCreador
Profesor
+ Profesor(): ctor
+crearLearningPath():LearningPath
+getLearningPathCreados():
+duplicarLearningPath(): void
<<build>>
learningPathsCreados
Persistencia
<<Interface>>
Persistencia
#DIRECTORIO=Directorio + "data/" :String
PersistenciaUsuario
+cargarUsuarios() :void
+salvarUsuarios() :void
PersistenciaAvance
+cargarAvance() :void
+salvarAvance() :void
PersistenciaLP
+cargarLP() :void
+salvarLP() :void
PersistenciaActividad
+cargarActividad() :void
+salvarActividad() :void
Contenido
Actividades previas
0..*
<<Abstract>>
Actividad
PRINCIPIANTE = "Principiante" :String
INTERMEDIO= "Intermedio" :String
AVANZADO = "Avanzado" :String
#numeroActividad: int
#titulo: String
#descripcion:String
#objetivo:String
#nivelDificultad:String
#duracionEsperada:double
#obligatoria: boolean
#fechaLimite:Date
+Actividad(numeroActividad: int, titulo: String, descripcion:String, objetivo:String, nivelDificultad:String, duracionEsperada:String, obligatoria: Boolean, fechaLimite:String) : ctor
+editarActividad(): void
+getNumeroActividad():int
+setNumeroActividad(int numero): void
+getTitulo(): String
+setTitulo(String titulo): void
+getDescripcion(): String
+setDescripcion(String descripcion): void
+getObjetivo(): String
+setObjetivo(String objetivo): void
+getNivelDificultad(): String
+setNivelDificultad(String): void
+getDuracionEsperada(): double
+calcularDuracionEsperada(): void
+getObligatoria(): boolean
+setObligatoria(boolean obligatoria): void
+getFechaLimite(): Date
+setFechaLimite(Date fechaLimite): void
Posibles dificultades:
- Principiante
- Intermedio
- Avanzado
Resenias
0..*
Resenia
-ID: String
-resena: String
-rating: double
+Resena(String ID, String resena, double rating, Usuario creador) : ctor
+resenarActividad(Actividad:actividad):void
+getID(): String
+setID(String id): void
+getResena(): String
+setResena(String resena): void
+getRating(): double
+setRating(Double rating): void
actividadesSeguimiento
Seccion
- numero: int
- titulo: String
- tipo: String
- descripcion: String
- contenido: String
- ejemplo: String
- explicacion: String
- pista: String
- resultadoEsperado: String
+Seccion(int numero, String titulo, String tipo, String descripcion, String contenido, String ejemplo, String explicacion, String pista, String resultadoEsperado): ctor
+getNumero(): int
+setNumero(int numero):void
+getTitulo(): String
+setTitulo(String titulo): void
+getTipo(): String
+setTipo(String tipo): void
+getDescripcion(): String
+setDescripcion(String descripcion): void
+getContenido(): String
+setContenido(String contenido): void +getEjemplo(): String
+setEjemplo(String ejemplo): void
+getExplicacion(): String
+setExplicacion(String explicacion): void
+getPista(): String
+setPista(String pista): void +getResultadoEsperado(): String
+setResultadoEsperado(String resultadoEsperado): void
tareaSeccion
secciones
1..*
1..1
Tarea
+Tarea(): ctor
+Tarea(List<Seccion> subtareas): ctor
+RealizarActividad(Tarea tarea, Estudiante estudiante): TareaRealizable
+addSeccion(): void
+delSeccion(Seccion seccion): void
Encuesta
+Encuesta() : ctor
+getPreguntasEncuesta():ArrayList<PreguntaAbierta>
+setPreguntasEncuesta(ArrayList<PreguntaAbierta>): void
+addPreguntaEncuesta(String enunciado): void
+delPreguntaEncuesta(Pregunta pregunta): void
Examen
+Examen() : ctor
+getPreguntasExamen():ArrayList<PreguntaAbierta>
+setPreguntasExamen(ArrayList<PreguntaAbierta>): void
+addPreguntaExamen(String enunciado): void
+delPreguntaExamen(Pregunta pregunta): void
Quiz
-calificacionMinima: double
+Quiz(calificacionMinima: double): ctor
+getCalificacionMinima():double
+setCalificacionMinima(double): void
+getPreguntasQuiz():ArrayList<PreguntaCerrada>
+setPreguntasQuiz(ArrayList<PreguntaCerrada> preguntas): void
RecursoEducativo
-tipoRecurso: String
-url: String
+RecursoEducativo(tipoRecurso: String, url: String): ctor
+getTipoRecurso(): String
+setTipoRecurso(String recurso): void
+getUrl(): String
+setUrl(String url): void
El tipo de recurso puede ser un video, libro, sitio web o PDF y se mostrará como un url.
preguntasEncuesta
preguntasExamen
preguntasQuiz
Realizable
<<build>>
TareaRealizable
+TareaRealizable(Tarea tareaBase, Estudiante estudiante): ctor
EncuestaRealizable
+EncuestaRealizable(Encuesta encuestaBase, Estudiante estudiante):ctor
ExamenRealizable
+ExamenRealizable(Examen ExamenBase, Estudiante estudiante): ctor
QuizRealizable
+QuizRealizable(Quiz quiz, Estudiante estudiante): ctor
+agregar
<<Abstract>>
ActividadRealizable
-comentarios: String
- estado: String
-actividadBase: Actividad
-estudiante: Estudiante
+ActividadRealizable(String comentarios, String estado , Actividad actividadBase, Estudiante estudiante) :ctor
+realizarActividad(): void
+guardarActividad(): void
+enviarActividad(): void
+getComentarios(): String
+setComentarios(String comentarios): void
+getEstado(): String
+setEstado(String estado): void
+getActividadBase(): Actividad
+setActividadBase(Actividad actividad): void
+getEstudiante(): Estudiante
0..*
PreguntaAbiertaRealizable
-respuesta: String
+preguntaAbiertaRealizable(String respuesta): ctor
+getRespuesta(): String
+setRespuesta(String Respuesta): void
PreguntaCerradaRealizable
-opcionSeleccionada: Opcion
-opcionesDisponibles: List<Opcion>
-opcionCorrecta: Opcion
+preguntaCerradaRealizable(): ctor
+getOpcionSeleccionada(): Opcion
+setOpcionSeleccionada(Opcion opcionSeleccionada): void
+getOpcionesDisponibles(): List<Opcion>
+setOpcionesDisponibles(List<Opcion>): void
+getOpcionCorrecta
RecursoRealizable
+QuizRealizable(Examen tarea, Estudiante estudiante): ctor
PreguntaRealizable
enunciado: String
+QuizRealizable(Examen tarea, Estudiante estudiante): ctor
Posibles estados:
Recurso Educativo/Encuesta: No completado o Completado
Tarea/ Examen: Exitoso o No exitoso
Quiz: Calificación
Pregunta
PreguntaAbierta
+PreguntaAbierta() : ctor
PreguntaCerrada
-correcta: Opcion
+PreguntaCerrada(Opcion correcta) : ctor
+getCorrecta(): String
+setCorrecta(Opcion opcion): void
+ addOpcion(Opcion opcion):void
+getOpciones(): String
+setOpciones(ArrayList<Opcion> opciones): void
Opciones
Opcion
-opcion:String
-explicacion: String
+Opcion(opcion:String,explicacion: String) : ctor
+getOpcion(): String
+setOpcion(String opcion): void
+getExplicacion(): String
+setExplicacion(String): void
Unicamente pueden existir cuatro opciones por pregunta
Pregunta
enunciado: String
+Pregunta(enunciado: String) : ctor
+getEnunciado(): String
+setEnunciado(String): void