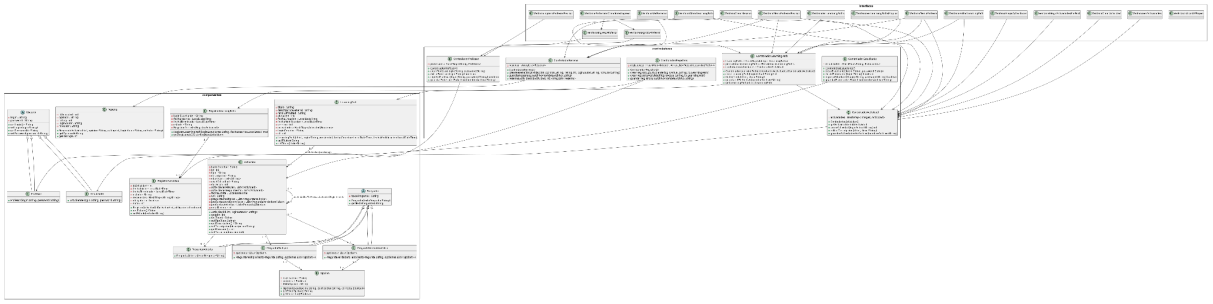
Documento de análisis proyecto 3:

Angela Bárcenas
Tomas Hernandez
Juan David Uribe

A Continuación presentamos los png de los UML y el diagrama:

Entendemos que se ven pequeños y es difícil de leerlos, por eso adjunto en el repositorio, adjuntamos los links que les permiten ver los UML de forma más grande y podes apreciar las relaciones mejor.
Sin embargo queremos aclarar que debido a la cantidad de componentes, clases, objetos y tipos de relación que utilizamos puede llegar a ser bastante confuso el diagrama completo, por eso realizamos algunos otros más específicos que permiten entender la totalidad del programa de una manera mucho más fácil y efectiva.

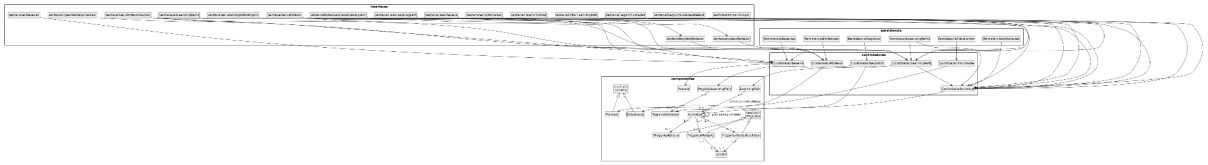
Diagrama completo:



Descripción:
Este diagrama muestra las clases, relaciones, atributos y métodos (en mayor detalle) del sistema. Incluye controladores, clases del dominio y clases de persistencia, reflejando la complejidad interna y las interdependencias lógicas.

Justificación del diagrama:
Útil para los desarrolladores que necesitan un entendimiento profundo del código, facilita la localización de responsabilidades, métodos y atributos, así como la detección de posibles puntos débiles o duplicaciones de lógica.

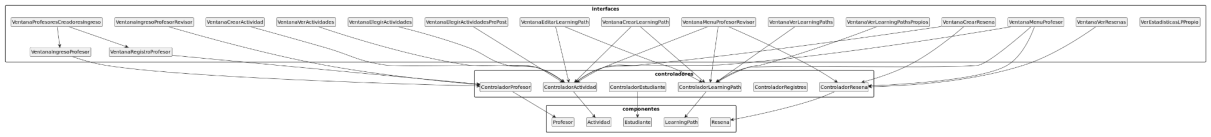
UML alto nivel:



Descripción:
El diagrama de alto nivel presenta todas las clases del sistema (interfaz, controladores, dominio, persistencia) y las relaciones más importantes entre ellas, pero sin detallar atributos o métodos.

Justificación del diagrama:
Provee una visión global, permitiendo entender la arquitectura del sistema de un solo vistazo. Es útil para la orientación de nuevos integrantes del equipo y para comunicar la arquitectura a partes interesadas que no requieren un nivel de detalle técnico exhaustivo.

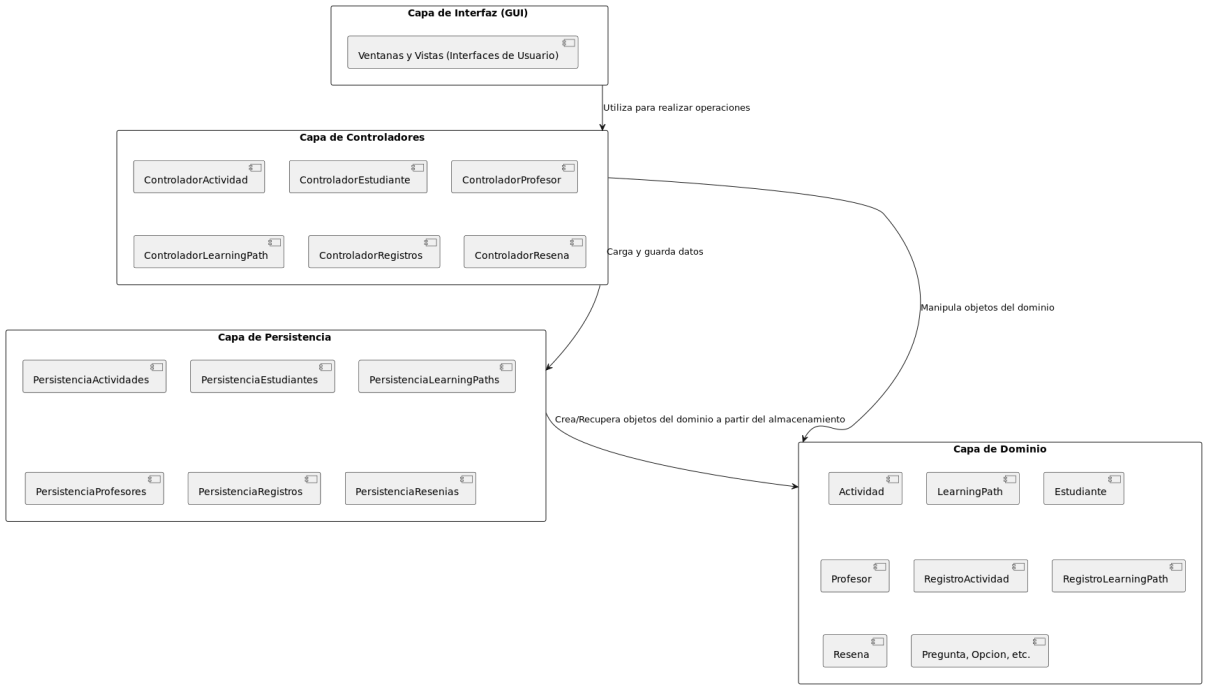
UML Interfaces:



Descripción:
Este diagrama muestra las clases de la capa de interfaz (GUI) y su interacción con los controladores. Aunque no se muestran todos los métodos, se destaca cómo las ventanas (VentanaCrearActividad, VentanaVerActividades, etc.) dependen de los controladores para realizar las operaciones solicitadas por el usuario.

Justificación del diagrama:
Este diagrama ayuda a visualizar qué componentes de la interfaz están presentes y cómo se relacionan con la capa de negocio. Esto facilita la identificación de dependencias y la planificación de pruebas de interfaz.

Diagrama de componentes:

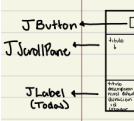


Descripción:
El diagrama de componentes muestra cómo se agrupan las clases en componentes o módulos lógicos (GUI, Controladores, Dominio, Persistencia) y cómo se comunican entre sí. Representa las dependencias entre las diferentes partes del sistema a un nivel de mayor abstracción.

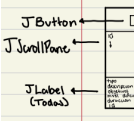
Justificación del diagrama:
Este diagrama facilita la identificación de las fronteras entre capas y la visión de cómo se ensamblan los diferentes módulos. Resulta esencial para planificar la implementación por equipos separados, el despliegue o futuras integraciones.

A Continuación se exponen los bocetos de diseño que nos permitieron tener una idea concreta acerca de que queríamos implementar y como lo queremos implementar, es decir teníamos clara una idea en el diseño de interfaces y queríamos que fuera clara y ordenada para todos, para que de esa forma nuestra forma de implementación fuera coherente en todos los aspectos posibles. Estos bocetos nos hicieron más fácil la implementación de las interfaces, al darnos una guía de lo que queríamos que tuviera cada una de las pantallas.

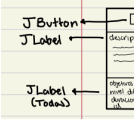
· Desarrollar actividad



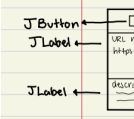
BorderLayout
- si o si se selecciona algo en lista
- si no, no permitir elegir
- no se puede elegir algo no inscrito
  ↳ si se intenta, generar popup o advertencia



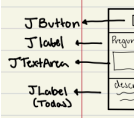
BorderLayout
- solo mostrar actividades en LP elegido
- si o si se selecciona algo en lista
- si no, no permitir elegir
- no permitir realizar actividades hechas



BorderLayout
- cerrar no envía tarea
- enviar notifica envío, no completa actividad
- Actividad Tarea

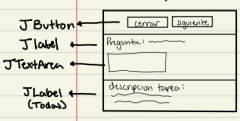


BorderLayout
- cerrar no envía tarea
- enviar completa actividad
- Actividad Recurso educativo



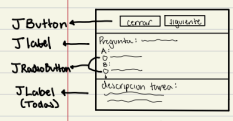
BorderLayout
- cerrar no envía tarea
- enviar completa actividad
- siguiente pasa a siguiente pregunta, la ultima tendrá boton que diga envío

- Actividad encuesta



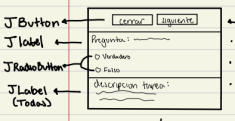
BorderLayout
- cerrar no envía tarea
- enviar notifica envío, no completa actividad
- siguiente pasa a siguiente pregunta, la ultima tendrá boton que diga envío

- Actividad examen



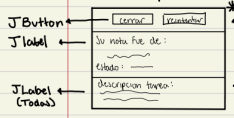
BorderLayout
- cerrar no envía tarea
- enviar completa actividad
- siguiente pasa a siguiente pregunta, la ultima tendrá boton que diga envío
- solo se puede elegir un ◉
- Actividad Quiz multiple

*



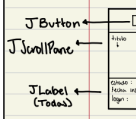
BorderLayout
- cerrar no envía tarea
- enviar completa actividad
- siguiente pasa a siguiente pregunta, la ultima tendrá boton que diga envío
- solo se puede elegir un ◉
- Actividad Quiz multiple True - False

*

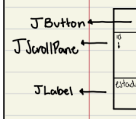


BorderLayout
- solo se llega aquí al enviar Quiz multiple
- reintentar reinicia el Quiz
- solo dejar reintentar si se desaprueba

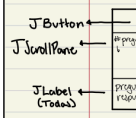
· Revisar progreso



BorderLayout
- si o si se selecciona algo en lista
- si no, no permitir elegir
- no se puede elegir algo no inscrito
  ↳ si se intenta, generar popup o advertencia



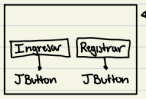
BorderLayout
- solo mostrar actividades en LP elegido
- si o si se selecciona algo en lista
- si no, no permitir elegir



BorderLayout

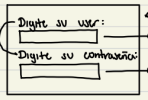
Consola profesor creador

· Pantalla inicial



FlowLayout

· Ingreso - Registro



GridLayout (2×1)

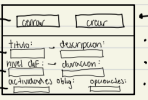
- usr inexistente o contraseña incorrecta (ingreso)
  ↳ generan popup o JLabel informando error
- usr existente (registro)

· Menu de opciones

• → ya cubiertos en consola estudiante

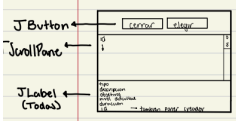
GridLayout (3×3)
- todos son JButton
- salir guardara todo, cerrar pestaña tambien
- si se quiere mapa, reorganizar botones

· Crear learning path



BorderLayout
- duracion si o si en minutos
- cerrar no crea nada
- no dejar enviar sin todo lleno

- elegir actividades



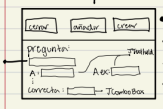
- BorderLayout
  - se permite elegir nada
  - se permite elegir mas de 1 actividad
  - sur se define por ultima actividad elegida

- Crear actividad

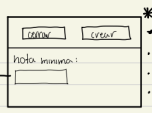
- BorderLayout
  - no dejar crear sin todo lleno (menos actividades)
  - depende de tipo se enviara a diferentes ventanas
  - tarea termina, el resto no

- BorderLayout
  - no crear sin todo lleno
  - Cerrar no crea nada
  - Recurso educativo

- BorderLayout
  - no crear o añadir sin todo lleno
  - Cerrar no crea nada
  - Encuesta y examen*



- BorderLayout
  - no crear o añadir sin todo lleno
  - cerrar no crea nada
  - Quiz multiple* y True - False*

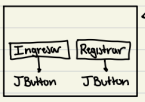


- BorderLayout
  - no crear sin todo lleno
  - cerrar no crea nada
  - para todos los *

- editar actividad y editar learning path sirven con la misma interfaz de creacion, pero, en los componentes estara por defecto lo original (no se puede editar tipo)

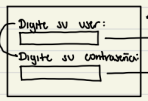
Consola profesores revisores

- Pantalla inicial



- FlowLayout

- Ingreso - Registro



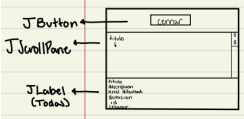
- GridLayout (2×1)
  - JTextfield
  - JPasswordfield

- user inexistente o contraseña incorrecta (ingreso)
  - generan popup o JLabel informando error
- user existente (registro)

- Menu de opciones

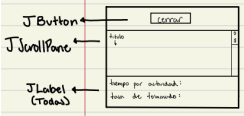
- → ya cubiertas en consola estudiante
- GridLayout (3×3)
  - todos son JButton
  - Salir guarda todo, cerrar pestaña tambien
  - Si se quiere mapa, reorganizar botones

- Ver learning paths propios



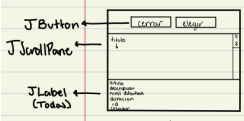
- BorderLayout
  - solo mostrar lps de login correspondiente

- Ver estadisticas learning path propio

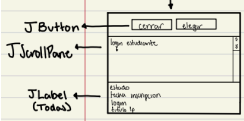


- BorderLayout
  - solo mostrar lps de login correspondiente

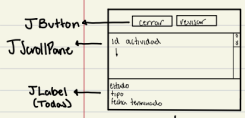
- Revisar actividad estudiante



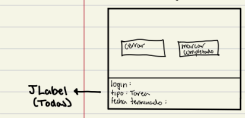
- BorderLayout
  - solo mostrar lps de login correspondiente
  - solo se permite elegir si se tiene seleccion



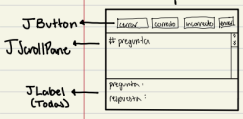
- BorderLayout
  - solo mostrar logns de lp seleccionado
  - solo se permite elegir si se tiene seleccion



- BorderLayout
  - solo se puede revisar estados pendientes
  - solo mostrar fecha terminado en completados
  - solo se permite revisar si se tiene seleccion
  - solo se permite revisar examen y tarea

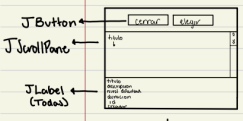


- BorderLayout
  - cerrar no marca nada
  - Actividad Tarea

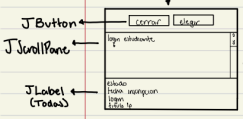


- BorderLayout
  - solo se puede enviar calificando todas las ?
  - cerrar no hace ninguna revision
  - enviar cambia estado

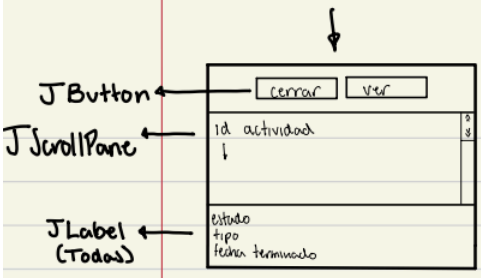
- Revisar progreso estudiante



- BorderLayout
  - solo mostrar lps de login correspondiente
  - solo se permite elegir si se tiene seleccion

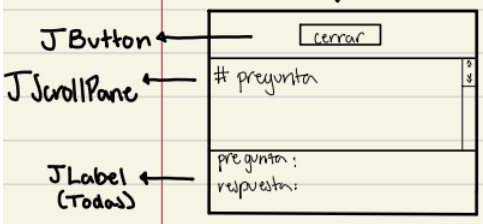


- BorderLayout
  - solo mostrar logns de lp seleccionado
  - solo se permite elegir si se tiene seleccion



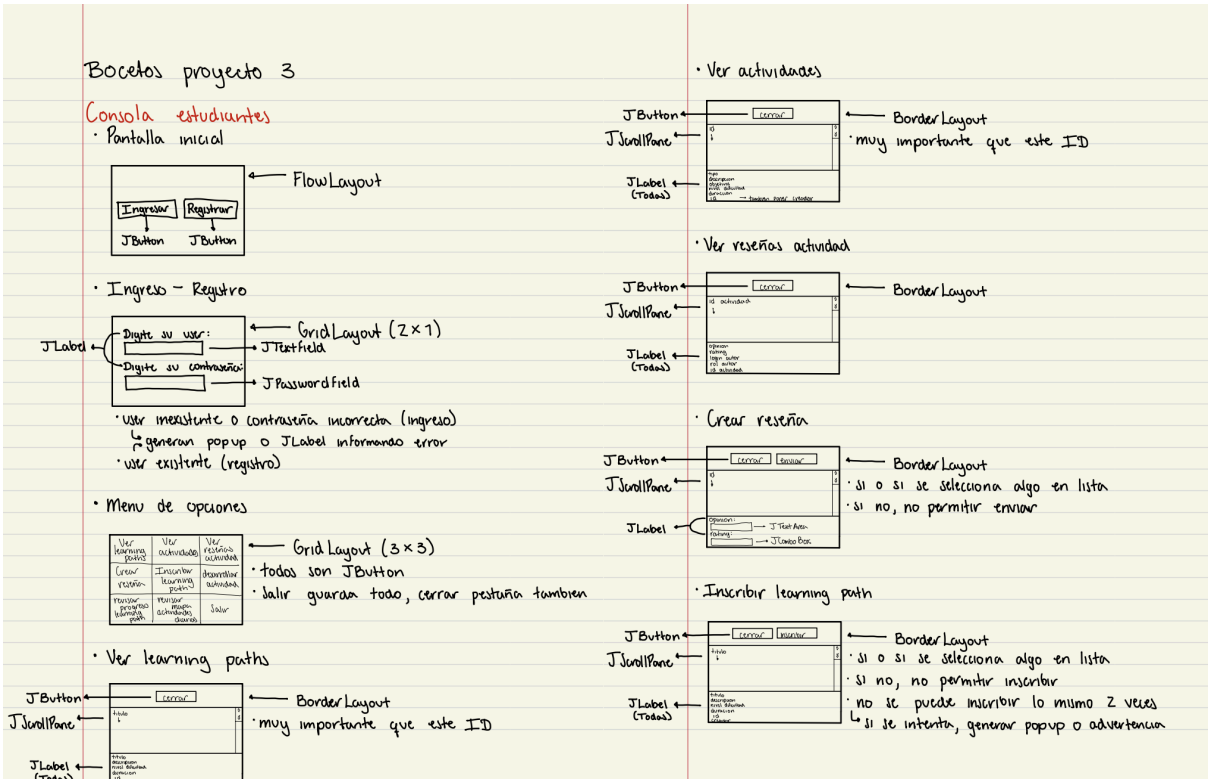
← BorderLayout

- solo permitir ver completadas
- solo permitir ver si se tiene seleccion
- se puede ver todo menos tarea y recurso



← BorderLayout

- Examen, quiz multiple, true - false, encuesta
- no se permite revision desde esta ventana



Por último las decisiones de diseño:

Estructura General del Sistema

El sistema se basa en un modelo que separa las capas de interfaz (GUI), lógica de aplicación (controladores) y lógica de dominio (modelos de negocio), con una capa de persistencia encargada del almacenamiento. Esta separación (inspirada en el patrón MVC) permite aislar responsabilidades, favorecer la reutilización del código y facilitar cambios futuros en la interfaz o en la forma en que se almacenan los datos.

Decisiones de Diseño Clave:

Separación por Capas (MVC/Arquitectura en Capas):

Justificación: Permite aislar la lógica de negocio de la lógica de presentación y la persistencia, facilitando el mantenimiento y el reemplazo de una capa sin afectar significativamente a las demás. Por ejemplo, cambiar la interfaz gráfica o la forma de persistir los datos no requiere alterar la lógica interna del sistema.

Uso de Controladores dedicados para cada Entidad/Funcionalidad:

Justificación: Cada controlador se encarga de administrar las acciones y operaciones sobre una parte específica del dominio (actividades, learning paths, estudiantes, etc.). Esto mantiene la lógica concentrada y reduce la complejidad, evitando controladores monolíticos.

Modelo de Dominio Enfocado en Entidades Clave (Actividad, LearningPath, Estudiante, Profesor, Resena):

Justificación: Las clases del dominio se diseñan para representar conceptualmente las entidades y sus relaciones. Esto facilita la trazabilidad entre los requisitos del negocio y las clases del sistema.

Persistencia Independiente a través de Clases de Persistencia:
Justificación: La persistencia se maneja mediante clases especializadas, permitiendo cambios en el método de almacenamiento (archivos JSON, bases de datos relacionales, NoSQL, etc.) sin alterar el núcleo del dominio. De esta forma, la adaptación a nuevos requisitos de almacenamiento es más sencilla.

Interfaces de Usuario Modulares:

Justificación: Las distintas ventanas y vistas se enfocan en acciones concretas (crear actividad, ver learning paths, editar un learning path, etc.). Esto facilita el mantenimiento y la evolución de la interfaz, así como la incorporación de nuevas funcionalidades sin recargar una sola vista.
Diagramas UML