Proyecto Relaciones/Componentes/ Interfaz Polimorfismo

PAREJAS Nombre Alumnos
Docente: Alejandro Rodas Vásquez
Universidad Tecnológica de Pereira

20 de mayo de 2025

Introducción

Una de las claves en este ejercicio es tener claridad los conceptos Arquitectura de Software, donde la aplicación del principio de *Modularidad* es fundamental.

1. Arquitectura de Software

Usted ha sido contratado para implementar un *Sistema de Información Hospitalaria* que se encuentra en su fase inicial. De modo que hasta el momento solo se utilizarán las entidades, *Hospital* y *Doctor*. El esquema de la base de datos es el siguiente:

Figura 1: Esquema Entidad/Relación Sistema de Información Hospitalaria.

También, desde una perspectiva de *Programación Orientada a Objetos* el esquema de la base de datos (Figura 1) se puede representar de la siguiente manera:

Figura 2: Modelo Sistema de Información Hospitalaria.

En la Figura 3 **se observar un ejemplo** de *Diagrama de Clases* con la *Interfaz Implementada del Crud*. Este es un **ejemplo hipotético** donde se utiliza el componente *Main*.

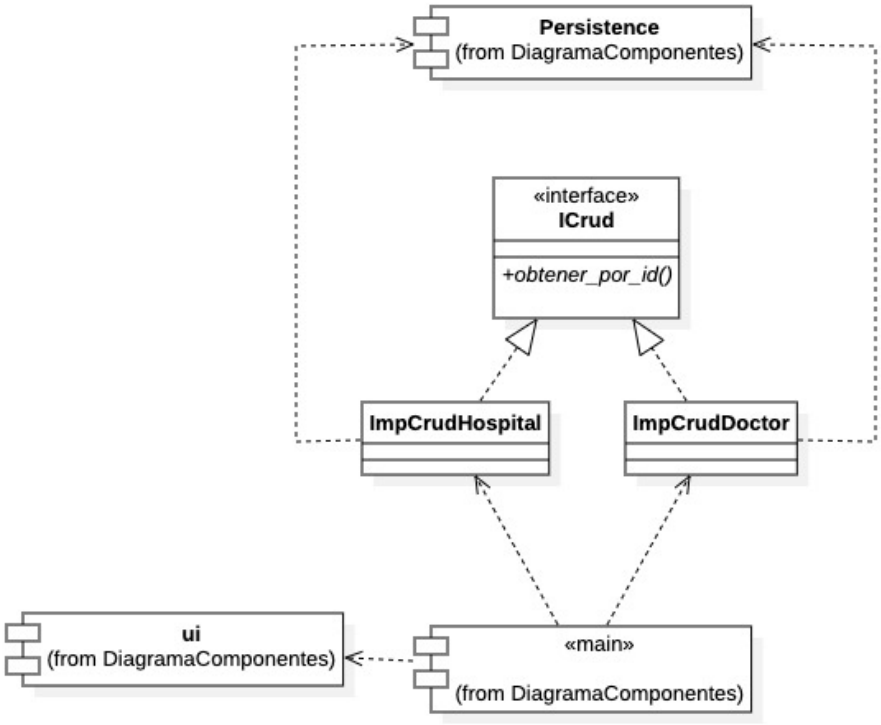

Figura 3: Diagrama de Clases - Interfaz Implementada.

2. Requerimientos Funcionales

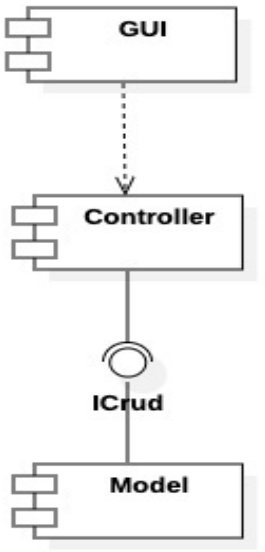

Figura 4: Arquitectura por Capas del Sistema

Esta aplicación debe de:

- Usted debe de implementar el *Sistema de Información Hospitalaria* siguiendo la Arquitectura previamente expuesta (Sección 1).
- Seguir el *Estilo Arquitectónico en Capas* presentado en la Figura 4.
- Ser desarrollada en Python (aplicación de Escritorio) utilizando los conceptos de *Módulos y NameSpace*.
- Implementar la función *obtener_por_id()* para obtener información del médico y del hospital utilizando el nombre del hospital y la identificación del médico.

2.1. Actividad de Investigación

Realizar los *Diagramas de Casos de Uso* del sistema de información.

3. ¿Cómo realizo la entrega?

Esta aplicación debe de ser construida bajo los siguientes parámetros arquitectónicos:

1. Los componentes para separar responsabilidades (Modelo, Controladores, Vista (UI) y Test).
2. Seguir el *Estilo Arquitectónico en Capas* presentado en la Figura 4.

3. Utilicen el concepto de *Separación de Preocupación, Delegación de Responsabilidades y Modularidad.*

4. **Es obligatorio implementar el concepto de *Interfaz.***

5. **Si se van a utilizar varias pantallas (es decir, GUI)** es necesario que cada uno esté un archivo independiente. Ej: $formulario\_productor.py$, $formulario\_producto\_fertilizante.py$

6. Cada *Clase* debe de estar en un archivo separado dentro del *Componente* de *Modelo.*

7. Pantallazos donde se corrobore el uso del **debug**. En estas imagenes de debe de observar la composición del objeto. Es decir, se envidencia que el objeto x tiene *asociado* n *insntacias* del objeto y. Lo mismo con la herencia.

8. Usted debe de entregar el código fuente en su repositorio de *github.*

9. Pantallazos donde se corrobore el funcionamiento del sofware (consultas realizadas y resultado obtenido) y se compruebe que los Requerimientos Funcionales (Sección 2) han sido cumplidos.

10. Realizar **Diagrama de Clases Estructurales**, implementación de las interfaces junto con sus dependencias a los controladores respectivos.

4. Evidencias

4.1. Diagramas de Casos de Uso

4.2. Diagramas de Clases Estructurales

4.3. Pantallazos

ACÁ VAN LOS PANTALLAZOS