PLAN DE PROYECTO: SISTEMA DE GESTIÓN DE INFORMACIÓN PARA PROYECTOS COMPLEMENTARIOS DE SALUD

Programado por Juliana Revelo M

1. Iniciación

El proyecto consiste en el desarrollo de un sistema para la gestión de la información de una base de datos sobre proyectos de salud pública. En este caso, se toma el caso de una Fundación que gestiona dos proyectos en varias instituciones y con diferentes participantes.

1.1. Nombre del proyecto: *Sistema de gestión: base de datos proyectos complementarios de salud*

1.2. Objetivo del proyecto (declaración smart):

Específico: Será un sistema de gestión de base de datos para dos proyectos terapéuticos (musicoterapia y arte terapia) en tres instituciones prestadoras de salud, que gestionará información de beneficiarios (género, edad, enfermedad/padecimiento, tratamiento y respuesta al tratamiento), clasificados en tres grupos (pacientes y sus cuidadores, trabajadores de salud y particulares.

Medible:

- 2 tipos de proyectos
- 3 instituciones por proyecto
- 3 categorías de beneficiarios
- 5 atributos por beneficiario

Alcanzable: Sistema básico de CRUD con clases aprendidas en el bootcamp de python, estructura de datos en memoria y funciones fáciles de usar y efectivas.

Relevante: Optimiza recursos de una organización con poco personal a la vez que facilita el seguimiento y análisis de efectividad de terapias alternativas en salud pública.

Tiempo: 3-5 días de desarrollo (análisis: 1 día, codificación: 2-3 días, pruebas: 1 día).

Plazo de entrega: 11 agosto de 2025

Justificación: Organizar información dispersa, optimizar los procesos del equipo de la Fundación de tal manera que puedan identificar las tendencias y los datos relevantes para sus reportes de efectividad.

1.3. Objetivos específicos:

- Aplicar los conceptos de clases y objetos para modelar entidades del mundo real.
- Implementar el principio de herencia para crear jerarquías de clases.
- Demostrar el uso de la encapsulación en la definición de atributos y métodos.
- Gestionar la interacción entre diferentes objetos dentro de una aplicación.

1.4. Stakeholders (interesados) principales: La población objetivo, así como sus principales inversores, son las instituciones prestadoras de salud y las organizaciones que proveen servicios complementarios de salud a dichas instituciones. Fuera del sector involucrado en salud pública, las

o los interesados en este proyecto son todos aquellos potenciales clientes que expresen interés comercial en el sistema propuesto. La involucrada en este proyecto es Juliana Revelo y el equipo de mentores del Bootcamp de programación con python.

2. Planificación

2.1. Alcance y requisitos principales:

Funcionalidades clave del sistema:

- Registro y gestión de proyectos, instituciones y beneficiarios
- Clasificación automática de beneficiarios por tipo
- Búsqueda y filtrado
- Generación de datos para reportes estadísticos
- Exportación de datos

2.2. Work breakdown structure (wbs) / desglose de tareas:

Elementos necesarios para el sistema:

- Herencias: Clase base Proyecto con subclases ProyectoMusicoterapia y ProyectoArteterapia
- Encapsulación: Atributos info privada con getters/setters
- Polimorfismo: Métodos sobrescritos para diferentes tipos de proyectos
- Composición: Proyectos contienen instituciones, instituciones contienen beneficiarios

Procesos necesarios para el sistema:

1. Análisis y Diseño de Clases

- Identificar entidades principales (Proyecto, Institución, Beneficiario
- Definir herencia con clase
- Establecer enums para tipos de datos categóricos
- Definir las clases principales que representarán los datos y procesos relevantes en el sistema de salud a desarrollar.

2. Definir atributos:
    - Para cada clase, identificar los atributos que almacenarán la información relevante i.e. rol, edad, género, enfermedad y/o sintomatología.

3. Herencias:
    - Subclases especializadas: ProyectoMusicoterapia y ProyectoArteterapia
    - Métodos

4. Definir objetos

- Proyectos contienen instituciones
- Instituciones contienen beneficiarios

5. 2da revision a métodos:
    - Para cada clase, definir los métodos que realizarán las acciones necesarias

i.e. agregar_beneficiario().

6. Establecer relaciones:
   - Cómo se relacionan las clases entre sí, qué elementos se repiten, cuales aplican a diferentes beneficiarios, por ej un tratamiento del que participa un beneficiario y una institución.

7. Implementación: Revisar código

- Atributos privados con __
- Properties
- Validación setters

2.3. Cronograma y estimación de tiempos:

Cronograma establecido aparte de las sesiones de preparación en clase

Sesión 1: Escribir la estructura base del proyecto. 8h

Sesión 2: Análisis: 1 día (dos jornadas de la tarde)
Revisión borrador 1 de la estructura: Análisis y diseño de clases

Sesión 3: Iteración código: 2-3 días.

Sesión 4: Pruebas código. 1 día

2.4. Identificación de riesgos iniciales:

- Problemas de configuración con VSCODE. Para mitigar ese riesgo descargué el programa en su última versión junto a Python como lenguaje de interpretación.
- Problemas con la pérdida del código, para mitigar ese riesgo subí el código a mi perfil de GITHUB así como a GOOGLE COLAB para tener un backup.

3. Ejecución

3.1. Bitácora de desarrollo: *Durante las sesiones de codificación, anota aquí las tareas principales que se van completando. Este es un registro del trabajo hecho.*

Fecha	***Tareas completadas***	***Responsable***
10 agosto	Prueba final código	Juliana Revelo
07, 08, 09 agosto	Trabajar y revisar ajustes a: Objetos. Relaciones. Atributos.	Juliana Revelo
08 08 25	Monitoria virtual con el profesor Iván: Revisión y corrección del código. Ajustes a los mensajes que lee el usuario final ya que	Juliana Revelo

	no aparecían durante mis pruebas.	
06 08 25	Análisis y diseño de clases (jornada 2)	Juliana Revelo
05 08 25	Análisis y diseño de clases (jornada 1)	Juliana Revelo
05 08 25	Ejercicio en clase de escritura de código de herencias.	Juliana Revelo
31 07 25	Estructura base del proyecto (4h)	Juliana Revelo
29 07 25	Estructura base del proyecto (4h)	Juliana Revelo

4. Monitoreo y control

4.1. Seguimiento del cronograma:

¿Las tareas planificadas se completaron a tiempo?

El único imprevisto fue la pérdida de internet el día domingo, tuve que desarrollar la prueba final en las horas de la noche.

4.2. Gestión de cambios (si aplica):

Al inicio tenía la idea de implementar la funcionalidad de remover programas de la cual desistí porque afectaba el registro estadístico que sí era una funcionalidad prioritaria y decidí que intentaría agregarla en una versión actualizada.

5. Cierre

5.1. Verificación del cumplimiento de requisitos: *Crea una lista de chequeo basada en los requisitos del punto 2.1 y marca si cada uno fue completado exitosamente.*

Requisito	***Completado (Si/No)***
Definición de objetivos	sí
Definicion de estructura	sí
Ejemplo/simulación de base de datos para prueba	sí
Prueba de código (monitorias para revisión)	sí

5.2. Lecciones aprendidas:

- *¿Qué funcionó bien durante el desarrollo del proyecto?*

Durante el bootcamp revisamos varios elementos del código que tuve que emplear en el proyecto. Para mí fue de gran ayuda reconocer cómo debería "comportarse" el código, incluso durante la etapa preliminar de preguntarme lo que quería medir y buscar. Me habría gustado tener más experiencias así durante las clases para aplicar el conocimiento en proyectos reales.

- *¿Qué harían diferente en un futuro proyecto?*

Incluiría una versión UX/UI del sistema. También me habría gustado tener más tiempo de ejercicios con proyectos personales como este, para probar con diferentes clases y variables Si bien en la prueba final del código pude estimar cómo podría interactuar la Fundación con el sistema, no me alcanzó el tiempo para pensar en el diseño de elementos gráficos que habría sido útil para usar e incluso mejorar el sistema.

- *¿Fueron precisas sus estimaciones de tiempo? ¿Por qué sí o por qué no?*

En términos generales sí, ya tenía bastante claridad sobre los elementos y la información que se manejaba en la Fundación, por lo cual la etapa de definir las clases, objetos y métodos fue más rápida. Creo que la familiarización con el problema a resolver y sobre todo el contexto en el que será usado el sistema facilita mucho el ejercicio porque ayuda a acotar las opciones.