**Sistema de gestión de socios y clases en un gimnasio**

La idea de este proyecto es gestionar los socios , clases y asistencias de las actividades ofrecidas en el gimnasio, con el objetivo de controlar eficientemente las inscripciones, control de asistencias y el seguimiento y formato de pago.

El sistema registra cada socio con un ID personal donde se guardan sus datos personales como, nombre, apellido, fecha de nacimiento, ID, método de pago y estado. Además de los socios, los instructores y clases también cuentan con un registro incluyendo su respectiva información.

Cada clase está vinculada al socio con un registro de asistencias, el cual permite generar estadísticas de las mismas.

Para poder ingresar al sistema este cuenta con un login. El programa también valida cada información que entra evitando que se ingresen caracteres no correspondientes, no juzga por mayúsculas y no se repiten Id 's, clases, horarios, entre otras.

El sistema se maneja con varios menús. Eligiendo por cada opción un número (1, 2, 3, 0, etc)

Menú principal: donde se pregunta a qué entidad se desea ver información

-Socios:

-Nuevo Socio

-Alta/Baja Socio (abre otro menú para alta o baja y luego pregunta el id)

-Modificar Socio (luego pregunta qué socio(id) y que desea modificar)

-Mostrar Socio

-Salir

-Instructores

-Nuevo instructor

-Alta/Baja instructor (abre otro menú para alta o baja)

-Modificar instructor (abre otro menú para alta o baja y luego pregunta el id)

-Mostrar instructor

-Salir

-Clases

-Nueva clase

-Alta/Baja clase (abre otro menú para alta o baja)

-Modificar clase (abre otro menú para alta o baja y luego pregunta el id)

-Mostrar clase

-Salir

-Estadísticas

-Cantidad de asistencias por clase

-Promedio de socios activos e inactivos

-Cantidad de clases por instructor

- Salir

Todos los “Salir” te envían al menú anterior.

El manejo de varios menús se debe a la idea de no volverlos menús extensos.

-Entidades involucradas:

socios: Nombre, apellido, fecha de nacimiento, ID, método de pago y estado.

Instructores: Nombre, apellido, fecha de nacimiento, ID y estado.

clases: IdClase, nombre, día, hora, IdInstructor y estado.

-Participantes:

Scrum master: Mateo Legarreta.

Otros integrantes: Lucas Iacobellis, Nicolas Varela y Ana Etchebarne.

-Problemas y soluciones:

Problema: Los import no funcionaban (import \*)

Solución*:* Con ayuda del profesor y cambiar al formato from --- import \*

Problema: Edición de entidades, que solo se edite lo que el usuario quiere y no todo.

Solución: Una función para cada entidad donde recibe por parámetro la lista de diccionarios y el id a modificar. Luego pregunta qué campo del diccionario

se quiere editar. Solo se pueden editar instructores en estado 'Activo*'*

Problema: Repetición de clases

Solución: Al querer utilizar tuplas tuvimos que modificar la función de crear una clase, ya que para validar la clase usamos una función donde pedimos el dia y la hora y comparamos que no exista una clase con el mismo dia y hora(dentro d las que se encuentran activas)

-Sprints:

Sprint 1

-Definición de matrices y entidades.

-Crear el repositorio en Git y probar commits de todos.

-Subir primer código con funciones básicas (cargar datos en matriz, mostrar lista).

-Documentar en informe: descripción inicial del proyecto + datos del equipo + Scrum Master.

Sprint 2

-Implementar CRUD básico: Crear, Leer, Eliminar.

-Manejo de cadenas (ej: mayúsculas, validaciones simples).

-Informe: estructura de menús definida.

Sprint 3

-Reformar el CRUD usando funciones bien separadas.

-Agregar función Actualizar.

-Implementar consulta con datos relacionados.

-Informe: entidades involucradas.

Sprint 4

-Migrar entidades de matrices a diccionarios.

-Implementar primeras estadísticas: conteo total, conteo por categoría.

-Validamos tipos de datos e información ingresada por el usuario.

-Informe: problemas encontrados y soluciones.

Sprint 5

-Usar tuplas para devolver múltiples valores.

-detectar duplicados y manejar categorías únicas.

-Implementar estadísticas avanzadas: Cantidad de clases por instructor, cantidad de

socios por tipo de abono, promedio de activos e inactivos, cantidad de asistencias por clases.

-Completar informe: backlog final + detalle de sprints.

-Revisar prolijidad del código y commits de todos los integrantes.