Enunciado para la creación de una aplicación Java de escritorio para la administración de un gimnasio (31-05-2024)

Objetivo: Desarrollar una aplicación Java de escritorio que permita administrar las actividades y servicios de un gimnasio. La aplicación debe gestionar la información de socios, entrenadores, clases, y membresías. La aplicación debe interactuar con una base de datos para almacenar y recuperar la información de manera eficiente.

Requisitos del Sistema:

Gestión de Socios:

- Los usuarios deben poder añadir nuevos socios a la base de datos especificando el nombre, apellido, edad, y detalles de contacto.
- Los usuarios deben poder visualizar la lista de socios.
- Los usuarios deben poder buscar socios por nombre o número de socio.

Gestión de Entrenadores:

- Los usuarios deben poder añadir nuevos entrenadores a la base de datos especificando el nombre y especialidad.
- Los usuarios deben poder visualizar la lista de entrenadores.
- Los usuarios deben poder buscar entrenadores por nombre y/o apellido o especialidad.

Gestión de Clases:

- Los usuarios deben poder añadir nuevas clases a la base de datos especificando el nombre de la clase, entrenador asignado, y horario.
- Los usuarios deben poder visualizar la lista de clases disponibles.
- Los usuarios deben poder buscar clases por nombre, entrenador u horario.
- Inscribir un socio en una clase determinada.

Gestión de Membresías:

- Los usuarios deben poder registrar nuevas membresías especificando el socio, cantidad de pases, y duración.
- Los usuarios deben poder visualizar el historial de membresías con posibilidad de filtrado por socio.
- Los usuarios deben poder renovar o cancelar membresías existentes.

Requisitos Técnicos:

Interfaz de Usuario:

La aplicación debe tener una interfaz gráfica de usuario (GUI) sencilla y fácil de usar creada con Swing.

Persistencia de Datos:

La aplicación debe utilizar JDBC para interactuar con una base de datos MySQL.

Las tablas de la base de datos deben ser creadas según el siguiente esquema:

Tabla: Socios (Socio)

ID_Socio (INT PK): Identificador único del socio.

DNI (VARCHAR): Documento Nacional de Identidad del socio que sea único.

Nombre (VARCHAR): Nombre del socio.

Apellido (VARCHAR): Apellido del socio.

Edad (INT): Edad del socio.

Correo (VARCHAR): Correo electrónico del socio que sea único.

Teléfono (VARCHAR): Número de teléfono del socio.

Estado(BOOLEAN): Indica si la entidad está activa o dada de baja.

Tabla: Entrenadores (Entrenador)

ID_Entrenador (INT PK): Identificador único del entrenador.

DNI (VARCHAR): Documento Nacional de Identidad del entrenador que sea único.

Nombre (VARCHAR): Nombre del entrenador.

Apellido (VARCHAR): Nombre del entrenador.

Especialidad (VARCHAR): Especialidad del entrenador.

Estado(BOOLEAN): Indica si la entidad está activa o dada de baja.

Tabla: Clases (Clase)

ID_Clase (INT PK): Identificador único de la clase.

Nombre (VARCHAR): Nombre de la clase.

ID_Entrenador (INT FK): Identificador del entrenador asignado.

Horario (TIME): Horario de la clase.

Capacidad (INT): Capacidad total de alumnos por clase.

Estado(BOOLEAN): Indica si la entidad está activa o dada de baja.

Tabla: Membresías (Membresía)

ID_Membresía (INT PK): Identificador único de la membresía.

ID_Socio (INT FK): Identificador del socio que tiene la membresía.

CantidadPases (INT): (8, 12 o 20, se debe decrementar cada vez que el alumno consume una inscripción a una clase)

Fecha_Inicio (DATE): Fecha de inicio de la membresía.

Fecha_Fin (DATE): Fecha de fin de la membresía 30 días a partir de la fecha de incio.

Costo (DECIMAL): Precio de membresía mensual.

Estado(BOOLEAN): Indica si la entidad está activa o dada de baja.

Tabla: Asistencia(Socio-Clase)

ID_Asistencia (INT PK): Identificador único de la inscripción.

ID_Socio (INT FK): Identificador del socio que se inscribe.

ID_Clase (INT FK): Identificador de la clase a la que se inscribe.

Fecha_Asistencia: (DATE): Fecha de fin de la inscripción.

Relaciones entre tablas:

Socios y Membresías: Un socio puede tener múltiples membresías a lo largo del tiempo, pero una membresía está asociada a un solo socio.

Clases y Entrenadores: Una clase está asociada a un entrenador específico, pero un entrenador puede impartir múltiples clases.

Socios y Clases: Un socio puede asistir a múltiples clases, y una clase puede tener múltiples socios en su asistencia teniendo el cuenta el cupo de la clase. Cada vez que un socio asiste a una clase, se deja un registro de la asistencia con fecha actual y se actualiza la membresía descontando un pase.

Operaciones CRUD:

La aplicación debe implementar las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) para todas las entidades (Socios, Entrenadores, Clases, Membresías).

Validación de Datos:

La aplicación debe incluir validaciones adecuadas para todos los campos de entrada (por ejemplo, formatos de texto, longitud, datos requeridos). **Asegurarse de que no asistan más alumnos para una clase específica, que la capacidad máxima permitida de la misma, que no tenga la membresía vencida al momento de registrar asistencia del socio y no haya consumido los pases contratados.**

Asegurarse que cuando se asigna un entrenador a una clase, no tenga otra clase en el mismo horario.

Manejo de Errores:

La aplicación debe manejar errores de manera adecuada, mostrando mensajes de error informativos al usuario en caso de problemas (como intentos de ingresar datos duplicados, campos vacíos obligatorios, etc.).

Observaciones:

- Asumiremos que una clase se dicta durante toda la semana en el horario especificado.
- Todas las clases tienen una hora de duración.