Universidad Nacional del Nordeste

Facultad de Ciencias Exactas y Naturales y Agrimensura

Ingeniería del Software II

Año:2024

Proyecto: Gymsy

Profesores: Maria de los Angeles Ferraro,Laura Ines Gomez Solis.

Integrantes: Gonzalez Rodrigo Alejandro,
Gonzalez Jeremias Ezequiel,
Maciel Lautaro,
Lezana Mauricio Sebastian.

ÍNDICE

Índice de Tabla

Índice de Figuras

1. Introducción

Gymsy es una aplicación para administrar tu gimnasio de manera eficiente y efectiva. Ya no se tendrá que lidiar con montañas de papeleo, largas horas de seguimiento manual o confusiones en la programación de pagos. Permite gestionar todos los aspectos de tu gimnasio en un solo lugar. Desde el registro y seguimiento de clientes hasta la programación y la gestión de pagos, nuestra aplicación simplifica cada paso del proceso.

1.1 Breve estado del arte

Es esencial que los servicios relacionados con la salud y el bienestar se adapten constantemente a las demandas de la población, aprovechando las tecnologías disponibles y su rápido avance. En este sentido, el uso de aplicaciones y software para la gestión de gimnasios se ha vuelto cada vez más común, permitiendo una administración más eficiente y efectiva de estos establecimientos.

En países desarrollados, la modernización de los servicios de salud y bienestar ha llevado a la adopción generalizada de aplicaciones y software especializados. Sin embargo, muchas de estas soluciones son costosas y pueden no adaptarse completamente a las necesidades de todos los gimnasios.

Gymsy surge como una solución para abordar esta problemática, ofreciendo una plataforma integral para la gestión de gimnasios que simplifica cada aspecto del proceso, desde el registro y seguimiento de clientes hasta la programación y gestión de pagos. Al aprovechar la información disponible y utilizarla de manera eficiente, Gymsy busca reducir los costos de desarrollo y ofrecer una solución accesible para gimnasios de todos los tamaños.

1.2 Objetivo

El objetivo principal de Gymsy es proporcionar una herramienta informática que permita una gestión eficiente de los gimnasios, simplificando las tareas

administrativas y mejorando la experiencia tanto para los propietarios como para los clientes.

1.3 Fundamentación

La necesidad de una gestión más eficiente en los gimnasios se hace evidente en la falta de herramientas adecuadas para la comunicación y coordinación entre diferentes áreas dentro del establecimiento. Gymsy busca abordar esta problemática mediante el desarrollo de una plataforma que centrice la información y facilite la comunicación entre los distintos departamentos del gimnasio, optimizando así el trabajo y mejorando la experiencia del cliente.

2. Metodología

2.1 Ciclo de vida

El proceso de producción de cualquier aplicación informática lleva consigo realizar una serie de tareas repartidas en cinco etapas, llamadas: Análisis, Diseño, Codificación, Pruebas. A estas cuatro etapas se las conoce como ciclo de vida de un producto software o, dicho de otra forma, el ciclo de vida de un programa son las distintas etapas por las que éste tiene que pasar durante su existencia. Para este proyecto utilizaremos la metodología Scrum, que es un marco de trabajo ágil para la gestión y desarrollo de proyectos. Se basa en la colaboración, la adaptabilidad y la entrega incremental del producto. Elegimos Scrum por encima de otras metodologías debido a que proporciona las siguientes
ventajas:

1. **Flexibilidad**: Scrum es una metodología de gestión de proyectos muy flexible que permite adaptarse fácilmente a cambios en los requisitos del proyecto. Los sprints cortos y la capacidad de priorizar y planificar tareas durante el proyecto permiten a los equipos de Scrum responder rápidamente a cambios en el proyecto.
2. **Colaboración**: Scrum fomenta una colaboración efectiva entre los miembros del equipo. La estructura de equipo autoorganizado de Scrum fomenta una comunicación más abierta y una mayor colaboración entre los miembros del equipo, lo que puede resultar en una mejor calidad del trabajo y una mayor satisfacción del equipo.
3. **Entregas frecuentes**: Scrum promueve la entrega frecuente y continua de incrementos de trabajo completados y evaluables, lo que permite a los interesados obtener comentarios tempranos sobre el producto y realizar ajustes en consecuencia.
4. **Mayor control del proyecto**: Scrum proporciona una mayor visibilidad y control sobre el progreso del proyecto, ya que los equipos de Scrum realizan

reuniones diarias, reuniones de revisión de sprint y reuniones de retrospectiva para revisar el trabajo completado y planificar el siguiente sprint.

5. **Mejora continua**: Scrum fomenta la mejora continua en el proceso de desarrollo de software, lo que puede llevar a una mayor eficiencia y una mayor calidad del producto a lo largo del tiempo.

2.2 Método de Educción

Se determinó que el método de educación más apropiado para el proyecto es “Entrevistas”, ya que este método permite conocer el estado actual del sistema, metas organizacionales, identificar requisitos innecesarios, etc. Lo que a su vez simplifica el proceso de desarrollo y mejora la calidad del producto final.
La estructura de la entrevista es en forma de “Diamante” para la conformación del método de educación seleccionado lo que significa que las preguntas empezaran siendo específicas, a medida que avance la entrevista preguntas más generales y para finalizar preguntas específicas.
A continuación, un extracto de la entrevista llevada a cabo:
1. ¿En el sistema se registrará a los clientes?

- "Sí, el sistema debe contar con un formulario de registro para nuevos clientes, la información."

2. ¿Qué información debe tener un plan?

- "Debe tener el título, la descripción, precio e instructor a cargo."

3. ¿Cuando se ingrese un dato erróneo como quiere el sistema lo informe?

- "El sistema debe validar lo ingresado por el operador y mostrar un mensaje de error en caso de datos no válidos."

4. El sistema debe contar con una forma de identificación única ¿Prefiere que se haga automáticamente o manualmente?

- "Que el sistema lo genere automáticamente."

5. ¿Cómo se gestionan los roles de los usuarios del sistema?

- "Deben existir de forma predeterminada un administrador y un recepcionista, luego el administrador puede agregar más instructores y estos a los clientes algo automático."

6. ¿Cómo se debe visualizar a los clientes?

- "El sistema debe permitir la visualización de una lista de clientes asociados a un instructor específico."

7. ¿Puede el instructor agregar clientes y realizarles seguimiento?
    - "Sí, el instructor tiene la capacidad de agregar clientes y hacerles seguimiento."

8. ¿Cómo se deben visualizar los usuarios?
    - "El sistema proporciona la opción de visualizar una lista de todos los usuarios registrados, junto con sus roles asignados."

9. ¿Desde que sistema operativo cuentan para usar el sistema?
    - "Windows 7 en adelante."

10. ¿Cómo le gustaría que fuera la aplicación?
    - "La aplicación debe ser fácil de usar y contar con una interfaz intuitiva."

11. Respecto a la fluidez de las acciones del usuario ¿Como deberían ser?
    - "La aplicación debe tener un tiempo de respuesta rápido para las acciones del usuario."

12. Sobre la fecha y hora que se manejen el sistema ¿Debe ser la del sistema operativo?
    - "Sí, la aplicación debe integrarse con el calendario del sistema operativo."

13. ¿Cuáles son las normas que debe cumplir la aplicación?
    - "La aplicación debe cumplir con legislación, regulaciones y leyes locales, principalmente sobre privacidad de datos y todo respecto a temas referentes a gimnasios."

14. ¿La aplicación debe garantizar la privacidad absoluta de la información de los clientes?
    - "Sí, la aplicación garantiza la privacidad absoluta de la información de los clientes."

2.3 Arquitectura Utilizada

La arquitectura utilizada fue MVP(Modelo Vista Presentador)

Se emplea principalmente en entornos donde se requiere una clara separación de responsabilidades entre la lógica de negocio, la presentación de la interfaz de usuario y la gestión de eventos.

Aquí hay algunas razones por las que podrías considerar utilizar la arquitectura MVP:

- **Separación de responsabilidades:** MVP permite separar claramente la lógica de negocio (modelo), la presentación de la interfaz de usuario (vista) y la gestión de eventos (presentador). Esto facilita la comprensión del código y el mantenimiento a largo plazo.
- **Facilita las pruebas unitarias:** Al separar la lógica de presentación de la lógica de negocio, se facilita la escritura de pruebas unitarias para cada componente por separado. Esto mejora la calidad del código y facilita la detección de errores.
- **Reutilización del código:** La separación de la lógica de presentación y la lógica de negocio permite reutilizar los componentes de manera más efectiva. Por ejemplo, puedes utilizar la misma lógica de negocio con diferentes interfaces de usuario.
- **Adaptabilidad a cambios en la interfaz de usuario:** Si necesitas realizar cambios en la interfaz de usuario, la arquitectura MVP te permite hacerlo sin afectar la lógica de negocio subyacente. Esto es útil en situaciones donde la interfaz de usuario debe ser actualizada con frecuencia.
- **Escalabilidad:** MVP es una arquitectura escalable que puede adaptarse fácilmente a proyectos de diferentes tamaños y complejidades. Desde aplicaciones pequeñas hasta grandes sistemas empresariales, MVP puede proporcionar una base sólida para el desarrollo.

  En resumen, la arquitectura Modelo-Vista-Presentador (MVP) es una opción sólida para proyectos donde se valora la separación de responsabilidades, la facilidad de prueba, la reutilización del código y la capacidad de adaptarse a cambios en la interfaz de usuario.

2.4 Especificación de Requerimientos de Software (ERS IEEE-STD-830-1998).

2.4.1 Introducción

El presente documento es una Especificación de Requisitos de Software perteneciente al desarrollo de un sistema para la gestión de clientes, ventas y pagos de la aplicación de Gymsy. Está estructurado según las directivas dadas por el estándar de IEEE 830. Esta especificación está dirigida a los desarrolladores del Sistema, al equipo de calidad, usuarios finales y tiene como objetivo analizar y documentar los requisitos funcionales y no funcionales del futuro sistema.

2.4.2 Propósito

El propósito de la propuesta es la de realizar una aplicación especialmente diseñada para la empresa “Gymsy” en el cual se brinda el servicio de compras online. El objetivo es informatizar el proceso de ventas, pagos a los empleados y

administración de planes de entrenamiento a fin agilizar los procesos y reducir costos.

2.4.3 Alcance

El sistema permitirá el alta y baja de planes de entrenamiento, instructores y clientes. Generar informes de cantidad de clientes por instructor, ganancias en el tiempo al administrador. Agregar los progresos del cliente. Visualizar clientes, instructores, planes y pagos realizados dentro de Gymsy.

2.4.4 Personal Involucrado

Nombre	Gauna Octavio
Rol	Analista, diseñador y programador
Responsabilidad	Diseñador/Programador
Información contacto	gaunavictoroctavio@gmail.com

Nombre	Jeremías Ezequiel Gonzales
Rol	Analista, diseñador y programador
Responsabilidad	Diseñador/Programador
Información contacto	jeremiasgonzalez7464@gmail.com

Nombre	Gonzalez Rodrigo Alejandro
Rol	Analista, diseñador y programador
Responsabilidad	Diseñador/Programador
Información contacto	gonzlrodrigo@gmail.com

Nombre	Maciel Lautaro
Rol	Analista, diseñador y programador
Responsabilidad	Diseñador/Programador
Información contacto	lautaromaciel77@gmail.com

Nombre	Lezana Mauricio Sebastian
Rol	Analista, diseñador y programador
Responsabilidad	Diseñador/Programador
Información contacto	lezanamauricio86@gmail.com

2.4.5 Definiciones, acrónimos y abreviaturas

Nombre	**Descripción**
Usuario	Persona que usará el sistema para gestionar procesos
RF	Requerimiento Funcional
RNF	Requerimiento No Funcional

Usuario: Persona que usará el sistema para gestionar procesos
RF: Requerimiento Funcional
RNF: Requerimiento No Funcional

2.4.5 Referencias

Título del Documento	**Referencia**
Standard IEEE 830 1998	IEE
SIS-I	Sistema de Información Web para la Gestión de Procesos Administrativos y Académicos
RF	Requerimiento Funcional
RF	Requerimiento No Funcional

2.5 Requisitos específicos

2.5.1 Requerimientos Funcionales

RF#01: El sistema debe proporcionar un formulario para el registro de nuevos clientes.

RF#02: El formulario de creación de un nuevo plan debe incluir campos obligatorios, como el título, la descripción, precio,instructor a cargo, imágen (mínimo 2).

RF#03: El sistema debe validar los campos ingresados por el usuario y mostrar un mensaje de error si se ingresan datos inválidos.

RF#04: El sistema debe generar automáticamente un identificador único para cada nuevo usuario/plan creado.

RF#05: El sistema debe tener los roles administrador, recepcionista, instructor y cliente, los primeros dos son predeterminados, clientes e instructores se asignan cuando se crean sus perfiles automáticamente.

RF#08: El sistema debe permitir la visualización de una lista de todos los clientes asociados a un cierto instructor.

RF#10: El usuario instructor debe poder agregar clientes y hacerles un seguimiento.

RF#11: El sistema debe permitir la visualización de una lista de todos los usuarios registrados, con sus roles asignados.

2.5.2 Requerimientos no funcionales

2.5.3 Requerimientos no Funcionales Producto

RNF	Descripciòn del Requerimiento	Clasificación
RNF-01	La aplicación deberá ser compatible con Windows 7 en adelante.	Portabilidad
RNF-02	La aplicación deberá ser fácil de usar y con una interfaz intuitiva	Usabilidad

RNF-03	La aplicación deberá tener un tiempo de respuesta rápido para las acciones del usuario.	Rendimiento
RNF-04	La aplicación debe tener una arquitectura modular, permitiendo la integración de nuevas funcionalidades con facilidad	Portabilidad

Tabla 1

2.5.4 Requerimientos no Funcionales Externos

RNF	**Descripciòn del Requerimiento**	**Clasificación**
RNF-01	La aplicación deberá integrarse con el calendario del sistema operativo	Interoperabilidad
RNF-02	La aplicación deberá cumplir con las regulaciones y leyes locales sobre privacidad de datos	Legislativos
RNF-03	La aplicación debe cumplir con los estándares de seguridad.,	Seguridad
RNF-04	La aplicación deberá cumplir con la privacidad absoluta de la información de los clientes	Privacidad
RNF-05	La aplicación deberá regirse por normas y reglas basada en la ética -	Éticos

Tabla 2

2.5.5 Requerimientos no Funcionales Organizacional

RNF	**Descripciòn del Requerimiento**	**Clasificación**
RNF-01	La especificación de requerimientos de software deberá realizar según la norma IEEE 830	Estándares

RNF-02	El diseño de la aplicación debe desarrollarse con herramienta CASE, por ejemplo: StarUML	Estándares
RNF-03	El apartado front-end se deberá desarrollar con las tecnologías: .NET Framework con C#	Implementación
RFN-04	El apartado back-end se deberá desarrollar con las tecnologías: .NET Framework con C#	Implementación

Tabla 3

2.5.6 Otros requisitos

Es importante que la aplicación contenga imágenes descriptivas, instructores tanto de hombres como mujeres, con variedad de edades.

Requerimiento de gestión del dinero dentro del programa (Educido por método de entrevista):

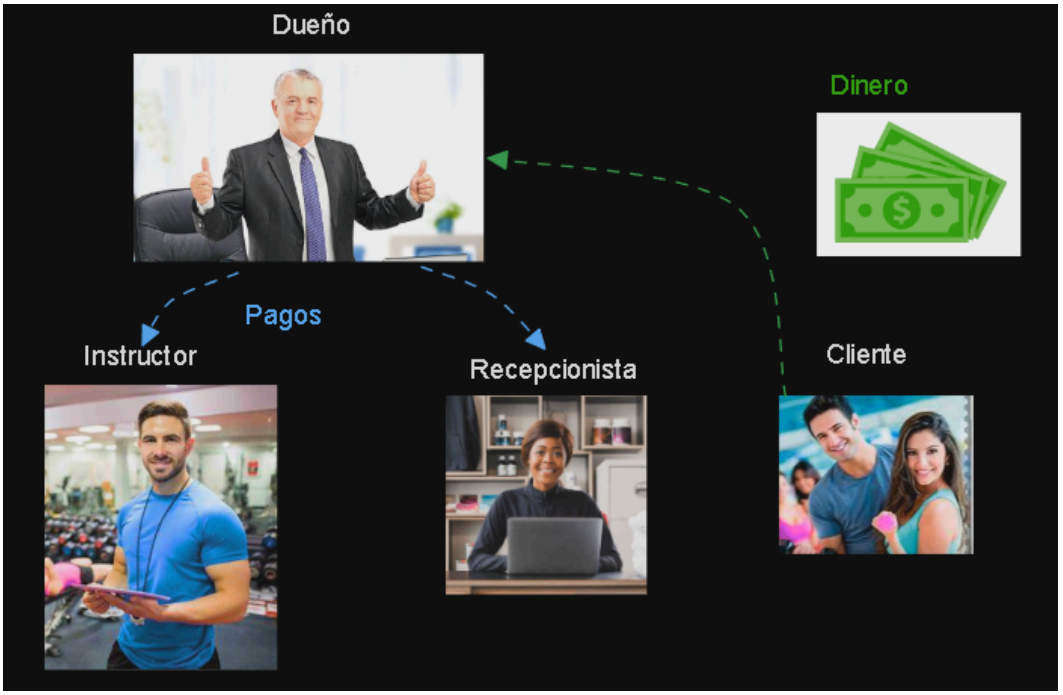

Figura 1

Bosquejo de vista del rol Recepcionista (Educido por método de entrevista):

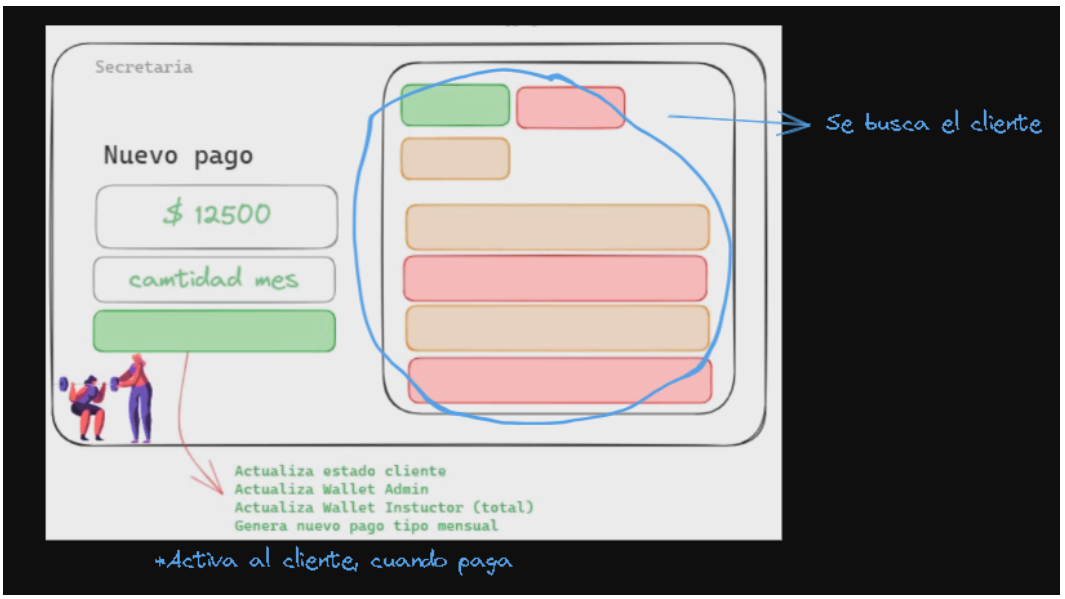

Figura 2

Paleta de Colores seleccionados en RGB:
Letras oscuras: 110; 118; 129
Fondo oscuro: 9; 0; 20
Fondo claro: 16; 8; 23
Fondo más Claro: 69; 34; 99
Fondo TextBox: 69; 34; 99
Botones Verdes: 41; 147; 45
Eliminaciones: 192; 0; 0
Ediciones: 0; 0; 192
Guardar: 192; 0; 0
Activar: 255;140;0

2.6 Análisis de Riesgo

2.6.1 Riesgo de producto

Problemas con la calidad del código y la seguridad:

- Probabilidad: Moderada
- Efecto: Crítico
- Estrategia: Implementar una política de control de calidad y pruebas de seguridad para garantizar que el código sea seguro y libre de errores antes de la implementación.

Requisitos incompletos o poco claros:

- Probabilidad: Baja
- Efecto: Moderado

- Estrategia: Establecer un proceso de revisión de requisitos para asegurarse de que sean claros y completos, y para identificar cualquier cambio necesario en el alcance del proyecto.

2.6.2 Riesgo de proyecto

El producto conlleva más tiempo de lo esperado (estimación):

- Probabilidad:moderada
- Efecto:Tolerable
- Estrategia: estimar los tiempos con nuevos cálculos basados en la experiencia que se obtuvo
- Subclasificación: estimación

Conflictos entre miembros del equipo de desarrollo:

- Probabilidad: Baja
- Efecto: Tolerable
- Estrategia: Promover la comunicación abierta y efectiva para prevenir y solucionar cualquier conflicto que pueda surgir.
- Subclasificación: personal

El personal involucrado no tiene experiencia lo que podría afectar los tiempos de entrega:

- Probabilidad: alta
- Efecto: moderado
- Estrategia: consultar a gente experimentada sobre resoluciones que toman un tiempo considerable
- Subclasificación: personal

2.6.3 Riesgo de Negocio

Solicitud de nuevos requerimientos por parte del cliente.

- Probabilidad: Baja
- Efecto: Alta
- Estrategia: Se tomará como actualización futura fuera del cronograma establecido.
- Subclasificación: Requerimientos

Cambios en las regulaciones o normativas:

- Probabilidad: Baja
- Efecto: Crítico
- Estrategia: Mantenerse actualizado sobre las regulaciones y normativas relevantes, y establecer un proceso para evaluar y ajustar el proyecto.

- ❖ Subclasificación: organizacional

2.6 Historias de Usuario

Los requisitos principales que se obtuvo en cuenta para desarrollar la aplicación fue:

Numero Requisito	#01	título	Crear y agregar usuarios
Estimación	9 días	Valor	10
RF		11	
Descripción		**Cómo** Admin **Quiero** que el sistema tenga un solo admin principal, una recepcionista, permita agregar varios instructores **Para** gestionar el negocio	
Dependencia		1	
Pruebas		El Sistema DEBE contar con los cuatro perfiles anteriormente especificados El Sistema NO DEBE tener más perfiles de usuario	

Tabla 4

Numero Requisito	#02	título	Crear planes de entrenamiento
Estimación	2 días	Valor	4
RF		2,3,4	
Descripción		**Como** instructor **Quiero** una vista en la cual se pueda crear planes personalizados **Para** asignarlos a un cliente	
Dependencia		1	
Pruebas		El instructor DEBE poder crear planes para sus clientes El sistema NO DEBE permitir planes con campos vacíos	

Tabla 5

Numero Requisito	#03	título	Agregar Clientes
Estimación	6 dias	Valor	8
RF		1,4	
Descripción		**Como** Instructor **Quiero** una vista **Para** agregar clientes	
Dependencia		1,3	
Pruebas		El Instructor DEBE poder introducir un cliente ingresando: nombre,apellido,nickname,contraseña,genero,dia de nacimiento, avatar, teléfono y cbu El sistema NO DEBE permitir ingresar clientes si se encuentra al menos un campo vacío El sistema NO DEBE permitir que se ingresen letras en los campos: teléfono,dia de nacimiento y cbu El sistema NO DEBE permitir que se ingresen números en los campos: nombre, apellido	

Tabla 6

Numero Requisito	#04	título	Visualizar clientes
Estimación	6 días	Valor	7
RF		8,10	
Descripción		**Como** Instructor **Quiero** una vista **Para** visualizar los clientes	
Dependencia		1,3	
Pruebas		El Instructor DEBE poder visualizar a los clientes del sistema El sistema NO DEBE mostrar otros roles en esta tabla	

Tabla 7

Numero Requisito	#05	título	Generar gráficos estadísticos sobre el negocio
Estimación	5 días	Valor	8
RF		5	
Descripción		**Como** Admin **Quiero** ver gráficos estadísticos sobre la ganancia por mes y la proporción de cantidad de clientes por cada instructor del top con mas cantidad **Para** tener información más accesible y poder tomar mejores decisiones sobre el negocio	
Dependencia		1,3	
Pruebas		El sistema DEBE tener un gráfico de líneas de ganancia por mes El Admin DEBE tener un de torta que muestre la proporción de cantidad de clientes por cada instructor del top con mas cantidad	

Tabla 8

Planificación

Id	Historia de Usuario	puntos usados
01	Crear y agregar usuarios	10
02	Crear planes de entrenamiento	4
03	Agregar Clientes	8
4	visualizar Clientes	7
5	Generar gráficos estadísticos sobre el negocio	8

Tabla 9

Tiempo disponible: 2 meses = 20 días por mes =40 días
Número de integrantes de desarrollo: 4
Número de Sprint: 4

Planificación de Sprint :
Se planifica un sprint de duración de 2 semanas
2 semanas = 10 días efectivos
5 integrantes
10 días efectivos * 4 integrantes = 40 días-hombre
Factor de dedicación = 7 días/40 días-hombre = 0.17
Velocidad estimada del nuevo sprint = 0.17 * 100 días-hombre= 17 puntos de historia

spring 1:

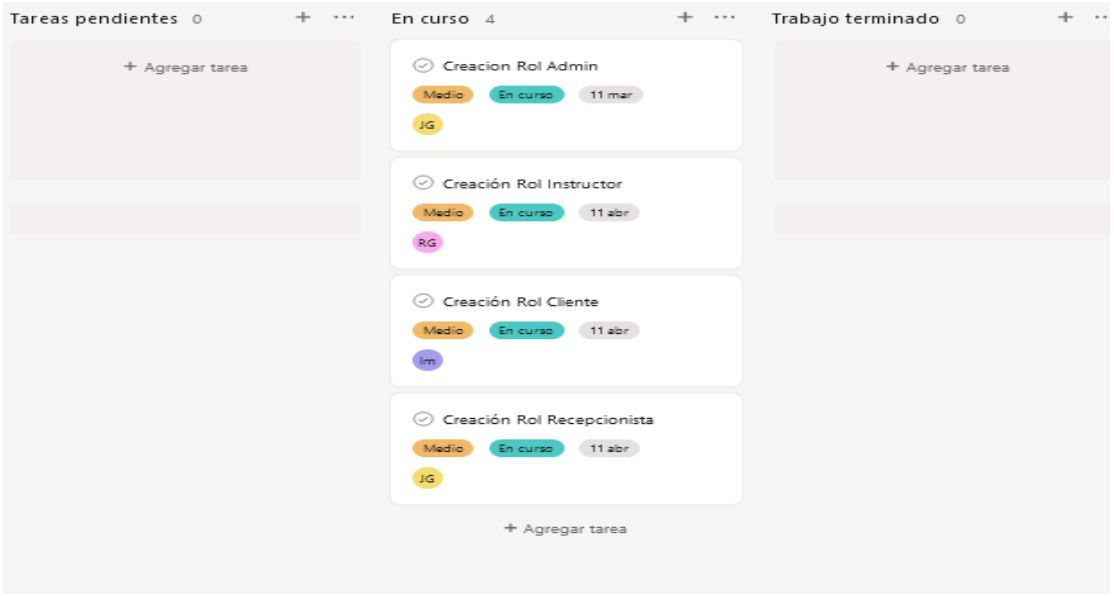

Figura 3

spring 2:

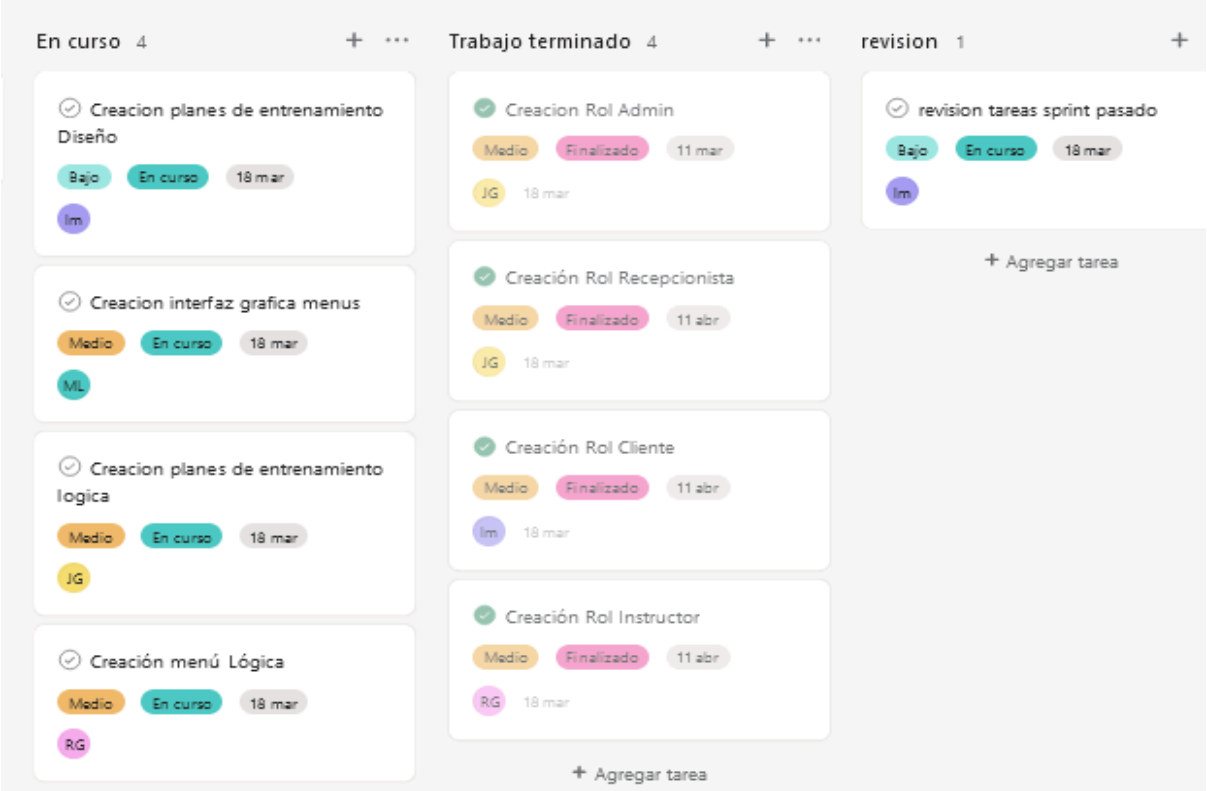

Figura 4

spring 3:

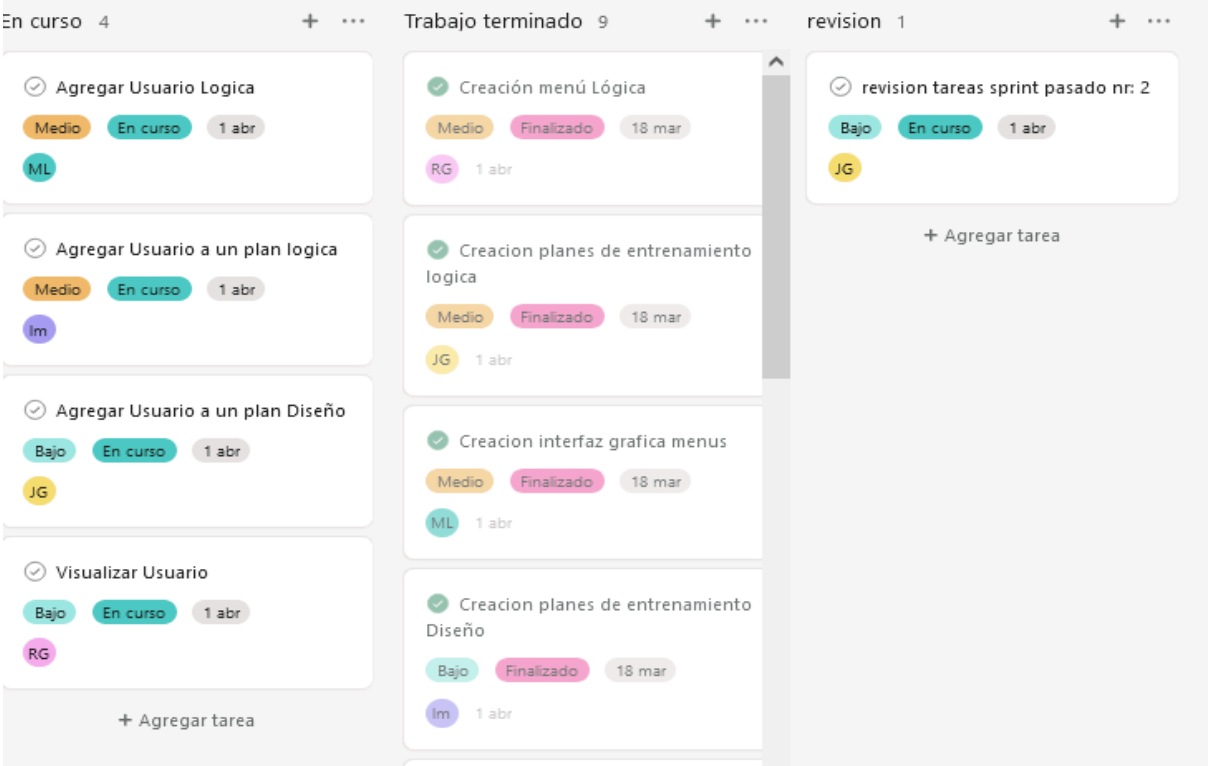

Figura 5

spring 4:

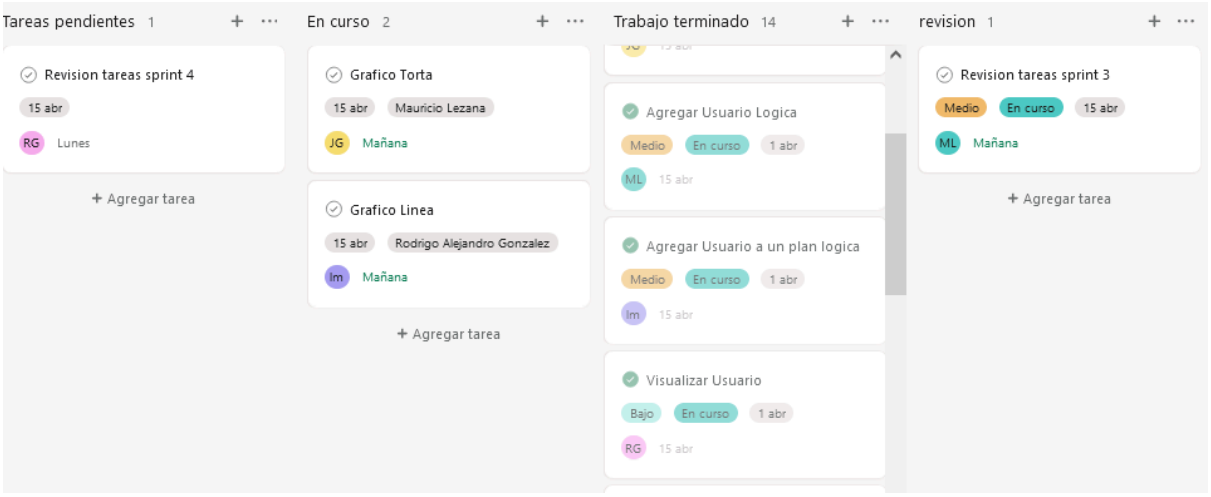

Figura 6

2.7 Casos de uso

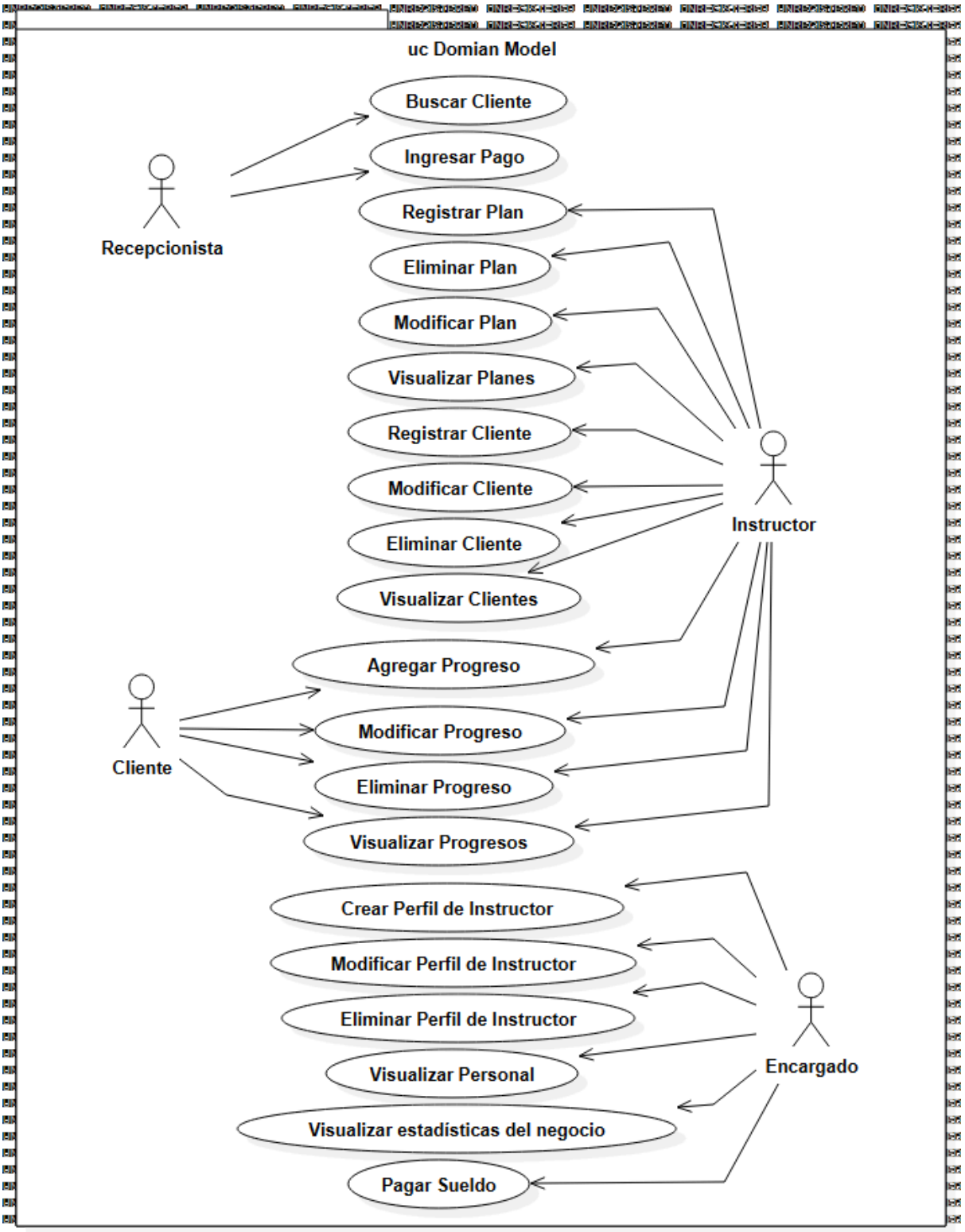

Figura 7

2.8 Conversación

Conversación: Caso de uso “Crear Perfil de Instructor”
Actor: Admin

Acción	Curso Normal	Curso Alternativo
1. A: Ingresa el nombre, apellido, un avatar, numero de telefono, fecha de nacimiento, género y un apodo		
2. S: Se comparan los datos ingresados con los registros existentes para evitar duplicados o errores en la información.	2.1 Los campos no estan vacios 2.2 se guardan los datos del instructor	2.1.1 Al Menos un campo esta vacio 2.1.2 Se muestra un mensaje “Revise y complete correctamente los campos.” 2.1.3 Volver a 1 2.2.1 ocurre algún error al guardar el instructor 2.2.2 Se muestra un mensaje “Error al guardar el perfil de instructor.” 2.2.3 volver a 1
3. S: se muestra el mensaje “Perfil del instructor guardado correctamente”		
4. Fin del caso de uso		

Tabla 10

Conversación: Caso de uso “Agregar Progreso”
Actor: Cliente-Instructor

Acción	Curso Normal	Curso Alternativo
1. A: Ingresa el título, el peso actual, la altura actual, notas sobre el progreso y una imagen		
2. S: Se comparan los títulos de los progresos	2.1 No hay campos vacios	2.1.1 Se encontró al menos un campo vacío

		2.1.2 Se muestra un mensaje “Verifique que los campos estén correctos.” 2.1.3 Ir a 1
	2.2 No hay otros progresos con ese título	2.2.1 Se encontró un título de progreso repetido 2.2.2 Se muestra un mensaje “Ya existe un progreso con ese título” 2.2.3 Ir a 1
	2.3 Se guarda el progreso	2.3.1 Se produce un error al guardar 2.3.2 Se muestra un mensaje “Error al guardar el progreso de entrenamiento” 2.3.3 Ir a 1
4. Se muestra un mensaje “Progreso guardado correctamente”		
5. Fin del caso de uso		

Tabla 11

Conversación: Caso de uso “Crear Plan”
Actor: Instructor

Acción	Curso Normal	Curso Alternativo
1. A: Ingresa el precio y la descripción		
2. S: Se compara la descripción con los demás planes para evitar duplicados	2.1 No hay campos vacios	2.1.1 Se encontró al menos un campo vacío 2.1.2 Se muestra un mensaje “Por favor, verifique que haya ingresado correctamente todos los campos.” 2.1.3 Ir a 1
	2.2 No hay otros planes con esa descripción	2.2.1 Hay planes con esa descripción 2.2.2 Se muestra un mensaje “Ya existe un plan con esa descripción”

	2.3 Se guarda el plan	2.2.3 Ir a 1 2.3.1 Se produce un error al guardar 2.3.2 Se muestra un mensaje “Error al guardar el plan” 2.3.3 Ir a 1
3. Se muestra un mensaje “Plan guardado correctamente”		
4. Fin del caso de uso		

Tabla 12

2.9 Diagramas de Secuencia

2.9.1 diagrama de secuencia “Crear Perfil Instructor”

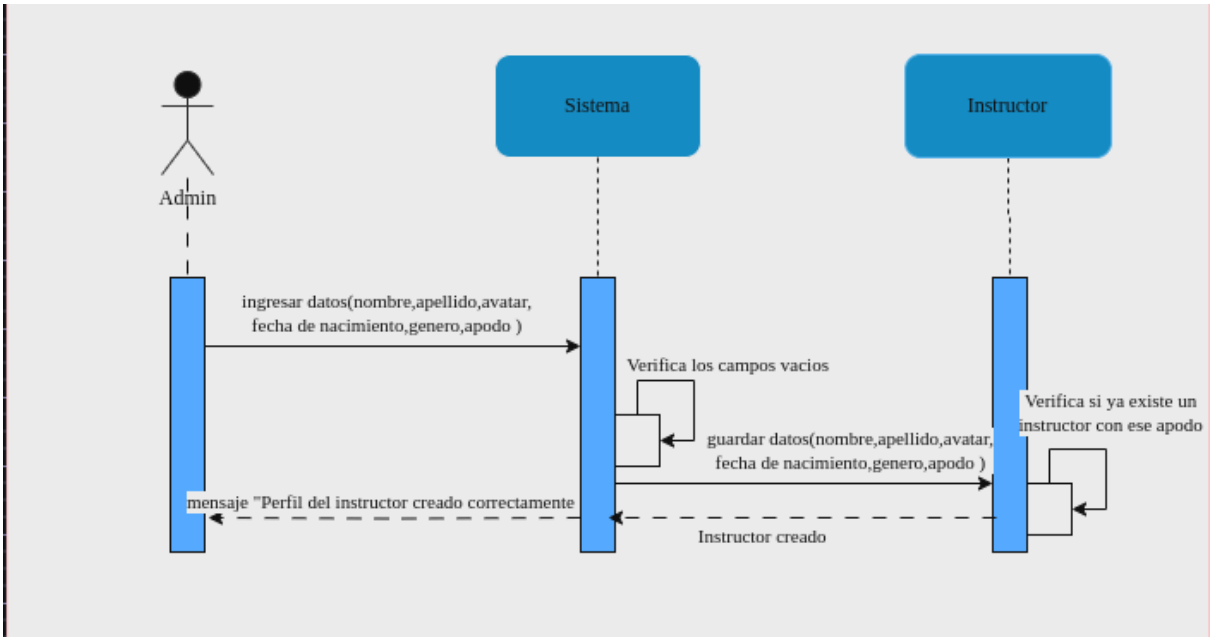

Figura 8

Diagrama de secuencia “Crear Perfil Instructor” alternativa 1

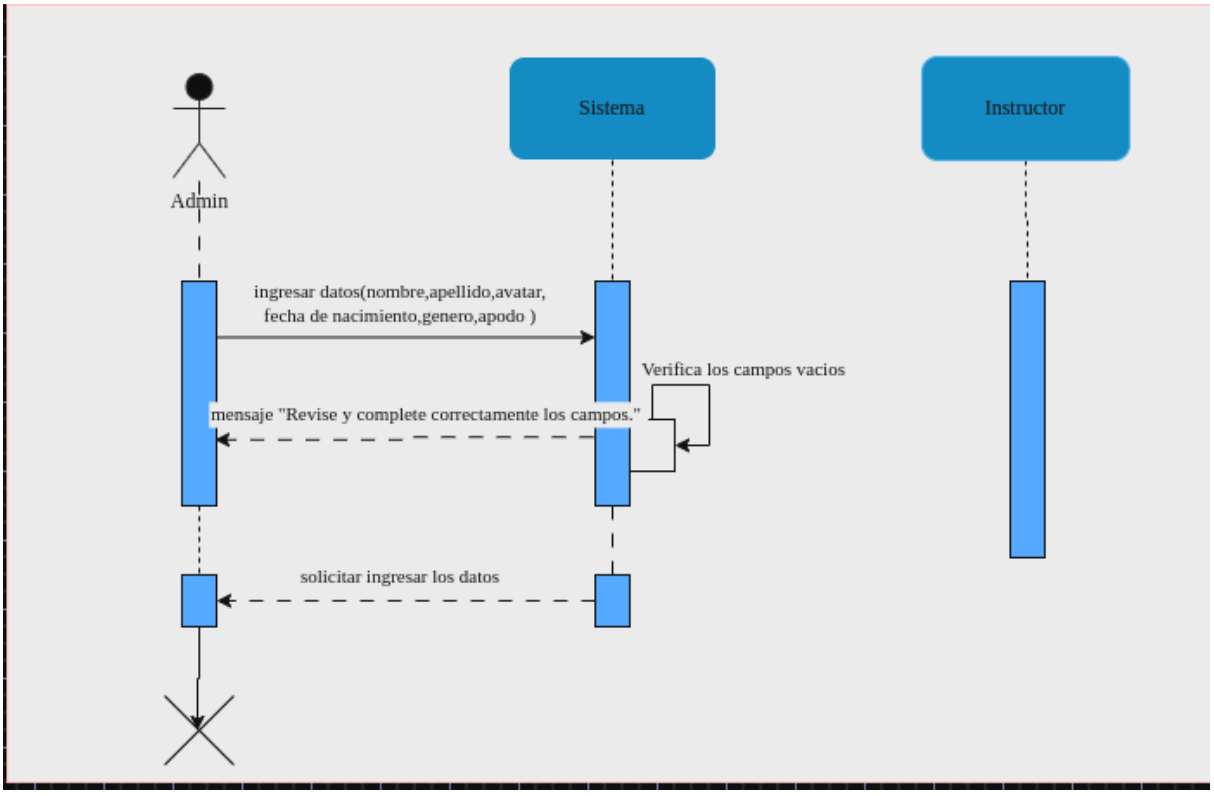

Figura 9

Diagrama de secuencia “Crear Perfil Instructor” alternativa 2

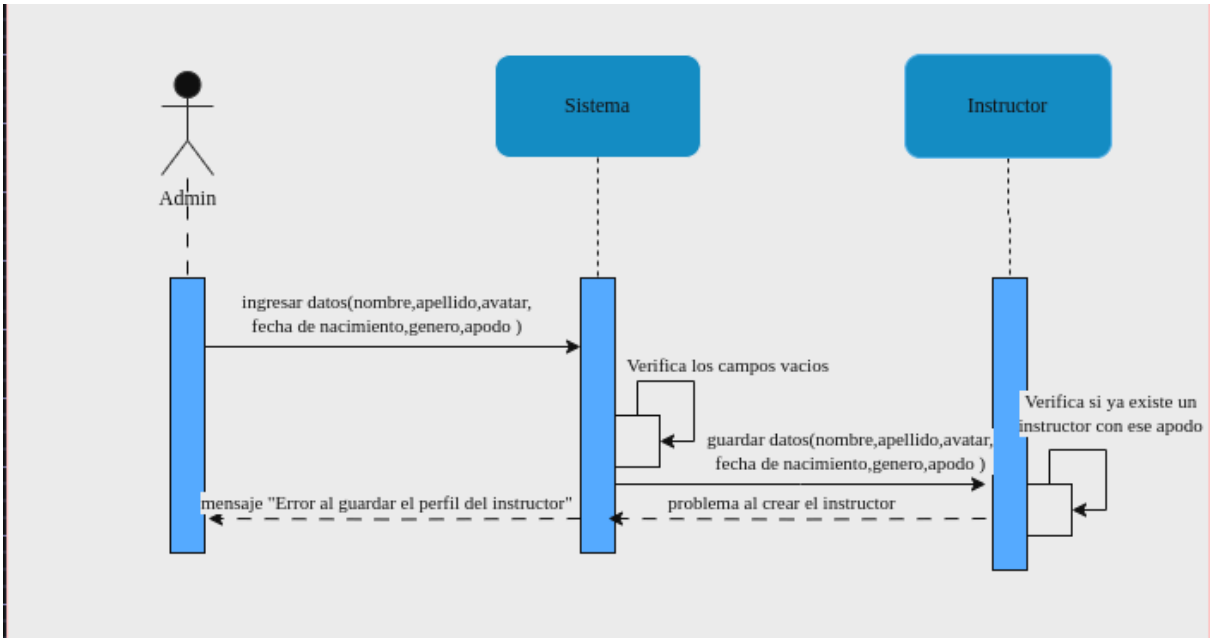

Figura 10

2.9.2 diagrama de secuencia “Agregar Progreso”

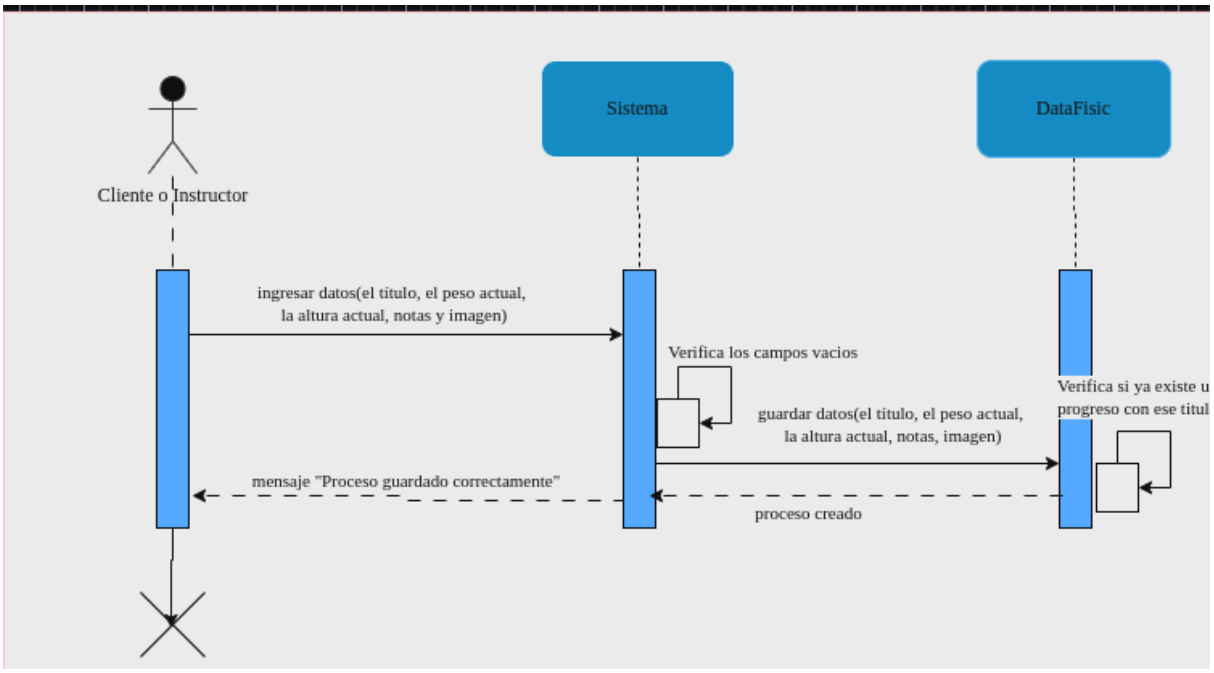

Figura 11

Diagrama de secuencia “Agregar Progreso” alternativa 1

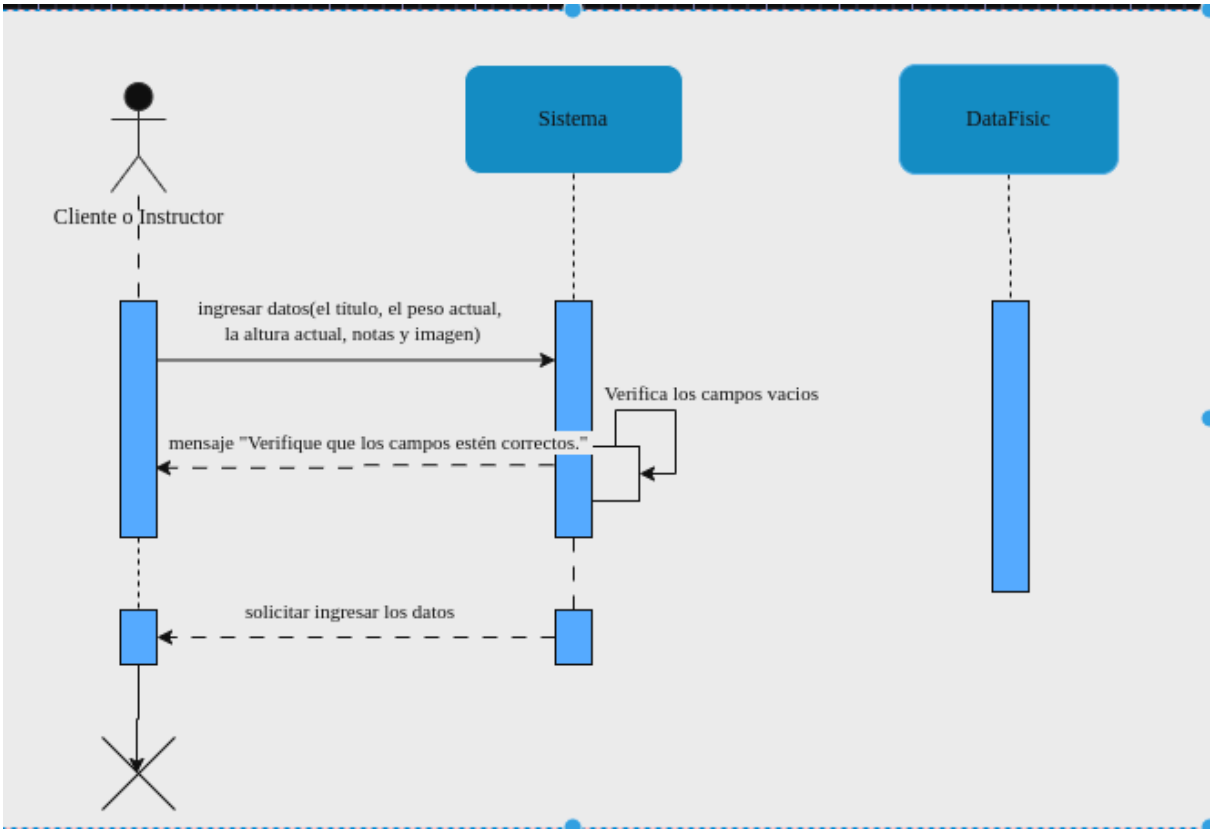

Figura 12

Diagrama de secuencia “Agregar Progreso” alternativa 2

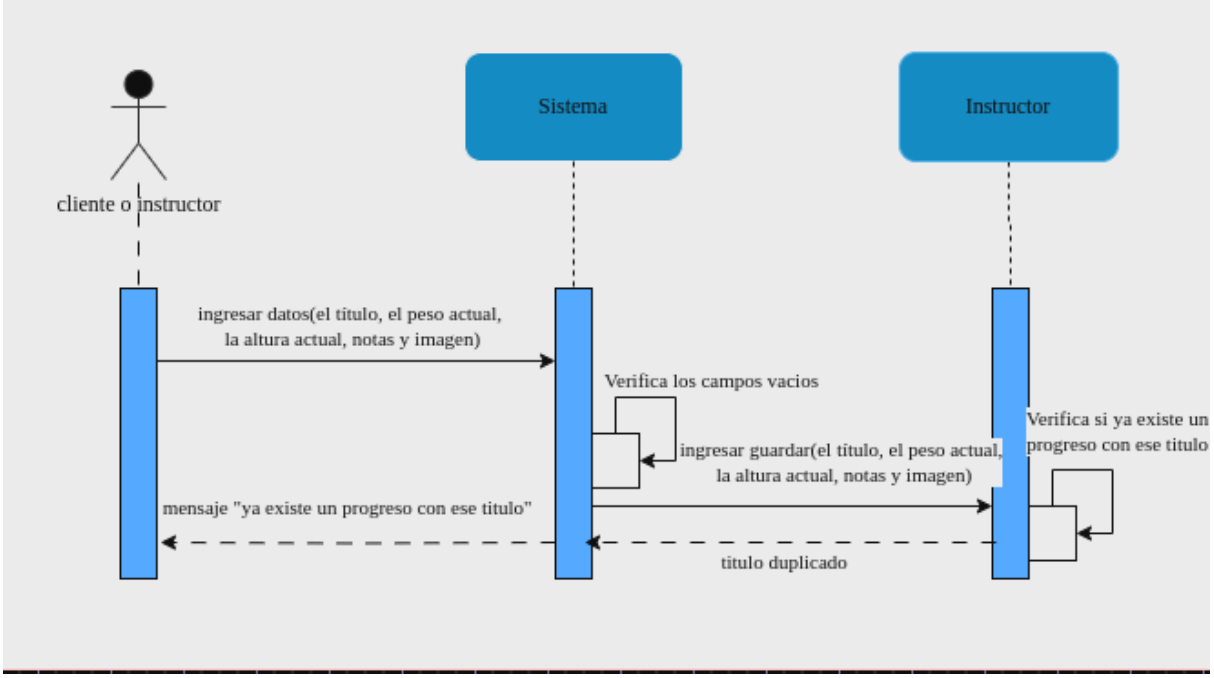

Figura 13

Diagrama de secuencia “Agregar Progreso” alternativa 3

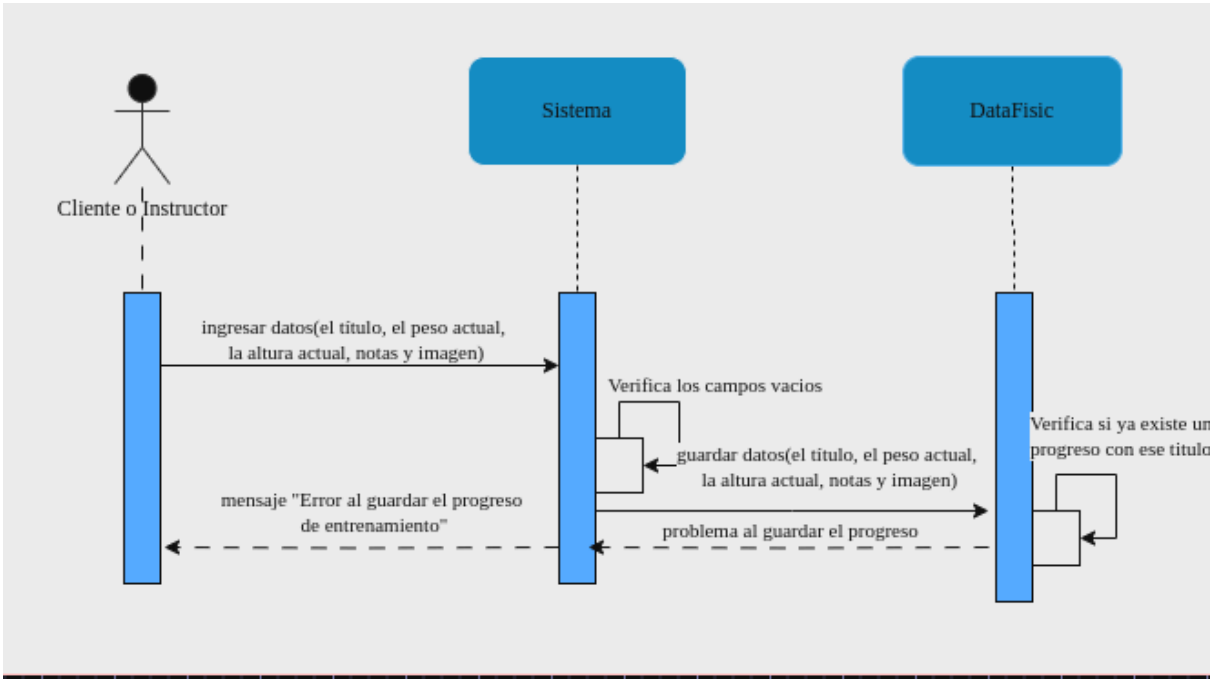

Figura 14

2.9.3 diagrama de secuencia “Crear Plan”

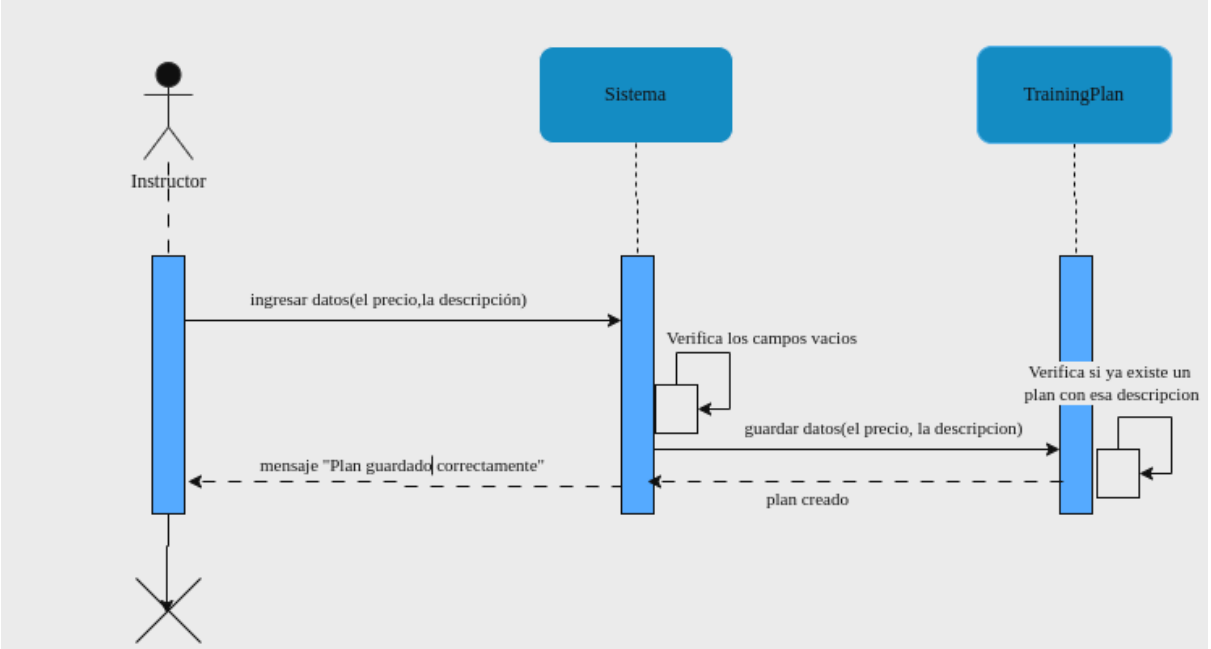

Figura 15

diagrama de secuencia “Crear Plan” alternativa 1

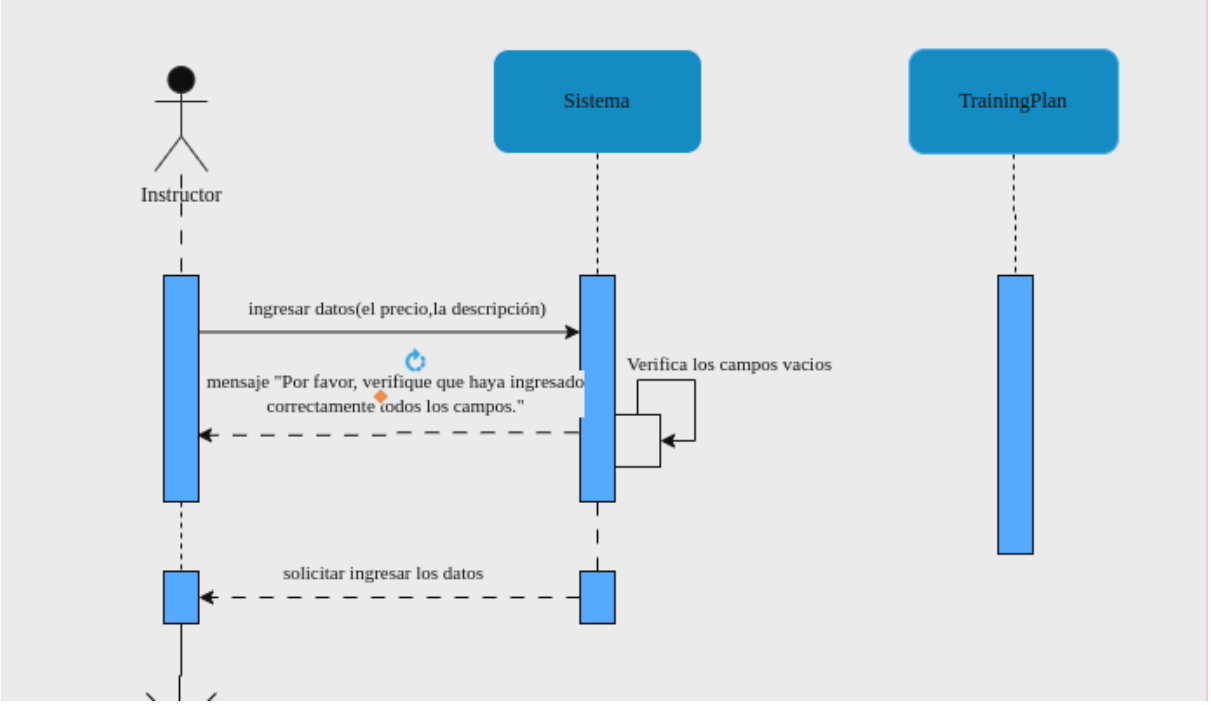

Figura 16

Diagrama de secuencia “Crear Plan” alternativa 2

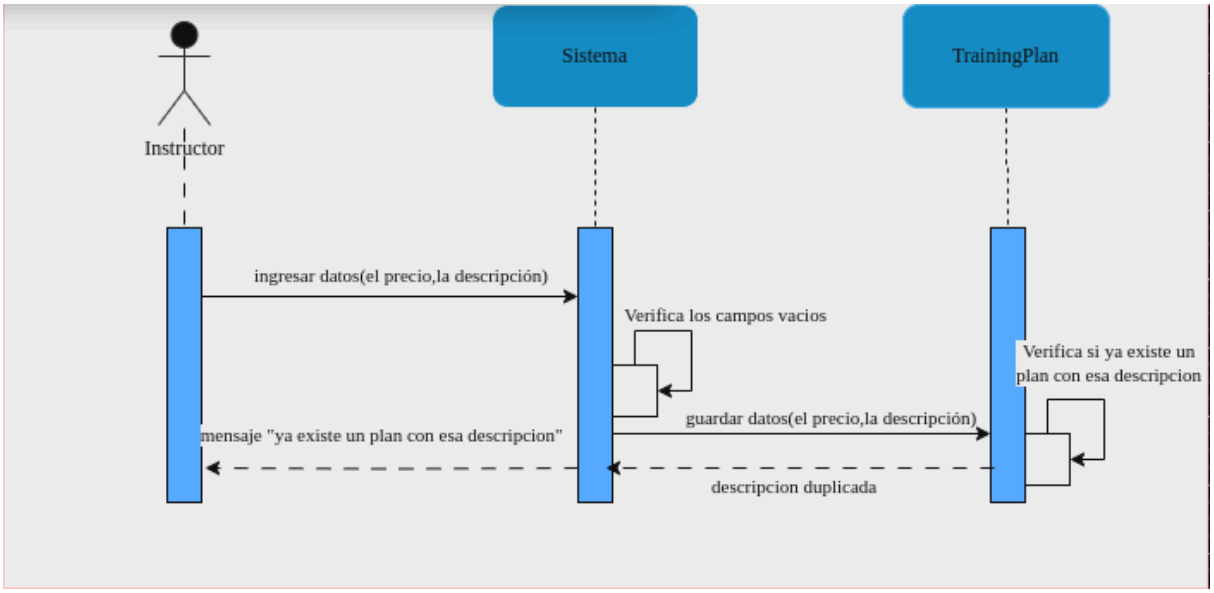

Figura 17

diagrama de secuencia “Crear Plan” alternativa 3

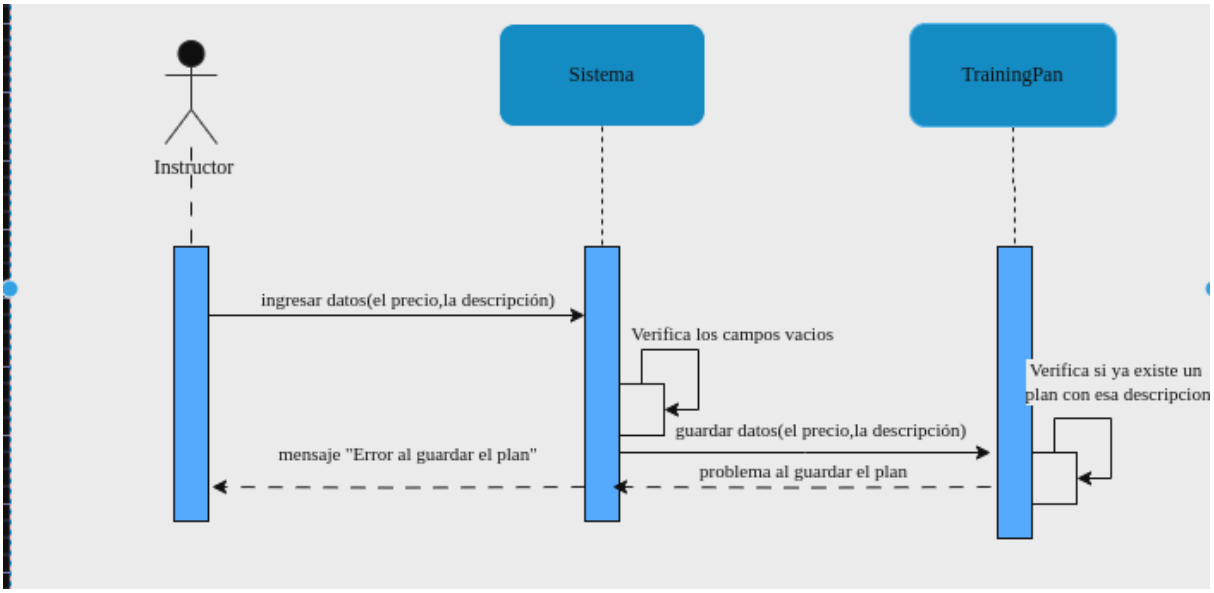

Figura 18

2.9.4 Contratos de operaciones

Nombre: AgregarPerfilInstructor(nombre, apellido, avatar, fecha_nacimiento, género, apodo)
Referencia cruzada: Agregar Perfil de instructor
Responsabilidades: Crear un nuevo perfil de instructor en base a los datos ingresados.
Excepciones:

- Si algún campo se encuentra vacío debe mostrar un mensaje, y permitir que vuelva a cargar el dato.

- ❖ Si el apodo del instructor ya existe se debe informar con un mensaje

Precondición:el usuario debe estar logueado y tener el rol de Administrador
Postcondición: se informa que se creó correctamente el instructor.

Nombre: AgregarProgreso(título, peso_actual, altura_actual, imagen)
Referencia cruzada: Agregar Progreso
Responsabilidades: Crear un nuevo progreso para un cliente específico en base a los datos ingresados.
Excepciones:

- ❖ Si algún campo se encuentra vacío debe mostrar un mensaje, y permitir que vuelva a cargar el dato.
- ❖ Notificar a través de un mensaje si ya existe un progreso con el mismo título.

Precondición: el usuario debe estar logueado y debe tener el rol de cliente o instructor.
Postcondición: se crea un progreso nuevo.

Nombre:CrearPlan(precio,descripción)
Referencia cruzada: Crear Plan
Responsabilidades: Crear un nuevo plan de entrenamiento en base a los datos ingresados.
Excepciones:

- ❖ Si algún campo se encuentra vacío debe mostrar un mensaje, y permitir que vuelva a cargar el dato.
- ❖ Se debe informar desde si ya existe un plan con la misma descripción.

Precondición: el usuario debe estar logueado con el rol de instructor.
Postcondición: se creará un nuevo plan de entrenamiento.

2.10 Diagrama DER

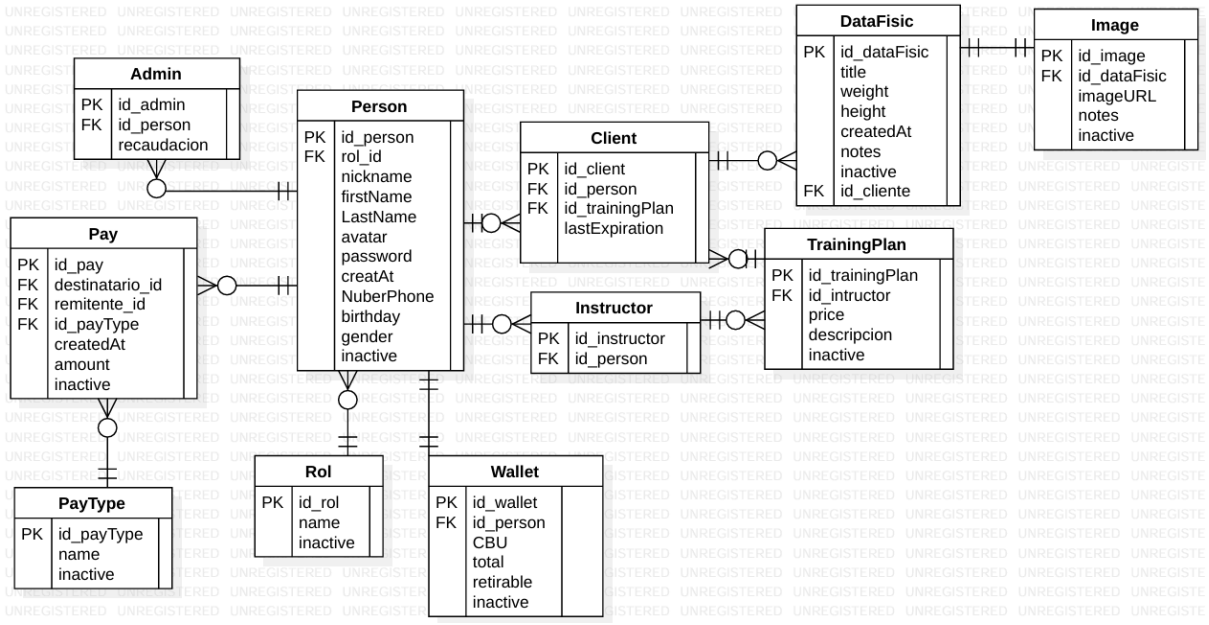

Figura 19

2.10.1 diccionario de datos

Características de la tabla			
Nombre	Person		
Módulo	Person		
Descripción	es la tabla que almacena los datos de las personas		
Características de los datos			
Campo	tipo	longitud	Significado
id_person	int	11	identificación única para las personas
nickname	varchar	7	nombre del usuario dentro del sistema
firstName	varchar	100	nombre de la persona
lastName	varchar	100	apellido de la persona
avatar	varchar	400	url de la imagen del avatar de la persona
password	varchar	500	contraseña de la persona

NumberPhone	varchar	50	numero de telefono de la persona
birthday	datetime	- -	fecha de nacimiento de la persona
gender	char	1	genero de la persona
rol_id	int	11	indica el id del rol de la persona
createdAt	datetime	- -	fecha en la cual fue creada por la persona
inactive	bit	1	guarda si la persona se encuentra activa o inactiva

Restricciones	
Campo	Tipo de Restricción
id_person	PRIMARY KEY

Claves Foráneas	
Campo	Entidad Asociada
rol_id	entidad rol

Tabla 13

Características de la tabla			
Nombre	Client		
Módulo	Client		
Descripción	almacena los datos de los clientes del sistema		
Características de los datos			
Campo	tipo	longitud	Significado
id_client	int	11	identifica única de cada cliente
lastExpiration	datetime	- -	último vencimiento pagado
id_person	int	11	idéntica a la persona
id_trainingPlan	int	11	idéntica al plan al cual está suscrito el cliente

Restricciones	
Campo	Tipo de Restricción
id_client	PRIMARY KEY
Claves Foráneas	
Campo	Entidad Asociada
id_person	entidad Person
id_trainingPlan	entidad TrainingPlan

Tabla 14

Características de la tabla			
Nombre	Admin		
Módulo	Admin		
Descripción	Es la tabla que almacena el admins del sistema		
Características de los datos			
Campo	tipo	longitud	Significado
id_admin	int	11	identificación única de los admins
recaudacion	float	8	la recaudación que genera el sistema
id_person	int	11	identifica a la persona
Restricciones			
Campo			Tipo de Restricción
id_admin			PRIMARY KEY
Claves Foráneas			
Campo			Entidad Asociada
id_person			Entidad Persona

Tabla 15

Características de la tabla

Nombre	Instructor		
Módulo	Instructor		
Descripción	es el instructor que ayuda a los clientes		
Características de los datos			
Campo	tipo	longitud	Significado
id_instructor	int	11	identificador único del instructor
id_person	int	11	identifica la persona
Restricciones			
Campo			Tipo de Restricción
id_instructor			PRIMARY KEY
Claves Foráneas			
Campo			Entidad Asociada
id_person			Entidad Person

Tabla 16

Características de la tabla			
Nombre	TrainingPlan		
Módulo	TrainingPlan		
Descripción	tabla que almacena el plan de entrenamiento del cliente		
Características de los datos			
Campo	tipo	longitud	Significado
id_trainingPlan	int	11	identificador unico de cada plan de entrenamiento
price	float	8	almacena el precio del plan de entrenamiento
description	varchar	500	describe el plan con detalle
inactive	bit	1	almacena la actividad o inactividad del plan
id_instructor	int	11	identifica al instructor a cargo

Restricciones	
Campo	Tipo de Restricción
id_trainingPlan	PRIMARY KEY
Claves Foráneas	
Campo	Entidad Asociada
id_instructor	Entidad Instructor

Tabla 17

Características de la tabla			
Nombre	wallet		
Módulo	wallet		
Descripción	Es la tabla que guarda los datos de los datos relacionados con los pagos que hacen los clientes		
Características de los datos			
Campo	tipo	longitud	Significado
id_wallet	int	11	identificador único de cada cartera
total	float	8	resguarda el valor total del dinero que el usuario tiene en su cartera
retirable	float	8	almacena la cantidad de dinero retirable
inactive	bit	1	almacena si el la billetera esta activa o inactiva
CBU	varchar	20	es el cbu de la billetera
id_person	int	11	identifica a la persona dueña de la billetera
Restricciones			
Campo			Tipo de Restricción
id_wallet			PRIMARY KEY
Claves Foráneas			
Campo			Entidad Asociada

id_person	Entidad Person

Tabla 18

Características de la tabla			
Nombre	Rol		
Módulo	Rol		
Descripción	Es la tabla que almacena los roles del sistema		
Características de los datos			
Campo	tipo	longitud	Significado
id_rol	int	11	identificador unico de cada rol
name	varchar	50	nombre del rol
inactive	bit	1	almacena si el rol esta activo o inactivo
Restricciones			
Campo		Tipo de Restricción	
id_rol		PRIMARY KEY	

Tabla 19

Características de la tabla			
Nombre	Pay		
Módulo	Pay		
Descripción	La tabla que almacena los detalles de pago de los usuarios		
Características de los datos			
Campo	tipo	longitud	Significado
id_pay	int	11	identificador unico de los pagos
createdAt	datetime	--	almacena la fecha en la que fue creado el pago

amount	float	8	almacena la cantidad del pago
inactive	bit	1	almacena si el registro esta activo o inactivo
destinatario_id	int	11	identifica al destinatario del pago
remitente_id	int	11	identifica al remitente del pago
id_payType	int	11	identifica el tipo de pago
Restricciones			
Campo			Tipo de Restricción
id_pay			PRIMARY KEY
Claves Foráneas			
Campo			Entidad Asociada
destinatario_id			Entidad Person
remitente_id			Entidad Person

Tabla 20

Características de la tabla			
Nombre	PayTyime		
Módulo	PayType		
Descripción	Es la tabla que guarda los tipos de pago		
Características de los datos			
Campo	tipo	longitud	Significado
id_payType	int	11	identificador unico que identifica el tipo de pago
name	varchar	500	almacena el nombre del tipo de pago
inactive	bit	1	almacena si el tipo de pago esta activo o inactivo
Restricciones			
Campo			Tipo de Restricción
id_payType			PRIMARY KEY

Tabla 21

Características de la tabla			
Nombre	DataFisic		
Módulo	DataFisic		
Descripción	La tabla que almacena el estado fisico de el cliente		
Características de los datos			
Campo	tipo	longitud	Significado
id_dataFisic	int	11	identificador unico del estado fisico de un cliente
weight	float	8	almacena el peso de el cliente
height	float	8	almacena la altura del cliente
createdAt	datetime	--	almacena la fecha en la que se crea el estado fisico del cliente
notes	varchar	500	almacena algunas notas para el cliente
inactive	bit	1	almacena si es un cliente activo o inactivo
id_client	int	11	identifica al cliente
title	varchar	100	almacena al titulo de los datos fisicos del cliente
Restricciones			
Campo			Tipo de Restricción
id_dataFisic			PRIMARY KEY
Claves Foráneas			
Campo			Entidad Asociada
id_client			Entidad Client

Tabla 22

Características de la tabla	
Nombre	wallet
Módulo	wallet
Descripción	Es la tabla que guarda los datos de los datos relacionados con

	los pagos que hacen los clientes		
Características de los datos			
Campo	tipo	longitud	Significado
id_image	int	11	identificador unico de cada imagen
imageURL	varchar	500	almacena la url de la imagen
notes	varchar	500	almacena notas acerca de la imagen
inactive	bit	1	almacena si la imagen esta activa o inactiva
id_dataFisic	int	11	identifica a los datos fisicos de la persona que subio la imagen
Restricciones			
Campo			Tipo de Restricción
id_image			PRIMARY KEY
Claves Foráneas			
Campo			Entidad Asociada
id_dataFisic			Entidad DataFisic

Tabla 23

3. Herramientas y/o Lenguajes de Programación

Las tecnologías utilizadas durante el desarrollo del proyecto fueron: C#, .NET , .Net Framework, SqlServer como administrador de base de datos, Azana

3.1 Descripción

La aplicación será desarrollada con la tecnología .NET en su versión .Net Framework y sus librerías asociadas, empleando el lenguaje C#, opcionalmente se podrá establecer conexiones con servicios de terceros

Asana: es una herramienta integral de gestión del trabajo diseñada para ayudar a personas y equipos a realizar un seguimiento de las tareas, delegar responsabilidades, monitorear el progreso y comunicarse en tiempo real.

Sql Server: Gestor de bases de datos utilizado en el proyecto.

Discord: Discord es una aplicación de chat gratuita que permite a los usuarios comunicarse en tiempo real mediante texto, voz o vídeo.

StartUML: una herramienta de modelado de software que permite a los desarrolladores crear diagramas UML (Unified Modeling Language) de manera intuitiva y eficiente. Los usuarios pueden diseñar diagramas de clases, diagramas de secuencia, diagramas de actividades, diagramas de casos de uso y otros tipos de diagramas utilizados en el desarrollo de software.

Draw.io: es una herramienta de diagramación en línea que permite a los usuarios crear una amplia variedad de diagramas, incluidos diagramas de flujo, diagramas de red, diagramas de clases, diagramas de secuencia y muchos más. Es una herramienta flexible y fácil de usar que ofrece una amplia gama de formas y símbolos que los usuarios pueden arrastrar y soltar para crear sus diagramas.

ChatGPT: ChatGPT es una tecnología desarrollada por OpenAI que utiliza inteligencia artificial para generar respuestas de texto en conversaciones humanas. ChatGPT se puede utilizar en una variedad de aplicaciones, como asistentes virtuales, chatbots, corrección de texto y generación de texto creativo. Es una herramienta poderosa para la reformulación de texto y la solución de problemas de lenguaje natural.

4. Resultados

Figura 20

Figura 21

5. Referencia

• Apunte de Análisis de Sistemas I. Mgter. Vallejos Oscar. http://exa.unne.edu.ar/informatica/anasistem1/public_html/home.html
• Análisis estructurado Moderno. Yourdon Edward.
• INGENIERÍA DEL SOFTWARE. UN ENFOQUE PRÁCTICO. Quinta edición. Roger S. Pressman
• Apunte de Análisis de Sistemas I. Mgter. Vallejos Oscar. http://exa.unne.edu.ar/informatica/anasistem1/public_html/home.html
• Análisis estructurado Moderno. Yourdon Edward.
• Alejandro Oliveros. Importancia de los Requerimientos, material de Tópicos I , perteneciente a Maestría Ingeniería de Software dictada en UNNE.
• Ingeniería del software 7º Edición. Ian Sommerville. Edición Pearson. Capítulo 6 Norma Estándar IEEE 830.
• Análisis y diseño de sistemas 6ª edición. Kenneth Kendall y Julie Kendall. Editorial earson. Capítulo 4
• Apuntes de la asignatura Ingeniería del Software I. Mgter.Oscar Vallejos.
• Scrum Manager http://www.scrummanager.net
•Metodología Scrum TFC. http://openaccess.uoc.edu/webapps/o2/bitstream/10609/17885/1/mtrigasTFC0612memoria.pdf
• UML y Patrones. Craig Larman.
• MVP: https://www.wildcrest.com/Potel/Portfolio/mvp.pdfd
    - MVP: https://help.windev.com/es-ES/?1000021496