Documento de Arquitectura de Software (DAS)

**Proyecto**

Transform4te

**Arquitectos**

Brahian Stiven Osorio Velasquez

Juan Jose Cardona Piedrahita

Steven Alejandro Ruíz Rendón

# Control de cambios y revisiones

| Versión | Fecha | Tipo | Descripción | Autor |
| --- | --- | --- | --- | --- |
| 1 | 01/10/2023 | Creación | Versión inicial del documento | Eusebio Cuartas |
| 2 | 15/10/2023 | Revisión | Se registran novedades de la revisión | Armando Contreras |
| 3 | 20/10/2023 | Aprobación | Aprobación versión 3 del documento | Camilo García |

Contenido

[Control de cambios y revisiones 2](#_heading=h.gjdgxs)

[1. Propósito del proyecto 7](#_heading=h.30j0zll)

[2. Motivadores de la arquitectura 7](#_heading=h.1fob9te)

[2.1 Restricciones técnicas 7](#_heading=h.3znysh7)

[2.2 Restricciones de negocio 9](#_heading=h.2et92p0)

[2.3 Atributos de calidad 11](#_heading=h.tyjcwt)

[2.3.1 Atributo calidad 1 11](#_heading=h.3dy6vkm)

[2.3.1.1 Característica CAR-SEG-0001 12](#_heading=h.1t3h5sf)

[2.3.1.1.1 Escenario de calidad CAL-CAR-SEG-0001-01 12](#_heading=h.4d34og8)

[2.3.1.1.2 Escenario de calidad ESL-CAL-CAR-SEG-0001-02 13](#_heading=h.2s8eyo1)

[2.3.1.2 Característica 13](#_heading=h.26in1rg)

[2.3.1.2.1 Escenario de calidad ESL-CAL-CAR-SEG-0002-01 14](#_heading=h.lnxbz9)

[2.3.1.2.2 Escenario de calidad ESL-CAL-CAR-SEG-0002-02 14](#_heading=h.35nkun2)

[2.3.1.2.3 Escenario de calidad ESL-CAL-CAR-SEG-0002-03 14](#_heading=h.1ksv4uv)

[2.3.1.2.4 Escenario de calidad ESL-CAL-CAR-SEG-0002-04 15](#_heading=h.57ogbmhy53no)

[2.3.1.2.5 Escenario de calidad ESL-CAL-CAR-SEG-0002-05 15](#_heading=h.nnsmhf6cbzjt)

[2.3.1.2.6 Escenario de calidad ESL-CAL-CAR-SEG-0002-06 16](#_heading=h.swmawxl255w7)

[2.3.2 Atributo calidad 2 Rendimiento 16](#_heading=h.2jxsxqh)

[2.3.2.1 Característica CAR-REN-0004 17](#_heading=h.z337ya)

[2.3.2.1.1 Escenario de calidad 1 ESL-CAL-CAR-REND-0004-01 17](#_heading=h.3j2qqm3)

[2.3.2.1.2 Escenario de calidad 2 17](#_heading=h.1y810tw)

[2.4 Funcionalidades críticas 18](#_heading=h.1pxezwc)

[3. Tácticas y estrategias 22](#_heading=h.49x2ik5)

[4. Modelo de contexto 25](#_heading=h.2p2csry)

[5. Arquetipo de solución/referencia 25](#_heading=h.147n2zr)

[6. Arquitectura de solución/referencia 26](#_heading=h.3o7alnk)

[7. Línea base arquitectónica 27](#_heading=h.23ckvvd)

[7.1 Línea base arquitectónica de componentes 27](#_heading=h.ihv636)

[7.1.1 Componente back 27](#_heading=h.32hioqz)

[7.1.2 Componente Front end 28](#_heading=h.1hmsyys)

[7.2 Estilos y patrones arquitectónicos adoptados 28](#_heading=h.41mghml)

[7.2.1 Estilo arquitectónico 1 28](#_heading=h.2grqrue)

[7.2.1.1 Nombre 28](#_heading=h.vx1227)

[7.2.1.2 Problema 29](#_heading=h.3fwokq0)

[7.2.1.3 Solución/Motivación 29](#_heading=h.1v1yuxt)

[7.2.2 Estilo arquitectónico 2 29](#_heading=h.4f1mdlm)

[7.2.2.1 Nombre 29](#_heading=h.2u6wntf)

[7.2.2.2 Problema 29](#_heading=h.19c6y18)

[7.2.2.3 Solución/Motivación 29](#_heading=h.3tbugp1)

[8. Justificación alternativa de solución 29](#_heading=h.46r0co2)

[8.1 Justificación 29](#_heading=h.2lwamvv)

[8.2 Ventajas 30](#_heading=h.111kx3o)

[8.3 Desventajas 30](#_heading=h.3l18frh)

[9. Vistas de arquitectura del sistema 30](#_heading=h.206ipza)

[9.1 Vista Funcional/Vista de Escenarios/Vista de Casos de Uso 30](#_heading=h.4k668n3)

[9.1.1 Modelo de procesos del negocio 31](#_heading=h.2zbgiuw)

[9.1.2 Modelado de dominio 31](#_heading=h.1egqt2p)

[9.1.3 Modelo de contextos 31](#_heading=h.3ygebqi)

[9.1.3.1 Diagrama 32](#_heading=h.2dlolyb)

[9.1.3.2 Documentación contextos 32](#_heading=h.sqyw64)

[9.1.4 Modelo de mapeo de contextos 33](#_heading=h.3cqmetx)

[9.1.4.1 Diagrama 33](#_heading=h.1rvwp1q)

[9.1.4.2 Documentación mapeo de contextos 34](#_heading=h.4bvk7pj)

[9.1.5 Modelos de dominio 34](#_heading=h.2r0uhxc)

[9.1.5.1 Contexto 1 35](#_heading=h.1664s55)

[9.1.5.2 Modelo anémico 35](#_heading=h.3q5sasy)

[9.1.5.3 Modelo enriquecido 37](#_heading=h.25b2l0r)

[9.1.6 Flujo de eventos/Event Storming 37](#_heading=h.2w5ecyt)

[9.1.6.1 Diagrama 37](#_heading=h.1baon6m)

[9.1.7 Glosario de términos del negocio 38](#_heading=h.2afmg28)

[9.1.8 Especificación de requisitos de software 39](#_heading=h.pkwqa1)

[9.1.8.1 Requisitos de usuario 39](#_heading=h.39kk8xu)

[9.1.8.2 Requisitos del sistema 39](#_heading=h.1opuj5n)

[9.1.8.2.1 Requisitos funcionales 39](#_heading=h.48pi1tg)

[9.1.8.2.2 Requisitos no funcionales 40](#_heading=h.2nusc19)

[9.1.8.2.3 Requisitos de información 40](#_heading=h.1302m92)

[9.1.8.2.4 Reglas de negocio 41](#_heading=h.3mzq4wv)

[9.1.9 Casos de uso 41](#_heading=h.2250f4o)

[9.1.9.1 Modelo de contexto 42](#_heading=h.haapch)

[9.1.9.1.1 Diagrama 42](#_heading=h.319y80a)

[9.1.9.1.2 Descripción 42](#_heading=h.1gf8i83)

[9.1.9.2 Diagramas de casos de uso 43](#_heading=h.40ew0vw)

[9.1.9.2.1 Componente gestión de gimnasios 43](#_heading=h.2fk6b3p)

[9.1.9.2.1.1 Diagrama de casos de uso 43](#_heading=h.upglbi)

[9.1.9.2.1.2 Especificación de casos de uso 44](#_heading=h.3ep43zb)

[9.1.9.2.1.2.1 Caso de uso CREACIÓN DE CLASE 44](#_heading=h.1tuee74)

[9.1.9.2.1.2.1.1 Datos básicos caso de uso 44](#_heading=h.4du1wux)

[9.1.9.2.1.2.1.2 Escenarios del caso de uso 44](#_heading=h.2szc72q)

[9.1.9.2.1.2.1.3 Flujo normal/flujo básico 44](#_heading=h.184mhaj)

[9.1.9.2.1.2.1.4 Flujo alterno 1 45](#_heading=h.3s49zyc)

[9.1.9.2.1.2.1.5 Flujo Excepcional 1 46](#_heading=h.36ei31r)

[9.1.9.2.1.2.1.6 Diagrama de actividades 46](#_heading=h.2koq656)

[9.1.9.2.1.2.1.6.1 Diagrama 47](#_heading=h.zu0gcz)

[9.1.9.2.1.2.1.6.2 Documentación 47](#_heading=h.3jtnz0s)

[9.1.9.2.1.2.1.7 Diagrama de estados 48](#_heading=h.1yyy98l)

[9.1.9.2.1.2.1.7.1 Diagrama 48](#_heading=h.4iylrwe)

[9.1.9.2.1.2.1.7.2 Documentación 48](#_heading=h.2y3w247)

[9.1.10 Incepción Ágil 49](#_heading=h.ly7c1y)

[9.1.11 Por qué estamos aquí 49](#_heading=h.1l354xk)

[9.1.12 Visión/Elevator Pitch 49](#_heading=h.452snld)

[9.1.12.1 Visión 49](#_heading=h.2k82xt6)

[9.1.12.2 Mapa de impacto 50](#_heading=h.3jd0qos)

[9.1.13 Tamaño/Talla de historias de usuario 50](#_heading=h.3b2epr8)

[9.1.13.1 Tallaje del producto 50](#_heading=h.1q7ozz1)

[9.1.13.1.1 Definiciones para el tallaje 50](#_heading=h.4a7cimu)

[9.1.13.1.2 Tallaje del producto 51](#_heading=h.2pcmsun)

[9.1.13.2 Release Plan 51](#_heading=h.14hx32g)

[9.2 Vista Lógica 51](#_heading=h.2h20rx3)

[9.2.1 Diagrama de clases 51](#_heading=h.w7b24w)

[9.2.1.1 Componente 1 52](#_heading=h.3g6yksp)

[9.2.1.1.1 Diagrama 52](#_heading=h.1vc8v0i)

[9.2.1.1.2 Documentación 52](#_heading=h.4fbwdob)

[9.2.2 Diagrama de objetos 52](#_heading=h.1n1mu2y)

[9.2.3 Diagrama de componentes 52](#_heading=h.3yqobt7)

[9.2.3.1 Componente 1 53](#_heading=h.2dvym10)

[9.2.3.1.1 Diagrama Componentes Frontend 53](#_heading=h.t18w8t)

[9.2.3.1.2 Documentación 53](#_heading=h.3d0wewm)

[9.2.3.2 Componente 2 55](#_heading=h.1s66p4f)

[9.2.3.2.1 Diagrama Componentes Backend 55](#_heading=h.4c5u7s8)

[9.2.3.2.2 Documentación 55](#_heading=h.2rb4i01)

[9.2.4 Diagrama de paquetes 57](#_heading=h.kqmvb9)

[9.2.4.1 Componente 1 Front 57](#_heading=h.34qadz2)

[9.2.4.1.1 Diagrama Front 58](#_heading=h.1jvko6v)

[9.2.4.1.2 Documentación 58](#_heading=h.43v86uo)

[9.2.4.2 Componente 2 back 60](#_heading=h.2j0ih2h)

[9.2.4.2.1 Diagrama 60](#_heading=h.y5sraa)

[9.2.4.2.2 Documentación 60](#_heading=h.3i5g9y3)

[9.3 Vista de Procesos 62](#_heading=h.1bkyn9b)

[9.3.1 Diagrama de secuencia 62](#_heading=h.3vkm5x4)

[9.3.1.1 Componente 1 62](#_heading=h.2apwg4x)

[9.3.1.1.1 Diagrama 62](#_heading=h.pv6qcq)

[9.3.1.1.2 Documentación 63](#_heading=h.39uu90j)

[9.3.2 Diagrama de colaboración 64](#_heading=h.hkkpf6)

[9.3.2.1 Componente 1 65](#_heading=h.31k882z)

[9.3.2.1.1 Diagrama 65](#_heading=h.1gpiias)

[9.3.2.1.2 Documentación 65](#_heading=h.40p60yl)

[9.4 Vista Física/Vista de Implantación 66](#_heading=h.18ewhd8)

[9.4.1 Diagrama de despliegue 66](#_heading=h.3sek011)

[9.4.1.1 Diagrama 67](#_heading=h.27jua8u)

# Propósito del proyecto

Para gimnasios, clubes deportivos, centros recreativos y otros establecimientos dedicados a actividades físicas o deportivas en el que los centros deportivos necesitan resolver la complejidad en la gestión de la programación de clases, la gestión de membresías y la facturación; además aprovechar la tendencia al uso de las aplicaciones y plataformas digitales en auge. Transfórmate es un sitio web que su principal beneficio es cambiar los trabajos manuales para ser automáticos, simplificando la gestión del centro deportivo. Esto implica: eficiencia operativa optimizando el tiempo y el esfuerzo invertido a tareas administrativas repetitivas como la programación de clases, gestión de membresías y la facturación, agilidad en la toma de decisiones al proporcionar acceso instantáneo a datos e informes. A diferencia de MindBody, GymMaster, ZenPlanner, Wodify, MyClup, PerfectGym, TeamUp, Clubworx, PlayVen, UpperHand, nuestro productos de software ofrece características respecto a las necesidades de cada centro deportivo y de fitness así mismo un soporte local al centro deportivo y se incluye la capacitación.

# Motivadores de la arquitectura

Razones o factores que influyen en las decisiones arquitectónicas tomadas durante el proceso de diseño y desarrollo del software. Estos motivadores son fundamentales para definir la estructura, componentes y patrones arquitectónicos de un sistema de software.

La priorización de requisitos como lo son las funcionalidades críticas, restricciones técnicas y de negocio, los atributos de calidad con su mapa de empatía y los escenarios de calidad influyen en las decisiones estratégicas del proyecto, como la elección de tecnologías, patrones arquitectónicos y enfoques de desarrollo.

## Restricciones técnicas

Las restricciones técnicas son limitaciones o condiciones relacionadas con las capacidades técnicas y los recursos disponibles que deben ser tenidos en cuenta en el diseño y desarrollo del software o proyecto. Estas restricciones pueden ser: Tecnológicas, Conocimiento Técnico, Limitaciones de rendimiento, Interoperabilidad

| **Tipo** | **Restricción técnica** | **Justificación** |
| --- | --- | --- |
| **Prácticas de diseño** | El desarrollo de la arquitectura debe seguir principios de DDD ( Domain Driven Design) | Permite una mejor mantenibilidad y evolución del código, facilita la implementación de cambios y mejoras en el sistema ya que como es cercano al modelo de dominio es más fácil de entender y modificar |
| **Prácticas de desarrollo** | Se debe realizar una definición de requisitos de requisitos | Debemos de conocer que tenemos que programar y para qué va a servir. Hay que centrarse con el cliente o usuario final de la aplicación para ver a dónde se quiere llegar y cómo lo haremos |
| Se deben dividir los desarrollos en fases o entregables | Nos servirá para ir realizando determinados ajustes y no llegar al final del proyecto y tener que cambiar más de la mitad desarrollado |
| Se deben seguir prácticas de diseño DRY | Nos brinda una mayor legibilidad y mantenibilidad al hacer el código más legible y fácil de entender, esto así mismo permite la reusabilidad que acelera la velocidad en el desarrollo |
| Se debe implementar el principio Orientado a Objetos | Para las modificaciones, mantenimiento, reusabilidad, abstracción, encapsulación y polimorfismo así mismo las sentencia deben mantenerse simples y elaborar una solución sencilla y corta. Cuanto más corta sea menos errores se producirán y más fácil será localizarlos y solventarlos |
| **Patrones de diseño** | Se debe de utilizar la Inversion of Control(IoC) en el desacoplamiento y separación de preocupaciones con sus pruebas | Para especificar respuestas deseadas a sucesos o solicitudes de datos concretos, dejando alguna entidad o arquitectura externa lleve a cabo las operaciones de control que se requieran en el orden necesario y para el conjunto de sucesos que tengan que ocurrir |
| Se debe de utilizar la Inyección de Dependencias (DI) | Para la creación de los objetos de una manera práctica, útil, escalable y con una alta versatilidad del codigo, asi como tambien nos ayuda a separar nuestro código por responsabilidades |
| Se debe utilizar Capas(Layers) de la arquitectura orientada a servicios y microservicios | Para la distribución jerárquica de los roles y responsabilidades proporciona una separación efectiva de las preocupaciones (cada cual se encarga de los que le corresponde) Se puede imaginar como las capas de un pastel donde cada una de ellas está donde les corresponde. Alta cohesión bajo acoplamiento |
| **Prácticas de código limpio** | Se deben de seguir prácticas de código limpio | Para que el código sea legible, mantenible y reutilizable se siguen las prácticas de código limpio permitiendo así una mejor extensión de vida útil de la aplicación |
| Se debe evidenciar la relación entre las diferentes partes del código | La relación entre las diferentes partes del código es claramente visible |
| Se debe de mostrar todo lo relacionado a las tareas o funciones | La tarea de cada clase, función, método y variable es visible a primera vista. Esto también se debe aplicar a los métodos deben ser reducidos y si es posible tienen una sola tarea clara, así mismo los métodos deben ser predecibles, funcionan como se espera y son de acceso público a través de API |
| **Lineamientos** | La base de datos que se tiene que utilizar es una base de datos transaccional postgreSQL | Se debe seguir por que tiene por sentado y definido por el cliente |
| La aplicación debe ser desarrolla en el lenguaje de programación Java | Se debe seguir por que tiene por sentado y definido por el cliente |
| **Prácticas de**  **diseño** | Se deben seguir prácticas de diseño de arquitecturas limpias. | No brinda el soporte para desarrollar un software más flexible y adaptable a cualquier reforma. Al separar la lógica de negocio de la infraestructura técnica, podemos cambiar y reemplazar componentes sin afectar la lógica de negocio |
| Se deben seguir prácticas de diseño KISS. | Este principio nos facilita un sistema desacoplado de los sistemas que depende, dando una mayor facilidad a modificarlo. Esto se basa en que si se diseña y desarrolla de manera simple |
| Principio SOLID | Nos permitirá crear software de manera más ordenada y fácil de mantener. Esto también ayudará a crecer en el código de manera eficiente y limpia |

## Restricciones de negocio

Son limitaciones o condiciones impuestas por consideraciones comerciales, económicas y financieras que deben ser tenidas en cuenta en la toma de decisiones de diseño y desarrollo del proyecto. Estas restricciones están relacionadas con aspectos empresariales y financieros del proyecto y pueden influir en diversas etapas del ciclo de vida del mismo.

Como restricciones del proyecto presentados en el presupuesto disponible y los plazos de entrega que se deben asegurar afectando así una planificación realista y se ajuste a los recursos disponibles, así como también se deben desarrollar estrategias para mitigar los riesgos y que el proyecto siga siendo viable.

Se deben definir qué funciones son prioritarias y factibles dentro de las restricciones de negocio, en la elección de tecnologías y plataformas a usar se debe tener en cuenta el presupuesto limitado. En el cumplimento de normativas fiscales y legales pueden influir en la arquitectura y el diseño para asegurarse de que el proyecto cumpla con las leyes y regulaciones legales

| **Tipo** | **Restricción de negocio** | **Justificación** |
| --- | --- | --- |
| **Humana** | Restricciones de edad | Algunas clases o actividades pueden tener restricciones de edad. El programa debe asegurarse de los usuarios cumplan estos requisitos antes de permitir reservas |
| **Financiera** | Requisitos de pago | Las politicas de pago, com los metodos de pago aceptados, las tarifas y los plazos de pago, debe ser seguidos por el programa |
| inicio del sistema | Debido a que apenas se comienza es necesario utilizar recursos de open source, así como aprovechar el crédito que se nos proporciona para poner el sistema en línea y de prueba gratuita |
| **Tiempo** | El proyecto debe estar disponible debe mostrar algunas funcionalidades para el 20 de noviembre | A fin de mostrar el progreso y el desarrollo del programa, este no solo debe cumplir con la documentación sino también mostrar que puede ser usado |
| Horarios de Operación | Los centros deportivos pueden tener horarios específicos de apertura y cierre que deben reflejarse en el programa. Esto podría incluir horas pico, días no laborales y horarios especiales |
| **Proceso** | A la fecha existen definiciones al nivel de proceso lo que no permite avanzar bien | Debido a que todavía se necesita realizar pruebas y feedback constante así como algunos documentos todavía se requiere mas revision |
| Seguridad de datos | Se deben aplicar medidas de seguridad para proteger la información confidencial y financiera de los usuarios, incluidos los métodos de autenticación y cifrado |
| Compatibilidad de dispositivos | El programa debe ser compatible con una variedad de dispositivos y navegadores web para garantizar que los miembros puedan acceder desde diferentes dispositivos |
| **Ley** | Un sistema de nómina debe de pagar la parte de la pensión y otros | La nómina debe de ser pagada de acuerdo con la ley de trabajadores de tiempo completo, en caso de que puede generar grandes problemas |
| Cumplimiento de privacidad | El programa debe cumplir con regulaciones de privacidad y protección de datos, como el cumplimiento con leyes como el Reglamento General de Protección de Datos (GDPR) en Europa. |
| Derechos de propiedad intelectual | Asegurarse de no infringir derechos de autor o propiedad intelectual al utilizar imágenes, logos o contenidos relacionados con marcas registradas |
| **Negocio** | Politicas de cancelacion | Permitir a cancelación a tiempo de reservas por parte del miembro pero así mismo estos deben ser realizados con tiempo o se aplicarán penalizaciones |
| Capacidad máxima | Cada clase o actividad podría tener una capacidad máxima de participantes. Esto debe considerarse para evitar la sobreventa o la congestión de espacios |
| Políticas de acceso | Definir quienes pueden acceder a las instalaciones y cómo se gestiona de acceso |

## Atributos de calidad

Son características o propiedades que se utilizan para evaluar y medir la calidad de un sistema de software. Estos atributos son fundamentales ya que determinan que tan bien cumple el software con las necesidades y expectativas de los usuarios así como las partes interesadas

## Atributo calidad 1

Evaluar la capacidad del software para proteger los datos y las operaciones de posibles amenazas o ataques. Incluye medidas de autenticación, autorización y cifrado.

Se debe realizar una evaluación de seguridad para identificar posibles amenazas y vulnerabilidades que el sistema pueda enfrentar. Esto implica considerar cómo los atacantes podrían explotar debilidades en el diseño para acceder, modificar o robar datos o interrumpir el funcionamiento del sistema, la especificación de controles de seguridad necesarios, como la autenticación, la autorización, el cifrado y la auditoría.

Estos requisitos guían el diseño del sistema para incorporar medidas de seguridad desde el principio. La arquitectura del sistema debe ser diseñada teniendo en cuenta principios de seguridad. Esto puede incluir la segmentación de componentes, la protección de la capa de datos y la implementación de cortafuegos y sistemas de detección de intrusiones, se deben establecer políticas de seguridad que definan cómo se gestionan los riesgos y las amenazas a lo largo del ciclo de vida del proyecto. Esto puede incluir políticas de acceso, políticas de gestión de claves y procedimientos de respuesta a incidentes.

Se deben seguir prácticas seguras de codificación para evitar vulnerabilidades comunes, como la inyección de SQL, las vulnerabilidades de XSS (cross-site scripting) y otros ataques conocidos. Esto implica la validación de entrada, el escapado de datos y el uso de bibliotecas de seguridad, se hace uso de pruebas de seguridad, como las pruebas de penetración, deben llevarse a cabo para identificar y solucionar posibles debilidades y vulnerabilidades en el software antes de su lanzamiento.

La gestión segura de claves y certificados es esencial para proteger los datos confidenciales y garantizar la integridad y autenticidad de las comunicaciones.

Se deben tener políticas y procedimientos claros para aplicar actualizaciones y parches de seguridad en el software en caso de que se descubran vulnerabilidades después del lanzamiento.

La auditoría y la monitorización son prácticas continuas para detectar y responder a eventos de seguridad. Esto implica el seguimiento de los registros de actividad del sistema y la implementación de alertas y medidas de respuesta.

## Característica CAR-SEG-0001

| **Identificador** | **Afirmación** | **Pregunta** | **Taxonomía** |
| --- | --- | --- | --- |
| CAR-SEG-0001 | Cuando el usuario no se acuerde de la contraseña tiene una opción que le permita recuperar la contraseña | ¿Quiere tener una recuperación de contraseña? | Autenticación |

## Escenario de calidad CAL-CAR-SEG-0001-01

| **Identificador** | ESL-CAL-CAR-SEG-0001-01 |
| --- | --- |
| **Característica** | CAR-SEG-0001 |
| **Taxonomía** | Autenticación |
| **Atributo de calidad** | Seguridad |
| **Tipo de escenario** | Exitoso |
| **¿Escenario cumplido?** | NO |
| **Descripción** | Recibir por correo electrónico un enlace para resetear/cambiar la contraseña de un usuario. |
| **Origen del estímulo** | Cualquier usuario final |
| **Estímulo** | Ingresar el nombre de usuario y ejecutar la acción de recordar clave |
| **Artefacto** | Sistema |
| **Ambiente** | Operación normal |
| **Respuesta** | El sistema le envía un enlace de cambio de contraseña al correo del usuario |
| **Medida de la respuesta** | Comprobar que al usuario que pidió el restablecimiento le haya llegado un enlace para que este pueda restablecer su contraseña y acceder al sistema |

## Escenario de calidad ESL-CAL-CAR-SEG-0001-02

| **Identificador** | ESL-CAL-CAR-SEG-0001-02 |
| --- | --- |
| **Característica** | CAR-SEG-0001 |
| **Taxonomía** | Autenticación |
| **Atributo de calidad** | Seguridad |
| **Tipo de escenario** | Exitoso |
| **¿Escenario cumplido?** | No |
| **Descripción** | Recibir por medio de SMS un código para resetear/ cambiar la contraseña de un usuario |
| **Origen del estímulo** | Cualquier usuario final |
| **Estímulo** | Ingresar el nombre y usuario ejecutar la acción de recordar la clave |
| **Artefacto** | Sistema |
| **Ambiente** | Operación normal |
| **Respuesta** | El sistema le enviara un código de cambio de contraseña al sms del usuario |
| **Medida de la respuesta** | Comprobar que al usuario que pidió el restablecimiento le haya llegado un código al sms para que este pueda restablecer su contraseña y acceder al sistema |

## Característica

| **Identificador** | **Afirmación** | **Pregunta** | **Taxonomía** |
| --- | --- | --- | --- |
| CAR-SEG-0002 | Para agilizar el proceso de ingreso al sistema el usuario prefiere ingresar con la autenticación por medio de terceros como son (facebook, gmail, linkedin, microsoft) | ¿Quiere una autenticación por terceros (¿Facebook, Gmail, Microsoft, LinkedIn?)? | Autenticación |

## Escenario de calidad ESL-CAL-CAR-SEG-0002-01

| **Identificador** | ESL-CAL-CAR-SEG-0002-01 |
| --- | --- |
| **Característica** | CAR-SEG-0002 |
| **Taxonomía** | Autenticación |
| **Atributo de calidad** | Seguridad |
| **Tipo de escenario** | Exitoso |
| **¿Escenario cumplido?** | NO |
| **Descripción** | el usuario ingresara con su cuenta de (google, facebook, linkedin, microsoft ) |
| **Origen del estímulo** | Cualquier usuario final |
| **Estímulo** | ingresar con(google, facebook, linkedin, microsoft ) |
| **Artefacto** | Sistema |
| **Ambiente** | Operación normal |
| **Respuesta** | el usuario ingresa al sistema de forma exitosa |
| **Medida de la respuesta** | el usuario hace uso de las funcionalidades del sistema |

## Escenario de calidad ESL-CAL-CAR-SEG-0002-02

| **Identificador** | ESL-CAL-CAR-SEG-0002-02 |
| --- | --- |
| **Característica** | CAR-SEG-0002 |
| **Taxonomía** | Autenticación |
| **Atributo de calidad** | Seguridad |
| **Tipo de escenario** | no exitoso |
| **¿Escenario cumplido?** | NO |
| **Descripción** | el usuario intenta ingresar con su cuenta de (google, facebook, linkedin, microsoft ) incorrecta |
| **Origen del estímulo** | Cualquier usuario final |
| **Estímulo** | ingresar con(google, facebook, linkedin, microsoft ) |
| **Artefacto** | Sistema |
| **Ambiente** | Operación normal |
| **Respuesta** | el sistema no reconoce al usuario y rechazará el ingreso al sistema |
| **Medida de la respuesta** | el usuario es redireccionado al inicio de sesión para ingresar al sistema |

## Escenario de calidad ESL-CAL-CAR-SEG-0002-03

| **Identificador** | ESL-CAL-CAR-SEG-0002-03 |
| --- | --- |
| **Característica** | CAR-SEG-0002 |
| **Taxonomía** | Autenticación |
| **Atributo de calidad** | Seguridad |
| **Tipo de escenario** | no exitoso |
| **¿Escenario cumplido?** | NO |
| **Descripción** | el usuario intenta ingresar con su cuenta de google correcta pero no hay conexión al servicio de google |
| **Origen del estímulo** | Cualquier usuario final |
| **Estímulo** | ingresar con google |
| **Artefacto** | Sistema |
| **Ambiente** | Operación normal |
| **Respuesta** | el sistema no reconoce al usuario y rechazará el ingreso al sistema |
| **Medida de la respuesta** | el usuario es redireccionado al inicio de sesión para ingresar al sistema |

## Escenario de calidad ESL-CAL-CAR-SEG-0002-04

| **Identificador** | ESL-CAL-CAR-SEG-0002-04 |
| --- | --- |
| **Característica** | CAR-SEG-0002 |
| **Taxonomía** | Autenticación |
| **Atributo de calidad** | Seguridad |
| **Tipo de escenario** | no exitoso |
| **¿Escenario cumplido?** | NO |
| **Descripción** | el usuario intenta ingresar con su cuenta de facebook correcta pero no hay conexión al servicio de facebook |
| **Origen del estímulo** | Cualquier usuario final |
| **Estímulo** | ingresar con facebook |
| **Artefacto** | Sistema |
| **Ambiente** | Operación normal |
| **Respuesta** | el sistema no reconoce al usuario y rechazará el ingreso al sistema |
| **Medida de la respuesta** | el usuario es redireccionado al inicio de sesión para ingresar al sistema |

## Escenario de calidad ESL-CAL-CAR-SEG-0002-05

| **Identificador** | ESL-CAL-CAR-SEG-0002-05 |
| --- | --- |
| **Característica** | CAR-SEG-0002 |
| **Taxonomía** | Autenticación |
| **Atributo de calidad** | Seguridad |
| **Tipo de escenario** | no exitoso |
| **¿Escenario cumplido?** | NO |
| **Descripción** | el usuario intenta ingresar con su cuenta de microsoft correcta pero no hay conexión al servicio de microsoft |
| **Origen del estímulo** | Cualquier usuario final |
| **Estímulo** | ingresar con microsoft |
| **Artefacto** | Sistema |
| **Ambiente** | Operación normal |
| **Respuesta** | el sistema no reconoce al usuario y rechazará el ingreso al sistema |
| **Medida de la respuesta** | el usuario es redireccionado al inicio de sesión para ingresar al sistema |

## Escenario de calidad ESL-CAL-CAR-SEG-0002-06

| **Identificador** | ESL-CAL-CAR-SEG-0002-06 |
| --- | --- |
| **Característica** | CAR-SEG-0002 |
| **Taxonomía** | Autenticación |
| **Atributo de calidad** | Seguridad |
| **Tipo de escenario** | no exitoso |
| **¿Escenario cumplido?** | NO |
| **Descripción** | el usuario intenta ingresar con su cuenta de linkedin correcta pero no hay conexión al servicio de linkedin |
| **Origen del estímulo** | Cualquier usuario final |
| **Estímulo** | ingresar con linkedin |
| **Artefacto** | Sistema |
| **Ambiente** | Operación normal |
| **Respuesta** | el sistema no reconoce al usuario y rechazará el ingreso al sistema |
| **Medida de la respuesta** | el usuario es redireccionado al inicio de sesión para ingresar al sistema |

## Atributo calidad 2 Rendimiento

Capacidad de evaluar la velocidad y capacidad de respuesta del software en situaciones de carga y bajo diferentes condiciones

El diseño debe tener en cuenta la escalabilidad, lo que significa que el sistema puede crecer para manejar un mayor número de usuarios o un mayor volumen de datos sin perder rendimiento. Esto implica la implementación de arquitecturas distribuidas o el uso de sistemas de almacenamiento y bases de datos escalables.

Se deben considerar estrategias de optimización para consultas de base de datos, algoritmos y estructuras de datos. La elección de algoritmos eficientes y la minimización de consultas innecesarias son aspectos importantes

## Característica CAR-REN-0004

| **Identificador** | **Afirmación** | **Pregunta** | **Taxonomía** |
| --- | --- | --- | --- |
| CAR-REN-0004 | En el momento en que el usuario cargue la lista de disponibilidad de clases y esta se demore más de 50 seg el sistema notificará al usuario sobre las dificultades que están ocurriendo | ¿Quieres que el sistema le notifique al usuario, el problema que está ocurriendo cuando el usuario carga la lista de disponibilidad de clases y se está demorando más de 50 seg? | Tiempo De Respuesta |

## Escenario de calidad 1 ESL-CAL-CAR-REND-0004-01

| **Identificador** | ESL-CAL-CAR-REND-0004-01 |
| --- | --- |
| **Característica** | CAR-REND-0004 |
| **Taxonomía** | Tiempo de respuesta |
| **Atributo de calidad** | Seguridad |
| **Tipo de escenario** | Exitoso |
| **¿Escenario cumplido?** | NO |
| **Descripción** | El usuario recibe una notificación que ocurrio un problema inesperado cuando cargue la disponibilidad de las clases |
| **Origen del estímulo** | Cualquier usuario final |
| **Estímulo** | El sistema presenta interrupciones |
| **Artefacto** | Sistema |
| **Ambiente** | Operación normal |
| **Respuesta** | El sistema le envía una notificación al usuario informando sobre los errores que están ocurriendo |
| **Medida de la respuesta** | el usuario se informa sobre lo que está pasando con su solicitud de disponibilidad de clases |

## Escenario de calidad 2

| **Identificador** | ESL-CAL-CAR-REND-0004-02 |
| --- | --- |
| **Característica** | CAR-RED-0004 |
| **Taxonomía** | Autenticación |
| **Atributo de calidad** | Seguridad |
| **Tipo de escenario** | no Exitoso |
| **¿Escenario cumplido?** | NO |
| **Descripción** | El sistema no identifica el o los errores que están ocurriendo cuando el usuario carga la disponibilidad de clases |
| **Origen del estímulo** | Cualquier usuario final |
| **Estímulo** | El sistema presenta interrupciones |
| **Artefacto** | Sistema |
| **Ambiente** | Operación normal |
| **Respuesta** | El sistema le envía una notificación al usuario indicando su incapacidad de encontrar el problema |
| **Medida de la respuesta** | el usuario cancela o recarga la solicitud de disponibilidad de clases |

## Funcionalidades críticas

Las funcionalidades críticas en el software del centro deportivo son aquellas que son esenciales para la seguridad y el funcionamiento efectivo de un sistema y cuyo mal funcionamiento puede tener consecuencias graves. por eso mismo es necesario plantearlas de una manera muy adecuada, ya que como se mencionó anteriormente son muy importantes para el funcionamiento del sistema y que todo vaya según lo acordado con el cliente y este quede satisfecho. **TOCA SEGUIR**

|  | **Requisito** | **Justificación** |
| --- | --- | --- |
| **RF-0001** | El sistema le debe permitir al usuario del centro deportivo poder cambiar su membresía en cualquier momento | Para cumplir con la expectativa que hemos puesto en la reducción de tiempo manejando la administración por parte del programa son una de las funcionalidades indispensables que debe tener el programa para el usuario final |
| **RF-0002** | El sistema debe permitir al centro deportivo registrar nuevas clases | Para cumplir con la personalización y uso del personal del centro deportivo se debe permitir el uso por parte del centro deportivo y su personal |
| **RF-0003** | El sistema le debe permitir al centro deportivo que este registre nuevas actividades | Para cumplir con la personalización y uso del personal del centro deportivo se debe permitir el uso por parte del centro deportivo y su personal |
| **RF-0004** | El sistema le permitirá al centro deportivo registrar nuevos eventos | Para cumplir con la personalización y uso del personal del centro deportivo se debe permitir el uso por parte del centro deportivo y su personal |
| **RF-0005** | El sistema debe facilitar la realización de pagos por membresías, clases y servicios a los miembros. | Para el cliente es importante que los miembros o usuarios del centro deportivo paguen de manera más fácil usando el sistema de pasarela de pago, así mismo cumpliendo con la gestión de membresías y la reducción del uso del papel |
| **RF-0006** | El sistema deberá generar facturas de todos los pagos realizados en el centro deportivo | Para el cliente es importante que los miembros o usuarios del centro deportivo paguen de manera más fácil usando el sistema de pasarela de pago, así mismo cumpliendo con la gestión de membresías y la reducción del uso del papel |
| **RF-0007** | El sistema debe permitir al centro deportivo rastrear los pagos de todas las personas que estén en este | Para el cliente es importante que los miembros o usuarios del centro deportivo paguen de manera más fácil usando el sistema de pasarela de pago, así mismo cumpliendo con la gestión de membresías y la reducción del uso del papel |
| **RF-0008** | El sistema debe gestionar el acceso a las instalaciones del centro deportivo, ya sea por tarjetas de acceso, códigos QR o mediante la huella de la persona | Debido a que el cliente opera con membresías, así como el miembro paga para utilizar los servicios prestados por el centro deportivo se debe tener un acceso al centro deportivo, evitando así que los que no son miembros ingresen al gimnasio |
| **RF-0009** | El sistema le debe de permitir al centro deportivo enviar notificaciones de manera inmediata a los miembros sobre cambios de horarios, eventos, ofertas especiales u otras actualizaciones importantes | Al momento de que sea un evento, un cambio en las clases como el horario o otras actividades que se realicen en el gimnasio es importante dar a conocer esta información tanto para el gimnasio como para el miembro |
| **RF-0010** | El sistema le debe permitir al miembro del centro deportivo ver su historial en cualquiera momento | Al tener reservas y guardar los datos, el sistema deberá manejar un historial de las acciones realizadas, permitiendo ver a la administración del centro deportivo la información de los miembros sobre asistencias y reservas como también el sistema debe permitir ver al miembro ver información sobre sí mismo |
| **RF-0011** | El sistema le deberá permitir al usuario del centro deportivo poder ver las próximas reservas disponibles | Al tener reservas y guardar los datos, el sistema deberá manejar un historial de las acciones realizadas, permitiendo ver a la administración del centro deportivo la información de los miembros sobre asistencias y reservas como también el sistema debe permitir ver al miembro ver información sobre sí mismo |
| **RF-0012** | El sistema le permitirá al usuario del centro deportivo poder ver toda la información de su membresía | Al tener reservas y guardar los datos, el sistema deberá manejar un historial de las acciones realizadas, permitiendo ver a la administración del centro deportivo la información de los miembros sobre asistencias y reservas como también el sistema debe permitir ver al miembro ver información sobre sí mismo |
| **RF-0013** | El sistema debe ser capaz de ofrecer informes generados a partir de datos recopilados | Como una de las necesidades en administración y cumpliendo con el hecho de evitar el uso de más papel asi como tambien una gestión en la administración, y que es una función que debe tener el sistema |
| **RF-0014** | El sistema permitirá a los centros deportivos tener unas estadísticas de asistencias de usuarios | Como una de las necesidades en administración y cumpliendo con el hecho de evitar el uso de más papel asi como tambien una gestión en la administración, y que es una función que debe tener el sistema |
| **RF-0015** | El sistema le permitirá al centro deportivo obtener un análisis de ingresos | Como una de las necesidades en administración y cumpliendo con el hecho de evitar el uso de más papel asi como tambien una gestión en la administración, y que es una función que debe tener el sistema |
| **RF-0016** | El sistema le permitirá a los centros deportivos administrar cuentas de usuarios | Para cumplir con el nivel de acceso, permisos, derechos y deberes que deben de cumplir así mismo esto permite diferenciar entre los entrenadores, miembros o gerentes evitando así posibles problemas |
| **RF-0017** | El sistema permitirá a los centros deportivos asignar roles y permisos dentro de este mismo | Para cumplir con el nivel de acceso, permisos, derechos y deberes que deben de cumplir así mismo esto permite diferenciar entre los entrenadores, miembros o gerentes evitando así posibles problemas |
| **RF-0018** | El sistema permitirá tener una gestión de la información de los usuarios de los centros deportivos | Para cumplir con el nivel de acceso, permisos, derechos y deberes que deben de cumplir así mismo esto permite diferenciar entre los entrenadores, miembros o gerentes evitando así posibles problemas |
| **RF-0019** | El sistema permitirá proporcionar un forma de personalización o configuración del centro deportivo | El cliente debe de tener la capacidad para operar y personalizar su sistema de acuerdo a sus necesidades o ofertas y capacidades permitiendo así una menor intervención del equipo de desarrollo a la gestión del centro deportivo como también permitir una personalización por parte del cliente |
| **RF-0020** | El sistema le permitirá al centro deportivo que tipos de clases ofrecerán a los usuarios | El cliente debe de tener la capacidad para operar y personalizar su sistema de acuerdo a sus necesidades o ofertas y capacidades permitiendo así una menor intervención del equipo de desarrollo a la gestión del centro deportivo como también permitir una personalización por parte del cliente |
| **RF-0021** | El sistema le permitirá al centro deportivo configurar las tarifas de las membresías o planes | El cliente debe de tener la capacidad para operar y personalizar su sistema de acuerdo a sus necesidades o ofertas y capacidades permitiendo así una menor intervención del equipo de desarrollo a la gestión del centro deportivo como también permitir una personalización por parte del cliente |
| **RF-0022** | El sistema le debe permitir al centro deportivo poder ajustar las preferencias sobre las clases | El cliente debe de tener la capacidad para operar y personalizar su sistema de acuerdo a sus necesidades o ofertas y capacidades permitiendo así una menor intervención del equipo de desarrollo a la gestión del centro deportivo como también permitir una personalización por parte del cliente |

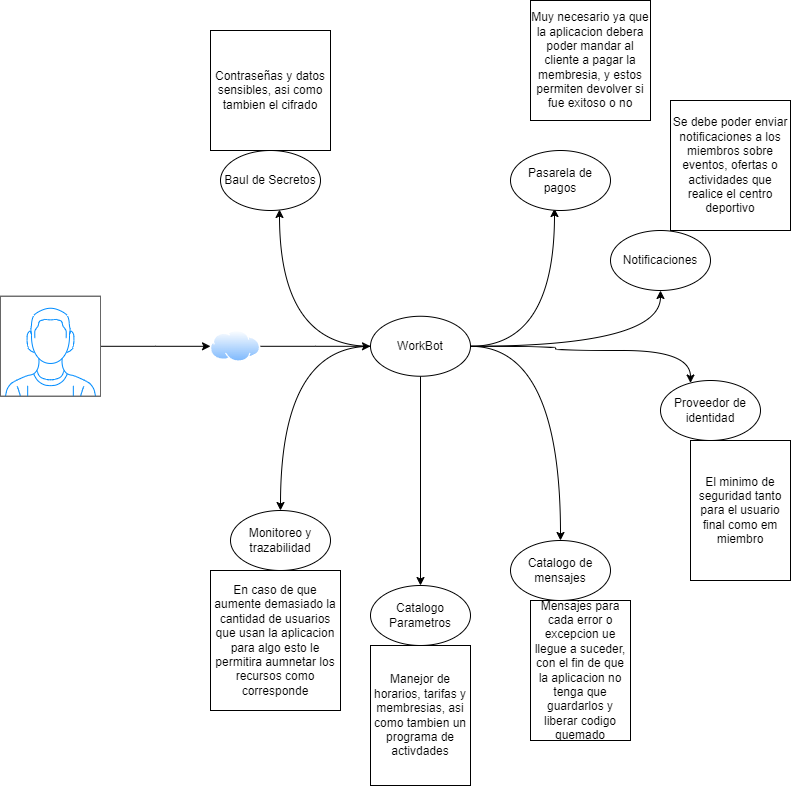
# Tácticas y estrategias

En pocas palabras las tácticas son las actividades que queremos que tenga nuestro proyecto del centro deportivo y las estrategias serían el cómo vamos abarcar las tácticas en un periodo definido, ya pueda ser a largo plazo o a corto plazo. De este modo obtendremos buenos resultados ya que al tener una planeación de esa forma todo va quedando más documentado y más ordenado.

| **Táctica** | **Priorización** | **Justificación** |
| --- | --- | --- |
| Se pretende realizar una implementación de recuperación de contraseña por medio de correo electrónico. | 3 | Porque el miembro como tal necesita en caso de que olvide su contraseña una forma de recuperarla permitiendo el uso de la cuenta y no tener que crear otra y volviendo a usarla desde el inicio |
| Se busca implementar un método por autenticación llamado (OAuth 2.0). El cual tendrá como objetivo principal permitir a los usuarios autenticarse en tu plataforma utilizando sus cuentas de terceros, como Facebook, Gmail, Microsoft o LinkedIn, en lugar de crear y recordar una nueva contraseña para tu servicio. | 2 | Lo que buscamos con El OAuth 2.0 es tener un ingreso a nuestras cuentas de forma segura, flexible y estándar de este modo estaremos buscando una autenticacion y autorizacion en las aplicaciones de nuestro sistema |
| Como medida de seguridad buscamos implementar alertas de Intrusión Fallida con el fin de notificar a los usuarios o administradores cuando alguien intenta acceder repetidamente de manera incorrecta a una cuenta o sistema. | 1 | Estas alertas serán importantes para nuestro sistema ya que mantendrá una seguridad en estos mismos, además de esto nos permitirá proporcionar una respuesta rápida a posibles amenazas y también nos permitirá a cumplir con todas las regulaciones de seguridad que va tener nuestro sistema |
| Se pretende implementar auditorías de Rendimiento Periódicas con el fin de evaluar el funcionamiento del sistema y detectar problemas de manera programada. | 10 | El tema de implementación de auditorías de rendimiento es muy importante debido a que vamos a tener un sistema que siempre va estar en constante cambio como en el tema de las clases, de igual forma permite que el sistema sea seguro y confiable |
| Se pretende implementar un manual de instrucciones para los usuarios que ingresen por primera vez de este modo los permita configurar la plataforma tal y como ellos quieran | 4 | El tema de manual de instrucciones en nuestro sistema es muy importante, por que lo que buscamos es que el usuario se sienta cómodo al ingresar a este mismo, buscando que este entienda todo y se adapte rápidamente al centro deportivo |
| Se pretende elaborar un informe personal a cada usuario del centro deportivo cuyo contenido contiene como objetivo mostrar el progreso que lleva el usuario con respectó a sus metas u objetivos para ello se implementará una librería especializada en reportes | 5 | Lo que buscamos con los informes personalizados es que el usuario tenga un seguimiento de llo que este ha hecho en las clases o a lo largo de su estadía, de este modo permite que este tenga un Feuerbach constante y se de cuenta si está mejorando o empeorando |
| Se pretende adaptar las normas WCAG (Web Content Accessibility Guidelines) para lograr hacer más cómoda la instancia al usuario discapacitado | 6 | Lo que buscamos con esta implementación es que todos los usuarios se sientan por igual y no excluir a nadie. De este modo estamos trabajando hacia un ambiente digital donde cada persona, independientemente de sus habilidades o discapacidades, pueda acceder y utilizar la plataforma de manera equitativa |
| Se va a observar y a trabajar de la mano con los usuarios con discapacidad para diseñar dichas ilustraciones, con el objetivo de que para ellos sea más precisa | 7 | Se implementa debido a que son esenciales para poder proporcionar un acceso a la información de una manera muy eficaz para estas personas, de igual forma lo que buscamos es apoyar la comunicación de dichas personas con el sistema y se sientan cómodas usándolo |
| El sistema implementará pruebas preprogramadas a una hora específica cuyo contenido se ejecutará a la par con las horas más cargadas de usuarios | 13 | Es muy importante para nuestro sistema ya que garantiza una estabilidad, confiabilidad y rendimiento del sistema y de este modo buscamos cumplir todas las expectativas del usuario en cada momento |
| Se generará un pequeño reporte con los resultados de las pruebas realizadas al administrador | 12 | Se busca tener un reporte de los hechos, de este modo el administrador del sistema identificará las fallas en el sistema y las corregirá, y a sí mismo le permitimos una buena estadía la usuario |
| Se va a tener en lista todas las programaciones del día y dicha lista tiene que estarse actualizando muy a menudo | 8 | Es muy importante para nosotros y para el usuario que la lista de programaciones de cada día se mantenga en constante actualización, si no fuera de este método cargaremos clases de días anteriores y causaría problemas en los usuarios. por eso es importante que esta se mantenga actualizada constantemente con las clases o eventos publicados en el centro deportivo |
| Cuando ocurra algún cambio por parte del administrador la lista se actualiza en todos los dispositivos de los usuarios | 9 | Es muy importante tener una lista actualizada de este modo evitamos confusiones en los usuarios |
| Se va a determinar una característica la cual le permite al sistema generar una notificación cuando se esté llegando al límite de los recursos | 11 | Es muy importante que el sistema informe de si este está llegando a su capacidad, de este modo se buscará la mejor forma para una optimización de recursos y este siga su funcionamiento de una manera adecuada |
|

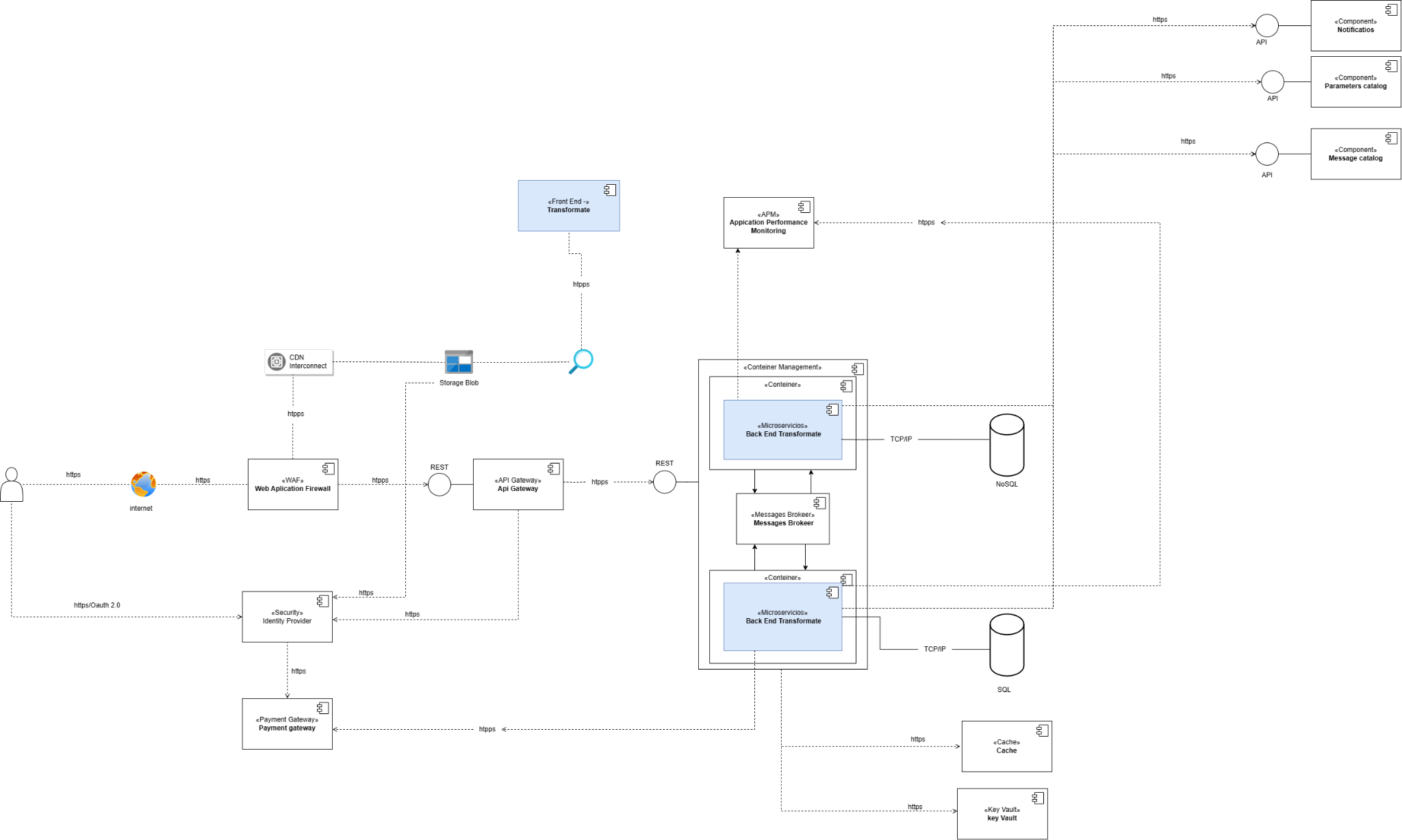
# Modelo de contexto

Representación visual que describe los elementos relevantes para la aplicación. Este modelo ayuda a comprender el entorno en el que la aplicación usa otros sistemas para la comunicación con el usuario con servicios externos, proporcionando una visión general de las interfaces del sistema ayuda a diseñar una solución que se ajuste a la aplicación



# Arquetipo de solución/referencia

Modelo de diseño que sirve como guía y plantilla para arquetipos y formas de cómo responderá ante el ingreso del usuario y diseñará el los diferentes componentes que va a utilizar la solución para la seguridad, recibido de múltiples entradas de usuario y como se representara para abordar el problema

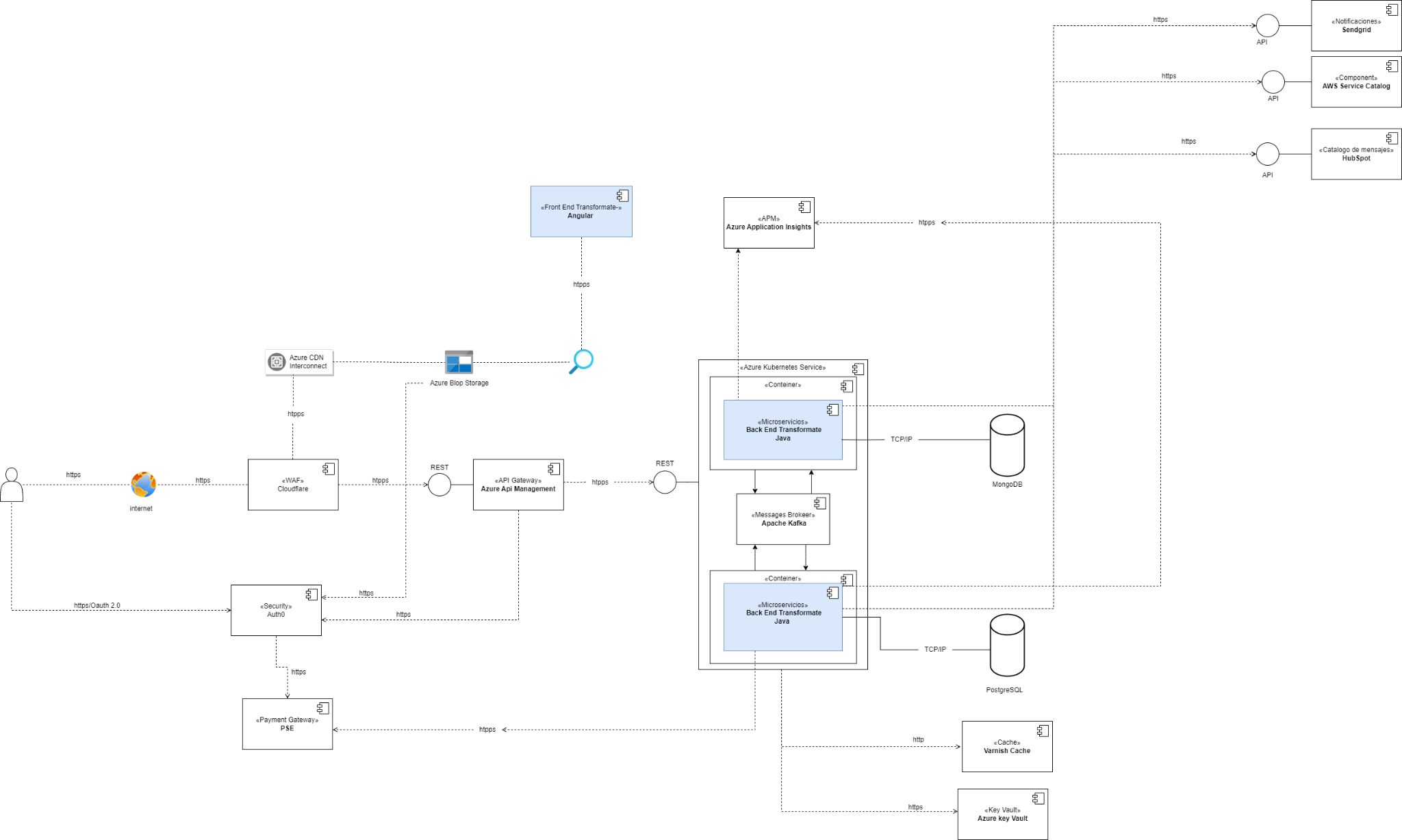


# Arquitectura de solución/referencia

Proporciona una estructura y guía para el desarrollo de la aplicación, estableciendo las principales decisiones de diseño, componentes y las relaciones con su forma de comunicación entre estas. Es una estandarización y forma de abordar la reutilización y el diseño del de la arquitectura de software

WAF Cloudflare: Recurso gratuito que solo se paga a partir de cierto cantidad de usos, muy recomendado en aplicaciones web y que permite una capa de seguridad manejada por la propia empresa CloudFlare

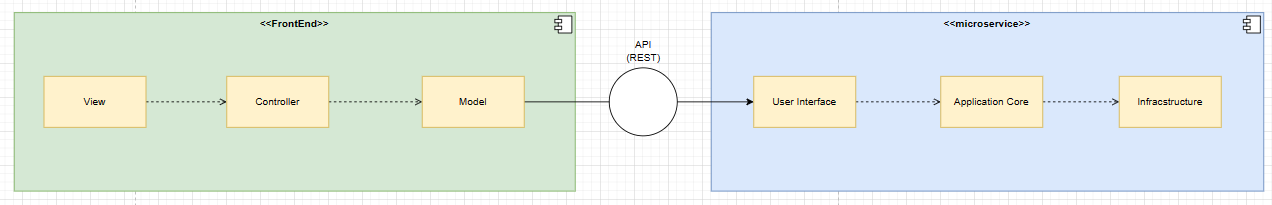
CDN Azure CDN Recurso gratuito que solo se paga a partir de cierto cantidad de usos, muy recomendado en aplicaciones web y que permite una capa de seguridad manejada por la propia empresa Microsoft, además también tiene la ventaja de que al pertenecer a azure hace que la integración con otros componente Azure sea más fluida



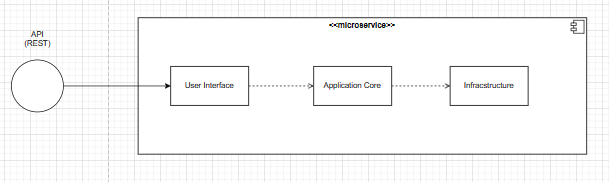
# Línea base arquitectónica

Una línea base arquitectónica es una vista desde un nivel alto sobre el cómo se construirá el sistema antes de comenzar a desarrollar, este se compone de decisiones y componentes que componen los cimientos y la estructura fundamental.

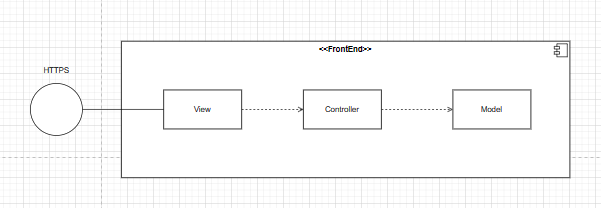
## Línea base arquitectónica de componentes



## Componente back



## Componente Front end



## Estilos y patrones arquitectónicos adoptados

Son un conjunto de prácticas utilizadas en el diseño de aplicaciones o software que ya se han presentado muchas veces y muy similares generando así estilos que ayudan como base o guia a la hora de diseñar un proyecto garantizando así que sea más seguro su manejo y desarrollo. Estos patrones han sido pulidos logrando que quien se basa en ellos tenga un proyecto de calidad y que se puedan manejar los problemas con mayor facilidad.

A la hora de identificar los requisitos se puede evidenciar un poco de como iria la aplicación, a menos de que el cliente pida un estilo o patrón arquitectónico específico casi todos estos por buena práctica se realizan revisiones y análisis a la hora del diseño permitiendo revisar mejoras o cambios que beneficien el proyecto, en el desarrollo estos patrones se deberán verse aplicados en la implementación del proyecto

## Estilo arquitectónico 1

Divide una aplicación en servicios pequeños e independientes encargados de una única funcionalidad

## Nombre

Arquitectura de microservicios

## Problema

Es Resiliencia es necesario que el proyecto tenga una gran tolerancia a fallos un proyecto donde se esté haciendo publicaciones en tiempo real y se esté usando constantemente debe tener gran capacidad para operar inclusive con fallos

## Solución/Motivación

Para solucionar el problema mensionado el sistema se define de manera diferente a lo que se espera en una aplicación monolítica donde se tiene el proyecto en una sola entidad, la Arquitectura de microservicios de basa en dividir el proyectos en múltiples microservicios más pequeños independientes y autónomos, adicional a esto la arquitectura permite una gran facilidad para escalar el sistema o de usar diferentes tecnologías.

## Estilo arquitectónico 2

dividir un microservicio en tres componentes principales

## Nombre

la arquitectura de tres capas en microservicios

## Problema

los microservicios pueden ser complejos en algunos casos el orden en el sistema es un problema grave cuando se le va a ser mantenimiento al microservicio en cuestión

## Solución/Motivación

este estilo arquitectónico permite organizar de forma reconocible un micro servicio ya que este permite al desarrollador encargado de hacer mantenimiento o en su defecto al desarrollador encargado de crear el microservicio seguir un orden, para este caso se implementa por tres capas una para la interfaz de usuario, otra para la lógica y por último una para el acceso de datos.

Con esta solución al tener una capa de acceso de datos se busca tener una base de datos única para cada micros servicio y adicional a esto con la capa de interfaz de usuario se busca enfatizar en que cada microservicio va tener su propia interfaz de usuario lo que permite al equipo enfocarse en dar un experiencia de usuario más enfatizada a ese microservicio en específico.

# Justificación alternativa de solución

A lo que se refiere una justificación de una alternativa es que se debe explicar el porqué de una opción específica es la mejor manera de resolver un problema, que en este caso nuestro problema sería resolver la complejidad en la gestión de la programación de clases, la gestión de membresías y la facturación. De este modo para la solución de este presentamos argumentos sólidos y pruebas que puedan respaldar cual es la opción más adecuada y eficaz de resolver nuestros problemas.

## Justificación

Lo que buscamos como grupo es desarrollar una aplicación móvil y una plataforma web interactiva y fácil de usar que mejore la conectividad, fomente una interacción entretenida entre usuarios y empleados, y facilite tareas como la creación de clases y otras actividades relacionadas con el centro deportivo.

## Ventajas

Las ventajas más importantes y que destacan serían las siguientes

Accesibilidad mejorada, esta es muy importante por que los usuarios pueden acceder a la información del centro deportivo en cualquiera lugar y además de eso, lo pueden hacer a través de sus dispositivos móviles o computadoras

Interacción en tiempo real, es muy importante que el usuario vea actividades actualizadas y que esto sea este presentando en constante momento, de este modo se permite y brinda una mejora entre el usuario y los entrenadores.

Facilidad de reservas, es muy importante para nosotros que el usuario pueda acceder a las reservas de las clases y actividades programadas por diferentes entrenadores, de este modo se simplifica el proceso de ir y hacerlo presencialmente

## Desventajas

Algunas de las desventajas que presentamos serían las siguientes

Costo Inicial, es un de las principales desventajas que tenemos debido a que el desarrollo de una aplicación móvil y una plataforma web personalizada puede ser muy costosa en los términos de desarrollo y un mantenimiento adecuado que es el que tenemos pensado

Requerimientos técnicos, Otra de las desventajas que puede llegar a presentar nuestro proyecto es que los usuarios necesiten dispositivos compatibles y con acceso a internet para poder usar la plataforma, dicho anterior esto puede excluir a personas que no tengan acceso a tecnología o no estén familiarizadas

Dependencia tecnológica, es otra de las desventajas más importantes que presenta nuestro proyecto y es la dependencia a la tecnología, esto puede traer cualquier interrupción en el servicio en línea que podría afectar la comunicación y la gestión de actividades planificadas.

# Vistas de arquitectura del sistema

Representaciones de los diferentes aspectos de la aplicación. Cada vista del modelo de 4 +1 vistas se enfoca en un conjunto de características y proporciona información

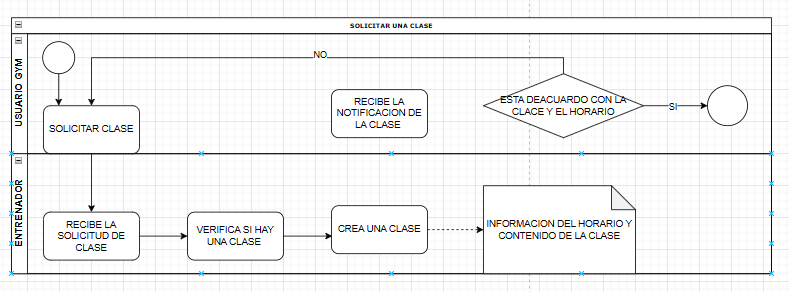
## Vista Funcional/Vista de Escenarios/Vista de Casos de Uso

Representación de las funciones que realiza la aplicación. La vista funcional se enfoca en describir características y funcionalidades que el sistema debe de cumplir para satisfacer las necesidades del usuario final.

Esta vista muestra cómo debería funcionar desde el principio las tareas o métodos de la aplicación y cómo responde a las políticas de negocio haciendo que se evalúe como están propuestas tanto las políticas y como el usuario interactúa al usar dicha función, proporcionando así la evaluación y retroalimentación tanto de mostrar cómo reacciona la función y cómo debería mejorarse

## Modelo de procesos del negocio

Representación visual del diagrama mostrando las actividades, entidades y flujos de los métodos y dentro de la aplicación, con la descripción de las diferentes etapas desde el inicio en el que se ejecuta el comando a las tareas y decisiones involucradas en el proceso de completar la misma, este diagrama ayuda a comprender e identificar posibles mejores y la comunicación entre las partes que usan los equipos.



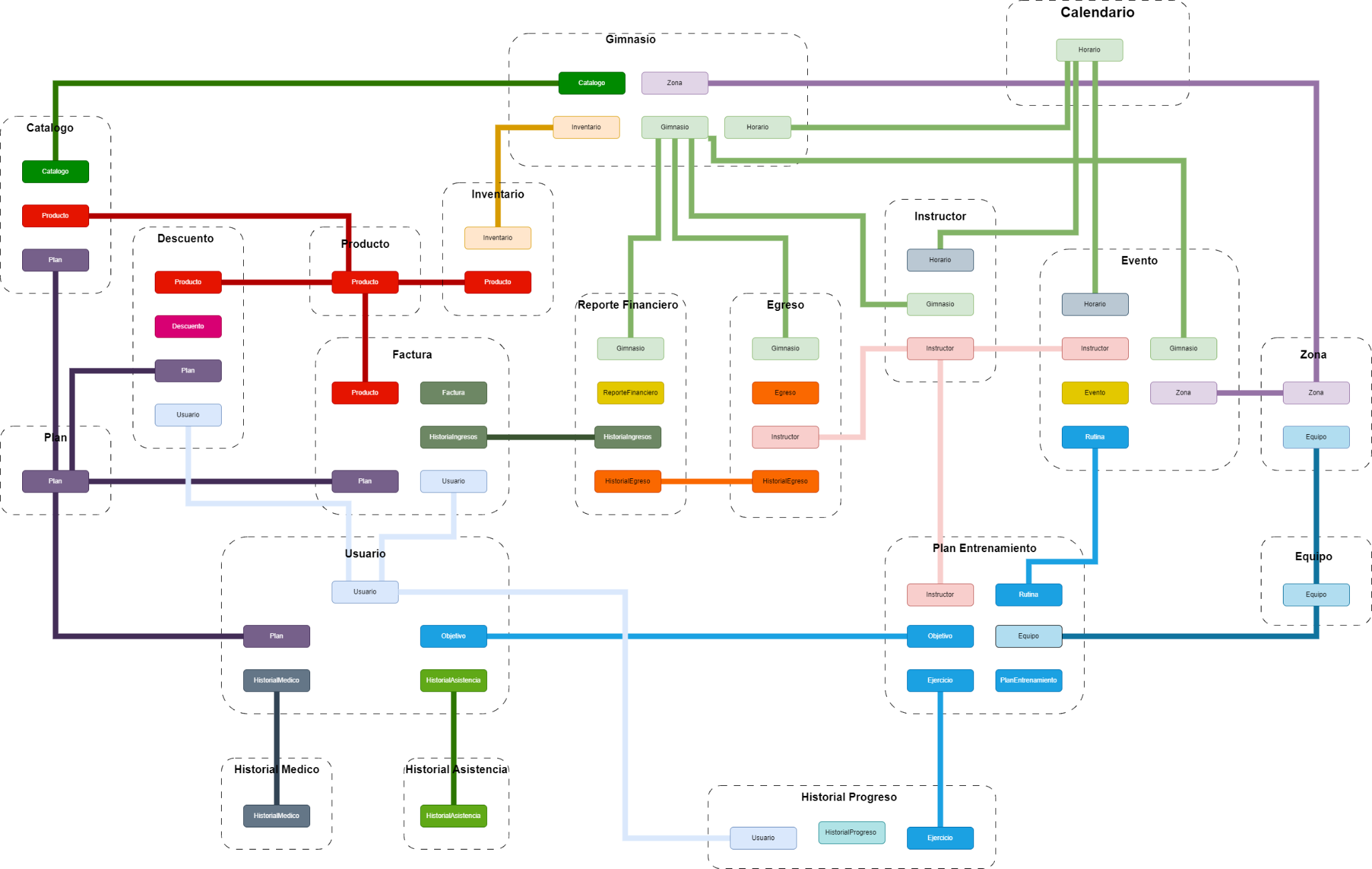
## Modelado de dominio

Representa las entidades y relaciones que existen en el dominio de negocio, creando un modelo o diagrama que captura los aspectos esenciales y que puede ser usado como base para el diseño y desarrollo

## Modelo de contextos

Un modelo de contexto acotados se refiere a leer una sola parte específica de un texto en lugar de leer todo el documento, básicamente se puede decir que este modelo se enfoca solo en una parte limitada de la información para entender y responder preguntas, en lugar de considerar toda la información que tenemos a nuestro alcance. En resumidas palabras este modelo nos ayuda a obtener respuestas más precisas y relevantes en situaciones específicas

## Diagrama



Debemos de capturar y mostrar la información relevante que rodea a una entidad comprendida como un contexto, motivandonos a comprender y analizar la situación con respecto al contexto o entidad involucrada

Debemos tener la capacidad de ver las diferentes situaciones complejas y tener una visión general de las situaciones y sus factores, útil para tomar decisiones efectivas, también es necesario para identificar las relaciones que pueden tener los contextos y sus interacciones

## Documentación contextos

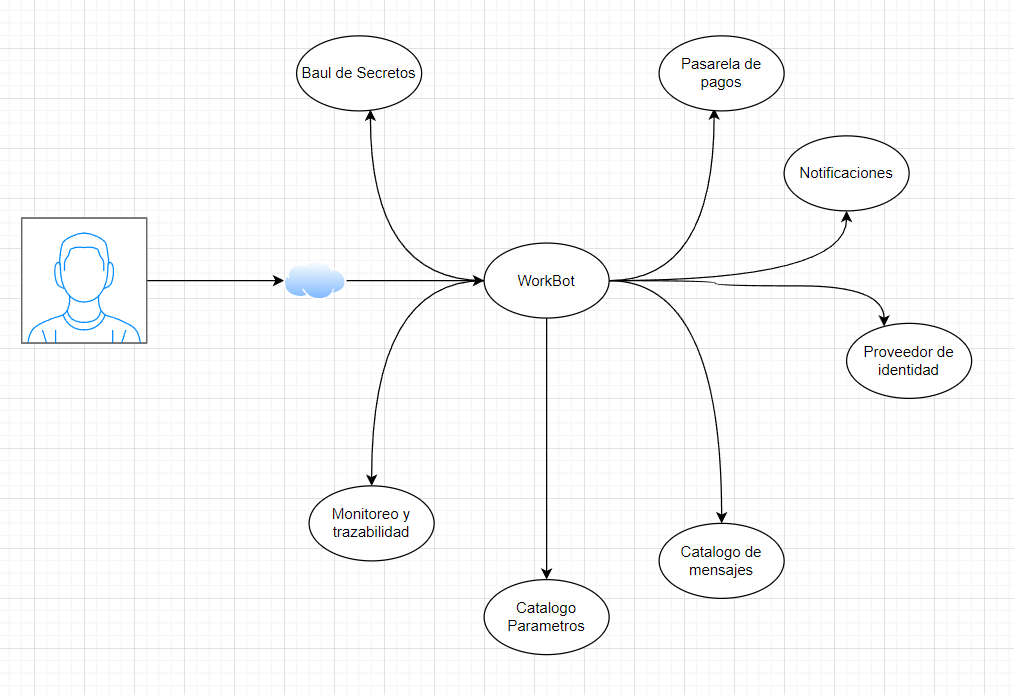
| Título | Contexto Gimnasio |
| --- | --- |
| Descripción | el sistema de gestión de centros deportivos contiene diferentes componentes los cuales se encargan de modelar el funcionamiento general de la aplicación |
| Contexto | gimnasio |
| Objetivo | mostrar las relaciones que se tiene en los diferentes aspectos de los componentes que interactúan en el sistema |
| Elementos | catálogo , descuento, factura, plan, inventario, egreso, instructor, evento, zona, calendario, usuario, historial, plan de entrenamiento. |

## Modelo de mapeo de contextos

Un modelo de mapeo de contextos acotados es una representación visual la cual nos muestra una parte específica y limitada de un sistema o proceso que queramos implementar en nuestro proyecto. Básicamente en lugar de ver todo el sistema , lo que hacemos es enfocarnos en una sola parte del sistema y de esta forma podemos entender como funciona todo el modelo en esa área específica.

## Diagrama

El mapeo de contextos es muy esencial para poder comprender la complejidad de los sistemas, como se puede ver en el diagrama anterior podemos observar algunos de los contextos que va tener nuestro sistema, los cuales son esenciales dentro de este.



## Documentación mapeo de contextos

Baúl de secretos : Contraseñas y datos sensibles, así como también el cifrado

Pasarela de pagos :Muy necesario ya que la aplicacion debera poder mandar al cliente a pagar la membresía, y estos permiten devolver si fue exitoso o no

Notificaciones : Se debe poder enviar notificaciones a los miembros sobre eventos, ofertas o actividades que realice el centro deportivo

Proveedor de identidad : El mínimo de seguridad tanto para el usuario final como el miembro

Catálogo de mensajes: Mensajes para cada error o excepción que llegue a suceder, con el fin de que la aplicación no tenga que guardarlos y liberar código quemado

Catálogo de parámetros: Manejo de horarios, tarifas y membresías, así como también un programa de actividades

Monitoreo y trazabilidad: En caso de que aumente demasiado la cantidad de usuarios que usan la aplicación para algo esto le permitirá aumentar los recursos como corresponde

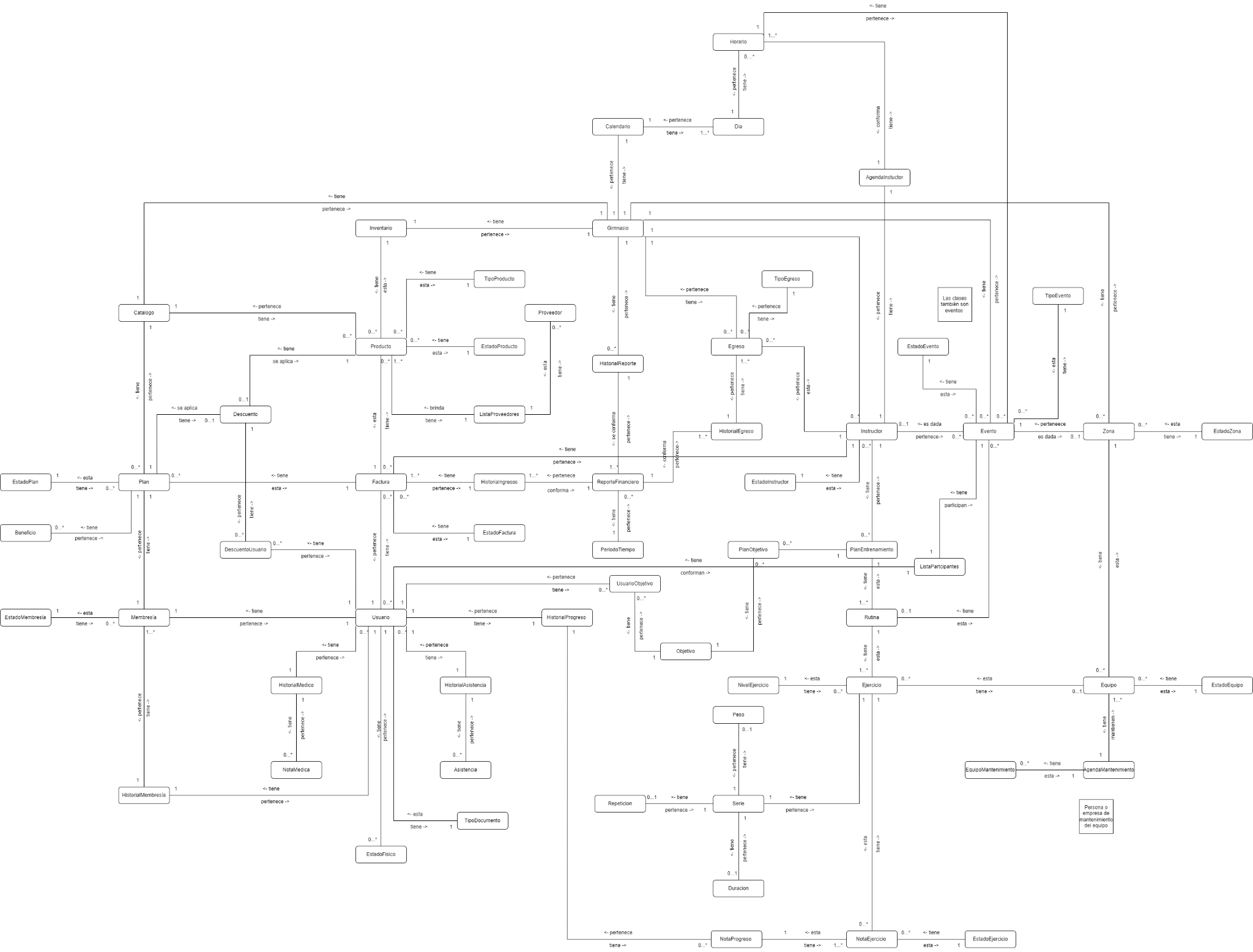
## Modelos de dominio

Un modelo de mapeo de contextos acotados es una representación visual la cual nos muestra una parte específica y limitada de un sistema o proceso que queramos implementar en nuestro proyecto. Básicamente en lugar de ver todo el sistema , lo que hacemos es enfocarnos en una sola parte del sistema y de esta forma podemos entender como funciona todo el modelo en esa área específica.

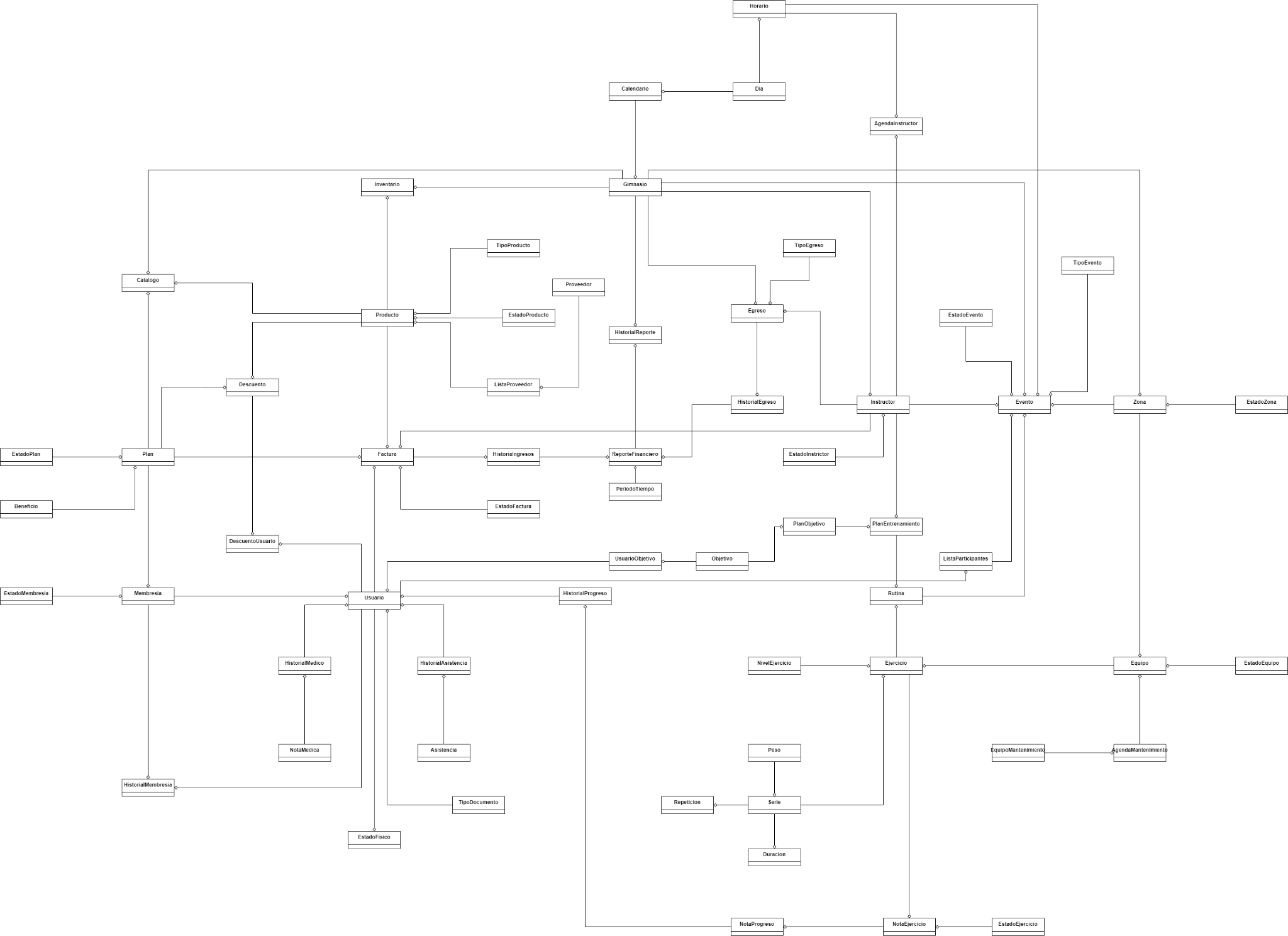
Este diagrama simplifica la complejidad, este mejora la comunicación y permite que el desarrollo del proyecto sea más eficiente y las decisiones de este sean más informadas, lo que sería esencial en el e desarrollo de software y en la gestión de proyecto

## Contexto 1

## Modelo anémico



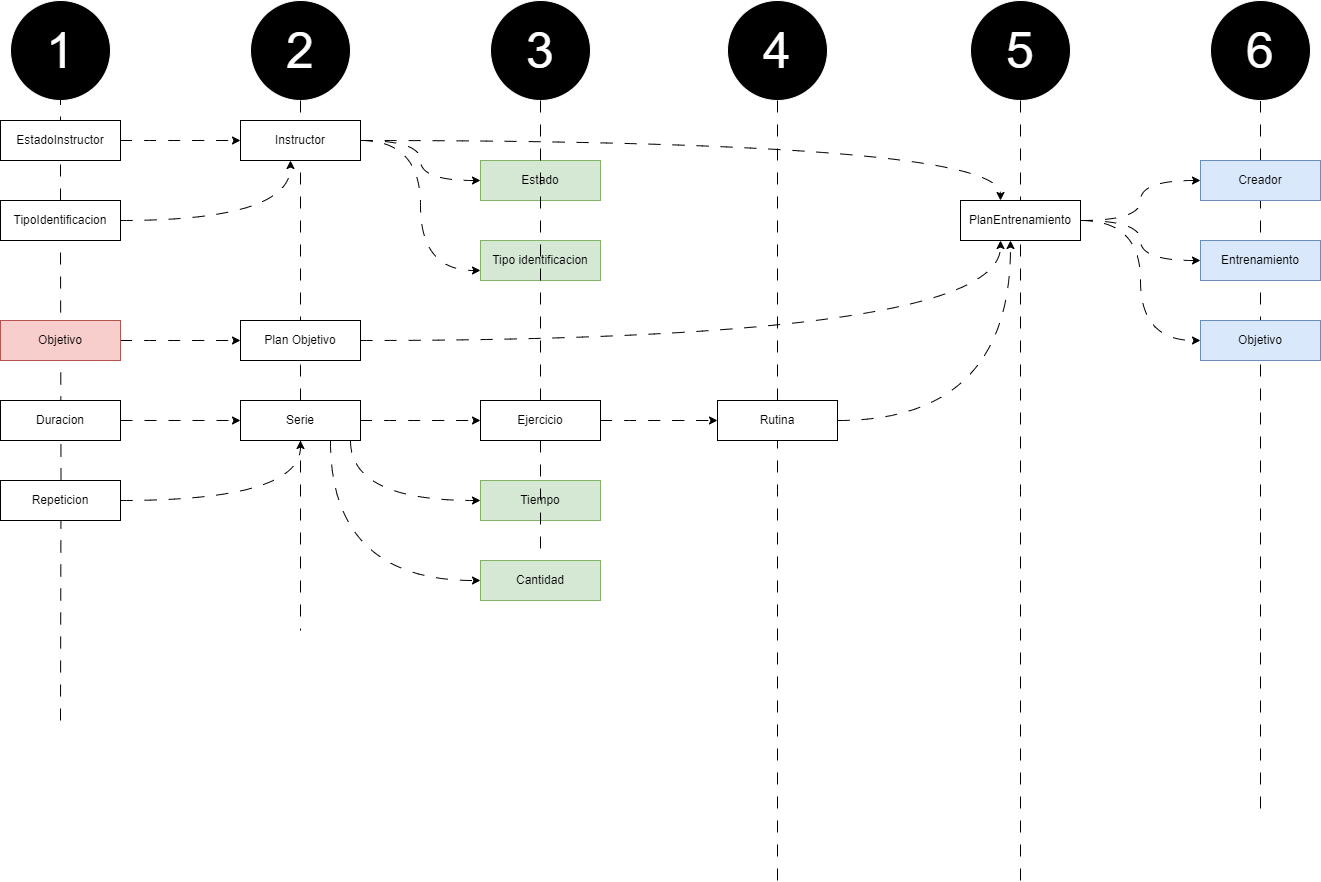
## Modelo enriquecido



## Flujo de eventos/Event Storming

Básicamente a lo que se refiere con un flujo de eventos seria a una lista ordenada de cosas o pasos que suceden en un proceso o situación , como se ha dicho anteriormente es como describir paso a paso de como van ocurriendo las acciones y de este modo nos ayuda a atender el proceso de principio a fin.

## Diagrama



## Glosario de términos del negocio

**Abonado**: Persona que ha pagado por una membresía o suscripción del centro deportivo

**Base de datos:** Almacenamiento de información sobre usuarios del centro deportivo

**Clases dirigidas:** Son sesiones o clases las cuales son dirigidas por un entrenador con el propósito de enseñar y mejorar la condición física de los clientes

**Equipamiento:** Maquinas y accesorios equipados en el centro deportivo con el fin satisfacer al cliente

**Entrenador personal:** Profesional especializado el cual brinda un entrenamiento personalizado y le ayuda al cliente que lo solicita a cumplir sus objetivos o metas

**Monitoreo de progreso:** Se trata del seguimiento y la evaluación regular del condicionamiento físico de cada usuario del centro deportivo

**Soporte al cliente:** Servicio de atención al cliente que responde a las preguntas y preocupaciones que estos tengan dentro del centro de deportivo

**Tarifas de inscripción:** El costo inicial que un nuevo miembro debe pagar al unirse al centro deportivo

**Torneos y competiciones:** Eventos organizados por el centro deportivo u otra organización asociada al centro deportivo, el cual le brinda al usuario poder competir en diferentes actividades deportivas

**Venta de productos:** La oferta de artículos relacionados con el deporte y el fitness

## Especificación de requisitos de software

Una especificación de requisitos de software es un documento que detalla lo que el software debe hacer de manera muy detallada y precisa. Este documento es muy importante por que le permite una guía a los desarrolladores donde establecen los objetivos claros y las funciones claves que tiene que ir dentro de este, además de esto ayuda la comunicación entre los stakeholders y los desarrolladores lo cual es muy importante. Este también nos define cómo debe ser la arquitectura, la interfaz de usuario y las funcionalidades específicas que debe tener el sistema.

## Requisitos de usuario

**Registros de cuentas:** Los usuarios que ingresan al centro deportivo se tienen que registrar, este registro proporciona la información como el nombre , edad, ciudad de origen, tipo de sangre , dirección del correo electronico y numero de celular

**Autenticación segura:** Se requiere un proceso de autenticación segura para que esta proteja la información de los clientes y garantice privacidad

**Reserva de instalaciones:** Los usuarios deben poder reservar instalaciones como lo son las lasas de entrenamiento en horarios específicos

**Clases y programas:** Los usuarios deben poder inscribirse en clases dirigidas, como yoga o spinning y también pueden acceder a información sobre horarios y disponibilidad de clases

**Monitoreo del progreso:** El sistema le debe permitir a los usuarios realizar un seguimiento de su progreso físico, como el peso, la frecuencia cardiaca y los logros que este ha tenido en cada uno de los entrenamientos

**Acceso a entrenadores personales**: Los usuarios deben poder programar sesiones con entrenadores personales

**Acceso a equipamiento:** Los usuarios deben poder acceder al equipamiento que brinda el centro deportivo de una forma segura y eficiente

## Requisitos del sistema

Los requisitos del sistema son condiciones o funciones que el sistema de software debe cumplir para poder que este cumpla la satisfacción de las necesidades del cliente y los usuarios finales . Estos sirven para detener un mapa detallado que guía al equipo de desarrollo en todas las etapas del proyecto, desde la planificación hasta el diseño, desarrollo, prueba y el mantenimiento. además de esto asegura que el producto final cumpla con las expectativas del cliente y los usuarios finales de manera efectiva y eficiente

## Requisitos funcionales

Describen lo que el sistema debe hacer y se centran en las funciones y características específicas que debe proporcionar.

El sistema debe ser seguro para el registro de miembros, control de acceso a instalaciones y la gestión de membresías para poder garantizar que solo los usuarios autorizados tengan acceso a las instalaciones del centro deportivo

Los usuarios deben poder reservar plazas en clases específicas que sean de su agrado, cancelar y recibir notificaciones sobre los cambios de horarios de las clases

El sistema debe permitir el registro y la gestión de los miembros, esto incluye la información personal del cliente, tipo de membresía, fechas de inicio y el vencimiento de estas , para así mismo tener un control del plan que tiene cada usuario

El sistema debe permitir la programación y la gestión de clases dirigidas, eventos especiales y torneos deportivos que se creen dentro del centro deportivo, de igual le brinda información al usuario por medio de notificaciones

Los usuarios deben poder hacer un seguimiento de su progreso físico y de salud, como el peso, medidas corporales y estadísticas de cada entrenamiento que hacen.

El sistema debe permitir enviar notificaciones a los miembros sobre clases creadas, canceladas, modificadas, eventos especiales y las promociones que el centro deportivo ofrece.

El sistema debe generar informes sobre la participación de los miembros, la ocupación de las instalaciones del centro deportivo.

## Requisitos no funcionales

Describen cómo debe ser el sistema en términos de calidad, rendimiento, seguridad, usabilidad y otros atributos que no se refieren directamente a las funciones del software.

El sistema debe ser capaz de manejar un número específico de usuarios concurrentes sin experimentar degradación del rendimiento, incluso en las horas pico del centro deportivo

El sistema debe cumplir con estándares de seguridad de datos y transacciones, utilizando encriptación y autenticaciones seguras para poder proteger a toda costa la información del usuario

El sistema debe ser escalable para que este pueda adaptarse al crecimiento futuro del número de usuarios y también al número de carga de trabajo sin que este requiera cambios significativos en la arquitectura

El sistema deber ser fácil de usar y comprender para los usuarios finales y más que todo para personas con algunas dificultades cognitivas, minimizando la necesidad de capacitación extensiva y que facilite la interacción del usuario

## Requisitos de información

Los requisitos de información en software se basan en detalles específicos sobre los datos que el sistema debe capturar, almacenar, procesar y mostrar . Estos requisitos son fundamentales para que el sistema tenga un funcionamiento efectivo y eficiente.

Detalles del miembro (RI 001) : Incluye nombre, dirección, fecha de nacimiento, tipo de sangre,peso, correo electrónico y número de teléfono del miembro.

Reserva de clases (RI 002): Contiene información sobre las clases reservadas, incluyendo la fecha, la hora y el nombre de la clase, al igual que el entrenador que dará dichas clases .

Información financiera del usuario (RI 003):Esta involucra detalles de pagos, facturas y transacciones de los miembros, como membresías y compras adicionales las cuales pueden hacer dentro del centro deportivo.

Progreso del usuario (RI 004): Registra todo el progreso del usuario, como medidas corporales, peso, registros de entrenamiento y logros alcanzados, este progreso se realiza durante el periodo que el cliente desee.

Seguridad y acceso al centro deportivo (RI 005): Incluye los registros de acceso, detalles de seguridad y eventos relacionados con la seguridad del centro deportivo.

## Reglas de negocio

Las reglas de negocio son las pautas específicas que determinan cómo se deben llevar a cabo ciertas actividades o decisiones dentro del negocio en el cual se implementan, estas reglas suelen incluir políticas sobre membresías, horarios de clases, pagos, seguridad entre otras muchas cosas. Estas son muy importantes ya que facilitan el orden y garantizan una experiencia positiva tanto para el usuario como para los empleados

**Regla de negocio 01(RN 001):** Los miembros deben poder reservar clases en línea con al menos 24 horas de antelación.

**Regla de negocio 02(RN 002):** Los pagos de membresía deben procesarse automáticamente cada mes.

**Regla de negocio 03(RN 003):** Los instructores deben estar disponibles para consultas de los miembros durante al menos dos horas al día.

**Regla de negocio 04(RN 004):** Los instructores deben estar certificados y tener experiencia relevante en el campo para enseñar clases especializadas.

**Regla de negocio 05(RN 005):** Los precios de las membresías y los servicios adicionales deben ser claros y transparentes, incluyendo impuestos y tarifas adicionales.

**Regla de negocio 06(RN 006):** Los descuentos o promociones especiales solo son válidos para los miembros que han estado inscritos durante al menos tres meses.

**Regla de negocio 07(RN 007):**Los registros de los usuarios, incluyendo datos personales y de pago, deben ser almacenados de forma segura y cumplir con las leyes de privacidad y protección de datos.

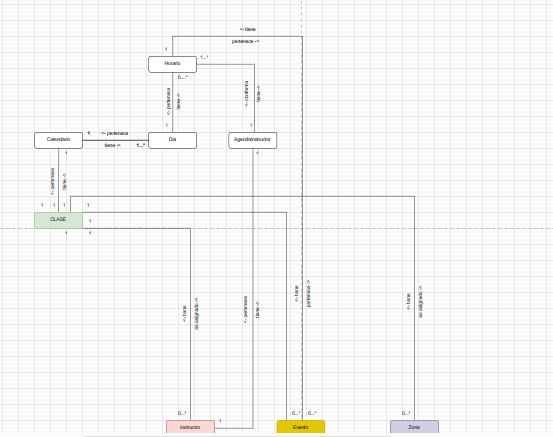
## Casos de uso

La táctica de casos de uso es una táctica la cual se enfoca en el cómo el sistema va a interactuar con el sistema,se enfoca en describir las funcionalidades del software desde la perspectiva del usuario y cómo se comportará en diferentes situaciones.

## Modelo de contexto

Un modelo de contexto es una representación visual o descriptiva que muestra donde el sistema o aplicación se va a operar, este permite observar cómo son las relaciones del sistema con su entorno.

## Diagrama



## Descripción

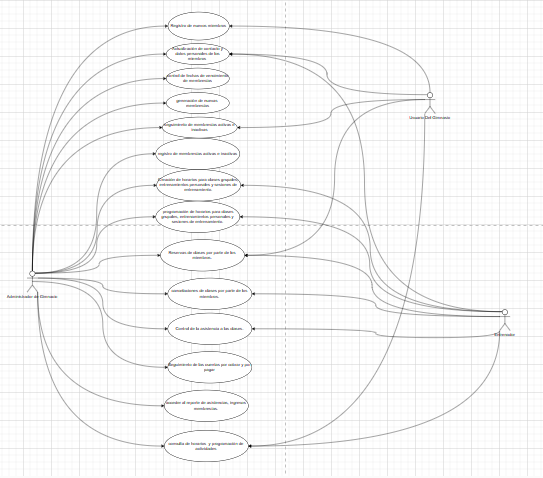
| **Diagrama** | diagrama de contexto de la clase | |
| --- | --- | --- |
| **Descripción** | modelo que muestra las relaciones que tiene el sistema con su entorno | |
| componente | relación | componente |
| clase | tiene | zona |
| calendario |
| evento |
| instructor |
| Calendario | tiene | dia |
| dia | tiene | horario |
| horario | tiene | agenda instructor |
| evento |
| agenda instructor | tiene | instructor |
| instructor | es asignado | clase |
| evento | tiene | clase |
| zona | es asignado | clase |

## Diagramas de casos de uso

La táctica de casos de uso es una táctica la cual se enfoca en el cómo el sistema va a interactuar con el sistema,se enfoca en describir las funcionalidades del software desde la perspectiva del usuario y cómo se comportará en diferentes situaciones.

## Componente gestión de gimnasios

## Diagrama de casos de uso



## Especificación de casos de uso

muestra detalladamente cómo el sistema interactúa con los usuarios o actores, enfocado a las respuestas del sistema a las acciones de los usuarios.

## Caso de uso CREACIÓN DE CLASE

Este caso define el modelo de funcionamiento de la funcionalidad de crear una clase

## Datos básicos caso de uso

| Datos Básicos de Casos de Uso | |
| --- | --- |
| identificador | CU1 |
| nombre | Crear Clase |
| descripción | este caso de uso muestra el proceso que se debe llevar a cabo cuando se crea una clase en el gimnasio |
| actores | instructor |
| referencias | instructor, Calendario, zona, eventos |
| precondiciones | el instructor debe estar logueado |
| post condiciones | el sistema debe funcionar correctamente |
|  |  |

## Escenarios del caso de uso

Un escenario de un caso de uso es una historia detallada que describe la interacción que tiene un usuario con el sistema en una situación específica, este relata paso a paso las acciones de los usuarios.

## Flujo normal/flujo básico

| Datos Básicos de Casos de Uso | |
| --- | --- |
| identificador | CU1 |
| nombre | Crear Clase |
| descripciòn | este caso de uso muestra el proceso que se debe llevar a cabo cuando se crea una clase en el gimnasio |
| actores | instructor |
| referencias | instructor, Calendario, zona, eventos |
| precondiciones | el instructor debe estar logueado |
| post condiciones | el sistema debe funcionar correctamente |
| FLUJO BÁSICO | 1. Un usuario solicita al instructor una clase |
| 2. El instructor ingresa al sistema los datos de la clase, horario, contenido, zona, instructor asignado |
| 3. El sistema valida los datos ingresados como correctos |
| 4. El sistema crea la clase |
| 5. El sistema notifica a los usuarios la creación de una nueva clase |
| clasificación | Primario |

## Flujo alterno 1

| Datos Básicos de Casos de Uso | |
| --- | --- |
| identificador | CU1 |
| nombre | Crear Clase |
| descripciòn | este caso de uso muestra el proceso que se debe llevar a cabo cuando se crea una clase en el gimnasio |
| actores | instructor |
| referencias | instructor, Calendario, zona, eventos |
| precondiciones | el instructor debe estar logueado |
| post condiciones | el sistema debe funcionar correctamente |
| FLUJO ALTERNO | 1. Un usuario solicita al instructor una clase |
| 2. El instructor ingresa al sistema los datos de la clase, horario, contenido, zona, instructor asignado |
| 3. El sistema valida los datos ingresados como incorrectos |
| 4. El sistema pide ingresar los datos de nuevo correctamente |
| 5. El sistema crea una nueva clase |
| 6. El sistema notifica a los usuarios la creación de una nueva clase |
| clasificación | Primario |

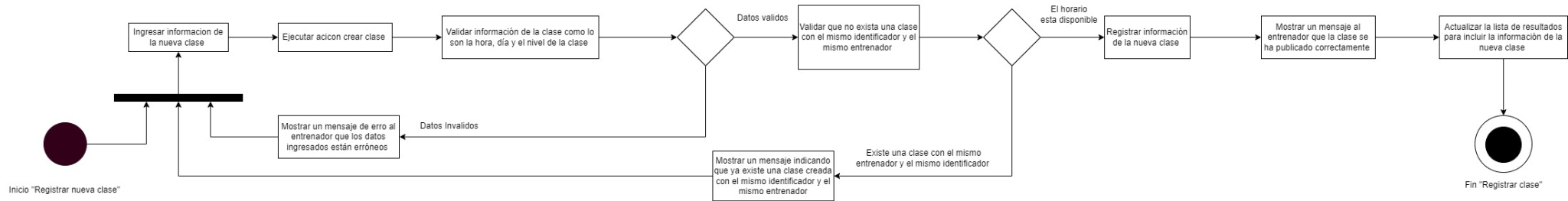
## Flujo Excepcional 1

| Datos Básicos de Casos de Uso | |
| --- | --- |
| identificador | CU1 |
| nombre | Crear Clase |
| descripciòn | este caso de uso muestra el proceso que se debe llevar a cabo cuando se crea una clase en el gimnasio |
| actores | instructor |
| referencias | instructor, Calendario, zona, eventos |
| precondiciones | el instructor debe estar logueado |
| post condiciones | el sistema debe funcionar correctamente |
| FLUJO EXCEPCIONAL | 1. Un usuario solicita al instructor una clase |
| 2. El instructor ingresa al sistema los datos de la clase, horario, contenido, zona, instructor asignado |
| 3. El sistema valida los datos ingresados como correctos |
| 5. El sistema crea una nueva clase |
| 6. El Sistema no notifica al usuario |
| clasificación | Primario |

## Diagrama de actividades

Los diagramas de actividades son especialmente útiles para modelar flujos de trabajo, procesos de negocios y cualquier actividad que involucra una secuencia de acciones. En el diagrama que se les enseñara a continuación se verá el proceso que tiene un entrenador para poder crear una clase y que esta se publique correctamente. De este modo el usuario del centro deportivo podrá ver las actividades publicadas y tomar la decisión de si asistir o simplemente ignorar la clase.

## Diagrama



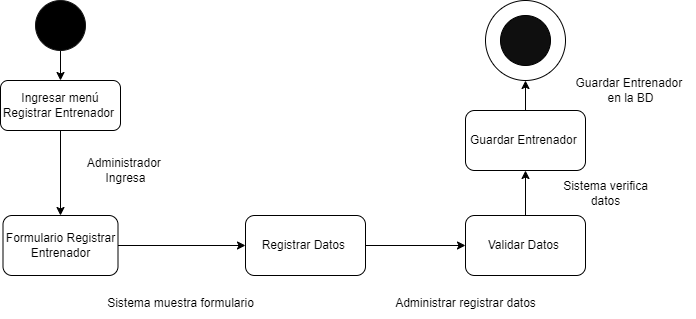
## Documentación

| **Diagrama** | Actividades | |
| --- | --- | --- |
| **Transición:** | Registrar la información de una nueva clase | |
| **Descripción** | Modela el flujo de actividades/operaciones que se deben llevar a cabo para registrar la información de una una clase del centro deportivo | |
| **Componente** | **Tipo** | **Descripción** |
| Ingresar la información de la nueva clase | Actividad | Corresponde a la actividad donde el entrenador ingresa la información relacionada con la nueva clase. En esta actividad a nivel de experiencia de usuario, se garantiza que los datos se validan en tiempo real , asegurando que sea muy amigable e intuitiva la interacción con el usuario |
| Decision Datos | Decisión | Punto de decisión que permite saber el flujo se debe tomar dependiendo del resultado de la validación de los dato ingresados |
| Decision Datos | Decisión | Punto de decisión que permite saber el flujo se debe tomar dependiendo del resultado de la validación de los dato ingresados como son como lo son la hora, día y el nivel de la clase |
| Decision Datos | Decisión | Punto de decisión que permite saber el flujo se debe tomar dependiendo del resultado de la validación de los datos con respecto a otros horarios y zonas u equipos a usar |
| Enviar Notificación | Actividad | Corresponde a la actividad donde el sistema envía la información relacionada con la nueva clase. En esta actividad a nivel de experiencia de usuario, se garantiza que los datos se envíen a tiempo real, asegurando que el usuario se mantenga informado asi como tambien que sea muy amigable e intuitiva la interacción con el usuario |

## Diagrama de estados

Un diagrama de estados es como un mapa visual el cual nos muestra las diferentes situaciones por las que un objeto o algo pueden pasar, este es demasiado útil ya que nos brinda la capacidad de entender el como un objeto o sistema se cambia de un estado a otro en respuesta de eventos o acciones específicas.

## Diagrama



## Documentación

| **Diagrama** | Estados | |
| --- | --- | --- |
| **Transacción** | Registrar a un entrenador | |
| **Descripción** | Modela los estados de registro por los que puede pasar un entrenador | |
| **Estado Origen** | **Estado Destino** | **Evento/Disparador** |
| Inicio | Ingresar menú registrar entrenador | Estado inicial |
| Ingresar menú registrar entrenador | Formulario registrar entrenador | El entrenador ingresa al formulario para registrar sus datos |
| Formulario registrar entrenador | Registrar datos | El entrenador registrar sus datos |
| Registrar datos | Validar datos | El entrenador espera la validación de sus datos |
| Validar datos | Guardar entrenador | El entrenador espera que sus datos sean guardados en la BD |
| Guardar entrenador | Fin | El entrenador ha sido guardado |

## Incepción Ágil

Básicamente cuando nos hablan de agile inception se trata a una reunión inicial en el desarrollo ágil de un proyecto, en este se reúnen las partes interesadas para establecer los objetivos claros, definir un plan de acción, identificar posibles problemas y lo más importante, obtener una retroalimentación temprana con el cliente para ver cuales son sus ideas y lo que este quiere . Esto es muy importante para nuestro proyecto del centro deportivo ya que nos permite desde un comienzo definir los objetivos claros con el usuario, nos permite también crear un plan de acción, nos permite identificar los riesgos y obstáculos que puede presentar este y lo más importante de nuestro proyecto sería la incorporación de la retroalimentación constante con el cliente.

## Por qué estamos aquí

Estamos aquí ya que es importante tener una reunión inicial en el desarrollo de nuestro proyecto del centro deportivo. Dicha reunión es muy crucial para nosotros como desarrolladores debido a que nos permite establecer los objetivos claros, crear un plan de acción, identificar posibles problemas que ocurran a lo largo del desarrollo y lo más importante que como hemos mencionado anteriormente obtener la opinión temprana del cliente. También queremos que no sea algo temprano queremos que las reuniones sean recurrentes para mostrarle el desarrollo al cliente y ver si este si está conforme o si tiene alguna otra idea por aportar. Debido a que al obtener una retroalimentación podemos asegurarnos de que el proyecto se alinee con sus expectativas, también nos permite ajustar y mejorar el diseño del proyecto para que se adapte a los deseos del cliente.

## Visión/Elevator Pitch

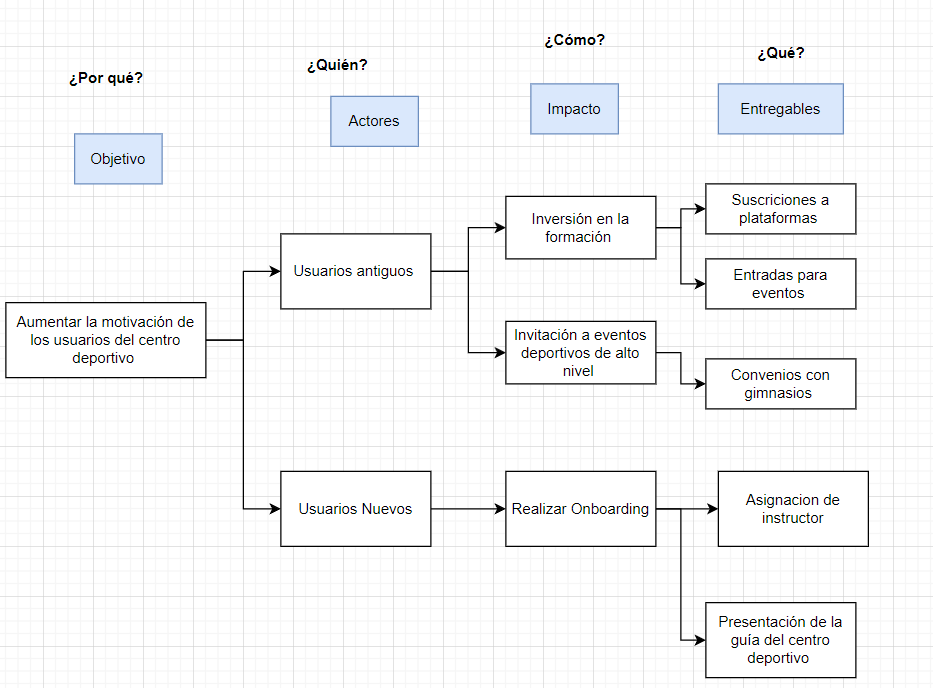
La visión es lo más importante debido a que esta es una descripción detallada de lo que nuestro proyecto pretende lograr y como se verá una vez terminado, En pocas palabras la visión sería un mapa que guía a todo nuestro equipo de desarrollo durante todo el desarrollo del proyecto del centro deportivo. Esta nos muestra una imagen clara del resultado final del proyecto y nos sirve para guiar a todos los involucrados como se mencionó anteriormente, también nos permite establecer objetivos claros facilitando la toma de decisiones y permitiendo tener un buen enfoque al proyecto

## Visión

Para gimnasios, clubes deportivos, centros recreativos y otros establecimientos dedicados a actividades físicas o deportivas en el que los centros deportivos necesitan resolver la complejidad en la gestión de la programación de clases, la gestión de membresías y la facturación; además aprovechar la tendencia al uso de las aplicaciones y plataformas digitales en auge. Transfórmate es un sitio web que su principal beneficio es cambiar los trabajos manuales para ser automáticos, simplificando la gestión del centro deportivo. Esto implica: eficiencia operativa optimizando el tiempo y el esfuerzo invertido a tareas administrativas repetitivas como la programación de clases, gestión de membresías y la facturación, agilidad en la toma de decisiones al proporcionar acceso instantáneo a datos e informes. A diferencia de MindBody, GymMaster, ZenPlanner, Wodify, MyClup, PerfectGym, TeamUp, Clubworx, PlayVen, UpperHand, nuestro productos de software ofrece características respecto a las necesidades de cada centro deportivo y de fitness así mismo un soporte local al centro deportivo y se incluye la capacitación.

## Mapa de impacto

Un mapa de impacto es una representación visual la cual nos muestra el como un proyecto afecta a diferentes partes interesadas y el cómo estas se influyen mutuamente, en otras palabras nos muestran los actores del sistema y estos como se ven afectados por sus acciones u opiniones que puedan influir en el proyecto. Este es util por que nos brinda una identificación por las partes interesadas, visualización de las relaciones, nos facilita una mejor planificación y apoya a la toma de decisiones



## Tamaño/Talla de historias de usuario

Cuando nos hablan del tamaño se refiere a la magnitud, complejidad o alcance que puede llegar a tener un proyecto en términos de recursos necesarios, los cuales serían el tiempo, dinero, personal y la tecnología que se usará en este. Al tener un tamaño destinado en un proyecto es muy importante por que este nos brinda una planificación, presupuesto, recursos que se utilizaran en este

## Tallaje del producto

Cuando nos hablan de tallaje de historias de usuario hace referencia a estimar cuánto trabajo tomará implementar cada tarea del proyecto a trabajar de software. Básicamente este nos ayuda a planificar, asignar recursos y a llevar un seguimiento cuidadoso del progreso que vamos teniendo al ir avanzando

## Definiciones para el tallaje

Algunas de las condiciones definidas que tiene nuestro proyecto serían las siguientes :

Recursos disponibles

Estabilidad tecnológica

Retroalimentación

Documentación

Complejidad

## Tallaje del producto

| **Historial usuario** | **Descripción** |
| --- | --- |
| **Registro de miembros** | Como usuario, quiero poder registrarme como miembro del centro deportivo para acceder a las instalaciones y clases. |
| **Reserva de clases** | Como miembro, quiero poder reservar clases de fitness en línea para asegurar mi lugar. |
| **Seguimiento del progreso** | Como usuario, quiero poder registrar mi progreso físico y ver mis estadísticas en un gráfico para seguir mi evolución. |
| **Administración de pagos** | Como administrador, quiero una interfaz para gestionar los pagos de membresía y las transacciones de los usuarios. |

## Release Plan

Un Reléase Plan es esencial para la planificación y gestión efectiva del proyecto a realizar .Esta proporciona una guía clara para el desarrollo, además de eso asegura la entrega puntual de las características clave y contribuye significativamente a la satisfacción del cliente y al éxito general del proyecto

## Vista Lógica

Cuando hablamos de vista lógica en el diseño de un sistema o proyecto se refiere a una representación abstracta o estructurada de cómo las diferentes partes o componentes de un sistema interactúan y se comunican entre sí para poder lograr los objetivos propuestos en un comienzo. Al tener un vista lógica nos ayuda mucho a entender el cómo se comporta y organiza internamente a la hora de realizar las tareas específicas.

Al trabajar con una vista lógica podemos obtener varios aportes los cuales serían Comprensión del sistema, identificación de interacciones, facilita la comunicación, modificación y el mantenimiento del sistema

## Diagrama de clases

Un diagrama de clases es una herramienta visual la cual se utiliza en software y se encarga de representar las clases y las relaciones entre ellas en un sistema. Esta brinda una ayuda para entender, organizar y comunicar la estructura del software de una manera clara y visual. Este diagrama es muy importante debido a que nos proporciona una comprensión clara y organizada de las estructuras y las relaciones que se tiene en un sistema

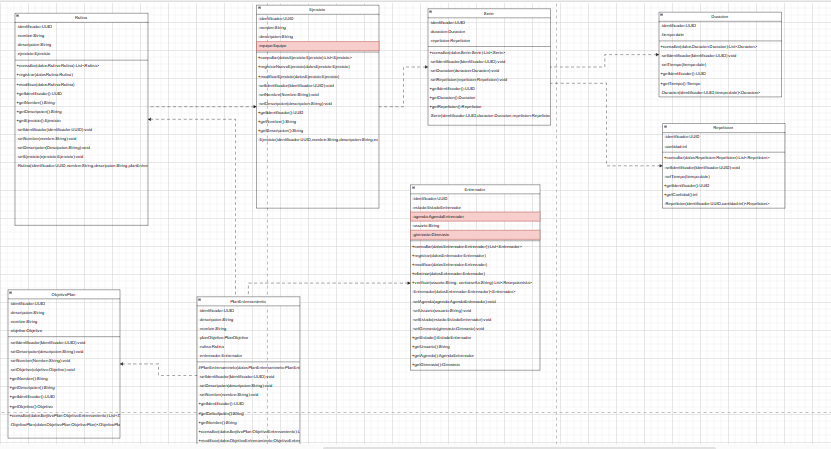
Lo que más nos aporta un diagrama de clases a nuestro proyecto serian las siguientes caracteristicas:

* Visualización de la estructura
* Identificación de clases y objetos
* Documentación del Diseño

## Componente 1

El objetivo de un diagrama de clases es brindar una representación visual de las clases relaciones y objetos que componen el sistema

## Diagrama



## Documentación

<Muestre y/o detalle el diagrama de clases del componente en cuestión con la documentación respectiva.>

## Diagrama de objetos

Un diagrama de objetos es una representación visual estética la cual muestra los objetos del sistema y sus relaciones en términos de estructura y asociaciones en un instante determinado, este diagrama es muy útil debido a que nos permite entender el cómo actúan los objetos y cómo se comunican en un estado particular del sistema. Lo que nos brinda este diagrama es en la necesidad de entender cómo los objetos de este mismo sistema se relacionan entre sí en un momento determinado

Las principales características que el diagrama de objetos brinda a nuestro sistema serían las siguientes :

* Claridad en la estructura
* Identificación de relaciones
* Análisis de estados
* Documentación Detallada

## Diagrama de componentes

Un diagrama de componentes es una herramienta visual la cual es utilizada para representar una estructura y las relaciones entre los componentes de un sistema. La motivación que tiene nuestro sistema con dicho diagrama es la necesidad de entender cómo están organizadas las partes de un sistema y cómo interactúan entre sí. Ya que si al este ser grande se puede volver un poco complejo y difícil de comprender

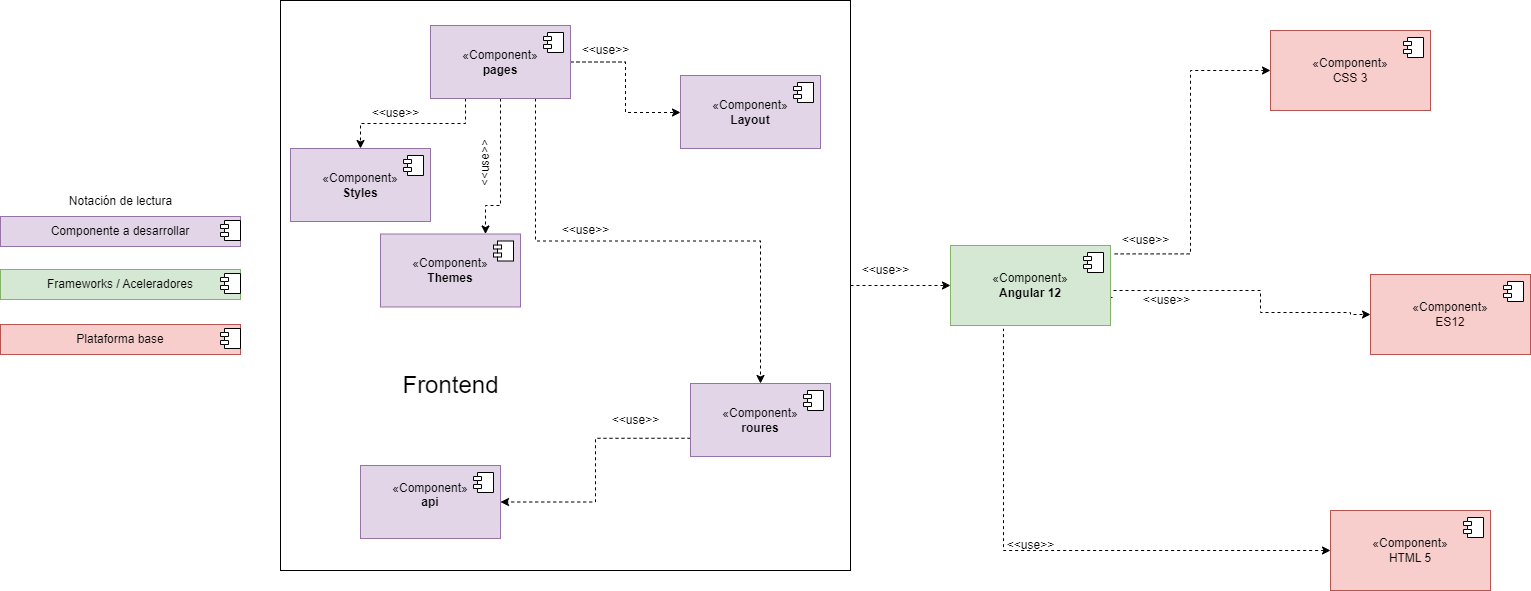
Lo que el diagrama de componentes aporta a nuestro sistema serían lo siguiente:

* Visualización clara
* Organización Modular
* Identificación de interacciones

## Componente 1

Un Diagrama de componentes tiene como objetivo mostrar a las desarrolladores y las personas implicadas el cómo las diferentes piezas del sistema interactúan entre sí para este caso las piezas que tenemos en un Frontend

## Diagrama Componentes Frontend



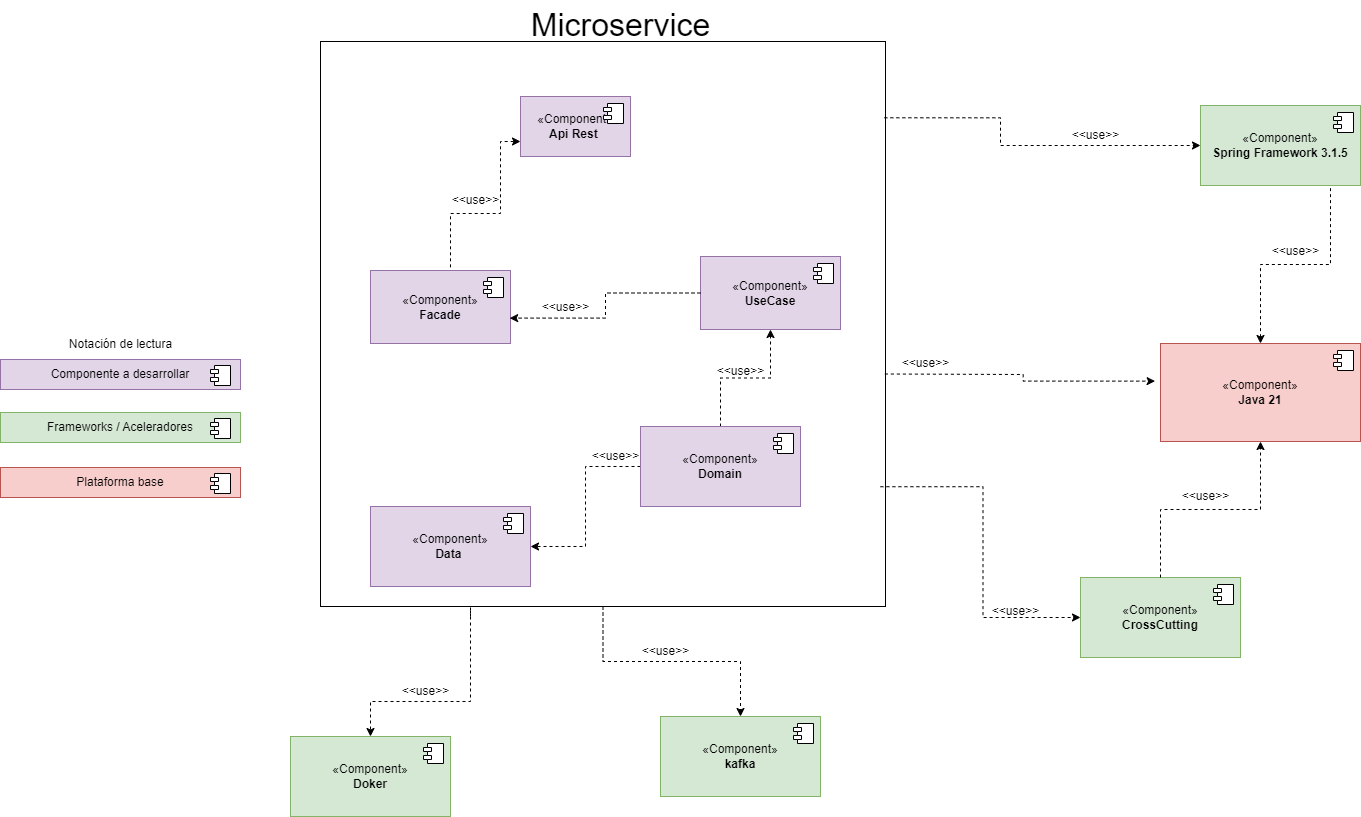
## Documentación

| **Diagrama** | Componentes | | | |
| --- | --- | --- | --- | --- |
| **Componente** | Arquetipo general | | | |
| **Descripción** | Diagrama que muestra el 100% de los componentes que serán construidos o adaptados para proceder con la construcción del componente indicado. Adicionalmente, indica cómo estos componentes están relacionados | | | |
| **Componente** | **Descripción** | **Motivación** | **Depende/Usa** | **Tipo Componente** |
| controller | Componente que representa la infraestructura de cada microservicio que se construirá dentro de la aplicación de forma genérica | componente que va a contener los elementos necesarios para que un microservicio funcione de manera eficiente y sea escalable. | CrossCutting | Componente a desarrollar |
| Spring Framework 3.1.5 |
| Java 21 |
| view | Componente que representa la parte de application core de cada microservicio que se construirá dentro de la aplicación de forma genérica | componente que va a contener la parte central del micro servicio es el componente que contiene la lógica del negocio | CrossCutting | Componente a desarrollar |
| Spring Framework 3.1.5 |
| Java 21 |
| model | Componente que representa los puntos de entrada como son los user interface de cada microservicio que se construirá dentro de la aplicación de forma genérica | componente que tiene como objetivo establecer la comunicación con los usuarios o con otros sistemas | CrossCutting | Componente a desarrollar |
| Spring Framework 3.1.5 |
| Java 21 |
| CrossCutting | Componente que ofrece las funciones transversales que pueden ser utilizadas por cualquier componente microservicio de forma general | Ofrece una gran cantidad de funciones, operaciones que pueden ser utilizados de forma general, transversal o común por cualquier otro componente desarrollado bajo la misma plataforma base (Java 21) con el objetivo de fomentar la reutilización | Java 21 | Frameworks/Aceleradores |
| Angular 12 | Componente usado para acelerar el desarrollo de una aplicación, respecto a aquellos aspectos que no hacen parte del core base o principal del negocio | Acelerar el desarrollo y concentrar los esfuerzos del equipo en tareas y aspectos netamente del negocio | Java 21 | Frameworks/Aceleradores |
| HTML 5 | Plataforma base de desarrollo bajo el lenguaje Java 21 | Es un lenguaje multiplataforma que por su gran capacidad, ha demostrado que puede ser utilizado para construir aplicaciones tanto livianas como empresariales, con excelentes aplicaciones de rendimiento y seguridad. | Java 21 | Frameworks/Aceleradores |
| ES12 | Plataforma base de desarrollo bajo el lenguaje Java 21 | Es un lenguaje multiplataforma que por su gran capacidad, ha demostrado que puede ser utilizado para construir aplicaciones tanto livianas como empresariales, con excelentes aplicaciones de rendimiento y seguridad. | Java 21 | Frameworks/Aceleradores |
| CSS 3 | Plataforma base de desarrollo bajo el lenguaje Java 21 | Es un lenguaje multiplataforma que por su gran capacidad, ha demostrado que puede ser utilizado para construir aplicaciones tanto livianas como empresariales, con excelentes aplicaciones de rendimiento y seguridad. | Java 21 | Frameworks/Aceleradores |

## Componente 2

Un Diagrama de componentes tiene como objetivo mostrar a las desarrolladores y las personas implicadas el cómo las diferentes piezas del sistema interactúan entre sí para este caso las piezas que tenemos en un Backend

## Diagrama Componentes Backend



## Documentación

| **Diagrama** | Componentes | | | |
| --- | --- | --- | --- | --- |
| **Componente** | Arquetipo general | | | |
| **Descripción** | Diagrama que muestra el 100% de los componentes que serán construidos o adaptados para proceder con la construcción del componente indicado. Adicionalmente, indica cómo estos componentes están relacionados | | | |
| **Componente** | **Descripción** | **Motivación** | **Depende/Usa** | **Tipo Componente** |
| infrastructure | Componente que representa la infraestructura de cada microservicio que se construirá dentro de la aplicación de forma genérica | componente que va a contener los elementos necesarios para que un microservicio funcione de manera eficiente y sea escalable. | CrossCutting | Componente a desarrollar |
| Spring Framework 3.1.5 |
| Java 21 |
| Application core | Componente que representa la parte de application core de cada microservicio que se construirá dentro de la aplicación de forma genérica | componente que va a contener la parte central del micro servicio es el componente que contiene la lógica del negocio | CrossCutting | Componente a desarrollar |
| Spring Framework 3.1.5 |
| Java 21 |
| user interface | Componente que representa los puntos de entrada como son los user interface de cada microservicio que se construirá dentro de la aplicación de forma genérica | componente que tiene como objetivo establecer la comunicación con los usuarios o con otros sistemas | CrossCutting | Componente a desarrollar |
| Spring Framework 3.1.5 |
| Java 21 |
| CrossCutting | Componente que ofrece las funciones transversales que pueden ser utilizadas por cualquier componente microservicio de forma general | Ofrece una gran cantidad de funciones, operaciones que pueden ser utilizados de forma general, transversal o común por cualquier otro componente desarrollado bajo la misma plataforma base (Java 21) con el objetivo de fomentar la reutilización | Java 21 | Frameworks/Aceleradores |
| Spring Framework 3.1.5 | Componente usado para acelerar el desarrollo de una aplicación, respecto a aquellos aspectos que no hacen parte del core base o principal del negocio | Acelerar el desarrollo y concentrar los esfuerzos del equipo en tareas y aspectos netamente del negocio | Java 21 | Frameworks/Aceleradores |
| Java 21 | Plataforma base de desarrollo bajo el lenguaje Java 21 | Es un lenguaje multiplataforma que por su gran capacidad, ha demostrado que puede ser utilizado para construir aplicaciones tanto livianas como empresariales, con excelentes aplicaciones de rendimiento y seguridad. | Java 21 | Frameworks/Aceleradores |

## Diagrama de paquetes

Un diagrama de paquetes es una herramienta visual la cual nos ayuda a organizar el sistema en diferentes grupos lógicos los cuales se le conocen como paquetes, donde nos muestran cómo estas partes del sistema se estructuran y se relacionan entre sí. La principal motivación a la hora de realizar el diagrama de paquetes es que nos facilita una mejor comprensión y una colaboración a la hora de dividir el sistema en partes manejables y lo más claras posibles.

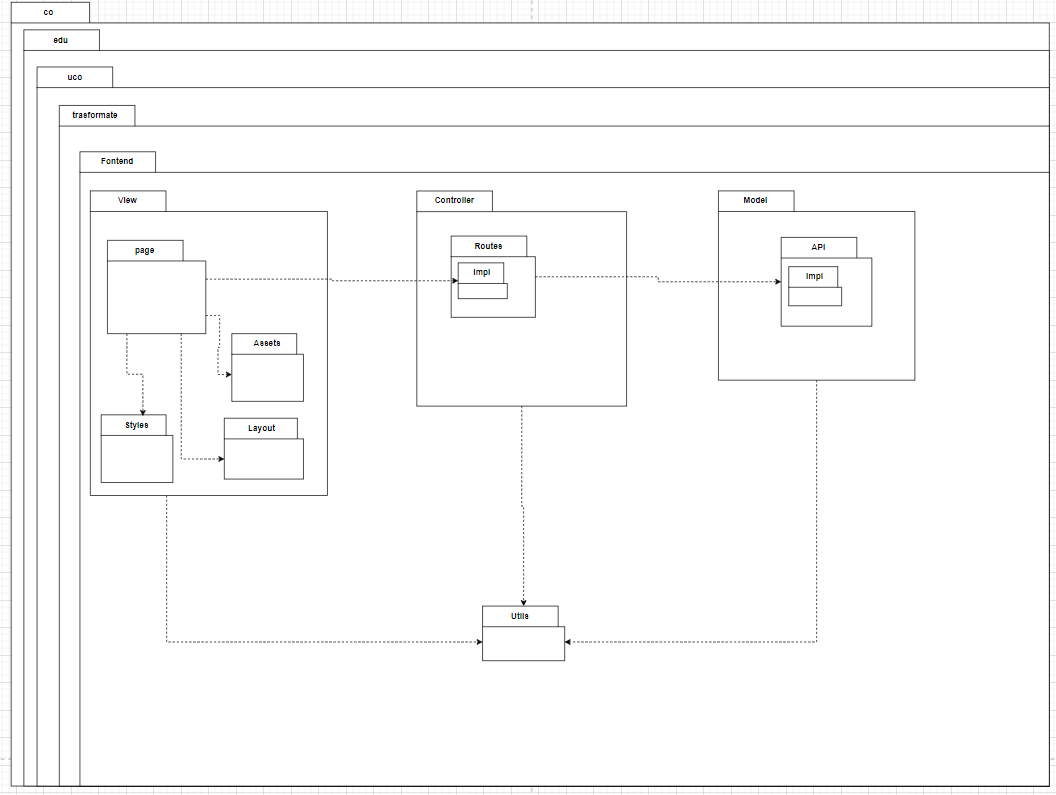
Los aportes más importantes que nos deja el diagrama de paquetes serian los siguientes:

* Organización
* Simplicidad
* Escalabilidad
* Claridad

## Componente 1 Front

El objetivo que tiene el diagrama de paquetes es mostrar cómo organizar y estructurar los elementos o componentes del sistema de manera ágil con la finalidad de que sea más fácil de entender

## Diagrama Front



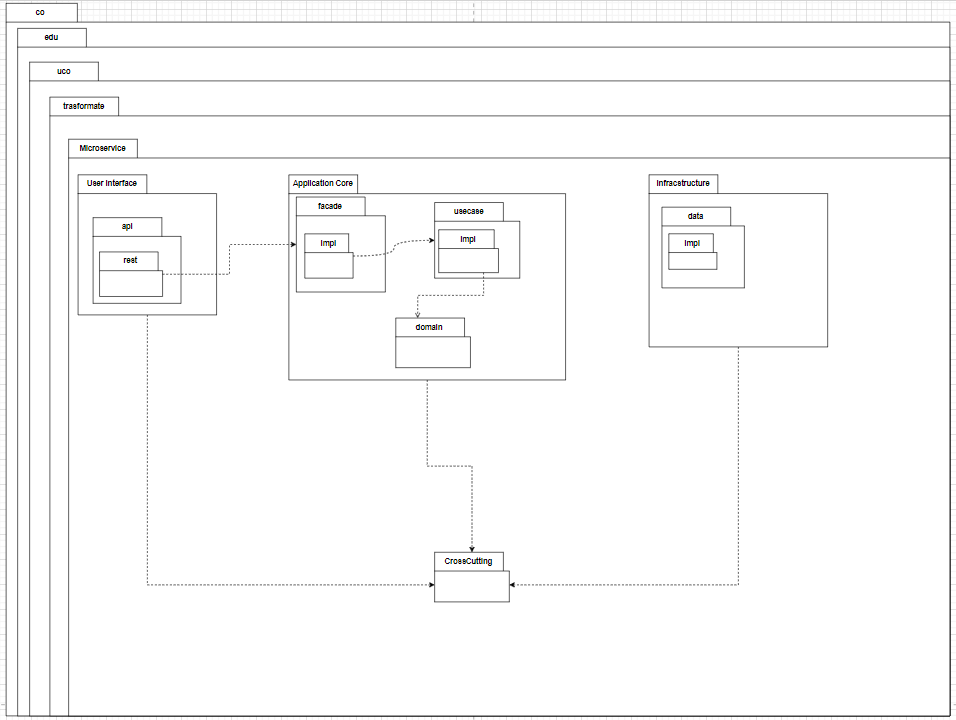
## Documentación

| **Diagrama** | Paquetes | | |
| --- | --- | --- | --- |
| **Componente** | Arquetipo general | | |
| **Descripción** | Diagrama que muestra la distribución de paquetes orientada a una estrategia por capas de una arquitectura limpia (Clean architecture) y la forma en que estos paquetes pueden usar las estructuras de los demás | | |
| **Paquete** | **Paquete Padre** | **Descripción** | **Usa/Importa** |
| co |  | Paquete raíz que indica o representa que la solución se desarrolla para el país Colombia |  |
| edu | co | Paquete que indica que la solución se desarrolla para una compañía del sector educativo |  |
| uco | edu | Paquete que indica que la solución se desarrolla para una compañía correspondiente a la universidad católica de oriente |  |
| trasformate | uco | paquete que indica que la solución es para el proyecto Transformate |  |
| frontend | trasformate | Paquete que indica que todos los paquetes que se encuentran dentro del frontend y conforman la solución |  |
| view | frontend | Paquete que agrupa todos los elementos de la capa view de una arquitectura limpia para la sección de interfaz de usuario |  |
| controller | frontend | Paquete que agrupa todos los elementos de la capa controller de una arquitectura limpia para la sección de interfaz de usuario |  |
| model | frontend | Paquete que agrupa todos los elementos de la capa model de una arquitectura limpia para la sección de interfaz de usuario |  |
| utils | frontend | Paquete que agrupa todos los elementos de la capa utils de una arquitectura limpia para la sección de interfaz de usuario |  |
| page | view | Paquete que agrupa todas las páginas correspondientes a la interfaz de usuario |  |
| assets | view | Paquete que agrupa todos los elementos o recursos que usa la interfaz de usuario |  |
| layout | view | Paquete que agrupa todos los componentes o archivos relacionados con el diseño y la estructura de la interfaz de usuario |  |
| styles | view | Paquete que agrupa todos los estilos que va usar las páginas de la interfaz de usuario |  |
| routes | controller | Paquete que agrupa todas las rutas en la capa de controller |  |
| impl | routes | Paquete que agrupa todas las implementaciones concretas de las rutas de la capa de controller |  |
| api | model | Paquete que agrupa todos todas las APi que sirven para la comunicación con las API del microservice |  |
| impl | api | Paquete que agrupa todas las implementaciones concretas de las api para la comunicación con los microservicios |  |

## Componente 2 back

El objetivo que tiene el diagrama de paquetes es mostrar cómo organizar y estructurar los elementos o componentes del sistema de manera ágil con la finalidad de que sea más fácil de entender

## Diagrama



## Documentación

| **Diagrama** | Paquetes | | |
| --- | --- | --- | --- |
| **Componente** | Arquetipo general | | |
| **Descripción** | Diagrama que muestra la distribución de paquetes orientada a una estrategia por capas de una arquitectura limpia (Clean architecture) y la forma en que estos paquetes pueden usar las estructuras de los demás | | |
| **Paquete** | **Paquete Padre** | **Descripción** | **Usa/Importa** |
| co |  | Paquete raíz que indica o representa que la solución se desarrolla para el país Colombia |  |
| edu | co | Paquete que indica que la solución se desarrolla para una compañía del sector educativo |  |
| uco | edu | Paquete que indica que la solución se desarrolla para una compañía correspondiente a la universidad católica de oriente |  |
| transformate | uco | Paquete que indica que todos los paquetes que se encuentran dentro del microservice y conforman la solución |  |
| microservice | transformate | Paquete que indica que todos los paquetes que se encuentran dentro de él conforman la solución de un dominio/contexto/aplicación denominada microservice |  |
| userinterface | microservice | Paquete que agrupa todos los elementos de la capa userInterface de una arquitectura limpia para la aplicación microservice |  |
| application core | microservice | Paquete que agrupa todos los elementos de la capa application core de una arquitectura limpia para la aplicación microservice |  |
| infrastructure | microservice | Paquete que agrupa todos los elementos de la capa infraestructura de una arquitectura limpia para la aplicación microservice |  |
| crosscutting | microservice | Paquete que agrupa todos los elementos de la capa de crosscutting o transversal de una arquitectura limpia para la aplicación microservice |  |
| api | userinterface | Paquete que agrupa todas las interfaces de programación de aplicación (api) de la aplicación de microservice |  |
| rest | api | Paquete que agrupa todas las api de tipo REST de la aplicación microservicio |  |
| facade | application core | Paquete que agrupa todos los contratos de fachada de la capa de aplicación core de la aplicación microservice |  |
| usecase | application core | Paquete que agrupa todos los contratos de los casos de uso de la application core de la aplicación microservice |  |
| domain | application core | Paquete que agrupa todos los dominios (entidades) de la capa application core de la aplicación microservice |  |
| impl | facade | Paquete que agrupa todas las implementaciones concretas de los contratos de fachada de la capa application core de la aplicación microservice | domain |
| impl | usecase | Paquete que agrupa todos los contratos de acceso a datos de la capa infrastructure de la aplicación microservice |  |
| data | infrastructure | Paquete que agrupa todas las implementaciones concretas de los contratos de acceso a datos de la capa infrastructure de la aplicación microservice |  |

## Vista de Procesos

es la representación visual de cómo se realizan las tareas y actividades dentro de un sistema, muestra las interacciones que tiene los componentes dentro del sistema

## Diagrama de secuencia

El diagrama de secuencia es una herramienta que nos va ayudar a representar la interacción entre objetos o componentes dentro de un sistema a lo largo del tiempo, en otras palabras son diferentes partes del sistema que se comunican entre sí y estos llevan un orden. Nuestra principal motivación con el diagrama de secuencia es que nos permite visualizar y entender cómo ocurren las interacciones entre los objetos como se mencionó anteriormente lo que nos permite ver un flujo entre las acciones que estos tienen.

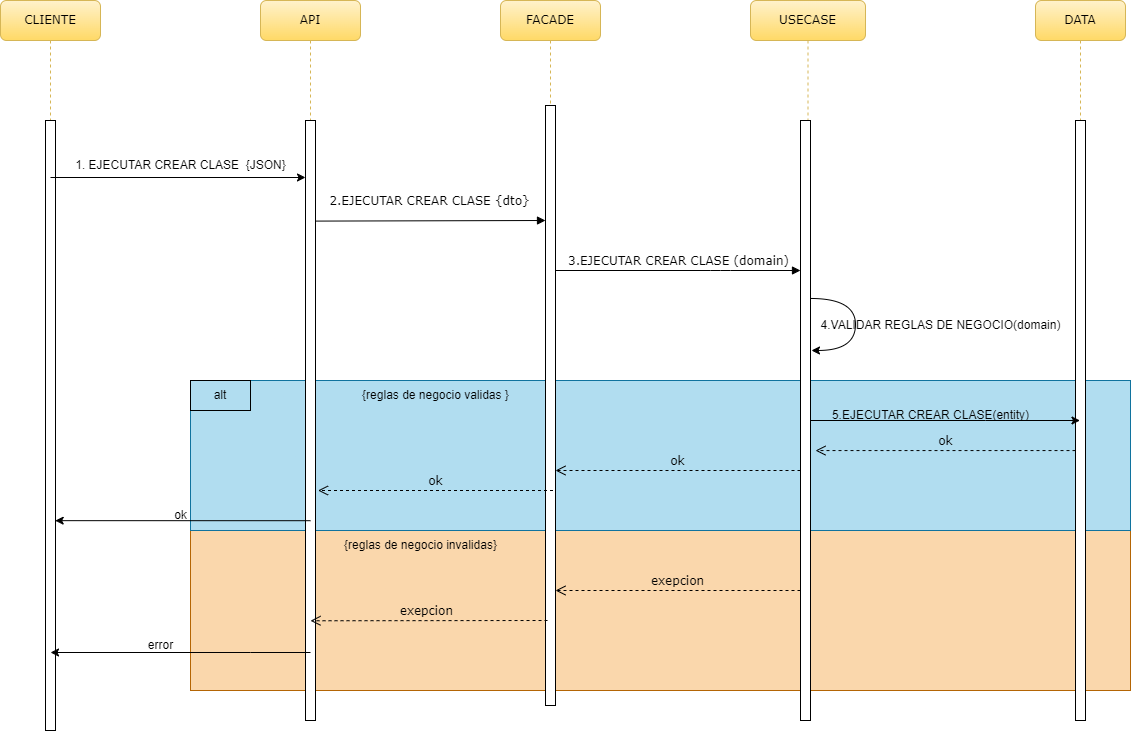
Algunos de los aportes que más se ven reflejados en el diagrama de secuencia serían los siguientes:

* Comunicación efectiva
* Optimización del diseño
* Claridad en la interacción
* Identificación de requisitos

## Componente 1

Este componente tiene el propósito de mostrar cómo se ejecuta el proceso de crear una clase

## Diagrama



## Documentación

| **Diagrama** | Componentes | | | |
| --- | --- | --- | --- | --- |
| **Transacción** | secuencia de una transacción general | | | |
| **Descripción** | modela una secuencia de una transacción general con el fin de poder mostrar cómo funciona la interacción entre las diferentes capas de la arquitectura definida | | | |
| líneas de tiempo | | | | |
| Nombre | Descripción | | | |
| Cliente | es el usuario final | | | |
| api | recibe la petición del usuario y llama al facade | | | |
| facade | crea el domaine de la petición | | | |
| use Case | se encarga de hacer validaciones y generar la respuesta | | | |
| data | es el encargado de la parte de persistencia de datos | | | |
| interacción | | | | |
| acción | origen | destino | Descripción |  |
| 1. EJECUTAR CREAR CLASE {JSON} | Cliente | api | el cliente solicita a la api llevar a cabo una acción determinada para lo cual le entrega los datos en formato JSON |  |
| CREAR CLASE {dto} | api | facade | la api delega la ejecución de la acción a la fachada correspondiente, entregando la información requerida en un objeto de tipo DTO |  |
| CREAR CLASE (domain) | facade | Use Case | la fachada delega la ejecución de la acción al caso de uso respectivo iniciando la transacción y entregando la información requerida en un objeto de tipo Dominio |  |
| validar Reglas de Negocio (Domain) | use CAse | Use Case | asegura el cumplimeminto las politicas o reglas de negocio de manera que la transacción sea consistente o valida |  |
| EJECUTAR CREAR CLASE(entity) | use CAse | Data | se ejecuta la actualización requerida contra la fuente de información a la cual se le entrega un objeto de tipo entidad. la acción se ejecuta cuando se ha garantizado el cumplimiento de las reglas de negocio . en caso contrario se detiene el proceso a raíz de las excepciones presentadas |  |

## Diagrama de colaboración

Un diagrama de colaboración muestra cómo las partes de un sistema se comunican y colaboran entre sí para poder que el sistema pueda funcionar correctamente. En otras palabras nos ayuda a entender las interacciones de manera visual y clara. Y de ahí viene la motivación más importante que ayuda a los desarrolladores o personas que por primer vez ven el diagrama visualizar las interacciones entre componentes y poder comprender cómo se comunican

Los principales aportes al diseño del sistema serían los siguientes

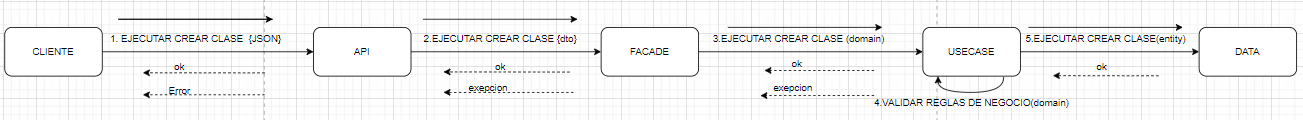
* Claridad en la interacción de los componentes
* Comprensión del Flujo de los componentes
* Optimización de diseño

## Componente 1

Este componente tiene el propósito de mostrar cómo se ejecuta el proceso de crear una clase

## Diagrama

ç



## Documentación

| **Diagrama** | Colaboración | | |
| --- | --- | --- | --- |
| **Transacción:** | Secuencia de una transacción general | | |
| **Descripción** | Modela una secuencia de una transacción general con el fin de mostrar cómo funciona la interacción entre las diferentes capas de la arquitectura definida | | |
| **Líneas de Tiempo** | | | |
|  | **Descripción** | | |
| **Nombre** |  | | |
| **Cliente** | Es el usuario final | | |
| **Api** | Recibe la petición del usuario y llama al facade | | |
| **Facade** | Crea el Domain de la aplicación | | |
| **UseCase** | Se encarga de hacer validaciones y generar la respuesta | | |
| **Data** | Es el encargado de la parte de persistencia de datos | | |
| **Interacción** | | | |
| **Acción** | **Origen** | **Destino** | **Descripción** |
| 1. ejecutarCrearClase(json) | Cliente | Api | El cliente solicita a la api llevar a cabo una acción determinada, para lo cual le entrega los datos en un formato JSON |
| 2. ejecutarCrearClase(dto) | Api | Facade | La api delega la ejecución a la fachada correspondiente entregando la información requerida en un objeto de tipo DTO |
| 3. ejecutar Crear Clase(domain) | Facade | UseCase | La fachada delega la ejecución al caso de uso respectivo, iniciando la transacción y entregando la información requerida en un objeto de tipo Dominio |
| 3. validar Reglas Negocio(domain) | UseCase | UseCase | Asegura el cumplimiento de todas las políticas o reglas de tal manera que la transacción sea consistente |
| 5.1 ejecutarCrearClase(entity) | UseCase | Data | Se ejecuta la actualización requerida contra la fuente de información a la cual se entrega un objeto tipo identidad. La acción solo se ejecuta cuando se ha garantizado el cumplimento de las reglas de negocio. En caso contrario, se detiene el proceso de a raíz de las excepción presentadas |

## Vista Física/Vista de Implantación

Tiene como propósito mostrar las conexiones de los componentes físicos o de hardware tanto en sistemas informáticos como de infraestructura.

## Diagrama de despliegue

Un diagrama de despliegue es una herramienta visual que se utiliza en la ingeniería de software, esta se usa más que todo para mostrar cómo los componentes de un sistema interactúan y se distribuyen en el entorno de ejecución. El propósito con el cual realizamos dicho diagrama es que nos permite mostrar la configuración física y lógica que tiene nuestro sistema

Algunos de los aportes de diseño que este diagrama le aporta al sistema serían los siguientes

* Visualizacion Clara
* Optimización de recursos
* Planificación de implementación
* Documentación

## Diagrama

