**Nombre del proyecto: GYM MANAGER**

**1 - Contexto del proyecto.**

El sistema será desarrollado usando la implementación del patrón de arquitectura de tres capas, en la cual los componentes serán ejecutados en las PC del gerente, recepcionista, departamento de registro, departamento de caja y departamento de recursos humanos respectivamente.

El sistema será desarrollado usando JAVA como lenguaje de programación para las diferentes capas de la aplicación, donde la capa de acceso a los datos utilizara la API JDBC para el acceso, modificación y borrado de los datos localizados en el depósito persistente de datos.

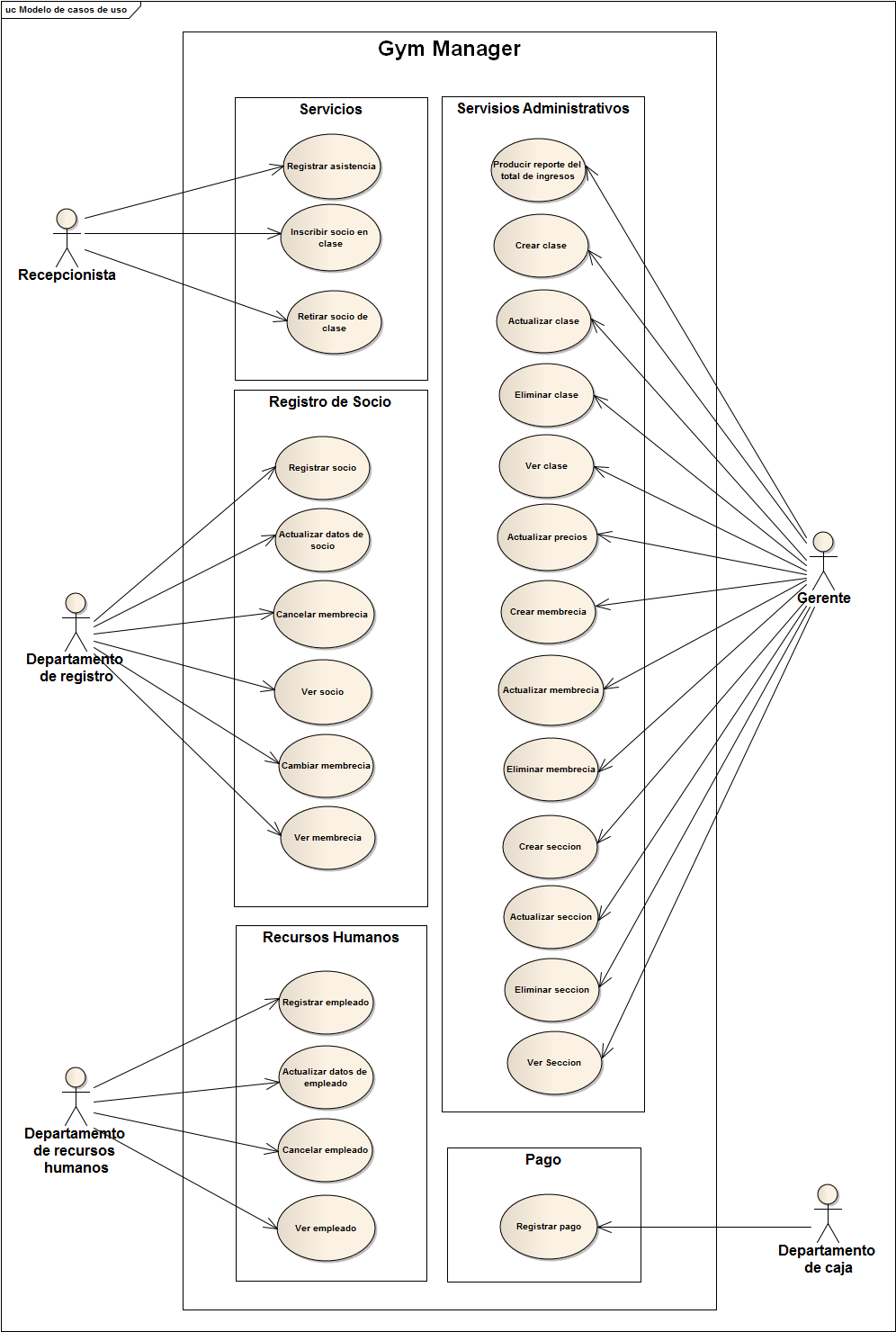
**2 - Requerimientos de la Arquitectura.**

**2.1 - Revisión de los objetivos claves.**

Este sistema será creado para proveer mecanismos automatizados para la gestión de los datos de los socios del gimnasio, así como su membresías y sus pagos realizados por cada ciclo y por sus diferentes métodos (registro, inscripción y pago por ciclo), membresía e inscripción, además de proveer el acceso, modificación y eliminación a estos datos a los diferentes empleados que tienen los privilegios necesarios para realizar estas operaciones.

**2.2 - Casos de uso de la Arquitectura.**

Los casos de uso del sistema que tienen funcionalidad central independiente del entorno de implementación se muestran en el diagrama de casos de uso presentado en la imagen de la página siguiente:

**Diagrama de Casos de Uso.**

**2.3 – Requerimientos arquitectónicos de las partes interesadas.**

El sistema deberá ofrecer las siguientes facilidades a los siguientes usuarios:

***Gerente***: deberá proveer al gerente las siguientes facilidades:

* Manejo eficiente de los datos concernientes a las membresías y sus diferentes tipos, así como la capacidad de consultar los datos de las clases y las secciones del gimnasio.
* Debe proveer facilidades para la modificación de los datos de las clases y sus secciones, además de los precios de inscripción del gimnasio, así como la facilidad de producción de reportes de los ingresos percibidos por el negocio.
* Debe permitir al Gerente la creación, modificación y eliminación de las membresías, las clases y las secciones.
* Una interfaz de aplicación amigable y de fácil entendimiento para un rápido y eficiente manejo del sistema.

***Recepcionista***: deberá proveer a la recepcionista las siguientes funcionalidades:

* Herramientas para la respuesta a las peticiones básicas requeridas por los clientes y socios del gimnasio tales como: ver asistencia del socio, registrar asistencia del mismo, inscripción y retiro de clases, así como la facilidad para la asignación y des asignación de instructores para entrenamiento personal.

***Departamento de caja***: deberá proveer a los empleados del departamento de caja las siguientes facilidades:

* Una rápida respuesta en la obtención y actualización de los datos relativos a las cuentas y deudas de los usuarios.

.

***Departamento de registro***: deberá proveer a los empleados del departamento de registro las siguientes facilidades:

* Los mecanismos para archivar y modificar los datos de los socios así como su membresía escogida.
* Cancelar las membresías de los socios del gimnasio en sus diferentes casos: petición de retiro voluntario por parte del socio o incumplimiento de los términos de permanencia de membrecía en el gimnasio.

***Departamento de recursos humanos***: deberá proveer a los empleados de recursos humanos las siguientes facilidades:

* Registro, actualización y cancelación de los empleados del gimnasio, así como la capacidad de consultar la información personal de los mismos, todo esto bajo una interfaz rápida y con facilidades para el acceso eficaz al depósito de datos.

**2.4 – Restricciones.**

1. El Gym Manager deberá correr bajo el sistema operativo Linux.

2. Las tablas de los datos persistentes deberán ser separadas en esquemas a nivel de la base de datos.

3. Los niveles de seguridad de la aplicación deberán ser controlador por medio de un inicio de sesión.

**2.5 – Requerimientos no funcionales.**

- Las respuestas de información deberán ser obtenidas en menos de 3 segundos desde la base de datos.

- El servidor de base de datos debe permitir el acceso concurrente al depósito de datos a los usuarios, sin que esto ocasione una pérdida de consistencia de los datos.

- Todas las peticiones realizadas a la base de datos serán hechas por medio de procedimientos almacenados en la base de datos, para lograr así que los resultados de las peticiones sean controlados por el RDBMS y no por la aplicación de manera exclusiva.

**2.6 – Riesgos.**

Los riesgos en el diseño de la aplicación a nivel de seguridad son mínimos ya que el acceso a la aplicación no está disponible por internet y no es necesaria una prevención exhaustiva contra los ataques exteriores, por lo que el entorno de la aplicación de manera global está libre de amenazas de acceso mal intencionado por la red.

Un factor importante es que la aplicación ha sido dividida en sub-sistemas a fin de mitigar el acceso por parte del personal que labora en el gimnasio a datos que no tiene autorización de ver y/o modificar.

Los mayores riesgos de la aplicación provienen de las posibles pérdidas de consistencia de la base de datos que pueden ser ocasionados por anomalías en el acceso concurrente a la información por los diferentes usuarios del sistema debido a que el control de lectura y escritura de los datos en la base de datos esta exclusivamente controlado por el RDBMS y solo él es responsable de proveer los datos a los usuarios y validar los datos incorrectos.

**3. Solución.**

**3.1 – Patrones Arquitectónicos Pertinentes.**

* Modelo de 3 Capas: Los diferentes tipos de usuarios solicitan servicios al sistema los datos son los clientes, los cuales hacen las peticiones a la capa de lógica del negocio ubicada en la capa intermedia, quien a su vez realiza las peticiones al depósito de datos usando la capa de acceso a los datos.
* Singleton: la capa de acceso a los datos está diseñada usando el patrón de diseño singleton para proveer una visibilidad global al objeto usado para la conexión con el depósito persistente de datos e impedir la instanciación de más de un objeto de acceso a datos.

**3.2 – Descripción de la arquitectura.**

La arquitectura de la aplicación Gym manager está centralizada sobre el patrón de 3 capas, siendo esta una aplicación de escritorio simple que interactúa con el depósito de datos usando el JDBC.

El sistema está subdivido en diferentes subsistemas, los cuales contienen los formularios necesarios para procesar las solicitudes de cada subsistema de manera separada.

Todos los módulos ejecutables instalados en las computadoras de los usuarios siguen internamente el patrón de 3 capas donde la última capa, la de acceso a datos envía y recibe peticiones a la base de datos usando como intermediario entre estos y la base de datos una capa de acceso a datos creada usando el patrón de diseño singleton.

**3.3 Vistas Estructurales.**

**Diagrama de Componentes.**

****

La descripción detallada de cada módulo ejecutable se encuentra estructurada en los diagramas de paquetes provistos en documentos separados a este, por lo cual queda fuera de alcance definir las partes de estos componentes ahora.

**Diagrama de despliegue**

**3.4 Vistas Dinámicas.**

**3.5. Cuestiones de su implementación.**

El enfoque de diseño seleccionado para la aplicación es el modelo de 3 capas, el cual fue seleccionado para satisfacer las necesidades de una independencia total entre la interfaz gráfica del usuario, el modelo de dominio de la lógica del negocio y el depósito persistente de los datos. Como plataforma de desarrollo fue seleccionada Java 2 Standard Edition ya que la aplicación no es muy grande en términos de desarrollo e implementación, además de que Java es independiente de la plataforma de Sistema Operativo lo que permite una independencia de la aplicación en términos de cambio de Sistema Operativo además de que existen muchas herramientas de código libre lo que reduce el costo de implementación de la aplicación.

Cabe mencionar que Java 2 Standard Edition soporta a la perfección el enfoque de 3 capas a utilizar y provee un fácil acceso a la base de datos a través de su controlador JDBC.

Para proveer un nivel de seguridad confiable sin agregar mucha complejidad en la codificación de la aplicación esta se subdivide en módulos separados para los distintos clientes que utilizan los servicios de la aplicación en forma de subsistemas compilados por separado pero que comparten la misma lógica del modelo de dominio previamente reseñado.

Cada subsistema posee una ventana de ingreso con protección nombre de usuario y contraseña, con lo cual se asegura la confidencialidad y la seguridad para usar la aplicación por usuarios indebidos y prevenir accesos malintencionados. Los datos de la ventana de ingreso se comprueban con los datos almacenados en el depósito persistente.

El depósito de datos es gestionado usando MySQL como gestor de base de datos DBMS, ya que tiene un costo de implementación razonable y puede correr en casi todos los sistemas operativos que se deseen utilizar en el servidor.

**4. Análisis de la arquitectura**

**4.1. Análisis de escenarios.**

* **Cambio del gestor de base de datos por un gestor de base de datos Oracle 10g o MS SQL Server 2008R2:** A menudo un cambio en el gestor de base de datos puede ocasionar cambios drásticos en la codificación de la aplicación en el lado de la capa de acceso a datos, en este caso una modificación al RDBMS de Oracle 10g o SQL Server 2008R2 solo implicaría una leve modificación en el driver de conexión a la base de datos usado en la capa de acceso a los datos así como la modificación de varios de los parámetros de conexión. Las consultas a la base de datos tendrían que ser re implementadas ya que la aplicación solo realiza peticiones a la base de datos a través de procedimientos almacenados.
* **Permitir el acceso a datos y las solicitudes de los socios y empleados a través de sitio Web:** Este es un cambio drástico para la arquitectura general de la aplicación ya que se necesitaría hacer un cambio en los servicios y protocolos que reciben las peticiones de los usuarios así como proveer un acceso controlados por protocolos de seguridad (SSL) a los clientes y personal del gimnasio a través de la internet. Esto se reflejaría en la creación de una interfaz Web para controlar las peticiones de los socios y la posibilidad de implementar un servidor de aplicaciones para desplegar los módulos ejecutables de la aplicación de escritorio inicial en un servidor Web de aplicaciones.
* **Apertura de nuevos locales del mismo gimnasio:**  la apertura de nuevos locales comerciales del mismo gimnasio provocaría que a nivel gerencial se necesite proveer al sistema de una conexión de área extensa WAN a los servidores y estaciones de trabajo además del re despliegue de las piezas de software en los posibles nuevos locales, además de la necesidad de crear un depósito central de los datos almacenados en todas las sucursales, y la necesidad de crear varios servidores de respaldo para asegurar que la información en todos los servidores este íntegra.

**4.2. Riesgos.**

Los siguientes riesgos pueden surgir a medida que el personal de trabajo y la carga de transacciones aumenten entre los diferentes módulos de la aplicación y la base de datos:

* **El intento para un cambio de aplicación de escritorio a aplicación Web sería muy costoso y complejo:** el cambio en la arquitectura de la aplicación a una interfaz Web necesitarían una inversión en equipos de capacidades considerables de almacenamiento y velocidad de procesamiento, lo cual termina siendo un riesgo económico muy grande a tomar por la empresa debido a su naturaleza de servicios centralizado en su localización.
* **Las respuestas a las necesidades de los usuarios no pueden ser negociadas:** ya que la pieza de software esta subdivida en módulos no existe ninguna posibilidad a los usuarios de un módulo en particular de obtener la funcionalidad de otro módulo de otra área, esto es debido a que se debe esconder a nivel de aplicación y a nivel de base de datos la información de modo que solo los usuarios con los privilegios necesarios puedan acceder a la información. Debido a que pueden existir planes futuros para un cambio a una interfaz Web, las peticiones de servicios deben ser negociadas por un servidor de aplicaciones lo que podría facilitar surgir en una amenaza de seguridad ante ataques externos; por tanto no debe ser posible negociar funcionalidades en la futura implementación sin los niveles de acceso apropiados.
* **El acceso concurrido a un dato en particular por parte de varias personas de la aplicación:** Uno de los riesgos que tienen que tienen que ser manejados a nivel de base de datos es el acceso concurrente a los datos, ya que al estar varias personas consultando o modificando una misma información pueden ocurrir estragos con la misma, lo cual causaría el almacenamiento de una información ineficiente.