DOCUMENTO DEL ARTÍCULO DE INVESTIGACIÓN DEL SISTEMA DE GYMRAT S.A.

**Autores:**

1. Ballesteros Holguín Diana Andrea
2. Bernardi Correa Walter Slaeter
3. González Astudillo Adrián Enrique
4. González Vera Héctor David
5. Ruiz Correa Noely Gisel

**Tabla de contenido**

[Resumen 3](#_Toc188893177)

[Abstract 4](#_Toc188893178)

[Introducción 5](#_Toc188893179)

[Objetivos 6](#_Toc188893180)

[Metodología 7](#_Toc188893181)

[Estimación Cocomo 7](#_Toc188893182)

[Cocomo Básico – proyecto orgánico 7](#_Toc188893183)

[Cocomo Intermedio – Proyecto Orgánico 8](#_Toc188893184)

[COCOMO II 12](#_Toc188893185)

[Herramientas para el desarrollo del sistema 15](#_Toc188893186)

[Metodología Scrum 17](#_Toc188893187)

[Resultados 21](#_Toc188893188)

[Conclusión 22](#_Toc188893189)

# Resumen

En la comunidad Gymrat, la falta de un sistema adecuado para la inscripción de clientes y la gestión de suscripciones ha generado procesos manuales, ineficiencias operativas y errores frecuentes. Estos problemas afectan la experiencia de los usuarios y dificultan la optimización de recursos del gimnasio, limitando su capacidad para crecer y competir en un mercado digitalizado.

Por lo tanto, se propone el desarrollo de un software que automatice y optimice la inscripción de clientes, la gestión de membresías, la coordinación de actividades mensuales y la facturación automática. Este sistema permitirá mejorar la eficiencia operativa, reducir errores y proporcionar una experiencia más satisfactoria para los usuarios.

Para alcanzar estos objetivos, se utilizarán los modelos COCOMO I y COCOMO II adaptados a las necesidades de Gymrat, con el fin de estimar los costos, el esfuerzo, el tiempo y los recursos necesarios para el proyecto. Asimismo, se empleará la metodología SCRUM para garantizar una colaboración activa entre los usuarios y el equipo de desarrollo, logrando resultados más realistas y efectivos.

El sistema integrará una base de datos diseñada en SQL Developer y será codificado en el IDE Microsoft Visual Studio utilizando el lenguaje C#. También se utilizarán herramientas como diagramas de clases, de componentes y el modelo de casos de uso para garantizar una organización óptima y el cumplimiento de los plazos establecidos.

**Palabras Clave:** Gymrat, Metodología SCRUM, COCOMO, Membresía, Software.

# Abstract

In the Gymrat community, the lack of an adequate system for customer enrollment and membership management has led to manual processes, operational inefficiencies, and frequent errors. These issues negatively impact user experience and hinder the gym’s ability to optimize resources, limiting its potential to grow and compete in a digitalized market.

To address this, the development of software is proposed to automate and optimize customer enrollment, membership management, monthly activity coordination, and automatic billing. This system aims to enhance operational efficiency, reduce errors, and deliver a more satisfying user experience.

To achieve these objectives, the COCOMO I and COCOMO II models, adapted to Gymrat’s needs, will be utilized to estimate project costs, effort, time, and required resources. Additionally, the SCRUM methodology will ensure active collaboration between users and the development team, leading to realistic and effective outcomes.

The system will integrate a database designed in SQL Developer and will be developed using the Microsoft Visual Studio IDE with the C# language. Tools such as class diagrams, component diagrams, and use case models will be employed to ensure optimal organization and adherence to the project’s deadlines.

**Keywords:** Gymrat S.A., SCRUM Methodology, COCOMO, membership , Software.

# Introducción

En el mundo donde la tecnología desempeña un papel fundamental en la vida cotidiana, las empresas y emprendimientos se ven impulsados a adaptarse y evolucionar constantemente. En este contexto, El emprendimiento gymrat ha decidido embarcarse en un emocionante proyecto: el desarrollo de un sistema que permita inscribir a los clientes de forma eficiente. Este sistema, concebido como una innovación clave, tiene como objetivo revolucionar la forma en que los clientes interactúan con el emprendimiento y adquieren suscripciones. Con un enfoque en la eficiencia y la satisfacción del cliente, gymrat planea ofrecer una amplia gama de servicios a sus usuarios, que incluyen la consulta de catálogos de suscripciones, gestión de suscripciones, logística de inscripción, servicio al cliente, entre otros.

El sistema permitirá registrar y asignar membresías para un gimnasio, además, contará con la información de las actividades que se realizaran por mes en el establecimiento. Una vez terminado el registro y asignación se facturará con los datos registrados anteriormente del cliente, la membresía correspondiente y las actividades que se realicen en el tiempo que este operante la membresía.

La atención al cliente será eficiente, ya que el sistema proporcionará un mecanismo para presentar planes que se realicen por mes además permitirá la asignación de una nueva membresía de un cliente ya registrado, cambiar la información de alguna actividad o membresía.

Un aspecto crítico de este proyecto es la integración con la base de datos de gymrat, desempeñará un papel fundamental en el ingreso y registro de los datos de los clientes. Además, tendrá la responsabilidad de coordinar la logística de las inscripciones. También indicara cuando se acaba la inscripción de los clientes con anticipación.

Por ende, este sistema interactuará de manera integral con la base de datos existente de gymrat, lo que simplificará el seguimiento de las inscripciones de clientes y garantizará una actualización precisa de los datos de los clientes a medida que se inscriban más clientes.

# Objetivos

1. Desarrollar un sistema que permita gestionar de manera eficiente la información de clientes y sus servicios contratados.
2. Mejorar la atención al cliente mediante funcionalidades que permitan presentar planes mensuales, asignar nuevas membresías y realizar modificaciones.
3. Garantizar una actualización precisa y oportuna de los datos de los clientes en la base de datos, integrando alertas sobre la finalización de inscripciones.

# Metodología

El desarrollo del sistema Gymrat se basa en una metodología estructurada que combina herramientas y enfoques modernos para garantizar la eficiencia y calidad del proyecto. Para la **estimación del esfuerzo, costos y tiempo**, se emplearán los modelos **COCOMO I y COCOMO II,** adaptados a las características específicas del proyecto, permitiendo una planificación precisa y realista. La implementación del sistema seguirá la **arquitectura de software MVC (Modelo-Vista-Controlador),** lo que asegura una separación clara entre la lógica de negocio, la interfaz de usuario y el acceso a los datos, facilitando la escalabilidad y el mantenimiento del sistema. El desarrollo se realizará en el **IDE Microsoft Visual Studio** utilizando el lenguaje de programación **C#**, mientras que la **base de datos relacional SQL Server** será utilizada para el almacenamiento seguro y eficiente de la información. Este enfoque garantiza un diseño modular y organizado, adecuado para satisfacer las necesidades del gimnasio y sus usuarios.

# Estimación Cocomo

## Cocomo Básico – proyecto orgánico

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Submodelos básicos** | **a** | **b** | **c** | **d** |
| Orgánico | 2,4 | 1,05 | 2,5 | 0,38 |
| Semi-acoplado | 3,0 | 1,12 | 2,5 | 0,35 |
| Empotrado | 3,6 | 1,20 | 2,5 | 0,32 |

* **SUBMODELOS DE COCOMO I**

**SIZE Estimado =** 10000 KLOC

* **ORGÁNICO**

**Esfuerzo** = =2.4 \* (10000)1.05 = 38 PM (Personas-Mes)

**DevTime** = = 2.5 \* (38)0.38 = 10 M (Meses)

=> **Personas**: 38/10 = 3.8 personas

=> **Productividad**: 10000/38 = 263.1 LOC/PM

* **SEMI-ACOPLADO**

**Esfuerzo** = =3.0 \* (10)1.12 = 39 PM (Personas-Mes)

**DevTime** = = 2.5 \* (39)0.35 = 9.01 M (Meses)

=> **Personas**: 39/9.01 = 4.32 personas

=> **Productividad**: 10000/39 = 256.4 LOC/PM

* **EMPOTRADO**

**Esfuerzo** = =3.6 \* (10)1.20 = 57 PM (Personas-Mes)

**DevTime** = = 2.5 \* (57)0.32 = 9.11 M (Meses)

=> **Personas**: 57/9.11= 6.25 personas

=> **Productividad**: 10000/57 = 175.4 LOC/PM

## Cocomo Intermedio – Proyecto Orgánico

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Submodelos intermedios** | **a** | **b** | **c** | **d** |
| Orgánico | 3,2 | 1,05 | 2,5 | 0,38 |
| Semi-acoplado | 3,0 | 1,12 | 2,5 | 0,35 |
| Empotrado | 2,8 | 1,20 | 2,5 | 0,32 |

**Tabla Factores de Costo**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Atributos | Valor | | | | | |
| Muy bajo | Bajo | Normal | Alto | Muy Alto | Extra alto |
| Atributos de Software | | | | | | |
| Fiabilidad | 0,75 | 0,88 | 1 | 1,15 | 1,4 |  |
| Tamaño de Base de datos |  | 0,94 | 1 | 1,08 | 1,16 |  |
| Complejidad | 0,7 | 0,85 | 1 | 1,15 | 1,3 | 1,65 |
| Atributos de hardware | | | | | | |
| Restricciones de tiempo de ejecución |  |  | 1 | 1,11 | 1,3 | 1,66 |
| Restricciones de memoria virtual |  |  | 1 | 1,06 | 1,21 | 1,56 |
| Volatilidad de la máquina virtual |  | 0,87 | 1 | 1,15 | 1,3 |  |
| Tiempo de respuesta |  | 0,87 | 1 | 1,15 | 1,3 |  |
| Atributo de personal | | | | | | |
| Capacidad de análisis | 1,46 | 1,19 | 1 | 0,86 | 0,71 |  |
| Experiencia en la aplicación | 1,29 | 1,13 | 1 | 0,91 | 0,82 |  |
| Calidad de los programadores | 1,42 | 1,17 | 1 | 0,86 | 0,7 |  |
| Experiencia en la máquina virtual | 1,21 | 1,1 | 1 | 0,9 |  |  |
| Experiencia en lenguaje | 1,14 | 1,07 | 1 | 0,95 |  |  |
| Atributos del proyecto | | | | | | |
| Técnicas actualizadas de programación | 1,24 | 1,1 | 1 | 0,91 | 0,82 |  |
| Utilización de herramientas de software | 1,24 | 1,1 | 1 | 0,91 | 0,83 |  |
| Restricciones de tiempo de desarrollo | 1,23 | 1,08 | 1 | 1,04 | 1,1 |  |

**Cálculo de SIZE**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Factores de Peso** | | | | |
| Factores Funcionales de Peso | Parámetros de Medida (1) | | | Contador (2) | Total Multiplicación (1) \*(2) |
| Simple | Media | Compleja |
| N.º Entrada de usuario | 7 | 10 | 15 | 20 | 200 |
| N.º Salida usuario | 5 | 7 | 10 | 10 | 70 |
| N.º Consulta usuario | 3 | 4 | 6 | 8 | 32 |
| N.º Archivos Lógicos Internos (tablas) | 4 | 5 | 7 | 4 | 20 |
| N.º Interfaces externas | 3 | 4 | 6 | 6 | 24 |
|  |  |  |  | | 346 |

**Cálculo fórmula LOC**

**Tabla de conversión de: Correlación código fuente a PF**

|  |  |
| --- | --- |
| Lenguaje | Correlación Código Fuente por PF (aprox) |
| Assembler | 320 |
| C | 128 |
| ALGOL | 105 |
| FORTRAN | 105 |
| PASCAL | 91 |
| RPG | 80 |
| PL/1 | 80 |
| Modula-2 | 80 |
| Prolog | 64 |
| LISP | 64 |
| BASIC | 64 |
| 4GL para BD | 40 |
| APL | 32 |
| Smaltallk | 29 |
| Query | 13 |
| Spreadsheet | 6 |
| Sql | 13 |
| VB | 24 |
| Java | 46 |
| Html | 14 |
| Delphi | 118 |
| C++ | 53 |
| COBOL | 107 |
| C# | 58 |

**Cálculo de la Variable FAE (multiplicador)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Atributos | Valor | | | | | |
| Muy bajo | Bajo | Normal | Alto | Muy Alto | Extra alto |
| Atributos de Software | | | | | | |
| Fiabilidad | 0,75 | 0,88 | 1 | 1,15 | 1,4 |  |
| Tamaño de Base de datos |  | 0,94 | 1 | 1,08 | 1,16 |  |
| Complejidad | 0,7 | 0,85 | 1 | 1,15 | 1,3 | 1,65 |
| Atributos de hardware | | | | | | |
| Restricciones de tiempo de ejecución |  |  | 1 | 1,11 | 1,3 | 1,66 |
| Restricciones de memoria virtual |  |  | 1 | 1,06 | 1,21 | 1,56 |
| Volatilidad de la máquina virtual |  | 0,87 | 1 | 1,15 | 1,3 |  |
| Tiempo de respuesta |  | 0,87 | 1 | 1,15 | 1,3 |  |
| Atributo de personal | | | | | | |
| Capacidad de análisis | 1,46 | 1,19 | 1 | 0,86 | 0,71 |  |
| Experiencia en la aplicación | 1,29 | 1,13 | 1 | 0,91 | 0,82 |  |
| Calidad de los programadores | 1,42 | 1,17 | 1 | 0,86 | 0,7 |  |
| Experiencia en la máquina virtual | 1,21 | 1,1 | 1 | 0,9 |  |  |
| Experiencia en lenguaje | 1,14 | 1,07 | 1 | 0,95 |  |  |
| Atributos del proyecto | | | | | | |
| Técnicas actualizadas de programación | 1,24 | 1,1 | 1 | 0,91 | 0,82 |  |
| Utilización de herramientas de software | 1,24 | 1,1 | 1 | 0,91 | 0,83 |  |
| Restricciones de tiempo de desarrollo | 1,23 | 1,08 | 1 | 1,04 | 1,1 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Submodelos intermedios | a | b | c | d |
| Orgánico | 3,2 | 1,05 | 2,5 | 0,38 |
| Semi-acoplado | 3,0 | 1,12 | 2,5 | 0,35 |
| Empotrado | 2,8 | 1,20 | 2,5 | 0,32 |

**ENTONCES SE TIENE:**

# COCOMO II

* **MODELO DE DISEÑO INICIAL**

**A =** 2.5

**SIZE Estimado =** 10 KLOC

**B =** 0.91 + 0.01 \* (Sum of rating on scaling factors for the project)

**Factores de Escala B:**

PREC = nominal (3.72)

FLEX = very high (1.01)

RESL = high (2.83)

TEAM = extra high (0.00)

PMAT = nominal (4.68)

**Multiplicadores M**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Cost Drivers** | **Extra Low** | **Very Low** | **Low** | **Nominal** | **High** | **Very High** | **Extra High** |
| **RCPX** | 0.73 | 0.81 | 0.98 | 1.0 | 1.30 | 1.74 | 2.38 |
| **RUSE** | - | - | 0.95 | 1.0 | 1.07 | 1.15 | 1.24 |
| **PDIF** | - | - | 0.87 | 1.0 | 1.29 | 1.81 | 2.61 |
| **PERS** | 2.12 | 1.62 | 1.26 | 1.0 | 0.83 | 0.63 | 0.50 |
| **PREX** | 1.59 | 1.33 | 1.12 | 1.0 | 0.87 | 0.71 | 0.62 |
| **FCIL** | 1.43 | 1.30 | 1.10 | 1.0 | 0.97 | 0.73 | 0.62 |
| **SCED** | - | 1.43 | 1.14 | 1.0 | 1.0 | 1.0 | - |

**Solución**

**Person Months**

**Person Months**

* **MODELO POST ARQUITECTURA**

**Cost Drivers:**

RELY = Low (0.88)

TIME = Very high (1.11)

ACAP = Very low (1.22)

PCAP = Very low (1.16)

TOOL = Low (1.12)

SCED = Very low (1.29)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Cost Drivers** | **Very Low** | **Low** | **Nominal** | **High** | **Very High** | **Extra High** |
| RELY | 0.75 | 0.88 | 1.00 | 2.48 | 1.24 | 0.00 |
| DATA |  | 0.93 | 1.00 | 2.03 | 2.03 | 0.00 |
| CPLX | 0.75 | 0.88 | 1.00 | 2.83 | 1.41 | 0.00 |
| RUSE |  | 0.91 | 1.00 | 2.19 | 1.10 | 0.00 |
| DOCU | 0.89 | 0.95 | 1.00 | 3.12 | 1.56 | 0.00 |
| TIME |  |  | 1.00 | 1.11 | 1.31 | 1.67 |
| STOR |  |  | 1.00 | 1.06 | 1.21 | 1.57 |
| PVOL |  | 0.87 | 1.00 | 1.15 | 1.30 |  |
| ACAP | 1.50 | 1.22 | 1.00 | 0.83 | 0.67 |  |
| PCAP | 1.37 | 1.16 | 1.00 | 0.87 | 0.74 |  |
| PCON | 1.24 | 1.10 | 1.00 | 0.92 | 0.84 |  |
| AEXP | 1.22 | 1.10 | 1.00 | 0.89 | 0.81 |  |
| PEXP | 1.25 | 1.12 | 1.00 | 0.88 | 0.81 |  |
| LTEX | 1.22 | 1.10 | 1.00 | 0.91 | 0.84 |  |
| TOOL | 1.24 | 1.12 | 1.00 | 0.86 | 0.72 |  |
| SITE | 1.25 | 1.10 | 1.00 | 0.82 | 0.84 | 0.78 |
| SCED | 1.29 | 1.10 | 1.00 | 1.00 | 1.00 |  |

**Solución**

# Herramientas para el desarrollo del sistema

***Visual Studio***

Visual Studio es un entorno de desarrollo integrado (IDE) creado por Microsoft. Proporciona un conjunto completo de herramientas y servicios para el desarrollo de software, incluyendo la escritura de código, la depuración, el diseño de interfaces de usuario y la administración de proyectos.

Visual Studio admite múltiples lenguajes de programación, como C#, C++, Visual Basic, F# y otros. Al utilizar Visual Studio en el proyecto, hemos obtenido varios beneficios y ventajas que han contribuido al éxito y eficiencia del desarrollo:

* Amplia compatibilidad de lenguajes: Visual Studio ofrece soporte para una amplia gama de lenguajes de programación, lo cual me permitió seleccionar el lenguaje más adecuado para el proyecto.
* Entorno de desarrollo integrado completo: Visual Studio proporciona un conjunto completo de herramientas y características que agilizaron mi flujo de trabajo. El entorno integrado incluye un editor de código altamente funcional, herramientas de depuración avanzadas y capacidades de prueba integradas. Esto nos permitió realizar todas las tareas de desarrollo desde un único lugar, lo que mejoró la productividad y la eficiencia general.
* Depuración y análisis avanzados: La funcionalidad de depuración y análisis de Visual Studio resultó invaluable durante el desarrollo del proyecto. Nos permitió identificar y solucionar rápidamente errores y problemas en el código, lo que aceleró el proceso de depuración y garantizó un software más estable y libre de errores.

***C#***

Como lenguaje de programación, escogimos C#, pues la sintaxis es clara, legible, y nos facilita la comprensión y escritura del código. Además, cuenta con un amplio conjunto de bibliotecas y frameworks, un entorno de desarrollo integrado poderoso y una comunidad activa de desarrolladores que brindan soporte y recursos adicionales. Al utilizar una estructura de código bien definida y fácilmente comprensible, nos reduce la posibilidad de cometer errores.

Además de su sintaxis, C# nos ofrece otras características:

* Amplio conjunto de bibliotecas y frameworks: C# cuenta con un vasto conjunto de bibliotecas y frameworks en la plataforma .NET, que proporcionan una funcionalidad predefinida y reutilizable para una amplia gama de tareas comunes en el desarrollo de software. Estas bibliotecas y frameworks permiten ahorrar tiempo y esfuerzo al aprovechar soluciones ya existentes en lugar de tener que desarrollar todo desde cero.
* Entorno de desarrollo integrado (IDE): Visual Studio es el entorno de desarrollo integrado (IDE) principal para el desarrollo en C#. Proporciona herramientas poderosas que facilitan la escritura, depuración y prueba del código. La interfaz intuitiva de Visual Studio, junto con las numerosas características de productividad, como la finalización automática de código, la refactorización y la depuración visual, contribuyen a una experiencia de programación más eficiente y cómoda.

# Metodología Scrum

**Backlog del Producto:** Se creó un backlog detallado con todos los requerimientos y funcionalidades del sistema, teniendo en cuenta los módulos clave como:

* + **Módulo de Creación de Clientes:** Permitir almacenar y gestionar la información de los clientes, con capacidad para editar, eliminar y generar reportes.
  + **Módulo de Membresías:** Crear, activar, desactivar y eliminar membresías, junto con la generación de reportes.
  + **Módulo de Gestión de Actividades:** Gestionar la información de los servicios y equipos disponibles en el gimnasio, y permitir asociar membresías a los servicios.
  + **Módulo de Facturación:** Registrar pagos de membresías, generar facturas, eliminar o restaurar facturas y generar reportes de facturación.

**Stakeholders:** Identificamos a los principales interesados, tales como los administradores del gimnasio, personal de ventas y gerentes de operaciones y nos aseguramos de alinear las expectativas y necesidades de cada uno de los roles.

**Equipo Scrum:** El equipo se conformó de la siguiente manera:

* + **Product Owner:** El gerente del gimnasio representó las necesidades de los usuarios y priorizó las funcionalidades del sistema.
  + **Scrum Master:** Un miembro del equipo de desarrollo asumió este rol, facilitando el cumplimiento de las prácticas ágiles y solucionando cualquier impedimento.
  + **Equipo de Desarrollo:** Se compuso de desarrolladores encargados de crear el sistema, utilizando Microsoft Visual Studio como entorno de desarrollo integrado (IDE) y SQL Server para gestionar las bases de datos.

**Sprint 0 (Preparación Inicial):**

* **Configuración del Entorno:** Usamos Microsoft Visual Studio para el desarrollo del sistema, configurando las aplicaciones de escritorio y las interfaces de usuario. La base de datos fue configurada en SQL Server, creando las tablas necesarias para clientes, membresías, actividades y facturación.
* **Arquitectura del Sistema:** Se definió la estructura de la base de datos en SQL Server, creando relaciones entre las tablas para asegurar la correcta integración del sistema. Además, diseñamos las pantallas iniciales en Visual Studio como las interfaces para crear clientes y gestionar membresías.
* **Historias de Usuario:** Se elaboraron historias de usuario iniciales como: "Como administrador, quiero agregar nuevos clientes para gestionar sus membresías" y "Como administrador, quiero generar reportes de clientes para visualizar estadísticas de asistencia y pagos".
* **Planificación y Estimación:** Estimamos las horas necesarias para completar la configuración inicial del entorno, así como las funcionalidades básicas de creación de clientes y generación de membresías.

**Sprint 1 (Desarrollo de Funcionalidades Iniciales):**

* **Reuniones Diarias:** Se realizaron reuniones diarias para evaluar el progreso y abordar cualquier inconveniente. Cada miembro del equipo reportó su avance y se resolvieron obstáculos rápidamente.
* **Desarrollo Prioritario:** El enfoque principal del primer sprint fue el módulo de creación de clientes y la gestión de membresías. A su vez, se implementaron las funcionalidades para crear y asignar membresías.
* **Revisión del Sprint:** Al final del sprint, se presentó el avance al Product Owner y stakeholders, quienes dieron feedback sobre las funcionalidades implementadas, lo que permitió hacer ajustes para los siguientes sprints.
* **Retrospectiva:** El equipo discutió lo que había ido bien y lo que se podía mejorar. Se establecieron acciones para optimizar el proceso en el siguiente sprint.

**Sprints Subsiguientes (Desarrollo Iterativo y Mejora Continua):**

* **Desarrollo Incremental:** En cada sprint, continuamos implementando nuevas funcionalidades, como la **gestión de actividades** en el gimnasio, la **facturación de membresías** y la capacidad de generar **reportes financieros**. Cada módulo fue desarrollado y probado.
* **Refinamiento del Backlog:** A medida que avanzábamos, se ajustaron las prioridades en el backlog para reflejar cambios en las necesidades del gimnasio, como la necesidad de funciones adicionales en los reportes de clientes y facturación.
* **Revisión y Feedback Constante:** Tras cada sprint, organizamos reuniones con los stakeholders para revisar el progreso y asegurarnos de que el sistema estaba alineado con los objetivos del negocio. Esto permitió ajustes rápidos y eficaces.

**Entrega del Producto:**

* **Revisión Final:** Al concluir los sprints planificados, se realizó una revisión completa del sistema, asegurando que todas las funcionalidades, como la creación de clientes, gestión de membresías, actividades y facturación, estuvieran implementadas correctamente.
* **Pruebas de Calidad:** Realizamos pruebas exhaustivas de todas las funcionalidades, tanto en la interfaz de usuario como en la base de datos para verificar que todos los datos estuvieran siendo gestionados correctamente.
* **Entrega Formal:** El sistema fue entregado al cliente final (administrador del gimnasio), quien fue capacitado en el uso del sistema para gestionar todas las operaciones del gimnasio de manera autónoma.

**Mantenimiento y Mejora Continua:**

* **Soporte Continuo:** Tras la entrega, se proporcionó soporte técnico para resolver problemas que surgieran en el uso del sistema y para asegurar el buen funcionamiento de la base de datos y las interfaces.
* **Retroalimentación:** Recogimos las opiniones de los usuarios finales para mejorar futuras iteraciones del sistema. Se identificaron áreas de mejora, como la creación de reportes más detallados y la optimización de la interfaz de usuario.
* **Lecciones Aprendidas:** Después de la entrega final, el equipo Scrum discutió las lecciones aprendidas durante todo el proceso, lo que nos permitió aplicar mejoras tanto en la metodología Scrum como en el desarrollo del sistema para futuros proyectos.

**Distribución de roles entre los integrantes**

|  |  |  |
| --- | --- | --- |
| **Nombres** | **Contacto** | **Rol** |
| Ballesteros Holguín Diana Andrea | [diana.ballesteroshol@ug.edu.ec](mailto:diana.ballesteroshol@ug.edu.ec) | Líder de diseño arquitectónico |
| Ruiz Correa Noelly Gisel | [noelly.ruizcor@ug.edu.ec](mailto:noelly.ruizcor@ug.edu.ec) | Líder de gestión y mejora |
| González Astudillo Adrián Enrique | [adrian.gonzalezast@ug.edu.ec](mailto:adrian.gonzalezast@ug.edu.ec) | Líder de diseño |
| González Vera Héctor David | [hector.gonzalezve@ug.edu.ec](mailto:hector.gonzalezve@ug.edu.ec) | Líder de requerimientos |
| Bernardi Correa Walter Slaeter | [walter.bernardic@ug.edu.ec](mailto:walter.bernardic@ug.edu.ec) | Líder de pruebas y Calidad |

# Resultados

El proyecto del sistema Gymrat tiene como resultado un sistema de gestión completo que facilita la administración eficiente de todas las operaciones dentro de un gimnasio. Está diseñado para optimizar la gestión de clientes, membresías, actividades y facturación, proporcionando herramientas que agilicen los procesos internos. El sistema no solo mejora la experiencia de los clientes, sino que también incrementa la eficiencia del personal administrativo, promoviendo una mayor productividad y ofreciendo información clave para la toma de decisiones informadas y basadas en datos.

**Objetivos y Funcionalidades del Sistema:**

• **Gestión eficiente de clientes:** El sistema permite almacenar, editar y eliminar la información de los clientes, facilitando la administración de sus datos de manera organizada. Además, genera reportes detallados que brindan una visión clara sobre los clientes registrados, optimizando la toma de decisiones.

• **Administración de membresías:** Facilita la creación de membresías personalizadas para cada cliente registrado. Permite activar, desactivar o eliminar membresías según sea necesario, asegurando que el acceso a los servicios esté siempre actualizado. Genera reportes sobre el estado de las membresías, mejorando el control administrativo.

• **Control de actividades y servicios:** Registra y organiza la información de las máquinas y servicios disponibles en el gimnasio. Vincula las membresías con los servicios contratados, garantizando que los clientes accedan únicamente a los servicios por los que han pagado.

• **Gestión integral de facturación:** Almacena los pagos de las membresías en formato de factura, asegurando claridad y transparencia en las transacciones. Incluye la posibilidad de eliminar o restaurar facturas según sea necesario, y genera reportes detallados que permiten un seguimiento preciso de los ingresos.

• **Optimización del flujo de trabajo:** Diseñado para ofrecer una interfaz intuitiva y fácil de usar, el sistema facilita la operación por parte del personal administrativo, reduciendo tiempos de gestión y mejorando la atención a los clientes.

# Conclusión

El desarrollo e implementación del sistema Gymrat ha permitido crear una solución integral y eficiente para la gestión de operaciones dentro de un gimnasio. A lo largo del proceso, desde la planificación hasta la entrega final, se aplicó una metodología ágil basada en Scrum que facilitó la adaptación continua a las necesidades cambiantes y las prioridades del cliente.

A través de módulos clave como la gestión de clientes, membresías, actividades y facturación, el sistema ha logrado optimizar y automatizar tareas críticas, mejorando la productividad tanto del personal administrativo como de los clientes.

El enfoque iterativo permitió una constante retroalimentación, lo que favoreció la mejora continua del producto y la satisfacción del usuario final. Los reportes generados, la facilidad para gestionar membresías y la automatización de facturación son solo algunas de las ventajas que ofrece al gimnasio para mantener un control más efectivo y eficiente de sus operaciones diarias.

En resumen, Gymrat no solo cumple con los requisitos planteados, sino que establece una base sólida para el futuro, con posibilidades de expansión de funcionalidades y mejoras continuas que permitirán seguir adaptándose al crecimiento y las demandas del negocio. Este proyecto ha sido un ejemplo de cómo una metodología ágil, herramientas adecuadas y una buena comunicación entre todos los actores involucrados pueden dar lugar a un sistema exitoso, alineado con los objetivos del cliente y con un alto impacto en la eficiencia operativa.