# CAPÍTULO IV: ANÁLISIS PARA EL DESARROLLO DEL SOFTWARE



## requerimientos funcionales

* + 1. El sistema debe permitir que los usuarios se autentiquen utilizando credenciales válidas (usuario y contraseña).
    2. El sistema debe permitir registrar, actualizar, eliminar y consultar la información de los clientes. Esto incluye datos como nombre, dirección, número de teléfono, fecha de pago, etc.
    3. El sistema debe almacenar un historial de pagos por cliente, permitiendo visualizar los pagos realizados y las fechas de vencimiento.
    4. El sistema permitirá la consulta del estado de cuenta del cliente, mostrando las mensualidades pendientes, pagadas y próximas a vencer.
    5. El sistema debe permitir registrar y gestionar información de los bancos donde se reciben los pagos, incluyendo detalles como nombre del banco, número de cuenta, y tipo de cuenta.
    6. El sistema debe permitir registrar pagos de mensualidades.
    7. El sistema tiene que generar reportes detallados de todos los pagos recibidos en un mes, con la capacidad de filtrar por cliente, fecha, banco, y método de pago.



## requerimientos no funcionales

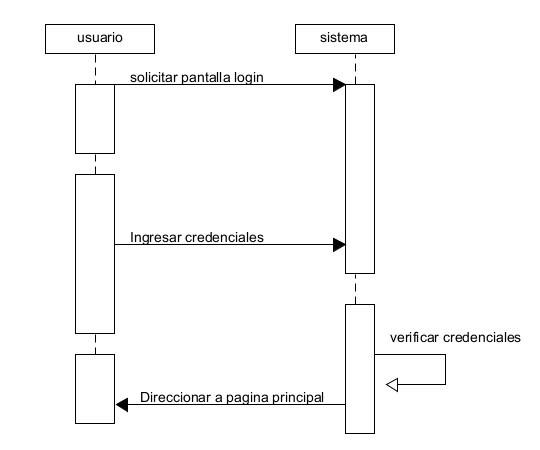
* + 1. El sistema debe ser capaz de asegurar que solo los usuarios autorizados puedan acceder a módulos y funciones específicas según su rol.
    2. El sistema permitirá registrar todos los accesos, cambios en la información de clientes, y transacciones de pago para fines de auditoría.
    3. El sistema debe ser intuitivo y fácil de usar, con una navegación clara y accesible para todos los tipos de usuarios.
    4. Las operaciones, especialmente las relacionadas con la generación de reportes y la consulta de pagos, deben realizarse en un tiempo aceptable.
    5. El sistema debe estar diseñado de manera modular para facilitar la adición de nuevas funciones y el mantenimiento general.

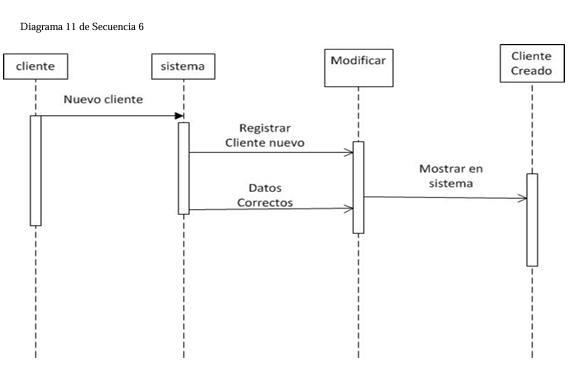


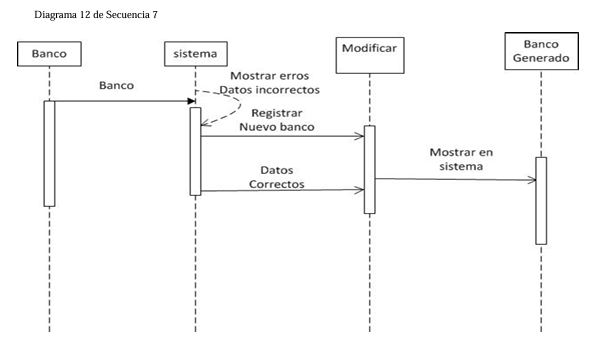
## Diagramas

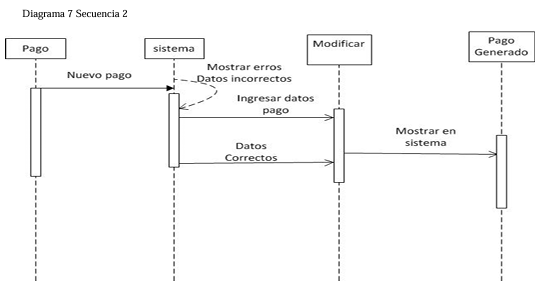


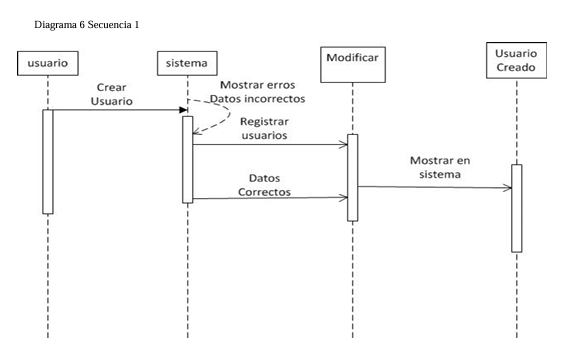
## Diagrama de secuencia







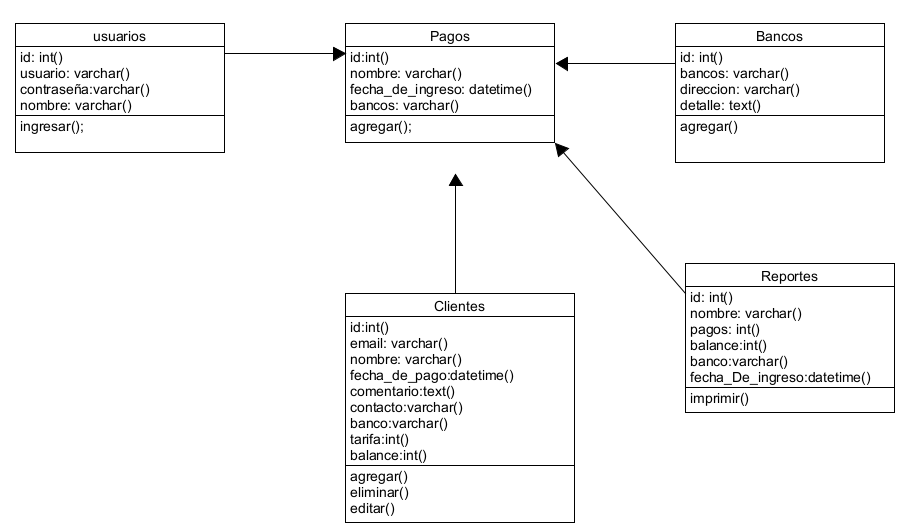






## Diagrama de Clases

Un diagrama de clases es una herramienta para comunicar el diseño de un programa que se creó para orientar objetos y que permite modelar relaciones entre diferentes entidades. En UML, una clase se representa con un rectángulo que posee tres divisiones, nombre de la clase, atributos que tiene y mensajes que entiende. (okdiario,2018)

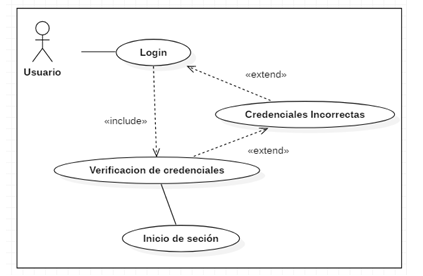




## Diagrama de Casos de Usos

**Tabla 1:** Descripción de actores en casos de uso.

|  |  |
| --- | --- |
| **Actor en el diagrama** | **Rol dentro de la empresa Jireh** |
| Usuario | administrador |
| Supervisor | Gerente General |
| Encargado | Encargados de Proyectos |

****

**Figura 1:** Diagrama caso de uso ingresar al sistema.

Fuente:Elaboración propia.

**Tabla 2:** Descripción caso de uso modulo cliente.

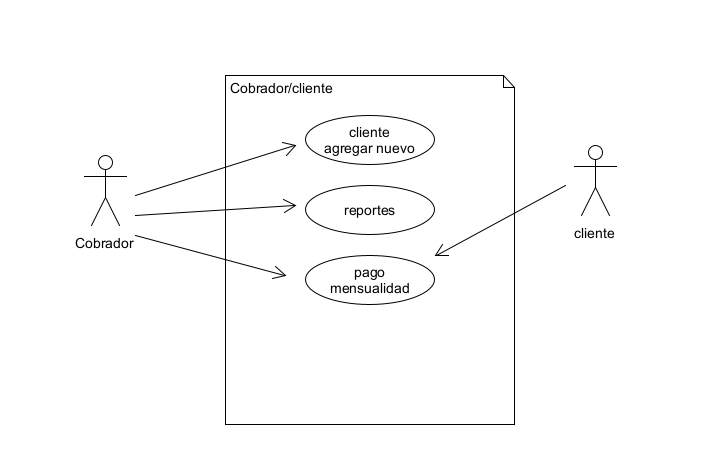
|  |  |
| --- | --- |
| **Caso de Uso** | Agregar cliente |
| **Actores** | Cobrador |
| **Tipo** | Primario |
| **Descripción** | Permite el ingreso de un nuevo cliente en el sistema. |
| **Precondición** | El usuario debe haber ingresado al sistema como administrador. |
| **Postcondición** | El sistema registra el nuevo cliente en la base de datos y muestra las opciones vinculadas al mismo. |
| **Flujo Normal** | 1. El usuario visitará el módulo para gestionar. 2. Presionará el botón agregar. 3. El usuario ingresará los datos solicitados por el formulario. 4. El sistema validará que los datos estén llenados de forma correcta. (FA1). 5. El sistema registrará el nuevo cliente en la base de datos. 6. El sistema mostrará un mensaje indicando el éxito de la operación. |
| **Flujo Alterno** | **FA1:** Existen errores en los datos ingresados.   1. El sistema mostrará en el formulario cuáles son datos ingresados que contienen errores. 2. Se retorna al usuario al paso 3. |

**Tabla 3:** Descripción caso de uso modulo pagos.

|  |  |
| --- | --- |
| **Caso de Uso** | Pago |
| **Actores** | Cobrador |
| **Tipo** | Primario |
| **Descripción** | Permite ver el pago de un cliente en el sistema. |
| **Precondición** | El usuario debe haber ingresado al sistema como administrador. |
| **Postcondición** | El sistema registra el pago en la base de datos y muestra las opciones vinculadas al mismo. |
| **Flujo Normal** | 1. El usuario visitará el módulo para gestionar. 2. Presionará el modulo pagos. 3. El usuario ingresará los datos solicitados por el formulario. 4. El sistema validará que los datos estén llenados de forma correcta. (FA1). 5. El sistema mostrara el pago del cliente en la base de datos. |
| **Flujo Alterno** | **FA1:** Existen errores en los datos ingresados.   1. El sistema no mostrar los clientes con pagos. |

**Tabla 4:** Descripción caso de uso modulo reportes.

|  |  |
| --- | --- |
| **Caso de Uso** | Reportes |
| **Actores** | Cobrador |
| **Tipo** | Primario |
| **Descripción** | Mostrará los clientes que estén solvente en el mes. |
| **Precondición** | El usuario debe haber ingresado al sistema como administrador. |
| **Flujo Normal** | 1. El usuario visitará el módulo para gestionar. 2. Presionará el modulo reportes. 3. El sistema permitirá la visualización de reportes. |

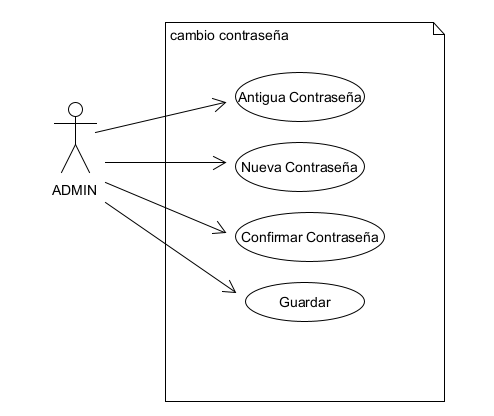


**Figura 2:** Diagrama caso de uso modulo cliente, pago, reportes

Fuente:Elaboración propia.

**Tabla 5:** Descripción caso de uso restablecer contraseña del usuario.

|  |  |
| --- | --- |
| **Caso de Uso** | Restablecer contraseña del usuario |
| **Actores** | Administrador |
| **Tipo** | Primario |
| **Descripción** | Permite cambiar la contraseña de un usuario. |
| **Precondición** | El usuario debe haber ingresado al sistema como administrador. |
| **Postcondición** | La contraseña del usuario se actualizará. |
| **Flujo Normal** | 1. El administrador visitará el módulo para gestionar configuraciones. 2. Presionará sobre la opción correspondiente al restablecimiento de contraseña para el usuario objetivo. 3. El sistema actualizará la contraseña del usuario. |
| **Flujo Alterno** | **FA1:** No se coloca ninguna contraseña.   1. No se actualiza la contraseña del usuario. 2. Fin del caso de uso. |



**Figura 3:** Diagrama caso de uso módulo configuración.

Fuente:Elaboración propia.



## Análisis de la Base de datos

MySQL es un sistema de gestión de bases de datos relacional (RDBMS) ha desempeñado un papel fundamental en la evolución de la tecnología de la información y ha sido un pilar esencial para aplicaciones web, empresas y proyectos de todo tipo. (García, 2024)

MySQL utiliza un modelo de base de datos relacional, donde los datos se organizan en tablas con relaciones definidas. Utiliza el lenguaje SQL para realizar consultas y manipular datos. Sus componentes principales incluyen un servidor de base de datos, motores de almacenamiento y clientes que permiten la interacción con la base de datos. (García, 2024)

El funcionamiento de MySQL se basa en almacenar los datos en el sistema de archivos del servidor. Cuando un programa del ordenador necesita acceder a los datos, envía una consulta SQL al servidor MySQL. El servidor MySQL procesa esta consulta y devuelve los resultados al programa. (García, 2024)