Documento Diagrama De UML

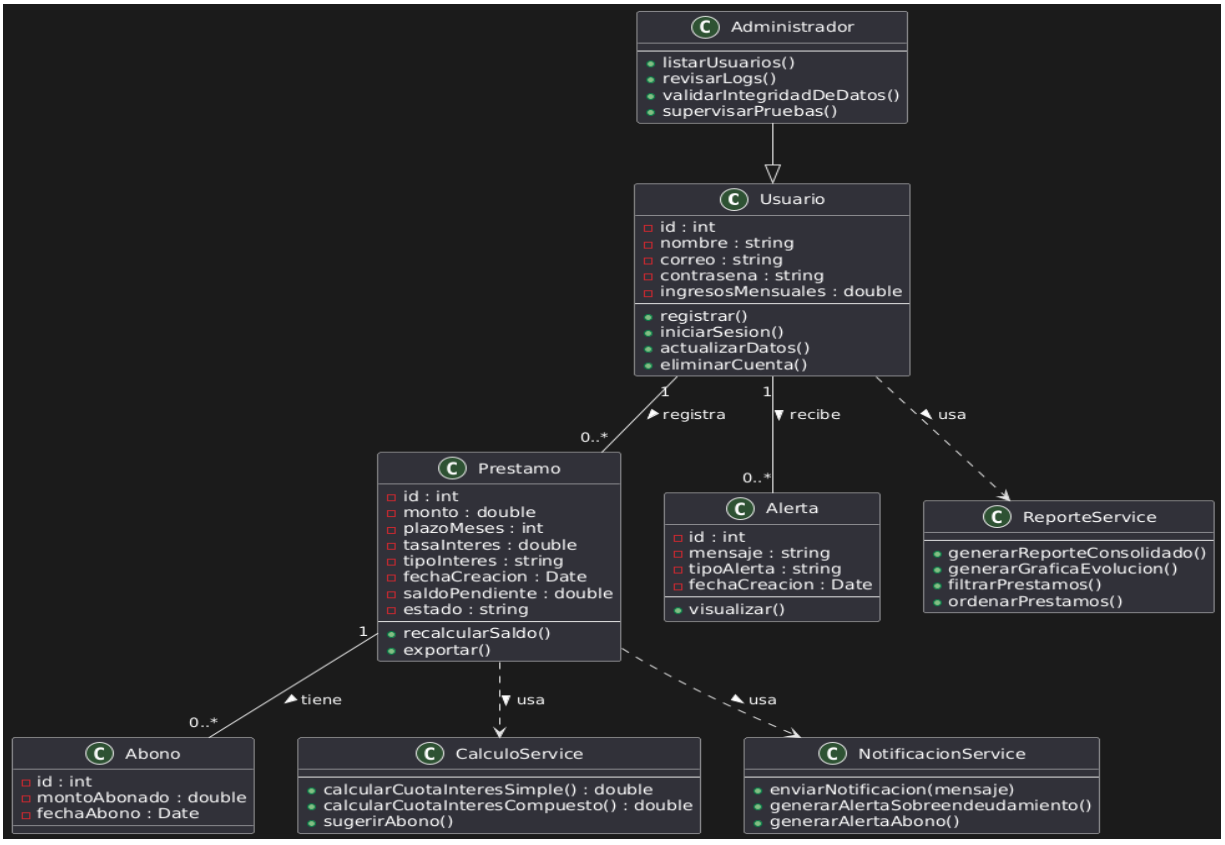
Proyecto: OptiCash - Prototipo de Sistema de Gestión de Préstamos
Versión: 1.0
Fecha: 31 de agosto de 2025
Autor: Equipo de Desarrollo

---

1 Introducción

Este documento de diseño de software establece la traza entre el **documento de requerimientos** (el "qué" se va a construir) y el **diagrama de clases UML** (el "cómo" se va a estructurar el sistema). Su propósito es servir como una guía técnica para el equipo de desarrollo, explicando el rol de cada clase y servicio en la arquitectura. La metodología de diseño consistió en:

1. **Identificar actores y entidades:** Se definieron los actores principales del sistema (Usuario, Administrador, Sistema) y las entidades clave del negocio (Préstamo, Abono, Alerta) a partir de los requerimientos funcionales.
2. **Agrupar funcionalidades:** Se agruparon las funcionalidades relacionadas en servicios independientes (CalculoService, NotificacionService), reflejando el enfoque de **microservicios** del proyecto.
3. **Establecer relaciones:** Se modelaron las relaciones entre las clases (herencia, asociación, dependencia) para asegurar la coherencia y consistencia de los datos, tal como lo requiere el proyecto



.

2 Mapeo Detallado del Diagrama de Clases

2.1 Clases de Entidad

Las clases de entidad representan los datos principales del sistema y su lógica.

- **Clase Usuario**
    - **Propósito:** Representa a la persona que usa la aplicación. Es la base de todos los datos personales y de acceso.
    - **Atributos:**
        - **id, nombre, correo, contraseña**: Permiten el registro de usuarios, el inicio de sesión y la actualización de datos. (RF1, RF2, RF3)
        - **ingresosMensuales:** Fundamental para el requerimiento de alertas de sobreendeudamiento. (RF16)
- **Clase Administrador**
    - **Propósito:** Encargado de la supervisión y mantenimiento del sistema a nivel académico.
    - **Relación:** El Administrador hereda de Usuario, lo que le permite compartir la funcionalidad de autenticación y gestionar usuarios.
    - **Métodos:**
        - **listarUsuarios():** Cumple con el requerimiento de supervisión del administrador.
        - **revisarLogs(), validarIntegridadDeDatos(), supervisarPruebas**(): Estos métodos reflejan la interacción académica del administrador.
- **Clase Préstamo**
    - **Propósito:** Es la entidad central para la gestión de créditos.
    - **Atributos:**
        - **monto, plazoMeses, tasaInteres:** Datos de un préstamo, necesarios para el registro y los cálculos.
        - **saldoPendiente, estado:** Claves para que el usuario pueda consultar su saldo y ver el estado de su crédito.
    - **Métodos:**
        - **recalcularSaldo():** Este método es activado por la clase Abono y cumple con el requerimiento de recalcular el saldo tras abonos.

- **Clase Abono**
    - **Propósito:** Registra los pagos parciales a un préstamo.
    - **Atributos:**
        - montoAbonado, fechaAbono: Los datos de cada pago realizado.
    - **Relación: Asociación** con Préstamo. Un

      préstamo puede tener muchos Abonos, lo que asegura la consistencia de datos.
- **Clase Alerta**
    - **Propósito:** Maneja todas las notificaciones que el sistema envía al usuario.
    - **Atributos:**
        - **mensaje, tipoAlerta**: Permite diferenciar entre alertas de sobreendeudamiento y otras notificaciones.
    - **Métodos:**
        - **visualizar ():** Permite que las alertas se muestren en la interfaz, cumpliendo con el requerimiento de visualización.

2.2 Clases de Servicio (Microservicios)

Estas clases representan la lógica de negocio y actúan como

microservicios independientes.

- **Clase CalculoService**
    - **Propósito:** Es el motor financiero del sistema, encargado de toda la lógica de cálculo.
    - **Relación: Dependencia** con Prestamo. Prestamo utiliza los métodos de CalculoService para calcular sus valores. Este diseño cumple con los requerimientos de **Escalabilidad** y **Escalabilidad horizontal**, ya que el servicio de cálculo puede ser un módulo independiente.

- **Métodos:**
    - **calcularCuotaInteresSimple():** Cumple con el requerimiento de cálculo de interés simple.
    - **calcularCuotaInteresCompuesto():** Cumple con el requerimiento de cálculo de interés compuesto.
    - **sugerirAbono():** Cumple con el requerimiento de sugerencias de pago para terminar antes el crédito.

- **Clase NotificacionService**
    - **Propósito:** Gestiona el envío de todas las alertas y notificaciones.
    - **Métodos:**
        - **enviarNotificacion():** Cumple con el requerimiento de notificaciones en tiempo real.
        - **generarAlertaSobreendeudamiento():** Cumple con el requerimiento de alertas de sobreendeudamiento.

- **Clase ReporteService**
    - **Propósito:** Centraliza la lógica para la generación de reportes y la visualización de datos.
    - **Métodos:**
        - **generarReporteConsolidado():** Cumple con el requerimiento de reportes consolidados
        - **generarGraficaEvolucion():** Cumple con el requerimiento de gráficas de evolución.
        - **filtrarPrestamos() y ordenarPrestamos():** Cumplen con los requerimientos de filtrar y ordenar préstamos.