Fernando A. Loncon P.

IV Medio A Técnico Profesional

Programación Orientada a Objetos

Colegio San Lorenzo – Octubre 2025

Noelia Lopez, Ignacia Salazar

Implementación Piloto de Gestor de Datos Académicos

Resumen Previo

El presente proyecto detalla el diseño, desarrollo e implementación de una aplicación de escritorio para la gestión de datos académicos, creada como proyecto piloto en el contexto de la asignatura de Programación Orientada a Objetos. El objetivo principal del software es centralizar y facilitar el manejo de información clave, como calificaciones, asistencia y registros de estudiantes, para el entorno del Colegio San Lorenzo.

En este documento se describe la arquitectura de la aplicación, las tecnologías utilizadas [Ejemplo: Java, C#, Python, SQL Lite, etc.] y las metodologías de desarrollo aplicadas. La fase piloto consistió en [Explica brevemente la prueba, ej: "la carga de datos ficticios de un curso"], y los resultados demuestran la viabilidad de la herramienta, su facilidad de uso y su potencial para optimizar los procesos administrativos académicos.

1. Introducción

En la era digital actual, la gestión eficiente de la información es crucial para cualquier institución educativa. A menudo, los colegios manejan grandes volúmenes de datos académicos (notas, informes de personalidad, asistencia, etc.) a través de métodos manuales o planillas de cálculo dispersas, lo cual es propenso a errores, consume tiempo y dificulta la consulta.

La problemática identificada en el [Menciona el contexto, ej: "Colegio San Lorenzo" o "en el ámbito educativo técnico"] es la ausencia de una herramienta informática unificada que permita a los docentes y administrativos gestionar esta información de manera ágil y segura.

Como respuesta a esta necesidad, este proyecto propone el desarrollo de una "Implementación Piloto de Gestor de Datos Académicos". Se trata de una aplicación de escritorio, desarrollada con los principios de la Programación Orientada a Objetos, que busca ser una solución inicial y funcional a este desafío.

Este informe documentará el proceso completo: desde la definición de los requisitos y el alcance de la aplicación (Capítulo 2), pasando por las decisiones de diseño e implementación (Capítulo 3) y las tecnologías específicas empleadas (Capítulo 4), hasta la presentación de los resultados y las conclusiones obtenidas tras la fase piloto (Capítulo 5).

2. Descripción y Alcance App

2.1 Descripción de la Aplicación

El "Gestor de Datos Académicos" es una aplicación de escritorio diseñada para centralizar el registro y la consulta de la información académica de los estudiantes. El software ofrece una interfaz gráfica de usuario (GUI) intuitiva que permite a los usuarios (como docentes o administradores) gestionar las siguientes entidades principales:

- Gestión de Alumnos: Permite crear, editar, eliminar y buscar perfiles de estudiantes, almacenando sus datos personales (nombre, RUT, curso, etc.).
- Gestión de Asignaturas: Habilita el registro de las materias o módulos que cursa cada nivel.
- Registro de Calificaciones: Facilita la introducción de notas por asignatura y por estudiante, calculando automáticamente los promedios parciales y finales. ● Control de Asistencia: Proporciona un módulo para registrar la asistencia diaria (presente, ausente, justificado) de los alumnos.

● Generación de Reportes (Básico): Ofrece funciones para visualizar informes simples, como la lista de notas de un curso o la ficha de un alumno.

2.2 Alcance de la Aplicación

El alcance define los límites funcionales del software desarrollado.

Funcionalidades Incluidas (En Alcance):

● Autenticación de usuarios (Login/Contraseña) para acceder al sistema. ● Operaciones CRUD (Crear, Leer, Actualizar, Borrar) para alumnos y asignaturas. ● Cálculo de promedios simples por asignatura.
● Almacenamiento persistente de datos en una base de datos local.

Funcionalidades Excluidas (Fuera de Alcance):

● Este proyecto no es una aplicación web; funciona de manera local en el computador donde se instala.
● No incluye módulos complejos como gestión de horarios, comunicación con apoderados (email o notificaciones) o conexión a redes externas.
● No genera reportes estadísticos avanzados ni certificados oficiales imprimibles.

3. Descripción y Alcance Implementación

3.1 Descripción de la Implementación Piloto

La implementación actual corresponde a una "versión Piloto" o un Producto Mínimo Viable (MVP). El objetivo de este piloto no era construir un sistema completo para todo el colegio, sino demostrar que el concepto es funcional y útil en un entorno controlado.

La implementación se centró en construir el "núcleo" del sistema: la base de datos y las ventanas (vistas) esenciales para la gestión de un curso. Se desarrollaron las interfaces para añadir alumnos y registrar sus calificaciones, asegurando que los datos se guarden y se puedan recuperar correctamente.

3.2 Alcance de la Implementación

El piloto se limitó a las siguientes metas concretas:

● Desarrollar y probar la conexión a la base de datos.
● Implementar la ventana de "Gestión de Alumnos" (altas, bajas y modificaciones).
● Implementar la ventana de "Ingreso de Notas" para una asignatura. ● Probar el cálculo de promedio para un alumno en esa asignatura. ● Se utilizaron datos de prueba (ficticios) para simular el funcionamiento de un curso [Ejemplo: "IV Medio A Técnico Profesional"].

Por lo tanto, esta implementación no está preparada para un despliegue masivo en el colegio, sino que sirve como la base fundamental sobre la cual se pueden construir futuras versiones.

4. Codificación y Tecnologías Utilizadas

Para el desarrollo de esta aplicación de escritorio se emplearon los principios de la Programación Orientada a Objetos (POO), aplicando conceptos como clases, objetos, encapsulamiento y herencia para modelar el problema.

El stack tecnológico seleccionado fue el siguiente:

- **Lenguaje de Programación:** [Java / C# / Python / Visual Basic .NET] ○ *Razón:* [Ej: "Se eligió Java por su portabilidad y el robusto manejo de la POO." / "Se eligió C# por su fuerte integración con el entorno de desarrollo Visual Studio."]
- **Entorno de Desarrollo Integrado (IDE):** [Apache NetBeans / Visual Studio / IntelliJ IDEA / VS Code]
  - ○ *Razón:* [Ej: "Se utilizó NetBeans por su facilidad para diseñar interfaces gráficas (Swing/JavaFX)".]
- **Gestor de Base de Datos:** [SQLite / MySQL / SQL Server Express / PostgreSQL]
  - ○ *Razón:* [Ej: "Se optó por SQLite por ser una base de datos ligera, basada en un solo archivo, ideal para una aplicación de escritorio piloto sin necesidad de un servidor."]
- **Bibliotecas de GUI (Interfaz Gráfica):** [JavaFX / Windows Forms (.NET) / WPF (.NET) / Tkinter (Python) / Java Swing]
  - ○ *Razón:* [Ej: "Se usó Windows Forms por la facilidad de 'arrastrar y soltar' componentes visuales en el IDE."]
- **Control de Versiones:** [Git / GitHub (Opcional, si lo usaste)] ○ *Razón:* [Ej: "Se utilizó Git para mantener un historial de cambios ordenado y facilitar el trabajo en el código."]

5. Resultados y Conclusiones

5.1 Resultados Obtenidos

La fase de implementación piloto arrojó resultados positivos que validan el diseño y la funcionalidad básica de la aplicación. Las pruebas se centraron en las funciones principales definidas en el alcance:

1. **Conexión exitosa:** Se logró una conexión estable y persistente con la base de

datos [SQLite/MySQL/etc.]. Los datos se guardan al cerrar la aplicación y se recuperan correctamente al iniciarla.

2. **Gestión de Alumnos (CRUD):** Las operaciones de Crear, Leer, Actualizar y Borrar (CRUD) para la entidad "Alumno" fueron implementadas y probadas exitosamente. Es posible añadir nuevos estudiantes, modificar sus datos (ej. cambiar de curso) y eliminarlos del sistema.
3. **Registro de Calificaciones:** El módulo de ingreso de notas demostró ser funcional. Las calificaciones ingresadas se asocian correctamente al alumno y a la asignatura correspondiente.
4. **Cálculo de Promedios:** La lógica de negocio para el cálculo de promedios simples funciona de acuerdo a lo esperado, actualizando el promedio del estudiante automáticamente tras el ingreso de una nueva calificación.
5. **Interfaz de Usuario:** La interfaz gráfica, aunque básica, resultó ser intuitiva en las pruebas iniciales, con una navegación clara entre las distintas ventanas (vistas) del programa.

5.2 Conclusiones

Del desarrollo y los resultados de este proyecto piloto, se concluye lo siguiente:

- Se cumplió el objetivo principal de implementar una aplicación de escritorio funcional que sirve como gestor básico de datos académicos.
- Las tecnologías seleccionadas (Ej: [Java y SQLite]) demostraron ser adecuadas para un proyecto de esta envergadura inicial, ofreciendo las herramientas necesarias para un desarrollo rápido y eficiente.
- El uso de la Programación Orientada a Objetos (POO) fue fundamental para estructurar el código de manera ordenada y modular. Conceptos como la clase `Alumno`, `Asignatura` y `GestorDB` permitieron encapsular la lógica de negocio y facilitar la mantenibilidad.
- El piloto demuestra la viabilidad de desarrollar una herramienta más completa a futuro. Las bases creadas en este proyecto (conexión a BD, clases de modelo, vistas principales) son un cimiento sólido para escalar la aplicación.

6. **Análisis y Recomendaciones**

6.1 Análisis del Proyecto (FODA)

- **Fortalezas:**
  - Solución Específica: La app está pensada para las necesidades concretas de un entorno escolar.
  - Facilidad de Uso: Al ser una app de escritorio nativa, su uso es rápido y la

interfaz es simple.
    - Funcionamiento Offline: No requiere conexión a internet para operar, lo que garantiza disponibilidad.
- **Debilidades:**
    - No Escalable: Al usar una base de datos local [SQLite], la aplicación no puede ser usada por múltiples usuarios simultáneamente (no es cliente-servidor).
    - Funcionalidad Limitada: Ausencia de módulos importantes como reportes avanzados (PDF), gestión de asistencia compleja o perfiles de usuario (roles de 'admin' vs. 'profesor').
    - Seguridad Básica: La seguridad se limita al acceso al equipo, sin un sistema de login robusto dentro de la app.
- **Oportunidades:**
    - Escalar a Web/Red: Migrar la aplicación a una tecnología web (o cliente-servidor) para permitir el acceso multiusuario.
    - Integración: Futuras versiones podrían integrarse con otros sistemas del colegio.
    - Prueba Real: Implementar el piloto en un curso real para obtener retroalimentación directa de los docentes.
- **Amenazas:**
    - Resistencia al Cambio: Posible dificultad para que los usuarios abandonen métodos tradicionales (ej. libros de clases físicos o Excel).
    - Pérdida de Datos: Al ser local, si el archivo de la base de datos se corrompe o elimina, se pierde la información (requiere política de respaldos manuales).

6.2 Recomendaciones y Trabajo Futuro

Basado en el análisis anterior, se recomienda para futuras versiones del proyecto:

1. **Migrar la Base de Datos:** Reemplazar [SQLite] por un gestor de base de datos robusto y centralizado (como MySQL, PostgreSQL o SQL Server) que permita conexiones en red.
2. **Implementar Roles de Usuario:** Desarrollar un módulo de autenticación que distinga entre "Administradores" (con control total) y "Profesores" (con acceso solo a sus cursos).
3. **Desarrollar un Módulo de Reportes:** Añadir la funcionalidad de generar e imprimir/exportar a PDF informes clave (libreta de notas, lista de curso, informe de asistencia).

4. **Mejorar la Interfaz de Usuario (UI/UX):** Realizar un rediseño de la interfaz basándose en la retroalimentación de usuarios finales para hacerla aún más amigable.

7. Antecedentes Bibliográficos (Los que estaban en inglés los traducimos con google)

- Define los conceptos clave del ciclo de vida del desarrollo de software, relevante para la planificación del proyecto.Recuperado de:
    - https://www.ibm.com/es-es/topics/software-engineering
- Microsoft (2025). Documentación oficial de .NET y C#. Recuperado de:
    - https://docs.microsoft.com/es-es/dotnet/csharp/
- W3Schools (2025). SQL Tutorial. Recuperado de:
    - https://www.w3schools.com/sql/