Reto 3. Aprendizajes y Reflexión Personal de Diseño de base de datos – Sistema de Gestión de Biblioteca Escolar

Rafael Alfonso Ruíz García – ID: Key_000060

Facultad de Ingenierías, Instituto Kriete de Ciencias e Ingenierías (Key)

Programación Orientada a Objetos

José Luis Montalvo

Miércoles 1 de octubre de 2025

Resumen

El presente informe describe la implementación de un sistema de gestión de biblioteca escolar utilizando C++. El proyecto maneja autores, libros, estudiantes y préstamos, con persistencia en archivos de texto. Se enfatizó la normalización de datos y la validación de entradas para garantizar la consistencia y confiabilidad de la información (W3Schools.com, s. f.).

Aprendizajes del reto y reflexión

Durante la implementación, se emplearon estructuras (struct) para representar las tablas principales y vectores (vector) para almacenar los datos en memoria (GeeksforGeeks, 2025). Cada entidad cuenta con operaciones básicas de CRUD (Crear, Leer, Actualizar y Eliminar), implementadas mediante un menú interactivo con switch-case. Se incluyó validación de ID duplicado para autores, libros, estudiantes y préstamos, así como control para evitar que un libro ya prestado fuera registrado nuevamente hasta ser devuelto.

La normalización permitió separar correctamente las entidades en tablas independientes: Autor, Libro, Estudiante y Préstamo. Esta separación evita redundancias y facilita la administración de relaciones mediante claves primarias y foráneas (Ibitola, 2025). Por ejemplo, cada libro se asocia a un autor mediante *id_autor* y cada préstamo a un estudiante y un libro mediante *id_estudiante e id_libro*. Gracias a esta práctica, se comprendió la importancia de mantener la integridad referencial y cómo esto impacta directamente en la calidad de los datos.

La persistencia se implementó utilizando archivos de texto en formato CSV (C++ File Handling, s. f.). Al iniciar el programa, los datos se cargan automáticamente, y cualquier modificación realizada durante la ejecución se guarda de manera segura, evitando la pérdida de información. Esta experiencia reforzó la comprensión de conceptos de bases de datos relacionales y la aplicación de buenas prácticas de programación en C++.

Durante el desarrollo del proyecto, aprendí a diseñar y normalizar datos para prevenir duplicados y errores de consistencia, lo que es fundamental en cualquier sistema de información. Además, comprendí cómo aplicar validaciones y controles de errores mejora la confiabilidad del programa. La experiencia también consolidó mis habilidades en C++ y me permitió apreciar cómo un diseño estructurado y una correcta gestión de relaciones entre entidades facilitan el mantenimiento y la escalabilidad de un sistema.

Conclusión

La implementación del sistema mostró que la normalización y el manejo adecuado de errores son esenciales para la consistencia y confiabilidad de la información. Asimismo, permitió consolidar habilidades técnicas en C++ y en la gestión de datos estructurados, proporcionando una base sólida para futuros proyectos relacionados con bases de datos y sistemas de información.

Referencias

C++ File Handling. (s. f.). https://www.programiz.com/cpp-programming/file-handling

GeeksforGeeks. (2025, 21 agosto). *File Handling through C++ Classes*. GeeksforGeeks. https://www.geeksforgeeks.org/cpp/file-handling-c-classes/

Ibitola, J. (2025, 29 septiembre). *Data Normalization Demystified: A Guide to Cleaner Data*. Flagright. https://www.flagright.com/post/data-normalization-demystified-a-guide-to-cleaner-data

W3Schools.com. (s. f.). https://www.w3schools.com/cpp/cpp_files.asp