TRƯỜNG ĐẠI HỌC HÀ TĨNH

**KHOA KỸ THUẬT – CÔNG NGHỆ**



**BÁO CÁO ĐỒ ÁN 2**

***Đề tài*: Xây dựng ứng dụng quản lý nề nếp**

Sinh viên thực hiện: Trần Đình Đăng

Lớp: K16 CNTT

Người hướng dẫn: ThS. Phan Thị Gấm

***Năm, 2025***

**LỜI CAM ĐOAN**

Tôi xin cam đoan rằng toàn bộ nội dung trong đồ án này là kết quả nghiên cứu, tìm hiểu của riêng tôi dưới sự hướng dẫn của cô Phan Thị Gấm.

Những số liệu, dữ liệu và tài liệu tham khảo trong đồ án được trích dẫn đầy đủ nguồn gốc và trung thực. Tôi chịu hoàn toàn trách nhiệm về tính chính xác và trung thực của nội dung trong báo cáo này.

Nếu có bất kỳ sai sót hay vi phạm nào được phát hiện, tôi xin hoàn toàn chịu trách nhiệm trước Hội đồng chấm đồ án/luận văn và các quy định của nhà trường.

Tôi xin chân thành cảm ơn!

*Ngày 25 tháng 08 năm 2025*

**Người thực hiện**

**Đăng**

**Trần Đình Đăng**

MỤC LỤC

[MỞ ĐẦU 5](#_Toc198491529)

[1. Lý do chọn đề tài 5](#_Toc198491530)

[2. Mục tiêu 5](#_Toc198491531)

[3. Phạm vi và đối tượng áp dụng 6](#_Toc198491532)

[3. Cấu trúc của đề tài 6](#_Toc198491533)

[Chương 1. GIỚI THIỆU TỔNG QUAN 8](#_Toc198491534)

[1.1. Giới thiệu tổng quan về đề tài 8](#_Toc198491535)

[1.2. Tổng quan về .Net Framework và ngôn ngữ lập trình C# 8](#_Toc198491536)

[1.3. Giới thiệu về hệ quản trị CSDL SQL SERVER 2022 8](#_Toc198491537)

[1.4. Các nền tảng và công cụ hỗ trợ khác 8](#_Toc198491538)

[Chương 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG … 10](#_Toc198491539)

[2.1. Giới thiệu về hệ thống 10](#_Toc198491540)

[2.1.1. Thực trạng của hệ thống 10](#_Toc198491541)

[2.1.2. Các yêu cầu của hệ thống 10](#_Toc198491542)

[2.2. Phân tích hệ thống 10](#_Toc198491543)

[2.2.1. Biểu đồ phân cấp chức năng 10](#_Toc198491544)

[2.2.2. Biểu đồ luồng dữ liệu 🡪tương tự như phân cấp chức năng, từ biểu đồ phân cấp chức năng, xác định các tác nhân (nhóm người dùng) để vẽ các luồng dữ liệu cung cấp cho hệ thống. Mỗi biểu đồ đều có hình ảnh và mô tả hoạt động của biểu đồ đó bằng lời 11](#_Toc198491545)

[2.2.3. Mô hình quan hệ thực thể 11](#_Toc198491546)

[2.3. Thiết kế hệ thống 11](#_Toc198491547)

[2.3.1. Thiết kế chức năng 12](#_Toc198491548)

[2.3.1. Thiết kế cơ sở dữ liệu 12](#_Toc198491549)

[2.3.2. Thiết kế một số giao diện chức năng 12](#_Toc198491550)

[Chương 3. CÀI ĐẶT VÀ KIỂM THỬ HỆ THỐNG 13](#_Toc198491551)

[3.1. Cài đặt hệ thống 13](#_Toc198491552)

[3.2. Một số giao diện hệ thống đã cài đặt 13](#_Toc198491553)

[3.3. Kết quả kiểm thử hệ thống 13](#_Toc198491554)

[KẾT LUẬN 14](#_Toc198491555)

[TÀI LIỆU THAM KHẢO 14](#_Toc198491556)

**DANH MỤC HÌNH ẢNH**

[Hình 1.1. Tên hình ảnh thứ nhất của chương 1 9](#_Toc198491825)

[Hình 2.1. Tên hình ảnh thứ 1 của chương 2 12](#_Toc198491826)

# MỞ ĐẦU

## Trong thời đại công nghệ số bùng nổ, việc quản lý và đánh giá nề nếp học sinh tại các trường học không chỉ dừng lại ở các phương pháp thủ công mà còn đòi hỏi sự đổi mới để đảm bảo tính chính xác và công bằng. Điều này đặt ra yêu cầu về việc quản lý hiệu quả hoạt động của Đội Cờ Đỏ, từ khâu ghi nhận các trường hợp vi phạm hoặc biểu dương, chấm điểm thi đua, cho đến tổng hợp và báo cáo kết quả. Tuy nhiên, việc vận hành Đội Cờ Đỏ theo cách truyền thống thường gặp phải những khó khăn như sai sót trong ghi nhận, chủ quan trong đánh giá, chậm trễ trong việc tổng hợp điểm và khó khăn trong việc theo dõi, đối chiếu dữ liệu.

## Phân tích và thiết kế hệ thống quản lý chấm điểm của Đội Cờ Đỏ là bước đi chiến lược nhằm tối ưu hóa toàn bộ quy trình này. Đây là giải pháp phần mềm giúp tự động hóa các tác vụ, từ đó nâng cao hiệu quả hoạt động của Đội Cờ Đỏ, giảm thiểu sai sót, đảm bảo tính khách quan và tiết kiệm thời gian, công sức. Hệ thống này không chỉ giúp nhà trường theo dõi tình hình nề nếp của các lớp theo thời gian thực mà còn cải thiện tính minh bạch và công bằng trong đánh giá, từ việc ghi nhận sự vụ nhanh chóng, chấm điểm chính xác theo tiêu chí đến việc công bố kết quả kịp thời và rõ ràng.

## Với sự hỗ trợ từ công nghệ, các nhà trường có thể tối ưu hóa hoạt động của Đội Cờ Đỏ, tăng cường ý thức chấp hành nội quy của học sinh và tạo nên một môi trường học tập kỷ luật, tiến bộ. Hệ thống quản lý hiện đại không chỉ là giải pháp ngắn hạn mà còn là nền tảng cho việc xây dựng và duy trì nề nếp học đường một cách bền vững.

## 1. Lý do chọn đề tài

## Trong thời đại công nghệ thông tin phát triển mạnh mẽ, việc ứng dụng các hệ thống quản lý tự động không chỉ mang lại hiệu quả cao mà còn tối ưu hóa nguồn lực trong mọi lĩnh vực. Đặc biệt, trong công tác quản lý nề nếp học sinh tại các trường học – một hoạt động yêu cầu sự chính xác, công bằng và kịp thời – hệ thống quản lý hiện đại sẽ giúp nâng cao hiệu quả của Đội Cờ Đỏ cũng như tối ưu hóa quy trình theo dõi và đánh giá.

## Nhiều trường học hiện vẫn thực hiện việc chấm điểm thi đua của Đội Cờ Đỏ một cách thủ công hoặc sử dụng các công cụ rời rạc, dẫn đến nguy cơ thiếu sót thông tin, sai lệch trong quá trình chấm điểm, mất nhiều thời gian tổng hợp và khó khăn trong việc lưu trữ, tra cứu, báo cáo kết quả một cách hệ thống. Vì vậy, xây dựng một hệ thống quản lý chấm điểm cho Đội Cờ Đỏ một cách khoa học và tự động là nhu cầu cấp thiết, góp phần nâng cao chất lượng công tác quản lý học sinh và xây dựng môi trường học tập kỷ luật, nề nếp.

## 2. Mục tiêu

## Mục tiêu chính của đề tài là tăng cường hiệu quả trong công tác ghi nhận, chấm điểm và tổng hợp kết quả thi đua của Đội Cờ Đỏ bằng cách áp dụng các công cụ công nghệ hiện đại và tự động hóa quy trình. Hệ thống thông minh này giúp tối ưu hóa việc ghi nhận các lỗi vi phạm hoặc các mặt tốt của từng lớp, tính toán điểm số một cách khách quan dựa trên các tiêu chí rõ ràng đã được thiết lập, và theo dõi sát sao tình hình nề nếp của các lớp theo từng tuần, từng tháng. Điều này không chỉ đảm bảo việc chấm điểm được thực hiện một cách chính xác, nhất quán và công bằng, mà còn giúp giảm thiểu rủi ro sai sót do ghi chép thủ công hay đánh giá chủ quan từ các thành viên Đội Cờ Đỏ. Hơn nữa, việc cải tiến công tác quản lý nề nếp theo hướng khoa học, minh bạch và kịp thời không chỉ giúp nhà trường duy trì kỷ luật học đường một cách hiệu quả mà còn góp phần xây dựng một môi trường học tập tích cực, nơi sự nỗ lực và tiến bộ trong việc chấp hành nội quy được ghi nhận và khích lệ đúng lúc.

## Cụ thể, các mục tiêu cần đạt được bao gồm:

## Giảm thiểu sai sót và tính chủ quan trong quá trình chấm điểm: Hệ thống hóa các tiêu chí chấm điểm rõ ràng, cho phép nhập liệu và tính điểm tự động hoặc bán tự động, hạn chế tối đa các lỗi do ghi chép thủ công, cộng điểm nhầm lẫn hoặc đánh giá cảm tính. Đồng thời, tiết kiệm thời gian và công sức cho các thành viên Đội Cờ Đỏ cũng như giáo viên phụ trách thông qua việc loại bỏ các bước tổng hợp, báo cáo thủ công không cần thiết, giúp tối ưu hóa toàn bộ quy trình từ khâu ghi nhận ban đầu đến khi công bố kết quả cuối cùng.

## Nâng cao tính minh bạch, công bằng và tạo động lực cho học sinh: Khi quy trình chấm điểm được số hóa, các tiêu chí đánh giá được công khai và kết quả được cập nhật nhanh chóng, chính xác, học sinh và các tập thể lớp có thể dễ dàng theo dõi và hiểu rõ hơn về tình hình chấp hành nội quy của lớp mình so với các lớp khác. Điều này tạo sự tin tưởng vào tính công bằng của việc đánh giá, đồng thời khích lệ ý thức tự giác chấp hành, phấn đấu cải thiện nề nếp, góp phần xây dựng môi trường học đường văn minh, kỷ luật.

## Cung cấp các công cụ báo cáo và phân tích dữ liệu trực quan, chi tiết: Cho phép Ban Giám Hiệu, giáo viên chủ nhiệm và tổng phụ trách Đội dễ dàng theo dõi tình hình nề nếp chung của toàn trường, cũng như của từng khối, từng lớp qua các biểu đồ, bảng thống kê. Từ đó, có thể nhanh chóng nắm bắt được các lỗi vi phạm phổ biến, các lớp có tiến bộ hoặc cần lưu ý, đánh giá hiệu quả của công tác giáo dục nề nếp và đưa ra các quyết định quản lý, biện pháp can thiệp, giáo dục kịp thời, phù hợp, hướng tới sự phát triển ổn định và toàn diện của học sinh.

## 3. Phạm vi và đối tượng áp dụng

* Phạm vi: Hệ thống được thiết kế để quản lý toàn diện quy trình chấm điểm thi đua nề nếp của Đội Cờ Đỏ trong phạm vi một trường học phổ thông (bao gồm các cấp Tiểu học, Trung học cơ sở, hoặc Trung học phổ thông).
* Đối tượng sử dụng: Bao gồm giáo viên tổng phụ trách đội và các thành viên của đội cờ đỏ

## 3. Cấu trúc của đề tài

* **Phân tích yêu cầu hệ thống:** Đây là giai đoạn nền tảng, nhằm tìm hiểu và xác định rõ ràng các chức năng cần có của hệ thống quản lý chấm điểm Đội Cờ Đỏ, các yêu cầu về mặt kỹ thuật, và quy trình nghiệp vụ cụ thể mà nhà trường đang áp dụng hoặc mong muốn cải tiến. Quá trình này đòi hỏi sự trao đổi, phỏng vấn với các bên liên quan như Ban Giám hiệu, Tổng phụ trách Đội, giáo viên chủ nhiệm và các thành viên Đội Cờ Đỏ để thu thập thông tin chi tiết về cách thức chấm điểm, các tiêu chí, biểu mẫu báo cáo, và các vấn đề cần giải quyết. Các công cụ như biểu đồ luồng dữ liệu (DFD), biểu đồ trường hợp sử dụng (Use Case Diagram) sẽ được vận dụng để mô tả trực quan hóa các yêu cầu, các tác nhân tương tác với hệ thống và các luồng xử lý chính, đảm bảo sự hiểu biết thống nhất và đầy đủ..
* **Thiết kế hệ thống:** Sau khi xác định yêu cầu, bước tiếp theo là tiến hành thiết kế chi tiết. Trong đó, sơ đồ phân cấp chức năng sẽ giúp hình dung rõ ràng các chức năng chính và phụ, đảm bảo mọi yếu tố quan trọng đều được đề cập. Việc thiết kế cơ sở dữ liệu là chìa khóa để quản lý dữ liệu hiệu quả, đảm bảo tính toàn vẹn và khả năng mở rộng trong tương lai. Ngoài ra, giao diện người dùng được thiết kế thân thiện, dễ sử dụng, tối ưu hóa trải nghiệm người dùng, đồng thời đảm bảo tính nhất quán và hiệu quả trong quá trình tương tác.
* **Triển khai và kiểm thử:** Đây là giai đoạn hiện thực hóa thiết kế thành một hệ thống hoàn chỉnh. Đầu tiên, hệ thống sẽ được triển khai tại môi trường thực tế, sau đó tiến hành kiểm thử để đảm bảo tính ổn định, độ tin cậy và hiệu suất. Các lỗi và vấn đề được phát hiện trong giai đoạn này sẽ được xử lý kịp thời nhằm đảm bảo hệ thống hoạt động đúng như mong đợi. Đồng thời, tài liệu hướng dẫn triển khai cũng được cung cấp để hỗ trợ đội ngũ vận hành và bảo trì hệ thống.

Kết luận và đề xuất phát triển: Cuối cùng, dự án được tổng kết lại bằng cách đánh giá các ưu điểm, hạn chế và hiệu quả mà hệ thống mang lại. Bên cạnh đó, các đề xuất cải tiến được đưa ra nhằm nâng cao chất lượng và tính năng của các hệ thống trong các giai đoạn phát triển tiếp theo sắp tới. Việc liên tục cải tiến không chỉ giúp hệ thống phù hợp với nhu cầu thay đổi của trường học mà còn đảm bảo sự ổn định và bền vững trong tương lai sắp tới.

# Chương 1. GIỚI THIỆU TỔNG QUAN

## 1.1. Giới thiệu tổng quan về đề tài

## Trong môi trường giáo dục phổ thông, công tác quản lý nề nếp và thúc đẩy thi đua học tập, rèn luyện của học sinh giữ một vai trò hết sức quan trọng. Đội Cờ Đỏ (hay Đội Thanh niên xung kích) là lực lượng nòng cốt hỗ trợ nhà trường trong việc theo dõi, đôn đốc và đánh giá việc chấp hành nội quy của các tập thể lớp. Nghiệp vụ cốt lõi của bài toán quản lý chấm điểm Đội Cờ Đỏ bao gồm các hoạt động chính:

## Thiết lập và quản lý các tiêu chí đánh giá: Xác định rõ ràng các hạng mục cần theo dõi (ví dụ: đồng phục, đi học đúng giờ, vệ sinh lớp học, kỷ luật trong giờ học, tham gia phong trào) và thang điểm tương ứng cho từng tiêu chí.

## Ghi nhận và theo dõi: Các thành viên Đội Cờ Đỏ thực hiện việc kiểm tra, ghi nhận các trường hợp vi phạm hoặc những biểu hiện tốt của từng lớp theo lịch phân công.

## Tổng hợp và tính điểm: Dữ liệu ghi nhận được tập hợp, đối chiếu và tính toán thành điểm thi đua cho mỗi lớp dựa trên các tiêu chí đã đặt ra.

## Báo cáo và xếp hạng: Từ điểm số tổng hợp, hệ thống cần tạo ra các báo cáo xếp hạng thi đua (theo tuần, tháng, học kỳ) để nhà trường có cơ sở khen thưởng, nhắc nhở và định hướng công tác giáo dục nề nếp.

## Tình hình sử dụng công nghệ cho nghiệp vụ này hiện nay tại đa số các trường học còn nhiều hạn chế. Quy trình chủ yếu vẫn dựa trên phương pháp thủ công: ghi chép bằng sổ sách, biên bản giấy, hoặc sử dụng các công cụ văn phòng cơ bản như Microsoft Excel một cách rời rạc. Cách làm này tiềm ẩn nhiều vấn đề như:

## Mất nhiều thời gian và công sức: Từ khâu ghi nhận, tổng hợp đến lập báo cáo đều tốn thời gian của cả học sinh trong Đội Cờ Đỏ và giáo viên phụ trách.

## Dễ sai sót và thiếu khách quan: Việc nhập liệu, tính toán thủ công dễ dẫn đến nhầm lẫn. Đánh giá có thể mang tính chủ quan nếu không có tiêu chí rõ ràng và công cụ hỗ trợ đồng bộ.

## Khó khăn trong lưu trữ và tra cứu: Dữ liệu phân tán, khó quản lý tập trung và truy xuất khi cần thiết, đặc biệt là khi cần xem lại lịch sử hoặc phân tích xu hướng.

## Thiếu tính kịp thời: Báo cáo thường được tổng hợp muộn, làm giảm hiệu quả của việc phản hồi thông tin đến các lớp và Ban Giám hiệu.

## Để giải quyết những thách thức trên và nâng cao hiệu quả quản lý, đề tài "Xây dựng Ứng dụng Quản lý Chấm điểm cho Đội Cờ Đỏ" được thực hiện. Mục tiêu là phát triển một phần mềm ứng dụng máy tính để bàn (desktop application) chuyên biệt, giúp số hóa và tự động hóa các khâu trong quy trình nghiệp vụ.

## Các công nghệ được lựa chọn để phát triển ứng dụng bao gồm:

## Ngôn ngữ lập trình: C#, một ngôn ngữ lập trình hiện đại, mạnh mẽ và hướng đối tượng, thuộc nền tảng .NET Framework/ .NET Core.

## Môi trường phát triển tích hợp (IDE): Visual Studio 2022, cung cấp bộ công cụ toàn diện để thiết kế, lập trình, gỡ lỗi và triển khai ứng dụng.

## Hệ quản trị cơ sở dữ liệu: SQL Server, một hệ quản trị CSDL quan hệ mạnh mẽ, đáng tin cậy từ Microsoft, phù hợp để lưu trữ và quản lý dữ liệu có cấu trúc của hệ thống như thông tin học sinh, lớp học, tiêu chí chấm điểm, các biên bản ghi nhận và kết quả điểm số.

## Bộ công cụ giao diện người dùng (UI Controls): DevExpress, một thư viện giao diện người dùng phong phú và chuyên nghiệp cho các ứng dụng .NET, giúp tạo ra các giao diện trực quan, dễ sử dụng, có tính tương tác cao và nhiều tính năng nâng cao (như lưới dữ liệu, biểu đồ, báo cáo).

## Sự kết hợp của các công nghệ này sẽ tạo nền tảng vững chắc để xây dựng một ứng dụng quản lý chấm điểm Đội Cờ Đỏ hiệu quả, ổn định, dễ bảo trì và có khả năng mở rộng trong tương lai, đáp ứng tốt các yêu cầu nghiệp vụ và mang lại giá trị thiết thực cho công tác quản lý nề nếp tại các trường học.

## 1.2. Tổng quan về .Net Framework và ngôn ngữ lập trình C#

Trong phần này, chúng ta sẽ tìm hiểu sâu hơn về nền tảng công nghệ .NET Framework, cách tiếp cận phát triển ứng dụng máy tính để bàn (desktop) với Windows Forms, và ngôn ngữ lập trình C# – những công cụ và kiến thức nền tảng được lựa chọn để xây dựng ứng dụng quản lý chấm điểm cho Đội Cờ Đỏ.

**1.2.1. Giới thiệu về .NET Framework**

**1.2.1.1. .NET Framework là gì?**

.NET Framework là một nền tảng phát triển phần mềm (software development platform) được phát triển bởi Microsoft. Nó ra mắt lần đầu tiên vào năm 2002, với mục tiêu cung cấp một môi trường lập trình nhất quán cho việc xây dựng các loại ứng dụng khác nhau, từ ứng dụng desktop, ứng dụng web, dịch vụ web, đến ứng dụng di động và nhiều hơn nữa.

**Mục tiêu chính của .NET Framework:**

* Cung cấp một môi trường quản lý mã (managed execution environment): Giúp đơn giản hóa việc phát triển bằng cách tự động hóa nhiều tác vụ phức tạp như quản lý bộ nhớ, xử lý lỗi, và bảo mật.
* Đơn giản hóa việc phát triển và triển khai ứng dụng: Cung cấp một thư viện lớp (Framework Class Library - FCL) phong phú, giúp lập trình viên tái sử dụng mã và tăng tốc độ phát triển.
* Hỗ trợ đa ngôn ngữ lập trình: Cho phép các nhà phát triển sử dụng nhiều ngôn ngữ lập trình khác nhau (như C#, VB.NET, F#) trên cùng một nền tảng.
* Tăng cường tính tương tác (interoperability): Cho phép các ứng dụng .NET dễ dàng tương tác với các thành phần phần mềm được viết bằng các công nghệ khác.

**Kiến trúc của .NET Framework:**

.NET Framework có kiến trúc phân lớp, với hai thành phần chính là:

1. Common Language Runtime (CLR): Là trái tim của .NET Framework, một môi trường thực thi ảo chịu trách nhiệm quản lý và thực thi mã .NET.
2. Framework Class Library (FCL) hay Base Class Library (BCL): Một thư viện toàn diện gồm các lớp, giao diện, và kiểu giá trị có thể tái sử dụng, cung cấp các chức năng hệ thống cơ bản và các khối xây dựng cho nhiều loại ứng dụng.

[Hình ảnh của Kiến trúc .NET Framework]

**1.2.1.2. Các thành phần chính của .NET Framework**

* **Common Language Runtime (CLR):**
  + Vai trò: CLR là một máy ảo ứng dụng (application virtual machine) cung cấp môi trường để mã .NET được thực thi. Nó biên dịch mã trung gian (Intermediate Language - IL, hay Microsoft Intermediate Language - MSIL) thành mã máy gốc (native machine code) ngay trước khi thực thi (Just-In-Time compilation - JIT).
  + **Các dịch vụ chính:**
    - Quản lý bộ nhớ (Memory Management): Tự động cấp phát và giải phóng bộ nhớ thông qua cơ chế thu gom rác (Garbage Collection - GC), giúp tránh các lỗi rò rỉ bộ nhớ phổ biến.
    - Bảo mật (Security Management): Cung cấp các cơ chế bảo mật dựa trên bằng chứng (evidence-based security) và bảo mật truy cập mã (Code Access Security - CAS) để kiểm soát quyền của mã thực thi.
    - Xử lý ngoại lệ (Exception Handling): Cung cấp một cơ chế nhất quán để xử lý lỗi và các tình huống ngoại lệ trong quá trình thực thi ứng dụng.
    - Quản lý luồng (Thread Management): Hỗ trợ phát triển ứng dụng đa luồng.
    - Tải và xác minh mã (Code Loading and Verification): Đảm bảo mã được tải đúng cách và tuân thủ các quy tắc an toàn kiểu.
    - Biên dịch JIT (Just-In-Time Compilation): Chuyển đổi MSIL thành mã máy cụ thể cho nền tảng đang chạy.
* **Framework Class Library (FCL) / Base Class Library (BCL):**
  + Giới thiệu: FCL là một tập hợp khổng lồ các lớp, giao diện, và kiểu giá trị có thể tái sử dụng, được tổ chức thành các không gian tên (namespaces). BCL là một tập con cốt lõi của FCL, cung cấp các chức năng cơ bản nhất.
  + **Các namespace quan trọng (ví dụ):**
    - System: Chứa các kiểu dữ liệu cơ bản, các lớp tiện ích hệ thống.
    - System.IO: Hỗ trợ các thao tác vào/ra tập tin và luồng dữ liệu.
    - System.Collections: Cung cấp các cấu trúc dữ liệu như danh sách, hàng đợi, ngăn xếp, bảng băm.
    - System.Data: Hỗ trợ truy cập và quản lý dữ liệu (ADO.NET).
    - System.Windows.Forms: Dành cho phát triển ứng dụng Windows Forms.
    - System.Web: Dành cho phát triển ứng dụng web (ASP.NET).
    - System.Xml: Hỗ trợ xử lý tài liệu XML.
    - System.Linq: Cung cấp Language Integrated Query, cho phép truy vấn dữ liệu từ nhiều nguồn khác nhau.
* **Common Language Specification (CLS):**
  + Mục đích: CLS là một tập hợp các quy tắc và ràng buộc mà các ngôn ngữ .NET phải tuân theo để đảm bảo khả năng tương tác giữa chúng. Nếu một thành phần được viết bằng một ngôn ngữ .NET tuân thủ CLS, nó có thể được sử dụng bởi mã viết bằng một ngôn ngữ .NET khác cũng tuân thủ CLS.
* **Common Type System (CTS):**
  + Mục đích: CTS định nghĩa cách các kiểu dữ liệu được khai báo, sử dụng và quản lý trong CLR. Nó đảm bảo rằng các đối tượng được viết bằng các ngôn ngữ .NET khác nhau có thể tương tác với nhau một cách liền mạch. CTS xác định các quy tắc cho việc khai báo kiểu, định nghĩa kiểu giá trị và kiểu tham chiếu.
* **Các công nghệ con trong .NET Framework:**
  + ASP.NET: Dùng để xây dựng ứng dụng web động và dịch vụ web.
  + ADO.NET: Cung cấp các thành phần để truy cập và thao tác với dữ liệu từ các nguồn dữ liệu khác nhau (ví dụ: SQL Server, Oracle).
  + Windows Presentation Foundation (WPF): Một framework đồ họa tiên tiến hơn Windows Forms để xây dựng giao diện người dùng cho ứng dụng desktop, hỗ trợ đồ họa vector, XAML.
  + Windows Communication Foundation (WCF): Một framework để xây dựng các ứng dụng hướng dịch vụ (service-oriented applications).
  + Windows Workflow Foundation (WF): Để xây dựng các ứng dụng dựa trên quy trình làm việc.
  + Language Integrated Query (LINQ): Cung cấp khả năng truy vấn dữ liệu trực tiếp trong cú pháp ngôn ngữ C# hoặc VB.NET.

**1.2.1.3. Lịch sử phát triển và các phiên bản của .NET Framework**

.NET Framework đã trải qua nhiều phiên bản, mỗi phiên bản mang đến những cải tiến, tính năng mới và hỗ trợ cho các công nghệ mới hơn:

* .NET Framework 1.0 (2002): Phiên bản đầu tiên, giới thiệu CLR, FCL, C# 1.0, VB.NET, ASP.NET Web Forms và Windows Forms.
* .NET Framework 1.1 (2003): Cải tiến và sửa lỗi, hỗ trợ tốt hơn cho ADO.NET và ASP.NET.
* .NET Framework 2.0 (2005): Một bản cập nhật lớn, giới thiệu Generics, các lớp collection mới, cải tiến ASP.NET (Master Pages, Themes), SQL Server 2005 integration, C# 2.0.
* .NET Framework 3.0 (2006): (Tên mã WinFX) Bổ sung bốn công nghệ lớn: WPF, WCF, WF, và Windows CardSpace. Vẫn sử dụng CLR của .NET 2.0.
* .NET Framework 3.5 (2007): Giới thiệu LINQ, ASP.NET AJAX, C# 3.0, VB.NET 9.0. Bao gồm Service Pack 1 cho .NET 2.0 và .NET 3.0.
* .NET Framework 4.0 (2010): Cải tiến CLR (CLR 4), hỗ trợ Dynamic Language Runtime (DLR), Managed Extensibility Framework (MEF), Parallel Computing Library (Task Parallel Library, PLINQ), C# 4.0 (dynamic, named/optional arguments).
* .NET Framework 4.5 (2012): Cải tiến hiệu suất, hỗ trợ async/await trong C# 5.0 và VB.NET 11, cải tiến ASP.NET, WPF, WCF.
* .NET Framework 4.6 (2015): Cải tiến JIT compiler (RyuJIT 64-bit), hỗ trợ C# 6.0.
* .NET Framework 4.7 (2017): Hỗ trợ C# 7.0, cải tiến WPF, ClickOnce.
* .NET Framework 4.8 (2019): Phiên bản cuối cùng của .NET Framework truyền thống, tập trung vào sửa lỗi và cải thiện khả năng truy cập, hiệu suất.

Microsoft khuyến nghị các dự án mới nên sử dụng .NET (trước đây là .NET Core) thay vì .NET Framework cho các ứng dụng mới, đặc biệt là các ứng dụng đa nền tảng, hiệu suất cao và microservices. Tuy nhiên, .NET Framework vẫn được hỗ trợ và phù hợp cho nhiều ứng dụng Windows desktop hiện có và các dự án bảo trì.

**1.2.1.4. Ưu điểm và nhược điểm của .NET Framework**

**Ưu điểm:**

* Năng suất cao: Thư viện lớp phong phú (FCL) và các công cụ phát triển mạnh mẽ (như Visual Studio) giúp tăng tốc độ phát triển.
* Quản lý bộ nhớ tự động: Giảm thiểu lỗi liên quan đến bộ nhớ.
* Hỗ trợ đa ngôn ngữ: Lập trình viên có thể chọn ngôn ngữ phù hợp nhất.
* Bảo mật tốt: Cung cấp các cơ chế bảo mật mạnh mẽ.
* Cộng đồng lớn và tài liệu phong phú: Dễ dàng tìm kiếm sự hỗ trợ và tài liệu học tập.
* Tích hợp tốt với các sản phẩm Microsoft: Dễ dàng làm việc với Windows, SQL Server, Office, v.v.
* Khả năng tương tác: Có thể gọi mã không được quản lý (unmanaged code) và ngược lại.

**Nhược điểm:**

* Chủ yếu cho Windows (đối với .NET Framework): Mặc dù có các dự án như Mono để chạy .NET trên các nền tảng khác, .NET Framework truyền thống được tối ưu hóa và hỗ trợ tốt nhất trên Windows. (Điều này đã được giải quyết với .NET Core và các phiên bản .NET sau này).
* Kích thước lớn: Các ứng dụng .NET Framework có thể yêu cầu cài đặt .NET Framework runtime, làm tăng kích thước triển khai.
* Hiệu suất: Mặc dù CLR và JIT compiler đã được cải tiến nhiều, mã được quản lý có thể chậm hơn mã máy gốc trong một số trường hợp cụ thể.

**1.2.1.5. Tương lai của .NET (Sự chuyển dịch sang .NET Core và .NET 5+)**

Microsoft đã giới thiệu .NET Core như một phiên bản mã nguồn mở, đa nền tảng, và module hóa của .NET. .NET Core được thiết kế lại từ đầu để có hiệu suất cao hơn, nhẹ hơn và có thể chạy trên Windows, macOS, và Linux.

Kể từ .NET 5 (phát hành năm 2020), Microsoft đã hợp nhất .NET Framework và .NET Core thành một nền tảng .NET duy nhất. Các phiên bản .NET mới (như .NET 6, .NET 7, .NET 8,...) tiếp tục phát triển theo hướng này, cung cấp một nền tảng thống nhất cho tất cả các loại ứng dụng .NET.

Đối với các ứng dụng Windows desktop như dự án này, .NET Framework vẫn là một lựa chọn hợp lệ, đặc biệt nếu không có yêu cầu chạy đa nền tảng. Tuy nhiên, việc tìm hiểu về .NET (Core) là quan trọng để nắm bắt xu hướng phát triển trong tương lai.

**1.2.2. Giới thiệu về ứng dụng Windows Forms**

**1.2.2.1. Windows Forms là gì?**

Windows Forms (WinForms) là một framework giao diện người dùng (UI) đồ họa, nằm trong .NET Framework, được sử dụng để phát triển các ứng dụng desktop phong phú (rich client applications) cho hệ điều hành Windows. Nó cung cấp một tập hợp các lớp được quản lý để dễ dàng tạo ra các cửa sổ, hộp thoại, điều khiển (controls) và các yếu tố đồ họa khác.

Mục đích: Windows Forms được thiết kế để đơn giản hóa việc phát triển các ứng dụng Windows truyền thống bằng cách cung cấp một mô hình lập trình dựa trên sự kiện và một bộ công cụ thiết kế trực quan trong Visual Studio.

Vị trí trong .NET Framework: Windows Forms là một phần của Framework Class Library, chủ yếu nằm trong namespace System.Windows.Forms.

**1.2.2.2. Đặc điểm của Windows Forms**

* Lập trình dựa trên sự kiện (Event-driven Programming): Các ứng dụng Windows Forms phản ứng với các hành động của người dùng (như nhấp chuột, nhấn phím) hoặc các sự kiện hệ thống thông qua các trình xử lý sự kiện (event handlers).
* Mô hình điều khiển (Controls): Cung cấp một thư viện lớn các điều khiển UI có thể tái sử dụng như nút bấm (Button), hộp văn bản (TextBox), nhãn (Label), danh sách (ListBox), lưới dữ liệu (DataGridView), v.v. Các điều khiển này có thể được kéo thả vào form trong trình thiết kế của Visual Studio.
* Thiết kế giao diện kéo thả (Drag-and-drop GUI Design): Visual Studio cung cấp một trình thiết kế trực quan mạnh mẽ cho Windows Forms, cho phép lập trình viên dễ dàng thiết kế giao diện bằng cách kéo thả các điều khiển từ Toolbox vào Form và tùy chỉnh thuộc tính của chúng trong cửa sổ Properties.
* Truy cập dễ dàng vào API của Windows: Windows Forms cho phép các nhà phát triển dễ dàng sử dụng các tính năng và API của hệ điều hành Windows.
* Hỗ trợ GDI+: Sử dụng GDI+ (Graphics Device Interface+) để vẽ đồ họa 2D, văn bản và hình ảnh.
* Tính kế thừa trực quan: Cho phép tạo các form và control tùy chỉnh bằng cách kế thừa từ các form và control hiện có.

**1.2.2.3. Các control phổ biến trong Windows Forms**

Windows Forms cung cấp một loạt các control, dưới đây là một số control thường được sử dụng:

* Form: Cửa sổ chính hoặc hộp thoại của ứng dụng.
* Label: Hiển thị văn bản tĩnh.
* Button: Cho phép người dùng thực hiện một hành động khi nhấp vào.
* TextBox: Cho phép người dùng nhập và chỉnh sửa văn bản.
* CheckBox: Cho phép người dùng chọn hoặc bỏ chọn một tùy chọn (true/false).
* RadioButton: Cho phép người dùng chọn một tùy chọn từ một nhóm.
* ListBox: Hiển thị một danh sách các mục mà người dùng có thể chọn.
* ComboBox: Kết hợp TextBox và ListBox, cho phép người dùng nhập hoặc chọn từ danh sách thả xuống.
* PictureBox: Hiển thị hình ảnh.
* DataGridView: Hiển thị dữ liệu dưới dạng bảng (hàng và cột), thường dùng để hiển thị dữ liệu từ cơ sở dữ liệu.
* MenuStrip: Tạo menu chính cho ứng dụng.
* ContextMenuStrip: Tạo menu ngữ cảnh (xuất hiện khi nhấp chuột phải).
* ToolStrip: Tạo thanh công cụ.
* StatusStrip: Hiển thị thông tin trạng thái ở cuối form.
* Timer: Thực hiện một hành động theo định kỳ.
* OpenFileDialog, SaveFileDialog, FolderBrowserDialog: Các hộp thoại để chọn tệp hoặc thư mục.
* Panel, GroupBox, TabControl: Các control chứa (container) để tổ chức các control khác.

**1.2.2.4. Ưu điểm và nhược điểm của Windows Forms**

**Ưu điểm:**

* Dễ học và sử dụng: Đặc biệt đối với những người mới làm quen với lập trình .NET hoặc phát triển ứng dụng desktop. Trình thiết kế kéo thả rất trực quan.
* Phát triển nhanh (Rapid Application Development - RAD): Cho phép tạo mẫu và xây dựng ứng dụng nhanh chóng.
* Hiệu suất tốt: Các ứng dụng Windows Forms thường có hiệu suất tốt trên Windows vì chúng sử dụng các API gốc của hệ điều hành.
* Cộng đồng lớn và ổn định: Là một công nghệ đã tồn tại lâu đời, có nhiều tài liệu, ví dụ và sự hỗ trợ từ cộng đồng.
* Nhiều control của bên thứ ba: Có sẵn nhiều bộ control thương mại (như DevExpress, Telerik) và mã nguồn mở giúp mở rộng chức năng.

**Nhược điểm:**

* Hạn chế về tùy biến giao diện nâng cao: So với các công nghệ UI hiện đại hơn như WPF hay UWP, việc tạo giao diện phức tạp, hiệu ứng đồ họa cao cấp hoặc tùy biến sâu các control trong Windows Forms có thể khó khăn hơn.
* Không hỗ trợ XAML: Việc định nghĩa giao diện chủ yếu thông qua trình thiết kế hoặc code-behind, không có ngôn ngữ khai báo UI mạnh mẽ như XAML.
* Chỉ dành cho Windows: Các ứng dụng Windows Forms chỉ chạy trên hệ điều hành Windows.
* Công nghệ cũ hơn: Mặc dù vẫn được hỗ trợ, Microsoft đang tập trung phát triển các công nghệ UI mới hơn cho .NET (như WPF, MAUI). Tuy nhiên, đối với nhiều ứng dụng nghiệp vụ nội bộ, Windows Forms vẫn là một lựa chọn rất tốt.

**1.2.2.5. Các ứng dụng tiêu biểu được xây dựng bằng Windows Forms**

Windows Forms đã và đang được sử dụng để xây dựng một loạt các ứng dụng desktop, bao gồm:

* Các ứng dụng quản lý doanh nghiệp (ERP, CRM ở quy mô nhỏ và vừa).
* Các công cụ tiện ích hệ thống.
* Phần mềm kế toán, quản lý kho, quản lý bán hàng.
* Các ứng dụng nhập liệu và xử lý dữ liệu.
* Giao diện người dùng cho các hệ thống điều khiển công nghiệp.
* Nhiều phần của chính Visual Studio cũng được xây dựng bằng công nghệ tương tự Windows Forms.

**1.2.2.6. Tại sao chọn Windows Forms cho dự án "Ứng dụng quản lý chấm điểm Đội Cờ Đỏ"?**

Đối với dự án "Ứng dụng quản lý chấm điểm Đội Cờ Đỏ", việc lựa chọn Windows Forms mang lại một số lợi thế:

* Phù hợp với loại ứng dụng: Đây là một ứng dụng nghiệp vụ, tập trung vào chức năng xử lý dữ liệu và giao diện người dùng không đòi hỏi quá nhiều hiệu ứng đồ họa phức tạp.
* Phát triển nhanh: Giúp nhanh chóng xây dựng giao diện và các chức năng cơ bản.
* Dễ tiếp cận: Nếu đội ngũ phát triển quen thuộc với .NET và C#, việc làm quen và phát triển với Windows Forms sẽ không quá khó khăn.
* Ổn định và đã được kiểm chứng: Là một công nghệ trưởng thành, đảm bảo tính ổn định cho ứng dụng.
* Tích hợp tốt với DevExpress: Bộ công cụ DevExpress cung cấp các control mạnh mẽ cho Windows Forms, giúp xây dựng giao diện chuyên nghiệp và giàu tính năng hơn so với các control chuẩn.

Mặc dù có các công nghệ UI mới hơn, Windows Forms vẫn là một lựa chọn thực tế và hiệu quả cho các ứng dụng desktop tập trung vào nghiệp vụ như dự án này, đặc biệt khi ưu tiên tốc độ phát triển và sự quen thuộc của đội ngũ.

**1.2.3. Giới thiệu về ngôn ngữ lập trình C#**

**1.2.3.1. C# là gì?**

C# (đọc là "C sharp") là một ngôn ngữ lập trình hiện đại, hướng đối tượng, an toàn kiểu (type-safe) được phát triển bởi Microsoft như một phần của sáng kiến .NET. C# được thiết kế để xây dựng một loạt các ứng dụng chạy trên .NET Framework (và các phiên bản .NET sau này).

Mối quan hệ với .NET: C# là ngôn ngữ chính và phổ biến nhất để phát triển trên nền tảng .NET. Nó được thiết kế để tận dụng tối đa các tính năng của CLR và FCL. Mã C# được biên dịch thành Mã Trung Gian (IL), sau đó được thực thi bởi CLR.

**1.2.3.2. Lịch sử hình thành và phát triển của C#**

* Người tạo ra: C# được thiết kế chủ yếu bởi Anders Hejlsberg, một kiến trúc sư phần mềm nổi tiếng người Đan Mạch tại Microsoft, người trước đó đã có công trong việc thiết kế Turbo Pascal, Delphi và J++.
* Ra đời: C# được giới thiệu lần đầu vào năm 2000 và chính thức phát hành cùng với .NET Framework 1.0 vào năm 2002.
* Các phiên bản chính và những thay đổi quan trọng: C# đã phát triển qua nhiều phiên bản, mỗi phiên bản bổ sung các tính năng ngôn ngữ mới mạnh mẽ:
  + C# 1.0 (2002): Phiên bản cơ bản, hỗ trợ các khái niệm OOP, properties, events, delegates, attributes.
  + C# 2.0 (2005 - .NET Framework 2.0): Giới thiệu Generics, partial types, anonymous methods, nullable types, iterators. Đây là một bước tiến lớn về khả năng biểu diễn và an toàn kiểu.
  + C# 3.0 (2007 - .NET Framework 3.5): Giới thiệu LINQ (Language Integrated Query), implicitly typed local variables (var), object and collection initializers, anonymous types, lambda expressions, extension methods. Các tính năng này làm cho việc truy vấn và thao tác dữ liệu trở nên tự nhiên hơn.
  + C# 4.0 (2010 - .NET Framework 4.0): Giới thiệu dynamic binding (dynamic keyword), named and optional arguments, generic covariance and contravariance.
  + C# 5.0 (2012 - .NET Framework 4.5): Giới thiệu asynchronous programming (async and await keywords), giúp đơn giản hóa việc viết mã bất đồng bộ, cải thiện khả năng đáp ứng của ứng dụng.
  + C# 6.0 (2015 - .NET Framework 4.6): Tập trung vào việc làm cho mã ngắn gọn hơn và dễ đọc hơn với các tính năng như auto-property initializers, string interpolation, nameof operator, null-conditional operator (?.).
  + C# 7.0 (2017 - .NET Framework 4.7, .NET Core 2.0): Giới thiệu tuples, pattern matching, local functions, out variables, ref returns and locals.
    - C# 7.1, 7.2, 7.3: Các bản cập nhật nhỏ bổ sung thêm tính năng.
  + C# 8.0 (2019 - .NET Core 3.0): Giới thiệu nullable reference types (giúp giảm lỗi NullReferenceException), async streams, default interface methods, ranges and indices, switch expressions.
  + C# 9.0 (2020 - .NET 5): Giới thiệu records, init-only setters, top-level statements, target-typed new expressions, pattern matching enhancements.
  + C# 10 (2021 - .NET 6): Global using directives, file-scoped namespaces, record structs, interpolated string improvements.
  + C# 11 (2022 - .NET 7): Generic math, raw string literals, list patterns, required members.
  + C# 12 (2023 - .NET 8): Primary constructors, collection expressions, default lambda parameters, alias any type.

Sự phát triển không ngừng của C# cho thấy cam kết của Microsoft trong việc giữ cho ngôn ngữ này luôn hiện đại, mạnh mẽ và đáp ứng nhu cầu của các nhà phát triển.

**1.2.3.3. Đặc điểm nổi bật của ngôn ngữ C#**

* Hướng đối tượng (Object-Oriented Programming - OOP): C# hỗ trợ đầy đủ các nguyên lý OOP:
  + Tính đóng gói (Encapsulation): Che giấu thông tin bên trong đối tượng và chỉ cho phép tương tác thông qua các giao diện công khai.
  + Tính kế thừa (Inheritance): Cho phép tạo lớp mới (lớp con) dựa trên lớp đã có (lớp cha), tái sử dụng và mở rộng mã.
  + Tính đa hình (Polymorphism): Cho phép đối tượng có thể thể hiện nhiều hình thái khác nhau, thường được thực hiện thông qua overriding và overloading phương thức.
  + Tính trừu tượng (Abstraction): Tập trung vào các đặc điểm thiết yếu của đối tượng và bỏ qua các chi tiết không cần thiết.
* Mạnh mẽ và linh hoạt: C# kết hợp sức mạnh của C++ với sự đơn giản của Visual Basic. Nó cung cấp quyền truy cập cấp thấp (ví dụ: thông qua con trỏ trong ngữ cảnh unsafe) khi cần thiết, nhưng chủ yếu hoạt động trong môi trường được quản lý an toàn.
* An toàn kiểu (Type-safe): C# yêu cầu các biến phải được khai báo với một kiểu cụ thể và trình biên dịch kiểm tra tính tương thích kiểu trong quá trình biên dịch, giúp phát hiện lỗi sớm.
* Hỗ trợ quản lý bộ nhớ tự động (Garbage Collection): CLR tự động quản lý việc cấp phát và thu hồi bộ nhớ, giúp lập trình viên không phải lo lắng về việc giải phóng bộ nhớ thủ công và tránh các lỗi như rò rỉ bộ nhớ.
* Tương thích với các ngôn ngữ .NET khác: Nhờ CLS và CTS, các thành phần được viết bằng C# có thể dễ dàng tương tác với các thành phần được viết bằng VB.NET, F#, v.v.
* Cú pháp hiện đại, dễ học (so với C++): Cú pháp của C# thừa hưởng nhiều từ C và C++, nhưng đã được đơn giản hóa và loại bỏ một số tính năng phức tạp và dễ gây lỗi của C++.
* Hỗ trợ LINQ (Language Integrated Query): Cho phép viết truy vấn dữ liệu (đến đối tượng, XML, cơ sở dữ liệu) trực tiếp trong mã C# một cách tự nhiên và mạnh mẽ.
* Hỗ trợ lập trình bất đồng bộ (Async/Await): Giúp viết mã không chặn (non-blocking) một cách dễ dàng, cải thiện khả năng đáp ứng và hiệu suất của ứng dụng, đặc biệt là các ứng dụng có thao tác I/O hoặc tính toán dài.
* Thư viện chuẩn phong phú: Được thừa hưởng từ .NET FCL, cung cấp nhiều chức năng sẵn có.
* Component-oriented: Được thiết kế để dễ dàng tạo và sử dụng các thành phần phần mềm (software components).

**1.2.3.4. Các ứng dụng tiêu biểu của C#**

C# là một ngôn ngữ đa năng và được sử dụng trong nhiều lĩnh vực phát triển phần mềm:

* Ứng dụng Desktop:
  + Windows Forms: Xây dựng các ứng dụng Windows truyền thống.
  + Windows Presentation Foundation (WPF): Xây dựng các ứng dụng Windows với giao diện người dùng hiện đại và phong phú hơn.
  + Universal Windows Platform (UWP): Phát triển ứng dụng cho các thiết bị Windows 10/11.
* Ứng dụng Web:
  + ASP.NET (Web Forms, MVC, Razor Pages, Blazor): Xây dựng các trang web động, ứng dụng web và dịch vụ web mạnh mẽ.
* Phát triển Game:
  + Unity: C# là ngôn ngữ kịch bản chính cho Unity, một trong những game engine phổ biến nhất thế giới, được sử dụng để tạo game cho PC, console, di động và VR/AR.
* Ứng dụng di động:
  + Xamarin (nay là một phần của .NET MAUI): Cho phép phát triển ứng dụng di động đa nền tảng (iOS, Android, Windows) bằng C# và chia sẻ mã nguồn giữa các nền tảng.
  + .NET MAUI (Multi-platform App UI): Sự phát triển tiếp theo của Xamarin.Forms, cho phép xây dựng ứng dụng gốc đa nền tảng từ một cơ sở mã duy nhất.
* Ứng dụng Cloud: Phát triển và triển khai các ứng dụng và dịch vụ trên các nền tảng đám mây như Microsoft Azure.
* Trí tuệ nhân tạo (AI) và Học máy (Machine Learning): Với các thư viện như ML.NET.
* Internet of Things (IoT): Phát triển ứng dụng cho các thiết bị IoT với .NET IoT Libraries.
* Microservices: Xây dựng các kiến trúc microservices.

**1.2.3.5. Tại sao chọn C# cho dự án "Ứng dụng quản lý chấm điểm Đội Cờ Đỏ"?**

Lựa chọn C# cho dự án này là một quyết định hợp lý vì nhiều lý do:

* Ngôn ngữ chính của .NET: Vì dự án sử dụng .NET Framework và Windows Forms, C# là lựa chọn tự nhiên và được hỗ trợ tốt nhất.
* Năng suất cao: Các tính năng hiện đại của C# (như LINQ, async/await, properties, events) kết hợp với Visual Studio và FCL giúp tăng tốc độ phát triển.
* Dễ học và sử dụng: Cú pháp rõ ràng, dễ đọc và cộng đồng hỗ trợ lớn giúp việc phát triển và giải quyết vấn đề trở nên dễ dàng hơn.
* Hướng đối tượng mạnh mẽ: Giúp xây dựng mã nguồn có cấu trúc tốt, dễ bảo trì và mở rộng.
* An toàn và ổn định: Tính năng an toàn kiểu và quản lý bộ nhớ tự động giúp giảm thiểu lỗi và tăng độ tin cậy của ứng dụng.
* Tích hợp tốt với SQL Server và DevExpress: C# cung cấp các cơ chế thuận tiện để làm việc với cơ sở dữ liệu SQL Server (thông qua ADO.NET hoặc các ORM như Entity Framework) và tích hợp mượt mà với các control của DevExpress.
* Tóm lại, sự kết hợp giữa .NET Framework, Windows Forms và ngôn ngữ C# cung cấp một nền tảng vững chắc, hiệu quả và quen thuộc để phát triển ứng dụng quản lý chấm điểm cho Đội Cờ Đỏ, đáp ứng các yêu cầu về chức năng, hiệu suất và khả năng bảo trì.

A computer screen with a graduation cap

AI-generated content may be incorrect.

Hình 1.1. Tên hình ảnh thứ nhất của chương 1

**1.3. Giới thiệu về Hệ quản trị Cơ sở dữ liệu SQL Server 2022**

Trong thế giới công nghệ thông tin không ngừng phát triển, việc quản lý và khai thác dữ liệu hiệu quả là yếu tố sống còn đối với mọi tổ chức và ứng dụng. Microsoft SQL Server, với lịch sử phát triển lâu dài và không ngừng cải tiến, đã khẳng định vị thế là một trong những hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) hàng đầu thế giới. Phiên bản SQL Server 2022 tiếp tục kế thừa những điểm mạnh của các phiên bản tiền nhiệm, đồng thời mang đến nhiều cải tiến đột phá về hiệu năng, bảo mật, tính sẵn sàng cao và đặc biệt là khả năng tích hợp sâu rộng với nền tảng đám mây Azure. Phần này sẽ cung cấp cái nhìn tổng quan về SQL Server 2022, bao gồm lịch sử phát triển, các đặc điểm nổi bật, ưu nhược điểm và những ứng dụng phổ biến.

**1.3.1. SQL Server 2022 là gì?**

Microsoft SQL Server 2022 là một hệ quản trị cơ sở dữ liệu quan hệ mạnh mẽ, được thiết kế để lưu trữ, truy xuất và quản lý dữ liệu một cách an toàn và hiệu quả. Nó cung cấp một nền tảng dữ liệu toàn diện, hỗ trợ đa dạng các ứng dụng từ quy mô nhỏ đến các hệ thống doanh nghiệp lớn, phức tạp với yêu cầu cao về hiệu năng, tính sẵn sàng và bảo mật.

SQL Server 2022 không chỉ đơn thuần là một RDBMS truyền thống. Nó đã phát triển thành một nền tảng dữ liệu thông minh, tích hợp các khả năng phân tích dữ liệu, báo cáo, học máy (Machine Learning) và đặc biệt là các tính năng kết nối và quản lý dữ liệu lai (hybrid) giữa môi trường tại chỗ (on-premises) và đám mây Microsoft Azure. Điều này cho phép các tổ chức tận dụng được sự linh hoạt và sức mạnh của cả hai môi trường.

Các đặc tính chính của SQL Server 2022 bao gồm:

* **Hiệu năng thông minh (Intelligent Performance):** Tự động tối ưu hóa truy vấn và cải thiện hiệu suất mà không cần thay đổi mã ứng dụng.
* **Bảo mật nâng cao (Enhanced Security):** Cung cấp các lớp bảo vệ dữ liệu đa tầng, từ mã hóa, kiểm soát truy cập đến các tính năng chống thất thoát dữ liệu tiên tiến.
* **Tính sẵn sàng cao và Phục hồi sau thảm họa (High Availability and Disaster Recovery):** Đảm bảo hoạt động liên tục của hệ thống và khả năng phục hồi dữ liệu nhanh chóng khi có sự cố.
* **Tích hợp Azure (Azure-enabled):** Kết nối liền mạch với các dịch vụ Azure như Azure Synapse Link, Azure Purview, và Azure SQL Managed Instance cho các kịch bản DR và phân tích dữ liệu.
* **Linh hoạt trên nhiều nền tảng (Platform Flexibility):** Hỗ trợ triển khai trên Windows, Linux và trong các container Docker.
* **Nền tảng dữ liệu hợp nhất (Unified Data Platform):** Khả năng truy vấn dữ liệu từ nhiều nguồn khác nhau thông qua PolyBase và tích hợp với các công cụ Business Intelligence (BI).

SQL Server 2022 hướng đến việc cung cấp một giải pháp quản lý dữ liệu toàn diện, giúp doanh nghiệp đưa ra quyết định dựa trên dữ liệu (data-driven decisions) và đổi mới nhanh chóng hơn.

**1.3.2. Lịch sử phát triển của SQL Server**

Microsoft SQL Server có một lịch sử phát triển phong phú, bắt đầu từ cuối những năm 1980.

* **Khởi đầu (Cuối 1980s - Đầu 1990s):** SQL Server ban đầu được phát triển dựa trên mã nguồn của Sybase SQL Server. Phiên bản đầu tiên, SQL Server 1.0 cho OS/2, được phát hành vào năm 1989. Microsoft sau đó đã phát triển phiên bản riêng cho Windows NT.
* **SQL Server 6.0/6.5 (Giữa 1990s):** Đánh dấu bước chuyển mình quan trọng với việc Microsoft tự phát triển và tối ưu hóa cho Windows NT, bổ sung nhiều tính năng quản trị và cải thiện hiệu năng.
* **SQL Server 7.0 (1998):** Một bản viết lại gần như hoàn toàn, thay đổi kiến trúc lưu trữ, giới thiệu công cụ quản lý Enterprise Manager, và cải thiện đáng kể khả năng mở rộng.
* **SQL Server 2000 (2000):** Tăng cường tính ổn định, khả năng mở rộng, hỗ trợ XML, và giới thiệu các phiên bản 64-bit.
* **SQL Server 2005 (2005):** Một bản phát hành lớn với nhiều tính năng mới như Common Language Runtime (CLR) integration, SQL Server Management Studio (SSMS), Service Broker, Database Mirroring, và các dịch vụ BI (SSIS, SSAS, SSRS) được tích hợp sâu hơn.
* **SQL Server 2008 và 2008 R2 (2008, 2010):** Tập trung vào quản lý dựa trên chính sách (Policy-Based Management), nén dữ liệu, cải thiện hiệu năng, và giới thiệu các tính năng như Resource Governor, PowerPivot (trong R2).
* **SQL Server 2012 (2012):** Giới thiệu AlwaysOn Availability Groups, Columnstore Indexes, và cải tiến đáng kể cho các công cụ BI.
* **SQL Server 2014 (2014):** Tập trung vào In-Memory OLTP (Hekaton), cải thiện hiệu năng cho các ứng dụng giao dịch trực tuyến cường độ cao.
* **SQL Server 2016 (2016):** Đánh dấu một bước ngoặt lớn với việc hỗ trợ Linux, giới thiệu Query Store, Always Encrypted, R Services, PolyBase cho phép truy vấn Hadoop.
* **SQL Server 2017 (2017):** Tiếp tục củng cố hỗ trợ Linux và container, giới thiệu Adaptive Query Processing, Graph Database, Python integration.
* **SQL Server 2019 (2019):** Mang đến Big Data Clusters, Intelligent Query Processing nâng cao, Data Virtualization mở rộng, và nhiều cải tiến về bảo mật và hiệu năng.
* **SQL Server 2022 (Phát hành tháng 11 năm 2022):** Phiên bản mới nhất, tập trung mạnh mẽ vào tích hợp Azure, cải tiến hiệu năng thông minh, tăng cường bảo mật với Ledger, và nâng cao khả năng phục hồi sau thảm họa.

Qua mỗi phiên bản, SQL Server không ngừng được Microsoft đầu tư và phát triển, đáp ứng các yêu cầu ngày càng phức tạp của thị trường và công nghệ.

**1.3.3. Đặc điểm nổi bật của SQL Server 2022**

SQL Server 2022 mang đến nhiều tính năng và cải tiến quan trọng, giúp nó trở thành một nền tảng dữ liệu hiện đại và mạnh mẽ:

1. **Tích hợp Azure liền mạch (Azure-enabled Capabilities):**
   * **Azure Synapse Link for SQL Server:** Cho phép phân tích dữ liệu hoạt động (operational data) gần thời gian thực từ SQL Server 2022 trong Azure Synapse Analytics mà không ảnh hưởng đến hiệu năng của CSDL nguồn.
   * **Azure Purview Integration:** Tự động quét SQL Server 2022 để khám phá dữ liệu, phân loại dữ liệu nhạy cảm và theo dõi dòng dữ liệu (data lineage), giúp quản trị dữ liệu và tuân thủ quy định hiệu quả hơn.
   * **Disaster Recovery to Azure SQL Managed Instance:** Cung cấp một liên kết Managed Instance cho phép thiết lập môi trường phục hồi sau thảm họa trên Azure một cách dễ dàng, giảm thiểu thời gian ngừng hoạt động và chi phí hạ tầng DR.
   * **Azure Active Directory Authentication:** Hỗ trợ xác thực bằng Azure AD cho cả SQL Server trên Windows và Linux, đơn giản hóa việc quản lý danh tính và truy cập.
2. **Hiệu năng thông minh thế hệ mới (Next generation Intelligent Query Processing - IQP):**
   * **Query Store enhancements:** Query Store được bật mặc định cho các CSDL mới, hỗ trợ cả read replicas trong AlwaysOn Availability Groups.
   * **Parameter Sensitive Plan (PSP) optimization:** Tối ưu hóa các kế hoạch truy vấn nhạy cảm với tham số, giải quyết vấn đề "parameter sniffing" một cách hiệu quả hơn.
   * **Degree of Parallelism (DOP) feedback:** Tự động điều chỉnh mức độ song song cho các truy vấn lặp lại để tối ưu hóa hiệu suất.
   * **Cardinality Estimation (CE) feedback:** Cải thiện độ chính xác của ước tính số lượng bản ghi, dẫn đến kế hoạch truy vấn tốt hơn.
   * **Optimized plan forcing:** Cải thiện khả năng ép buộc kế hoạch truy vấn tối ưu.
3. **Bảo mật và Tuân thủ nâng cao (Enhanced Security and Compliance):**
   * **SQL Server Ledger:** Giới thiệu tính năng sổ cái bất biến (immutable ledger) cho CSDL. Các bảng sổ cái theo dõi lịch sử thay đổi dữ liệu một cách mật mã, giúp phát hiện và ngăn chặn giả mạo dữ liệu, rất quan trọng cho các ứng dụng yêu cầu tính toàn vẹn và kiểm toán cao.
   * **Always Encrypted with secure enclaves enhancements:** Hỗ trợ các truy vấn bí mật (confidential queries) phong phú hơn trên dữ liệu được mã hóa.
   * **Hỗ trợ TLS 1.3:** Tăng cường bảo mật cho kết nối mạng.
   * **Certificate management improvements:** Đơn giản hóa việc quản lý chứng chỉ.
4. **Tính sẵn sàng cao và Khả năng phục hồi (High Availability and Resiliency):**
   * **Contained Availability Groups:** Đơn giản hóa việc quản lý và di chuyển các Availability Groups bằng cách chứa các đối tượng metadata (logins, permissions, SQL Agent jobs) bên trong Availability Group.
   * **Cross-platform snapshot backups:** Hỗ trợ sao lưu snapshot nhất quán trên các nền tảng khác nhau.
5. **Quản lý dữ liệu và Phát triển (Data Management and Development):**
   * **Cải tiến T-SQL:** Bổ sung nhiều hàm và tính năng mới cho Transact-SQL, ví dụ như JSON\_OBJECT, IS DISTINCT FROM, GREATEST/LEAST, các hàm xử lý thời gian (DATETRUNC), và các hàm bit manipulation.
   * **Data Virtualization (PolyBase):** Cho phép truy vấn dữ liệu từ các nguồn dữ liệu bên ngoài như Oracle, Teradata, MongoDB, Cosmos DB, và các kho dữ liệu S3-compatible object storage mà không cần di chuyển dữ liệu. Hỗ trợ REST API cho phép truy vấn dữ liệu từ các API.
   * **Nâng cấp cho Machine Learning Services:** Cải tiến runtime cho R, Python, và Java.
6. **Linh hoạt nền tảng (Platform Choice):** SQL Server 2022 tiếp tục hỗ trợ triển khai trên Windows, các bản phân phối Linux phổ biến (RHEL, SLES, Ubuntu), và trong các container Docker/Kubernetes, mang lại sự linh hoạt tối đa cho việc lựa chọn môi trường triển khai.

**1.3.4. Các phiên bản của SQL Server 2022**

SQL Server 2022 được cung cấp với nhiều phiên bản khác nhau để đáp ứng nhu cầu và ngân sách đa dạng của người dùng:

* **Enterprise Edition:** Phiên bản cao cấp nhất, cung cấp đầy đủ các tính năng về hiệu năng, bảo mật, tính sẵn sàng cao, và Business Intelligence. Phù hợp cho các ứng dụng quan trọng, quy mô lớn với yêu cầu khắt khe.
* **Standard Edition:** Cung cấp các chức năng CSDL cốt lõi, báo cáo và phân tích cơ bản. Phù hợp cho các phòng ban, doanh nghiệp vừa và nhỏ, hoặc các ứng dụng không yêu cầu tất cả các tính năng cao cấp của Enterprise.
* **Web Edition:** Một lựa chọn chi phí thấp cho các nhà cung cấp dịch vụ hosting web và các ứng dụng web.
* **Developer Edition:** Cung cấp đầy đủ tính năng của Enterprise Edition, nhưng chỉ dành cho mục đích phát triển và kiểm thử (không được sử dụng trong môi trường production). Đây là một công cụ tuyệt vời cho các nhà phát triển.
* **Express Edition:** Phiên bản miễn phí, giới hạn về kích thước CSDL (10GB mỗi CSDL), bộ nhớ và CPU. Rất phù hợp cho các ứng dụng nhỏ, học tập, hoặc phát triển các ứng dụng desktop và web quy mô nhỏ.

**1.3.5. Ưu điểm của SQL Server**

SQL Server nói chung và SQL Server 2022 nói riêng mang lại nhiều lợi thế cho các tổ chức và nhà phát triển:

* **Hiệu năng và Khả năng mở rộng (Scalability and Performance):** SQL Server được biết đến với khả năng xử lý khối lượng công việc lớn và có thể mở rộng để đáp ứng nhu cầu tăng trưởng dữ liệu và người dùng. Các tính năng như In-Memory OLTP, Columnstore Indexes, và Intelligent Query Processing giúp tối ưu hóa hiệu suất cho nhiều loại ứng dụng.
* **Bảo mật mạnh mẽ (Robust Security Features):** Cung cấp một bộ tính năng bảo mật toàn diện bao gồm mã hóa dữ liệu (Transparent Data Encryption, Always Encrypted), kiểm soát truy cập chi tiết, kiểm toán, và các tính năng mới như SQL Server Ledger.
* **Tính sẵn sàng cao và Phục hồi sau thảm họa (High Availability and Disaster Recovery):** Các công nghệ như AlwaysOn Availability Groups, Failover Cluster Instances, và khả năng tích hợp DR với Azure SQL Managed Instance đảm bảo thời gian hoạt động tối đa và bảo vệ dữ liệu quan trọng.
* **Công cụ Business Intelligence và Phân tích dữ liệu tích hợp (Integrated BI and Analytics):** SQL Server cung cấp một nền tảng BI mạnh mẽ với SQL Server Analysis Services (SSAS), SQL Server Reporting Services (SSRS), và SQL Server Integration Services (SSIS), cùng khả năng tích hợp Machine Learning Services.
* **Công cụ quản lý và phát triển toàn diện (Comprehensive Tooling):** SQL Server Management Studio (SSMS) và Azure Data Studio cung cấp môi trường đồ họa mạnh mẽ cho việc quản trị, phát triển, và giám sát CSDL.
* **Tích hợp tốt với hệ sinh thái Microsoft:** SQL Server hoạt động liền mạch với các sản phẩm và công nghệ khác của Microsoft như .NET Framework, Windows Server, Active Directory, Power BI, và Azure.
* **Hỗ trợ đa nền tảng:** Khả năng chạy trên Linux và container mở rộng sự linh hoạt trong triển khai.
* **Cộng đồng lớn và Tài liệu phong phú:** Có một cộng đồng người dùng và chuyên gia SQL Server đông đảo trên toàn cầu, cùng với nguồn tài liệu, hướng dẫn và khóa học phong phú từ Microsoft và các bên thứ ba.

**1.3.6. Nhược điểm của SQL Server**

Bên cạnh những ưu điểm vượt trội, SQL Server cũng có một số hạn chế cần cân nhắc:

* **Chi phí bản quyền (Licensing Costs):** Đối với các phiên bản thương mại như Enterprise và Standard, chi phí bản quyền có thể khá cao, đặc biệt đối với các doanh nghiệp nhỏ hoặc các dự án có ngân sách hạn chế. Tuy nhiên, các phiên bản Developer (miễn phí cho phát triển) và Express (miễn phí cho production quy mô nhỏ) có thể là giải pháp thay thế.
* **Độ phức tạp (Complexity):** Với bộ tính năng phong phú và nhiều tùy chọn cấu hình, việc quản trị và tối ưu hóa SQL Server có thể trở nên phức tạp, đòi hỏi kiến thức và kinh nghiệm chuyên sâu, đặc biệt là đối với các hệ thống lớn.
* **Yêu cầu tài nguyên hệ thống (Resource Intensive):** Để đạt hiệu suất tối ưu, đặc biệt là với các phiên bản cao cấp và khối lượng công việc lớn, SQL Server có thể yêu cầu phần cứng máy chủ mạnh mẽ với dung lượng RAM, CPU và lưu trữ đáng kể.
* **Mặc dù hỗ trợ Linux tốt, vẫn có cảm giác "Windows-centric":** Dù Microsoft đã nỗ lực rất lớn để đưa SQL Server lên Linux và container, một số công cụ hoặc thói quen quản trị truyền thống vẫn có xu hướng tập trung vào môi trường Windows. Tuy nhiên, điều này đang dần thay đổi.

**1.3.7. Các ứng dụng phổ biến của SQL Server**

SQL Server được sử dụng rộng rãi trong nhiều ngành nghề và loại hình ứng dụng khác nhau do tính linh hoạt, hiệu năng và độ tin cậy của nó:

* **Hệ thống hoạch định nguồn lực doanh nghiệp (ERP Systems):** Nhiều giải pháp ERP lớn sử dụng SQL Server làm CSDL nền tảng để quản lý dữ liệu tài chính, nhân sự, sản xuất, chuỗi cung ứng, v.v.
* **Hệ thống quản lý quan hệ khách hàng (CRM Systems):** Các ứng dụng CRM lưu trữ và phân tích dữ liệu khách hàng, lịch sử tương tác, và các chiến dịch marketing thường dựa trên SQL Server.
* **Ứng dụng thương mại điện tử (E-commerce Platforms):** Các trang web bán hàng trực tuyến với lượng giao dịch lớn, quản lý sản phẩm, đơn hàng và thông tin khách hàng.
* **Kho dữ liệu (Data Warehousing) và Business Intelligence (BI):** Xây dựng các kho dữ liệu để tổng hợp, phân tích dữ liệu từ nhiều nguồn, tạo báo cáo và dashboard hỗ trợ ra quyết định.
* **Ứng dụng tài chính và ngân hàng:** Xử lý giao dịch, quản lý tài khoản, phân tích rủi ro, và các ứng dụng yêu cầu tính toàn vẹn và bảo mật dữ liệu cao.
* **Ứng dụng y tế và chăm sóc sức khỏe:** Quản lý hồ sơ bệnh án điện tử (EMR/EHR), hệ thống thông tin bệnh viện (HIS), phân tích dữ liệu y tế.
* **Các ứng dụng web và dịch vụ trực tuyến quy mô lớn:** Cung cấp backend CSDL cho các trang web có lưu lượng truy cập cao, mạng xã hội, và các dịch vụ trực tuyến khác.
* **Ứng dụng quản lý nội dung (CMS) và cổng thông tin doanh nghiệp.**
* **Hệ thống quản lý học tập (LMS) và các ứng dụng giáo dục.**

**1.3.8. Đánh giá sự phù hợp của SQL Server 2022 cho dự án "Xây dựng ứng dụng Cờ Đỏ"**

Việc lựa chọn SQL Server 2022 cho dự án "Xây dựng ứng dụng Cờ Đỏ" phụ thuộc vào quy mô, yêu cầu về tính năng và định hướng phát triển của ứng dụng.

**Trường hợp SQL Server 2022 có thể phù hợp:**

1. **Nhiều người dùng đồng thời:** Nếu ứng dụng cần hỗ trợ nhiều thành viên Đội Cờ Đỏ, giáo viên, và ban giám hiệu cùng lúc truy cập, cập nhật và xem dữ liệu từ các thiết bị khác nhau, mô hình client-server của SQL Server sẽ phù hợp hơn SQLite.
2. **Dữ liệu tập trung và quản lý tập trung:** Khi cần một kho dữ liệu trung tâm để lưu trữ tất cả thông tin liên quan đến hoạt động Đội Cờ Đỏ, giúp dễ dàng quản lý, sao lưu, và bảo mật.
3. **Yêu cầu bảo mật cao:** Nếu dữ liệu về học sinh và các đánh giá cần được bảo vệ với các cơ chế bảo mật mạnh mẽ, phân quyền chi tiết, SQL Server cung cấp nhiều tính năng vượt trội.
4. **Khả năng mở rộng trong tương lai:** Nếu dự kiến ứng dụng sẽ phát triển về số lượng người dùng, lượng dữ liệu, hoặc cần tích hợp thêm các tính năng phân tích, báo cáo phức tạp, SQL Server có khả năng mở rộng tốt hơn.
5. **Tích hợp với các hệ thống khác của trường:** Nếu trường học đã có sẵn hạ tầng Microsoft hoặc cần tích hợp dữ liệu từ ứng dụng Cờ Đỏ với các hệ thống quản lý học sinh khác, SQL Server có thể là lựa chọn tự nhiên.
6. **Yêu cầu về báo cáo và phân tích:** Nếu cần xây dựng các báo cáo động, dashboard phân tích tình hình nề nếp, SQL Server Reporting Services (SSRS) hoặc tích hợp với Power BI sẽ rất hữu ích.

**Những điều cần cân nhắc:**

* **Chi phí và Hạ tầng:** Ngay cả khi sử dụng phiên bản Express (miễn phí nhưng có giới hạn) hoặc Developer (miễn phí cho phát triển), việc triển khai và quản trị một máy chủ SQL Server đòi hỏi nhiều nỗ lực và tài nguyên hơn so với SQLite (là một tệp CSDL nhúng). Nếu ứng dụng chỉ dành cho một vài người dùng cục bộ, SQLite có thể đơn giản và hiệu quả hơn.
* **Độ phức tạp triển khai:** Việc cài đặt và cấu hình SQL Server phức tạp hơn.
* **Mục tiêu của ứng dụng:** Nếu ứng dụng Cờ Đỏ chỉ là một công cụ ghi chép cá nhân hoặc cho một nhóm rất nhỏ, hoạt động chủ yếu offline trên một thiết bị, thì SQL Server có thể là quá mức cần thiết (overkill).

**Khuyến nghị:**

* **Đối với giai đoạn phát triển và học tập:** SQL Server Developer Edition là một lựa chọn tuyệt vời vì nó miễn phí và cung cấp đầy đủ tính năng.
* **Đối với triển khai quy mô nhỏ hoặc thử nghiệm:** SQL Server Express Edition có thể đủ dùng.
* **Nếu ứng dụng yêu cầu các tính năng của một hệ thống client-server thực thụ, hỗ trợ nhiều người dùng và dữ liệu tập trung:** SQL Server (Standard hoặc cao hơn nếu cần) là một lựa chọn mạnh mẽ và đáng cân nhắc cho dự án "Xây dựng ứng dụng Cờ Đỏ", đặc biệt nếu dự án được phát triển trên nền tảng .NET như đã đề cập trong đề cương.

Tóm lại, SQL Server 2022 là một hệ quản trị CSDL mạnh mẽ, đa năng, với nhiều cải tiến tập trung vào tích hợp đám mây, hiệu năng thông minh và bảo mật. Việc lựa chọn nó cho dự án cần dựa trên sự phân tích kỹ lưỡng các yêu cầu cụ thể và nguồn lực sẵn có.

**1.4. Các nền tảng và công cụ hỗ trợ khác**

Để đảm bảo quá trình phát triển "Ứng dụng Cờ Đỏ" diễn ra một cách hiệu quả, chuyên nghiệp và sản phẩm cuối cùng đạt chất lượng cao, việc lựa chọn và sử dụng các nền tảng, thư viện và công cụ hỗ trợ phù hợp là vô cùng quan trọng. Bên cạnh môi trường phát triển tích hợp Visual Studio 2022, ngôn ngữ lập trình C# và hệ quản trị cơ sở dữ liệu SQL Server, dự án này đặc biệt tận dụng sức mạnh của Hệ thống Quản lý Phiên bản Git kết hợp với nền tảng lưu trữ GitHub, cùng với bộ công cụ giao diện người dùng nâng cao từ DevExpress. Những công cụ này không chỉ giúp tối ưu hóa quy trình làm việc cá nhân mà còn đặt nền móng cho các thực hành phát triển phần mềm tiên tiến.

**1.4.1. Hệ thống Quản lý Phiên bản – Git và Nền tảng Lưu trữ GitHub**

Trong phát triển phần mềm hiện đại, việc quản lý mã nguồn một cách hiệu quả là yếu tố then chốt. Git và GitHub đã trở thành bộ đôi công cụ tiêu chuẩn, được sử dụng rộng rãi từ các dự án cá nhân nhỏ lẻ đến các dự án doanh nghiệp quy mô lớn.

**1.4.1.1. Giới thiệu về Git**

Git là một hệ thống quản lý phiên bản phân tán (Distributed Version Control System - DVCS) mã nguồn mở, được tạo ra bởi Linus Torvalds vào năm 2005. Mục tiêu chính của Git là cung cấp một công cụ mạnh mẽ, linh hoạt và hiệu quả để theo dõi mọi thay đổi trong mã nguồn của dự án theo thời gian. Khác với các hệ thống quản lý phiên bản tập trung truyền thống (như SVN), mô hình phân tán của Git cho phép mỗi nhà phát triển có một bản sao đầy đủ của toàn bộ lịch sử dự án trên máy tính cá nhân của mình. Điều này mang lại nhiều lợi ích về tốc độ, khả năng làm việc ngoại tuyến và tính linh hoạt trong quy trình làm việc.

Các khái niệm cốt lõi của Git bao gồm:

* **Repository (Kho chứa):** Là nơi lưu trữ toàn bộ mã nguồn, lịch sử thay đổi và các thông tin liên quan đến dự án. Mỗi dự án sẽ có một kho chứa Git riêng.
* **Commit (Lưu thay đổi):** Là hành động ghi lại một tập hợp các thay đổi vào kho chứa. Mỗi commit đại diện cho một "phiên bản" của dự án tại một thời điểm nhất định và thường đi kèm với một thông điệp mô tả những thay đổi đó.
* **Branch (Nhánh):** Cho phép tạo ra các dòng phát triển độc lập trong cùng một kho chứa. Nhánh chính thường được gọi là main hoặc master. Các nhà phát triển có thể tạo các nhánh mới để phát triển tính năng, sửa lỗi mà không ảnh hưởng đến nhánh chính.
* **Merge (Gộp nhánh):** Là quá trình kết hợp các thay đổi từ một nhánh này sang một nhánh khác (ví dụ, gộp nhánh phát triển tính năng vào nhánh chính sau khi hoàn thành).
* **Clone (Sao chép kho chứa):** Tạo một bản sao cục bộ của một kho chứa từ xa (ví dụ, từ GitHub).
* **Push (Đẩy thay đổi):** Gửi các commit từ kho chứa cục bộ lên kho chứa từ xa.
* **Pull (Kéo thay đổi):** Lấy các thay đổi mới nhất từ kho chứa từ xa về kho chứa cục bộ.

**1.4.1.2. Giới thiệu về GitHub**

GitHub là một nền tảng dịch vụ lưu trữ dựa trên web dành cho các kho chứa Git. Nó không chỉ đơn thuần là nơi lưu trữ mã nguồn từ xa mà còn cung cấp một hệ sinh thái các công cụ và tính năng hỗ trợ mạnh mẽ cho việc cộng tác, quản lý dự án và phát triển phần mềm.

Các tính năng nổi bật của GitHub bao gồm:

* **Lưu trữ kho chứa Git từ xa:** Cung cấp một nơi an toàn và tin cậy để lưu trữ mã nguồn dự án trực tuyến.
* **Quản lý mã nguồn cộng tác:** Hỗ trợ các quy trình làm việc nhóm thông qua Pull Requests (yêu cầu gộp mã), Code Reviews (xem xét mã nguồn), và Issue Tracking (theo dõi lỗi và yêu cầu tính năng).
* **Quản lý dự án:** Cung cấp các công cụ như Projects (bảng Kanban), Milestones (cột mốc) để tổ chức và theo dõi tiến độ công việc.
* **GitHub Actions:** Cho phép tự động hóa các quy trình phát triển như xây dựng (build), kiểm thử (test), và triển khai (deploy) ứng dụng (CI/CD).
* **Tài liệu và Wiki:** Mỗi kho chứa có thể có một Wiki riêng để viết tài liệu dự án.
* **Cộng đồng lớn:** GitHub là nơi quy tụ hàng triệu nhà phát triển và dự án mã nguồn mở, tạo ra một môi trường học hỏi và chia sẻ kiến thức phong phú.

**1.4.1.3. Lợi ích và ứng dụng của Git và GitHub cho dự án "Ứng dụng Cờ Đỏ"**

Mặc dù "Ứng dụng Cờ Đỏ" có thể là một dự án được thực hiện bởi một sinh viên, việc áp dụng Git và GitHub mang lại nhiều lợi ích thiết thực:

1. **Quản lý phiên bản mã nguồn hiệu quả:**
   * **Theo dõi lịch sử chi tiết:** Mọi thay đổi trong mã nguồn C#, các tệp thiết kế giao diện, hay các script SQL của "Ứng dụng Cờ Đỏ" đều được Git ghi lại. Sinh viên có thể dễ dàng xem lại các phiên bản trước đó, so sánh sự khác biệt giữa các phiên bản, và hiểu rõ quá trình phát triển của từng tính năng.
   * **Khả năng phục hồi:** Nếu có lỗi nghiêm trọng xảy ra hoặc một tính năng mới gây ra vấn đề, Git cho phép dễ dàng quay trở lại một phiên bản ổn định trước đó, giảm thiểu rủi ro và thời gian khắc phục sự cố.
2. **Tổ chức phát triển khoa học:**
   * **Phân nhánh cho từng tính năng:** Sinh viên có thể tạo các nhánh riêng biệt để phát triển từng module hoặc tính năng của "Ứng dụng Cờ Đỏ" (ví dụ: nhánh feature/quan-ly-hoc-sinh, feature/cham-diem-tuan, feature/xuat-bao-cao). Điều này giúp cô lập các thay đổi, tránh ảnh hưởng đến phần mã nguồn chính đang hoạt động tốt và cho phép tập trung vào từng phần việc cụ thể. Sau khi hoàn thành và kiểm thử, các nhánh này sẽ được gộp vào nhánh chính.
3. **Sao lưu mã nguồn an toàn và truy cập linh hoạt:**
   * Lưu trữ mã nguồn trên GitHub đóng vai trò như một bản sao lưu từ xa quan trọng. Nếu máy tính cá nhân gặp sự cố (hỏng hóc phần cứng, mất dữ liệu), toàn bộ mã nguồn và lịch sử dự án vẫn được bảo vệ an toàn trên GitHub.
   * Sinh viên có thể truy cập và làm việc với mã nguồn từ bất kỳ máy tính nào có cài đặt Git và kết nối internet.
4. **Thực hành quy trình làm việc chuyên nghiệp:**
   * Việc sử dụng Git và GitHub giúp sinh viên làm quen với các công cụ và quy trình làm việc tiêu chuẩn trong ngành phát triển phần mềm. Đây là một kỹ năng quan trọng và được đánh giá cao bởi các nhà tuyển dụng.
5. **Hỗ trợ quá trình hướng dẫn và đánh giá:**
   * Giáo viên hướng dẫn (ThS. Phan Thị Gấm) có thể dễ dàng truy cập kho chứa GitHub của dự án để theo dõi tiến độ, xem xét mã nguồn, và đưa ra những nhận xét, góp ý kịp thời. Điều này tạo điều kiện cho sự tương tác hiệu quả hơn giữa sinh viên và người hướng dẫn.
6. **Xây dựng "Portfolio" cá nhân:**
   * Một kho chứa GitHub công khai với các commit đều đặn và mã nguồn được tổ chức tốt có thể trở thành một phần quan trọng trong hồ sơ năng lực (portfolio) của sinh viên khi tìm kiếm cơ hội thực tập hoặc việc làm sau này.

**1.4.1.4. Tích hợp Git và GitHub với Visual Studio 2022**

Visual Studio 2022 cung cấp khả năng tích hợp sâu rộng và mạnh mẽ với Git và GitHub. Sinh viên có thể thực hiện hầu hết các thao tác Git cơ bản và nâng cao trực tiếp từ giao diện của Visual Studio mà không cần sử dụng dòng lệnh (mặc dù việc hiểu dòng lệnh Git vẫn rất hữu ích). Các tính năng tích hợp bao gồm:

* Khởi tạo kho chứa Git mới cho dự án hoặc clone một kho chứa từ GitHub.
* Xem các thay đổi, tạo commit, push và pull thay đổi.
* Quản lý nhánh: tạo, chuyển đổi, gộp và xóa nhánh.
* Xem lịch sử commit.
* Giải quyết xung đột khi gộp nhánh (merge conflicts).

Sự tích hợp này giúp quy trình làm việc với Git trở nên liền mạch và thuận tiện hơn trong môi trường phát triển quen thuộc của Visual Studio.

**1.4.2. Bộ công cụ Giao diện Người dùng Nâng cao – DevExpress UI Controls**

Việc xây dựng một giao diện người dùng (UI) trực quan, chuyên nghiệp và giàu tính năng là một yếu tố quan trọng quyết định sự thành công và khả năng sử dụng của bất kỳ ứng dụng desktop nào, bao gồm cả "Ứng dụng Cờ Đỏ". DevExpress là một trong những nhà cung cấp hàng đầu thế giới về các bộ công cụ (suite) giao diện người dùng và thư viện phát triển cho nhiều nền tảng, trong đó có .NET (WinForms, WPF, ASP.NET).

**1.4.2.1. Giới thiệu về DevExpress và các bộ control cho .NET**

DevExpress cung cấp một loạt các control (điều khiển) giao diện người dùng được thiết kế tỉ mỉ, có hiệu suất cao và tích hợp nhiều tính năng mạnh mẽ. Đối với dự án "Ứng dụng Cờ Đỏ" phát triển trên nền tảng .NET (giả định là Windows Forms dựa trên tính phổ biến và sự phù hợp với loại ứng dụng quản lý), bộ DevExpress WinForms UI Controls sẽ là một lựa chọn mạnh mẽ. Các control này không chỉ thay thế các control chuẩn của Windows Forms mà còn mở rộng và cung cấp nhiều khả năng tùy biến vượt trội.

**1.4.2.2. Các loại control chính của DevExpress và ứng dụng tiềm năng cho "Ứng dụng Cờ Đỏ"**

1. **Data Grid (Lưới dữ liệu) – DevExpress.XtraGrid.GridControl:**
   * **Mô tả:** Đây là một trong những control mạnh mẽ và linh hoạt nhất của DevExpress. Nó cho phép hiển thị và quản lý dữ liệu dưới dạng bảng (lưới) với vô số tính năng như sắp xếp, lọc, nhóm dữ liệu theo nhiều cấp, tìm kiếm, tính toán tổng hợp (summaries), chỉnh sửa dữ liệu tại chỗ, định dạng có điều kiện, xuất dữ liệu ra Excel, PDF, Word, v.v.
   * **Ứng dụng cho "Ứng dụng Cờ Đỏ":**
     + Hiển thị danh sách học sinh của các lớp, với các thông tin chi tiết.
     + Hiển thị danh sách các vi phạm hoặc thành tích được ghi nhận, kèm theo thông tin ngày tháng, người ghi nhận, mô tả.
     + Trình bày bảng điểm thi đua hàng tuần, hàng tháng của các lớp một cách rõ ràng, dễ theo dõi.
     + Cho phép người dùng dễ dàng tìm kiếm, lọc và sắp xếp dữ liệu để tra cứu thông tin nhanh chóng.
2. **Charting (Biểu đồ) – DevExpress.XtraCharts.ChartControl:**
   * **Mô tả:** Cung cấp khả năng vẽ nhiều loại biểu đồ 2D và 3D khác nhau (cột, đường, tròn, vùng, thanh, điểm, v.v.) với khả năng tùy biến cao về giao diện và tương tác.
   * **Ứng dụng cho "Ứng dụng Cờ Đỏ":**
     + Trực quan hóa tình hình nề nếp của các lớp thông qua biểu đồ so sánh điểm số.
     + Hiển thị xu hướng thay đổi điểm số của một lớp hoặc một khối theo thời gian.
     + Thống kê các loại vi phạm phổ biến bằng biểu đồ tròn hoặc cột.
3. **Ribbon, Bars & Menus (Thanh Ribbon, Thanh công cụ & Menu) – DevExpress.XtraBars suite:**
   * **Mô tả:** Cho phép tạo giao diện người dùng hiện đại theo phong cách Microsoft Office với thanh Ribbon, các thanh công cụ (toolbars), menu ngữ cảnh (context menus), và thanh trạng thái (status bars). Thanh Ribbon giúp tổ chức các lệnh và chức năng của ứng dụng một cách logic và dễ tiếp cận.
   * **Ứng dụng cho "Ứng dụng Cờ Đỏ":**
     + Tạo một giao diện chính chuyên nghiệp, nơi các chức năng như "Quản lý Học sinh", "Ghi nhận Sự vụ", "Chấm điểm Thi đua", "Xem Báo cáo", "Cài đặt Hệ thống" được nhóm lại một cách khoa học trên thanh Ribbon.
     + Cung cấp các nút lệnh nhanh trên toolbar cho các thao tác thường dùng.
4. **Layout Control (Kiểm soát Bố cục) – DevExpress.XtraLayout.LayoutControl:**
   * **Mô tả:** Giúp đơn giản hóa việc thiết kế các form nhập liệu và hiển thị phức tạp. LayoutControl tự động quản lý vị trí và kích thước của các control con, đảm bảo sự căn chỉnh đồng đều, và cho phép giao diện tự động thích ứng khi kích thước cửa sổ thay đổi.
   * **Ứng dụng cho "Ứng dụng Cờ Đỏ":**
     + Thiết kế các form nhập thông tin vi phạm, form cập nhật thông tin học sinh, form cấu hình tiêu chí chấm điểm một cách nhanh chóng và nhất quán.
     + Đảm bảo giao diện người dùng trông gọn gàng và chuyên nghiệp trên các độ phân giải màn hình khác nhau.
5. **Editors (Các Control Nhập liệu) – DevExpress.XtraEditors suite:**
   * **Mô tả:** Cung cấp một bộ sưu tập phong phú các control nhập liệu nâng cao, thay thế và mở rộng các control chuẩn. Ví dụ: DateEdit (chọn ngày tháng với lịch), LookUpEdit (chọn giá trị từ danh sách thả xuống, có thể tìm kiếm), ComboBoxEdit, SpinEdit (nhập số), TextEdit với khả năng định dạng và xác thực đầu vào.
   * **Ứng dụng cho "Ứng dụng Cờ Đỏ":**
     + Sử dụng DateEdit để chọn ngày ghi nhận vi phạm.
     + Sử dụng LookUpEdit hoặc ComboBoxEdit để chọn lớp, chọn học sinh, chọn loại vi phạm từ danh sách đã được định nghĩa sẵn.
     + Áp dụng các quy tắc xác thực (validation) trực tiếp trên các editor để đảm bảo dữ liệu nhập vào là hợp lệ.
6. **Reporting (Thiết kế và Xuất Báo cáo) – DevExpress.XtraReports.UI.XtraReport:**
   * **Mô tả:** Một công cụ mạnh mẽ để thiết kế, xem trước và xuất các báo cáo động hoặc tĩnh ra nhiều định dạng phổ biến như PDF, Excel, Word, HTML. Cung cấp một trình thiết kế báo cáo trực quan (Report Designer).
   * **Ứng dụng cho "Ứng dụng Cờ Đỏ":**
     + Tạo các báo cáo tổng kết điểm thi đua tuần, tháng, học kỳ của các lớp.
     + In danh sách chi tiết các vi phạm của một lớp hoặc một học sinh.
     + Xuất các biểu đồ thống kê ra file để lưu trữ hoặc chia sẻ.

**1.4.2.3. Lợi ích của việc sử dụng DevExpress cho dự án "Ứng dụng Cờ Đỏ"**

1. **Nâng cao chất lượng giao diện người dùng:** Giúp "Ứng dụng Cờ Đỏ" có một giao diện chuyên nghiệp, hiện đại, trực quan và dễ sử dụng, tạo ấn tượng tốt cho người dùng cuối (giáo viên, thành viên đội Cờ Đỏ).
2. **Tiết kiệm đáng kể thời gian và công sức phát triển UI:** Thay vì phải tự lập trình các thành phần giao diện phức tạp từ đầu, sinh viên có thể tận dụng các control sẵn có, giàu tính năng của DevExpress. Điều này cho phép tập trung nhiều hơn vào việc xây dựng logic nghiệp vụ cốt lõi của ứng dụng.
3. **Tăng năng suất phát triển:** Các control của DevExpress thường đi kèm với nhiều API và sự kiện mạnh mẽ, cho phép tùy biến và xử lý các tương tác người dùng một cách linh hoạt.
4. **Cung cấp nhiều tính năng tích hợp sẵn:** Nhiều control như Data Grid đã tích hợp sẵn các chức năng phức tạp (sắp xếp, lọc, nhóm, tìm kiếm, xuất dữ liệu), giúp giảm thiểu lượng mã cần viết.
5. **Đảm bảo tính nhất quán trong thiết kế:** Sử dụng một bộ control thống nhất từ DevExpress giúp giao diện của các màn hình khác nhau trong ứng dụng có sự đồng bộ về mặt thẩm mỹ và trải nghiệm người dùng.

**1.4.2.4. Tích hợp DevExpress với Visual Studio 2022 và C#**

DevExpress được thiết kế để tích hợp một cách liền mạch với môi trường phát triển Visual Studio. Sau khi cài đặt, các control của DevExpress sẽ xuất hiện trong Toolbox của Visual Studio, cho phép kéo thả vào các form Windows Forms một cách dễ dàng. DevExpress cũng cung cấp các trình hướng dẫn (wizards) và trình thiết kế (designers) trực quan ngay trong Visual Studio để giúp cấu hình và tùy chỉnh các control một cách nhanh chóng. Việc lập trình sự kiện và tương tác với các control DevExpress bằng ngôn ngữ C# cũng tương tự như với các control chuẩn, nhưng với nhiều thuộc tính và phương thức mở rộng hơn.

Tóm lại, việc kết hợp sức mạnh của Git và GitHub để quản lý mã nguồn một cách chuyên nghiệp, cùng với việc sử dụng bộ công cụ giao diện người dùng DevExpress để xây dựng một ứng dụng desktop "Ứng dụng Cờ Đỏ" giàu tính năng và có giao diện hấp dẫn, sẽ đóng góp quan trọng vào sự thành công của dự án. Những công cụ này không chỉ hỗ trợ hoàn thành đồ án hiện tại mà còn trang bị cho sinh viên những kỹ năng và kinh nghiệm quý báu cho sự nghiệp phát triển phần mềm trong tương lai.

# Chương 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG CỜ ĐỎ

## 2.1. Giới thiệu về hệ thống

### 2.1.1. Thực trạng của hệ thống quản lý nề nếp

Tại Trường Phổ thông A, công tác quản lý nề nếp học sinh hiện đang được triển khai theo phương pháp thủ công, chủ yếu dựa vào sự phối hợp của đội cờ đỏ, ban nề nếp, giáo viên chủ nhiệm và giáo viên bộ môn.

Cụ thể, vào đầu mỗi buổi học nhà trường phân công đội cờ đỏ và ban nề nếp trực tại cổng trường để ghi nhận các lỗi vi phạm nề nếp cá nhân của học sinh, bao gồm: đi học muộn, không đeo phù hiệu, không mặc đúng đồng phục, vi phạm về tóc hoặc trang phục, vi phạm trật tự nơi cổng trường,… Cùng thời điểm đó, ban nề nếp cũng tiến hành kiểm tra tại các lớp học để ghi nhận các lỗi tập thể như: lớp không vệ sinh sạch sẽ, không trực nhật đúng quy định, không tổ chức sinh hoạt lớp theo chủ đề, vi phạm trật tự trong lớp,...

Song song với đó, trong từng tiết học, giáo viên bộ môn cũng thực hiện việc nhận xét ý thức học tập, thái độ và hành vi của học sinh thông qua phần ghi chú trong sổ đầu bài. Những nhận xét này phản ánh trực tiếp mức độ tập trung, trật tự, chuẩn bị bài vở và chấp hành nội quy lớp học của học sinh trong mỗi tiết học. Đây là một nguồn dữ liệu quan trọng làm căn cứ tổng hợp điểm rèn luyện, đánh giá thi đua của từng lớp và từng cá nhân học sinh theo tuần, tháng và học kỳ.

Tuy nhiên, toàn bộ quá trình ghi nhận và tổng hợp hiện vẫn được thực hiện thủ công, chủ yếu bằng cách ghi tay hoặc điền vào biểu mẫu giấy. Việc thống kê vi phạm, đối chiếu thông tin từ các nguồn khác nhau như sổ ghi của cờ đỏ, ban nề nếp và sổ đầu bài thường mất nhiều thời gian, dễ sai sót, thiếu chính xác và không đảm bảo tính đồng bộ. Đặc biệt, khi số lượng lớp học và học sinh đông, việc tổng hợp dữ liệu để kịp thời phát hiện các học sinh tái phạm hoặc xếp loại thi đua theo thời gian thực là rất khó khăn.

Thực trạng này không chỉ gây áp lực cho cán bộ giám thị, giáo viên chủ nhiệm và ban thi đua, mà còn làm giảm hiệu quả công tác giáo dục nề nếp và kỷ luật học đường. Do đó, việc xây dựng một hệ thống ứng dụng công nghệ thông tin nhằm số hóa quy trình ghi nhận, tổng hợp và phân tích vi phạm nề nếp là hết sức cần thiết. Hệ thống không chỉ giúp giảm tải khối lượng công việc, mà còn góp phần nâng cao tính chính xác, minh bạch và hiệu quả trong công tác quản lý, từ đó thúc đẩy xây dựng môi trường học tập kỷ luật, tích cực và hiện đại.

### 2.1.2. Mô tả quy trình hoạt động của Đội Cờ Đỏ:

Quy trình hoạt động điển hình của Đội Cờ Đỏ theo phương pháp thủ công, có tích hợp thông tin từ sổ đầu bài, thường bao gồm các bước chính sau:

* Phân công nhiệm vụ (Đầu tuần)
* Trưởng ban nề nếp hoặc Trưởng Đội Cờ Đỏ (là học sinh) tiến hành phân công khu vực, thời gian và nội dung kiểm tra (ví dụ: kiểm tra đồng phục, giờ giấc, vệ sinh, kỷ luật trong giờ học, xếp hàng ra vào lớp,...) cho từng thành viên hoặc nhóm thành viên Đội Cờ Đỏ.
* Việc phân công này thường được ghi vào sổ trực hoặc thông báo miệng.
* Thực hiện kiểm tra và Thu thập thông tin (Hàng tuần):
* **Kiểm tra trực tiếp bởi Đội Cờ Đỏ:** Các thành viên Đội Cờ Đỏ đến các khu vực được phân công (lớp học, hành lang, sân trường, cổng trường) vào các thời điểm quy định (đầu giờ, giờ ra chơi, cuối buổi học). Khi phát hiện các trường hợp vi phạm nội quy hoặc những biểu hiện tốt, gương tốt, thành viên Đội Cờ Đỏ sẽ ghi nhận thông tin vào sổ trực cá nhân hoặc các biểu mẫu giấy được phát sẵn. Thông tin ghi nhận thường bao gồm: Lớp vi phạm/có biểu hiện tốt, thời gian, địa điểm, nội dung sự việc, mức độ (nếu có), và tên người ghi nhận.
* **Thu thập thông tin từ Sổ đầu bài:** Cuối mỗi buổi học hoặc cuối ngày, thành viên Đội Cờ Đỏ được phân công (hoặc Trưởng Đội) sẽ thu thập hoặc tham khảo các ghi nhận trong Sổ đầu bài của từng lớp. Sổ đầu bài do giáo viên bộ môn và cán bộ lớp (lớp trưởng, lớp phó học tập) ghi lại các nhận xét về tình hình học tập, kỷ luật, các trường hợp học sinh vi phạm hoặc có thành tích trong từng tiết học.
* Tổng hợp sự vụ (Hàng ngày, hàng tuần):
* Trưởng Đội Cờ Đỏ hoặc nhóm được phân công tổng hợp sẽ tập hợp các ghi nhận từ sổ trực của các thành viên Đội Cờ Đỏ và các thông tin liên quan từ Sổ đầu bài của các lớp.
* Quá trình này bao gồm việc đối chiếu, xác minh thông tin, loại bỏ các trường hợp trùng lặp hoặc chưa rõ ràng. Kết quả là một biên bản tổng hợp các sự vụ (vi phạm, thành tích) của cả tập thể lớp và của từng cá nhân học sinh trong ngày.
* Xử lý điểm cá nhân và quy đổi ra điểm trừ/cộng cho lớp (Thường vào cuối ngày Thứ 6):
* Dựa trên biên bản tổng hợp sự vụ hàng ngày và các quy định, tiêu chí của nhà trường, lớp trực tuần sẽ thực hiện nhiệm vụ tổng hợp và xếp loại thi đua của tuần đó. Bảng xếp loại sẽ được gửi cho trưởng ban nề nếp xét duyệt và công bố vào cuối tuần. Nhiệm vụ tổng hợp gồm có:
  + Xác định các lỗi hoặc thành tích của cá nhân học sinh.
  + Quy đổi các lỗi này thành điểm trừ cá nhân và các thành tích thành điểm cộng cá nhân (nếu có quy định).
  + Tổng hợp các điểm trừ/cộng cá nhân này và các lỗi/thành tích của tập thể lớp để tính toán ra điểm trừ/cộng chung cho từng lớp trong tuần. Ví dụ, tổng số điểm trừ của các cá nhân trong lớp sẽ đóng góp vào tổng điểm trừ của lớp đó.
* Chấm điểm thi đua tuần (Thường vào ngày cuối tuần hoặc đầu tuần kế tiếp):
* Biên bản tổng hợp sự vụ và kết quả điểm trừ/cộng chung của các lớp được trình lên Giáo viên Tổng phụ trách Đội hoặc Ban thi đua của nhà trường.
* Dựa trên các tiêu chí thi đua chung của nhà trường (thường được in sẵn hoặc ghi trong sổ theo dõi) và kết quả điểm trừ/cộng của lớp đã được xử lý, giáo viên Tổng phụ trách Đội hoặc Ban thi đua tiến hành chấm điểm thi đua chính thức cho từng lớp trong tuần.
* Quá trình này bao gồm việc áp dụng thang điểm, có thể có sự điều chỉnh dựa trên các yếu tố khác.
* Tổng kết, xếp hạng và công bố kết quả (Đầu tuần kế tiếp):
* Điểm thi đua tuần của các lớp được tổng hợp lại để xếp hạng.
* Kết quả xếp hạng thường được công bố trên bảng tin của trường, trong các buổi sinh hoạt dưới cờ hoặc thông báo đến giáo viên chủ nhiệm các lớp.
* Lưu trữ hồ sơ:

Các sổ sách ghi chép (sổ trực Cờ Đỏ, Sổ đầu bài), biên bản tổng hợp sự vụ, phiếu chấm điểm, báo cáo tổng kết được lưu trữ thủ công trong các tủ hồ sơ của văn phòng Đoàn - Đội hoặc phòng giáo viên Tổng phụ trách.

Sơ đồ hoạt động của hệ thống quản lý ban nề nếp

A diagram of a flowchart

AI-generated content may be incorrect.

Hình 2.1. Sơ đồ hoạt động của hệ thống quản lý ban nề nếp

Quy trình hoạt động

* **Bắt đầu (Đầu tuần):** Quy trình khởi đầu từ việc Giáo viên Tổng phụ trách Đội hoặc Trưởng Đội Cờ Đỏ thực hiện phân công nhiệm vụ.
* **Thu thập và Ghi nhận (Hàng ngày):** Các thành viên Đội Cờ Đỏ tiến hành kiểm tra trực tiếp và ghi nhận vào sổ/phiếu. Song song đó, thông tin từ Sổ đầu bài của các lớp cũng được thu thập hoặc tham khảo.
* **Tổng hợp sự vụ (Hàng ngày, hàng tuần):** Người được phân công (thường là Trưởng Đội) tổng hợp tất cả các ghi nhận từ Đội Cờ Đỏ và Sổ đầu bài để tạo ra Biên bản tổng hợp sự vụ, bao gồm cả các vấn đề của tập thể lớp và các ghi nhận liên quan đến cá nhân học sinh.
* **Xử lý điểm cá nhân và lớp (Ngày cuối tuần):** Dựa trên biên bản tổng hợp, Trưởng Đội hoặc Giáo viên TPT Đội sẽ xử lý các vi phạm/thành tích cá nhân để quy ra điểm trừ/cộng cá nhân. Từ đó, kết hợp với các vấn đề của tập thể, tính toán ra điểm trừ/cộng chung cho mỗi lớp trong tuần.
* **Chấm điểm Thi đua (Ngày cuối tuần hoặc đầu tuần sau):** Giáo viên TPT Đội hoặc Ban Thi đua sử dụng điểm trừ/cộng chung của lớp và các tiêu chí thi đua của nhà trường để chấm điểm và xếp hạng thi đua chính thức cho các lớp.
* **Công bố và Lưu trữ:** Kết quả xếp hạng được công bố rộng rãi. Toàn bộ hồ sơ, sổ sách liên quan được lưu trữ thủ công.

## 2.2. Yêu cầu của hệ thống cần xây dựng

### 2.2.1. Yêu cầu chức năng

Dựa trên phân tích thực trạng hoạt động thủ công của Đội Cờ Đỏ với những khó khăn đã được xác định (tốn thời gian, dễ sai sót, thiếu khách quan, khó khăn trong tổng hợp và tra cứu, thiếu minh bạch), hệ thống "Ứng dụng quản lý nề nếp" cần được xây dựng với các nhóm chức năng chính sau để số hóa và tối ưu hóa toàn bộ quy trình:

#### 1. Quản lý Hệ thống và Người dùng:

• **Đăng nhập và Phân quyền:**

* Hệ thống cho phép người dùng đăng nhập bằng tài khoản và mật khẩu được cấp
* Phân quyền truy cập chức năng dựa trên vai trò người dùng với cấu trúc phân cấp:
* **Quản trị viên hệ thống**: Có toàn quyền quản lý hệ thống
* **Ban Nề nếp** (Trưởng ban thường là Hiệu phó hoặc Phó hiệu trưởng phụ trách): Có quyền quản lý toàn bộ hoạt động, cấu hình hệ thống, xem báo cáo tổng hợp
* **Giáo viên chủ nhiệm (GVCN)**: Có quyền xem và quản lý thông tin liên quan đến lớp mình phụ trách
* **Giáo viên được phân công**: Có quyền ghi nhận sự vụ trong phạm vi được giao
* **Bảo vệ**: Có quyền ghi nhận các vi phạm về giờ giấc, an ninh trật tự
* **Thành viên Đội Cờ Đỏ**: Có quyền ghi nhận sự vụ theo khu vực được phân công
* **Lớp trưởng/Tổ trưởng**: Có quyền xem thông tin và báo cáo của lớp/tổ mình

• **Quản lý Tài khoản Người dùng:**

* Quản trị viên và Ban Nề nếp có thể tạo mới, chỉnh sửa, xóa, khóa/mở khóa tài khoản người dùng
* Cho phép người dùng tự đổi mật khẩu và cập nhật thông tin cá nhân
* Quản lý lịch sử đăng nhập và hoạt động của người dùng

• **Sao lưu và Phục hồi Dữ liệu:**

* Cung cấp chức năng sao lưu dữ liệu định kỳ (tự động) hoặc thủ công
* Cho phép khôi phục dữ liệu từ các bản sao lưu khi cần thiết
* Lưu trữ nhiều phiên bản sao lưu với nhãn thời gian

• **Cấu hình Tham số Hệ thống:**

* Cho phép Ban Nề nếp cấu hình các tham số chung: năm học, học kỳ, tuần học hiện tại
* Thiết lập các quy định tính điểm cơ bản của nhà trường

**2. Quản lý Danh mục Cơ bản:**

• **Quản lý Thông tin Khối/Lớp học:**

* Thêm, sửa, xóa thông tin các khối học và lớp học (tên lớp, khối, giáo viên chủ nhiệm, sĩ số)
* Quản lý danh sách lớp theo từng năm học
* Ví dụ: Thêm lớp 10A1 thuộc khối 10, GVCN: Nguyễn Thị A, sĩ số: 42 học sinh

• **Quản lý Thông tin Học sinh:**

* Thêm, sửa, xóa thông tin chi tiết học sinh (mã học sinh, họ tên, ngày sinh, giới tính, lớp)
* Import danh sách học sinh từ file Excel để tiết kiệm thời gian nhập liệu
* Tìm kiếm học sinh theo nhiều tiêu chí (theo tên, lớp, mã học sinh)
* Chuyển lớp cho học sinh khi cần thiết

• **Quản lý Tiêu chí Thi đua và Thang điểm:**

* Định nghĩa các tiêu chí thi đua (đồng phục, đi học đúng giờ, vệ sinh, kỷ luật...)
* Thiết lập thang điểm cộng/trừ cho từng mức độ vi phạm hoặc thành tích
* Cho phép cập nhật, điều chỉnh khi có thay đổi từ nhà trường
* Ví dụ: Vi phạm "Đi học muộn" - Trừ 2 điểm/lần; "Quên phù hiệu" - Trừ 1 điểm/lần

• **Quản lý Danh mục Loại Vi phạm/Thành tích:**

* Tạo và quản lý danh mục các loại vi phạm thường gặp và thành tích cần biểu dương
* Liên kết mỗi loại với tiêu chí thi đua và điểm số tương ứng
* Phân loại theo mức độ nghiêm trọng (nhẹ, trung bình, nặng)

**3. Quản lý Hoạt động Ghi nhận và Chấm điểm:**

• **Ghi nhận Sự vụ Vi phạm/Thành tích:**

* Giao diện ghi nhận nhanh cho tất cả các thành phần có quyền (GVCN, GV phân công, Bảo vệ, Thành viên Đội Cờ Đỏ)
* Thông tin ghi nhận bao gồm: ngày giờ, địa điểm, lớp, học sinh (nếu là cá nhân), loại vi phạm/thành tích, mô tả chi tiết, người ghi nhận
* Hỗ trợ ghi nhận từ nhiều nguồn khác nhau với phân biệt rõ ràng nguồn ghi nhận
* Ví dụ: Ngày 15/03, lớp 10A1, học sinh Trần Văn B, vi phạm "Không đeo phù hiệu", người ghi: Bảo vệ Nguyễn Văn D

• **Tổng hợp và Xác minh Sự vụ:**

* Tự động tổng hợp các sự vụ được ghi nhận theo ngày, tuần từ tất cả các nguồn
* Cho phép Ban Nề nếp xem xét, xác minh, chỉnh sửa hoặc xóa các ghi nhận chưa chính xác
* Hiển thị danh sách chi tiết các sự vụ với đầy đủ thông tin và nguồn ghi nhận
* Đánh dấu trạng thái xác minh cho từng sự vụ

• **Xử lý và Quy đổi Điểm:**

* Tự động quy đổi vi phạm/thành tích thành điểm trừ/cộng cá nhân
* Tổng hợp điểm cá nhân và tập thể để tính điểm chung cho từng lớp
* Cho phép Ban Nề nếp điều chỉnh thủ công khi cần thiết với ghi chú lý do
* Ví dụ: Lớp 10A1 có 3 em vi phạm đi muộn (-6đ), 2 em quên phù hiệu (-2đ), tổng điểm trừ: -8đ

• **Chấm điểm Thi đua Tuần:**

* Hiển thị tổng hợp điểm trừ/cộng của các lớp trong tuần
* Cho phép Ban Nề nếp xem xét và xác nhận điểm cuối cùng
* Tự động tính điểm thi đua dựa trên điểm chuẩn và điểm trừ/cộng
* Lưu lịch sử chấm điểm với thông tin người chấm và thời gian

**4. Quản lý Thi đua và Xếp hạng:**

• **Tổng hợp Điểm Thi đua:**

* Tự động tổng hợp điểm thi đua theo tuần, tháng, học kỳ, năm học
* Hiển thị chi tiết điểm của từng lớp với breakdown theo các tiêu chí
* So sánh điểm giữa các kỳ để thấy xu hướng tiến bộ/tụt lùi

• **Xếp hạng Thi đua:**

* Tự động xếp hạng các lớp dựa trên tổng điểm thi đua
* Xếp hạng theo khối và toàn trường
* Hiển thị bảng xếp hạng với thông tin điểm số và thứ hạng
* Đánh dấu các lớp đạt danh hiệu (Xuất sắc, Tốt, Khá, Trung bình)

• **Công bố Kết quả:**

* Xuất kết quả xếp hạng ra file PDF để in ấn, dán bảng tin
* Xuất file Excel để lưu trữ và chia sẻ
* Tạo bản công bố với format chuẩn của nhà trường

**5. Báo cáo và Thống kê:**

• **Báo cáo Tổng hợp Nề nếp:**

* Báo cáo chi tiết tình hình vi phạm, thành tích của từng lớp, từng học sinh
* Báo cáo tổng hợp điểm thi đua với nhiều góc nhìn khác nhau
* Báo cáo theo nguồn ghi nhận (GVCN, Bảo vệ, Đội Cờ Đỏ...)
* Ví dụ: Báo cáo tổng kết tháng 3 cho khối 10 với danh sách chi tiết vi phạm

• **Thống kê Trực quan:**

* Biểu đồ cột thể hiện điểm thi đua của các lớp
* Biểu đồ tròn phân tích tỷ lệ các loại vi phạm phổ biến
* Biểu đồ đường thể hiện xu hướng nề nếp theo thời gian
* Dashboard tổng quan cho Ban Giám hiệu và Ban Nề nếp

• **Tìm kiếm và Lọc Dữ liệu:**

* Tìm kiếm vi phạm theo nhiều tiêu chí kết hợp
* Lọc theo khoảng thời gian, lớp, loại vi phạm, mức độ, nguồn ghi nhận
* Xuất kết quả tìm kiếm ra file
* Ví dụ: Tìm tất cả vi phạm "Đi học muộn" được ghi nhận bởi Bảo vệ trong tháng 3

**6. Tích hợp Sổ đầu bài Điện tử:**

• **Ghi nhận từ Sổ đầu bài:**

* Cho phép giáo viên bộ môn ghi nhận nhanh vi phạm/biểu dương trong giờ học
* Đồng bộ thông tin với hệ thống chính của Ban Nề nếp
* Phân biệt rõ ràng nguồn ghi nhận để thống kê và báo cáo

Các yêu cầu chức năng này được thiết kế để giải quyết triệt để các vấn đề của quy trình thủ công, đảm bảo tính tự động hóa cao, giảm thiểu sai sót, tăng cường minh bạch và cung cấp đầy đủ thông tin kịp thời cho công tác quản lý nề nếp học sinh trong nhà trường theo cấu trúc phân cấp rõ ràng.

#### **b) Yêu cầu phi chức năng**

#### Ngoài các yêu cầu về chức năng, hệ thống "Ứng dụng Cờ Đỏ" cần đáp ứng các yêu cầu phi chức năng sau đây để đảm bảo chất lượng sử dụng, tính ổn định và khả năng mở rộng trong tương lai:

#### **1. Yêu cầu về Môi trường và Nền tảng:**

#### **Hệ điều hành:**

* Ứng dụng được phát triển dạng Desktop Application chạy trên hệ điều hành Windows
* Hỗ trợ các phiên bản: Windows 10, Windows 11 (32-bit và 64-bit)
* Tương thích ngược với Windows 8.1 để phù hợp với các máy tính cũ trong trường học

#### **Framework và Runtime:**

* Yêu cầu cài đặt .NET Framework 4.8 hoặc cao hơn
* SQL Server 2022 Express Edition trở lên (cho môi trường triển khai nhỏ) hoặc SQL Server Standard/Enterprise (cho triển khai quy mô lớn)

#### **2. Yêu cầu về Hiệu năng:**

#### **Tốc độ xử lý:**

#### **Tốc độ xử lí các tác vụ không được quá 1 phút**

#### **Khả năng xử lý đồng thời:**

* Xử lý được tối thiểu 100 giao dịch ghi nhận vi phạm/phút

#### **Dung lượng dữ liệu:**

* Lưu trữ dữ liệu tối thiểu 5 năm họcs
* Kích thước cơ sở dữ liệu dự kiến: 5-10GB cho 5 năm hoạt động

#### **3. Yêu cầu về Giao diện Người dùng:**

#### **Thiết kế giao diện:**

* Giao diện hiện đại, chuyên nghiệp sử dụng bộ control DevExpress
* Bố cục rõ ràng, nhất quán trên toàn bộ ứng dụng
* Sử dụng hệ thống icon và biểu tượng trực quan
* Font chữ: Tahoma hoặc Segoe UI, cỡ chữ 10-12pt cho nội dung chính

#### **Màu sắc:**

* Màu cảnh báo: Đỏ cho lỗi, Vàng cho cảnh báo
* Màu thành công: Xanh lá
* Đảm bảo độ tương phản cao để dễ đọc

#### **4. Yêu cầu về Cấu hình Phần cứng:**

#### **Cấu hình tối thiểu (Client):**

* CPU: Intel Core i3 thế hệ 3 hoặc hơn
* RAM: 4GB
* Ổ cứng: 2GB dung lượng trống cho ứng dụng
* Kết nối mạng

#### **Cấu hình khuyến nghị (Client):**

* CPU: Intel Core i3 thế hệ 6 trở lên
* RAM: 8GB
* Ổ cứng: SSD với 4GB dung lượng trống
* Kết nối mạng

#### **5. Yêu cầu về Bảo mật:**

#### **Xác thực và Phân quyền:**

* Mật khẩu tối thiểu 8 ký tự, bao gồm chữ hoa, chữ thường và số
* Khóa tài khoản sau 5 lần đăng nhập sai liên tiếp
* Lưu log tất cả hoạt động đăng nhập/đăng xuất

#### **Bảo mật Dữ liệu:**

* Mã hóa mật khẩu trong database bằng thuật toán bcrypt hoặc SHA-256
* Sao lưu dữ liệu tự động hàng ngày
* Phân quyền truy cập dữ liệu theo vai trò người dùng

#### **6. Yêu cầu về Độ tin cậy và Khả dụng:**

* Uptime: Hệ thống cần đạt độ khả dụng tối thiểu 95% trong giờ làm việc
* Xử lý lỗi: Có cơ chế bắt lỗi và thông báo lỗi rõ ràng cho người dùng
* Khôi phục: Khả năng khôi phục dữ liệu trong vòng 4 giờ khi có sự cố

#### **7. Yêu cầu về Khả năng Bảo trì và Mở rộng:**

* Kiến trúc: Thiết kế theo mô hình 3 lớp (Presentation - Business Logic - Data Access)
* Mã nguồn: Tuân thủ coding convention của C#, có comment đầy đủ
* Database: Thiết kế chuẩn hóa
* Tài liệu: Có đầy đủ tài liệu hướng dẫn sử dụng và tài liệu kỹ thuật

#### **8. Yêu cầu về Tương thích và Tích hợp:**

* Import/Export: Hỗ trợ import/export dữ liệu từ/sang Excel
* Báo cáo: Xuất báo cáo sang định dạng PDF, Excel, Word
* In ấn: Tương thích với các loại máy in phổ biến
* Backup: Hỗ trợ backup/restore của SQL Server

#### Các yêu cầu phi chức năng này đảm bảo hệ thống không chỉ đáp ứng được các chức năng nghiệp vụ mà còn mang lại trải nghiệm sử dụng tốt, phù hợp với môi trường và điều kiện thực tế của các trường học, đồng thời có khả năng mở rộng và bảo trì trong tương lai.

## 2.2. Phân tích hệ thống

### 2.2.1. Biểu đồ phân cấp chức năng

A computer screen shot of a diagram

AI-generated content may be incorrect.

Hình 2.1. Biểu đồ phân cấp chức năng của ứng dụng quản lý nề nếp

Biểu đồ phân cấp chức năng mô tả cấu trúc của hệ thống quản lý nề nếp, bắt đầu từ chức năng gốc là "HỆ THỐNG QUẢN LÝ NỀ NẾP" và được chia thành 6 nhóm chức năng chính. Mỗi nhóm chức năng chính lại bao gồm các chức năng con cụ thể để thực hiện các nghiệp vụ chi tiết.

1. **Quản lý Tài khoản:**
   * **Chức năng:** Đây là nhóm chức năng chịu trách nhiệm quản lý việc truy cập của người dùng vào hệ thống. Nó đảm bảo chỉ những người dùng có thẩm quyền mới có thể đăng nhập và thực hiện các thao tác theo vai trò đã được phân công.
   * **Các tính năng con:**
     + Tạo tài khoản: Cho phép quản trị viên cấp tài khoản mới cho người dùng.
     + Trao hoặc nhập tài khoản theo lô: Hỗ trợ việc tạo hàng loạt tài khoản để tiết kiệm thời gian.
     + Cập nhật thông tin: Cho phép người dùng hoặc quản trị viên chỉnh sửa thông tin cá nhân.
     + Xóa tài khoản: Xóa bỏ tài khoản người dùng khỏi hệ thống.
     + Tìm kiếm tài khoản: Tìm kiếm thông tin một tài khoản cụ thể.
     + In danh sách tài khoản: Xuất danh sách các tài khoản người dùng.
2. **Quản lý Danh mục:**
   * **Chức năng:** Nhóm chức năng này quản lý các dữ liệu nền tảng, là cơ sở để các chức năng khác hoạt động. Việc quản lý các danh mục này đảm bảo tính nhất quán và chính xác cho toàn bộ hệ thống.
   * **Các tính năng con:**
     + Quản lý Khối & Lớp: Thêm, sửa, xóa, tìm kiếm thông tin về các khối, lớp học trong trường.
     + Quản lý Học sinh: Quản lý danh sách học sinh của toàn trường, bao gồm các thao tác thêm, sửa, xóa, tìm kiếm và import danh sách từ file ngoài.
     + Quản lý Tiêu chí Thi đua: Thiết lập các quy định, tiêu chí về nề nếp và thang điểm trừ/cộng tương ứng. Trong đó bao gồm chức năng con Quản lý Loại Vi phạm & Thành tích để định nghĩa cụ thể từng hành vi cần ghi nhận (ví dụ: đi muộn, quên phù hiệu, phát biểu xây dựng bài,...).
3. **Quản lý Sự vụ & Chấm điểm:**
   * **Chức năng:** Đây là nhóm chức năng nghiệp vụ cốt lõi, thực hiện việc ghi nhận, tổng hợp và xử lý các sự vụ nề nếp hàng ngày, hàng tuần.
   * **Các tính năng con:**
     + Ghi nhận Sự vụ: Cho phép người dùng có thẩm quyền (Đội Cờ Đỏ, giám thị, giáo viên) ghi lại các trường hợp vi phạm hoặc biểu dương của học sinh, lớp học.
     + Tổng hợp & Xác minh sự vụ: Hệ thống tự động gom các ghi nhận từ nhiều nguồn và cho phép người có trách nhiệm kiểm tra, xác thực lại thông tin.
     + Xử lý & quy đổi điểm: Dựa trên các tiêu chí đã thiết lập, hệ thống tự động tính toán điểm trừ hoặc điểm cộng cho các sự vụ đã được xác minh.
     + Chấm điểm Thi đua Tuần: Tổng hợp điểm của các lớp và cho phép ban nề nếp xác nhận, chốt điểm thi đua cuối cùng của tuần.
4. **Thống kê & Xếp loại:**
   * **Chức năng:** Nhóm chức năng này xử lý dữ liệu đã chấm để đưa ra kết quả xếp hạng thi đua.
   * **Các tính năng con:**
     + Tổng hợp Điểm Thi đua: Tính toán tổng điểm của các lớp theo các chu kỳ dài hơn như tháng, học kỳ.
     + Xếp hạng Thi đua: Tự động sắp xếp thứ hạng các lớp dựa trên tổng điểm.
     + Công bố Kết quả: Hỗ trợ xuất kết quả xếp hạng để thông báo công khai.
5. **Báo cáo:**
   * **Chức năng:** Cung cấp các công cụ để khai thác, phân tích dữ liệu nề nếp dưới nhiều góc độ khác nhau, hỗ trợ công tác quản lý và ra quyết định.
   * **Các tính năng con:**
     + Xem Báo cáo Chi tiết: Tạo các báo cáo cụ thể về tình hình vi phạm của một lớp, một học sinh hoặc một loại vi phạm nào đó.
     + Xem Thông tin Trực quan (Biểu đồ): Trực quan hóa dữ liệu bằng các loại biểu đồ (cột, tròn, đường) để dễ dàng nắm bắt xu hướng và so sánh.
     + Tìm kiếm & Lọc Dữ liệu: Cung cấp bộ lọc nâng cao để tra cứu thông tin theo nhiều điều kiện kết hợp.
6. **Tích hợp Sổ đầu bài:**
   * **Chức năng:** Mở rộng nguồn thu thập thông tin bằng cách cho phép nhập các ghi nhận từ sổ đầu bài vào hệ thống.
   * **Các tính năng con:**
     + Ghi nhận Sự vụ từ Sổ đầu bài: Cung cấp giao diện để nhập các nhận xét, đánh giá của giáo viên bộ môn trong từng tiết học vào luồng xử lý chung của hệ thống.

### 2.2.2. Biểu đồ luồng dữ liệu 🡪 tương tự như phân cấp chức năng, từ biểu đồ phân cấp chức năng, xác định các tác nhân (nhóm người dùng) để vẽ các luồng dữ liệu cung cấp cho hệ thống. Mỗi biểu đồ đều có hình ảnh và mô tả hoạt động của biểu đồ đó bằng lời

#### a) Biểu đồ ngữ cảnh

#### A computer screen shot of a diagram AI-generated content may be incorrect.

#### b) Biểu đồ mức 1

#### c) Biểu đồ mức 2

#### lưu ý: mức 2 có thể chỉ chọn một số chức năng quan trọng đẻ vẽ

### 2.2.3. Mô hình quan hệ thực thể

Vẽ và mô tả

## 2.3. Thiết kế hệ thống

### 2.3.1. Thiết kế chức năng

Thiết kế hoạt động của một số chức năng chính, đánh mục a), b), c) 🡪 mỗi chức năng mô tả: đầu vào là gì? Đầu ra là gì? Quá trình xử lý như thế nào?

### 2.3.1. Thiết kế cơ sở dữ liệu

Vẽ lược đồ dữ liệu và mô tả các bảng dữ liệu

### 2.3.2. Thiết kế một số giao diện chức năng

Dùng các công cụ vẽ giao diện để vẽ bản mẫu của một số chức năng quan trọng (giao diện dự kiến sẽ cài đặt)

# Chương 3. CÀI ĐẶT VÀ KIỂM THỬ HỆ THỐNG

## 3.1. Cài đặt hệ thống

Mô tả rõ hệ thống của mình cài đặt trên môi trường nào? Yêu cầu cấu hình máy tính ra sao: Hệ điều hành, NetFramework? Hệ quản trị CSDL? Cấu hình CPU, cấu hình Ram, …

## 3.2. Một số giao diện hệ thống đã cài đặt

Màn hình demo một số giao diện chức năng của hệ thống 🡪 dưới hình ảnh là mô tả hoạt động

## 3.3. Kết quả kiểm thử hệ thống

Mô tả kết quả chạy một số chức năng của hệ thống

# KẾT LUẬN

Nêu lên đã làm được gì? Còn chưa được gì? Hướng phát triển trong tương lai là gì?

# TÀI LIỆU THAM KHẢO

[1]. Tên tác giả, Tên sách, Nhà xuất bản, năm xuất bản

[2]. Tên các tác giả, “Tiêu đề bài báo khoa học”, *Tên tạp chí,*  Tập – số, năm xuất bản.