KHOA KỸ THUẬT VÀ CÔNG NGHỆ

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**



**THỰC TẬP ĐỒ ÁN CƠ SỞ NGÀNH**

**HỌC KỲ I, NĂM HỌC 2024 - 2025**

**XÂY DỰNG PHẦN MỀM ỨNG DỤNG QUẢN LÝ**

**CỬA HÀNG BÁN VẬT TƯ NÔNG NGHIỆP**

|  |  |
| --- | --- |
| *Giảng viên hướng dẫn:*  TS. Nguyễn Trần Diễm Hạnh | *Sinh viên thực hiện:*  Họ tên: Phạm Thế Vinh  MSSV: 110122208  Lớp: DA22TTC |

***Trà Vinh, tháng 12 năm 2024***

KHOA KỸ THUẬT VÀ CÔNG NGHỆ

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**



**THỰC TẬP ĐỒ ÁN CƠ SỞ NGÀNH**

**HỌC KỲ I, NĂM HỌC 2024 - 2025**

**XÂY DỰNG PHẦN MỀM ỨNG DỤNG QUẢN LÝ**

**CỬA HÀNG BÁN VẬT TƯ NÔNG NGHIỆP**

|  |  |
| --- | --- |
| *Giảng viên hướng dẫn:*  TS. Nguyễn Trần Diễm Hạnh | *Sinh viên thực hiện:*  Họ tên: Phạm Thế Vinh  MSSV: 110122208  Lớp: DA22TTC |

***Trà Vinh, tháng 12 năm 2024***

**NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

*Trà Vinh, ngày ….. tháng …… năm ……*

**Giảng viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

*Trà Vinh, ngày ….. tháng …… năm ……*

**Giáo viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

**NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG**

*Trà Vinh, ngày ….. tháng …… năm ……*

**Thành viên hội đồng**

*(Ký tên và ghi rõ họ tên)*

**LỜI CẢM ƠN**

Tôi xin gửi lời cảm ơn đến giảng viên Nguyễn Trần Diễm Hạnh, người đã hướng dẫn tôi thực hiện đồ án này. Cô đã chia sẻ và truyền đạt nhiều kiến thức cùng với những kinh nghiệm bổ ích, đồng thời cô cũng giải đáp những thắc mắc giúp tôi trong quá trình thực hiện đồ án.

Tôi cũng xin gửi lời cảm ơn đến các thầy cô ở Bộ môn Công nghệ thông tin đã giảng dạy nhiệt tình, giúp tôi tích lũy được nhiều kiến thức để vận dụng vào việc thực hiện đồ án.

Tôi xin chân thành cảm ơn!

# MỤC LỤC

[MỤC LỤC 4](#_Toc183802929)

[DANH MỤC HÌNH ẢNH 6](#_Toc183802930)

[DANH MỤC BẢNG BIỂU 7](#_Toc183802931)

[TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH 8](#_Toc183802932)

[MỞ ĐẦU 10](#_Toc183802933)

[CHƯƠNG 1. TỔNG QUAN 11](#_Toc183802934)

[1.1. Giới thiệu về cửa hàng bán vật tư nông nghiệp 11](#_Toc183802935)

[1.2. Sự cần thiết của việc số hóa quản lý cửa hàng 11](#_Toc183802936)

[1.3. Giới thiệu các chức năng chính của ứng dụng 12](#_Toc183802937)

[CHƯƠNG 2. NGHIÊN CỨU LÝ THUYẾT 13](#_Toc183802938)

[2.1. Nền tảng .NET Framework 13](#_Toc183802939)

[2.1.1. Giới thiệu .NET Framework 13](#_Toc183802940)

[2.1.2. Kiến trúc của .NET Framework 13](#_Toc183802941)

[2.1.3. Các thư viện lớp cơ sở trong .NET Framework 14](#_Toc183802942)

[2.1.4. Ưu điểm của .NET Framework 15](#_Toc183802943)

[2.2. Ngôn ngữ lập trình C# 15](#_Toc183802944)

[2.2.1. Giới thiệu ngôn ngữ lập trình C# 15](#_Toc183802945)

[2.2.2. Các thành phần cơ bản của chương trình C# 16](#_Toc183802946)

[2.2.3. Ưu điểm của ngôn ngữ C# 17](#_Toc183802947)

[2.3. Lập trình hướng đối tượng trong C# 17](#_Toc183802948)

[2.3.1. Giới thiệu về lập trình hướng đối tượng 17](#_Toc183802949)

[2.3.2. Các thành phần cơ bản trong lập trình hướng đối tượng 17](#_Toc183802950)

[2.3.3. Phạm vi truy cập 18](#_Toc183802951)

[2.3.4. Các tính chất quan trọng trong lập trình hướng đối tượng 18](#_Toc183802952)

[2.3.5. Giao tiếp (Interface) trong lập trình hướng đối tượng 21](#_Toc183802953)

[2.3.5. Ưu điểm của lập trình hướng đối tượng 22](#_Toc183802954)

[2.4. Console Application 23](#_Toc183802955)

[2.4.1. Giới thiệu Console Application 23](#_Toc183802956)

[2.4.2. Ưu điểm Console Application 23](#_Toc183802957)

[2.5. Môi trường phát triển tích hợp Visual Studio 24](#_Toc183802958)

[2.5.1. Giới thiệu Visual Studio 24](#_Toc183802959)

[2.5.2. Ưu điểm của Visual Studio 25](#_Toc183802960)

[CHƯƠNG 3. HIỆN THỰC HÓA NGHIÊN CỨU 26](#_Toc183802961)

[3.1. Các bước hiện thực hóa nghiên cứu 26](#_Toc183802962)

[3.1.1. Phân tích yêu cầu 26](#_Toc183802963)

[3.1.2. Thiết kế hệ thống 26](#_Toc183802964)

[3.1.3. Xây dựng ứng dụng 27](#_Toc183802965)

[3.1.4. Kiểm thử và hoàn thiện ứng dụng 27](#_Toc183802966)

[3.2. Mô hình ứng dụng 28](#_Toc183802967)

[3.2.1. Sơ đồ lớp (Class Diagram) 28](#_Toc183802968)

[3.2.2. Lưu đồ xử lý (Flow Chart) 29](#_Toc183802969)

[CHƯƠNG 4. KẾT QUẢ NGHIÊN CỨU 34](#_Toc183802970)

[CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 35](#_Toc183802971)

[DANH MỤC TÀI LIỆU THAM KHẢO 36](#_Toc183802972)

[PHỤ LỤC 38](#_Toc183802973)

# DANH MỤC HÌNH ẢNH

[Hình 2.1. Ví dụ về chương trình C# đơn giản 16](#_Toc183803049)

[Hình 2.2. Ví dụ về tính đóng gói 18](#_Toc183803050)

[Hình 2.3. Ví dụ về tính thừa kế 19](#_Toc183803051)

[Hình 2.4. Ví dụ về tính trừu tượng 20](#_Toc183803052)

[Hình 2.5. Ví dụ về tính đa hình 21](#_Toc183803053)

[Hình 2.6. Ví dụ về giao tiếp 22](#_Toc183803054)

[Hình 2.7. Giao diện của Console Application 23](#_Toc183803055)

[Hình 2.8. Quy trình phát triển phần mềm trên Visual Studio 24](#_Toc183803056)

[Hình 3.1. Hình sơ đồ lớp của ứng dụng 28](#_Toc183803057)

[Hình 3.2. Hình lưu đồ xử lý tổng quát của ứng dụng 29](#_Toc183803058)

[Hình 3.3. Hình lưu đồ xử lý thao tác thêm dữ liệu 30](#_Toc183803059)

[Hình 3.4. Hình lưu đồ xử lý thao tác xóa dữ liệu 30](#_Toc183803060)

[Hình 3.5. Hình lưu đồ xử lý thao tác tìm kiếm dữ liệu 31](#_Toc183803061)

[Hình 3.6. Hình lưu đồ xử lý thao tác sửa dữ liệu 32](#_Toc183803062)

# DANH MỤC BẢNG BIỂU

# TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH

**Vấn đề nghiên cứu**

Đề tài "Xây dựng phần mềm ứng dụng quản lý cửa hàng bán vật tư nông nghiệp" xoay quanh việc số hóa quy trình quản lý và xử lý thông tin tại các cửa hàng bán vật tư nông nghiệp. Hiện nay, nhiều cửa hàng vẫn sử dụng các phương pháp quản lý thủ công, không chỉ làm tốn thời gian mà còn dễ xảy ra sai sót trong quá trình quản lý, đặc biệt khi lượng thông tin tại cửa hàng ngày càng nhiều. Do đó, việc nghiên cứu và xây dựng một ứng dụng sẽ giúp tự động hóa quy trình quản lý, hỗ trợ chủ cửa hàng trong việc quản lý thông tin sản phẩm, các giao dịch, nhân viên và nhà cung cấp. Ứng dụng giúp tiết kiệm thời gian, công sức và tối ưu hoạt động của cửa hàng bán vật tư nông nghiệp, góp phần nâng cao hiệu quả quản lý và thuận lợi trong quá trình kinh doanh.

**Các hướng tiếp cận và giải quyết vấn đề**

Phân tích yêu cầu: xác định các yêu cầu của ứng dụng như quản lý nhóm sản phẩm, thông tin từng sản phẩm, thành phần, công dụng, quản lý thông tin nhân viên để xác định các chức năng cần thiết và hướng giải quyết phù hợp.

Xây dựng mô hình: dựa trên các yêu cầu chức năng xây dựng các lớp, đối tượng, thuộc tính và phương thức xử lý, đồng thời xác định các mối quan hệ giữa các đối tượng đảm bảo toàn vẹn dữ liệu.

Lập trình ứng dụng: sử dụng ngôn ngữ C# và .NET Framework với phương pháp lập trình hướng đối tượng để xây dựng ứng dụng gồm các chức năng cơ bản như thêm, sửa, xóa và tìm kiếm trên thông tin sản phẩm. Tạo các menu lựa chọn trực quan để dễ dàng tương tác với các chức năng của ứng dụng.

Kiểm tra và chỉnh sửa ứng dụng: đảm bảo hoạt động tốt, các chức năng hoạt động đúng với yêu cầu.

**Kết quả đạt được**

Xây dựng thành công ứng dụng quản lý cửa hàng bán vật tư nông nghiệp, đáp ứng đúng với các yêu cầu chức năng đã đặt ra.

Nắm vững kiến thức về lập trình C#, rèn luyện kỹ năng lập trình hướng đối tượng và cải thiện khả năng phân tích yêu cầu, xây dựng ứng dụng.

Ứng dụng không chỉ giải quyết các vấn đề nghiên cứu mà còn là nền tảng cho việc phát triển các hệ thống quản lý thực tế trong tương lai.

# MỞ ĐẦU

**Lý do chọn đề tài**

Ngày nay, các cửa hàng bán vật tư nông nghiệp đóng vai trò quan trọng trong việc cung cấp các sản phẩm thiết yếu cho công việc nông nghiệp của người dân. Tuy nhiên, việc quản lý thông tin tại các cửa hàng này thường gặp nhiều khó khăn do quy trình thủ công, dẫn đến những sai sót trong quản lý và xử lý thông tin. Việc số hóa quy trình này thông qua một ứng dụng quản lý sẽ giúp tối ưu hóa hoạt động kinh doanh, nâng cao hiệu quả và giảm thiểu sai sót. Vì vậy, việc thực hiện đề tài “Xây dựng phần mềm ứng dụng quản lý cửa hàng bán vật tư nông nghiệp” là cần thiết.

**Mục đích nghiên cứu**

Mục đích chính của nghiên cứu là xây dựng một ứng dụng quản lý cửa hàng bán vật tư nông nghiệp, nhằm giúp các cửa hàng dễ dàng hơn trong việc theo dõi và quản lý thông tin về sản phẩm. Ứng dụng này không chỉ giúp chủ cửa hàng tiết kiệm thời gian mà còn nâng cao độ chính xác trong công việc quản lý. Bên cạnh đó, việc thực hiện đồ án cũng nhằm giúp áp dụng kiến thức lý thuyết vào thực tiễn và rèn luyện kỹ năng lập trình ứng dụng.

**Đối tượng nghiên cứu**

Đối tượng nghiên cứu của đề tài là các cửa hàng bán vật tư nông nghiệp. Đề tài sẽ nghiên cứu và xây dựng ứng dụng với các chức năng cần thiết để quản lý thông tin về danh mục sản phẩm, thành phần, công dụng, thương hiệu nhà sản xuất và thông tin nhân viên chịu trách nhiệm từng nhóm sản phẩm.

**Phạm vi nghiên cứu**

Phạm vi nghiên cứu của đề tài giới hạn trong việc xây dựng một ứng dụng quản lý dành cho các cửa hàng bán vật tư nông nghiệp. Ứng dụng sẽ được xây dựng bằng ngôn ngữ lập trình C# và .NET Framework bằng phương pháp lập trình hướng đối tượng với các chức năng cơ bản như thêm, sửa, xóa và tìm kiếm thông tin sản phẩm.

# CHƯƠNG 1. TỔNG QUAN

## 1.1. Giới thiệu về cửa hàng bán vật tư nông nghiệp

Trong nền kinh tế hiện đại, cửa hàng bán vật tư nông nghiệp đóng một vai trò quan trọng trong việc cung cấp các sản phẩm thiết yếu như phân bón, thuốc bảo vệ thực vật, hạt giống và công cụ nông nghiệp, giúp nông dân cải thiện chất lượng sản phẩm nông nghiệp. Với việc tư vấn kỹ thuật, cung cấp kiến thức và hướng dẫn sử dụng sản phẩm, các cửa hàng vật tư nông nghiệp không chỉ là điểm mua sắm mà còn là nơi người dân học hỏi và trao đổi kinh nghiệm. Điều này giúp người dân áp dụng những phương pháp canh tác hiệu quả hơn, bảo vệ môi trường, giảm thiểu rủi ro và nâng cao hiệu quả kinh tế. Do đó, cửa hàng bán vật tư nông nghiệp trở thành một nhân tố rất quan trọng trong quá trình vận hành và phát triển của ngành nông nghiệp.

## 1.2. Sự cần thiết của việc số hóa quản lý cửa hàng

Mặc dù các cửa hàng bán vật tư nông nghiệp là nhân tố quan trọng trong ngành nông nghiệp nhưng do một số cửa hàng vẫn còn sử dụng hình thức quản lý truyền thống dẫn đến khó khăn trong việc kiểm soát thông tin sản phẩm, giao dịch và nhân viên. Ngày nay, với sự phát triển của công nghệ thông tin đã mở ra nhiều cơ hội cho việc số hóa quy trình quản lý trong ngành nông nghiệp. Việc ứng dụng công nghệ vào quản lý cửa hàng không chỉ giúp tự động hóa các công việc thủ công mà còn tăng cường độ chính xác trong xử lý thông tin. Lợi ích của việc số hóa quy trình quản lý:

* Tiết kiệm thời gian và công sức: Việc sử dụng ứng dụng quản lý giúp giảm thiểu thời gian cho các công việc như theo dõi tồn kho, ghi chép, quản lý thông tin sản phẩm và nhân viên.
* Cải thiện chất lượng dịch vụ: Ứng dụng cung cấp khả năng lưu trữ và xử lý thông tin chính xác và nhanh chóng từ đó nâng cao trải nghiệm và sự hài lòng về quy trình phục vụ khách hàng.

## 1.3. Giới thiệu các chức năng chính của ứng dụng

Quản lý danh mục các nhóm sản phẩm cho phép tạo, sửa, xóa và cập nhật thông tin về các nhóm sản phẩm trong cửa hàng. Các nhóm sản phẩm giúp phân loại mặt hàng theo đặc điểm chung, dễ dàng quản lý và tìm kiếm.

Quản lý danh mục các sản phẩm có trong cửa hàng giúp thêm mới, sửa đổi và cập nhật thông tin của từng sản phẩm cụ thể trong cửa hàng, bao gồm tên sản phẩm, giá bán, số lượng tồn kho và nhóm sản phẩm. Điều này hỗ trợ việc kiểm soát hàng hóa và theo dõi tình hình tồn kho của từng sản phẩm.

Quản lý danh mục thành phần có trong sản phẩm cho phép quản lý các thành phần trong mỗi sản phẩm. Đây là thông tin quan trọng đối với các sản phẩm như phân bón hoặc thuốc bảo vệ thực vật, giúp hiểu rõ thành phần hóa học hoặc sinh học trong sản phẩm để đáp ứng yêu cầu của khách hàng.

Quản lý danh mục các công dụng của từng sản phẩm giúp lưu trữ và quản lý thông tin về công dụng chính của mỗi sản phẩm. Thông tin công dụng giúp khách hàng chọn lựa sản phẩm phù hợp với nhu cầu.

Quản lý danh mục các nhân viên của cửa hàng chịu trách nhiệm về một nhóm sản phẩm cho phép quản lý danh sách nhân viên và phân công trách nhiệm cho từng nhóm sản phẩm như quản lý, tư vấn hoặc hỗ trợ khách hàng đối với các sản phẩm trong nhóm được giao, giúp cửa hàng quản lý công việc hiệu quả hơn.

Quản lý danh mục thương hiệu (tên) nhà sản xuất sản phẩm giúp quản lý thông tin về các thương hiệu nhà sản xuất của sản phẩm trong cửa hàng. Người dùng có thể thêm mới, sửa đổi và cập nhật thông tin nhà sản xuất, giúp phân biệt các sản phẩm theo thương hiệu và cung cấp nguồn gốc rõ ràng cho khách hàng.

# CHƯƠNG 2. NGHIÊN CỨU LÝ THUYẾT

## 2.1. Nền tảng .NET Framework

### 2.1.1. Giới thiệu .NET Framework

.NET Framework là một nền tảng công nghệ do Microsoft phát hành vào năm 2002, cung cấp môi trường để phát triển và chạy các ứng dụng trên hệ điều hành Windows và các dịch vụ web. Với những thư viện phong phú và mạnh mẽ, .NET Framework hỗ trợ hiệu quả cho việc xây dựng các ứng dụng từ đơn giản đến phức tạp, giúp tối ưu hóa quá trình phát triển và nâng cao hiệu suất hoạt động của ứng dụng.

### 2.1.2. Kiến trúc của .NET Framework

.NET Framework được tạo ra để giải quyết các mục tiêu sau:

* Cung cấp môi trường phát triển ứng dụng nhất quán
* Giảm thiểu xung đột khi triển khai và quản lý phiên bản phần mềm
* Tăng cường an toàn và bảo mật khi thực thi mã
* Cải thiện hiệu suất so với lập trình theo kịch bản
* Hỗ trợ đa nền tảng và tính nhất quán trong phát triển ứng dụng
* Tương thích và dễ tích hợp với bất kỳ loại mã nào.

.NET Framework gồm hai phần chính là: Common Language Runtime (CLR) và Framework Class Library (FCL).

#### 2.1.2.1. Common Language Runtime (CLR)

CLR chịu trách nhiệm về quản lý thực thi mã cung cấp các chức năng sau:

* Quản lý bộ nhớ: cấp phát và giải phóng bộ nhớ cho các đối tượng giúp giảm thiểu lỗi bộ nhớ.
* Thực thi mã: biên dịch mã nguồn thành mã máy để hệ thống hiểu và thực hiện.
* An toàn mã: đảm bảo thực thi mã an toàn toàn thông qua xác minh quyền truy cập.
* Quản lý luồng: đảm bảo chạy hiệu quả các tác vụ đồng thời.
* Quản lý lỗi và ngoại lệ: đảm bảo phát hiện và xử lý các lỗi trong quá trình thực thi.

#### 2.1.2.2. Framework Class Library (FCL)

FCL là tập hợp các kiểu dữ liệu có thể tái sử dụng, cung cấp các thành phần để xây dựng, phát triển ứng dụng và dịch vụ web như:

* Ứng dụng Console và GUI Windows: Cung cấp các lớp để xây dựng các ứng dụng Console, Windows Forms, và Windows Presentation Foundation (WPF).
* Các dịch vụ web và ứng dụng server: Hỗ trợ phát triển ứng dụng ASP.NET, các dịch vụ sử dụng Windows Communication Foundation (WCF) và các ứng dụng sử dụng Windows Workflow Foundation (WF).

### 2.1.3. Các thư viện lớp cơ sở trong .NET Framework

.NET Framework hỗ trợ các thư viện cơ sở với rất nhiều lớp, phục vụ cho việc xây dựng và phát triển phần mềm. Một số thư viện lớp thông dụng để xây dựng ứng dụng:

* System: bao gồm các lớp cơ bản và lớp các kiểu dữ liệu phổ biến, các sự kiện, giao diện, thuộc tính và xử lý các ngoại lệ. Một số lớp trong System: Console, Exception, Array, Object, Enum, Int32, Boolean, String, Math, Random, Type,…
* System.IO: chứa các kiểu dữ liệu cho phép đọc và ghi vào các tệp tin và luồng dữ liệu, có các kiểu dữ liệu hỗ trợ cho tệp tin và thư mục. Một số lớp trong System.IO: File, Stream, Directory, Path,…
* System.Collection: chứa các giao tiếp và lớp định nghĩa các tập hợp đối tượng khác nhau. Một số lớp trong System.Collection: ArrayList, BittArray, Stack, Queue, HashTable,…
* System.Collection.Generic: chứa các giao tiếp và lớp định nghĩa các tập hợp tổng quát, giúp tạo ra các tập hợp với kiểu dữ liệu cụ thể. Một số lớp trong System.Collection.Generic: Comparer<T>, HashSet<T>, List<T>, LinkList<T>, Stack<T>, Queue<T>, SortedList<TKey, TValue>,…
* System.Text: chứa các lớp để đại diện cho mã hóa ký tự ASCII và Unicode, các lớp cơ sở để chuyển đổi giữa các khối ký tự và khối byte, các lớp hỗ trợ thao tác định dạng các chuỗi. Một số lớp trong System.Text: Json, StringBuilder, Encoder, Decoder,…
* System.Threading: chứa các lớp và giao tiếp giúp lập trình đa luồng, đồng bộ hóa hoạt động giữa các luồng và truy cập dữ liệu. Một số lớp trong System.Threading: Monitor, Mutex, ThreadPool, Timer,…

### 2.1.4. Ưu điểm của .NET Framework

Cung cấp rất nhiều thư viện và công cụ hỗ trợ, giúp xây dựng ứng dụng nhanh chóng mà không cần viết lại từ đầu các tính năng cơ bản.

Cho phép phát triển ứng dụng trên nhiều hệ điều hành, cũng như ứng dụng di động và web.

Hỗ trợ nhiều ngôn ngữ lập trình như C#, VB.NET, F#, cho phép chọn ngôn ngữ phù hợp và dễ dàng tích hợp mã từ nhiều ngôn ngữ khác nhau trong một ứng dụng.

.NET Framework là một nền tảng phổ biến với cộng đồng đông đảo, cùng với sự hỗ trợ mạnh mẽ từ Microsoft, giúp dễ dàng tìm kiếm tài liệu và trợ giúp khi cần.

Với tính linh hoạt và khả năng tương thích đa nền tảng, .NET Framework đã trở thành nền tảng mạnh mẽ và là một lựa chọn phổ biến trong xây dựng và phát triển ứng dụng.

## 2.2. Ngôn ngữ lập trình C#

### 2.2.1. Giới thiệu ngôn ngữ lập trình C#

C# là một ngôn ngữ lập trình thiên về hướng đối tượng do Microsoft phát hành năm 2000. Đây là ngôn ngữ chính và phổ biến nhất của nền tảng .NET. C# không chỉ hỗ trợ lập trình ứng dụng trên máy tính, di động, mà còn cho phép phát triển trang web, trò chơi, thực tế ảo và nhiều ứng dụng khác. Dựa trên phương pháp lập trình hướng đối tượng, C# cung cấp khả năng viết mã an toàn và hiệu quả với hiệu suất cao, đặc biệt là lập trình chức năng. Chính vì những khả năng mà C# cung cấp, ngôn ngữ này đã trở thành một trong những ngôn ngữ lập trình phổ biến nhất thế giới.

### 2.2.2. Các thành phần cơ bản của chương trình C#

Từ khóa using: dùng để khai báo sử dụng thư viện của .NET Framework.

Namespace: có thể hiểu là vùng làm việc, giúp giới hạn phạm vi làm việc của các thành phần trong chương trình, tránh xung đột với các thành phần khác. Một thành phần sẽ chỉ hoạt động tại vùng làm việc của nó mà không ảnh hưởng đến thành phần ở vùng làm việc khác.

Class: được hiểu là lớp, đây là nơi chứa các dữ liệu và các hàm xử lý. Một chương trình C# cần ít nhất lớp và phải nằm trong namespace.

Phương thức Main: là phương thức sẽ được gọi đầu tiên khi chương trình chạy và các lệnh trong phương thức này cũng sẽ chạy. Phương thức Main phải nằm trong class.

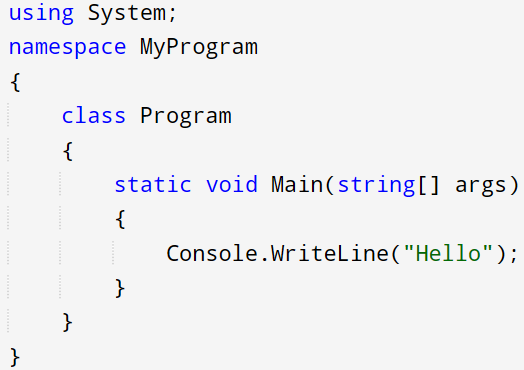
Biến và kiểu dữ liệu: tương tự như các ngôn ngữ lập trình khác, biến dùng để lưu giá trị dữ liệu và kiểu dữ liệu giúp khai báo kiểu lưu giá trị dữ liệu của biến.

Cấu trúc rẽ nhánh: C# hỗ trợ các cấu trúc rẽ nhánh như câu lệnh if (if … else,

if … else if … else) và câu lệnh switch … case, giúp kiểm tra và xử lý điều kiện.

Vòng lặp: dùng để thực hiện các công việc như lặp lại một khối lệnh nhiều lần hay duyệt qua các phần tử con của một thành phần như mảng, danh sách, chuỗi,… C# hỗ trợ các câu lệnh lặp như for, while, do … while và foreach.

Generic: cho phép xây dựng các lớp, phương thức mà không xác định trước kiểu dữ liệu, sử dụng các kiểu dữ liệu làm tham số, giúp tăng tính linh hoạt và tái sử dụng mã nguồn, không cần phải viết lại mã cho mỗi kiểu dữ liệu.



Hình 2.1. Ví dụ về chương trình C# đơn giản

### 2.2.3. Ưu điểm của ngôn ngữ C#

C# có cú pháp rõ ràng, dễ hiểu, đặc biệt dễ tiếp cận với người mới học lập trình hoặc những ai đã quen với các ngôn ngữ khác như Java hay C++ dễ dàng hiểu và viết mã.

C# là một ngôn ngữ rất mạnh nhờ tích hợp nhiều tính năng hiện đại như xử lý bất đồng bộ, thư viện phong phú, quản lý bộ nhớ tự động, giúp tối ưu hóa hiệu suất và giảm thiểu lỗi.

C# là một ngôn ngữ lập trình hướng đối tượng, có các khái niệm như lớp, đối tượng, giúp tổ chức mã dễ bảo trì và mở rộng.

C# có thể chạy trên nhiều hệ điều hành, đồng thời có thể phát triển ứng dụng di động và web, giúp tăng tính linh hoạt khi phát triển ứng dụng cho nhiều nền tảng.

C# là một ngôn ngữ lập trình linh hoạt và mạnh mẽ nhưng đơn giản và dễ học. Với những tính năng nổi bật C# đã trở thành một trong các ngôn ngữ phổ biến nhất và là lựa chọn tốt cho các dự án xây dựng và phát triển ứng dụng.

## 2.3. Lập trình hướng đối tượng trong C#

### 2.3.1. Giới thiệu về lập trình hướng đối tượng

Lập trình hướng đối tượng (OOP – Object Oriented Programing) là một phương pháp lập trình mà cấu trúc của nó được xây dựng dựa trên “đối tượng”. Trong đối tượng có các thuộc tính (properties) dùng để chứa dữ liệu, các đối tượng tương tác với nhau thông qua phương thức (methods) để xử lý một nhiệm vụ cụ thể. Để ngăn chặn truy cập dữ liệu tùy tiện, đảm bảo an toàn dữ liệu bằng việc giới hạn quyền truy cập vào các thuộc tính và phương thức của đối tượng ở những mức độ cụ thể.

### 2.3.2. Các thành phần cơ bản trong lập trình hướng đối tượng

Lớp: được hiểu như là bản thiết kế hay cấu trúc mẫu cho đối tượng, nó định nghĩa các thuộc tính và phương thức xử lý cho đối tượng. Một lớp chứa các liệu thành viên, hàm thành viên.

Đối tượng: có thể được xem như là một thực thể trong thế giới thực chứa các thành phần dữ liệu đặc trưng của riêng nó và các hành động để xử lý những dữ liệu đó. Đối tượng là thể hiện của lớp đã được định nghĩa.

Dữ liệu thành viên: dùng để lưu trữ thông tin, trạng thái của đối tượng.

Hàm thành viên: là các hành động thực hiện xử lý dữ liệu của đối tượng.

### 2.3.3. Phạm vi truy cập

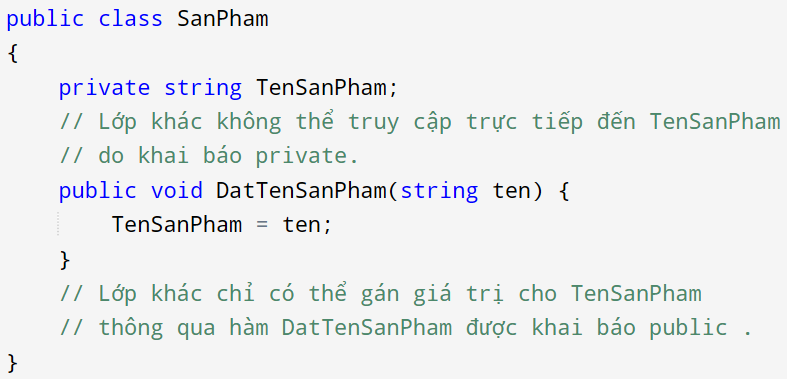
Phạm vi truy cập lớp là cách giới hạn quyền truy cập vào các thuộc tính và phương thức, giúp bảo vệ dữ liệu tránh truy cập và thay đổi tùy ý dữ liệu của lớp. Các phạm vi truy cập phổ biến:

* Public: các thuộc tính và phương thức sẽ có thể được truy cập ở bất kỳ đâu trong chương trình.
* Private: các thuộc tính và phương thức chỉ có thể truy cập trong phạm vi của một lớp.
* Protected: các thuộc tính và phương thức được truy cập trong phạm vi của một lớp và trong lớp con của lớp đó khi có kế thừa.
* Internal: các thuộc tính và phương thức được truy cập trong phạm vi của một tập hợp mã nguồn (dự án, thư viện).

### 2.3.4. Các tính chất quan trọng trong lập trình hướng đối tượng

#### 2.3.4.1. Đóng gói (Escapsulation)

Đóng gói có thể hiểu là sử dụng các phạm vi truy cập như private, public, protected, internal để kiểm soát quyền truy cập các thành phần trong chương trình. Đóng gói cho phép ngăn quyền truy cập các dữ liệu và hàm thành viên, chỉ cho phép truy cập thông qua các hàm công khai để đảm bảo an toàn dữ liệu.



Hình 2.2. Ví dụ về tính đóng gói

#### 2.3.4.2. Thừa kế (Inheritance)

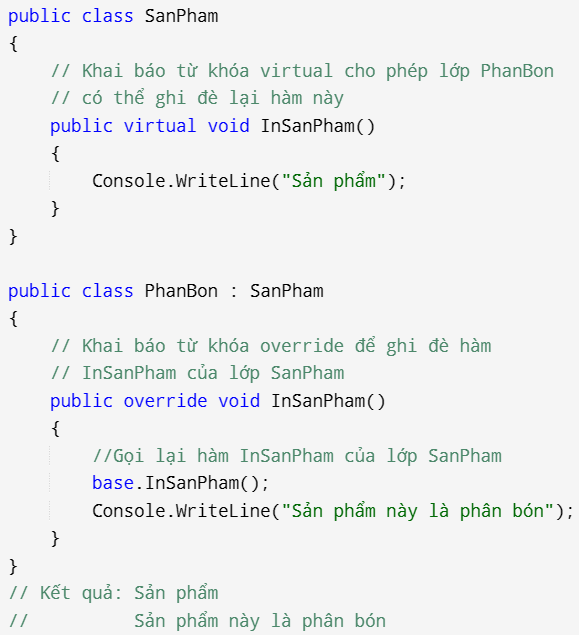
Thừa kế cho phép một lớp có thể sử dụng lại các dữ liệu và hàm thành viên từ một lớp khác, tránh trùng lặp và tái sử dụng mã.

Lớp thừa kế gọi là lớp dẫn xuất. Lớp được thừa kế gọi là lớp cơ sở.

Cú pháp khai báo một thừa kế: <Tên lớp dẫn xuất> : <Tên lớp cơ sở>

Lớp dẫn xuất có thể gọi các thành phần của lớp cơ sở bằng từ khóa base.

Một hàm thành viên của lớp cơ sở có thể được định nghĩa lại ở lớp dẫn xuất bằng cách khai báo hàm với từ khóa virtual ở lớp cơ sở và dùng từ khóa override để khai báo lại hàm đó ở lớp dẫn xuất, tại đây có thể định nghĩa lại hàm thành viên đã thừa kế.

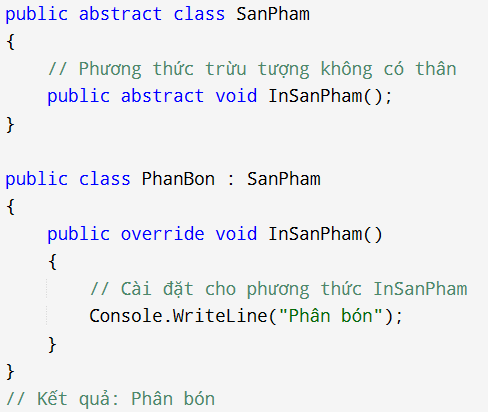


Hình 2.3. Ví dụ về tính thừa kế

#### 2.3.4.3. Trừu tượng (Abstraction)

Lớp trừu tượng là lớp không tạo ra được đối tượng. Lớp trừu tượng được khai báo bằng từ khóa abstract. Lớp có phương thức trừu tượng được xem là lớp trừu tượng.

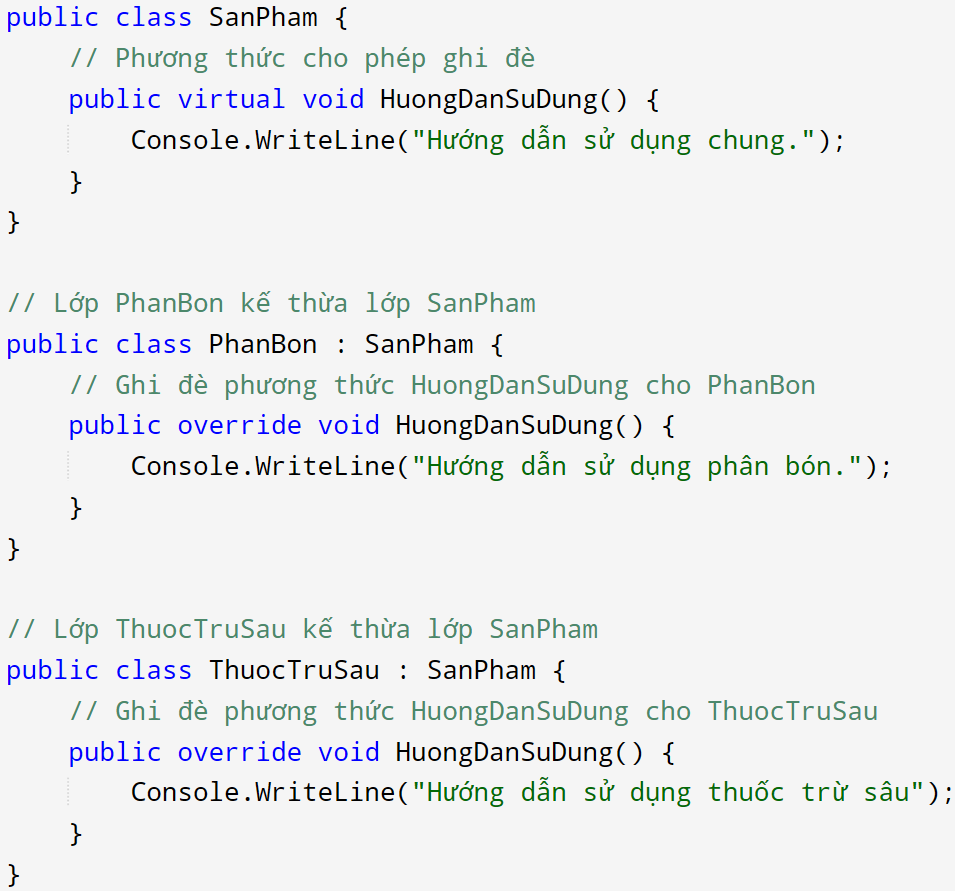
Trong lớp trừu tượng có các phương thức đã cài đặt và phương thức trừu tượng, các phương thức trừu tượng không cần cài đặt mà chỉ cần khai báo ở lớp trừu tượng. Mặc định, phương thức trừu tượng được đặt là virtual. Các phương thức trừu tượng phải được cài đặt ở lớp thừa kế và nó phải khai báo từ khóa override để có thể ghi đè.



Hình 2.4. Ví dụ về tính trừu tượng

#### 2.3.4.4. Đa hình (Polymorphism)

Đa hình cho phép các lớp khác nhau thực thi cùng một phương thức theo các cách khác nhau. Trong C#, đa hình được thực hiện thông qua việc ghi đè phương thức thừa kế và cài đặt các giao tiếp.

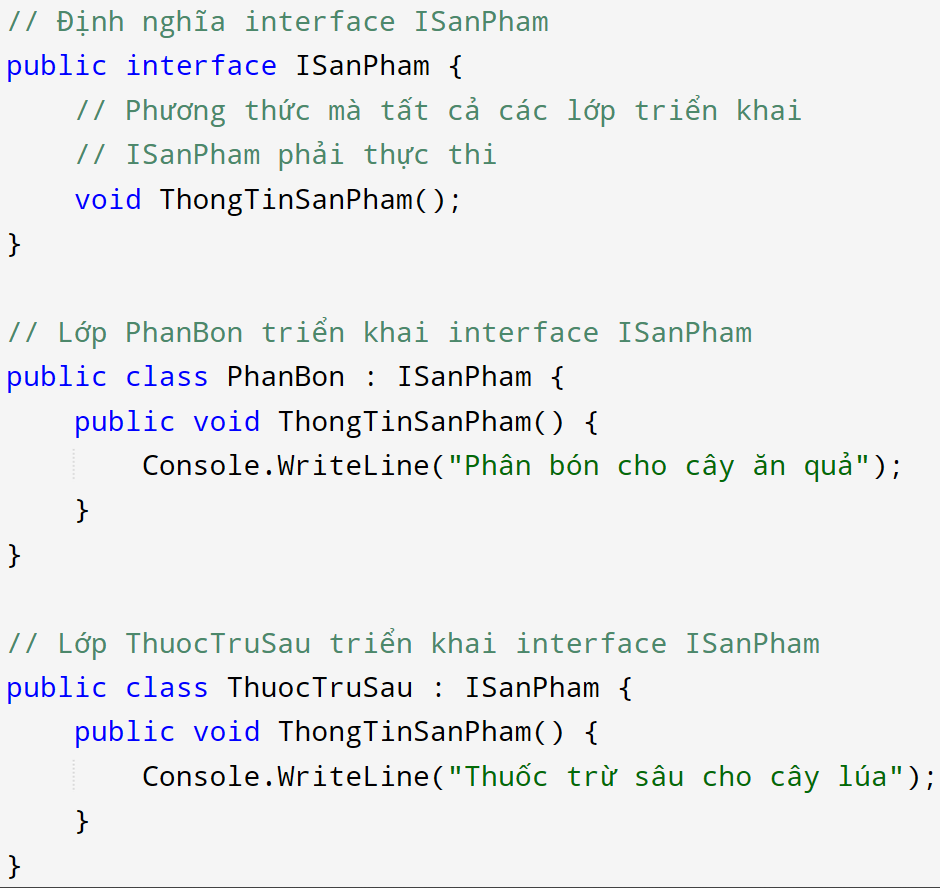


Hình 2.5. Ví dụ về tính đa hình

### 2.3.5. Giao tiếp (Interface) trong lập trình hướng đối tượng

Interface có thể hiểu là một kiểu trừu tượng hoàn toàn, giúp định nghĩa một tập hợp các phương thức chung mà không cần cài đặt. Trong interface không chứa các trường dữ liệu hay thuộc tính, nó chỉ chứa các phương thức không có thân.

Các phương thức của interface được cài đặt ở các lớp triển khai (implement) interface đó. Một lớp có thể triển khai nhiều interface tạo nên tính đa thừa kế, khác với thừa kế thông thường (một lớp dẫn xuất chỉ có thể thừa kế từ một lớp cơ sở).



Hình 2.6. Ví dụ về giao tiếp

### 2.3.5. Ưu điểm của lập trình hướng đối tượng

Cho phép tái sử dụng mã thông qua các lớp và kế thừa. Ngoài ra, các lớp và đối tượng có thể được tái sử dụng trong các dự án hoặc phần khác nhau của cùng một dự án.

Mã nguồn được tổ chức thành các lớp độc lập, giúp dễ dàng bảo trì và mở rộng mã trong các lớp riêng mà không ảnh hưởng đến phần còn lại của ứng dụng.

Cho phép thêm các tính năng mới bằng cách bổ sung các lớp mới hoặc mở rộng các lớp mà không ảnh hưởng nhiều đến mã hiện tại.

Tính đóng gói giúp bảo vệ dữ liệu và phương thức trong lớp, ngăn chặn các thay đổi hoặc truy cập trái phép vào dữ liệu của lớp.

Tính đa hình giúp tăng cường tính linh hoạt và khả năng mở rộng, giúp thay đổi cách thức hoạt động của chương trình mà không cần thay đổi các đoạn mã hiện tại.

Mô hình hóa các đối tượng và mối quan hệ của chúng trong thế giới thực. Các lớp và đối tượng có thể đại diện cho bất kỳ thực thể nào trong ứng dụng, giúp dễ hiểu và dễ triển khai.

Lập trình hướng đối là một phương pháp mạnh mẽ và hiệu quả trong việc phát triển phần mềm hiện đại. Đây một kỹ năng quan trọng và không thể thiếu đối với công việc xây dựng và phát triển phần mềm.

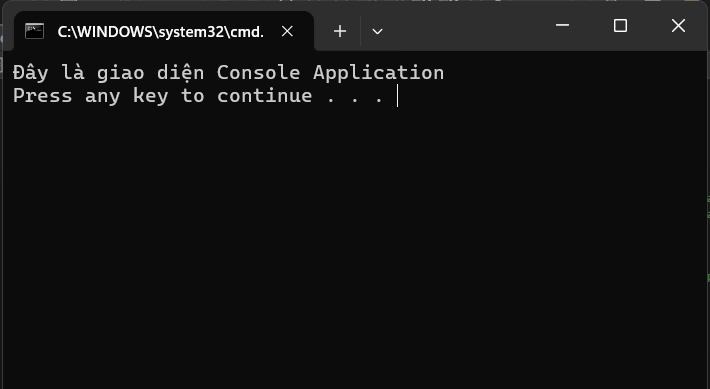
## 2.4. Console Application

### 2.4.1. Giới thiệu Console Application

Console Application còn gọi là ứng dụng dòng lệnh, là một dạng phần mềm đơn giản mà các tương tác được thực hiện thông qua việc nhập các lệnh văn bản. Console Application không sử dụng nút bấm hoặc các thành phần đồ họa, chỉ cần nhập lệnh vào bàn phím và nhận lại kết quả hiển thị dưới dạng văn bản.

Trong lập trình C#, một ứng dụng dòng lệnh sử dụng ba luồng dữ liệu cơ bản: luồng đầu vào, luồng đầu ra và luồng lỗi. Mặc dù không có giao diện đồ họa, các ứng dụng dòng lệnh rất hiệu quả trong việc thực hiện các chức năng hệ thống.

Trong môi trường .NET Framework, lớp System.Console là một trong những thành phần quan trọng nhất để xây dựng ứng dụng dòng lệnh. Lớp này hỗ trợ việc đọc và ghi dữ liệu, cho phép xử lý các chức năng như nhập liệu và xuất kết quả ra màn hình.



Hình 2.7. Giao diện của Console Application

### 2.4.2. Ưu điểm Console Application

Console Application không yêu cầu nhiều tài nguyên hệ thống như các ứng dụng giao diện đồ họa, giúp ứng dụng dòng lệnh chạy mượt mà trên các hệ thống cấu hình thấp.

Console Application dễ dàng thiết kế và phát triển nhờ vào giao diện đơn giản chỉ sử dụng văn bản. Điều này giúp giảm thời gian và chi phí phát triển.

Do không phụ thuộc vào các thành phần đồ họa, các Console Application có thể chạy trên nhiều hệ điều hành khác nhau mà không gặp phải vấn đề về tương thích hệ điều hành.

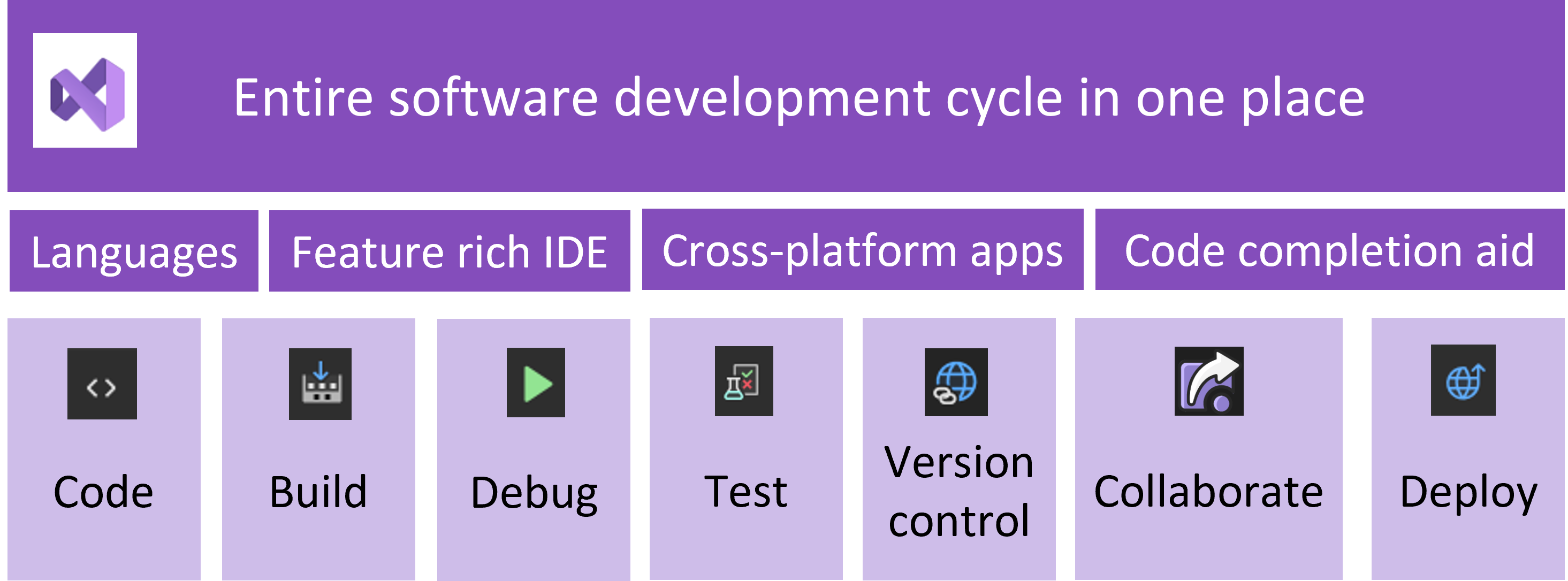
Console Application có thể thực hiện các tác vụ nhanh do không cần xử lý đồ họa.

Console Application mang lại nhiều tính năng quan trọng, đặc biệt phù hợp với các tác vụ tự động hóa, quản lý hệ thống và phát triển phần mềm. Nhờ tính đơn giản, hiệu quả và khả năng tương thích cao, Console Application là một công cụ không thể thiếu trong môi trường làm việc với máy tính.

## 2.5. Môi trường phát triển tích hợp Visual Studio

### 2.5.1. Giới thiệu Visual Studio

Visual Studio là một môi trường phát triển tích hợp do Microsoft phát hành vào năm 1997. Đây là môi trường hỗ trợ triển khai và phát triển các ứng dụng desktop, ứng dụng web, ứng dụng di động, đám mây,… trên nhiều nền tảng và ngôn ngữ lập trình khác nhau. Visual Studio cung cấp nhiều phiên bản phục vụ nhu cầu sử dụng của đa dạng người dùng. Với giao diện trực quan, hỗ trợ nhiều tính năng mạnh mẽ và khả năng mở rộng linh hoạt, Visual Studio trở thành công cụ phổ biến giúp cho việc phát triển phần mềm trở nên dễ dàng và hiệu quả hơn.



Hình 2.8. Quy trình phát triển phần mềm trên Visual Studio

### 2.5.2. Ưu điểm của Visual Studio

Cung cấp một môi trường phát triển phong phú, hỗ trợ nhiều ngôn ngữ lập trình như C#, C++, Python, JavaScript,… cho phép thực hiện các dự án đa dạng mà không cần phải chuyển đổi giữa các IDE khác nhau, giúp tiết kiệm thời gian và tăng tính linh hoạt.

Trang bị một bộ công cụ đa dạng cung cấp các tính năng như hỗ trợ tự động hoàn thành mã, gợi ý mã, cùng với khả năng gỡ lỗi và kiểm thử, giúp tối ưu hóa năng suất làm việc và giảm thiểu lỗi trong quá trình phát triển.

Cho phép xây dựng ứng dụng trên nhiều nền tảng khác nhau như Windows, MacOS, Android, IOS và web, giúp cho việc triển khai trở nên dễ dàng hơn, từ đó mở rộng đối tượng người dùng và tăng cường tính linh hoạt của sản phẩm.

Hỗ trợ Git và các hệ thống quản lý mã nguồn khác, cho phép dễ dàng lưu trữ và quản lý mã nguồn với các thao tác như commit, pull, push và merge có thể thực hiện ngay trong IDE.

Hỗ trợ cộng tác với tính năng Live Share cho phép các thành viên trong nhóm làm việc trên cùng một dự án mà không cần gặp mặt trực tiếp. Tính năng này cải thiện hiệu suất làm việc nhóm và thúc đẩy sự hợp tác hiệu quả.

Cho phép triển khai ứng dụng nhanh chóng lên các nền tảng đám mây như Azure với các công cụ triển khai tự động hóa quy trình, tiết kiệm thời gian và giảm thiểu rủi ro.

Với việc tích hợp nhiều tính năng mạnh mẽ và đa dạng, Visual Studio giúp tối ưu hóa quy trình phát triển phần mềm từ việc viết mã, kiểm thử cho đến triển khai. Đây là công cụ mạnh mẽ có thể đáp ứng các nhu cầu lập trình từ đơn giản đến phức tạp.

# CHƯƠNG 3. HIỆN THỰC HÓA NGHIÊN CỨU

## 3.1. Các bước hiện thực hóa nghiên cứu

### 3.1.1. Phân tích yêu cầu

#### 3.1.1.1. Xác định mục tiêu

Ứng dụng cần hỗ trợ quản lý cửa hàng bán vật tư nông nghiệp với các danh mục như: nhóm sản phẩm, sản phẩm, thành phần, công dụng, thương hiệu nhà sản xuất và nhân viên chịu trách nhiệm cho nhóm sản phẩm.

Cung cấp các thao tác cơ bản, bao gồm: xem, thêm, sửa, xóa, tìm kiếm để quản lý dữ liệu.

#### 3.1.1.2. Phân tích các thực thể

Các thực thể được xác định:

* Nhóm sản phẩm: Phân loại các sản phẩm theo nhóm (ví dụ: phân bón hữu cơ, phân bón hóa học).
* Sản phẩm: Bao gồm thông tin chi tiết như tên, giá, số lượng, nhóm sản phẩm và thương hiệu.
* Nhà sản xuất: Đại diện cho thương hiệu cung cấp các sản phẩm.
* Nhân viên: Người chịu trách nhiệm quản lý các nhóm sản phẩm cụ thể.
* Thành phần: Bao gồm các thông tin về thành phần có trong sản phẩm.
* Công dụng: Thể hiện cho công dụng mà sản phẩm mang lại.

#### 3.1.1.3. Xác định phạm vi

Ứng dụng này là loại Console application được viết bằng ngôn ngữ C# và .NET Framework dựa trên phương pháp lập trình hướng đối tượng.

Dữ liệu sẽ được lưu trữ và xử lý trực tiếp trong bộ nhớ bằng cấu trúc dữ liệu là danh sách (List).

Giao diện dòng lệnh tương tác bằng bàn phím với menu lựa chọn chức năng trực quan.

### 3.1.2. Thiết kế hệ thống

#### 3.1.2.1. Xây dựng sơ đồ lớp (Class Diagram)

Mô tả các lớp tương ứng với các thực thể cần thực hiện quản lý trong ứng dụng

Các lớp với nhiệm vụ khác nhau: lớp dữ liệu, lớp nghiệp vụ, lớp trừu tượng và giao diện.

Các thuộc tính và phương thức của từng lớp được xác định để đáp ứng việc lưu trữ và xử lý dữ liệu.

#### 3.1.2.2. Xây dựng lưu đồ xử lý (Flow Chart)

Thiết kế các lưu đồ xử lý tổng quát cho ứng dụng và các thao tác thêm, sửa, xóa và tìm kiếm.

Mục đích là mô tả rõ ràng luồng hoạt động từ khi người dùng thực hiện chức năng đến khi dữ liệu được xử lý và cho ra kết quả.

### 3.1.3. Xây dựng ứng dụng

Dựa trên thiết kế đã xây dựng, tiến hành xây dựng ứng dụng:

* Xây dựng giao diện để lớp nghiệp vụ triển khai các thao tác: thêm, sửa, xóa và tìm kiếm.
* Xây dựng lớp trừu tượng chứa các thông tin dùng chung cho các lớp dữ liệu.
* Xây dựng các lớp dữ liệu kề thừa từ lớp trừu tượng. Trong đó, có các thuộc tính và phương thức cho từng đối tượng
* Xây dựng lớp nghiệp vụ có chức năng quản lý danh mục các đối tượng từ lớp dữ liệu và cài đặt các phương thức đã xây dựng ở giao diện.
* Tạo menu hiển thị chức năng và lựa chọn tương ứng để người dùng dễ dàng thực hiện các thao tác.
* Hiển thị thông tin rõ ràng và xử lý nhập liệu từ bàn phím.

### 3.1.4. Kiểm thử và hoàn thiện ứng dụng

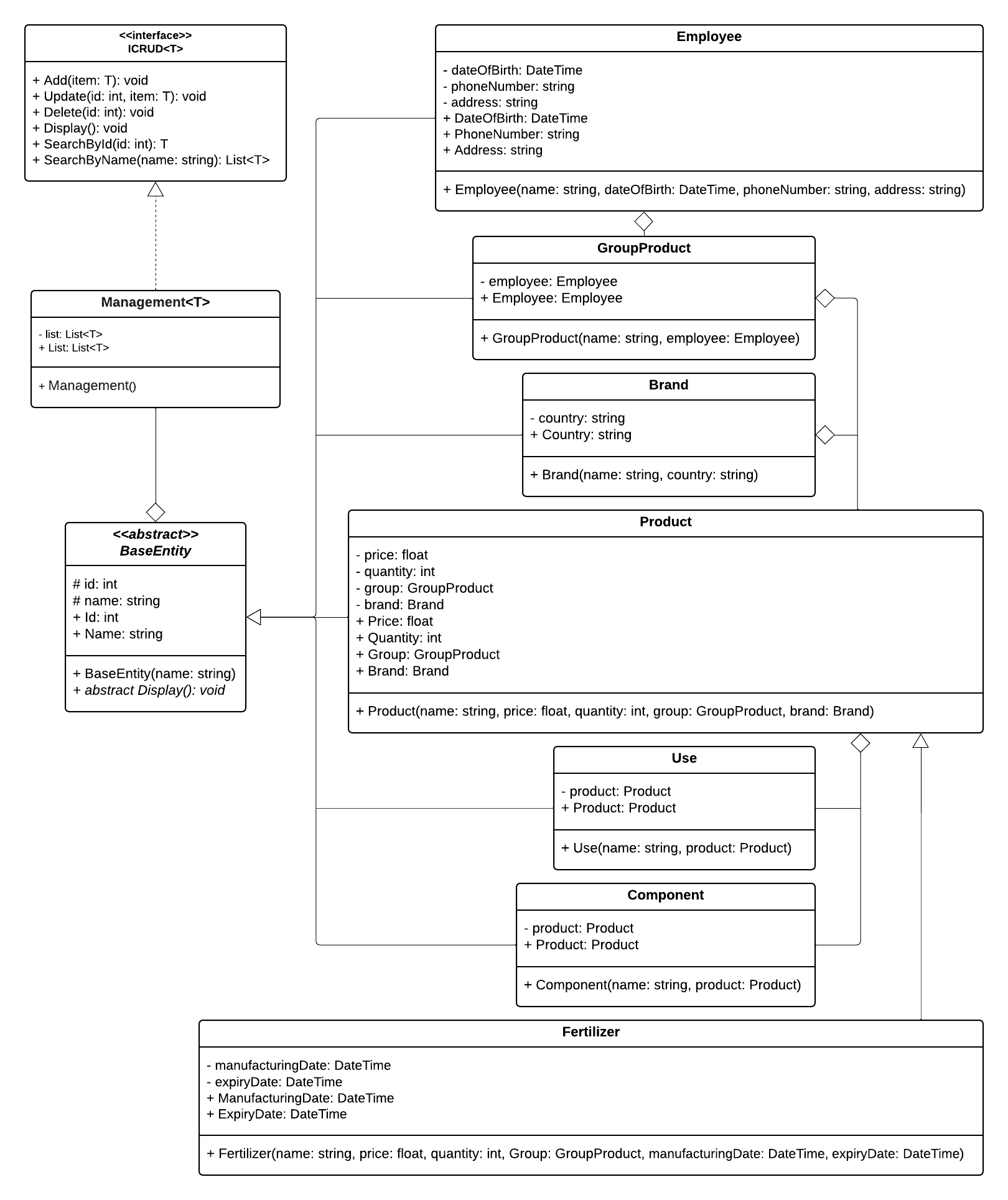
Sau khi hoàn thiện việc cài đặt, tiến hành kiểm thử để đảm bảo ứng dụng hoạt động đúng theo yêu cầu:

* Thử nghiệm tất cả các thao tác: xem, thêm, sửa, xóa, tìm kiếm dữ liệu.
* Đánh giá tốc độ xử lý dữ liệu khi danh sách lưu trữ có nhiều phần tử.
* Đánh giá mức độ dễ sử dụng của giao diện dòng lệnh: hiển thị thông tin, hướng dẫn người dùng.
* Thực hiện các chỉnh sửa để tối ưu mã nguồn và cải thiện giao diện.
* Đánh giá ứng dụng đáp ứng đầy đủ yêu cầu và sẵn sàng triển khai.

## 3.2. Mô hình của ứng dụng

### 3.2.1. Sơ đồ lớp (Class Diagram)

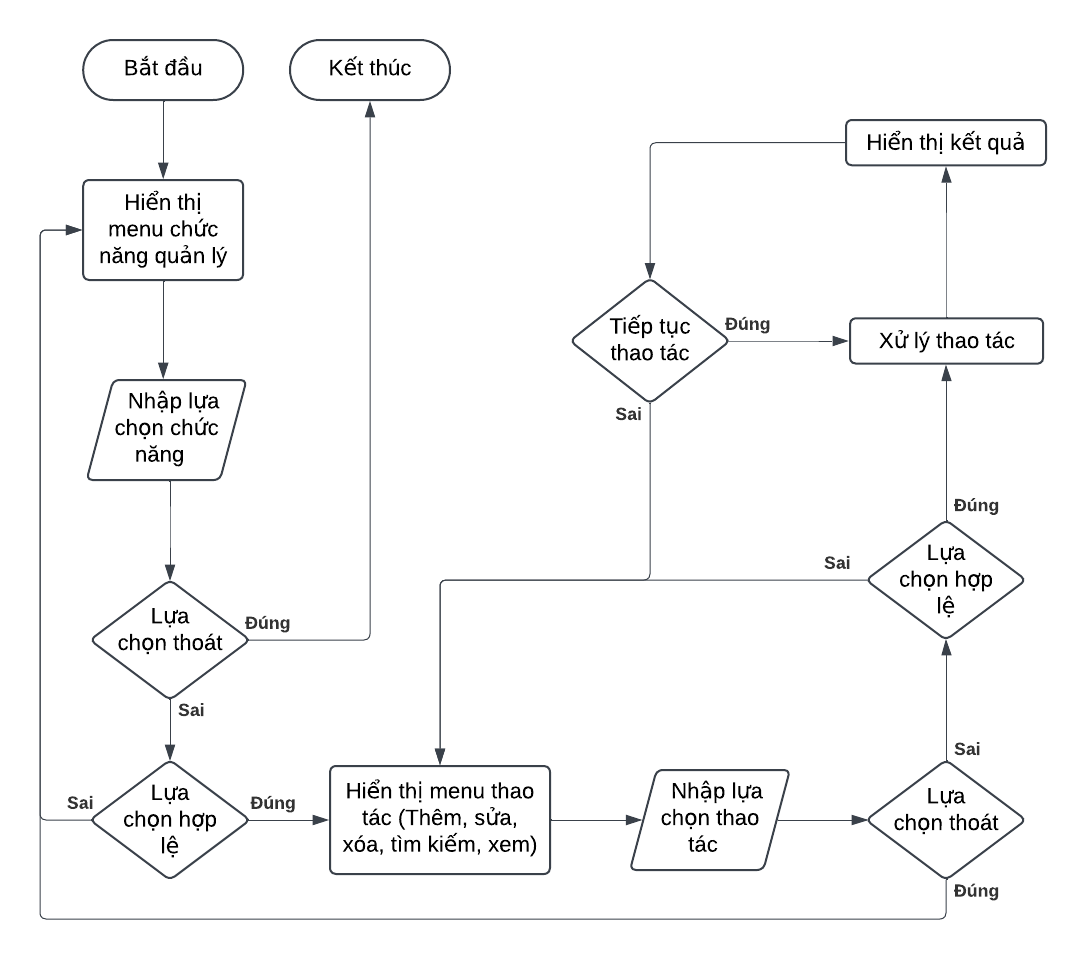
Sơ đồ lớp sau thể hiện cấu trúc của các thực thể và mối quan hệ giữa chúng.



Hình 3.1. Hình sơ đồ lớp của ứng dụng

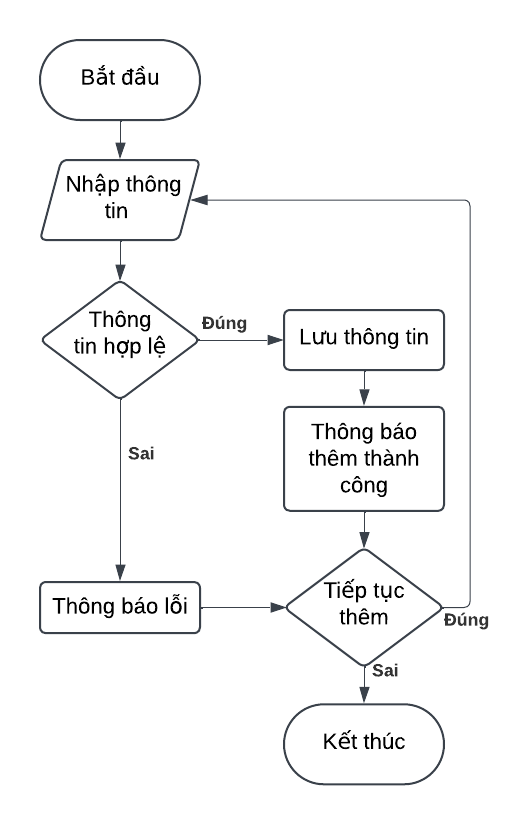
### 3.2.2. Lưu đồ xử lý (Flow Chart)

Lưu đồ xử lý đã xây dựng thể hiện một cách trực quan quy trình hoạt động của ứng dụng.

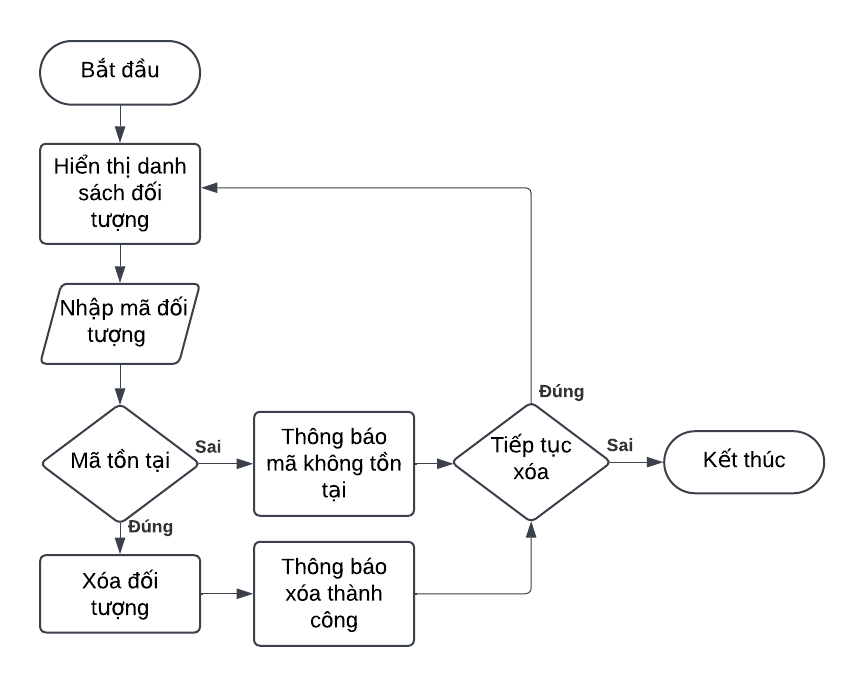


Hình 3.2. Hình lưu đồ xử lý tổng quát của ứng dụng

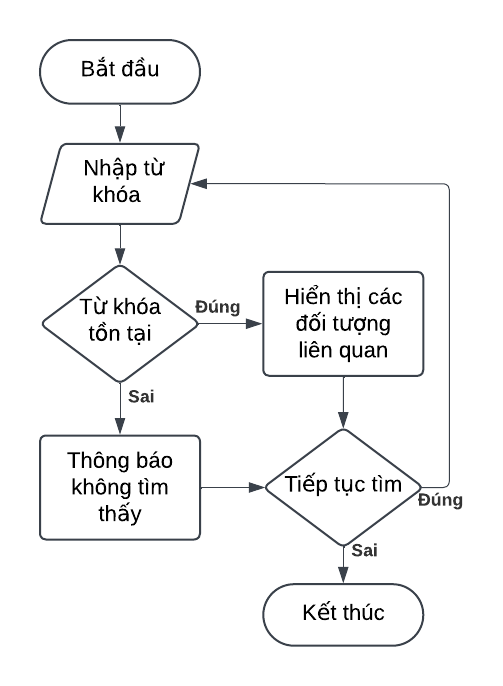
Để hiểu chi tiết hơn về ứng dụng, dưới đây là các lưu đồ của các thao tác thêm, sửa xóa và tìm kiếm.



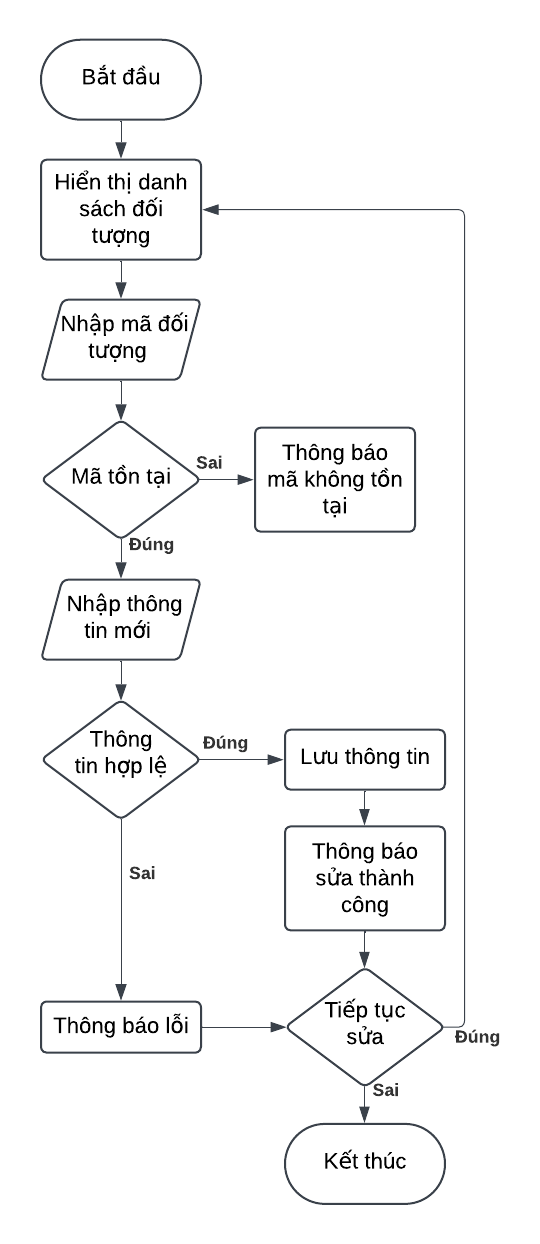
Hình 3.3. Hình lưu đồ xử lý thao tác thêm dữ liệu



Hình 3.4. Hình lưu đồ xử lý thao tác xóa dữ liệu



Hình 3.5. Hình lưu đồ xử lý thao tác tìm kiếm dữ liệu



Hình 3.6. Hình lưu đồ xử lý thao tác sửa dữ liệu

# CHƯƠNG 4. KẾT QUẢ NGHIÊN CỨU

# CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

# DANH MỤC TÀI LIỆU THAM KHẢO

1. Nguyễn Nhứt Lam (2014). Lập trình hướng đối tượng

2. Phạm Minh Đương (2014). Lập trình ứng dụng trên Windows

3. Phạm Văn Việt, Trương Lập Vĩ. Ngôn ngữ lập trình C#.

4. <https://learn.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide>

5. <https://learn.microsoft.com/vi-vn/dotnet/framework/get-started>

6. <https://www.w3schools.com/cs/index.php>

7. <https://www.geeksforgeeks.org/console-application>

8. <https://learn.microsoft.com/en-us/dotnet/csharp/fundamentals/tutorials/oop>

9. <https://learn.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/overview>

10. <https://learn.microsoft.com/vi-vn/dotnet/api/?view=netframework-4.8>

11. <https://dotnet.microsoft.com/en-us/languages/csharp>

# PHỤ LỤC