# MỤC LỤC

[MỤC LỤC 1](#_Toc155286723)

[MỞ ĐẦU 4](#_Toc155286724)

[CHƯƠNG 1 TỔNG QUAN VỀ QUY TRÌNH PHÁT TRIỂN PHẦN MỀM THEO HƯỚNG DEVOPS 5](#_Toc155286725)

[1. Giới thiệu về DevOps 5](#_Toc155286726)

[1.1. Khái niệm về DepOps 5](#_Toc155286727)

[1.2. Các nguyên tắc chính 5](#_Toc155286728)

[1.3. Lợi ích của DevOps 9](#_Toc155286729)

[1.4. Các công cụ DevOps 9](#_Toc155286730)

[1.5. Tương lai của DevOps 10](#_Toc155286731)

[2. Giới thiệu về ngôn ngữ lập trình C# 10](#_Toc155286732)

[2.1. Khái niệm về C# 10](#_Toc155286733)

[2.2. Các đặc trưng của ngôn ngữ C# 11](#_Toc155286734)

[2.3. Ưu nhược điểm của C# 12](#_Toc155286735)

[3. Giới thiệu về SQL Server 12](#_Toc155286736)

[3.1. Khái niệm SQL Server 12](#_Toc155286737)

[3.2. Cấu trúc SQL Server 13](#_Toc155286738)

[3.3. Các dịch vụ và công cụ SQL Server cung cấp 14](#_Toc155286739)

[3.4. Ưu điểm của SQL Server 14](#_Toc155286740)

[3.5. Một số phiên bản hệ quản trị cơ sở dữ liệu quan hệ - SQL server 14](#_Toc155286741)

[4. Giới thiệu về Github 15](#_Toc155286742)

[4.1. Khái niệm Github 15](#_Toc155286743)

[4.2. Tính năng của GitHub 16](#_Toc155286744)

[4.3. Lợi ích của GitHub 19](#_Toc155286745)

[4.4. Quy trình làm việc trên GitHub 19](#_Toc155286746)

[CHƯƠNG 2 PHÂN TÍCH YÊU CẦU 22](#_Toc155286747)

[1. Quy trình tổng quát thực hiện nhiệm vụ khoa học 22](#_Toc155286748)

[2. Yêu cầu phần mềm 24](#_Toc155286749)

[3. Đối tượng người dùng 27](#_Toc155286750)

[CHƯƠNG 3 THIẾT KẾ HỆ THỐNG 29](#_Toc155286751)

[1. Mô hình tổng thể phần mềm 29](#_Toc155286752)

[2. THIẾT KẾ LOGIC CSDL 30](#_Toc155286753)

[2.1. Các bảng của CSDL 30](#_Toc155286754)

[2.2. Bảng tổng quát 31](#_Toc155286755)

[2.3. Bảng LoaiDeTai 32](#_Toc155286756)

[2.4. Bảng CapBac 32](#_Toc155286757)

[2.5. Bảng ChucVu 33](#_Toc155286758)

[2.6. Bảng DonVi 33](#_Toc155286759)

[2.7. Bảng TacGia 33](#_Toc155286760)

[2.8. Bảng DeTaiNghienCuu 34](#_Toc155286761)

[2.9. Bảng TacGia-DeTai 34](#_Toc155286762)

[2.10. Bảng Vaitro 34](#_Toc155286763)

[2.11. Bảng TroLy 35](#_Toc155286764)

[2.12. Bảng MucDoHoanThanh 35](#_Toc155286765)

[2.13. Bảng TaiKhoaDangKy 35](#_Toc155286766)

[2.14. Bảng TaiKhoanDuyet 35](#_Toc155286767)

[2.15. Bảng PhanQuyen 35](#_Toc155286768)

[CHƯƠNG 4 KẾT QUẢ TRIỂN KHAI PHẦN MỀM 36](#_Toc155286769)

[1. Kết quả đạt được 36](#_Toc155286770)

[2. Một số khó khăn 36](#_Toc155286771)

# MỞ ĐẦU

Nghiên cứu khoa học (NCKH) ở trong Quân đội nói chung, tại các quân binh chủng nói riêng là một trong những nhiệm vụ cơ bản trong quá trình phát triển và hiện đại hóa Quân đội. Nó không chỉ góp phần nâng cao chất lượng, uy tín của các quân binh chủng, mà còn là quá trình không thể thiếu để huấn luyện và rèn luyện cho đội ngũ các cán bộ, chiến sĩ, nhân viên và lớp kế cận trong việc thực hiện sứ mệnh truyền bá, áp dụng tri thức khoa học, đáp ứng yêu cầu phát triển và hiện đại hóa Quân đội. Theo đó, quản lý NCKH tại các quân binh chủng gồm nhiều nội dung. Tuy nhiên, việc quản lý cơ sở dữ liệu các nhiệm vụ NCKH là một việc định kỳ hàng năm, tốn nhiều thời gian và có thể chi phối cũng như kiểm soát từ khi bắt đầu đến khi kết thúc quá trình nghiên cứu này. Để nâng cao hiệu quả NCKH, đặc biệt là các nhiệm vụ NCKH cấp cơ sở tại các quân binh chủng áp dụng được vào thực tiễn, cần phải đưa ra các giải pháp theo hướng: Số hóa việc quản lý trong từng quá trình cụ thể; Phải nâng cao vai trò, trách nhiệm của chủ thể quản lý và các chủ thể có liên quan cũng như phát triển mối quan hệ tương tác của các chủ thể khi thực hiện chức năng quản lý của mình trong toàn bộ quá trình quản lý.

Hiện nay, công tác nghiên cứu khoa học ở Binh chủng Thông tin liên lạc đang được quản lý bằng các văn bản giấy. Việc này gây nhiều bất tiện trong công tác quản lý, lưu trữ và tra cứu thông tin khi cần thiết vì khối lượng lưu trữ lớn, nhân lực thực hiện nhiệm vụ chưa được biên chế đầy đủ. Vì vậy, nhóm đề tài đã nghiên cứu và thiết kế xây dựng hệ thống cơ sở dữ liệu khoa học và môi trường trong Binh chủng Thông tin liên lạc nhằm số hóa công tác quản lý, lưu trữ và tra cứu thông tin về nhiệm vụ khoa học thực hiện trong Binh chủng. Phần mềm có chức năng quản lý các nhiệm vụ khoa học đã có quyết định giao nhiệm vụ và quyết định thành lập các nhóm nghiên cứu do Thủ trưởng Binh chủng và các đơn vị trực thuộc ký quyết định ban hành.

Quá trình thực hiện do chưa có nhiều kinh nghiệm thực tiễn nên khó tránh khỏi sai sót. Kính mong thầy cô tạo điều kiện và giúp đỡ để nhóm hoàn thành nhiệm vụ môn học đạt kết quả tốt nhất. Nhóm đề tài xin chân thành cảm ơn!

# CHƯƠNG 1 TỔNG QUAN VỀ QUY TRÌNH PHÁT TRIỂN PHẦN MỀM THEO HƯỚNG DEVOPS

Để nghiên cứu hệ thống cơ sở dữ liệu khoa học và môi trường trong Binh chủng Thông tin liên lạc, nhóm đề tài đã thu thập, phân tích và định nghĩa các yêu cầu và các tính năng của dự án; với mục đích đề cập đến các tính năng mà khách hàng mong muốn, giúp khách hàng hiểu rõ hơn về phần mềm mình sắp được sử dụng. Chi tiết của phần mềm Hệ thống cơ sở dữ liệu khoa học và môi trường trong Binh chủng thông tin liên lạc sẽ đáp ứng được các yêu cầu được mô tả chi tiết trong các tình huống sử dụng và các đặc tả phi chức năng. Phần mềm sử dụng ngôn ngữ lập trình C#, phần mềm Visual Studio 2022, cơ sở dữ liệu SQL Server.

# Giới thiệu về DevOps

## Khái niệm về DepOps

DevOps là một phương pháp làm việc kết hợp giữa các kỹ sư phát triển phần mềm (Development - Dev) và các kỹ sư vận hành (Operations - Ops). Mục đích của DevOps là giảm thiểu thời gian và chi phí cần thiết để phát triển, triển khai và quản lý phần mềm.

## Các nguyên tắc chính

1. Tự động hóa: DevOps sử dụng tự động hóa để giảm thiểu các tác vụ thủ công và lỗi. Tự động hóa có thể được sử dụng cho các tác vụ như build, kiểm tra và triển khai.
2. Sự hợp tác: DevOps khuyến khích sự hợp tác giữa các nhóm phát triển (Dev) và vận hành (Ops). Sự hợp tác giúp đảm bảo rằng các thay đổi được thực hiện một cách hiệu quả và có trách nhiệm.
3. Phát triển liên tục (CI) và triển khai liên tục (CD): DevOps sử dụng CI/CD để liên tục triển khai các thay đổi nhỏ vào môi trường sản xuất. CI/CD giúp giảm thiểu rủi ro và cải thiện chất lượng của các bản phát hành.

* **Continuous Intergration (CI - Tích hợp liên tục)**

CI là phương pháp đòi hỏi các developer phải thường xuyên merge code thay đổi. Với cách làm việc truyền thống, các developer sẽ làm việc riêng biệt với nhau và sau một khoảng thời gian nhất định chẳng hạn như 1 tuần họ sẽ tiến hành merge code. Tuy nhiên, với CI thì developer phải merge code của họ mỗi ngày và sẽ chạy auto test để detect những vấn đề khi merge code. CI cũng được tự động hóa, và thông thường được hỗ trợ bở một CI Server. Khi developer commit source code thay đổi của họ lên, CI Server sẽ thấy sự thay đổi này và bắt đầu thực hiện build, test source code thay đổi một cách tự động. Quá trình này sẽ được thực hiện nhiều lần trong 1 ngày, và nếu CI server phát hiện có vấn đề xảy ra nó sẽ ngay lập tức hiển thị thông báo cho developer.

Trường hợp có một người khác đưa code của họ lên và xảy ra lỗi trong quá trình build hệ thống sẽ thông báo lỗi cho người đó tiến hành fix, đồng thời sẽ rollback lại để không làm ảnh hưởng đến những người khác.

Như vậy, CI sẽ giúp phát hiện ra bug sớm, thông báo cho developer. Developer có thể fix ngay lập tức hoặc rollback để không làm ảnh hưởng đến người khác. Tiếp theo, áp dụng CI sẽ giúp tránh việc phải merge một lượng code lớn khi release. Thay vào đó code sẽ được merge tự động mỗi ngày. Đồng thời, nhờ vào việc code được merge mỗi ngày nên chúng ta có thể release thường xuyên chứ không cần chờ đến cuối giai đoạn khi tất cả mọi thứ đã xong xuôi mới có thể release được. Khi code được build liên tục nó cũng tạo ra Continuos Testing (Test liên tục), QA có thể test ngay lập tức những chỉnh sửa đã được đưa lên mà không cần chờ đến khi mọi thứ hoàn thành. CI tạo ra một thói quen tốt cho develop, việc thường xuyên commit sẽ làm developer viết ra những đoạn code đơn giản, đúng chuẩn ko rườm rà.

* **Continuous Delivery và Continuous Deployment**

Continuous Delivery (CD) là method đảm bảo code có thể được deploy bất cứ lúc nào. Thay vì phải quyết định có nên deploy code hay không thì team phải build, merge, test… để đảm bảo code luôn ở trạng thái có thể deploy.

Continuous Delivery thường được viết tắt là CD và gây nhầm lẫn giữa Continuous Delivery và Continuous Deployment.

Continuous Deployment là một practice đảm bảo code thay đổi (có size nhỏ) được deploy liên tục lên product.

Continous Delivery đảm bảo cho code có thể deploy bất cứ lúc nào. Trong khi Continuous Deployment là deploy thực tế lên product nhiều lần trong ngày.

Trên thực tế Continuous Delivery và Continuous Deployment được thực hiện theo flow sau:

* Mỗi version code đưa lên đều trải qua một quy trình giống nhau, bao gồm: build automation, test automation... Và nếu không có vấn đề gì xảy ra nghĩa là code đó có thể deploy.
* Sau khi quyết định deploy thì sẽ thực hiện deploy tự động (deploy automation).
* Nếu việc deploy gặp lỗi, hệ thống sẽ tự động rollback.
* Developer sẽ nhanh chóng fix lỗi và redeploy version fix lỗi.

Với cách làm như vậy Continuous Delivery và Continuous Deployment sẽ đem lại những lợi ích như dưới đây:

* Nhanh chóng đưa sản phẩm đến tay khách hàng
* Giảm thiếu các vấn đề xảy ra khi deploy
* Giảm thiếu risk: lượng deploy trong 1 lần càng nhiều, risk càng cao. Việc chia nhỏ lượng deploy sẽ giảm thiểu risk.
* Rollback lập tức khi xảy ra lỗi
* Giúp developer không còn lo lắng khi deploy khi đã có chức năng roll back automation.
* **Infrastructure as Code (IaC)**

Infrastructure as Code là method giúp quản lý, xây dựng cơ sở hạ tầng (Infra) cho sản phẩm của bạn. Với Infrastructure as Code, thay vì thực hiện thủ công theo manual. Bạn có thể viết code và cho chạy tự động.

Các đối tượng xây dựng trong Infra bao gồm: Server, Instance, môi trường, Container...

Lí do nên sử dụng IaC

* Đầu tiên, IaC đảm bảo việc tạo và quản lý resource đúng tiêu chuẩn. Tất cả mọi việc được thực hiện automation, ko có những sai sót nếu thực hiện bằng tay.
* Tính linh hoạt: IaC cho phép thực hiện cùng một thay đổi giống nhau trên nhiều host, và có thể sử dụng lại trong tương lai.
* Tính co dãn (Scalability): khi cần thêm instance bạn chỉ cần cho chạy lại config giống với instance có sẵn và một instance mới được tạo ra chỉ trong vài phút hoặc vài giây.
* Self-documenting: với IaC để xem những config của hệ thống chỉ cần xem trong source control, bạn không cần phải log lại hoặc tạo 1 tài liệu bất kỳ nào để lưu giữ thông tin config. Chính bản thân code được dùng để build infra chính là document.

## Lợi ích của DevOps

DevOps mang lại một số lợi ích cho các tổ chức, bao gồm:

1. Tăng tốc độ phát triển phần mềm: DevOps giúp các tổ chức phát triển và triển khai phần mềm nhanh hơn. Điều này là do DevOps sử dụng tự động hóa để giảm thiểu các tác vụ thủ công và khuyến khích sự hợp tác giữa các nhóm Dev và Ops.
2. Giảm thiểu lỗi: DevOps giúp giảm thiểu lỗi bằng cách tự động hóa các tác vụ và khuyến khích sự hợp tác giữa các nhóm Dev và Ops. Tự động hóa giúp giảm thiểu lỗi do con người gây ra, trong khi sự hợp tác giúp đảm bảo rằng các thay đổi được kiểm tra kỹ lưỡng trước khi được triển khai.
3. Tăng cường khả năng đáp ứng: DevOps giúp các tổ chức đáp ứng nhanh hơn với các thay đổi yêu cầu của khách hàng. Điều này là do DevOps sử dụng CI/CD để liên tục triển khai các thay đổi nhỏ vào môi trường sản xuất.

DevOps là một phương pháp làm việc phức tạp, nhưng có thể mang lại lợi ích đáng kể cho các tổ chức. Các tổ chức có thể cân nhắc áp dụng DevOps nếu họ muốn tăng tốc độ phát triển phần mềm, giảm thiểu lỗi và tăng cường khả năng đáp ứng.

## Các công cụ DevOps

Có một số công cụ DevOps có sẵn để giúp các tổ chức áp dụng phương pháp này. Một số công cụ DevOps phổ biến bao gồm:

1. Hệ thống quản lý phiên bản (VCS): VCS được sử dụng để lưu trữ và quản lý mã nguồn. Một số VCS phổ biến bao gồm Git, GitHub và Bitbucket.
2. Máy chủ build: Máy chủ build được sử dụng để tự động hóa quá trình build mã nguồn. Một số máy chủ build phổ biến bao gồm Jenkins, Travis CI và CircleCI.
3. Máy chủ kiểm tra: Máy chủ kiểm tra được sử dụng để tự động hóa quá trình kiểm tra mã nguồn. Một số máy chủ kiểm tra phổ biến bao gồm Selenium, JUnit và Mocha.
4. Máy chủ triển khai: Máy chủ triển khai được sử dụng để tự động hóa quá trình triển khai mã nguồn. Một số máy chủ triển khai phổ biến bao gồm Ansible, Puppet và Chef.

## Tương lai của DevOps

DevOps là một lĩnh vực đang phát triển nhanh chóng. Các tổ chức ngày càng áp dụng DevOps để tăng tốc độ phát triển, giảm thiểu lỗi và tăng cường khả năng đáp ứng.

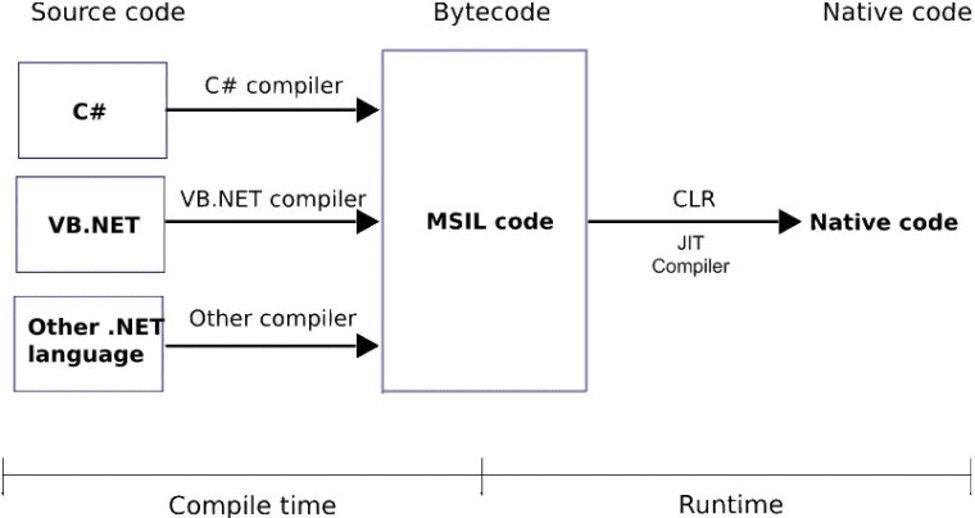
Tương lai của DevOps được dự đoán là tiếp tục phát triển mạnh mẽ. Các công nghệ mới, chẳng hạn như trí tuệ nhân tạo (AI) và học máy (ML), sẽ được tích hợp vào DevOps để tạo ra các phương pháp làm việc mới hiệu quả hơn.

# Giới thiệu về ngôn ngữ lập trình C#

## Khái niệm về C#

C# ra đời năm 2000, được thiết kế chủ yếu bởi **Anders Hejlsberg** – kiến trúc sư phần mềm nổi tiếng với các sản phẩm Turbo Pascal, Delphi… C# (hay C sharp) là sự kết hợp của C++ và Java. C# khá giống C++ và Java cả về diện mạo, cú pháp, biểu thức, toán tử… Vì vậy, C# được miêu tả là ngôn ngữ có sự cân bằng giữa C++, Visual Basic, Delphi và Java. C# là ngôn ngữ lập trình hiện đại, hướng đối tượng, hỗ trợ đầy đủ tính chất của lập trình hướng đối tượng như tính trừu tượng, tính đóng gói, kế thừa, đa hình.

Trong các ứng dụng Windows truyền thống, mã nguồn chương trình được biên dịch trực tiếp thành mã thực thi của hệ điều hành. Trong các ứng dụng sử dụng .NET Framework, mã nguồn chương trình (C#, VB.NET) được biên dịch thành mã ngôn ngữ trung gian MSIL (Microsoft intermediate language). Sau đó mã này được biên dịch bởi Common Language Runtime (CLR) để trở thành mã thực thi của hệ điều hành.



C# với sự hỗ trợ mạnh mẽ của .NET Framework giúp cho việc tạo một ứng dụng Windows Forms hay WPF (Windows Presentation Foundation), phát triển game, ứng dụng Web, ứng dụng Mobile trở nên rất dễ dàng.

## Các đặc trưng của ngôn ngữ C#

- C# là ngôn ngữ đơn giản

C# loại bỏ một vài sự phức tạp và rối rắm của những ngôn ngữ như Java và c++, bao gồm việc loại bỏ những macro, những template, đa kế thừa, và lớp cơ sở ảo (virtual base class).

Ngôn ngữ C# đơn giản vì nó dựa trên nền tảng C và C++. Nếu chúng ta thân thiện với C và C++ hoặc thậm chí là Java, chúng ta sẽ thấy C# khá giống về diện mạo, cú pháp, biểu thức, toán tử và những chức năng khác được lấy trực tiếp từ ngôn ngữ C và C++, nhưng nó đã được cải tiến để làm cho ngôn ngữ đơn giản hơn.

- C# là ngôn ngữ hiện đại

Những đặc tính như là xử lý ngoại lệ, thu gom bộ nhớ tự động, những kiểu dữ liệu mở rộng, và bảo mật mã nguồn là những đặc tính được mong đợi trong một ngôn ngữ hiện đại. C# chứa tất cả những đặc tính trên.

- C# là một ngôn ngữ lập trình thuần hướng đối tượng

Lập trình hướng đối tượng (OOP: Object-oriented programming) là một phương pháp lập trình có 4 tính chất. Đó là tính trừu tượng (abstraction), tính đóng gói (encapsulation), tính đa hình (polymorphism) và tính kế thừa (inheritance). C# hỗ trợ cho chúng ta tất cả những đặc tính trên.

- C# là một ngôn ngữ ít từ khóa

C# là ngôn ngữ sử dụng giới hạn những từ khóa (khoảng 80 từ khóa). Phần lớn các từ khóa được sử dụng để mô tả thông tin. Dù sử dụng ít từ khóa nhưng C# có thể được sử dụng để làm bất cứ nhiệm vụ nào.

## Ưu nhược điểm của C#

- Ưu điểm:

Gần gũi với các ngôn ngữ lập trình thông dụng (C++, Java, Pascal).

Xây dựng dựa trên nền tảng của các ngôn ngữ lập trình mạnh nên thừa hưởng những ưu điểm của những ngôn ngữ đó.

Cải tiến các khuyết điểm của C/C++ như con trỏ, các hiệu ứng phụ, . . .

Dễ tiếp cận, dễ phát triển.

Được sự hỗ trợ mạnh mẽ của .NET Framework.

- Nhược điểm

C# chỉ chạy trên nền Windows và có cài .NET Framework

Thao tác đối với phần cứng yếu hơn so với ngôn ngữ khác. Hầu hết phải dựa vào Windows.

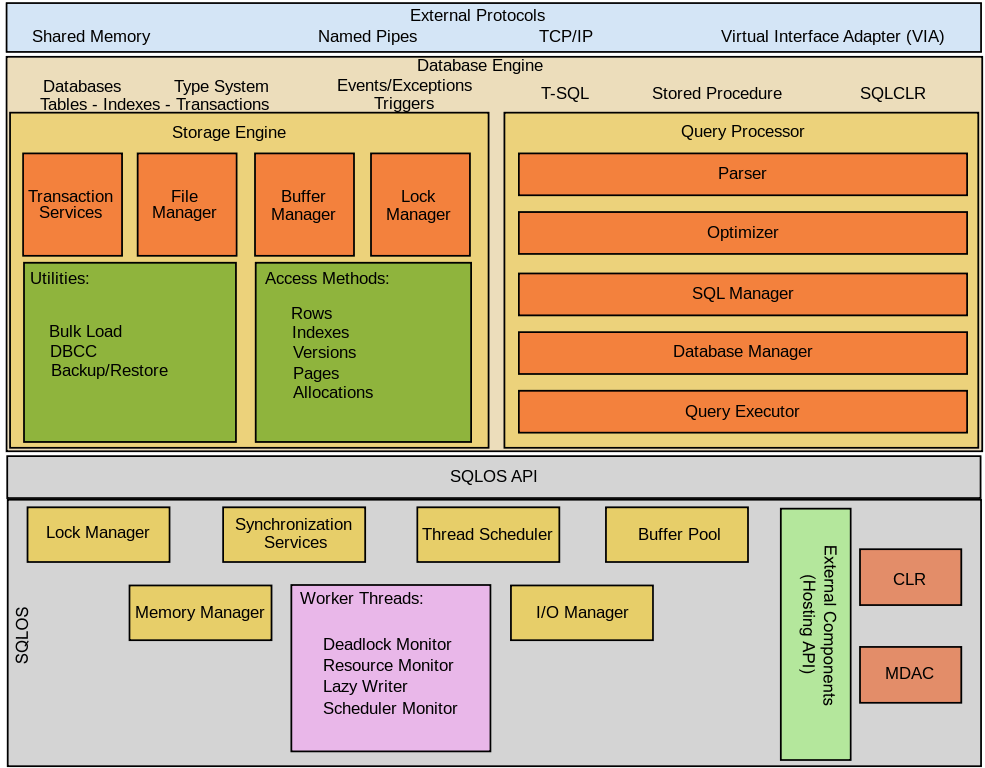
# Giới thiệu về SQL Server

## Khái niệm SQL Server

 hay Microsoft SQL Server là phần mềm ứng dụng cho hệ thống quản trị cơ sở dữ liệu quan hệ (Relational Database Management System – RDBMS) được phát triển bởi Microsoft vào năm 1988. Nó được sử dụng để tạo, duy trì, quản lý và triển khai hệ thống RDBMS.

Phần mềm SQL Server được sử dụng khá rộng rãi vì nó được tối ưu để có thể chạy trên môi trường cơ sở dữ liệu rất lớn lên đến Tera – Byte cùng lúc phục vụ cho hàng ngàn user. Bên cạnh đó, ứng dụng này cung cấp đa dạng kiểu lập trình SQL từ ANSI SQL (SQL truyền thống) đến SQL và cả T-SQL (Transaction-SQL) được sử dụng cho cơ sở dữ liệu quan hệ nâng cao.

## Cấu trúc SQL Server



SQL Server bao gồm 2 thành phần chính: Database Engine và SQLOS.

Thành phần cốt lõi của SQL Server là Database Engine. Nó bao gồm Relation Engine giúp xử lý các truy vấn và Storage Engine giúp quản lý các tệp cơ sở dữ liệu, các trang, chỉ mục…

Dưới Database Engine là hệ điều hành SQL Server hay SQLOS (SQL Server Operating System). SQLOS giúp quản lý bộ nhớ và I/O, lên lịch nhiệm vụ và khóa dữ liệu để tránh các xung đột xảy ra khi update.

## Các dịch vụ và công cụ SQL Server cung cấp

Microsoft cung cấp công cụ, dịch vụ quản lý dữ liệu (data management) và BI.

Với quản lý dữ liệu, SQL Server cung cấp các dịch vụ như: SQL Server Integration Services (SSIS), SQL Server Data Quality Services và SQL Server Master Data Services. Để phát triển cơ sở dữ liệu, ta có thể sử dụng công cụ SQL Server Data. Để quản lý, triển khai và giám sát cơ sở dữ liệu ta có SQL Server Management Studio (SSMS).

Với việc phân tích dữ liệu, SQL Server cung cấp SQL Server Analysis Services (SSAS), SQL Server Reporting Services (SSRS) và The Machine Learning Services (R Services).

## Ưu điểm của SQL Server

* Giao diện SQL Server rất dễ dàng để các Backend Developer tập trung nhiều vào việc chăm sóc dữ liệu hơn là việc nó hoạt động như thế nào.
* Tích hợp với giao diện người dùng: SQL Server được tích hợp với ứng dụng giao diện người dùng, thường là các ứng dụng web để cung cấp cơ chế thay đổi dữ liệu động.
* Vì là sản phẩm của Microsoft nên việc tích hợp framework .Net sẽ dễ dàng hơn do chúng có cùng một tổ chức.
* So với các phương tiện lưu trữ dữ liệu khác như Excel, văn bản… cơ sở dữ liệu luôn được ưa chuộng hơn chủ yếu do khả năng lưu trữ lớn, tính bảo mật cao và sức mạnh xử lý dữ liệu.

## Một số phiên bản hệ quản trị cơ sở dữ liệu quan hệ - SQL server

* Enterprise (Bản Doanh nghiệp): Phiên bản này chứa gần như đầy đủ nhất các tinh hoa của SQL Server. Trong đó phải kể đến như nhân bộ máy cơ sở dữ liệu và các dịch vụ đi kèm cùng với các công cụ cho tạo và quản lý phân cụm SQL Server. Với phiên bản Enterprise, nó hướng đến các CSDL lớn tới 524 [petabytes](https://en.wikipedia.org/wiki/Petabyte) và hỗ trợ tới 640 bộ vi xử lý.
* Standard (Bản Tiêu chuẩn): Phiên bản này rất thích hợp cho các công ty vừa và nhỏ. Một phần vì giá thành rẻ hơn nhiều so với phiên bản Enterprise, một phần vì những công ty vừa và nhỏ cũng chưa có lượng dữ liệu lớn đến thế.
* Developer (Bản Developer): Phiên bản này có đầy đủ các tính năng của bản Enterprise nhưng nó được tuỳ biến để nhằm phục vụ cho việc phát triển và kiểm tra ứng dụng.
* Workgroup (Bản Workgroup): Phiên bản SQL Server này chỉ bao gồm chức năng lõi cơ sở dữ liệu nhưng không có các dịch vụ đi kèm. Đặc biệt hơn từ phiên bản SQL server 2012, phiên bản này đã bị Microsoft khai tử.
* Express (Bản Express): SQL Server phiên bản này được đánh giá là dễ sử dụng. Có thể nó hướng đến dùng cho việc quản trị cơ sở dữ liệu đơn giản. Microsoft tích hợp nó cùng với Microsoft Visual Studio, nên giúp dễ dàng để phát triển các ứng dụng dữ liệu. Đặc biệt hơn nữa đây là một phiên bản miễn phí. Bạn sẽ không lo bị giới hạn về số cơ ở dữ liệu hoặc người sử dụng. Tuy nhiên nhược điểm là nó chỉ dùng cho 1 CPU với 1 GB bộ nhớ và 10 GB file cơ sở dữ liệu. Nếu như bạn làm nhà phát triển ứng dụng mới, hoặc bạn thường xuyên xây dựng các ứng dụng nhỏ thì đây sẽ là một sự hỗ trợ của Microsoft dành cho bạn.

# Giới thiệu về Github

## Khái niệm Github

GitHub được thành lập vào năm 2008 bởi Chris Wanstrath, PJ Hyett và Tom Preston-Werner. Tính đến tháng 4 năm 2023, GitHub có hơn 90 triệu người sử dụng với hơn 200 triệu kho mã nguồn, làm cho nó trở thành máy chủ chứa mã nguồn lớn nhất trên thế giới.

GitHub là một hệ thống quản lý dự án cũng như các phiên bản Code. Nền tảng này được ví như mạng xã hội dành riêng cho các lập trình viên. Tại đó, Coder được quyền clone lại mã nguồn từ một Repository.

GitHub cũng đóng vai trò là một dịch vụ máy chủ Repository. Mục đích sử dụng để lưu trữ code, mỗi người có thể tạo tài khoản riêng và tạo ra kho chứa của mình. Nhờ thế, mỗi lần cần tái sử dụng, người dùng có thể nhanh chóng tìm thấy và vận dụng dễ dàng.

GitHub có đầy đủ tính năng của Git. Bên cạnh đó là các đặc điểm bổ sung về Social giúp các nhà phát triển tương tác với nhau một cách thuận lợi.

Người sử dụng có thể khai thác chức năng feeds, follower và network graph. Từ đó, mọi người học hỏi được kinh nghiệm từ lịch sử commit. Hiện nay, nền tảng này cung cấp hai hình thức là miễn phí và trả phí.

- Với phiên bản có phí, các doanh nghiệp công nghệ sử dụng để tăng khả năng quản lý đội nhóm. Đồng thời, Manager cũng phân quyền bảo mật dự án một cách dễ dàng khi hiểu GitHub là gì.

- Đối với phiên bản miễn phí, hầu hết người dùng đều sử dụng để lưu trữ source code.

## Tính năng của GitHub

1. Lưu trữ mã nguồn

GitHub cho phép người dùng lưu trữ mã nguồn của các dự án phần mềm. Mã nguồn được lưu trữ trên các máy chủ của GitHub và có thể được truy cập từ bất kỳ đâu trên thế giới.

Để lưu trữ mã nguồn trên GitHub, người dùng cần tạo một kho lưu trữ (repository). Kho lưu trữ là một thư mục chứa mã nguồn và các tài liệu liên quan đến dự án.

Sau khi tạo kho lưu trữ, người dùng có thể tải lên mã nguồn của dự án. Mã nguồn có thể được tải lên theo nhiều cách khác nhau, chẳng hạn như tải lên từ máy tính cục bộ, kéo và thả từ trình duyệt web hoặc sử dụng các công cụ tích hợp sẵn của GitHub.

1. Quản lý phiên bản

GitHub sử dụng hệ thống quản lý phiên bản Git để theo dõi các thay đổi đối với mã nguồn. Điều này giúp các nhà phát triển có thể theo dõi lịch sử phát triển của dự án và dễ dàng quay lại các phiên bản trước đó.

Git là một hệ thống quản lý phiên bản phân tán. Điều này có nghĩa là mã nguồn được lưu trữ trên các máy chủ của GitHub và trên máy tính của các nhà phát triển.

Khi một nhà phát triển thực hiện thay đổi đối với mã nguồn, họ sẽ tạo một "commit". Commit là một bản sao của mã nguồn tại một thời điểm cụ thể.

Commit được lưu trữ trong kho lưu trữ dưới dạng một chuỗi liên tục. Mỗi commit đều có một ID duy nhất.

Các nhà phát triển có thể sử dụng Git để theo dõi lịch sử phát triển của dự án bằng cách xem danh sách các commit.

1. Tương tác

GitHub cung cấp một số tính năng để giúp các nhà phát triển tương tác với nhau, chẳng hạn như các công cụ cộng tác, hệ thống theo dõi lỗi và hệ thống issue tracking.

1. Công cụ cộng tác

GitHub cung cấp một số công cụ để giúp các nhà phát triển cộng tác với nhau trên cùng một dự án. Các công cụ này bao gồm:

* Pull request: Pull request là một cách để đề xuất các thay đổi đối với mã nguồn của một dự án.
* Review: Review là một quá trình xem xét các thay đổi đối với mã nguồn trước khi chúng được chấp nhận.
* Merge: Merge là quá trình hợp nhất các thay đổi từ một pull request vào mã nguồn chính của dự án.

1. Hệ thống theo dõi lỗi

GitHub cung cấp một hệ thống theo dõi lỗi để giúp các nhà phát triển theo dõi và giải quyết các vấn đề trong dự án.

Hệ thống theo dõi lỗi bao gồm các tính năng như:

* Tạo lỗi: Tạo lỗi để ghi lại các vấn đề trong dự án.
* Theo dõi lỗi: Theo dõi trạng thái của các lỗi.
* Xem xét lỗi: Xem xét các lỗi và đề xuất giải pháp.

1. Hệ thống issue tracking

GitHub cung cấp một hệ thống issue tracking để giúp các nhà phát triển theo dõi các yêu cầu và yêu cầu thay đổi đối với dự án.

Hệ thống issue tracking bao gồm các tính năng như:

* Tạo issue: Tạo issue để ghi lại các yêu cầu và yêu cầu thay đổi đối với dự án.
* Theo dõi issue: Theo dõi trạng thái của các issue.
* Xem xét issue: Xem xét các issue và thảo luận về các giải pháp.

1. Một số tính năng khác

* Wiki: Wiki là một trang web được sử dụng để lưu trữ tài liệu liên quan đến dự án.
* Gists: Gists là một cách để chia sẻ mã nguồn nhỏ.
* Marketplace: Marketplace là một nơi để mua và bán các dịch vụ và sản phẩm liên quan đến GitHub.

## Lợi ích của GitHub

* Hỗ trợ quản lý source code

Khi khởi tạo một đoạn repo, toàn bộ source code sẽ được lưu trữ lên GitHub nên có thể dễ dàng xem và theo dõi quá trình hoạt động, biết được ai đã commit và commit những gì. Như vậy, GitHub sẽ giúp cho source code phát triển được nhiều nhánh hơn.

* Tracking thay đổi qua từng phiên bản

GitHub hữu ích cho các dự án có nhiều thành viên cùng thực hiện. Nó sẽ lưu lại các nội dung được thay đổi, vị trí file lưu và push lên Repository để người dùng tiện theo dõi.

* Tối ưu hiển thị tài liệu bằng Markdown

Markdown là định dạng text trên web, cho phép người dùng chỉnh sửa, thay đổi hiển thị của tài liệu, tạo danh sách và thêm hình ảnh. Markdown được tích hợp trên GitHub tại Git, Issue, Pull Request hoặc ở các file có đuôi .md và .markdown extension.

* Có thêm một kho tài nguyên chất lượng

Chức năng Explore trong GitHub sẽ giúp người dùng tìm kiếm được các dự án mã nguồn mở theo đúng technology pattern yêu cầu. Bên cạnh đó, server của GitHub còn có các Workflow Script tự động, hỗ trợ cho việc phản hồi sự kiện trên Repositories.

## Quy trình làm việc trên GitHub

Bước 1: Project manager (Nhóm trưởng) tạo Repository mới bao gồm tên, mô tả và loại dự án và first commit với file Readme.md để test Repository đã hoạt động chưa.

Bước 2: Add Collaborators.

Bước 3: Tạo branch dev và add protection rule cho các nhánh.

Bước 4: Tạo Issues để giao nhiệm vụ cụ thể cho từng thành viên theo nội dung nhóm đã họp, bàn bạc và thống nhất.

Bước 5: Các thành viên up code từ local lên remote.

* git checkout dev
* git checkout -b feature/<ten-nhanh-moi> dev
* git add <ten-file>
* git commit -m ‘#<Issues-ID> - <Issues-title>’
* git push --set-upstream origin feature/<ten-nhanh-moi>

Bước 5: Project manager tạo pull request

* Create new pull request 
* Lựa chọn nhánh gốc  và nhánh up code 
* Creat pull request và ghi rõ tiêu đề, mô tả sự thay đổi của code, thêm Reviewers và Assignees. Labels đặt documentation với tài liệu hoặc feature với file code. 
* Sau khi Git kiểm tra khả năng sáp nhập với nhánh dev không có lỗi thì tiến hành merge pull request và confirm merge.
* Xóa nhánh phụ được tạo trong khi up code và close issue.

Bước 6: Add tag và tạo release.

* git checkout -b release-1.0.0 dev
* git tag 'v1.0.0'
* git push –tags
* git push
* Xóa nhánh release như bước 5.

Bước 7: Cập nhật branch main và dev để phát triển các version tiếp theo của phần mềm. Quay lại bước 4.

* Git checkout dev
* Git merge origin/main

# CHƯƠNG 2 PHÂN TÍCH YÊU CẦU

# Quy trình tổng quát thực hiện nhiệm vụ khoa học

| **Bước** | **Tiến trình thực hiện** | **Đơn vị thực hiện** | **Đơn vị phối hợp** | **Thời gian thực hiện** | **Biểu mẫu (nếu có)** |
| --- | --- | --- | --- | --- | --- |
|  | Đề xuất (thông qua Ban Khoa học quân sự - KHQS) nhiệm vụ khoa học và công nghệ (KH&CN) cho cơ quan, dự kiến đơn vị thụ hưởng, sử dụng kết quả nghiên cứu | Cá nhân/ đơn vị thuộc Nhà trường | Các đơn vị liên quan; Ban Khoa học quân sự | Theo thông báo của cơ quan có thẩm quyền |  |
|  | Tổng hợp danh sách các nhiệm vụ KH&CN do cá nhân/ đơn vị đề xuất; gửi danh sách và thuyết minh sơ bộ cho Hội đồng Khoa học Nhà trường xem trước; đồng thời lưu trữ tại cơ quan quản lý nhiệm vụ KH&CN để theo dõi | Ban KHQS | Các cá nhân, đơn vị liên quan | Theo thông báo của cơ quan có thẩm quyền |  |
|  | Căn cứ vào danh mục nhiệm vụ KH&CN các cấp được phê duyệt, cá nhân/ đơn vị chuẩn bị các hồ sơ theo quy định để tham gia tuyển chọn hoặc xét giao trực tiếp nhiệm vụ KH&CN | Cá nhân/ đơn vị đăng ký tham gia tuyển chọn hoặc xét giao trực tiếp nhiệm vụ KH&CN | Các cá nhân, đơn vị liên quan | Trước khi nộp hồ sơ cho cơ quan quản lý ít nhất 15 ngày |  |
|  | Tổ chức họp Hội đồng Khoa học của Nhà trường, bảo vệ đề cương chi tiết trước Hội đồng Khoa học và thông qua các NVKH | Ban KHQS, Hội đồng khoa học Nhà trường, Các nhóm đề tài | Các cá nhân, đơn vị liên quan | Theo thông báo của cơ quan có thẩm quyền |  |
|  | Ban hành quyết định giao nhiệm vụ thực hiện NVKH cho các nhóm đề tài | Thủ trưởng Nhà trường, Ban KHQS | Các cá nhân, đơn vị liên quan | Theo thông báo của cơ quan có thẩm quyền |  |
|  | Theo dõi tiến độ thực hiện NVKH, giải quyết các vướng mắc trong quá trình thực hiện NVKH của các nhóm đề tài | Ban KHQS, Thủ trưởng các đầu mối có các nhóm đề tài thực hiện NVKH | Các cá nhân, đơn vị liên quan | Theo kế hoạch |  |
|  | Tổ chức họp Hội đồng nghiệm thu chính thức kết quả thực hiện nhiệm vụ KH&CN | Ban KHQS, Hội đồng khoa học Nhà trường, Các nhóm đề tài | Các cá nhân, đơn vị liên quan | Theo thông báo của cơ quan có thẩm quyền |  |
|  | Ban hành quyết định đưa vào sử dụng các NVKH đã nghiệm thu | Ban KHQS, Hội đồng khoa học Nhà trường, Các nhóm đề tài | Các cá nhân, đơn vị liên quan | Theo thông báo của cơ quan có thẩm quyền |  |
|  | Phối hợp thu thập thông tin về ứng dụng kết quả thực hiện nhiệm vụ KH&CN trong Nhà trường và gửi phản hồi về các nhóm đề tài để tiếp tục cập nhật, bổ sung | Ban KHQS, Các nhóm đề tài | Các cá nhân, đơn vị liên quan | Theo thông báo của cơ quan có thẩm quyền |  |

# Yêu cầu phần mềm

| **Mã yêu cầu** | **Tên yêu cầu** | **Mô tả** | **Ghi chú** |
| --- | --- | --- | --- |
| SRS01 | Phân quyền chức năng | Phân nhóm các quyền: Quyền quản trị; Quyền của thủ trưởng; Quyền của trợ lý; Quyền của người dùng tham khảo |  |
| SRS02 | Quản lý NVKH | Quản lý cập nhật nhiệm vụ (Thêm, sửa, xóa nhiệm vụ) với các trường dữ liệu:   * Tên nhiệm vụ, * Thuyết minh nhiệm vụ, * Quyết định phê duyệt hoặc giao nhiệm vụ, * Cơ quan chủ trì và cấp thực hiện * Đơn vị thực hiện, * Thời gian bắt đầu và hoàn thành, * Tiến độ nhiệm vụ * Kinh phí, * Các tác giả, * Hội đồng nghiệm thu, Biên bản, * Quyết định đưa vào sử dụng * Ghi chú |  |
| SRS03 | Quản lý lý lịch khoa học | Quản lý cập nhật các tác giả thực hiện nhiệm vụ (Thêm, sửa, xóa nhiệm vụ) với các trường dữ liệu:   * Tên tác giả * Ngày tháng năm sinh * Nhập ngũ * Cấp bậc * Chức vụ * Đơn vị * Học hàm * Học vị * Chức danh chuyên môn kỹ thuật * Quá trình công tác * Số lượng NVKH theo phân loại: Đề tài KH, Bài báo KH, Tài liệu biên soạn * Giải thưởng, bằng phát minh * Chức vụ tham gia ở các đơn vị khoa học khác * Số điện thoại * Email * Địa chỉ * Thông tin khác |  |
| SRS04 | Quản lý nhân sự | Quản lý cập nhật các thực hiện nhiệm vụ quản lý (Thêm, sửa, xóa nhiệm vụ) với các trường dữ liệu:   * Tên nhân sự * Ngày tháng năm sinh * Nhập ngũ * Cấp bậc * Chức vụ * Đơn vị * Học hàm * Học vị * Chức danh chuyên môn kỹ thuật * Quá trình công tác * Số điện thoại * Email * Địa chỉ * Thông tin khác |  |
| SRS05 | Tìm kiếm và tra cứu thông tin theo yêu cầu | Tìm kiếm các thông tin khi nhập từ khóa về:   * Tên đề tài * Tên tác giả * Năm thực hiện * Cơ quan chủ trì, cấp thực hiện * Đơn vị thực hiện * Kinh phí |  |
| SRS05 | Thống kê và in báo cáo | In ấn thống kê, báo cáo theo đúng các biểu mẫu do phía khách hàng cung cấp |  |

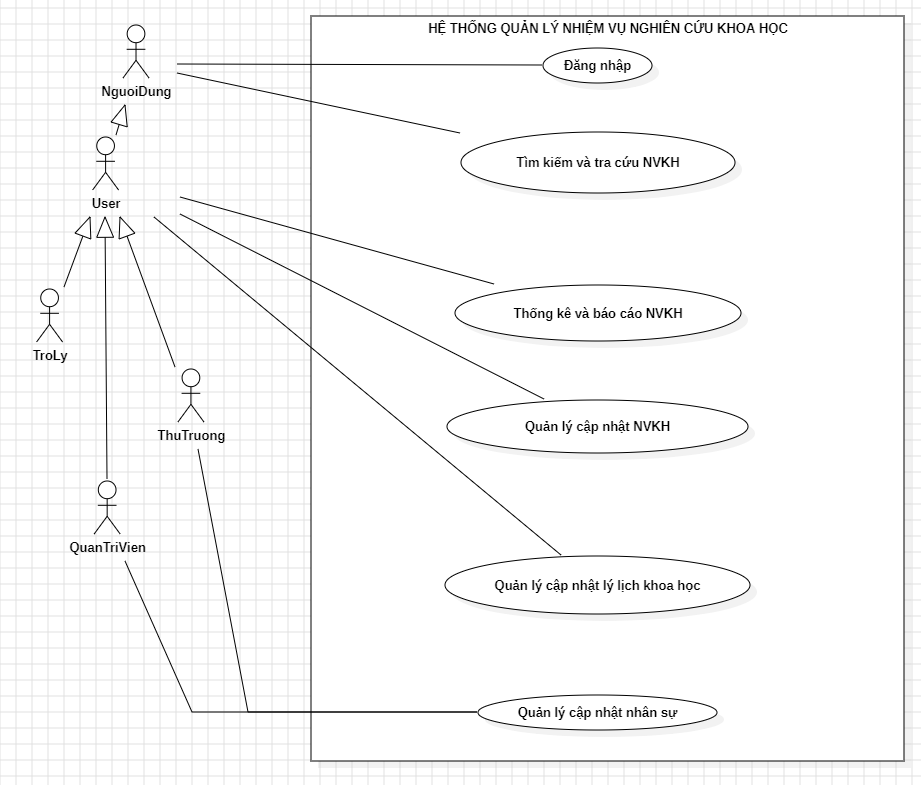
# Đối tượng người dùng

Hệ thống có các nhóm người dùng là:

* Nhóm 1 - Nhóm Quản trị: kiểm soát ở mức tổng thể toàn bộ hệ thống; Quản lý tất cả các tính năng của phần mềm; Thực hiện tính năng chỉnh sửa và cập nhật phần mềm
* Nhóm 2 – Nhóm thủ trưởng: kiểm soát ở mức cơ bản toàn bộ hệ thống; Quản lý hầu hết tính năng của phần mềm, trừ tính năng chỉnh sửa và cập nhật phần mềm
* Nhóm 3 – Trợ lý: kiểm soát ở mức thực hiện các tính năng về quy trình nghiệp vụ quản lý, báo cáo, tra cứu và tìm kiếm, kiểm soát người dùng.
* Nhóm 4 – Người dùng tham khảo: kiểm soát ở mức thực hiện chức năng tìm kiếm và tra cứu thông tin.

# CHƯƠNG 3 THIẾT KẾ HỆ THỐNG

# Mô hình tổng thể phần mềm

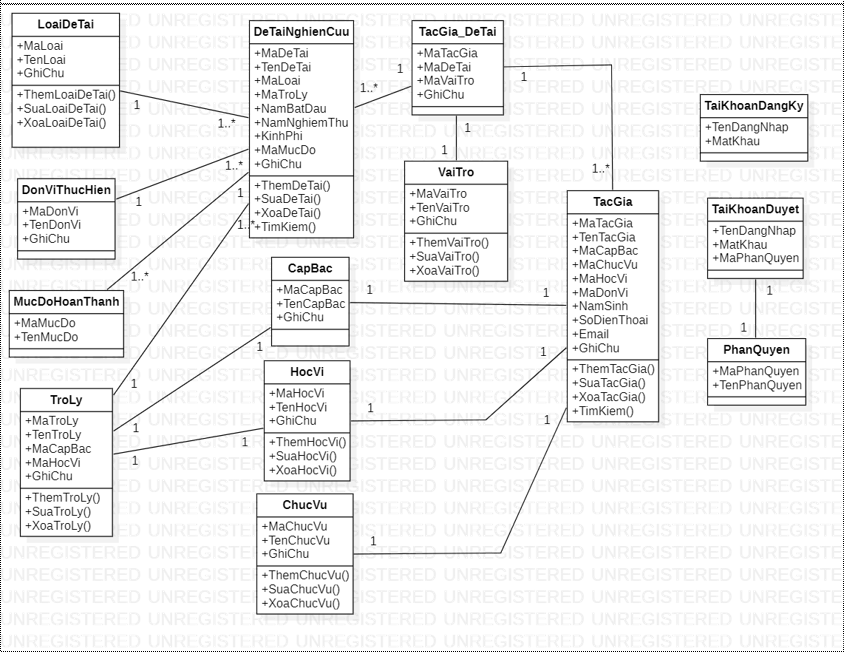


# THIẾT KẾ LOGIC CSDL

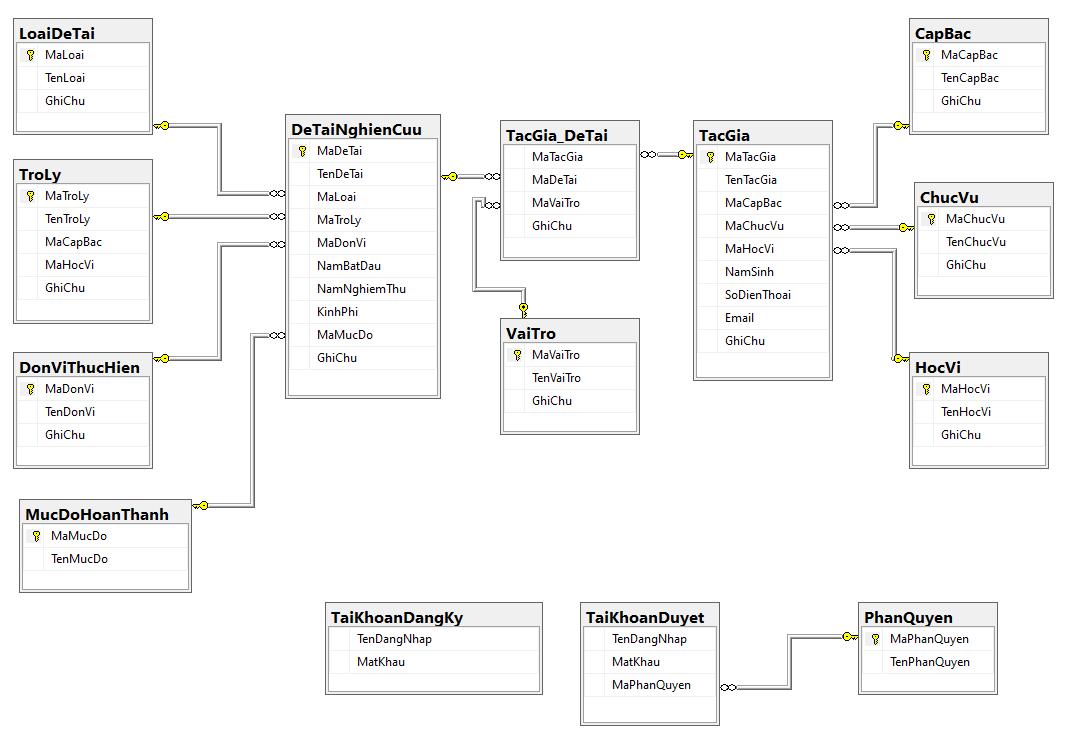
## Các bảng của CSDL

| **STT** | **Tên bảng** | **Alias** | **Mô tả** |
| --- | --- | --- | --- |
|  | Loại đề tài | LoaiDeTai | Là bảng lưu trữ các thông tin về loại đề tài: Cấp trường, Cấp Binh chủng, Cấp ngành |
|  | Cấp bậc | CapBac | Là bảng lưu trữ những thông tin về hệ thống cấp bậc trong Quân đội |
|  | Chức vụ | ChucVu | Là bảng lưu trữ những thông tin về hệ thống chức vụ trong Nhà trường |
|  | Học vị | HocVi | Là bảng lưu trữ các thông tin về học vị: Tiến sĩ, Thạc sĩ |
|  | Đơn vị | DonVi | Là bảng lưu trữ thông tin về các đơn vị trong Nhà trường |
|  | Tác giả | TacGia | Là bảng lưu trữ thông tin cá nhân về tác giả thực hiện đề tài |
|  | Đề tài nghiên cứu | DeTaiNghienCuu | Là bảng lưu trữ thông tin chi tiết về các nhiệm vụ nghiên cứu khoa học |
|  | Tác giả - Đề tài | TacGia\_DeTai | Là bảng lưu trữ thông về các mã liên kết của tác giả và đề tài |
|  | Nhiệm vụ | NhiemVu | Là bảng lưu trữ thông tin về nhiệm vụ của tác giả khi thực hiện đề tài: Chủ nhiệm, Tham gia, Thư ký |
|  | Trợ lý | TroLy | Là bảng lưu trữ thông tin về các trợ lý phụ trách quản lý đề tài |

## Bảng tổng quát



**Hình 5.1. Class Diagram**



**Hình 5.2. Database Diagram**

## Bảng LoaiDeTai

A screenshot of a data

Description automatically generated

## Bảng CapBac

A screenshot of a computer

Description automatically generated

## Bảng ChucVu

A screenshot of a data

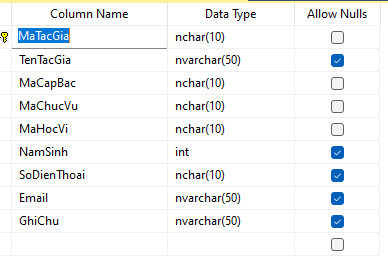
Description automatically generated

## Bảng DonVi

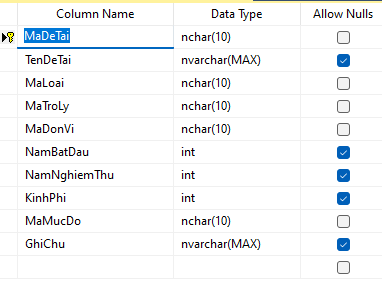
A screenshot of a data type

Description automatically generated

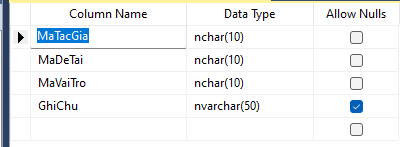
## Bảng TacGia



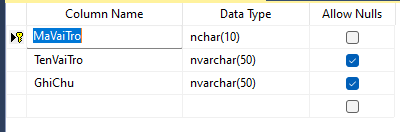
## Bảng DeTaiNghienCuu



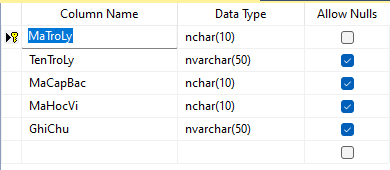
## Bảng TacGia-DeTai



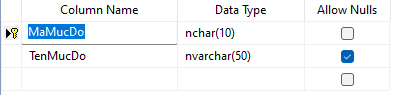
## Bảng Vaitro



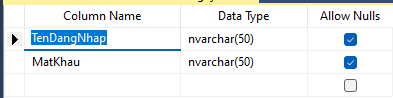
## Bảng TroLy



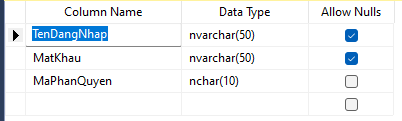
## Bảng MucDoHoanThanh



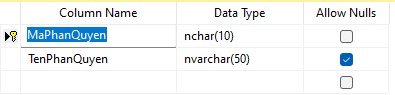
## Bảng TaiKhoaDangKy



## Bảng TaiKhoanDuyet



## Bảng PhanQuyen



# CHƯƠNG 4 KẾT QUẢ TRIỂN KHAI PHẦN MỀM

# Kết quả đạt được

***a) Về lý thuyết***

- Nhóm đã tiến hành nghiên cứu và trình bày các khái niệm về DevOps, Ngôn ngữ lập trình C#, Hệ quản trị cơ sở dữ liệu SQL Server.

- Nắm được cách thức tổ chức họp bàn, phân công nhiệm vụ cụ thể cho từng thành viên vào từng giai đoạn. Các thành viên cơ bản nhiệt tình, trách nhiệm, nắm rõ nội dung nhiệm vụ, thực hiện đầy đủ các nội dung được giao.

- Nắm được quá trình triển khai phần mềm, thao tác tải phần mềm từ GitHub và thao tác sử dụng trên máy tính cá nhân của các thành viên trong nhóm. Quá trình sử dụng cho thấy phần mềm hoạt động cơ bản đúng với yêu cầu đặt ra. Một số lỗi gặp phải trong quá trình sử dụng đã được nhóm khắc phục và update qua các phiên bản.

***b) Về thực nghiệm***

- Xây dựng được chương trình quản lý đề tài hỗ trợ được các chức năng quan trọng như: Đăng ký tài khoản; Các chức năng quản lý và tra cứu đề tài, tác giả; Báo cáo tổng hợp các nội dung.

- Hướng phát triển sắp tới: Nghiên cứu và triển khai thêm các tính năng đăng ký trực tuyến thực hiện nhiệm vụ đề tài, Lập kế hoạch và Dự đoán kinh phí, thời gian.

# Một số khó khăn

Mất nhiều thời gian làm quen với phương pháp làm việc thông qua GitHub do các thành viên chưa có nhiều kinh nghiệm.