TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO MÔN KIẾN TẬP CÔNG NGHIỆP**

**TÌM HIỂU GITHUB/ AZURE/ JENKINS**

*Người hướng dẫn*: **TS TRẦN THANH PHƯỚC**

*Người thực hiện*: **NGUYỄN THIỆN HƯNG – 51603131**

**TRẦN BẢO DUY – 51603080**

Lớp : **16050301**

**16050303**

Khoá : **20**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2019**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO MÔN KIẾN TẬP CÔNG NGHIỆP**

**TÌM HIỂU GITHUB/ AZURE/ JENKINS**

*Người hướng dẫn*: **TS TRẦN THANH PHƯỚC**

*Người thực hiện*: **NGUYỄN THIỆN HƯNG – 51603131**

**TRẦN BẢO DUY – 51603080**

Lớp : **16050301**

**16050303**

Khoá : **20**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2019**

LỜI CẢM ƠN

Trên thực tế không có sự thành công nào mà không gắn liền với những sự hỗ trợ, giúp đỡ dù ít hay nhiều, dù trực tiếp hay gián tiếp của người khác. Trong suốt thời gian từ khi bắt đầu học tập ở giảng đường đại học đến nay, chúng em đã nhận được rất nhiều sự quan tâm, giúp đỡ của quý Thầy Cô, gia đình, bạn bè cùng các anh trong công ty.

Với lòng biết ơn sâu sắc nhất, chúng em xin gửi đến quý Thầy Cô ở Khoa Công Nghệ Thông Tin – Trường Đại Học Tôn Đức Thắng đã cùng với tri thức và tâm huyết của mình để truyền đạt vốn kiến thức quý báu cho chúng em trong suốt thời gian học tập tại trường. Và đặc biệt, trong học kỳ này, Khoa đã tổ chức cho chúng em được tiếp cận với môn học mà theo chúng em là rất hữu ích đối với sinh viên ngành Khoa Học Máy Tính. Đó là môn “Kiến tập công nghiệp”.

Chúng em xin chân thành cảm ơn Thầy Trần Thanh Phước đã tận tâm theo dõi trên hành trang bước vào doanh nghiệp. Nếu không có những lời hướng dẫn, dạy bảo của thầy thì chúng em nghĩ đồ án của chúng em rất khó có thể hoàn thiện được. Một lần nữa, chúng em xin chân thành cảm ơn thầy. Bước đầu đi vào thực tế, tìm hiểu về Integration Azure/ Jenkins/ GitHub , kiến thức của chúng em còn hạn chế và còn nhiều bỡ ngỡ. Do vậy, không tránh khỏi những thiếu sót là điều chắc chắn, chúng em rất mong nhận được những ý kiến đóng góp quý báu của Thầy và các bạn cùng lớp để kiến thức của chúng em trong lĩnh vực này được hoàn thiện hơn.

Lời Cảm Ơn thầy Trần Thanh Phước | SVTH : Nguyễn Thiện Hưng, Trần Bảo Duy. Sau cùng, chúng em xin kính chúc quý Thầy Cô trong Khoa Công Nghệ Thông Tin thật dồi dào sức khỏe, niềm tin để tiếp tục thực hiện sứ mệnh cao đẹp của mình là truyền đạt kiến thức cho thế hệ mai sau.

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của riêng tôi / chúng tôi và được sự hướng dẫn của TS Trần Thanh Phước. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày 14 tháng 6 năm 2019*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Nguyễn Thiện Hưng*

*Trần Bảo Duy*

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

MỤC LỤC

[LỜI CẢM ƠN i](#_Toc11759725)

[PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN iii](#_Toc11759726)

[MỤC LỤC 1](#_Toc11759727)

[DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ 3](#_Toc11759728)

[CHƯƠNG 1 – GITHUB 4](#_Toc11759729)

[1.1 Tổng quan về GitHub: 4](#_Toc11759730)

[1.1.1. Khái niệm về GitHub 4](#_Toc11759731)

[1.1.2 Những khái niệm liên quan: 5](#_Toc11759732)

[1.1.1.1 Tiểu mục cấp 3 10](#_Toc11759733)

[1.1.1.2 Tiểu mục cấp 3 tiếp theo. 10](#_Toc11759734)

[1.1.2 Tiểu mục cấp 2 tiếp theo 10](#_Toc11759735)

[1.2 Nội dung của chương này 10](#_Toc11759736)

[2.1 TÍCH HỢP LIÊN TỤC 11](#_Toc11759737)

[2.1.1 Tích hợp liên tục là gì? 11](#_Toc11759738)

[2.1.2 Đặc điểm của việc tích hợp liên tục: 13](#_Toc11759739)

[2.1.3 Lợi ích và khó khăn của tích hợp liên tục, 14](#_Toc11759740)

[2.2 JENKINS 15](#_Toc11759741)

[2.2.1 Jenkins là gì ? 15](#_Toc11759742)

[2.2.2 CI với Jenkins 17](#_Toc11759743)

[2.2.3 Lợi ích và khó khăn của Jenkins 19](#_Toc11759744)

[3.1 Microsoft Azure là gì ? 20](#_Toc11759745)

[3.2 Các dịch vụ, mô hình dịch vụ đám mây và lợi ích của Microsoft Azure: 22](#_Toc11759746)

[3.2.1 Các dịch vụ của Microsoft Azure: 22](#_Toc11759747)

[3.2.2 Mô hình dịch vụ đám mây của Microsoft Azure: 24](#_Toc11759748)

[3.2.3 Lợi ích của Microsoft Azure: 27](#_Toc11759749)

[3.3 Máy ảo trên Microsoft Azure: 29](#_Toc11759750)

[3.4 Lưu trữ trên Microsoft Azure 30](#_Toc11759751)

[3.5 Virtual network trong Microsoft Azure: 35](#_Toc11759752)

DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ

**DANH MỤC HÌNH**

Hình 1.1 Mô hình Git hoạt động trên Repo 6

Hình 1.2 Mô hình làm việc trên Repo của Dev 9

Hình 2.2 Các bước của một mô hình tích hợp liên tục 14

Hình 2.3 Build , test và deploy với Jenkins 17

Hình 2.4 Một số các công cụ và nền tảng có thể tích hợp với Jenkins 18

Hình 2.5 Lược đồ mô tả CI khi tích hợp với Jenkins 18

Hình 3.1 Điện toán đám mây với Microsoft Azure 22

Hình 3.2 Các nhóm dịch vụ nổi bật trong Microsoft Azure 24

Hình 3.3 Mô hình On premises và mô hình IaaS 25

Hình 3.4 Mô hình On premises và mô hình PaaS 26

Hình 3.4 Mô hình On premises và mô hình SaaS 27

CHƯƠNG 1 – GITHUB

* 1. Tổng quan về GitHub:
     1. Khái niệm về GitHub:

***Trước khi tìm hiểu về GitHub thì ta cần biết về Version Control là gì?***

Version Control giúp các developer kiểm soát và quản lý được những thay đổi đến với phần code của dự án phần mềm. Vì sự phát triển của phần mềm, version control trở nên cần thiết hơn.

Với **branching**, một developer nhân đôi phần mã nguồn (được gọi là **repository)**. Developer có thể thực hiện thay đổi một cách an toàn cho phần code mà không ảnh hưởng đến phần còn lại của dự án.

Sau đó, khi developer lấy phần code hoạt động chính xác, người đó có thể **merge**  code đó trở lại mã nguồn chính.

***Vậy GitHub là gì?***

GitHub là một công ty vì lợi nhuận cung cấp dịch vụ lưu trữ kho Git dựa trên Cloud. Về cơ bản, nó giúp các cá nhân và nhóm sử dụng Git để kiểm soát phiên bản và cộng tác dễ dàng hơn rất nhiều.

Giao diện GitHub, đủ thân thiện với người dùng để ngay cả các lập trình viên mới làm quen cũng có thể tận dụng được Git. Không có GitHub, sử dụng Git thường đòi hỏi một chút hiểu biết về kỹ thuật và sử dụng dòng lệnh (command line).

GitHub rất thân thiện với người dùng, mặc dù vậy, một số người thậm chí còn sử dụng GitHub để quản lý các dự án khác – như viết sách.

Ngoài ra, bất kỳ ai cũng có thể đăng ký và lưu trữ kho lưu trữ mã công khai miễn phí, điều này khiến Github đặc biệt phổ biến với các dự án nguồn mở.

Là một công ty, GitHub kiếm tiền bằng cách bán các kho **private repository** được lưu trữ, cũng như các kế hoạch tập trung vào kinh doanh khác giúp các tổ chức dễ dàng quản lý các thành viên nhóm và bảo mật.

Ở đây chúng em sẽ nói những thứ cơ bản nhất và mang tính thiết yếu về WorkFlow GitHub trong: Repositories, Branches, commits và Pull Request. Ví dụ về cách tạo ra một Repository có tên Hello World, và workflow của việc Pull Request, một cách quen thuộc để create và review code. Github được xem là một ứng dụng giúp xây dựng các remote repository, các chức năng của nó đều phát triển dựa vào thư viện Git.

Ví dụ khi chúng ta làm một dự án và cần lưu trữ trên remote repo, lúc này bắt buộc phải mua server và xây dựng remote trên đó, điều này khá tốn chi phí và có khi lại không bảo mật. Thay vào đó chúng ta nên sử dụng Github để tạo remote repo, sau đó các thành viên sẽ liên kết tới remote này để làm việc.

* + 1. Những khái niệm liên quan:

1. ***Repository:***

Repository hay còn gọi là Repo, dịch ra tiếng Việt có nghĩa là kho, đây chính là nơi chứa tất cả mã nguồn cho một dự án được

Phân loại Repo:

***Local repository:*** Là repo được cài đặt trên máy tính của lập trình viên, repo này sẽ đồng bộ hóa với remote repo bằng các lệnh của git

***Remote repository:*** Là repo được cài đặt trên server chuyên dụng, điển hình hiện nay là GitHub.



Hình 1.1 Mô hình Git làm việc trên Repo

Nguồn: stackoverflow.com

Khi thực hiện các thay đổi dự án một cách cục bộ (locally), bạn có thể cập nhật chúng với kho lưu trữ từ xa. Trong Git, một điều khiển từ xa là máy chủ nơi code lưu trữ. Trong trường hợp, máy chủ đó là một kho lưu trữ trên GitHub hoặc GitHub Enterprise.

1. ***Branch:***

Một branch là phiên bản song song của **repository**. Nó chứa trong **repository**, nhưng không ảnh hưởng đến nhánh chính hoặc nhánh chính cho phép làm việc tự do mà không làm gián đoạn phiên bản “trực tiếp”. Khi ta tạo ra những thay đổi, ta có thể **merge** branch vào lại ***branch master*** để publish những thay đổi của bạn.

1. ***Clone:***

Một **clone** là một bản copy của repo đang sống trong máy tính của bạn thay vì trên server website ở nơi nào đó, hoặc hành động tạo ra bản copy đó. Với bản **clone** chúng ta có thể edit các file trong các editor và sử dụng Git để theo dõi những thay đổi của bản mà không cần phải trực tuyến. Tuy nhiên, nó được kết nối với phiên bản từ xa để những thay đổi có thể được đồng bộ hóa giữa hai phiên bản. Ta có thể **push** các thay đổi cục bộ đểgiữ cho chúng được đồng bộ hóa khi ta dùng phiên bản remote/ online.

1. ***Commit:***

Một commit hay “revision”, là một thay đổi riêng lẻ đối với một tệp (hoặc bộ tệp). Giống như khi bạn lưu tệp, ngoại trừ với Git, mỗi khi bạn lưu tệp đó sẽ tạo một ID duy nhất (a.k.a. “SHA” hoặc là “băm”) cho phép bạn ghi lại những thay đổi được thực hiện khi nào và bởi ai.Commit thường chứa một commit message là một mô tả ngắn gọn về những thay đổi đã được thực hiện.

Commit này đang được chứa tại repository, các commit nối tiếp với nhau theo thứ tự thời gian. Bằng việc lần theo commit này từ trạng thái mới nhất thì có thể biết được lịch sử thay đổi trong quá khứ hoặc nội dung thay đổi đó.

1. ***Fork:***

Một Fork là một bản copy của một repository(Kho chứa source code của bạn trên GitHub). Việc fork một repository cho phép bạn dễ dàng chỉnh sửa, thay đổi source code mà không cho phép bạn dễ dàng chỉnh sửa, thay đổi source code mà không ảnh hưởng tới source gốc.

Một ví dụ về việc sử dụng fork, là khi bạn muốn fix bug source code trên repository của một ai đó, khi đó bạn cần thực hiện theo quy trình sau:

1. Fork repository đó về tài khoản Github của mình.
2. Thực hiện fix bug.
3. Gửi một Pull Request tới repository gốc.

Khi chủ sở hữu của repository nơi ta fork, sẽ review chỉnh sửa thay đổi của bạn, và tiến hành merge nội dung chỉnh sửa vào source gốc.

1. ***Merge:***

Merge source từ một nhánh khác vào nhánh hiện tại. Bạn sẽ hiểu sâu hơn về nó trong các ví dụ ở các bài sau.

Chú ý:

- Kiểm tra branch hiện đang làm việc trước khi merge

- Phải đẩy tất cả những thay đổi dưới máy local lên Git trước khi merge

- Trước khi merge phải lấy hết những thay đổi mới nhất của các branch khác, hay ít nhất branch cần merge về máy.

- Merge thành công thì nên đẩy source lên lại server

- Nên merge bằng GUI tool.

1. ***Push:***

Push được sử dụng để đưa nội dung kho lưu trữ cục bộ lên server. Push là cách bạn chuyển giao các commit từ kho lưu trữ cục bộ của bạn lên server.Nếu bạn có những thay đổi gì, thì bạn muốn đẩy những thay đổi đó để người khác có thể truy cập chúng.

1. ***Pull:***

Pull đề cập đến khi ta đang fetching các thay đổi và merging chúng. Ví dụ: nếu ai đó đã chỉnh sửa tệp từ xa mà cả hai bạn đang làm việc, bạn sẽ muốn **pull** các thay đổi đó sao cho bản sao cục bộ của mình để cập nhật.

1. ***Pull Request:***

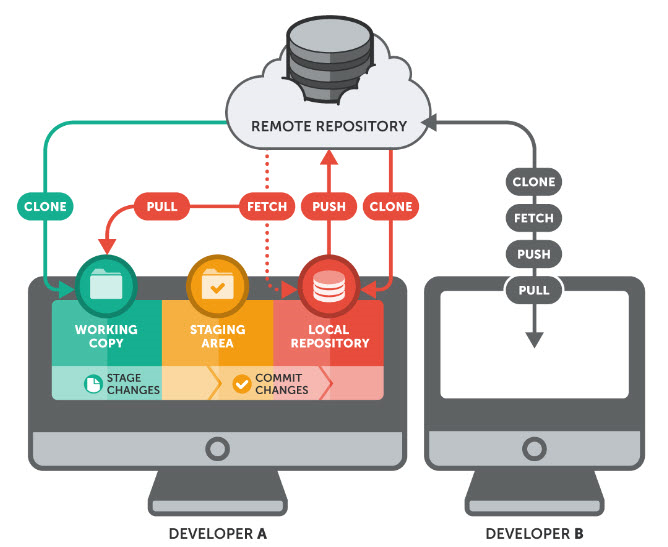
**Pull Request** để bắt đầu thảo luận về những commit của bạn. Khi commit được mở, bạn có thể thảo luận và xem xét các thay đổi tiềm năng với các collaborator va theo dõi các commit trước khi các thay đổi được merge vào master branch.Vì chúng tích hợp chặt chẽ với repo Git bên dưới, bất cứ ai cũng có thể thấy chính xác những thay đổi sẽ được merge nếu họ chấp nhận lời request của bạn.

Sau khi khởi tạo PR, bạn sẽ thấy trang đánh giá hiển thị tổng quan high-level về các thay đổi giữa các branch của bạn

Bạn có thể mở 1 cái pull request tại bất kỳ thời điểm nào trong quá trình phát triển: khi bạn có ít hoặc không có mã nhưng muốn chia sẻ một số screenshot hoặc ý tưởng chung. Khi bạn gặp khó và cần giúp đỡ hoặc lời khuyên, hoặc bạn đã sẵn sàng để ai đó xem xét công việc của bạn. Bằng cách sử dụng hệ thống @mention của GitHub trong tin nhắn Pull Request, bạn có yêu cầu phản hồi từ những người hoặc nhóm cụ thể.

1. ***Issue:***

Issue là nơi để đề xuất các cải tiến, nhiệm vụ hoặc câu hỏi liên quan đến kho lưu trữ. Các issue được tạo ra bởi bất kỳ ai (cho public repository), và được kiểm duyệt bởi repository collaborator (người cộng tác trong kho lưu trữ). Mỗi issue chứa diễn đàn thảo luận riêng, có thể được gắn nhãn và gán cho người dùng.



Hình 1.2 Mô hình làm việc trên Repository của các Developer

Nguồn: csc.edu.vn

**CHƯƠNG 2 – TÍCH HỢP LIÊN TỤC VÀ JENKINS**

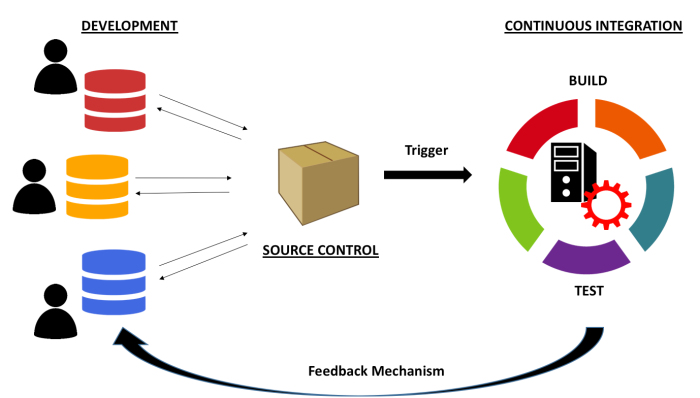
2.1 TÍCH HỢP LIÊN TỤC

2.1.1 Tích hợp liên tục là gì?

Trong môi trường tích hợp liên tục, một vấn đề rất phổ biến và thường xuyên gặp phải đối với nhiều nhóm phát triển là một quy trình làm việc phân mảnh. Không hiệu quả trong việc phát triển phần mềm có thể gây ảnh hưởng đến kết quả đầu ra.

Ở môi trường phát triển thông thường, các cá nhân trong nhóm có xu hướng làm việc độc lập. Các kỹ sư phần mềm thường xuyên tạo ra các đoạn mã lớn mà không màng đến việc kiểm soát phiên bản. Sau khi một nhà phát triển, kỹ sư hoàn thành công việc, thì họ thêm phần việc của họ vào mã cơ sở. Sau đó, một nhóm khác tự chạy các thử nghiệm để xác minh bản dựng của mã cơ sở. Việc này xảy ra rất nhiều trong môi trường phát triển phần mềm hiện nay, và nó trở thành một trong những vấn đề rất nhức nhói và khó chịu trong giai đoạn phát triển phần mềm.

Tích hợp liên tục (Continuous Integration - CI) là một thực tiễn phát triển trong đó các nhà phát triển cần phải gửi (commit) các thay đổi mã nguồn đến kho lưu trữ chung nhiều lần trong ngày hoặc thường xuyên hơn. Mỗi lần gửi được thực hiện trong kho lưu trữ sau đó được xây dựng (build). Điều này cho phép các nhóm phát hiện sớm các vấn đề. Ngoài ra, tùy thuộc vào công cụ tích hợp liên tục,còn có một số chức năng khác như triển khai ứng dụng xây dựng trên máy chủ thử nghiệm, cung cấp cho các nhóm liên quan về kết quả xây dựng và thử nghiệm.

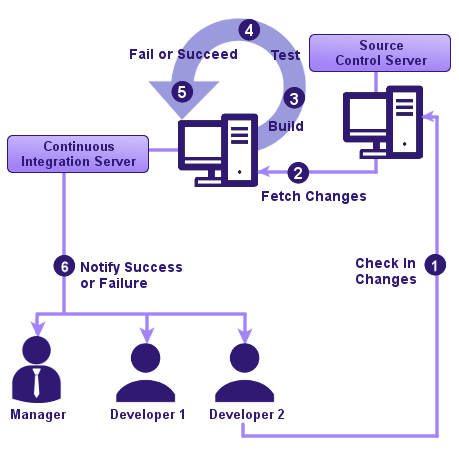


Hình 2.1 Mô hình tích hợp liên tục

Nguồn, dotnetvibes.com

CI nhắm tới mục đích cắt giảm sự thiếu hiệu quả của chu trình phát triển bằng cách cho phép các nhà phát triển biên dịch code của nhóm từ kho lưu trữ kiểm soát phiên bản dùng chung. CI cũng cho phép bạn tự động hóa kiểm tra để bạn có thể thiết lập hệ thống để tự động chạy các thử nghiệm đơn vị hoặc các thử nghiệm tích hợp.

CI tự động theo dõi các lần gửi mà mỗi cá nhân thực hiện. Điều này tinh giản hóa việc xây dựng và xác minh các đoạn code để việc thử nghiệm không thất thoát quá nhiều. CI được chạy trên một máy chủ dùng chung làm tăng khả năng hiển thị giữa các nhà phát triển, vì vậy tất cả các nhà phát triển trong một dự án đều biết về những thay đổi trong mã cơ sở ngày này qua ngày khác. Ngoài ra, họ có thể định cấu hình máy chủ để thông báo cho các nhà phát triển khi họ gửi một mã bị lỗi hoặc hỏng để họ có thể sửa bất kỳ lỗi nào họ đã gửi lên hệ thống.



Hình 2.2 Các bước của một mô hình tích hợp liên tục

Nguồn, www.thinksys.com

2.1.2 Đặc điểm của việc tích hợp liên tục:

* Quản lý phiên bản phần mềm, giúp các nhà phát triển kiểm soát các phiên bản đã code đã được gửi lên server.
* Tự động hóa trong việc xây dựng và thử nghiệm code.
* Đội ngũ các nhà phát triển sẽ thường xuyên gửi các đoạn mã của mình đến nơi lưu trữ chính của đội ngũ, thường là một kho lưu trữ được chia sẻ (repository).
* Khi code có sự thay đổi thì sẽ được xây dựng lại thông qua build server.
* Khi xây dựng hoặc thử nghiệm có xuất hiện ra lỗi thì server sẽ tự động báo lỗi cho đội ngũ lập trình cũng như nhà quản lý dự án.
* Phát hành phiên bản hoàn chỉnh cho khách hàng khi không còn lỗi nữa.
* Tự động phân phối phiên bản mới lên môi trường phát triển hoặc là tới khách hàng.
* Với cơ chế phản hồi lỗi của tích hợp liên tục, Đội ngũ phát triển cũng như nhà quản lý dự án sẽ nhìn thấy sớm được những khó khăn, lỗi xảy ra trong gian đoạn phát triển phần mềm để có thể nhanh chóng khắc phục để không bị ảnh hưởng trong tương lai.

2.1.3 Lợi ích và khó khăn của tích hợp liên tục,

***Lợi ích***:

* + Giảm thiểu rủi ro do lỗi được phát hiện sớm trong quá trình phát triển.
  + Giảm thiểu sự lặp lại của các quá trình, tiến trình.
  + Tạo phần mềm có giá trị sử dụng sớm nhất có thể và sẵn sang triển khai mọi lúc mọi nơi.
  + Cung cấp cái nhìn xuyên suốt tổng quan và cụ thể cho từng giai đoạn.
  + Nâng cáo kỹ năng của đội ngũ nhân viên phát triển phần mềm.
  + Cải thiện chất lượng phần mềm.

***Khó khăn:***

* + Cần thời gian thiết lập hệ thống ban đầu.
  + Yêu cầu các thành viên, các cá nhân trong đội ngũ dự án phải có các kỹ năng và am hiểu các mô hình như Agile, hệ thống tích hợp CI, các công cụ hỗ trợ cho CI,..
  + Các chi phí cần thiết để phát triển và duy trì server cho CI.

2.2 JENKINS

2.2.1 Jenkins là gì ?

Khi nhắc đến CI, những nhà phát triển sẽ nghĩ ngay đến Jenkins. Mặc dù có rất nhiều công cụ để hỗ trợ CI như GitLab CI, Travis CI, Bamboo,… nhưng Jenkins vẫn luôn là công cụ được nhắc đến nhiều nhất và được ưu tiên trong việc phát triển, áp dụng CI.

Jenkins là một máy chủ tích hợp liên tục mã nguồn mở có khả năng phối hợp một chuỗi các hành động giúp đạt được quy trình tích hợp liên tục theo một cách cách tự động. Jenkins giúp tự động hóa phần máy móc của quy trình phát triển phần mềm, với sự tích hợp liên tục và tạo điều kiện cho các khía cạnh kỹ thuật của việc phân phối liên tục (Continuous Delivery).

Jenkins được sử dụng để xây dựng và thử nghiệm các dự án phần mềm của đội ngũ các nhà phát triển một cách liên tục, giúp các họ dễ dàng tích hợp các thay đổi cho dự án hơn và giúp người dùng dễ dàng có được bản dựng mới. Nó cũng cho phép bạn liên tục cung cấp phần mềm của mình bằng cách tích hợp với một số lượng lớn các công nghệ thử nghiệm và triển khai. Với Jenkins, các tổ chức có thể đẩy nhanh quá trình phát triển phần mềm thông qua tự động hóa. Jenkins tích hợp các quy trình vòng đời phát triển của tất cả các loại, bao gồm xây dựng, tạ o dựng tài liệu, thử nghiệm, đóng gói, giai đoạn, triển khai, phân tích tĩnh và nhiều hơn nữa.

Nó cũng được viết bằng ngôn ngữ Java. Điều này khiến cho nó có thể được tích hợp va áp dụng dễ dàng trên nhiều nền tảng khác nhau như Windows, Linux, macOS. Việc này cũng ảnh hưởng đến số lượng người sử dụng Jenkins, với hơn 300 ngàn lượt cài đặt và đang ngày càng tăng.

Hình 2.3 Build, test và deploy với Jenkins

Nguồn, tech.vccloud.vn

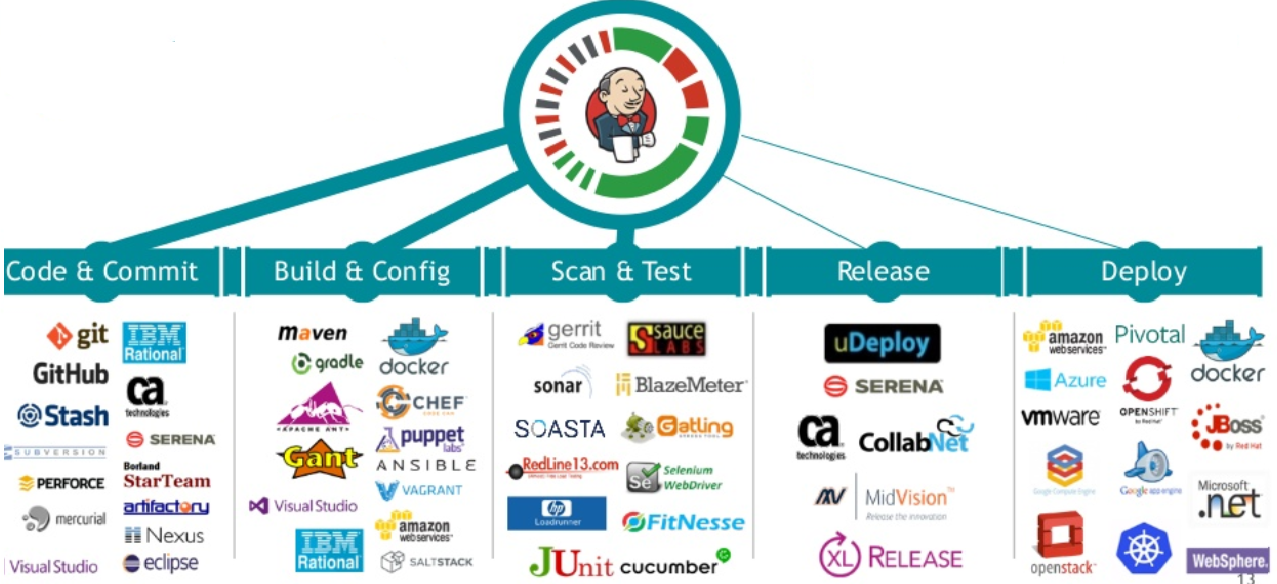


Hình 2.3 Build , test và deploy với Jenkins

Nguồn : tech.vccloud.vn

Một trong những điểm nổi bật của Jenkins là sự hỗ trợ plugins. Ở cài đặt mặc định, Jenkins không hỗ trợ nhiều tính năng cho quá trình CI. Nhưng với sự hỗ trợ của plugins, việc này khiến cho Jenkins thành một công cụ mạnh mẽ để có thể tích hợp với bất kì công cụ nào cũng như bất kì nền tảng nào.

Jenkins khác biệt so với các máy chủ CI khác vì sự đa dạng plugins của nó. Các plugins được viết bởi cộng đồng các nhà phát triển, các kỹ sư phần mềm với ý định hướng tới sự đa dạng hóa cho Jenkins. Hiện tại với hơn 1000+ plugins và đang tăng, Jenkins trở thành một công cụ đáng chú ý cho cộng đồng các nhà phát triển phần mềm để áp dụng thực tiễn CI của họ.



Hình 2.4 Một số các công cụ và nền tảng có thể tích hợp với Jenkins

Nguồn: awesome-tech.readthedocs.io

2.2.2 CI với Jenkins



Hình 2.5 Lược đồ mô tả CI khi tích hợp với Jenkins

Nguồn www.edureka.co

Dựa vào biểu đồ trên, nó diễn tả những hành động sau:

* Đầu tiên, một nhà phát triển gửi code vào kho lưu trữ mã nguồn. Trong khi đó, máy chủ Jenkins kiểm tra kho lưu trữ theo định kỳ để thay đổi (hoặc có thể kiểm tra theo event được trigger).
* Ngay sau khi một lần gửi xảy ra ở kho, máy chủ Jenkins phát hiện các thay đổi đã xảy ra trong kho lưu trữ mã nguồn. Jenkins sẽ kéo (pull) những thay đổi đó và sẽ bắt đầu chuẩn bị một bản dựng mới.
* Nếu việc xây dựng thất bại, thì đội ngũ phát triển sẽ được thông báo.
* Nếu quá trình xây dựng thành công, thì Jenkins sẽ triển khai máy chủ thử nghiệm.
* Sau khi thử nghiệm, Jenkins tạo phản hồi và sau đó thông báo cho các nhà phát triển về kết quả xây dựng và thử nghiệm.
* Nếu việc thử nghiệm thành công, nó sẽ tiếp tục đưa đến máy chủ làm việc.
* Nó sẽ tiếp tục kiểm tra kho lưu trữ mã nguồn để biết các thay đổi được thực hiện trong mã nguồn và toàn bộ quá trình tiếp tục lặp lại.

Việc áp dụng Jenkins khiến cho hiệu quả công việc cũng như quy trình phát triển phần mềm thay đổi rõ rệt, sau đây là bảng so sánh “trước khi áp dụng Jenkins” và “Sau khi áp dụng Jenkins”:

|  |  |
| --- | --- |
| Trước khi áp dụng Jenkins | Sau khi áp dụng Jenkins |
| Các nhà phát triển khi đã hoàn thành xong phần code của họ, họ sẽ gửi cùng một lúc và việc xây dựng và thử nghiệm sẽ diễn ra sau,  Quy trình xảy ra rất ít và một việc xây dựng có thể làm theo nhiều cách, | Các code được xây dựng và kiểm tra ngay khi nhà phát triển gửi mã lên kho. Jenkin sẽ xây dựng và kiểm tra code nhiều lần trong ngày,  Nếu quá trình xây dựng thành công, thì Jenkins sẽ triển khai code vào máy chủ thử nghiệm và thông báo cho nhóm triển khai.  Nếu quá trình xây dựng thất bại, thì Jenkins sẽ thông báo lỗi cho nhóm nhà phát triển. |
| Vì mã được xây dựng cùng một lúc, một số nhà phát triển sẽ cần đợi cho đến khi các nhà phát triển khác hoàn thành việc code của họ để kiểm tra bản dựng, | Code được xây dựng ngay lập tức sau khi bất kỳ lần gửi nào của nhà phát triển. |
| Không phải là một nhiệm vụ dễ dàng để cô lập, phát hiện và sửa lỗi cho nhiều lần gửi, | Vì code được xây dựng sau mỗi lần gửi của một nhà phát triển, nên rất dễ phát hiện code của ai gây ra lỗi khi xây dựng |
| Quá trình xây dựng code và kiểm tra là hoàn toàn thủ công, vì vậy có rất nhiều cơ hội cho thất bại, | Tự động xây dựng và quá trình kiểm tra tiết kiệm thời gian và giảm khuyết điểm, lỗi lầm. |
| Code được triển khai khi tất cả các lỗi được sửa chữa và kiểm tra. | Mã được triển khai sau mỗi lần xây dựng và thử nghiệm thành công. |
| Chu kỳ phát triển chậm | Chu kỳ phát triển nhanh. Các tính năng mới có sẵn hơn cho người dùng. Tăng lợi nhuận. |

Như đã miêu tả, với Jenkins, quá trình phát triển phần mềm được thay đổi rõ rệt,

linh động hơn, giảm bớt rủi ro cũng như làm giảm đi khó khăn cho nhà phát triển.

2.2.3 Lợi ích và khó khăn của Jenkins

***Lợi ích:***

* Jenkins đang được quản lý bởi cộng đồng rất cởi mở. Hàng tháng, họ tổ chức các cuộc họp công cộng và lấy ý kiến của công chúng để phát triển dự án Jenkins.
* Khi công nghệ phát triển, Jenkins cũng vậy. Cho đến nay Jenkins có khoảng hơn 1000 plugin được xuất bản trong cơ sở dữ liệu plugin của nó. Với các plugin, Jenkins càng trở nên mạnh mẽ và giàu tính năng.
* Jenkins cũng hỗ trợ kiến trúc dựa trên đám mây để bạn có thể triển khai Jenkins trong các nền tảng dựa trên đám mây.
* Lý do tại sao Jenkins trở nên phổ biến là vì nó được tạo ra bởi một nhà phát triển cho các nhà phát triển.

***Khó khăn:***

* Giao diện của nó đã lỗi thời và không thân thiện với người dùng so với các xu hướng UI hiện tại.
* Mặc dù Jenkins được nhiều nhà phát triển yêu thích, nhưng không dễ để duy trì nó vì Jenkins chạy trên máy chủ và yêu cầu một số kỹ năng như quản trị viên máy chủ để theo dõi hoạt động của nó cũng như chi phí để duy trì máy chủ Jenkins.
* Một trong những lý do khiến nhiều người không triển khai Jenkins là do khó cài đặt và định cấu hình Jenkins.
* CI thường xuyên bị phá vỡ do một số thay đổi cài đặt nhỏ. CI sẽ bị tạm dừng và do đó đòi hỏi một số nhà phát triển chú ý.

**CHƯƠNG 3 – MICROSOFT AZURE**

3.1 Microsoft Azure là gì ?

Có nhiều công ty gần đây đã bắt đầu sử dụng dịch vụ đám mây cho cơ sở hạ tầng mạng của họ. Với sự phát triển của công nghệ đám mây, việc các công ty sử dụng dịch vụ đám mây sẽ hiệu quả và tiết kiệm chi phí hơn nhiều. Một trong những nhà cung cấp dịch vụ đám mây phổ biến nhất là Microsoft với Microsoft Azure dành cho những người muốn chọn sử dụng các sản phẩm và dịch vụ của Microsoft Cloud. Microsoft Azure cùng với hai đối thủ cạnh tranh lớn nhất của nó là Amazon Web Service và Google Cloud platform là nền tảng điện toán đám mây có thể cung cấp mọi thứ mà doanh nghiệp cần để chạy tất cả hoặc một phần hoạt động điện toán của nó hầu như là bao gồm máy chủ, lưu trữ, cơ sở dữ liệu, mạng, phân tích và hơn thế nữa.

Lúc trước, lựa chọn duy nhất có sẵn cho các công ty là xây dựng và quản lý phần cứng vật lý cần thiết cho việc tính toán, bao gồm máy chủ, lưu trữ đĩa và chuyển mạch Ethernet. Nhưng ngày nay, các công ty có thể sử dụng một nền tảng điện toán đám mây công cộng như Azure, mua và duy trì tất cả các phần cứng máy tính. Điều này có nghĩa là các công ty có thể thuê tài nguyên phần cứng một cách hiệu quả khi cần thiết.



Hình 3.1 Điện toán đám mây với Microsoft Azure

Nguồn : newsignature.com

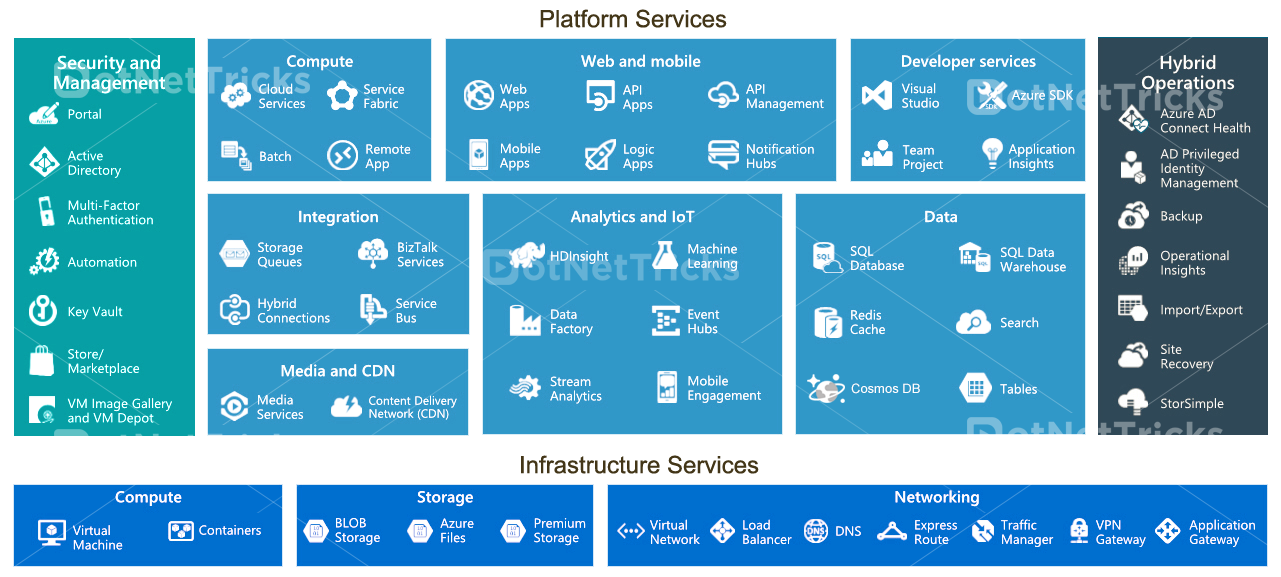
Azure cung cấp một loạt các sản phẩm và dịch vụ được thiết kế để đáp ứng tất cả các nhu cầu của các công ty như là xây dựng, thử nghiệm, triển khai và quản lý các ứng dụng và dịch vụ thông qua mạng lưới các trung tâm dữ liệu do Microsoft quản lý.

3.2 Các dịch vụ, mô hình dịch vụ đám mây và lợi ích của Microsoft Azure:

3.2.1 Các dịch vụ của Microsoft Azure:

Microsoft Azure cung cấp một số lượng lớn dịch vụ với hơn 600 dịch vụ, với các dịch vụ thuộc nhiều nhóm dịch vụ khác nhau. Việc này khiến các công ty và các nhà phát triển có thể dễ dàng lựa chọn các dịch vụ phù hợp cho môi trường phát triển của mình. Trong đó, các nhóm dịch vụ nổi bật của Microsoft Azure là :

* **Dịch vụ tính toán với máy ảo** : Tạo máy ảo Microsoft hoặc Linux (VM) chỉ trong vài phút từ nhiều lựa chọn mẫu thị trường hoặc từ hình ảnh máy tùy chỉnh của riêng người dùng. Các máy ảo dựa trên đám mây này sẽ lưu trữ các ứng dụng và dịch vụ của người dùng như thể chúng nằm trong trung tâm dữ liệu của riêng họ.
* **Dịch vụ quản lý dữ liệu với SQL database** : Azure cung cấp cơ sở dữ liệu quan hệ SQL được từ một đến số lượng không giới hạn, dưới dạng dịch vụ. Điều này giúp người dùng tiết kiệm chi phí cho phần cứng, phần mềm và nhu cầu chuyên môn nội bộ.
* **Dịch vụ tên miền Active Directory Azure :** Được xây dựng trên cùng một công nghệ đã được chứng minh như Windows Active Directory, dịch vụ này của Azure cho phép người dùng quản lý từ xa chính sách nhóm, xác thực và mọi thứ khác. Điều này làm cho việc di chuyển một phần hoặc toàn bộ cấu trúc bảo mật hiện có sang đám mây dễ dàng hơn chỉ bằng vài cú nhấp chuột.
* **Các dịch vụ ứng dụng :** Với Azure, các nhà phát triển dễ dàng hơn bao giờ hết để tạo và triển khai các ứng dụng tương thích trên tất cả các nền tảng di động và web phổ biến. Với đám mây đáng tin cậy, có thể mở rộng cho phép người dùng phản hồi nhanh chóng và doanh nghiệp của họ, tiết kiệm thời gian và tiền bạc. Với việc giới thiệu Azure WebApps cho Azure Marketplace, việc quản lý sản xuất, thử nghiệm và triển khai các ứng dụng web có tính mở rộng nhanh như doanh nghiệp của họ sẽ dễ dàng hơn bao giờ hết. Các API dựng sẵn cho các dịch vụ đám mây phổ biến như Office 365, Salesforce và nhiều hơn nữa giúp thúc đẩy việc phát triển các ứng dụng của doanh nghiệp.
* **Dịch vụ nhóm Visual Studio :** Một dịch vụ bổ trợ có sẵn trong Azure, các dịch vụ nhóm Visual Studio cung cấp giải pháp quản lý vòng đời ứng dụng hoàn chỉnh trong đám mây của Microsoft. Các nhà phát triển có thể chia sẻ và theo dõi các thay đổi code, thực hiện kiểm tra tải và cung cấp các ứng dụng cho sản xuất trong khi cộng tác trong Azure từ khắp nơi trên thế giới. Dịch vụ nhóm Visual Studio đơn giản hóa việc phát triển và phân phối cho các công ty lớn hoặc những công ty mới xây dựng danh mục dịch vụ.
* **Dịch vụ lưu trữ :** Dựa vào cơ sở hạ tầng toàn cầu của Microsoft để cung cấp lưu trữ dữ liệu an toàn, dễ truy cập. Với khả năng mở rộng lớn và cấu trúc giá thành thông minh cho phép người dùng lưu trữ dữ liệu không thường xuyên truy cập với mức tiết kiệm lớn, việc xây dựng gói lưu trữ an toàn và hiệu quả là rất đơn giản trong Microsoft Azure.



Hình 3.2 Các nhóm dịch vụ nổi bật trong Microsoft Azure

Nguồn : [DotNetTricks.com](https://www.dotnettricks.com/learn/azure/getting-started-with-microsoft-azure-platform)

3.2.2 Mô hình dịch vụ đám mây của Microsoft Azure:

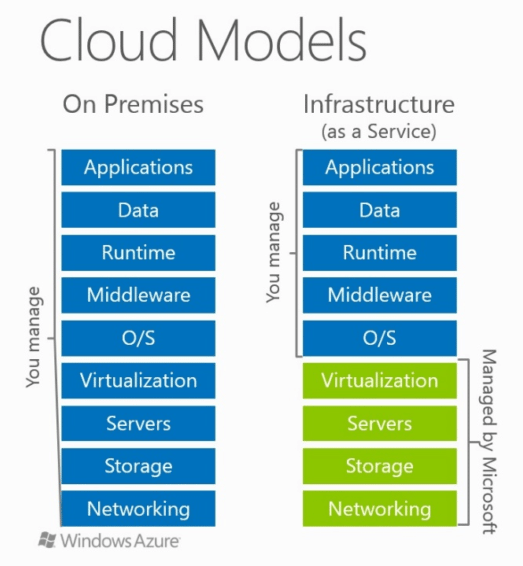
Microsoft Azure cung cấp ba mô hình đám mây, đó là : [infrastructure as a service (IaaS)](https://en.wikipedia.org/wiki/Infrastructure_as_a_service), [platform as a service (PaaS)](https://en.wikipedia.org/wiki/Platform_as_a_service) và [software as a service (SaaS)](https://en.wikipedia.org/wiki/Software_as_a_service). Tùy thuộc vào mô hình mà sự quản lý các thành phần giữa doanh nghiệp và Microsoft Azure sẽ khác nhau.

***Infastructure as a service (IaaS) :***

Nhóm dịch vụ cơ bản nhất nhanh chóng trở nên phổ biến là mô hình điện toán đám mây IaaS.

So sánh IaaS với các trung tâm dữ liệu tại chỗ ( on-premise datacenters ) truyền thống. Với cơ sở tại chỗ, doanh nghiệp có trách nhiệm quản lý mọi thành phần của trung tâm dữ liệu, từ các ứng dụng kinh doanh đến các máy ảo và mạng.

Bằng cách sử dụng dịch vụ Azure IaaS, về cơ bản các doanh nghiệp đưa dữ liệu lên máy chủ trong đám mây mà người dùng có thể kiểm soát hoàn toàn. Việc duy trì phần cứng vật lý không trở thành một vấn đề vì Microsoft chăm sóc các máy chủ cho họ. Họ cũng không phải lo lắng về các chi phí liên quan đến lỗi ổ cứng và bảo trì phần cứng. Những gì doanh nghiệp sẽ phải làm là theo dõi, quản lý và vá ( patch ) các máy ảo và các chức năng hoạt động của chúng.



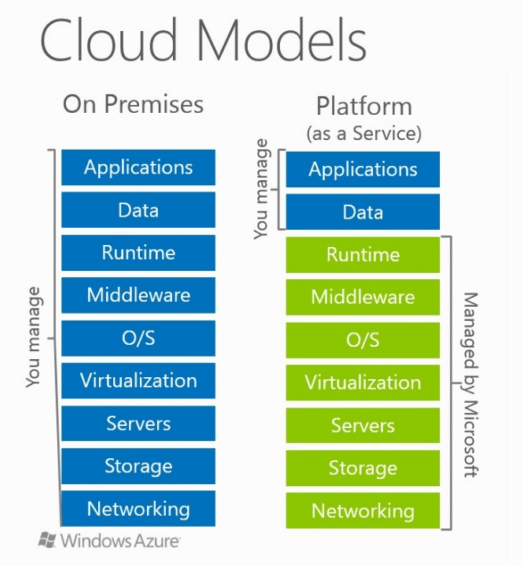
Hình 3.3 Mô hình On premises và mô hình IaaS

Nguồn : blog.5nine.com

***Platform as a service ( PaaS ) :***

PaaS là bước tiếp theo trong các mô hình dịch vụ đám mây, đó là nền tảng mà doanh nghiệp triển khai các ứng dụng của họ. Trong khi sử dụng PaaS, trách nhiệm của các nhà doanh nghiệp là quản lý các ứng dụng và cơ sở dữ liệu cụ thể, trong khi Microsoft lo tất cả các dịch vụ khác cần thiết để chạy ứng dụng của họ (bao gồm phần mềm trung gian, hệ điều hành, VM, máy chủ, lưu trữ và kết nối mạng). Do đó, trách nhiệm cho sự thành công của ứng dụng của người dùng được chia sẻ giữa tổ chức của doanh nghiệp và Microsoft, với cả hai bên cung cấp các yếu tố chính.

Việc chuyển sang PaaS cho phép các doanh nghiệp dành nhiều thời gian hơn để phát triển ứng dụng của mình, thay vì duy trì các chức năng hoạt động hỗ trợ nó. Hầu hết các công ty sử dụng PaaS đều làm như vậy bởi vì họ quan tâm đến việc chuyển đổi một ứng dụng truyền thống, nguyên khối và dựa trên bảng điều khiển (ví dụ: những ứng dụng chạy trong cửa sổ bảng điều khiển riêng thay vì trong trình duyệt, như Word hoặc Outlook,) tới các ứng dụng dựa trên nền tảng web.



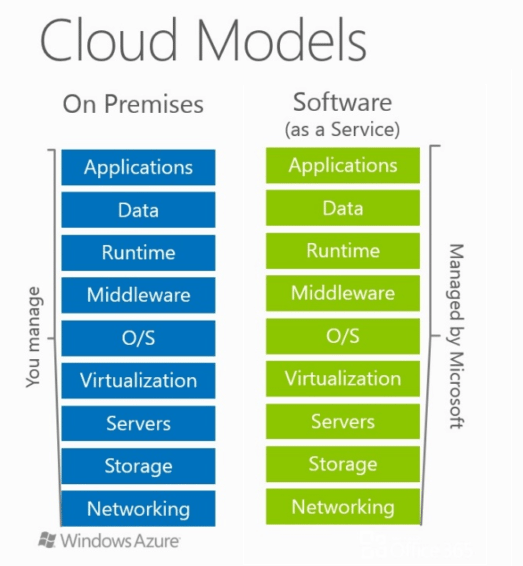
Hình 3.4 Mô hình On premises và mô hình PaaS

Nguồn : blog.5nine.com

***Software as a service ( SaaS )***

Một thỏa thuận SaaS với Azure xử lý tất cả các chức năng cơ sở hạ tầng và các hoạt động IT, để ứng dụng SaaS chạy trong đám mây trên Azure. Nó không phải là một nền tảng, nó là một ứng dụng thực tế.

Các doanh nghiệp có thể xem xét chuyển từ PaaS sang SaaS nếu họ muốn gần như hoàn toàn rảnh tay. Việc chuyển đổi sang SaaS có thể là lý tưởng nếu ứng dụng của họ được tự động hóa cao và có thể truy cập qua internet và không có bất kỳ sự phụ thuộc kéo dài nào vào một hệ điều hành cụ thể hoặc một loại phần mềm trung gian cụ thể.



Hình 3.4 Mô hình On premises và mô hình SaaS

Nguồn : blog.5nine.com

3.2.3 Lợi ích của Microsoft Azure:

1. ***Phục hồi thảm họa:***

Một lợi thế lớn của Microsoft Azure nằm ở cơ sở hạ tầng phi tập trung về tốc độ và địa lý, tạo ra các tùy chọn vô hạn cho các kế hoạch khắc phục thảm họa. Bởi vì tính linh hoạt của nó, khả năng phục hồi tiên tiến và tính tích hợp sẵn.

Là một giải pháp dựa trên đám mây, Azure rất linh hoạt - nó có thể sao lưu dữ liệu ở hầu hết mọi ngôn ngữ, trên mọi hệ điều hành và từ bất kỳ vị trí nào. Ngoài ra, người dùng còn có thể xác định tần suất và mức độ của lịch trình sao lưu dựa trên ngày, tuần hoặc tháng. Azure lưu trữ ba bản sao dữ liệu của người dùng ở ba vị trí khác nhau trong trung tâm dữ liệu và ba bản sao khác trong trung tâm dữ liệu Azure từ xa, do đó họ không bao giờ phải lo lắng về việc mất dữ liệu.

Với Azure, người dùng còn đạt được các giải pháp khắc phục thảm họa mạnh mẽ, như là :

* **Nhiều trung tâm dữ liệu để lưu trữ dữ liệu**, cho phép người dùng triển khai dịch vụ đám mây đến các địa điểm khác nhau trên khắp thế giới.
* **Azure Site Recovery**, một dịch vụ giúp đảm bảo các ứng dụng kinh doanh quan trọng của người dùng luôn chạy trực tuyến trong lúc bị mất điện hoặc gián đoạn bằng cách sao chép các khối lượng công việc đó từ một trang web chính sang một vị trí phụ.
* **Azure Traffic Manager**, tự động định tuyến mạng đến các vị trí khác nhau ( được xác định trước bởi người sử dụng ) trong trường hợp xảy ra lỗi cụ thể theo vùng.

1. ***Tính linh hoạt :***

Với Microsoft Azure, các doanh nghiệp có thể tạo các dịch vụ mới và mở rộng quy mô lưu trữ dữ liệu của mình một cách nhanh chóng. So sánh điều này với một trung tâm dữ liệu tĩnh, các nhà phát triển sẽ phải mua, cung cấp và triển khai phần cứng và hệ điều hành mới, việc này làm tốn rất nhiều thời gian cũng như chi phí của doanh nghiệp. AutoScale là một tính năng được tích hợp trong Azure Web Apps tự động điều chỉnh tài nguyên dựa trên lưu lượng khách hàng truy cập vào web để hệ thống có đủ tài nguyên cần thiết khi lưu lượng truy cập cao và tiết kiệm tiền khi hệ thống không ở trong thời gian cao điểm.

1. ***Các công cụ phát triển :***

Môi trường điện toán tại chỗ không có nhiều không gian để phát triển và thử nghiệm. Điều đó thường có nghĩa là các công ty phải mua phần cứng / phần mềm mới cho mục đích phát triển hoặc ngừng các nhiệm vụ phát triển cho đến khi họ có đủ chỗ trong hệ thống. Kết quả là năng suất giảm, ít đổi mới và chi phí cao hơn.

Với Azure, các nhà phát triển có thể dễ dàng xây dựng tất cả các kịch bản có thể và chạy qua thử nghiệm mà không phải mua thiết bị mới. Trên hết, các nhà phát triển có quyền truy cập ngay vào nhiều mẫu thiết kế, dịch vụ và giải pháp khác nhau, giúp giảm thời gian đưa các cải tiến mới vào thị trường.

1. ***Giảm chi phí :***

Azure không chỉ giúp các nhà doanh nghiệp nhanh hơn và dễ dàng hơn trong việc thêm và mở rộng cơ sở hạ tầng, nó làm cho chúng rẻ hơn. Thông thường, các dịch vụ vật lý và thiết bị cơ sở hạ tầng như bộ định tuyến, cân bằng tải nhanh chóng bổ sung chi phí lên đến hàng ngàn hoặc thậm chí hàng trăm ngàn đô la. Sau đó, cần phải có các chuyên môn cần thiết để chạy thiết bị này, tương đương với chi phí trả lương lớn.

Có ba lý do chính khiến Azure trở nên thiết thực khi nói đến chi phí:

Các doanh nghiệp không cần phải tạo một khoản đầu tư ban đầu lớn liên quan đến việc xây dựng một trung tâm dữ liệu tại chỗ hoặc từ xa. Điện toán đám mây cũng loại bỏ nhu cầu mua phần mềm bổ sung để phát triển hoặc tăng khối lượng công việc trong tương lai cũng như việc loại bỏ nhu cầu mua phần cứng thay thế.

Các doanh nghiệp sẽ tránh các chi phí liên quan đến các cuộc gọi dịch vụ và gia hạn bảo hành.

Các doanh nghiệp cũng không cần phải trả tiền cho bất kỳ tài nguyên nào ngoài những tài nguyên người dùng cần, vì hóa đơn của Azure được tính theo mỗi giây được sử dụng, làm tròn đến phút cuối cùng.

1. ***Dễ dàng tiếp cận tới tài nguyên :***

Dịch vụ điện toán đám mây Azure, cung cấp cho các bộ phận IT quyền truy cập vào một số lượng lớn tài nguyên mà họ thường không thể làm với một trung tâm dữ liệu tại chỗ. Ví dụ, nhiều công ty tiên tiến đang tìm cách áp dụng học máy và tự động hóa vào phân tích dữ liệu của họ để cho phép ra quyết định tốt hơn. Sử dụng Azure, các công ty đó có thể dễ dàng kết hợp các khả năng đó vào hệ thống của mình. Không cần chuyên môn về khoa học dữ liệu hoặc tài nguyên tốn kém. Hoặc, một số khác có thể muốn triển khai một máy chủ Linux, trong môi trường điện toán truyền thống, đòi hỏi một loạt các quy tắc để khởi động và chạy. Điều này có thể mất từ vài giờ đến vài ngày để hoàn thành. Nhưng với Azure, họ có thể thiết lập máy chủ Linux của mình trong vòng chưa đầy năm phút3.3.2 Qui định của Khoa Công nghệ thông tin.

3.3 Máy ảo trên Microsoft Azure:

Azure Virtual Machines (VM) là một trong một số loại tài nguyên máy tính có thể mở rộng theo yêu cầu mà Azure cung cấp. Máy ảo có thể được triển khai theo nhiều phương thức như sử dụng giao diện người dùng trong cổng thông tin Azure, sử dụng bản mẫu ứng dụng được định sẵn trong Azure marketplace, viết mẫu lệnh ( script ) thông qua Azure Powershell, triển khai từ mẫu thiết kế định nghĩa bằng tệp tin JSON hoặc thẳng từ Visual Studio.

thông thường, người dùng chọn VM khi họ cần sự kiểm soát lớn hơn đối với môi trường máy tính so với các lựa chọn khác cung cấp. Máy ảo Azure cung cấp cho họ sự linh hoạt của ảo hóa mà không phải mua và bảo trì phần cứng vật lý chạy nó. Tuy nhiên, người dùng vẫn cần duy trì VM bằng cách thực hiện các tác vụ, chẳng hạn như định cấu hình, vá lỗi và cài đặt phần mềm chạy trên nó.

Máy ảo Azure có thể được sử dụng theo nhiều cách như là :

* **Phát triển và thử nghiệm** - Máy ảo Azure cung cấp một cách nhanh chóng và dễ dàng để tạo ra một máy tính có cấu hình cụ thể cần thiết để code và kiểm tra một ứng dụng.
* **Các ứng dụng trên đám mây** - Vì nhu cầu cho ứng dụng của người dùng có thể dao động, nên có thể có ý nghĩa kinh tế khi chạy nó trên máy ảo trong Azure. Các doanh nghiệp trả tiền cho các máy ảo bổ sung khi cần và tắt chúng khi họ không cần đến chúng nữa.
* **Trung tâm dữ liệu mở rộng** - Các máy ảo trong mạng ảo Azure có thể dễ dàng được kết nối với mạng tổ chức của công ty

3.4 Lưu trữ trên Microsoft Azure

Azure storage là một dịch vụ quản lý nhằm mục đích cung cấp giải pháp cho lưu trữ đám mây cho các dữ liệu hiện nay. Azure storage cung cấp các kho lưu trữ dữ liệu là các đối tượng khác nhau: kho các file dữ liệu cho hệ thống file, kho lưu trữ các tin nhắm cho các kênh tin nhắn bảo mật, kho lưu trữ dữ liệu NoSQL.

Các đặc điểm nổi bật của Azure storage là :

**Độ bền và có sẵn cao** : Các bản dự phòng đảm bảo rằng dữ liệu của người dùng an toàn trong trường hợp gặp lỗi phần cứng. họ cũng có thể chọn sao chép dữ liệu trên các trung tâm dữ liệu hoặc khu vực địa lý để bảo vệ thêm khỏi thảm họa cục bộ hoặc thảm họa tự nhiên. Dữ liệu được sao chép theo cách này vẫn sẵn sằng trong trường hợp mất điện đột xuất.

**Đảm bảo** : Tất cả dữ liệu được ghi vào Azure Storage được mã hóa bởi dịch vụ. Azure Storage cung cấp cho người dùng quyền kiểm soát chi tiết đối với những người có quyền truy cập vào dữ liệu của họ.

**Có thể mở rộng** : Azure Storage được thiết kế để có thể mở rộng để đáp ứng nhu cầu lưu trữ dữ liệu và hiệu năng của các ứng dụng ngày nay.

**Quản lý** : Microsoft Azure xử lý bảo trì phần cứng, cập nhật và các sự cố quan trọng cho người sử dụng.

**Truy cập rộng rãi :** Dữ liệu trong Azure Storage có thể truy cập từ mọi nơi trên thế giới qua HTTP hoặc HTTPS. Microsoft cung cấp SDK cho Azure Storage bằng nhiều ngôn ngữ -.NET, Java, Node.js, Python, PHP, Ruby, Go và các ngôn ngữ khác - cũng như REST API. Azure Storage hỗ trợ tập lệnh trong Azure PowerShell hoặc Azure CLI. Cổng thông tin Azure và Azure Storage Explorer cung cấp các giải pháp trực quan để dễ dàng làm việc với dữ liệu của người dùng.

Azure cung cấp cho người dùng nhiều loại dịch vụ lưu trữ khác nhau và mỗi loại được tối ưu cho mỗi trường hợp sử dụng khác nhau. Trong đó có 4 loại chính là :

**Blob storage :** Azure Blob storage là giải pháp lưu trữ đối tượng của Microsoft cho đám mây. Blob storage được tối ưu hóa để lưu trữ một lượng lớn dữ liệu phi cấu trúc, chẳng hạn như dữ liệu văn bản hoặc nhị phân. Blog storage thường được sử dụng để :

* Đưa hình ảnh hoặc tài liệu trực tiếp tới trình duyệt.
* Lưu trữ các tập tin để truy cập phân tán.
* Truyền phát video và âm thanh.
* Lưu trữ dữ liệu để sao lưu và khôi phục, khôi phục thảm họa và lưu kho trữ.
* Lưu trữ dữ liệu cho việc phân tích bởi một dịch vụ tại chỗ hoặc dịch vụ lưu trữ Azure.

Các đối tượng trong Blob storage có thể được truy cập từ mọi nơi trên thế giới thông qua HTTP hoặc HTTPS. Người dùng hoặc ứng dụng khách có thể truy cập các blob thông qua URL, API API lưu trữ Azure, Azure PowerShell, Azure CLI hoặc thư viện máy khách Azure Storage. Các thư viện máy khách lưu trữ có sẵn cho nhiều ngôn ngữ, bao gồm.NET, Java, Node.js, Python, PHP và Ruby.

**File storage :** Azure file cho phép người dùng thiết lập chia sẻ tệp mạng mang tính khả dụng cao,có thể được truy cập bằng cách sử dụng giao thức khối thông điệp máy chủ (Server message Block – SMB ) tiêu chuẩn. Điều đó có nghĩa là nhiều VM có thể chia sẻ cùng một tệp với cả quyền truy cập đọc và ghi. Họ cũng có thể đọc các tệp bằng giao diện REST hoặc các thư viện máy khách lưu trữ.

Một điều khác biệt giữa Azure file với các tệp khác trên hệ thống chia sẻ tệp của công ty là người dùng có thể truy cập các tệp từ bất kỳ nơi nào trên thế giới bằng cách sử dụng URL trỏ đến tệp và bao gồm thẻ truy cập được chia sẻ (Shared Access Signature - SAS).

Azure file được sử dụng trong một số trường hợp sau :

* Nhiều ứng dụng tại chỗ sử dụng chia sẻ tập tin. Tính năng này giúp việc di chuyển các ứng dụng chia sẻ dữ liệu sang Azure dễ dàng hơn. Nếu người dùng đưa chia sẻ tệp vào cùng một ký tự ổ đĩa mà ứng dụng tại chỗ sử dụng, phần ứng dụng của họ truy cập vào chia sẻ tệp sẽ hoạt động với mức tối thiểu, nếu có bất kì thay đổi gì.
* Các tệp cấu hình có thể được lưu trữ trên một tệp chia sẻ và được truy cập từ nhiều VM. Các công cụ và tiện ích được sử dụng bởi nhiều nhà phát triển trong một nhóm có thể được lưu trữ trên một tệp chia sẻ, đảm bảo mọi người đều có thể tìm thấy chúng và họ sử dụng cùng một phiên bản.
* Nhật ký chẩn đoán, số liệu và kết xuất sự cố chỉ là một số ví dụ về những dữ liệu có thể được ghi vào chia sẻ tệp và được xử lý hoặc phân tích sau đó.

**Queue storage :** Azure queue storage là một dịch vụ để lưu trữ số lượng lớn thông điệp có thể được truy cập từ mọi nơi trên thế giới thông qua các cuộc gọi được xác thực bằng HTTP hoặc HTTPS. Một tin nhắn hàng đợi có thể có kích thước lên tới 64 KB và một hàng đợi có thể chứa hàng triệu tin nhắn, tối đa tổng giới hạn dung lượng của tài khoản lưu trữ.

**Table storage :** Azure table storage là một dịch vụ lưu trữ dữ liệu NoQuery có cấu trúc trong đám mây, cung cấp kho lưu trữ khóa / thuộc tính với thiết kế phi lược đồ. Bởi vì table storage là phi lược đồ, nên nó rất dễ dàng để điều chỉnh dữ liệu của người dùng khi nhu cầu của ứng dụng của họ phát triển. Truy cập vào dữ liệu trong Table storage sẽ nhanh chóng và hiệu quả về chi phí hơn đối với nhiều loại ứng dụng và thường có chi phí thấp hơn so với SQL truyền thống đối với khối lượng dữ liệu tương tự.

Người dùng có thể sử dụng Table storage để lưu trữ các bộ dữ liệu linh hoạt như dữ liệu các khách hàng cho các ứng dụng web, sổ địa chỉ, thông tin thiết bị hoặc các loại siêu dữ liệu khác mà dịch vụ của người dùng yêu cầu. Họ có thể lưu trữ bất kỳ số lượng thực thể nào trong một bảng và tài khoản lưu trữ có thể chứa bất kỳ số lượng bảng nào, tối đa giới hạn dung lượng của tài khoản lưu trữ.

Một số cách dùng của Table storage là :

* Lưu trữ TB dung lượng dữ liệu có cấu trúc có khả năng phục vụ các ứng dụng quy mô web.
* Lưu trữ các bộ dữ liệu không yêu cầu các phép nối phức tạp, khóa ngoại hoặc các thủ tục được lưu trữ và có thể được chuẩn hóa để truy cập nhanh
* Truy vấn nhanh dữ liệu bằng chỉ mục nhóm.
* Truy cập dữ liệu bằng giao thức OData và truy vấn LINQ với Thư viện dịch vụ dữ liệu WCF.NET.

Để lưu trữ, người dùng cần phải tạo một storage account, mỗi account có thể lưu trữ tới 500TB dữ liệu trên đám mây. Microsoft Azure có một số loại storage account, mỗi loại hỗ trợ các tính năng khác nhau và có mô hình định giá riêng. Storage account được chia làm 2 loại chính :

* **General storage account :** Tài khoản lưu trữ đa năng cung cấp một không gian để người dùng có thể truy cập vào các blob, hàng đợi, tệp và bảng, tất cả các dịch vụ này trong một tài khoản hợp nhất. Tài khoản lưu trữ đa năng có thể được sử dụng để lưu trữ dữ liệu đối tượng, có thể được sử dụng làm kho lưu trữ dữ liệu NoQuery, có thể được sử dụng để xác định và sử dụng hàng đợi để xử lý thư và thiết lập chia sẻ tệp trong đám mây.
* **Blob storage account :** Tài khoản lưu trữ Blob chuyên lưu trữ dữ liệu blob và cũng có thể được sử dụng để chọn bậc truy cập, cho phép người dùng chỉ định tần suất dữ liệu được truy cập trong tài khoản. Người dùng có thể chọn một cấp truy cập phù hợp với lưu trữ của họ và phù hợp với chi phí của họ.

Có 2 bậc truy cập trong tài khoản lưu trữ blob :

* **Nóng ( hot ) :** Tầng truy cập này cấp cho người dùng độ trễ thấp nhất có thể. Do đó, nó nên được sử dụng với dữ liệu thường xuyên được truy cập. Đương nhiên, vì nó cung cấp độ trễ thấp, nó có giá thành cao hơn.
* **Lạnh ( cold ) :** Tầng truy cập này có hiệu suất thấp hơn tầng truy cập của Hot. Nó cung cấp độ trễ cao hơn tầng truy cập Hot. Điều đó đang được nói, nó đi kèm với một giá thành thấp hơn và do đó có thể được sử dụng cho dữ liệu ít được truy cập.

Để đảm bảo dữ liệu của người dùng không bị thất thoát, Azure Storage sao chép nhiều bản sao dữ liệu của người dùng. Khi họ thiết lập tài khoản lưu trữ, họ có thể chọn tùy chọn dự phòng.

Các Tùy chọn sao chép cho tài khoản lưu trữ bao gồm:

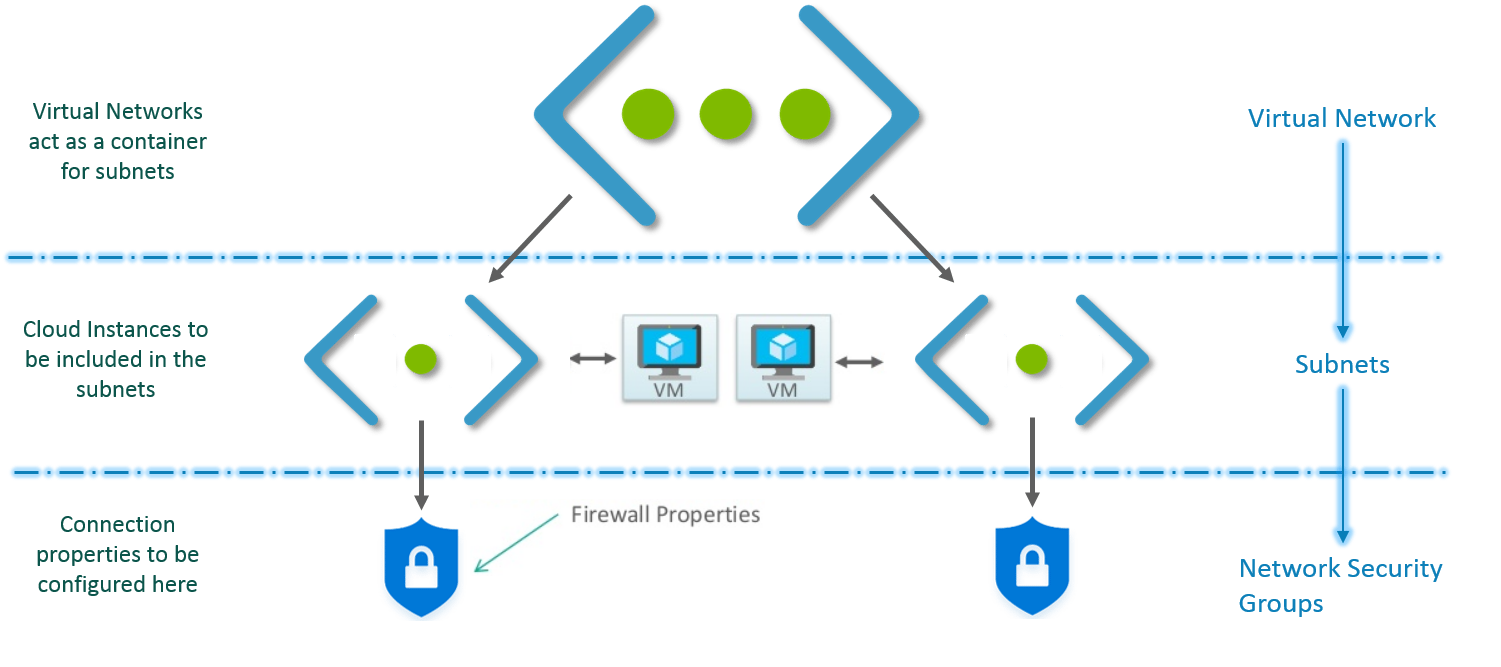
* [**Locally-redundant storage (LRS)**](https://docs.microsoft.com/en-us/azure/storage/common/storage-redundancy-lrs) : LRS sao chép dữ liệu của người dùng ba lần trong một đơn vị tỷ lệ lưu trữ, tức là bên trong một trung tâm dữ liệu. Trung tâm dữ liệu nằm trong khu vực nơi người dùng đã tạo tài khoản lưu trữ của mình. Một yêu cầu write chỉ trả về thành công khi nó đã được write cho cả ba bản sao. Mỗi bản sao này nằm trong các miền lỗi riêng biệt và các miền nâng cấp trong một đơn vị tỷ lệ lưu trữ.
* **Zone-Redundant Storage (ZRS)** : sao chép dữ liệu của người dùng một cách không đồng bộ trên các trung tâm dữ liệu trong một hoặc hai vùng ngoài việc lưu trữ ba bản sao tương tự LRS, do đó cung cấp độ bền cao hơn LRS. Dữ liệu được lưu trữ trong ZRS là bền ngay cả khi trung tâm dữ liệu chính không có sẵn hoặc không thể phục hồi.
* **Geo-redundant storage (GRS)** : sao chép dữ liệu của người dùng vào một khu vực mà cách hàng trăm dặm từ vùng chính. Nếu tài khoản lưu trữ của người dùng đã bật GRS, thì dữ liệu của họ sẽ bền ngay cả trong trường hợp mất điện hoàn toàn trong khu vực hoặc thảm họa trong đó khu vực chính không thể phục hồi được.
* **Read-access geo-redundant storage (RA-GRS)** : tối đa hóa khả năng khả dụng cho tài khoản lưu trữ của người dùng, bằng cách cung cấp quyền truy cập chỉ đọc vào dữ liệu ở vị trí thứ cấp, ngoài việc sao chép qua hai vùng do GRS cung cấp.

3.5 Virtual network trong Microsoft Azure:

Thiết lập mạng là một thành phần quan trọng của bất kỳ cơ sở hạ tầng đám mây. Các bộ phận IT cần kết nối các tài nguyên và tối ưu hóa hiệu suất ứng dụng - tất cả đều phụ thuộc vào kiến trúc mạng vững chắc.Các dịch vụ mạng Microsoft Azure cung cấp nhiều khả năng khác nhau để kết nối và quản lý tài nguyên đám mây. Một trong số đó là mạng ảo (Virtual Network - VNet).

Mạng ảo Azure về cơ bản cho phép người dùng tạo một bản sao riêng biệt như mạng của họ vào trong đám mây Azure. Nó được gọi là “ảo” vì nó không dựa vào bộ định tuyến và chuyển mạch thực tế để cung cấp vật trung gian cho máy chủ của người dùng để tương tác. Bản thân VNet là vật trung gian cung cấp cho người dùng kết nối riêng giữa các máy ảo, các thiết bị được nối mạng khác và một số dịch vụ Azure. Một mạng ảo được đặt trong một vùng Azure. Vùng Azure là một tập hợp các trung tâm dữ liệu được triển khai trong phạm vi có độ trễ đã được xác định và được kết nối thông qua mạng có độ trễ thấp dành riêng cho khu vực.

Cũng như với AWS VPC (Amazon Web Service Virtual Private Cloud), Azure Virtual Network cũng có thể cung cấp một loạt các tính năng mạng khác, bao gồm khả năng tùy chỉnh các khối DHCP, DNS, định tuyến, kết nối liên VM, kiểm soát truy cập và Mạng riêng ảo (VPN). Mạng ảo Azure cho phép nhiều loại tài nguyên Azure, chẳng hạn như Máy ảo Azure (VM), giao tiếp an toàn với nhau, internet và mạng tại chỗ (On-premises network ). Mạng ảo được tạo thành từ các mạng con. Mạng con là một dải địa chỉ IP trong mạng ảo. Các mạng con, giống như các mạng ảo, nằm trong một khu vực Azure duy nhất.



Hình 3.5 Cấu trúc Mạng ảo và subnet của Azure

Nguồn : www.edureka.co

Mạng ảo Azure cung cấp các khả năng chính sau :

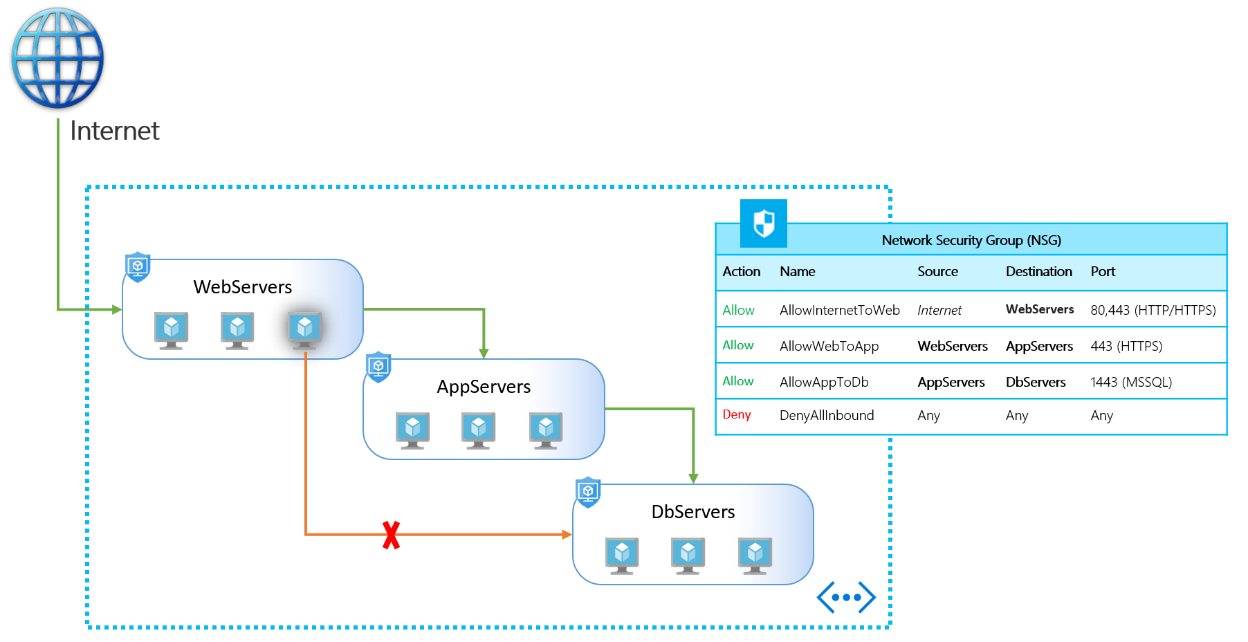
* **Cách ly và phân khúc :**

Người dùng có thể triển khai nhiều mạng ảo trong mỗi thuê bao Azure và vùng Azure. Mỗi mạng ảo được cách ly với các mạng ảo khác. Đối với mỗi mạng ảo, họ có thể:

* + Chỉ định không gian địa chỉ IP riêng bằng địa chỉ công cộng và địa chỉ riêng (RFC 1918). Azure gán tài nguyên trong mạng ảo một địa chỉ IP riêng từ không gian địa chỉ mà người dùng gán.
  + Phân đoạn mạng ảo thành một hoặc nhiều mạng con và phân bổ một phần không gian địa chỉ của mạng ảo cho mỗi mạng con.
  + Sử dụng nghị quyết tên do Azure cung cấp hoặc chỉ định máy chủ DNS của riêng người dùng, để sử dụng bởi các tài nguyên trong mạng ảo.
* **Giao tiếp với internet :**

Theo mặc định, tất cả các tài nguyên trong một mạng ảo có thể giao tiếp với internet. Người dùng có thể giao tiếp inbound đến một tài nguyên bằng cách chỉ định một địa chỉ IP công cộng hoặc bộ cân bằng tải công khai.

* **Giao tiếp giữa các tài nguyên Azure :**
  + Tài nguyên Azure liên lạc an toàn với nhau theo các cách sau :
  + Thông qua mạng ảo.
  + Thông qua một dịch vụ đầu cuối mạng ảo



Hình 3.6 Các cụm máy ảo liên lạc với nhau trong mạng ảo

Nguồn : azure.microsoft.com

* **Giao tiếp với các tài nguyên tại chỗ :**

Người dùng có thể kết nối máy tính và mạng tại chỗ của mình với mạng ảo bằng bất kỳ kết hợp nào trong các tùy chọn sau:

* **Mạng riêng ảo point-to-site (VPN):** Được thiết lập giữa một mạng ảo và một máy tính trong mạng của người dùng. Mỗi máy tính muốn thiết lập kết nối với mạng ảo phải cấu hình kết nối của nó.
* **Site-to-site VPN**: Được thiết lập giữa thiết bị VPN tại chỗ của người dùng và cổng VPN Azure được triển khai trong mạng ảo.
* **Azure ExpressRoute**: thiết lập giữa mạng của người dùng và Azure, thông qua đối tác ExpressRoute.
* **Lọc lưu lượng mạng :**

Người dùng có thể lọc lưu lượng mạng giữa các mạng con bằng các cách sau :

* **Nhóm bảo mật**: Nhóm bảo mật mạng và nhóm bảo mật ứng dụng có thể chứa nhiều quy tắc bảo mật inbound và outbound cho phép người dùng lọc lưu lượng đến và từ tài nguyên theo địa chỉ IP nguồn và đích, cổng và giao thức.
* **Thiết bị ảo mạng:** Thiết bị ảo mạng là máy ảo thực hiện chức năng mạng, như tường lửa, tối ưu hóa mạng WAN hoặc các chức năng mạng khác.

**TÀI LIỆU THAM KHẢO**

**Tiếng Việt**

1. Git là gì – Những khái niệm cơ bản khi làm việc trên Git,07/06/2019, <https://csc.edu.vn/lap-trinh-va-csdl/tin-tuc/kien-thuc-lap-trinh/Git-la-gi--Nhung-khai-niem-co-ban-khi-lam-viec-tren-Git-4133>
2. Nguyen Hong Son , Continuous Integration with Jenkins - Bài 1: Giới thiệu về CI và Jenkins , 5/5/2017 , <https://viblo.asia/p/continuous-integration-with-jenkins-bai-1-gioi-thieu-ve-ci-va-jenkins-OeVKBggEZkW>
3. Tran Van Luan , Giới thiệu về Jenkins: Một Open Source Continuous Integration Server , 26/1/2016 , <https://viblo.asia/p/gioi-thieu-ve-jenkins-mot-open-source-continuous-integration-server-zb7vDV0OMjKd>
4. Chaupm , Jenkins là gì? Mô hình kiểm tra code tự động nổi tiếng , 22/12/2018 , <https://tech.vccloud.vn/jenkins-la-gi-mo-hinh-kiem-tra-code-tu-dong-noi-tieng-2018122210502845.htm>
5. Phung The Tai , Tìm hiểu về Microsoft Azure: Azure Storage phần 1 , 23/8/2018 , <https://viblo.asia/p/tim-hieu-ve-microsoft-azure-azure-storage-phan-1-maGK7j0M5j2>

**Tiếng Anh**

1. GitHubHelp, Enterprise Server 2.17, GitHub Glossary, <https://help.github.com/en/enterprise/2.17/user/articles/github-glossary#contributor>
2. GitGuildes, Hello World, 07/04/2016, <https://guides.github.com/activities/hello-world/>
3. Kinsta Ltd, What is Git Hub? A Beginner’s Introduction to GitHub, 20/08/2018, <https://kinsta.com/knowledgebase/what-is-github/>
4. Martin Heller , What is Jenkins? The CI server explained , 5/12/2017, <https://www.infoworld.com/article/3239666/what-is-jenkins-the-ci-server-explained.html>
5. Bugfender , What Is Jenkins and Why Should You Be Using It? , 21/11/2017 , <https://dev.to/bugfenderapp/what-is-jenkins-and-why-should-you-be-using-it-2pe>
6. Saurabh , Jenkins Tutorial | Continuous Integration Using Jenkins | Edureka , 22/5/2019 , <https://www.edureka.co/blog/jenkins-tutorial/>
7. Saurabh , What is Jenkins? | Jenkins For Continuous Integration | Edureka , 22/5/2019 , <https://www.edureka.co/blog/what-is-jenkins/>
8. Margaret Rouse , Microsoft Azure (Windows Azure) , 8/2018 , <https://searchcloudcomputing.techtarget.com/definition/Windows-Azure>
9. Nicole Shortslef , Microsoft Azure Explained: What It Is and Why It Matters , <https://ccbtechnology.com/what-microsoft-azure-is-and-why-it-matters/>
10. Sumo Logic , What is Microsoft Azure and Why Use It ? , 16/9/2016 , <https://www.sumologic.com/brief/what-is-azure/>
11. Ryan Oistacher , Azure Cloud Services - A Complete Guide , 27/11/2018 , <https://blog.5nine.com/what-is-microsoft-azure>
12. Hemant Sharma , What is Azure? – An Introduction To Microsoft Azure Cloud , 22/5/2019 , <https://www.edureka.co/blog/what-is-azure/>
13. Nicola Wright , Microsoft Azure: virtual machines for the confused , <https://www.nigelfrank.com/blog/microsoft-azure-virtual-machines-for-the-confused/>
14. Duan van der Westhuizen , Explaining Microsoft Azure Virtual Machines to a Windows SysAdmin , 24/5/2017 , <https://blog.rackspace.com/explaining-microsoft-azure-virtual-machines-to-a-windows-sysadmin>
15. Hemant Sharma , Microsoft Azure Tutorial – Cloud Computing With Azure , 28/5/2019 , <https://www.edureka.co/blog/microsoft-azure-tutorial>
16. Hemant Sharma , Azure Storage Tutorial – Tables, Blobs, Queues & File Storage in Microsoft Azure , 22/5/2019 , <https://www.edureka.co/blog/azure-storage-tutorial/>
17. Russell Smith , Understanding Azure Storage: Storage Types and Redundancy , 17/3/2017 , <https://www.petri.com/understanding-azure-storage-storage-types-redundancy>
18. Chandan Patra , Azure Storage: Service Overview and Introduction , 11/11/2015 , <https://cloudacademy.com/blog/azure-storage-service-overview/>
19. Denis G , Microsoft Azure Storage Types Explained , 3/10/2017 , <https://www.cloudberrylab.com/resources/blog/microsoft-azure-storage-types-explained/>
20. Minette Steynberg , Different Azure Storage types (File, Blob, Queue and Table) , 3/5/2017 , <https://www.sqlshack.com/different-azure-storage-types-file-blob-queue-table/>
21. Hemant Sharma , Azure Virtual Network For Beginners – Securing Your Applications Using VPC , 22/5/2019 , <https://www.edureka.co/blog/azure-virtual-network-tutorial/>
22. Kathleen Casey , Words to go: Microsoft Azure networking services , 9-2017 , <https://searchcloudcomputing.techtarget.com/feature/Words-to-go-Microsoft-Azure-networking-services>
23. Edward Smith , Securing Azure Virtual Network , 28/2/2019 , <https://blog.cloudpassage.com/2019/02/28/securing-azure-virtual-network/>
24. <https://www.guru99.com/jenkin-continuous-integration.html>
25. <https://en.wikipedia.org/wiki/Jenkins_(software)>
26. <https://jenkins.io/doc/pipeline/tour/getting-started/>
27. <https://www.quora.com/What-is-Jenkins-When-and-why-is-it-used>
28. <https://www.quora.com/What-is-Microsoft-Azure-and-its-uses>
29. <https://en.wikipedia.org/wiki/Microsoft_Azure>

<https://www.tutorialspoint.com/microsoft_azure/microsoft_azure_storage.htm>