TRƯỜNG ĐẠI HỌC SƯ PHẠM THÀNH PHỐ HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO CUỐI KỲ

NHẬP MÔN DEV-OPS

ĐỀ TÀI: NGHIÊN CỨU VÀ TRIỂN KHAI TRÊN AWS ELASTIC BEANSTALK, CODE PIPELINE, EC2 SERVER

Sinh viên thực hiện	*Trịnh Hoàng Tùng*	46.01.104.211
	Nguyễn Ngọc Như Ý	46.01.104.225
	Nguyễn Minh Phát	43.01.104.125
	Nguyễn Trọng Hậu	46.01.104.050
	Bùi Thị Ánh Tuyết	46.01.104.214

Giảng viên hướng dẫn *Thầy Mai Văn Phương Vũ*

Chuyên ngành Công nghệ phần mềm

Lớp học phần COMP170401

THÀNH PHỐ HỒ CHÍ MINH, THÁNG 11 NĂM 2022

MỤC LỤC

LỜI MỞ ĐẦU

// viết ở đây

CHƯƠNG 1. MỞ ĐẦU VỚI DEV-OPS:

1.1. Dev-Ops là gì:

- DevOps là sự kết hợp giữa nhiều *triết lý văn hóa, biện pháp thực hành* và *công cụ* giúp tăng khả năng phân phối ứng dụng và dịch vụ của một tổ chức ở tốc độ cao

+ Phát triển và cải tiến sản phẩm ở nhịp độ nhanh hơn các tổ chức sử dụng quy trình quản lý cơ sở hạ tầng và phát triển phần mềm truyền thống.

+ Với tốc độ này thì cho phép các tổ chức phục vụ khách hàng tốt hơn và cạnh tranh hiệu quả hơn trên thị trường

1.1.1. Triết lý văn hóa của DevOps:

- Quá trình chuyển tiếp sang DevOps cần thay đổi về văn hóa và tư duy. Đơn giản nhất, mục đích của DevOps là xóa bỏ rào cản giữa hai nhóm phát triển và nghiệp vụ, thường được tổ chức theo mô hình tách biệt. Thậm chí một số tổ chức có thể không có các nhóm phát triển và nghiệp vụ riêng; các kỹ sư sẽ đảm nhiệm cả hai mảng. Với DevOps, hai nhóm làm việc cùng nhau để tối ưu hóa cả năng suất của nhà phát triển lẫn độ tin cậy của hoạt động nghiệp vụ. Họ nỗ lực giao tiếp thường xuyên, tăng hiệu suất và nâng cao chất lượng dịch vụ cung cấp cho khách hàng. Các nhà phát triển hoàn toàn làm chủ dịch vụ của mình, thường là vượt trên vai trò hoặc chức vụ đã định theo truyền thống đã được khoanh vùng rõ bằng cách cân nhắc về nhu cầu của người dùng cuối và cách thức họ có thể tham gia giải quyết các nhu cầu đó. Các nhóm đảm bảo chất lượng và bảo mật cũng có thể phối hợp chặt chẽ với các nhóm trên. Các tổ chức sử dụng mô hình DevOps, không phụ thuộc vào cấu trúc tổ chức, có các nhóm coi toàn bộ quá trình phát triển và vòng đời của cơ sở hạ tầng là một phần trách nhiệm của mình.

1.1.2. Biện pháp thực hành DevOps:

- Sau đây là các biện pháp thực hành DevOps:

+ Tích hợp liên tục

+ Phân phối liên tục

+ Vi dịch vụ

+ Cơ sở hạ tầng dưới dạng mã

+ Giám sát và ghi nhật ký

+ Giao tiếp và cộng tác

- *Tích hợp liên tục*

Tích hợp liên tục là biện pháp phát triển phần mềm trong đó các nhà phát triển thường xuyên hợp nhất các thay đổi mã vào kho lưu trữ trung tâm, sau đó là chạy các bản dựng và kiểm thử tự động. Mục tiêu chính của tích hợp liên tục là tìm và khắc phục lỗi nhanh hơn, cải thiện chất lượng phần mềm và giảm thời gian bỏ ra để thẩm định và phát hành các bản cập nhật mới.

- *Phân phối liên tục*

Phân phối liên tục là biện pháp phát triển phần mềm trong đó các thay đổi mã được tự động xây dựng, kiểm thử và chuẩn bị phát hành vào quá trình sản xuất. Biện pháp thực hành này mở rộng dựa trên quá trình tích hợp liên tục bằng cách triển khai tất cả các thay đổi mã vào môi trường kiểm thử và/hoặc môi trường sản xuất sau công đoạn xây dựng. Khi phân phối liên tục được thực hiện phù hợp, các nhà phát triển sẽ luôn có một thành phần lạ trong bản dựng sẵn sàng triển khai đã vượt qua quy trình kiểm thử được tiêu chuẩn hóa.

- *Vi dịch vụ*

Kiến trúc vi dịch vụ là phương pháp thiết kế để xây dựng một ứng dụng đơn lẻ dưới dạng một tập hợp các dịch vụ nhỏ. Mỗi dịch vụ chạy trong một quy trình riêng và giao tiếp với các dịch vụ khác thông qua một giao diện được xác định rõ sử dụng một cơ chế gọn nhẹ, điển hình là một giao diện lập trình ứng dụng (API) dựa trên HTTP. Vi dịch vụ được xây dựng xoay quanh các chức năng của doanh nghiệp, mỗi dịch vụ được xác định phạm vi cho một mục đích đơn lẻ. Bạn có thể sử dụng các framework hoặc ngôn ngữ lập trình khác để viết các vi dịch vụ và triển khai chúng độc lập, dưới dạng một dịch vụ đơn lẻ hoặc dưới dạng một nhóm các dịch vụ.

- *Cơ sở hạ tầng dưới dạng mã*

Cơ sở hạ tầng dưới dạng mã là biện pháp thực hành trong đó cơ sở hạ tầng được cung cấp và quản lý bằng mã và các kỹ thuật phát triển phần mềm, ví dụ như kiểm soát phiên bản và tích hợp liên tục. Mô hình điều khiển bởi API của đám mây cho phép các nhà phát triển và quản trị viên hệ thống tương tác với cơ sở hạ tầng về mặt lập trình và ở quy mô phù hợp, thay vì phải thiết lập và đặt cấu hình theo cách thủ công cho các tài nguyên. Nhờ đó, các kỹ sư có thể giao tiếp với cơ sở hạ tầng bằng các công cụ dựa trên mã và thao tác với cơ sở hạ tầng theo cách tương tự như cách thao tác với mã ứng dụng. Vì được xác định bằng mã, cơ sở hạ tầng và máy chủ có thể được triển khai nhanh chóng bằng các mẫu hình được tiêu chuẩn hóa, được cập nhật các bản vá và phiên bản mới nhất hoặc được sao chép theo cách có thể lặp lại.

- *Giám sát và ghi nhật ký*

Các tổ chức giám sát số liệu và nhật ký để xem hiệu năng của ứng dụng và cơ sở hạ tầng ảnh hưởng như thế nào đến trải nghiệm của người dùng cuối sản phẩm. Bằng cách nắm bắt, phân loại rồi phân tích dữ liệu và nhật ký được tạo ra bởi các ứng dụng và cơ sở hạ tầng, các tổ chức sẽ hiểu được những thay đổi hoặc các bản cập nhật ảnh hưởng như thế nào đến người dùng, qua đó làm sáng tỏ được những thông tin giúp xác định các nguyên nhân gốc của vấn đề và những thay đổi ngoài dự kiến. Giám sát chủ động đang ngày càng quan trọng khi các dịch vụ phải hoạt động 24/7 và tần suất cập nhật ứng dụng và cơ sở hạ tầng tăng lên. Việc tạo các thông báo hoặc thực hiện phân tích dữ liệu này trong thời gian thực cũng giúp các tổ chức giám sát các dịch vụ của mình một cách chủ động hơn.

- *Giao tiếp và cộng tác*

Tăng cường giao tiếp và cộng tác trong tổ chức là một trong những khía cạnh văn hóa trọng điểm của DevOps. Việc sử dụng bộ công cụ DevOps và tự động hóa quy trình phân phối phần mềm giúp xây dựng sự cộng tác bằng cách kết hợp các quy trình công việc và trách nhiệm của nhóm phát triển và nghiệp vụ lại với nhau trong hoạt động thực tế. Khi xây dựng dựa trên cơ sở như vậy, các nhóm sẽ đặt ra được các chuẩn mực văn hóa bền

vững xoay quanh vấn đề chia sẻ thông tin và thúc đẩy giao tiếp thông qua việc sử dụng các ứng dụng trò chuyện, các hệ thống theo dõi sự cố hay dự án, cũng như các wiki. Điều này giúp đẩy nhanh giao tiếp giữa các nhà phát triển, các hoạt động nghiệp vụ và thậm chí là giữa các nhóm khác như tiếp thị hoặc bán hàng, cho phép tất cả các bộ phận trong tổ chức bám sát với mục tiêu và dự án hơn.

1.1.3. Các công cụ DevOps

Mô hình DevOps phụ thuộc vào bộ công cụ hiệu quả để giúp các nhóm triển khai cũng như cải tiến cho khách hàng một cách nhanh chóng và đáng tin cậy. Các công cụ này tự động hóa các tác vụ thủ công, giúp các nhóm quản lý các môi trường phức tạp ở quy mô phù hợp và giúp các kỹ sư luôn kiểm soát được tốc độ cao mà DevOps mang lại. AWS cung cấp các dịch vụ được thiết kế cho DevOps và được xây dựng trước tiên để sử dụng với đám mây AWS. Các dịch vụ này giúp bạn sử dụng các biện pháp thực hành DevOps được mô tả ở trên.

1.2. Cách thức hoạt động của DevOps

Trong mô hình DevOps, các nhóm phát triển và nghiệp vụ không còn bị "cô lập". Đôi khi, hai nhóm này được hợp nhất thành một nhóm duy nhất, trong đó các kỹ sư làm việc với toàn bộ vòng đời của ứng dụng, từ quá trình phát triển và kiểm thử cho đến triển khai và hoạt động, đồng thời phát triển một loạt các kỹ năng không chỉ giới hạn ở một chức năng đơn lẻ.

Trong một số mô hình DevOps, các nhóm đảm bảo chất lượng và bảo mật cũng có thể được gắn kết chặt chẽ hơn với nhóm phát triển và nghiệp vụ và xuyên suốt vòng đời của ứng dụng.

Các nhóm này áp dụng các biện pháp thực hành để tự động hóa các quy trình mà từ trước đến nay vốn diễn ra theo cách thủ công và chậm chạp. Các nhóm này sử dụng một bộ công nghệ và bộ công cụ giúp họ vận hành và phát triển các ứng dụng một cách nhanh chóng và ổn định.

1.3. Lợi ích của DevOps

- *Về tốc độ*: Hoạt động ở tốc độ cao giúp bạn có thể cải tiến nhanh hơn cho khách hàng, thích ứng tốt hơn với thị trường liên tục thay đổi và tăng trưởng hiệu quả hơn với kết quả kinh doanh ấn tượng. Mô hình DevOps cho phép các nhà phát triển và nhóm nghiệp vụ của bạn đạt được những kết quả này. Ví dụ: Vi dịch vụ và phân phối liên tục cho phép các nhóm làm chủ các dịch vụ và phát hành các bản cập nhật nhanh hơn.

- *Khả năng phân phối nhanh chóng*: Tăng tần suất và nhịp độ phát hành để bạn có thể cải tiến và nâng cấp sản phẩm nhanh hơn. Việc có thể phát hành các tính năng mới và sửa lỗi nhanh hơn đồng nghĩa rằng bạn có thể đáp ứng được các nhu cầu của khách hàng và tạo dựng được lợi thế cạnh tranh sớm hơn. Tích hợp liên tục và phân phối liên tục là các biện pháp thực hành giúp tự động hóa quy trình phát hành phần mềm, từ xây dựng cho đến triển khai.

- *Về mức độ tin cậy*: Đảm bảo chất lượng cho các bản cập nhật ứng dụng và nội dung thay đổi cơ sở hạ tầng để bạn có thể phân phối một cách đáng tin cậy ở nhịp độ nhanh hơn mà vẫn duy trì được trải nghiệm tích cực cho người dùng cuối. Sử dụng các biện pháp thực hành như tích hợp liên tục và phân phối liên tục để kiểm tra rằng từng thay đổi đều hoạt động chính xác và an toàn. Biện pháp thực hành giám sát và ghi nhật ký giúp bạn luôn nhận được thông tin về hiệu năng trong thời gian thực.

- *Về quy mô vận hành và quản lý*: Sự tự động hóa và tính nhất quán giúp bạn quản lý hiệu quả những hệ thống phức tạp hoặc luôn thay đổi ở mức rủi ro được giảm thiểu. Ví dụ: Cơ sở hạ tầng dưới dạng mã giúp bạn quản lý các môi trường phát triển, kiểm thử và sản xuất theo cách thức hiệu quả hơn và có thể lặp lại.

- *Khả năng cải thiện khả năng cộng tác*: Xây dựng các nhóm hiệu quả hơn theo mô hình văn hóa DevOps, giúp nhấn mạnh các giá trị như tinh thần làm chủ và trách nhiệm giải trình. Các nhà phát triển và các nhóm nghiệp vụ cộng tác chặt chẽ với nhau, cùng gánh vác chung nhiều trách nhiệm và phối hợp các quy trình công việc. Điều này giúp giảm thiểu tình trạng kém hiệu quả và tiết kiệm thời gian (ví dụ: giảm thời gian bàn giao giữa nhà phát triển và nhóm nghiệp vụ, viết mã có xem xét tới môi trường hoạt động).

- *Độ bảo mật*: Tiến nhanh hơn nhưng vẫn duy trì kiểm soát và đảm bảo tuân thủ. Bạn có thể áp dụng mô hình DevOps mà không phải giảm bớt tính bảo mật nhờ việc sử dụng các chính sách tuân thủ được tự động hóa, các công cụ kiểm soát được tinh chỉnh và các kỹ thuật quản lý cấu hình. Ví dụ: Khi sử dụng cơ sở hạ tầng dưới dạng mã và chính sách dưới dạng mã, bạn có thể xác định và sau đó theo dõi sự tuân thủ ở quy mô phù hợp.

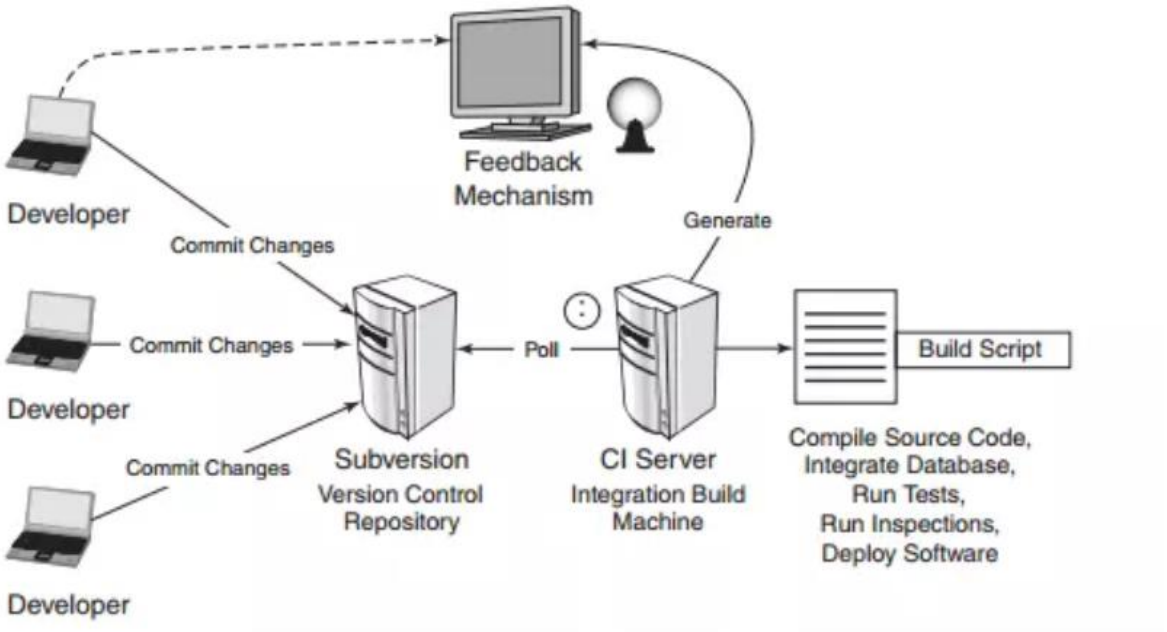
CHƯƠNG 2. QUY TRÌNH PHÁT TRIỂN PHẦN MỀM THEO HƯỚNG TÍCH HỢP LIÊN TỤC VÀ QUY TRÌNH CHUYỂN GIAO PHẦN MỀM LIÊN TỤC (CI – CD)

2.1. Quy trình tích hợp liên tục - Continuous Integration (CI):

- Đây là thuật ngữ được Grady Booch lần đầu tiên đặt ra và đề nghị vào năm 1991, trong bối cảnh mà ngành công nghệ phần mềm cần có một quy trình mới nhằm đẩy mạnh tiến độ sản xuất phần mềm

- Trong kỹ thuật phần mềm, Tích hợp liên tục là việc trộn và biên dịch tất cả các phiên bản mã nguồn làm việc của các lập trình viên trên một bản chính thường xuyên

- Hệ thống kiểm soát phiên bản mã nguồn là mấu chốt của quy trình này, hệ thống này cũng được vận hành để kiểm tra chất lượng mã, kiểm tra lỗi, là công cụ xem xét kiểu cú pháp.

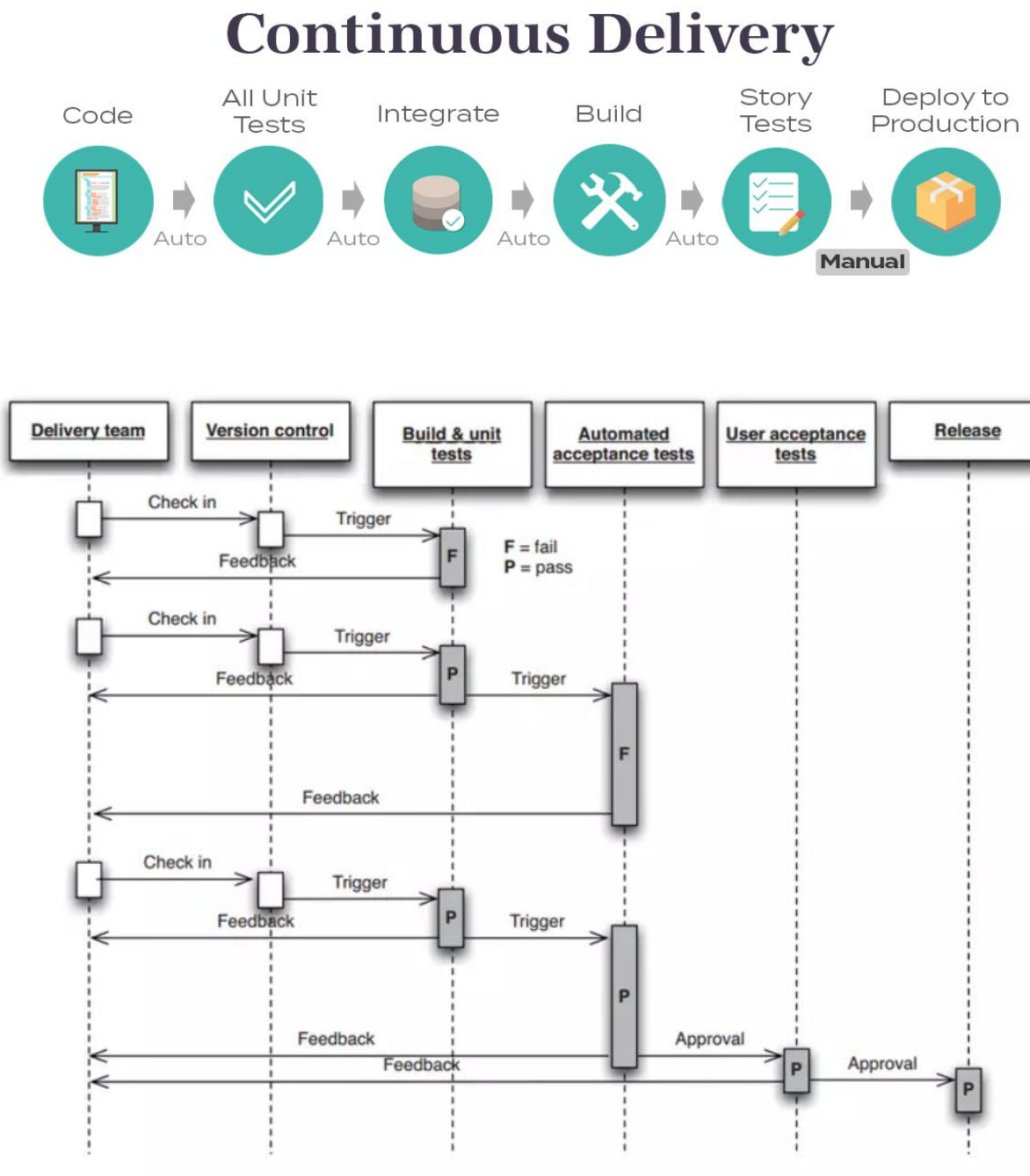


Hình 2.1. Chu trình CI

2.2. Quy trình chuyển giao liên tục – Continuous Delivery (CD):

- Trong khi CI là quy trình để xây dựng và kiểm tra tự động, thì CD lại nâng cao hơn một chút, bằng cách triển khai tất cả thay đổi về mã (đã được xây dựng và kiểm tra) đến môi trường kiểm thử. CD cho phép người lập trình tự động hóa phần kiểm tra phần mềm qua nhiều thước đo trước khi triển khai cho khách hàng. Những bài kiểm tra này có thể kể đến bao gồm kiểm thử giao diện, kiểm thử độ chịu tải, xử lí, API,...

- CD tuyệt đối là yêu cầu cho việc thực hiện triết lý DevOps. Chỉ khi mã nguồn được chuyển giao liên tục, chúng ta mới có thể tự tin rằng những thay đổi từ mã sẽ phục vụ cho khách hàng sau chỉ vài phút với một vài thao tác đơn giản.



Hình 2.2. Chu trình CD

CHƯƠNG 3. DỊCH VỤ AMAZON WEB SERVICE (AWS)

3.1. Tiền đề:

- *Amazon Web Service* (AWS) là một nền tảng điện toán đám mây phát triển toàn diện được cung cấp bởi gã khổng lồ Amazon. Dịch vụ này được hiểu đơn thuần như dịch vụ đám mây hoặc các dịch vụ điện toán từ xa. Các dịch vụ AWS đầu tiên đã được ứng dụng vào năm 2006 để cung cấp các dịch vụ trực tuyến cho các trang web và các ứng dụng phía máy khách. Mục tiêu ban đầu là để giảm thiểu việc bị mất điện đột ngột và đảm bảo tính mạnh mẽ của hệ thống, AWS đa dạng về địa lý theo khu vực.

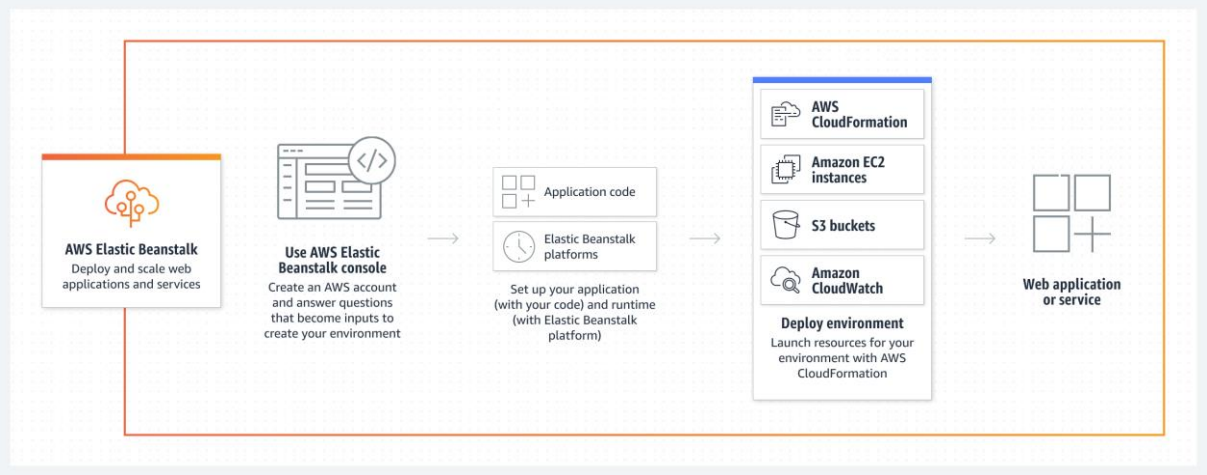
- Hiện nay, các khu vực mà máy chủ AWS đang vận hành có ở Đông Mỹ, Tây Mỹ, Brazil, Ireland, Singapore, Nhật Bản và Úc

3.2. Các dịch vụ được AWS cung cấp:

Có rất nhiều dịch vụ đang được AWS cung cấp cho người dùng sử dụng, phục vụ cho chu trình Dev-Ops. Trong đó được sử dụng nhiều nhất phải kể đến *EC2, DynamoDB, Lambda, IAM, CloudDrive, CloudWatch, Elastic Beanstalk, CodePipeline,...*

3.2.1. Amazon Elastic Beanstalk:

- Amazon Elastic Beanstalk là một dịch vụ điều phối được cung cấp bởi AWS để triển khai các ứng dụng mà các ứng dụng này đang được điều phối bởi các dịch vụ AWS khác nhau, bao gồm *EC2, S3, CodePipeline*,…
- Được giới thiệu như là một dịch vụ giúp tải lên và triển khai các ứng dụng Web một cách nhanh chóng, thuận tiện. Không đòi hỏi về vấn đề quản lý cơ sở hạ tầng, vì mọi thứ sẽ được thực thi tự động

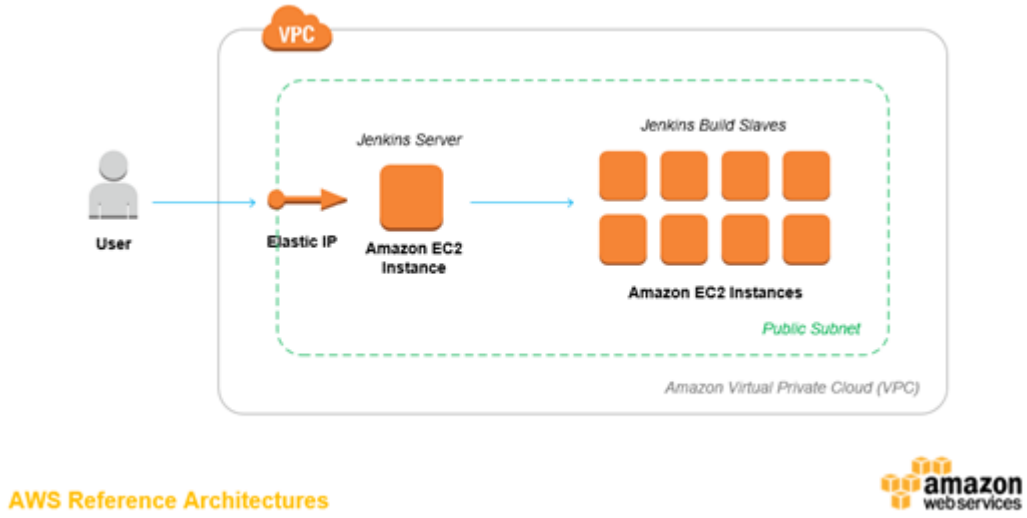


Hình 3.2.1. Mô hình cách hoạt động của Amazon Elastic Beanstalk

3.2.2. Amazon EC2:

- Amazon Elastic Compute Cloud (hay được biết đến là Amazon EC2) là một dịch vụ cung cấp máy chủ đám mây, dịch vụ mang đến nền tảng điện toán sâu rộng nhất, cùng với đó là hơn 500 phiên bản và tuyển tập bộ xử lý, dung lượng lưu trữ, kết nối mạng, hệ điều hành và mô hình mua hàng mới nhất để giúp bạn đáp ứng tốt nhất những nhu cầu về khối lượng công việc của mình.
- Cung cấp điện toán bảo mật cho các ứng dụng của người dùng. Khả năng bảo mật được tích hợp vào nền tảng này luôn được đảm bảo
- Dễ dàng xây dựng, chi chuyển các ứng dụng của người dùng thông qua các dịch vụ khác, phục vụ các quy trình CI/CD[1]

[1] CI - Continuous Integration: phương pháp phát triển phần mềm bằng cách tích hợp thường xuyên. CD – Continuous Delivery: quy trình chuyển giao phần mềm một cách liên tục



Hình 3.2.2. Mô hình cách hoạt động của Amazon EC2

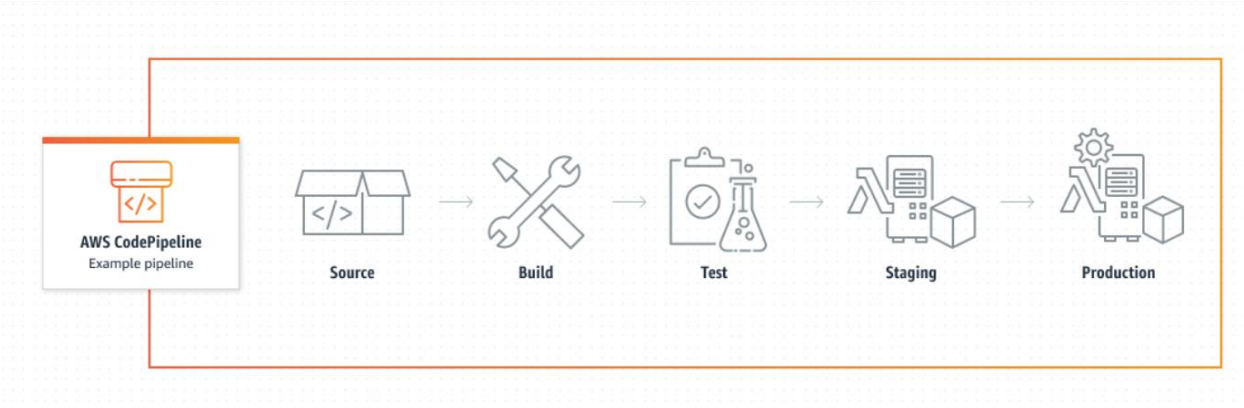
3.2.3. AWS CodePipeline:

- AWS CodePipeline là dịch vụ phân phối liên tục được quản lý hoàn toàn bởi Amazon giúp bạn tự động hóa các quy trình phát hành để cung cấp các bản cập nhật về ứng dụng và cơ sở hạ tầng một cách nhanh chóng và ổn định.

- Tích hợp hệ thống tùy chỉnh: Đăng ký một hành động tùy chỉnh (như thực hiện Git mã nguồn thông qua GitHub) và liên kết máy chủ vào quy trình của bạn bằng cách tích hợp CodePipeline với các máy chủ (như máy chủ Amazon EC2).

- Kiểm soát và cấp quyền truy cập: Quản lý những cá nhân có thể thay đổi và kiểm soát quy trình làm việc phát hành với Quản lý danh tính và truy cập (IAM) trong AWS.

- Nhận thông báo cho các sự kiện: Giám sát các sự kiện tác động đến quy trình bằng Dịch vụ thông báo đơn giản (SNS) của chính Amazon với chức năng cung cấp tin nhắn trạng thái và liên kết đến nguồn sự kiện.



Hình 3.2.3. Mô hình cách hoạt động của Amazon Web Service CodePipeline

CHƯƠNG 4. ỨNG DỤNG – TRIỂN KHAI:

4.1. Triển khai chu trình CI:

4.1.1. Mã nguồn:

- Là một trang Landing Page sử dụng ngôn ngữ PHP, giao diện được thiết kế với HTML, CSS.

- Ngoài ra còn có thêm chức năng lấy dữ liệu API từ OpenWeather để trích xuất dữ liệu thời tiết tùy khu vực.

4.1.2. Thực hiện Git mã nguồn với GitHub:

- Sử dụng GitHub để lưu trữ và quản lý phiên bản của mã nguồn

- Nơi repository[2] được lưu trữ: *https://github.com/TrHgTung/php_app_deploy*

- Khi thực hiện commit vào repository này, các thay đổi, các tính năng mới cũng sẽ được cập nhật trên máy chủ

[2] Repository được hiểu là một kho lưu trữ nơi chứa các tệp tin, thư mục của dự án. Cụ thể thì chúng có thể là các tệp mã, hình ảnh, âm thanh hoặc mọi thứ liên quan đến dự án

4.1.3. Sử dụng Amazon CodePipeline để kết nối với mã nguồn trên GitHub:

4.2. Triển khai chu trình CD:

4.2.1. Cấu hình Elastic Beanstalk:

- Truy cập AWS, tìm đến dịch vụ Elastic Beanstalk và tạo một ứng dụng (application) mới

- Đặt tên cho ứng dụng, kế tiếp đến mục *Platform*: Ở Platform, ta sẽ chọn ngôn ngữ/framework của mã nguồn (Đối với mã nguồn đang được nghiên cứu hiện sử dụng PHP)

- Sau khi thiết lập, nhấn *Create Application* để hoàn tất tạo một ứng dụng Elastic Beanstalk

- Lúc này hãy chờ để AWS xử lý, mọi cấu hình khởi tạo của ứng dụng sẽ được thực hiện một cách tự động

4.2.2. Đơn giản hóa chu trình CI – CD với AWS CodePipeline:

- Truy cập vào bảng điều khiển chính của AWS, tìm dịch vụ CodePipeline

- Tạo mới một Pipeline (nhấn *Create pipeline*)

- Đặt tên cho pipeline, thiết lập tạo role mới (*New service role*), sau đó nhấn Next

- Ở Add *Source Stage*: chọn nơi cung cấp mã nguồn (GitHub) và kết nối với GitHub chứa repository của dự án, sau đó nhấn Next

- Ở Add *Deploy Stage*: chọn nơi để triển khai (AWS Elastic Beanstalk) và chọn ứng dụng Elastic Beanstalk đã tạo, sau đó nhấn Next

- Ở *Review*: kiểm tra thông tin và nhấn Create pipeline để hoàn tất tạo một chu trình CodePipeline

- Lúc này hãy chờ để AWS xử lý, các khởi tạo Pipeline sẽ được thực hiện một cách tự động

4.2.3. Cài đặt máy chủ EC2 để quản lý (instance):

- Truy cập bảng điều khiển của AWS, tìm đến dịch vụ EC2

- Lúc này trong tab *Instances (running)*: Phpappdeploy-env chính là instance của dự án đang chạy trên AWS Beanstalk

- Các tùy chọn ở EC2 cho phép đóng dừng/tiếp tục hoặc loại bỏ hoàn toàn instance, hoặc cấu hình khởi động lại

DANH MỤC HÌNH ẢNH, BẢNG BIỂU