**DevOps and CI/CD**

**นิยามโมเดล DevOps**

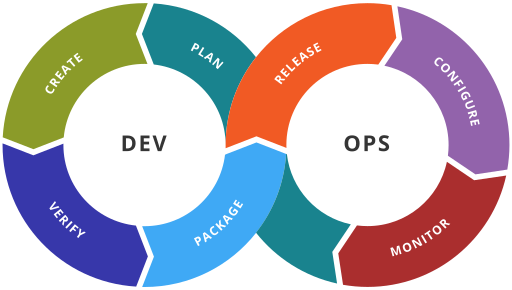
DevOps คือการผสมผสานแนวความคิดเชิงวัฒนธรรม แนวทางปฏิบัติ และเครื่องมือต่างๆ ที่ช่วยเพิ่มความสามารถขององค์กรในการส่งมอบแอปพลิเคชันและบริการอย่างรวดเร็ว โดยพัฒนาและปรับปรุงผลิตภัณฑ์ต่างๆ ให้เร็วกว่ากระบวนการการพัฒนาซอฟต์แวร์และการจัดการโครงสร้างพื้นฐานแบบดั้งเดิม ความรวดเร็วนี้ช่วยให้องค์กรสามารถให้บริการแก่ลูกค้าของตนได้ดีขึ้น และสามารถแข่งขันในตลาดได้อย่างมีประสิทธิภาพมากขึ้น [2]



**ภาพที่ 1** ภาพ Pipeline การทำงาน [2]

**DevOps Process**

มีรูปแบบการทำงานที่ตายตัวเป็นรูปแบบ โดยจะพยายามให้Process ต่างๆ ทำงานไปได้โดยอัตโนมัติ ไม่ต้องมีคนคอยเข้าไปกด Deploy เองอีกต่อไป ยกเว้นในส่วนของการการวางแผน การพัฒนา และดูผลลัพธ์ ในส่วนนี้ยังจำเป็นต้องอาศัยสมองคนต่อไปอยู่ โดยมีกระบวนการตามภาพด้านล่างดังนี้ [2]



**ภาพที่ 2** DevOps Process [3]

Plan

การจะพัฒนาโปรแกรมขึ้นมา ต้องมีการวางแผนร่วมกันจากทุกฝ่ายที่เกี่ยวข้องอย่างละเอียดก่อน โปรแกรมเราจะทำอะไรบ้าง จะพัฒนาด้วยภาษาอะไร Framework อะไร จะรันบน Platform อะไร จะไปลงไว้ที่ไหน จะเก็บข้อมูลอย่างไร จะแก้ปัญหาที่เกิดขึ้น และจะพัฒนา Process การทำงานอย่างไร

Create

การสร้างหรือการพัฒนา นี่ก็คือขั้นตอนของการเขียนโปรแกรมขึ้นมา DevOps เข้ามาช่วยทำให้ Developers ทุกคนทำงานบน Environment แบบเดียวกัน ไม่ว่าเขาจะพัฒนาบน OS ใดก็ตามเวลา Developer ทำงานบนคอมพิวเตอร์คนละ OS กัน มักจะมีปัญหา “ทำไมโปรแกรมรันบนเครื่องเธอไม่ได้ล่ะ บนเครื่องฉันก็รันได้อยู่นะ!” ซึ่งมันไม่ควรจะเกิดขึ้นแล้วในยุคนี้ที่เรามี Docker ที่ช่วยให้ทุกคนพัฒนาโปรแกรมอยู่บนพื้นฐาน Environment เดียวกันเป๊ะ ตั้งแต่ขั้นตอนการพัฒนาไปจนถึงการส่งโปรแกรมขึ้น Production

Verify

การตรวจสอบอยู่อย่างสม่ำเสมอว่าโปรแกรมที่พัฒนาขึ้นมา ไม่ว่าจะเพิ่มฟีเจอร์อะไรเข้ามาจะไม่ไปกระทบหรือไปก่อบั๊กทำให้การใช้งานเดิมพังเสียหายนั้นเป็นเรื่องสำคัญมาก โดยเฉพาะโปรแกรมขนาดใหญ่ที่ร่วมกันทำกับทีมงานจำนวนมาก เราจะมั่นใจได้อย่างไรว่า Code ที่ Developer X ส่งขึ้นมาจะเข้ากันได้ดีกับการทำงานเดิมวิธีการที่จะรู้ได้เร็วที่สุดคือ เราจะต้องมีการทำ Testing อย่างครอบคลุมในทุกๆ Components ของโปรแกรม แล้วทำให้มันรันโดยอัตโนมัติทุกครั้งก่อนจะปล่อยโปรแกรมออกไป เมื่อใดที่ Test Fail ก็จะรู้ได้ทันทีว่ามีปัญหาเกิดขึ้นแล้ว ให้หยุดการส่งงานที่มีปัญหาไปก่อน สามารถเข้าไปหาสาเหตุและแก้ไขปัญหาได้อย่างทันท่วงที

Package

การจะส่งโปรแกรมขึ้นไปรันบน Server เพื่อความสะดวกรวดเร็ว จะต้องมีการเอาโปรแกรมไปใส่ใน Technology ที่ออกแบบมาเพื่อนำโปรแกรมไปรันได้อย่างราบลื่น ซึ่งในปัจจุบันนี้เราใช้ Docker กันเวลาจะใช้ เราจะสร้าง Docker Image ที่เอาตัวโปรแกรมพร้อมสำหรับการรันใส่อยู่ในนั้น แล้วเอา Docker Image นี้ไป Deploy ที่ไหนก็ได้สามารถรัน Docker ได้ โปรแกรมก็จะรันขึ้นมาพร้อมใช้งาน

Release

เมื่อมี Docker Image พร้อมแล้ว ก็พร้อมสำหรับการนำโปรแกรมไปรันบน Deployment Platform ที่ต้องการ ตอนนี้กำลังทำของ Development หรือทำหรือ Production อยู่ จะต้องเลือกไป Deploy ให้ถูกที่

Configure

โปรแกรมที่พัฒนานั้น จำเป็นอย่างมาก ที่จะต้องตั้งค่าได้ ตัวอย่างการตั้งค่าที่ควรมีเลยก็เช่น การเลือก Database ตั้งค่า Email เป็นต้นยิ่งถ้ามี Service แยกอื่นๆ ที่ต้องใช้แล้ว ยิ่งสำคัญ เพราะ Development Server และ Production Server ควรอยู่แยกกันอย่างเด็ดขาด เราจะต้องหาทางสร้าง Docker Image มา 1 อัน แล้วเราจะใช้มัน Deploy ลงบน Environments ที่ต่างกันให้ได้

วิธีการที่จะทำได้ก็คือการทำให้โปรแกรมของเราตั้งค่าได้เนี่ยแหละ คือโปรแกรมที่ไปรันเป็นตัวเดียวกัน แต่เปิดด้วยการตั้งค่าที่แตกต่างกัน ทำให้ได้การทำงานที่แตกต่างกัน โดยปรกติจะใช้ไฟล์ .env และ Environment Variable ในการตั้งค่ากันเป็นมาตรฐานที่ใช้กันบนทุก Platform

Monitor

สุดท้ายที่ขาดไม่ได้เลยคือการตรวจสอบว่าโปรแกรมมันขึ้นไปแล้วทำงานได้ปรกติ มีสุขภาพแข็งแรงดี การดู Log ว่าโปรแกรมเราทำงานมีปัญหาตรงไหน มีความสำคัญอย่างมากในการแก้ไขบั๊ก ยิ่งในตอนนี้ที่มักจะออกแบบให้โปรแกรมมันสามารถทำให้ Horizontal Scale ได้ (การสร้างเครื่องขึ้นมาหลายๆ เครื่องให้มาแบ่ง Load ทำงานแยกกันได้) จำเป็นอย่างยิ่งที่จะต้องมี Centralized Logging System รวบรวม Log เอามาไว้ในที่เดียว เพราะการ Remote เข้าไปอ่าน Log ในแต่ละเครื่องเป็นไปไม่ได้อีกแล้ว

**CI and CD**

Continuous Integration (CI) คือ กระบวนท่าที่ใช้สำหรับการรวบรวมซอฟแวร์ที่มีการพัฒนาแยกส่วนกันอย่างอัตโนมัติ อาจจะโดยหนึ่งหรือหลายนักพัฒนาก็ตามที สุดท้ายแล้วซอฟแวร์ที่พัฒนาชิ้นเล็กๆ ที่พัฒนาขึ้นมาจะต้องนำมารวมกันเป็นชิ้นใหญ่หนึ่งชิ้น จะทำอย่างไรให้มั่นใจได้ว่า ไม่มีชิ้นส่วนใดที่จะส่งผลให้ชิ้นส่วนอื่นๆ พังเสียหาย เนื่องจากเป็นการพัฒนาโดยโปรแกรมเมอร์หลายคน

ซึ่งเป็นไปได้ว่าจะมี bug หลุดมาจากส่วนใดส่วนหนึ่ง แล้วเราจะป้องกันได้อย่างไรละ ดังนั้นจึงต้องมีการเขียน script test ที่คอยทดสอบความเข้ากันได้ของแต่ละชิ้นส่วนโดยอัตโนมัตินั่นเอง โดยการ Testing จะเริ่มตั้งแต่ Unit Testing ซึ่งสร้างจากทีมพัฒนา และเป็นส่วนจะใช้ตรวจสอบว่าสิ่งที่ทีมพัฒนายังทำงานถูกต้องและจะใช้เวลาช่วงสั้น ๆ เท่านั้น

โดยในโลกของการพัฒนานั้น มักใช้ Build Server มาช่วยเพื่อให้เป้าหมายที่ตั้งไว้สำเร็จ กล่าวคือ จะเริ่มทำการ Integration กันตั้งแต่เมื่อมีการเปลี่ยนแปลง Source Code ที่ Repository กลาง ระบบจะทำการตรวจสอบ Code หลังจากการเปลี่ยนแปลงว่าทำงานร่วมกันได้หรือไม่ตั้งแต่ Compile, Testing

Continuous Delivery และ Continuous Deployment (CD) มักจะเกิดคำถามขึ้นมาเสมอว่า มันแตกต่างกันอย่างไร ? ทำไมต้องใช้ชื่อแตกต่างกันด้วยล่ะ ? ดังนั้นมาดูกันว่าแต่ละ คำคืออะไร และแตกต่างกันอย่างไร

Continuous Deployment คือ ในทุกๆ ขั้นตอนจนถึงการ deployment ขึ้น production จะทำแบบอัตโนมัติทั้งหมด

Continuous Delivery คือ การงานต่างๆ ใน deployment pipeline นั้น จะเริ่มต้นทำงานตั้งแต่การ compile, build ไปจนถึงขั้นตอนการทดสอบต่างๆ เช่น Acceptance test เป็นแบบอัตโนมัติทั้งหมด ส่วนในขั้นตอนการ deployment ขึ้น production นั้น จะต้องได้รับการอนุมัติหรือการตัดสินใจกันก่อนจากทาง Business ซึ่งเป็นการทำงานแบบ manual นั่นเอง หรืออาจจะเป็น One Click Deploy ก็ได้ [4]

**อ้างอิง**

[1] Chaonithi, K. (2019, November 16). DevOps คืออะไร นำมาประโยชน์ได้ยังไง และตัวอย่างการทำ DevOps ที่ Credit OK. SPICYDOG’s Blog. https://www.spicydog.org/blog/introduction-to-devops-and-the- practical-use-cases-at-credit-ok/

[2] DevOps คืออะไร—Amazon Web Services (AWS). (n.d.). Amazon Web Services, Inc. Retrieved February 26, 2021, from <https://aws.amazon.com/th/devops/what-is-devops/>

[3] Khvan, X. (09:26:16 UTC). *DevOps in Redmine described and explained*. RedmineUP Blog. <https://www.redmineup.com/pages/blog/devops-in-redmine>

[4] narissara. (2017, December 10). *การนำ Automated testing มารวมเข้ากับ กระบวนการ CI/CD*. Medium. https://medium.com/@narissara.tan/%E0%B8%81%E0%B8%B2%E0%B8%A3%E0%B8%99%E0%B8 %B3-automated-testing- %E0%B8%A1%E0%B8%B2%E0%B8%A3%E0%B8%A7%E0%B8%A1%E0%B9%80%E0%B8%82%E0% B9%89%E0%B8%B2%E0%B8%81%E0%B8%B1%E0%B8%9A- %E0%B8%81%E0%B8%A3%E0%B8%B0%E0%B8%9A%E0%B8%A7%E0%B8%99%E0%B8%81%E0% B8%B2%E0%B8%A3-ci-cd-58c97e31e230