พื้นฐานเกี่ยวกับ Version Control และ Git

11. โปรแกรม version control มีประโยชน์อย่างไร

ตอบ Version control คือ ระบบที่จัดเก็บการเปลี่ยนแปลงที่เกิดขึ้นกับไฟล์หนึ่งหรือหลายไฟล์เพื่อที่คุณสามารถเรียกเวอร์ชั่นใดเวอร์ชั่นหนึ่งกลับมาดูเมื่อไรก็ได้

12. ข้อได้เปรียบของ distributed version control เมื่อเทียบกับ centralized version control คืออะไร

ตอบ Distributed Version Control Systems (DVCSs) หรือระบบ VCS แบบกระจายศูนย์ ในระบบนี้ (เช่น Git, Mercurial, Bazaar หรือ Darcs) แต่ละคนไม่เพียงได้ก๊อปปี้ล่าสุดของไฟล์เท่านั้น แต่ได้ทั้งก๊อปปี้ของ repository เลย หมายความว่าถึงแม้ว่าเซิร์ฟเวอร์จะเสีย client ก็ยังสามารถทำงานร่วมกันได้ต่อไป และ repository เหล่านี้ของ client ยังสามารถถูกก๊อปปี้กลับไปที่เซิร์ฟเวอร์เพื่อกู้ข้อมูลกลับคืนก็ได้ การ checkout แต่ละครั้งคือการทำสำรองข้อมูลทั้งหมดแบบเต็ม ๆ นั่นเอง นอกจากนั้นระบบนี้ยังทำงานกับหลาย ๆ repository ได้อย่างดี ทำให้คุณสามารถทำงานกับคนหลายกลุ่มซึ่งทำงานในรูปแบบต่างกันในโปรเจคเดียวกันได้อย่างง่ายดาย เนื่องจากระบบนี้สนับสนุนการทำงานได้หลากหลายรูปแบบ ซึ่งอาจทำได้ยากในระบบแบบรวมศูนย์

13. ข้อได้เปรียบของ centralized version control เมื่อเทียบกับ distributed version control คืออะไร

ตอบ Centralized Version Control Systems (CVCSs) หรือระบบ Version Control Systems แบบรวมศูนย์ เช่น CVS, Subversion และ Perforce มีเซิร์ฟเวอร์กลางที่เก็บไฟล์ทั้งหมดไว้ในที่เดียวและผู้ใช้หลาย ๆ คนสามารถต่อเข้ามาเพื่อดึงไฟล์จากศูนย์กลางนี้ไปแก้ไขได้ ระบบการทำงานแบบรวมศูนย์นี้ได้ถูกนำมาใช้เป็นเวลานานหลายปี การทำงานแบบนี้มีประโยชน์เหนือ local VCS ในหลายด้าน เช่น ทุกคนสามารถรู้ได้ว่าคนอื่นในโปรเจคกำลังทำอะไร ผู้ควบคุมระบบสามารถควบคุมได้อย่างละเอียดว่าใครสามารถแก้ไขอะไรได้บ้าง การจัดการแบบรวมศูนย์ในที่เดียวทำได้ง่ายกว่าการจัดการฐานข้อมูลใน client แต่ละเครื่องเยอะ

14. บอกแนวทางในการแก้ไข conflict ที่เกิดขึ้นเมื่อมีการ merge โปรแกรมของผู้พัฒนาหลายๆคนเข้าด้วยกัน

ตอบ โดยปกติแล้ว Git merge จะรวมโค๊ดให้เราเองอัตโนมัติ แต่ก็จะมีข้อยกเว้นเมื่อ แก้ไขไฟล์เดียวๆกัน ลองนึกถึงกรณีที่เราและเพื่อนร่วมทีม แก้ไขไฟล์เดียวกัน Git จะเกิดการ conflict เมื่อเราจะ merge โค้ด โดยไม่รู้ว่าจะใช้โคhดของเราหรือของเพื่อน วิธีแก้ก็คือ ทำการ edit แล้ว commit ไปใหม่นั่นเอง

15. บอกแนวทางในการลด conflict ที่เกิดขึ้นเมื่อมีการ merge โปรแกรมของผู้พัฒนาหลายๆคนเข้าด้วยกัน

ตอบ ทีมพัฒนานั้นจำเป็นต้องตัดสินใจเลือก workflow ว่าจะไปในทิศทางไหน เพื่อลด conflict ต่างๆ เพื่อลดความสับสนต่างๆ เพื่อทำให้ทีมใช้งาน Git ได้อย่างเต็มความสามารถโดย workflow ที่ดี จะทำให้ทีมมี productivity ที่สูงขึ้น
รูปแบบของ workflow ที่ได้รับความนิยมสูง คือ Git-Flow

16. Git คืออะไร แตกต่างจาก GitHub อย่างไร

ตอบ Git คือ Version Control ตัวหนึ่ง ซึ่งเป็นระบบที่มีหน้าที่ในการจัดเก็บการเปลี่ยนแปลงของไฟล์ในโปรเจ็คเรา มีการ backup code ให้เรา สามารถที่จะเรียกดูหรือย้อนกลับไปดูเวอร์ชั่นต่างๆของโปรเจ็คที่ใด เวลาใดก็ได้ หรือแม้แต่ดูว่าไฟล์นั้นๆ ใครเป็นคนเพิ่มหรือแก้ไข หรือว่าจะดูว่าไฟล์นั้นๆถูกเขียนโดยใครบ้างก็สามารถทำได้ ฉะนั้น Version Control ก็เหมาะอย่างยิ่งสำหรับนักพัฒนาไม่ว่าจะเป็นคนเดียวโดยเฉพาะอย่างยิ่งจะมีประสิทธิภาพมากหากเป็นการพัฒนาเป็นทีม

GitHub คือ เว็บเซิร์ฟเวอร์ที่ให้บริการในการฝากไฟล์ Git (ทั่วโลกมักนิยมใช้ในการเก็บโปรเจ็ค Open Source ต่างๆ ที่ดังๆ ไม่ว่าจะเป็น Bootstrap, Rails, Node.js, Angular เป็นต้น)

Git และ GitHub แตกต่างกันที่ Git เป็น Version Control ตัวหนึ่ง แต่ GitHub เป็นเว็บเซิร์ฟเวอร์

17. จุดประสงค์หลักในการ branch คืออะไร

ตอบ Git branch เป็น feature ที่ช่วยให้นักพัฒนาสามารถที่จะทำงานได้สะดวกขึ้น ยกตัวอย่างเช่น เรามีโค้ดที่ดีอยู่แล้ว แต่อยากจะทดลองอะไรนิดๆหน่อย หรือแก้ไขอะไรก็ตาม ไม่ให้กระทบกับตัวงานหลัก ก็เพียงแค่สร้าง branch ใหม่ขึ้นมา เมื่อแก้ไขหรือทำอะไรเสร็จแล้ว ก็ค่อยเซฟกลับมาที่ master เหมือนเดิม

18. Fast forward merge คืออะไรและทำไมการ push ไปที่ remote repo จึงควรจะต้อง merge แบบนี้

ตอบ If Master has diverged since the feature branch was created, then merging the feature branch into master will create a merge commit. This is a typical merge. If Master has not diverged, instead of creating a new commit, git will just point master to the latest commit of the feature branch. This is a “fast forward.” Passing “--no-ff” creates a new commit to represent the merge, even if git would normally fast forward.

19. หน้าที่หลักของคำสั่ง git pull คืออะไร

ตอบ git pull ก็คือรวมโค้ดจาก remote มายัง local โดยที่เราไม่สามารถรู้ได้เลยว่าจะรวมโค๊ดอะไรบ้าง รู้แค่หลังจาก pull เสร็จแล้วนั่นเอง ซึ่งจริงๆแล้ว git pull มันก็คือการทำ git fetch และต่อด้วย git merge อัตโนมัตินั่นเอง

20. แผนภาพดานล่างนี้ต้องการสื่อความหมายอะไร

ตอบ merge และ branch