1.แล้วแต่คนจะชอบจะถนัดด้านไหน waterfall เน้นรูปแบบเป็นส่วนใหญ่มีการวางแผนออกแบบไว้เป็นขั้นตอนมีการวิเคราะห์ระบบส่วนมากจะเป็นเอกสารส่วน agile เน้นความพึงพอใจของลูกค้าและที่สำคัญการติดต่อสื่อสาร เน้นความง่ายไม่ซับซ้อน ไม่ค่อยเน้นเอกสารส่วนมากเน้นซอฟแวร์ที่ส่งให้ลูกค้า ติ้งมีความกล้าตัดสินใจและเคารพซึ่งกันและกัน เน้นว่าใครถนัดอะไร และการพูดคุยสื่อสารกัน มากกว่า การยึดติดที่เครื่องมือและกระบวนการ โดยยึดที่ผลผลิตหรือ software เป็นหลักและผล feedback จากลูกค้า

2.เป็นการทำงานคนละแบบแล้วแต่คนจะชอบหรือถนัด ถ้าชอบอะไรง่ายๆก้ Git มีการ pull upload file ที่ง่าย Git เป็น version control แบบ distributed คือ หลังจากที่ clone remote repo. มาแล้ว ก็เท่ากับว่าเรามี local repo. ที่เหมือนกันกับ remote repo. ด้วย จะ commit ก็สามารถทำได้แบบ offline เลยเพราะเป็นการ commit ไปที่ local repo. แต่ถ้าเราจะ push ไปไว้ที่ remote repo. แน่นอนว่าต้องเชื่อมต่อ network เช่นเดียวกันครับ ส่วน SVN เป็น version control แบบ centralized หมายความว่า ทุกอย่างจะถูกเก็บในตัวกลาง ซึ่งก็คือ remote repo. ทั้งหมด ทำใช้งานจำเป็นต้องเชื่อมต่อ network ไปยัง repo. ไม่เช่นนั้นก็จะ commit หรือ check revision ก่อนๆ ไม่ได้เลย

3. $ git add feature 1

$ git commit feature 1

$ git branch feature 1

$ git checkout feature 1

4.จริงยังไงทุกคนก้ต้องเจอปัญหานี้เพราะเวลาจะทำการ merge ต้องทำ pull และ fetch ก่อนเป็นขั้นตอนที่ซับซ้อน

5. ('e'..'e').each {|ch|print ch}

6.ก็อยู่ที่คนจะคิดหรือตัดสินใจอ่ะครับเป็นความชอบส่วนบุคลละอยู่ที่การออกแบบไปใช้เพื่ออะไรเหมาะสมแค่ไหนข้อดีของมันก้ใช้งานง่ายมีค่าใช้จ่ายต่ำมีข้อมูลจัดเก็บที่เดียวเชื่อมต่อกับเว็บแอพหรือบริการออนไลน์อื่นๆได้ง่าย ข้อเสีย รูป ร่างหน้าตา และการใช้งานมีได้จำกัด อาจไม่เหมาะกับงานบางประเภทที่ต้องการรูปแบบโปรแกรมที่แตกต่างจากโปรแกรม ทั่วไป เช่น โปรแกรมตกแต่งรูป โปรแกรมตัดต่อวีดีโอ เว็บแอพหลายๆตัวต้องการอินเตอร์เน็ตในการใช้งานเสมอ (มีบางตัวที่สามารถทำงานออฟไลน์ได้ด้วยเช่น Gmail)

7.ผู้ส่ง Request ข้อมูลไปยัง ส่วน Controller หลังจากนั้นก้ไปส่วนของ Model และส่งไปยังฐานข้อมูลหลังจากนั้นก้ส่งผ่านกลับไปยัง Controller ละก้ไปหน้าแสดงผลหรือหน้า View ละไปยังผู้ใช้

8.ระบบจองตั๋วหนังออนไลน์ มีลักษณะการทำงานเป็น MVC (Model-View-Controller)เหมือนกันข้อดีข้อเสียก้เลยไม่ค่อยแตกต่างกันมาก ข้อดีก้คือมีธรรมเนียมการเขียนที่ชัดเจน โปรแกรมเมอร์คนเก่าลาออก คนใหม่เข้ามารู้ Framework ตัวนั้นเขียนต่อได้เลยลดเวลาในการสร้าง Create, Read, Update และ Delete หรือ CRUD เพราะ Framework ส่วนใหญ่มีระบบ Generator ให้ พูดง่ายๆ ว่า Gen Code Insert Update Delete ให้ได้เลย ลดเวลาในการเขียนส่วนนี้ไปอีก ส่วนข้อเสีย มี Class จำนวนมากมาย เพื่อให้การทำงานอย่างเป็นระบบ ทำให้ต้องเขียน Class ที่เป็นตัวช่วยให้ผู้พัฒนาใช้ แต่บาง Class อาจไม่จำเป็นใช้เวลาในการเรียนรู้และศึกษาธรรมเนียมของการเขียนโปรแกรมใน Framework นั้นๆ (แต่ระยะหลังนี้มีการใช้มาตรฐาน PSR ในการพัฒนา ต่อไปเรียนรู้ Framework ตัวเดียวแต่สามารถเขียนได้ทุก Framework)มีระบบในการเชื่อมต่อไปยังฐานข้อมูลได้มากกว่า 1 ชนิด เช่น เชื่อมต่อไปยัง MySQL, MS SQL, ProgreSQL, SQLite และอื่นๆ ได้หลากหลาย (ทั้งนี้ขึ้นกับความสามารถของแต่ละ Framework)

9. Platform as a Service (Paas) ที่ให้เราใช้งานได้ฟรี (มีแบบเสียเงินด้วย) โดยรองรับภาษาโปรแกรมที่หลากหลาย เช่น Ruby, PHP, Node.js, Python, Java, Clojure, Scala และยังสามารถสร้าง buildpack สำหรับภาษาอื่นๆได้ เช่น Lua ที่รันอยู่บน OpenResty ได้อีกด้วย บทบาทสะดวกต่อการใช้งาน เช่น นักศึกษาอยากลองเขียนเว็บด้วย PHP แต่ไม่ได้เช่า Hosting ก็สามารถใช้ Heroku ได้ หรือแม้แต่บริษัท Start up ที่ไม่อยากวางเครื่องเอง คอนฟิกเอง ก็ใช้ได้ เพราะมันสามารถ scale ให้รองรับผู้ใช้เยอะๆได้โดยง่าย นอกจากรองรับภาษาโปรแกรมที่หลากหลายแล้ว ตัว Heroku มี App Store ของมันด้วยเรียกว่า add-ons สำหรับเพิ่มเติมบริการอื่นๆเข้าไปในแอปของเรา เช่น PostgreSQL, MongoDB, Redis เป็นต้น ซึ่งก็มีทั้งฟรี และไม่ฟรีให้เลือกใช้งาน

10.เพื่อให้นิสิตเรียนรู้การพัฒนาซอฟแวร์และนำไปใช้ให้เกิดประโยชน์สูงสุด