ข้อสอบ Full Stack Developer

1. ให้พัฒนาระบบ CRUD อย่างง่าย โดยใช้ Stack ดังนี้

   - Frontend: Angular
   - Backend: Typescript, Python, Golang เลือกใช้อย่างใดอย่างหนึ่ง
   - Database: เลือกใช้ตามที่เหมาะสม
   - (Optional) Test Automation: ต้อง test ได้แบบ automation (Unit Testing, API Testing, E2E Testing เลือกได้ตามที่เห็นว่าเหมาะสมสำหรับการพัฒนาระบบอย่างต่อเนื่อง)
   - (Optional) Deployment: Docker Compose

   *** Optional คือจะทำหรือไม่ทำก็ได้ ***

>> https://github.com/KonlawatPach/my-order-test - Shopeed เป็นเว็บไซต์ให้สามารถอัปโหลดสินค้า โดยสามารถเลือกสินค้าได้ตามประเภท สร้างสินค้าใหม่ๆ ซื้อสินค้าโหวตให้คะแนน แก้ไข และลบสินค้าออกได้ รันด้วยคำสั่ง “docker-compose up” และมีรายละเอียดต่างๆดังนี้

1) Frontend – Angular

ใช้โมดูลเพิ่มเติม คือ Sweetalert2 สำหรับแจ้งเตือนกระบวนการหลังจากการส่งข้อมูลมายัง Backend เพื่อช่วยให้งานเสร็จได้ไวขึ้น

2) Backend - Python

มีการนำเข้า Library Flask สำหรับทำ API และ Sqlite3 สำหรับเชื่อมต่อฐานข้อมูล

3) Database – SQLite3

เหมาะสำหรับฐานข้อมูลแบบ Relational Database ที่มีข้อมูลขนาดเล็กและจัดการง่าย ซึ่งผมเลือกใช้เพราะข้อมูลที่มีจำนวนไม่มาก และอยากลองฝึกใช้ SQLite ด้วย ซึ่งจะถูกสร้างเมื่อรัน backend หรือ app.py

4) Test Automation - ไม่ได้ทำ

ไม่เคยเขียนโปรแกรมสำหรับทดสอบระบบมาก่อนแต่วางแผนไว้ตอนแรกว่า การทำ API Testing จะมีการใช้ Tool อย่าง Postman หรือ Python Request เพื่อยิง Request สำหรับทดสอบ Response มาเปรียบเทียบความถูกต้อง ทำนองนั้นครับ

5) Deployment - Docker Compose

มีการสร้าง Dockerfile สำหรับรัน Frontend และ Backend แยกกันให้รันคนละ Port จึงทำให้ Docker Compose เมื่อรันจะ Build Image ขึ้นมาใหม่ จากนั้นจึงจะสร้าง Container และรัน Container เลย โดยถ้าเข้า http://localhost:4200/ จะเข้าถึงหน้าเว็บฝั่ง Frontend และ http://localhost:5000/ จะเข้าถึง Api ของหน้าเว็บ

2. คุณคิดว่า Software Tester เป็นบุคคลที่เขียน Code ไม่เก่งใช่หรือไม่ เพราะอะไร

   >> ส่วนตัวมีความเห็นว่าไม่ใช่ทั้งหมด เนื่องจาก Software Tester จะต้องมีความรู้ต่อยอดจากการเขียนโปรแกรมระดับเบื้องต้น โดยมีสองเหตุผลหลักๆ ที่ผมให้ได้ดังนี้

   1) ทำให้ทราบถึงการไหลของข้อมูลในซอฟต์แวร์ ข้อจำกัดของภาษาโปรแกรมมิ่งหรือประเภทของข้อมูล ทำให้คาดเดาความเป็นไปได้ของเหตุการณ์ในการทำกระบวนการหนึ่งๆ ได้ดีขึ้น เช่น การทดสอบการใส่ข้อมูลประเภทตัวเลข จะต้องครอบคลุมไปตั้งแต่การระบุตัวเลขในระยะ ตัวเลขนอกระยะ ทั้งในรูปแบบติดลบและบวก รวมไปถึงเลข 0 ที่อาจจะไปทำให้ระบบที่มีกระบวนการหารเกิด error ภายในได้
   2) ในบางสถานการณ์ Software Tester อาจจะต้องเขียนซอฟต์แวร์สำหรับทดสอบระบบ ซึ่งการกระทำดังกล่าว จะช่วยหา Input ที่ส่งผลให้เกิด error เมื่อ Input มีความเป็นไปได้จำนวนมาก

3. จงออกข้อสอบเองพร้อมเฉลย

>> การใช้งาน AI เพื่อช่วยตอบคำถาม หรือสร้างไอเดียใหม่ๆ อย่าง ChatGPT, Google Bard เป็นสิ่งที่หลีกเลี่ยงไม่ได้และต้องปรับตัวเข้าหาเทคโนโลยีเหล่านี้ในเวลาเดียวกัน คุณมีความเห็นกับเทคโนโลยีเหล่านี้อย่างไร และได้ใช้ AI เหล่านี้ในการสมัครงานเข้าบริษัทเราหรือไม่ ในส่วนใด

คำตอบของผม : สำหรับผม AI เหล่านี้นับเป็น Tool เหมือนกับการใช้ Google เพื่อค้นหาข้อมูล แต่มีลักษณะที่ละเอียดอ่อนและตรงไปตรงมากกว่า แน่นอนว่าในการทำแบบทดสอบนี้ผมได้ใช้ ChatGPT ในข้อ 1 เท่านั้น เป็นการทวนความรู้เก่าๆ เช่น การสรุป document ของ tech ที่อัพเดทมาให้อ่านง่ายขึ้น การแก้บัคที่เกิดขึ้นและไม่เคยเห็นมาก่อน และการรัน Angular บน Dockerfile ที่ผมไม่เคยลองทำมาก่อน แต่โค้ดโปรแกรมส่วนใหญ่ไม่ได้ก็อปมาวางโดยตรง เพราะสามารถดูได้จากโพรเจคเก่าๆของผมที่ทำไปแล้วน่าจะได้แนวทางการโค้ดที่ดีกว่า

** คำตอบจะแตกต่างไปตามแนวคิดของผู้ตอบ สามารถแสดงให้เห็นถึงทัศนคติที่มีต่อเทคโนโลยี และความตรงไปตรงมาในการตอบคำถามได้ **

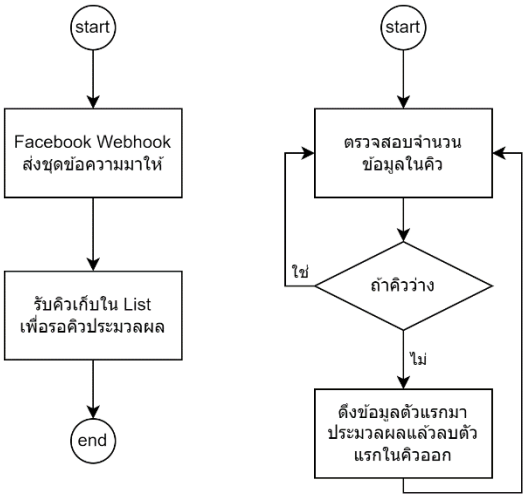
4. ถ้ามีระบบสำหรับรับข้อความแชทจาก Facebook Page ผ่านทาง Webhook ไป process 1000 ข้อความต่อวินาทีแบบ Real Time โดยที่ process time ต่อ 1 ข้อความประมาณ 1 วินาที จะออกแบบระบบ, ทดสอบ และ Monitoring อย่างไร (ข้อนี้ทำหรือไม่ก็ได้)

   >> จากลักษณะของระบบที่กล่าวมา Webhook จะข้อความจำนวนมากทำให้เกิดคอขวดตรงจุดที่ใช้ประมวลผล ซึ่งส่งผลให้ต้องใช้เวลาประมวลผลจำนวน 1000 วินาที หรือราวๆ 16 วินาทีกว่าๆ ต่อ 1 Webhook ออกแบบตามส่วนต่างๆดังนี้

   1) การออกแบบระบบ

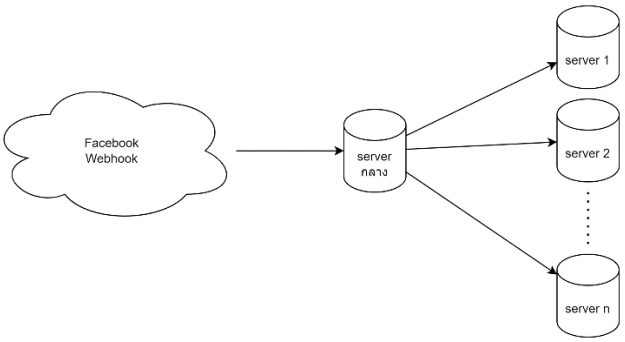
   แบ่งออกเป็น 3 ระดับ โดยค่าใช้จ่ายและประสิทธิภาพจะเพิ่มขึ้นตามลำดับ ดังนี้

   ระดับที่ 1 ถ้าระบบไม่จำเป็นต้องประมวลผลแบบ Realtime มาก ต้องการลดค่าใช้จ่ายจากจำนวน Server ประมวลผลหรือ Webhook ส่งมาไม่บ่อย มีการทิ้งช่วงให้ประมวลผล อาจจะใช้วิธีจัดเรียงคิวประมวลผลแทน



   ระดับที่ 2 ถ้าต้องการระบบที่ประมวลผลเร็วขึ้นมาหน่อย แต่ไม่ต้องการเพิ่ม Server ประมวลผล สามารถใช้วิธีประมวลผลควบคู่กัน เช่น การทำ Threading ในภาษา Python ให้ประมวลผลได้มากกว่า 1 ข้อความในเวลาเดียวกัน แต่จะต้องมีการจัดการให้ประมวลผลไม่ซ้ำข้อความกันด้วยเช่นกัน

   ระดับที่ 3 เพิ่มประสิทธิภาพสูงสุดคือการทำ Load balancing หรือการกระจายงานให้ Server หลายๆ Server ช่วยกันทำงาน โดยแต่ละ Server จะใส่วิธีในระดับที่ 2 ไปด้วยก็ได้ วิธีนี้จะไวสุดและทำระบบสำรองได้ด้วย



2) การทดสอบ

จากลักษณะของระบบ จะต้องทดสอบหลายส่วน ได้แก่

- **Server รับข้อมูล** จะต้องผ่านการประมวลผลครบทุกข้อความ และมีข้อความที่ผ่านการประมวลผลเท่ากับข้อความก่อนการประมวลผล ถ้ามีมากกว่าหรือน้อยกว่าแสดงว่าระบบประมวลผลข้อความซ้ำ หรือแบ่งข้อความแล้วมีการสูญหายของข้อความ
- **Server ประมวลผล** แล้วแต่ลักษณะของงานที่ประมวลผลข้อความ ถ้ามีการเก็บข้อมูลเข้าฐานข้อมูลก็จำเป็นต้องตรวจสอบข้อมูลในฐานข้อมูลด้วย
- **ส่วนแสดงผลหรือ Monitor ในขั้นตอนต่อไป** ตรวจสอบได้จากสถานะของข้อความที่ถูกประมวลผลหรือการแสดงผลข้อมูล รวมไปถึงการแสดงผลผ่าน User Interface ให้ถูกต้อง

3) การ Monitoring

แสดงผลข้อความที่เข้าคิวรอการประมวลผล ข้อความที่กำลังถูกประมวลผลแบบ Realtime และ Log ของข้อความที่ประมวลผลไปแล้ว ซึ่งจะถูกอัพเดทโดย Server ประมวลผลและจัดเก็บใน Database เป็นค่าบันทึกตาม Timestamp

หากว่าต้องมีการนำเสนอข้อมูลแบบสรุปรายวันหรือรายเดือนก็ให้ผู้ประกอบการ ก็สามารถออกปแบบระบบให้สรุปข้อมูลแต่ละ Record ไว้ในฐานข้อมูลแล้วค่อยนำมาแสดงผลในรูปของกราฟวงกลม กราฟแท่ง ตามความเหมาะสมของข้อมูลได้เลย