***Bài 2:***  
1/ Một số khái niệm liên quan đến KTPM – Software Testing

1.1/ The classic statements ( câu phát biểu kinh điển)  
- Phân biệt các thuật ngữ: **error, fault, failure, defect, mistake, bug**. Phải phân biệt trong ngữ cảnh, đoạn văn, câu chuyện  
=> Cùng có 1 ý nghĩa chung: có 1 thứ không như mong đợi xảy ra, sự sai lệch nào đó theo dự kiến nó phải là thế này, mà nó lại ra thế khác!!!

Khái niệm bug: Bug là 1 lỗi hay xuất hiện ở trong app, sai sót trong phần mềm và được tester phát hiện thấy được những người phát triển phần mềm sửa nó.

=> Bug là thứ gây nên sự bất ổn trong app, developer phải fix

***Bài 4:***

1/ Định nghĩa kiểm thử

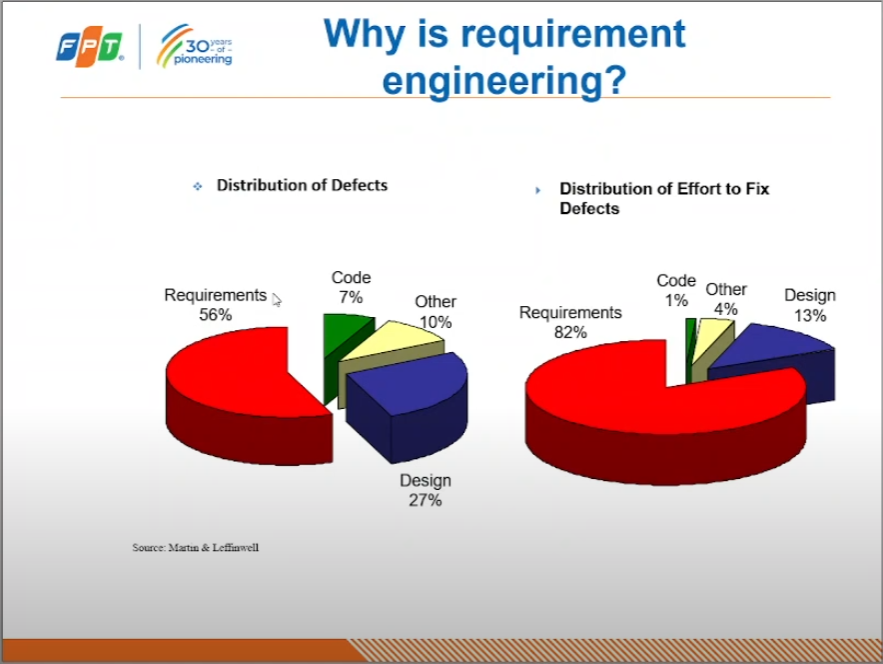
So sánh:

- Giữa cái mình kì vọng app phải như thế này ( Expected ) và cái thực tế App đang run và trả ra kết quả không như mong đợi

- Chạy app, chạy 1 chức năng, màn hình và nhìn kết quả và đánh giá ( evaluate/compare ) kết quả tính toán/hiển thụ của app so với cái ta kì vọng

( dynamic testing – chạy app kiểm tra )

- Kiểm tra xem app đã viết/ implement/ hiện thực có đúng như thiết kế/ mô tả / specification ( static testing – không chạy app chỉ kiểm tra các bản thiết kế đối chiếu với việc app được code/ design ra sao )

- Kiểm tra requirement có ổn không, khả thi không, phù hợp vs nhu cầu người dùng hay không  


***Bài 5:***

1/ Ai tham gia vào phán quyết App ổn, App có bug?

- Developer: - Người chịu trách nhiệm chất lượng code của mình

- Đọc code mình viết

- Chạy code mình viết, kiểm tra xem Expected có == Actual hay không

- Kỹ thuật nào để kiểm tra code của mình hiệu quả  
 - TDD, Unit Test

Test First Development

Test Driven Development ( thay đổi 1 chút hành vi viết code )

🡪 dẫn đến CI/CD/DevOps Continuous Integration

- Tester ( QC – Quality Control ): thánh soi, dân kiểm thử phần mềm ( thường sẽ không mở code hiểu thuật toán, cách viết code…) lỗi được ghi nhận bởi Tester/Dev team, sau đó fix

- Tester căn cứ trên cái gì để tìm sai sót/bug/lỗi

- Document (Specification, SRS, FRS, BRS,…) 🡺 static testing, là thánh soi, soi ngay khi viết Reqs, thiết kế UI, DB, kiến trúc

\* Mở rộng: Job title trong nghề làm P/M

- BA ( Business Analyst ) = reqs

- Designer ( UI )

- DB Designer

- Developer ( code ): FE, BE, Full-stack

- Tester ( QC )

- Supporter ( cài đặt, hướng dẫn sử dụng, ghi nhận sự cố… )

- Kiến trúc sư ( Solution Architect/ Software Architect 4-5k$/month )

- PM ( Project Manager )

- PM khác ( Product Manager )

- CIO ( cao cấp ) Chief Information officer

- CTO

***Bài 6:***

- Sếp bên tester: QC Manager, QC Lead/Leader | QC: Quality Control

- Quản lý các công việc liên quan đến quá trình kiểm soát chất lượng sản phẩm, quản lý các công việc liên quan quá trình tìm bug

- Lên kế hoạch về việc kiểm thử, lúc nào thì test app

- Phân bổ nguồn lực vào việc kiểm thử, chia người, chia task, bố trí bao nhiêu người, máy móc thiết bị dành cho việc test

Giả sử cần Test app bán hàng của siêu thị GS25

Manager: -Biết lộ trình phát triển App của Dev: tháng nào, sprint nào xong module nào

- Biết được độ phức tạp của app cần test: thuật toán, xử lí, thiết bị, AI…

- Cần bao nhiêu người để kiểm thử APP

- Lên kế hoạch tìm hiểu/ nghiên cứu bản thiết kế App, Specification – mô tả reqs hiểu được app xử lí làm gì -> Phân chia người thiết kế kịch bản test/ test cases tưởng tượng các luồng đi của user khi xài app, test thử demo, **luồng đi => test case, phân công người design luồng này**

KHI NÀO BẮT ĐẦU TIẾN HÀNH KIỂM THỬ CHẤT LƯỢNG PHẦN MỀM

- Càng sớm càng tốt, ngay cả khi chưa viết code

- Test ngay từ lúc lấy reqs, test ngay cái design: DB, UI, Architecture,…

OOP: 4 + 5 ( 4 NGUYÊN LÝ CĂN BẢN CỦA OOP + 5 SOLID )

TỔNG QUAN CHUNG VỀ CÁC GÓC NHÌN LÀM APP: 4 + 1 MODEL

SCRUM/AGILE: 4 ( 4 ĐIỀU TRONG AGILE MANIFESTO )

TESTING: 7 (7 ĐỊNH LUẬT/ĐỊNH LÝ//NGUYÊN LÝ) CẦN NHỚ KHI LÀM NGHẾ KIỂM THỬ