KỸ THUẬT TÌM YÊU CẦU PHẦN MỀM – SWR – SOFTWARE REQUIREMENTS – REQUIREMENTS ENGINEERING

Ta học về các kĩ năng tìm ra yêu cầu phần mềm từ bên đặt hàng làm app. Bên đặt hàng làm app có thể rất đa dạng: từ công ty, tổ chức, doanh nghiệp, trung tâm, trường học, siêu thị, nhà hàng, khách sạn, quán ăn, chuỗi nhượng quyền, bệnh viện, phòng ban, từ công ty phần mềm khác...

- Quy ước viết tắt:
  - Requirements: *REQs, reqs*
  - Product: *prod*
  - Manager: *mgr*
  - Project: *prj*
  - Specification: *spec*
  - Engineering: *eng*
  - Developer: *dev*

I. 3E: Engine, Egineer, Engineering

II. VÍ DỤ VỀ REQS

III. WHY R/E? TẠI SAO CẦN HỌC VỀ YÊU CẦU PHẦN MỀM

IV.SCOPE OF R/E: WHY, WHAT, WHO

V. CÁC KHÁI NIỆM CẦN PHÂN BIỆT, DỄ GÂY NHẦM LẪN

- Project vs. Product
- Project Manager vs. Product Manager vs. Product Owner vs. IT BA (Business Analyst) vs. BrSE (Bride Software/System Engineer)
- As is system/Legacy system vs. To be system
- Project Requirements vs. Product Requirements (xem phần phân loại reqs)

VI. PHẦN MỀM VÀ NƠI TÌM RA CHÚNG – NƠI NÀO MÀ APP ĐƯỢC VIẾT, ĐƯỢC TẠO RA

1. **CÔNG TY CÔNG NGHỆ - 100% MÔ HÌNH KINH DOANH KIẾM TIỀN LÀ DÙNG APP**
   - Ví dụ: Grab, Lazada, Shopee, Tiki, Sendo, MoMo, Elsa,...
   - Còn gọi là CÔNG TY LÀM PRODUCT, NHƯNG ÍT PRODUCT

2. **CÔNG TY TO/BỰ DÙNG APP TRONG KINH DOANH VÀ TRONG CÁC HOẠT ĐỘNG NÓI CHUNG ko cần 100% xài app trong việc kiếm tiền**
   - *Ví dụ*: TGDD, Bảo hiểm nhân thọ, TPBank, ACB Bank, HAGL, BeerSG, Thép Hòa Phát, Petrolimex...
     - Có các app dùng trong hoạt động nội bộ của họ: app Quản lí kho, Chăm sóc khách hàng
     - Có thể có các app kiểu product/generic: eBanking, ATM cho bá tánh xài
   - *Đặc trưng SE Jobs*: Developer nhận yêu cầu phần mềm từ PRODUCT MANAGER, PO cho app generic, nhận từ BA cho app làm cho bá tánh xài

3. **CTY THUẦN IT**

   3.1 NHÁNH LÀM PRODUCT – TỰ NGHĨ RA SẢN PHẨM ĐỂ LÀM CHO BÁ TÁNH XÀI
   - Ví dụ: Riot, VNG, EASport, Nintendo, Microsoft, Adobe, Facebook, Bytedance (Tóp tóp)
   - Đặc trưng SE Jobs: Developer nhận yêu cầu phần mềm từ PROD MGR, PO

   3.2 NHÁNH LÀM DỊCH VỤ - AI KÊU LÀM APP THÌ LÀM – SERVICE-BASED VÀ OUTSOURCING
   - Ví dụ: Fsoft, TMA, ELCA, Larion, KMS, Axon Active,...
   - Đặc trưng SE Jobs:
     - Developer nhận yêu cầu phần mềm từ BA nếu là khách hàng thông thường

       BrSE nếu là khách hàng là 1 cty IT khác

4. **CÔNG TY START-UP – KHỞI NGHIỆP, LÀM CÁI LOẠI APP HAY SẢN PHẨM CHƯA CÓ TRÊN THỊ TƯỜNG, VÀ HƯỚNG ĐẾN BÁ TÁNH XÀI – FOUNDER/SHARK CEO**
   - ❖ "Thương vụ bạc tỷ VTV3, SharkTank VN"
   - ❖ HỌC NHANH KIẾN THỨC VỀ KINH TẾ, LÀM SẾP CTY, QUẢN LÍ/QUẢN TRỊ CÔNG TY
   - ❖ TRƯỜNG ĐÀO TẠO DOANH NHÂN PACE, FSB (trường đào tạo kinh tế FPT)

- Ví dụ: Jupviec, Veca, Shine30, Thế giới thợ, Elsa,...
- Đặc trưng SE Jobs: Developer nhận yêu cầu phần mềm từ Prod Mgr, PO

➔ **CHỐT HẠ**:

- Cho dù chức vụ là gì đi chăng nữa, PROD MGR, PO, IT BA, BRSE, thì tất cả đều làm việc với yêu cầu phầm mềm, với góc nhìn WHAT và WHO
- Nếu cty làm product thì PHẢI TỰ BỎ TIỀN TÚI RA ĐỂ LÀM APP, VÀ HY VỌNG BÁN ĐƯỢC CHO NHIỀU NGƯỜI DÙNG HAY NHIỀU NGƯỜI SẼ BỎ TIỀN RA SỬ DỤNG APP → ĐỂ CHẾ CÒN NẾU Ế KO CÓ ĐỦ LƯỢNG NGƯỜI DÙNG CẦN THIẾT, LỖ VỐN, SẬP TIỆM
- Nếu cty làm theo kiểu dịch vụ, đặt hàng, thì BÊN ĐẶT HÀNG SẼ TRẢ TIỀN ĐỂ CÓ APP CHO HỌ XÀI, NÊN SẼ LUÔN CÓ TIỀN NẾU LÀM TỬ TẾ VÀ CHẤT LƯỢNG
  - NHƯNG KHÓ KHĂN LÀ: AI SẼ ĐẾN ĐẶT HÀNG MÌNH, MÌNH PHẢI CÓ SỐ MÁ LÀM APP + CHI PHÍ LÀM APP ĐỦ CHO NGTA CHI TRẢ

- Bạn là dev của CÔNG TY IT, CÔNG TY LẬP TRÌNH -> Bạn có danh phận
- Bạn là dev của CÔNG TY KO PHẢI IT -> Danh phận ko cao

- App viết cho bá tánh xài -> Ta cần po/prod mgr để đề xuất trính năng
- App viết cho đơn đặt hàng của ai đó -> Ta cần ba để tìm hiểu nhu cầu khách hàng
- Cty phần mềm khác (outsourcing) -> Ta cần ba/brse
- App viết cho bá tánh - product. Công ty product
- App viết theo nhu cầu đặt hàng riêng. Công ty service-based, công ty outsourcing

5. (14/02/2023) FREELANCER – DÂN CODE DẠO TỰ DO

- Đây là các bạn SE tốt nghiệp và ko muốn đi làm theo giờ hành chính, ko muốn nộp đơn vào 1 cty nào đó chờ lãnh lương tháng
- Đây là 1 gã SE code rất ngầu, và lên trên các diễn đàn việc làm kiểu freelance, đấu thầu các dự án mà người khác post lên, thắng thầu, nhận job, nhận deadline, làm xong nộp tiền vào tài khoản từ người đặt hàng
- Thu thập theo dự án, làm tự do giờ giấc, miễn đúng deadline
- Thu nhập ko cố định, tùy dự án, có khi rất cao, có khi cả tháng ko có job
- Bạn đóng vai: coder, vừa BA, tự tìm solution,...

- **Làm sao để tham gia**:
  - Code "xiềng"
  - *Xây dựng thương hiệu code*:
    - Tham gia các cuộc thi trên mạng để lấy giải thưởng
    - Tham gia các website luyện code để lấy chứng chỉ
    - Học các chứng chỉ nghề nghiệp: chứng chỉ code, CSDL, CLOUD,...
    - Xây dựng profile đẹp: có nhiều bài viết trên stackoverflow, medium và các forum/blog IT có tài khoản linked-in

VII. PHÂN LOẠI REQS:

Reqs nói chung là NHỮNG CÂU PHÁT BIỂU – A STATEMENT YÊU CẦU AI ĐÓ THỰC THI ĐIỀU GÌ

1. PROJECT REQS:

- Định nghĩa: *Những câu phát biểu liên quan đến các hoạt động quản lí dự án*
  - Quản lí dự án dính dáng đến các hoạt động, công việc mà tập trung vào các góc nhìn: TIME, SCOPE, BUDGET, HR, QUALITY
  - Người cần giải quyết thực thi các reqs này chính là PM
- Ví dụ:
  - Tối nay anh em tăng ca để kịp tiến độ heng? (OT - Overtime)
  - Cần 3 máy đọc barcode thương hiệu X để cài đặt tính năng TẠO MỚI ĐƠN HÀNG
  - Cần mua máy in nhiệt để in BILL trong việc cài đặt tính năng TẠO MỚI ĐƠN HÀNG
  - Cần mua 2 CAMERA với cấu hình... để detect biển số xe máy trong app Bãi giữ xe thông minh

2. PRODUCT REQS – SWR, R/E, BA, PROD MGR, PO: TẬP TRUNG VÀO CÁI MÓN REQS NÀY:

- Định nghĩa:
  - Những câu phát biểu đề cập đến TÍNH NĂNG CỦA PHẦN MỀM ĐANG LÀM, ĐỘ TRẢI NGHIỆM CỦA USER VỚI CÁC TÍNH NĂNG
  - Những câu phát biểu đề cập đến góc nhìn WHY, WHAT, WHO, đặc biệt là 2 góc nhìn WHAT, WHO
  - Những câu phát biểu đề cập, liệt kê tên tính năng, tên màn hình, tên chức năng
  - Người đưa ra các câu phát biểu này chính là: IT BA, BA , BRSE, PROD MGR, PO
  - Người giải quyết các reqs này chính là DEVELOPER, CÀI ĐẶTVIẾT CODE CHO TÍNH NĂNG ĐƯỢC MÔ TẢ
- Ví dụ:
  - App cần cung cấp tính năng tạo mới hồ sơ bệnh nhân cho đội ngũ nhận bệnh/receptionist dùng
  - App cần cung cấp tính năng/màn hình book lịch khám bệnh, cho bệnh bệnh nhân dùng và book trước ngày giờ khám bệnh
  - App cần cung cấp tính năng ghi nhận việc mượn sách của độc giả (app Quản lí thư viện)
  - App cần cung cấp tính năng gửi tin nhắn khuyến mãi, tặng voucher cho khách hàng thân thiết (app 7-Eleven)
  - App cần cung cấp các loại bản đồ, các loại gameplay khác nhau (video game)
  - App có tính năng cho phép huấn luyện viên xây dựng đội hình cho mỗi trận đấu (game FIFA)
  - App có tính năng khách mua theo dõi trạng thái đơn hàng, tài xế giao hàng nhận đơn hàng cần giao (app Lazada, app bán hàng online..)
  - App có tông màu phù hợp với màu phong thủy của cty
  - Màn hình của quầy thu ngân là màn hình đa chạm của Samsung!!!

➔ **CHỐT HẠ**:

- MÔN HỌC QUẢN LÍ DỰ ÁN ĐỂ SAU NÀY LÀM JOB/POSITION PM THÌ TẬP TRUNG VÀO PROJECT REQUIREMENTS
- MÔN HỌC SWR THÌ TẬP TRUNG VÀO PRODUCT REQUIREMENTS
- TRONG MÔN HỌC NÀY, TRONG NGHỀ BA, BRSE, PO, PROD MGR THÌ CHỈ TẬP TRUNG VÀO PRODUCT REQUIREMENTS

PRODUCT REQS LẠI ĐƯỢC CHIA NHỎ THÀNH CÁC LOẠI NHỎ HƠN

2.1 BUSINESS REQUIREMENTS – GÓC NHÌN WHY – TẠI SAO LÀM APP

2.2 USER REQUIREMENTS: YÊU CẦU NGƯỜI DÙNG ~~~ USER STORY!! – Góc nhìn WHAT/WHO

* Định nghĩa: USER REQS là những câu phát biểu ở góc nhìn của người dùng, họ tức là người dùng app sẽ sử dụng được những tính năng gì của app để giúp cho các công việc của họ

- Những câu phát biểu nói về chức năng, tính năng, tên màn hình mà user sẽ sử dụng!!!
- Những câu phát biểu này rất dễ hiểu với người bình thường!!!
- Tính năng, chức năng, tên màn hình: bàn về sau khi sử dụng xong màn hình thì user đạt được kết quả gì. Ta cũng hk cần đề cập chi tiết thiết kế màn hình, các ô nhập/component có trên màn hình ta chỉ nói màn hình làm được điều gì cho user mà thôi!!!

- Ví dụ:
  - App có tính năng cho phép khách hàng search các chuyến bay (app bán vé máy bay)
  - App có tính năng cho phép người dùng book 1 hoặc nhiều phòng khách sạn (app du lịch)
  - App có tính năng cho phép game thủ trao đổi các vật phẩm họ "cày" được (app game)
  - App có tính năng tạo nhóm chat (app Discord)
  - App có tính năng enable/disable bình luận của cộng đồng (app Youtube)

- **Tương đương** với cái khái niệm **USERSTORY** trong phương pháp phát triển phần mềm *Agile* ***(As a <ROLE>, I want to <WHAT/VERB> so that <PURPOSE>)***

  Vd:
  - Tôi muốn app giúp lưu hồ sơ bệnh nhân
  - Userstory as a receptionist, i want to create a patient profile so that the hospital can tracking his her treatment progress

2.3 SYSTEM REQUIREMENTS – Góc nhìn WHAT/WHO

- Là những câu phát biểu chi tiết hơn cho các câu user requirements
- Nó được quyền dùng các yếu tố, thuật ngữ IT, của kỹ thuật

- Người đọc hiểu system reqs là dân dev, dev team
- "System" nghĩa là mô tả chi tiết hơn cho các app to-be sắp viết code

*2.3.1 FUNCTIONAL REQUIREMENTS: YÊU CẦU CHỨC NĂNG (**VERB**)*

* Định nghĩa:

- Những câu phát biểu về chức năng, tính năng tên màn hình của app mà người dùng sẽ tận hưởng tính năng này và dùng nó trong công việc của họ
- Những câu phát biểu này mô tả khá chi tiết chức năng, màn hình, mang yếu tố, lập trình

* Ví dụ

- APP giúp tạo/lưu hồ sơ bệnh nhân (quầy nhận bệnh/ receptionist) với các thông tin cơ bản: tên, tuổi địa chỉ, số điện thoại, các dị ứng, mã số bệnh nhân - tự generate tăng dần theo định dạng campus yyyy mm xxxx (ví dụ: hcm20230212345)

- Website tgdd có tính năng tìm kiếm sản phẩm (user requirements)
- Website tgdd có tính năng tìm kiếm sản phẩm với ô nhập gõ keyword radio button để chọn loại
- Sản phẩm: điện thoại, máy tính, tablet, phụ kiện (System req-funtion req)

*2.3.2 NON-FUNCTIONAL REQUIREMENTS: YÊU CẦU PHI CHỨC NĂNG **(ADJ nutri: độ đo)***

* Trong các câu phát biểu tính năng phần mềm, có những câu thuộc về nhóm không đề cập đến 1 chức năng cụ thể, mà đề cập chung cho toàn bộ app, và nó nói về các trải nghiệm xài app, những câu phát biểu chứa tính từ mô tả tính chất của app, tính chất của các chức năng nói chung, chứ không nói chi tiết màn hình làm gì cho user. Những câu này phát biểu/ yêu cầu nói về tính chất của app, trải nghiệm xài app gọi là non-function reqs

- Mẹo/tips nhận diện cái phát biểu/ yêu cầu thuộc non-func: tốt/ xấu/ đẹp; nhanh/chậm/to/nhỏ; hài lòng/yêu thích/ghét; an toàn /bảo mật; tuân thủ

Vd:

- App có màu sắc/ layout hài hoà
- App dễ sử dụng - easy to use...
- App có tông màu phản ánh tông màu phong thuỷ / tông màu nhận diện thương hiệu công ty
- App chọn tông màu phù hợp ngành hàng!!!
- App xử lí mọi request trong vòng 3s đổ lại
- Dàn máy thu ngân màn hình xài dell
- Việc tính thuế phải theo thông tư.. của BTC phát hành
- Hệ thống sử dụng chứng chỉ mã hoá "HTTPS:

❖ CẦN PHẢI CÓ ĐỘ ĐO CHO CÁC CÂU NÀY – METRICS ĐỂ ĐO CÁI CẢM GIÁC ĐƯỢC CHÍNH XÁC → SAU NÀY TESTING CÒN ĐÁNH GIÁ ĐƯỢC

NF mơ hồ	NF có độ đo (metric)
• App chạy nhanh • App dễ dùng	• App chạy nhanh, 3s đổ lại • App dễ dùng: sau 1h huấn luyện xài ngon!

• App màu sắc hài hòa • App gọn nhẹ về kích thước • Màn hình thu ngân xài màn hình thương hiệu	• App xài theme download tiwf chuẩn thiết kế Material UI... • Bản download không vượt quá 100MB • Màn hình thu ngân xài màn hình Samsung

VIII. STAKE HOLDERS

- Để làm 1 phần mềm cần nhiều người tham gia, mỗi người 1 vai trò, tuỳ theo app sẽ có thêm 1 ai đó đặc biệt!
- Danh sách những người tham gia, liên quan đến việc làm phần mềm thì gọi chung là stakeholders những người liên quan

1. **DEV TEAMS**
   - Designer
   - Scrum
2. **(14/02/2023) IT/BA/PO/BRSE**
   - Là người làm việc với khách hàng để tìm ra yêu cầu của họ về app sẽ cần làm
   - Là người kết nối giao tiếp với dev team để truyền đạt tính năng phần mềm
   - Là người viết ra SRS - tài liệu mô tả toàn bộ chức năng phần mềm
   - Nếu là app GENERIC, PRODUCT, app viết cho số đông người dùng, cần PO/Prod Mgr
   - Nếu là app custom, CUSTOMIZED, BesPOKE, app viết theo đặt hàng của ai đó cụ thể, cần gã hiểu và nói chuyện được với khách hàng
     - Analysis (hiểu)
     - Business (khách hàng, doanh nghiệp, nghiệp vụ)
     - Business analysis/business analyst
     - ➢ Cần: BA/BrSE (Japan-N2)

❖ **PHÂN BIỆT 1 SỐ KHÁI NIỆM:**

- *BA - Business Administration:* Quản trị kinh doanh
- *MBA - Master of Business Administration:* ThS. Quản trị kinh doanh (Ai cũng học được, miễn có bằng ĐH, cần học thêm vài môn chuyển đổi thuộc kinh tế nếu ko xuất thân từ đại học có liên quan kinh tế)
- *BA - Business Analytics:* Phân tích kinh doanh: dùng tập data kinh doanh của công ty trong nhiều năm để rút ra được các kết luận:
  - *Nhóm hàng nào trong cty cần cắt giảm, làm việc hiệu quả hơn...*
  - *Bộ phận nào trong cty cần cắt giảm, làm việc hiệu quả hơn...*
  - *Kiến thức kinh tế + áp dụng công nghệ thông tin để ra được quyết định cần làm gì cho doanh nghiệp*

- *BA - Business Analyst:* làm việc với bất kì ai đặt hàng làm app để tìm ra yêu cầu phần mềm, tìm ra các màn hình, các chức năng của app sẽ làm

❖ Định nghĩa BA:
- Là 1 gã đứng giữa giao tiếp các bên trong dự án phần mềm
  + Giao tiếp với QC/Tester để nói với họ về các chức năng mà app sẽ có để QC/Tester còn biết đường chuẩn bị test những gì
  + Giao tiếp với Developer để nói họ code những màn hình gì
  + Giao tiếp với sếp PM để nói về số lượng reqs, độ khó
  + Giao tiếp với Customer/user để hiểu nhu cầu của họ

❖ CÔNG VIỆC CHÍNH MÀ BA CẦN PHẢI LÀM: (*SLIDE CHAPTER 4*)

- Làm việc với khách hàng và user để hiểu nhu cầu của họ
- Tìm ra xem, phân loại xem, trả lời câu hỏi:
  - "APP VIẾT CHO NHỮNG AI XÀI"
  - "NHẬN DIỆN NHỮNG LOẠI USER, USER CLASSES"
  - "NHẬN DIỆN CÁC ROLE"
- Tìm ra các reqs, những màn hình sẽ phục vụ cho ROLE nào đó ELICIT REQUIREMENTS
- Viết ra cái document "đẹp trai xinh gái" liệt kê tất cả các requirements tìm thấy từ phía khách hàng hay user ****** NHIỆM VỤ CHÍNH
  - Có các loại document sau mà dân BA phải viết ra:
  - BRD: Business Reqs Document
  - SRS: Software Reqs Specification
  - FRS/FRD: Functional Reqs Specification/Document/Documentation
  - Quản lí được sự thay đổi của các Requirements theo các version khác nhau của phần mềm cho các khách hàng khác nhau, mỗi version release ra thị trường hay bàn giao cho khách hàng thì có thêm/bớt tính năng gì
  - Liệt kê các chức năng đi theo mỗi version
    Để dành cho phần hỗ trợ khách hàng, fixbug – biết tính năng nào xh ở version nào!!!
    VD: + ChatGPT bản free -> ko hỗ trợ user VN
    Plus -> hỗ trợ user VN
    + App X nào đó: trả phí, free được dùng tính năng nào (Virus)
  - Bạn đã từng quen: CHANGE LOG, README, RELEASE NOTES LÀ NHỮNG TẬP TIN GHI LẠI SỰ THAY ĐỔI CỦA APP THEO THỜI GIAN, VỚI MỤC TIÊU KHI KHÁCH HÀNG KHIẾU NẠI, BUG, DEV TEAM CÒN BIẾT BUG ĐÓ, TÍNH NĂNG ĐÓ NẰM VERSION NÀO, ĐỂ XEM LẠI CODE MÀ FIX

❖ CÁC KỸ NĂNG CẦN CÓ CỦA DÂN BA:

2 kỹ năng lớn: KỸ NĂNG CỨNG – HARD-SKILL VÀ KỸ NĂNG MỀM – SOFT-SKILL

KỸ NĂNG CỨNG – KỸ NĂNG CHUYÊN MÔN SE:

- Để làm 1 dân BA tốt, tốt nhất nên có chuyên môn từ SE, nên xuất thân từ SE, hoặc được trang bị kiến thức SE (ko cần code giỏi)
  - Biết SE sẽ dễ dàng trong việc đưa ra các phương án giải pháp kỹ thuật phù hợp với túi tiền khách hàng, phù hợp với thời gian làm dự án, và ko gây khó cho các bạn dev dễ thuyết phục các bên khi ta cũng là dân kỹ thuật
- Khách hàng MUỐN LOGIN BẰNG VÕNG MẠC/FACE ID:
  - BA xuất thân SE, biết ràng xác thực bằng gương mặt, mắt sẽ có độ chính xác tương đối, phụ thuộc vào cả ánh sáng, thuật toán có độ chính xác x%, tốn tiền tốn t.gian để cài đặt do cần mua/sử dụng thư viện bên ngoài

KỸ NĂNG MỀM – SOFT-SKILL:

- Kỹ năng lắng nghe:
  - Lắng nghe khách hàng trình bày, mô tả cái **AS IS SYSTEM**
  - Lắng nghe cách khách hàng đang làm công việc của họ khi chưa có app mình sẽ đang viết
  - Lắng nghe để hiểu được nhu cầu, câu chuyện, nỗi đau trong công việc của khách hàng khi chưa có app của mình
- Kỹ năng đặt câu hỏi:
  - Để ngta nói cho mình nghe thì mình (BA) cũng phải biết đặt câu hỏi cho khéo để lấy được info (môi trường, công việc của họ) từ khách hàng.
- Phân tích/tổng hợp, tư duy hệ thống (**SYSTEM THINKING**):
  - Khi BA làm việc với khách hàng, làm 1 app cụ thể nào đó, cần nhìn rộng ra cái app của mình viết có tương tác với app khác hay có đang nằm trong 1 hệ thống khác to lớn hơn hay hk.
    VD: Khi làm app quản lí nhân sự:
    + Quản lí hồ sơ, bằng cấp
    + quản lí thăng tiến, lên chức, chuyển phòng ban, quản lí khen thưởng, kỷ luật
    + quản lí người phụ thuộc (tính thuế TNCN)
    + Nghĩ xa hơn: có hồ sơ nhân viên, vậy có làm phần chấm công??
    + có làm phần tính lương?
    * Chấm công: ghi nhận ngày đi làm, ngày off
    + Làm lương yhangs thì ghi nhận ngày nghỉ phép
    + Làm theo ca, khoán, partime thì phải check từng ngày
    + Hình thức check điểm danh ra sao?
    * TÍNH LƯƠNG: rất nhiều loại tiền góp vào thành lương cuối tháng
    + Lương cơ bản, phụ cấp (độc hại, thiết bị), phụ cấp trách nhiệm, tăng ca, thưởng, phạtr, thuế TNCN, BHYT, BHXH, BHTN,..
  - Quản lí nhân sự có dính đến QL chấm công dính đến Kế toán và Lương
    + Hoặc là làm từng phần, hoặc là QLNS giao tiếp/export danh sách sang hệ thống Chấm công, Kế toán
- Kỹ năng tổ chức/điều phối (**OGANIZATION/FACILITATION**)
  - Trong những tình huống cần gặp 1 lúc nhiều khách hàng/nhiều user, lúc này cái tài tổ chức của BA phải cần đến
    VD: app QL bệnh viện Chợ Rẫy
    + Có tình huống BA sang BV hẹn gặp cùng lúc: tiếp tân (quầy nhận bệnh), kế toán thu ngân, bác sĩ khám, bs chiếu chụp, bs xét nghiệm, bs điều trị nội trú...

+ Gặp cùng lúc để nắm được luôn cái trình vận hành của bệnh viện (phần khám bệnh)

+ Ai phát biểu, nội dung gì, ghi chép ra sao, hỏi đáp thế nào, tương tác ra sao tất cả nhờ 1 tay BA.

- Kỹ năng mô hình hóa (**MODELING**)
  - Dân BA cần có khả năng "vẽ", biểu diễn những luồng xử lí, quy trình làm việc của khách hàng thành những sơ đồ cho dễ hiểu để theo dõi
  - Có khả năng vẽ 1 số mô hình liên quan đến quy trình phần mềm

    + Sơ đồ Context diagram, sơ đồ Activity, sơ đồ State Machine, sơ đồ Use Case, sơ đồ BPMN, sơ đồ ERD...
  - ⇨ Mục tiêu vẽ sơ đồ: biểu diễn lại info mình thu lượm được từ khách hàng thành thứ dễ hiểu dễ xem

NHỮNG KỸ NĂNG QUAN TRỌNG KHÁC BA CẦN PHẢI CÓ – ESSENTIAL KNOWLEDGE

- Kỹ năng quan sát, tìm hiểu các vấn đề của xã hội!!! Để biết cơ hội làm 1 cái app nào đó cho xã hội hoặc chính sách luật pháp thay đổi sẽ ảnh hưởng đến code 1 số app nào đó

  VD: hệ thống đua xe đạp AR, VR. Hệ thống chơi Golf, huấn luyện golfer qua app

  Hệ thống lại xe ô tô qua cabin ảo – trường dạy lái ô tô, tàu thủy,...

  Bộ Tài Chính/Bộ LDTB&XH chuẩn bị thay đổi công thức tính thuế TNCN -> chuẩn bị sửa code!
  - ⇨ Những bài báo về khởi nghiệp, ứng dụng công nghệ đọc để có kiến thức, biết tính năng của các loại app, biết được các lĩnh vực cần có app
- Cần có kiến thức chuyên môn về 1 lãnh vực mà app sẽ được viết – DOMAIN KNOWLEDGE, kiến thức này giúp ta hiểu bài toán của khách hàng, hiểu câu chuyện của khách hàng!

  VD: làm app về bệnh viện thì cần có kiến thức cơ bản về khám chữa bệnh, hồ sơ bệnh án, đơn thuốc, báo cáo về bệnh tật với bộ y tế... hiểu quy trình thăm khám chữa bệnh cấp cứu!

  Làm app về bán hàng online, thì cần biết về: quản lí sản phẩm khuyến mãi, voucher, chiết khấu, theo dõi đơn hàng, khiếu nại đổi trả, giao hàng, thanh tonas, kho hàng...

  ➢ NẮM ĐƯỢC CÁC THUẬT NGỮ CHUYÊN MÔN CỦA LĨNH VỰC – DOMAIN MÀ APP SẼ CHẠY:
  - Làm app y khoa, cần biết cơ bản thuật ngữ, quy tình y khoa
  - Làm app giáo dục, cần biết cơ bản thuật ngữ, quy trình đào tạo
  - Làm app thương mại điện tử cần biết cơ bản thuật ngữ, quy trình bán hàng, quản lí sản phẩm, khiếu nại chăm sóc khách hàng,...

BA KHÔNG CHỈ LÀ NGƯỜI BIẾT LẮNG NGHE KHÁCH HÀNG MÀ CÒN LÀ NGƯỜI TƯ VẤN NGƯỢC LẠI CHO KHÁCH HÀNG CÁC SOLUTION (GIẢI PHÁP) TỐT VÀ PHÙ HỢP – GIẢI PHÁP THƠM, NGON, BỔ, GIÁ TỐT

- Ví dụ: App QLNS, có phần chấm công/điểm danh
  - Chấm công = vân tay. Bao nhiêu máy vân tay? COVID
  - Chấm công = app. Nhân viên scan barcode! (Check đúng wifi công ty)
  - Chấm công = thẻ RFID. Đi qua khu vực, tự check!!
  - Chấm công = nhận diện gương mặt, đưa mặt vào cam!

❖ **LÀM SAO ĐỂ TRỞ THÀNH BA?**

Có nhiều cách, con đường để vào ngành BA:

1. Chủ động muốn theo nghề BA khi học hành/khi lựa chọn nghề nghiệp (ROOKIE – lính mới)

- Bạn cần có những kỹ năng như mô tả trên, phải hiểu được đặc trưng công việc
- Bạn cần "tháp tùng", làm đệ tử, đi theo 1 tay BA kinh nghiệm để học nghề

2. Dân QC chuyển sang BA, Option này hay, vì QC là gã rất hiểu app, rất hiểu tính năng app, rất hiểu các màn hình làm gì, xử lí gì???

❖ Dân QC là dân hiểu nhất về mặt nghiệp vụ, tính năng, hành xử của app; hơn cả dân Dev

⇨ Chuyển sang BA tuyệt vời do quen làm việc với tính năng

3. Developer, PM: gã này ổn khi chuyển sang BA, quen với tính năng, quen với quy trình, quen luôn viết code đằng sau 1 tính năng!

❖ Developer hơi "ngại ngần" về giao tiếp!

4. Expert user – Gã user, gã người dùng rất kinh nghiệm về loại app nào đó

- Gamer xịn xò, streamer xịn xò, chơi game giỏi, review game, đánh giá được tính năng nào ngon, hấp dẫn -> gã PO tốt ở góc độ đề xuất tính năng cho app mới/ game mới
- Điểm hạn chế: gã này chỉ làm BA/PO của 1 nhánh kiến thức/nhánh domain
- Hoặc cần trang bị thêm kiến thức SE, đa ngành thêm nếu có thể
- Không thì đào sâu thật sâu đúng cái mình đnag theo!

CẦN HỌC THÊM CÁC CHỨNG CHỈ LIÊN QUAN BA/PO NỮA!! CHỈ NÊN HỌC KHI ĐÃ ĐI LÀM 1 NĂM

❖ Chứng chỉ IIBA có nhiều level: bacs.vn; fpt software academy; BA BOK (Body of Knowledge – Dàn khung kiến thức BA – cuốn luyện thi)

❖ Chứng chỉ PO: SCRUM.ORG

3. CUSTOMER (KHÁCH HÀNG)

❖ Định nghĩa:

- Là gã đặt hàng nhóm dev (cty có nuôi nhóm dev) hoặc cty phần mềm để làm app cho họ
- Là gã đưa ra yêu cầu về phần mềm mà họ cần.

(VD:

1. Ba má đặt hàng Fsoft làm app Quản lí chuỗi khách sạn/homestay! Cho nhân viên xài, và khách hàng của ba má xài

+ Ba má: customer

+ Nhân viên, người thuê phòng xfai

2. TGDD: Nhân viên bán hàng, Phòng kinh doanh/bán hàng đặt hàng dev team của TGDD luôn làm cái app bán hàng cho: chính họ xài
3. Bệnh viện Chợ Rẫy đặt hàng làm app QLBVCR

Ông giám đốc đặt hàng Fsoft làm app cho các Bsi xài

)

4. **<u>USER (NGƯỜI DÙNG), END-USER (END trong back-end, end-user)</u>:** Tôi là người cuối cùng thụ hưởng/sử dụng data từ app trả về END

❖ Định nghĩa: Là gã xài/dùng cái app giúp cho công việc của họ!
Là gã đưa ra yêu cầu về phần mềm mà họ cần.

VD: y chang trên

❖ USER được chia làm 2 loại:

- USER 1: Direct User: người trực tiếp xài app, trực tiếp chạm các màn hình, tính năng của app; dùng các tính năng của app cho công việc của họ
  Vd: app thu ngân: cô bé/chàng trai đứng ở quầy tính tiền
- USER 2: Indirect User: người ko chạm vào app, nhưng cần data từ app để ra quyết định là sếp trong cty: ông hiệu trưởng

[CHỐT HẠ QUAN TRỌNG]

❖ Ví dụ: Phần mềm FAP của trường mình:

- Năm 2006 thầy hiệu trưởng đặt hàng Fsoft viết app này!
  Customer of Fsoft
  Ai xài app? - sinh viên, giảng viên, nhân viên đào tạo
  Thầy hiệu trưởng có xài ko? – Ko xài dù ông ấy trả tiền để có app, nếu thầy cần data gì có nhân viên lấy cho!

❖ TUYỆT ĐỐI KO ĐƯỢC CÓ SUY NGHĨ CÁC SẾP TRONG CÔNG TY SẼ LÀ ADMIN!!

XIN LỖI, ADMIN CHỈ LÀ THẰNG KU LÀM THUÊ, ĐƯỢC MƯỚN ĐỂ QUẢN LÍ DATA

ADMIN CÓ QUYỀN SINH SÁT TRÊN DATA NHƯNG ADMIN KO PHẢI LÀ LÃNH ĐẠO CTY

❖ KHI VIẾT APP PHẢI TÍNH LUÔN TÍNH NĂNG CHO ÔNG INDIRECT USER, NHỮNG ÔNG DIRECT USER NHƯNG LÀ SẾP VỪA VỪA, HỌ CẦN SỐ LIỆU

BA SẼ PHẢI GIAO TIẾP GẦN GŨI VỚI CUSTOMER/USER ĐỂ HIỂU NHU CẦU CỦA HỌ CHO CÁI APP ĐANG SẮP LÀM!! Các kĩ năng cần có cho BA đem ra xài!

NGOẠI TRUYỆN:

- Đặt hàng mà ko xài, có hay ko ngoài đời?
  + Người đặt hàng và người xài có thể là 1 hoặc 2 tùy tình huống, nhưng họ đều chung 1 thứ: có đưa ra yêu cầu
  (
  vd:
- Ba má trả tiền mua con xe Exciter cho mình xài

Cust. User
Yêu cầu: giảm giá Độ lại xe
Biển số đẹp

Mình mua (trả tiền) và mình xài luôn, đóng 2 vai!
+ Mình yêu cầu nhiều thứ

- BA MÁ MUA/ĐẶT LÀM PHẦN MỀM KẾ TOÁN, AI XÀI? Nhân viên kế toán xài
  ĐẶT LÀM APP/WEBSITE BÁN HÀNG, AI XÀI? Thu ngân, sales ba má mướn

)

tôi, bạn ---------- browser -------- trang web ---------- server ---------- database
lướt tgdd.com trên browser code
điện thoại, tablet

trình duyệt nhận data từ back-end render trang web đập vào mặt user và chuyện show trước mặt user – front-end

- Full-stack: chồng đĩa Front-end và Back-end

5. USER PERSONA/PERSONA

- Chân dung người dùng "người dùng giả!"
- Chỉ xài khái niệm này nếu ta làm app theo kiểu PRODUCT!!! (App cho bá tánh xài – bất kì ai xài – game, tool, lazada, shopee, momo)
  + KO CÓ USER CỤ THỂ, VÌ USER CÓ THỂ LÀ BẤT KÌ AI
  + KO CÓ NGƯỜI ĐẶT HÀNG CỤ THỂ, VÌ CÓ THỂ LÀ BẤT KÌ AI
- Game dự định viết ra, bất kì ai thích game này đfeu có thể trả tiền để chơi!
  >>>> Ta viết game cho/theo nhu cầu của ai? RẤT KHÓ ĐOÁN
  >>>> App start-up cũng thuộc nhóm Product!

VIẾT APP CHO SỐ ĐÔNG THÌ PHỈA CÓ CÁCH TIẾP CẬN KHÁC, VÌ USER/CUSTOMER LÀ BẤT KÌ AI

- Làm cái survey/questionaire để điều tra thị trường, xem gu, xem trend
- Lục lọi, tìm kiếm trên MXH, Blog để tìm thông itn về nhu cầu của thị trường
- Xem các reviwer, bài comment bình luận chê khen 1 sản phẩm nào đó

- TA KHẢO SÁT THỊ TƯỜNG ĐỂ TÌM RA ĐƯỢC NHU CẦI CỦA 1 ĐÁM ĐÔNG NÀO ĐÓ

TA DỰNG LÊN 1 NGƯƠIG DÙNG GIẢ, CÓ ĐỦ INFO TÊN TUỔI ĐỊA CHỈ, THÓI QUEN, MONG ƯỚC ĐỂ CHO TEAM MEMBER DỄ HÌNH DUNG VỀ 1 USER CỤ THỂ, ĐẶT MÌNH TRONG VAI TRÒ USER ĐÓ ĐỂ THẤY HIỂU NHU CẦU CỦA HỌ

➔ TÌM RA TÍNH NĂNG

USER GIẢ NÀY ĐƯỢC GỌI LÀ USER PERSONA – CHÂN DUNG NGƯỜI DÙNG

VD: muốn làm 1 game về giáo dục

(ngược lại là app làm theo đặt hàng, thì cứ bám theo người đặt hàng mà hỏi chuyện)

6. NHỮNG NGƯỜI CÒN LẠI, SẾP CÁC BÊN

[CHỐT HẠ]

- USER/CUSTOMER LÀ 2 NHÂN VẬT CỰC KÌ QUAN TRỌNG, LÀM BA GIAO TEIEPS THƯỜNG XUYÊN ĐỂ HIỂU NHU CẦU CỦA HỌ!! VÀ SAU NÀY CHÍNH LÀ VIẾT PHẦN MỀM CHO HỌ DÙNG
- KO SUY NGHĨ VỀ ADMIN LÀ CÁC ÔNG SẾP, ADMIN LÀ USER/NGƯỜI ĐƯỢC THUÊ ĐỂ QUẢN LÍ DATA
- APP VIẾT CHO BÁ TÁNH THÌ KO CÓ AI CỤ THỂ ĐỂ MÀ HỎI, CHO NÊN PHẢI KHẢO SÁT BÁ TÁNH VÀ TÌM RA CHÂN DÙNG NGƯỜI DÙNG – USER PERSONAS
- APP VIẾT THEO ĐẶT HÀNG CỦA AI ĐÓ, THÌ CỨ BÁM HỌ MÀ HỎI NHU CẦU!

IX. CÁC CÔNG ĐOẠN/các bước, các hành động, CÔNG VIỆC KHI LÀM VỀ REQUIREMENTS SOFEWARE REQUIREMENTS ACTIVITIES ()

- Tìm ra yêu cầu phần mềm có 2 công đoạn lớn:

A. REQUIREMENTS DEVELOPMENT

1. *ELICITATION ********** quan trọng nhất (Nguyên lý kỹ thuật tìm REQs)*

❖ Reqs Elicitation: là các kỹ thuật, cách thức để tìm ra reqs có 10+ các kỹ thuật để tìm ra reqs

1.1 INTERVIEWING (phỏng vấn 1-1 với k/hàng, user)

⇨ LÀ PHƯƠNG PHÁP HIỆU QUẢ NHẤT HAY ĐƯỢC DÙNG NHẤT KHI LÀM VIỆC VỚI USER/CUSTOMER

- Muốn hiểu người khác, tốt nhất là giao tiếp, hỏi đáp!!!

1.2. WORKSHOP (làm việc với 1 nhóm k/h, user về app sẽ làm)
1.3. FOCUS GROUP (làm việc với 1 nhóm k/h, user về app sẽ làm)
1.4. QUESTIONAIRE (lập bảng câu hỏi, đi điều tra)
1.5. OBSERVATION/ETHNOGRAPHY (quan sát, cách user/khách hàng đang làm việc để hiểu AS IS)
1.6. DOCUMENT COLLECTION (xin, gom các tài liệu, document mà khách hàng đang dùng trong cv
AS IS đang lưu trữ xử lí info gì)
1.7. USER INTERFACE ANALYSIS (xem sản phẩm của các đối thủ, của các app khác đang có sẵn trên thị trường)
1.8. SYSTEM INTERFACE ANALYSIS (app của mình ko đứng 1 mình mà có tương tác với app khác!!!)
1.9. PROTOTYPING (vẽ phác thảo ý tưởng, UI)
1.10. BRAIN-STORMING (công não – cả team cùng suy nghĩ và đề xuất các tính năng)
....

➔ **TỔNG KẾT CHUNG CHO 10+ PHƯƠNG PHÁP:**

- 10+ phương pháp đều giúp tìm ra các chức năng, tính năng, tên màn hình mà user/customer sẽ dùng, trải nghiệm khi xài app sẽ làm.
- + phương pháp đều giúp tìm ra FUNCTIONAL, NON-FUNCTIONAL REQUIREMENTS.

❖ ***BÍ QUYẾT, THẦN CHÚ ĐỂ TÌM RA CÁC TÍNH NĂNG CỦA PHẦN MỀM:***

3 LOẠI CÂU HỎI BẠN – BA SẼ PHẢI TẬP TRUNG VÀO KHI LÀM VIỆC VỚI USER/KHÁCH HÀNG!!!

1. APP VIẾT CHO AI DÙNG, AI XÀI???

- Ko quan tâm mobile hay web app, hay desktop app, bạn hãy suy nghĩ xem đứng trước app sẽ có những loại user nào (phân quyền) xài app theo chức năng được phân công
- Nếu app ko cần login, mà vẫn xài được 1 số tính năng, màn hình, nút nhấn; ta gọi loại người dùng này là: GUEST, VISITOR,...
- Nếu người dùng cần phải login để xài, dùng các tính nẵng đắc biệt nào đó, thì nhận diện những người này theo: ROLE- VAI TRÒ

- Vi du: app FAP thì có:
  + ko có GUEST (hok login la ko xài đc, nên ko có guest)
  + nếu login thì Hoàng sẽ khác Nhân, Trường, Phát, Dũng, Minh...
  Lecturer student
- Vi du: app QLBV Chợ Rẫy thì có:
  + quầy nhận bệnh login vào đế tạo hồ sơ, thu tiền khảm RECEPTIONIST
  + bác sĩ dùng DOCTOR
  + bệnh nhân có xài ko??? tuỳ bv muốn bệnh nhân làm gì với app

book lich khám, xem lại đơn, ko cần số thì okie PATIENT sẽ là 1 user!!!

- Ví du: app thegioididong.com:

  + GUEST, ko cần login, trải nghiệm ngay đi

  + ??? AI KHÁC PHẢI LOGIN DỄ DÙNG???

  + CASHIER: thu ngân

  + SALES: nhân viên kinh doanh – cập nhật giá sản phẩm

  + ADMIN: ban người dùng hay spam!!

❖ **<u>MỘT APP BẤT KÌ LUÔN CÓ 2 NHÓM USER, HOẶC 2 KHỐI CHỨC NẰNG</u>**

- *<u>NHÓM USER THƯỜNG, RẤT ĐÔNG</u>!!!*

  + Nhóm này chủ yếu xài data, xem data, CRUD ko nhiều

  + Nhưng nhóm này là nhóm quan trọng, thường là đem về nguồn thu cho chủ app!!!

  + UI CHO NHÓM NÀY PHẢI ĐẸP, COOL

- *<u>NHÓM USER QUẢN TRỊ (MOD, MODERATOR, AD, ADMIN, ADMINISTRATOR), ÍT THÔI</u>!!!*

  + Nhóm này có quyền trên các data trong hệ thống, trong app

  + Nhóm này có quyền CRUD trên các loại data tùy chức năng

  + UI CHO NHÓM NÀY KO CẦN QUÁ COOL, ĐẸP

  + Bất kỳ ai mà có quyền thao tác (CRUD) nhiều trên data của app để data sẵn dùng cho mọi người còn lại, sẵn dùng cho đám user thường, những user này được xếp vào nhóm quản trị, quản lý (data) ko nhầm quản lí ngoài đời

  VD:

  + App TGDD.com thì nhóm user quản trị có thể bao gồm:

  . Mod/Moderator: quản trị viên, là người có quyền CRUD trên các câu hỏi của kkhachs mua hàng điện thoại

  . Cô thu ngân (cashier): có quyền tạo đơn hàng, quản lý đơn hàng (từ chung chung)

  . Cô tiếp tân: giúp khám phá sản phẩm điện thoai, tư vấn sp cho mình, giống mình chỉ xem data, xem thêm lịch sử mua hàng của mình

  VD2:

  + app FAP:

. sv là nhóm user thường; vì toàn dùng data của ng khác đưa cho

Có create 1 chút (ít gửi request/form; feedback, chọn môn học, đk lớp, đổi tkb lớp)

. giảng viên là nhóm user thường; còn kém hơn cả sv, chỉ được quyền điểm danh

. phòng đào tạo là admin, quản trị. Vì họ tạo được nhiều data cho gv, sv dùng.

+ app Lazada, Shopee:

- tui , bạn, người mua hàng là USER THƯỜNG, VÌ RẤT ÍT QUYỀN TẠO DATA.

  NẾU CÓ THÌ LÀ QUẢN LÍ GIỎ HÀNG CỦA CHÍNH MÌNH

  MANAGE (create, update, delete, payment)

- Chủ shop/owner: chính là admin của 1 shop, quản lí 1 shop

  CRUD món đồ họ bán, CRUD đơn hàng của họ
- Chủ sàn/admin tổng: quản lí toàn bộ hệ thống

  + ban 1 shop

---

2. AI (Ở TRÊN) SẼ LÀM ĐƯỢC GÌ, DÙNG ĐƯỢC GÌ, TRẢI NGHIỆM MÀN HÌNH GÌ CỦA APP (TÍNH NĂNG)

- Hỏi khách hàng/user/người đặt hàng làm app, mỗi loại user sẽ làm gì với app
DỰA trên AS IS SYSTEM ĐỂ SUY LUẬN RA
- Bác sĩ bình thường làm việc gì ở BV (AS IS) thì sau này convert cái gì thành tính năng ở TO BE
- Tiếp tân làm gì khi chưa có app? Sau này sẽ làm app – tính năng, màn hình

3. ĐỐI TƯỢNG DỮ LIỆU NÀO CẦN LƯU TRỮ, XỬ LÍ?

NHẬP MÔN USE CASE DIAGRAM: SƠ ĐỒ TÌNH HUỐNG SỬ DỤNG APP

- Là 1 loại sơ đồ thuộc trường phái vẽ tên là UML – Unified Modeling Language là 1 tập hợp các bản vẽ dùng cho các công đoạn làm phần mềm từ lúc reqs đến lúc bàn giao phần mềm cho user
- UCD là bản vẽ dùng ở giai đoạn REQS
- NÓ DÙNG ĐỂ LIỆT KÊ CÁC CHỨC NĂNG CỦA 1 PHẦN MỀM KÈM VỚI NHÓM USER XÀI CHỨC NĂNG ĐÓ!!

1. THÀNH PHẦN TẠO NÊN UCD

1.1 SYSTEM BOUNDARY (biên giới của app):

- là hình chữ nhật, đi kèm tên app đang sẽ làm

- HCN này giới hạn lại những tính năng của app sẽ làm
- Những tính năng nếu có của app sẽ nằm trong HCN; ko được vẽ các tính nnagw của app nằm ngoài hình chữ nhật. Những tính năng/màn hình /màn hình (hình esclip) sẽ nằm trong HCN

1.2 ACTOR (hình người rơm): TÁC NHÂN, ĐỨNG NGOÀI APP, CÓ TƯƠNG TÁC VỚI APP, VỚI CÁC TÍNH NĂNG

- Tác nhân chính là bao gồm

+ **Những NGƯỜI DÙNG sẽ xài app**

. NGƯỜI DÙNG THƯỜNG: guest, người dùng có account nhưng chủ yếu view data là chính

. NGƯỜI DÙNG QUYỀN LỰC có quyền CRUD data cho đám người dùng thường

. ad, admin, administrator, mod, moderator, manager... (có thể ứng với sếp ngoài đời, hoặc được mướn để quản lí data)

+ **NHỮNG ACTOR ĐẶC BIỆT KHÁC**

. NHỮNG APP KHÁC MÀ APP MÌNH CÓ TƯƠNG TÁC VỚI NÓ! CŨNG ĐƯỢC XEM LÀ ACTOR, vì actor được định nghĩa là những thứ ngoài app mà mình có chơi với app mình.

Vd: app mình có nối với MoMo để thanh toán, vậy MoMo được vẽ như 1 actor; app mình có nối với MoMo để thanh toán, vậy MoMo được vẽ như là 1 actor

. actor đặc biệt được gọi là: SYSTEM HANDLER

Dùng actor này khi app có làm những thứ TỰ ĐỘNG HÓA THEO LỊCH ĐỊNH SẴN, TỰ ĐỘNG HÓA KHI DATA ĐẠT 1 NGƯỠNG NÀO ĐÓ

Ví dụ app có tính năng:

- Tự động định kỳ hàng tuần lên trang TGDD, cellphoneS, hoanghamobile quét/cào data về thông tin bán các điện thoại iphone, samsung lưu vào db
- Order của khách sau 24 giờ ko xác nhận thanh toán hệ thống app mình tự hủy.đổi trang thái
- App có tính năng hẹn giờ, nhắc lịch: 3 PM thì reng nhắc uống thuốc TRÊN SERVER, TRONG APP CÀI 1 ĐOẠN CODE ĐÓNG VAI TRÒ NGƯỜI NHẮC, ĐOẠN CODE NÀY GỌI LÀ CRON JOB

(Đoạn code đóng vai ADMIN CLICK 1 HÀNH ĐỘNG ĐỂ APP CHẠY

NGƯỜI TƯƠNG TÁC VỚI APP ĐỂ ĐẨY VIỆC GÌ ĐÓ CHẠY GỌI LÀ ACTOR SYSTEM HANDLER)

. NHỮNG DEVICE CÓ TƯƠNG TÁC VỚI APP CHÚNG TA, ĐƯA DATA VÀO APP, TA CẦN NHẤN MẠNH VAI TRÒ CỦA DEVICE NÀY, THÌ TA VẼ LÀ 1 ACTOR:

VD: Camera an ninh trong hệ thống GIÁM SÁT

Camera chụp ảnh biển số xe trong app GIỮ XE

Sensor cảm biến trong app IOT

Máy barcode thì ko cần vẽ, vì nó chueyenr info đơn giản

QUY ƯỚC TÊN ACTOR:

NOUN Danh từ, gọi chung theo nhóm USER gọi chung theo ROLE khoong gọi tên riêng từng người Lan, Hồng, Huệ, Đào các bạn đều đứng ở quầy thu ngân các bạn được gọi là CASHIER/Thu Ngân Actor

1.3 USE CASE (hình Elipse) – Tình huống dùng app của user

- Nó chính là tên của 1 chức năng, 1 tình huống, 1 màn hình mà khi user xài chức năng này, sẽ đạt được điều gì đó
- Là 1 xử lý của app và phải trả về 1 kết quả gì đó cho người dùng
  Kết quả này được dùng trong công việc hàng ngày của người dùng

- Manh mối tìm:
  + VERB/WHAT, trả về 1 kết quả
  + Thường sẽ là: LƯU TRỮ, XỬ LÝ GÌ ĐÓ
  + Nó ứng với 1 hay 1 vài màn hình UI
- NÓ CÓ THỂ HIỂU TƯƠNG ĐƯƠNG LÀ: User story, functional, reqs, user reqs
  + VD: *As a student, I want to login by gmail so that I can ....*

❖ KO NHẦM LẪN USE CASE VỚI 1 STEP, 1 HOẠT ĐỘNG, THAO TÁC ĐỂ HOÀN TẤT USE CASE

HOẠT ĐỘNG, HÀNH ĐỘNG KO TRẢ VỀ KẾT QUẢ CÓ Ý NGHĨA KO GỌI LÀ USE CASE

(VD: Input user name là 1 step của LOGIN
Choose destination là 1 step của BOOK XE

⇨ Input user, Input pass: thỏa là động từ nhưng app ko xử lý gì để có ý nghĩa cho user do đó ko được là Use Case, NÓ CHỈ LÀ 1 STEP CỦA UC LOGIN!!
. LOGIN MỚI LÀ UC, gồm 3 step nhỏ: inpput username, input pass, và nhấn login button

⇨ Select country, select destination, select ticket type. LÀ STEPS CỦA UC: BOOK VÉ
.

❖ QUY ƯỚC TÊN UC, VERB + OBJECT, ĐỘNG TỪ KÈM BỔ NGHĨA: Login, Logout, Create an order, Search products, Add to cart, Manage cart, Update profile, Deactive account, Ban user,....

❖ TUYỆT ĐỐI KO ĐƯỢC CÓ 2 USE CASE TRÙNG TÊN NHAU! ~ 2 CHỨC NĂNG TRÙNG TÊN NHAU

○ Các actor đều xài chung 1 màn hình -> ok

1.4 NÉT NỐI (Association, Generalization)

- *SẼ nối giữ các actor với nhau, giữa các UC với nhau, actor với UC*
- **Giữa ACTOR với nhau**: CHỈ ĐƯỢC DÙNG NÉT NỐI GENERALIZATION, hàm ý 2 cha con, khi có actor chia sẻ, xài chung tính năng, tam giác rỗng hướng về cha, mối quan hệ IS A, đọc 1 chiều
- **Giữa ACTOR & USE CASE**: người dùng xài tính năng nào thì chỉ được dùng nét nối ASSOCIATION
- **Giữa các USE CASE với nhau**, TÍNH NĂNG với nhau: có 3 cách dùng nét nối

- Nét nối <GENERALIZATION>: cũng là cha con, con hướng tam giác rỗng về cha

⇨ Giữa các UC đọc được câu: IS A (KIND OF) 1 CHIỀU
. template/hay gặp: MANAGE X GỒM CON LÀ CRUD

VD: . Manage cart gồm các con: Add to cart, Update cart, Remove item form cart
. Manage item gồm các con: Add a new item, Update an item, Delete an item, Search items, Filter items

- Nét nối <INCLUDE>: diễn tả cho 2 mqh cực kì chặt chẽ, (Hắt xì: "base UC")
(BASE – UC CHÍNH) ←---------- <<include>>---------(UC PHỤ THUỘC, UC BỊ TRIỆU GỌI)
Đọc từ hướng chính đọc sang bên triệu gọi
Đọc từ gốc mũi tên
Tao cần mày lắm luôn vào giúp tao ------→ giúp tao
Included, inclusion
SỰ PHỤ THUỘC 2 THẰNG NÀY CỰC KỲ CHẶT CHẼ, LÀ SỐNG CÒN: BASE MUỐN LÀM XONG THÌ BẮT BUỘC PHẢI CÓ THẰNG TRIỆU GỌI CŨNG GIÚP
    - Dùng khi trong 1 UC có phần xử lí mà phần xử lí này được lặp lại ở UC khác Nên ta tách phần xử lí chung này ra 1 UC riêng, cho 2 thằng kia triệu gọi nó <<include>> như là thue viện cần dùng
    - Dùng để tách UC to thành những UC nhỏ hơn trực thuộc, mục tiêu dễ hiểu, dễ đọc tính năng!!

    (VD: * Đăng ký member/account/signup gồm 2 xử lý con

    - Lưu form xuống CSDL
    - Gửi email thông báo)

    *reset account/pass gồm 2 xử lý :
    - kiểm tra email có tồn tại

- Nét nối <EXTEND> (BASE – UC CHÍNH) ←---------- <<EXTEND>>---------(UC MỞ RỘNG)

- Đọc từ hướng chính đọc sang bên mở rộng
- Đọc từ ngọn mũi tên
- extend, extension, option, plugin

Dùng mũi tên này để liên kết 2 tính năng, từ tính năng chính, tiện đường ghé qua tính năng kia. UI thì là: màn hình này có link, menu, nút bấm gọi màn hình kia cho tiện!!

- SỰ PHỤ THUỘC CỦA EXTEND LÀ RẤT LỎNG LẺO
(BASE UC) LÀM XONG, CHẢ CẦN GỌI BÊN KIA (UC MỞ RỘNG), THÍCH THÌ GỌI, KO THÍCH THÌ THÔI

Ko nên dùng quá nhiều, gây rối sơ đồ!!

QUAN TRỌNG NHẤT SƠ ĐỒ UC PHẢI LIỆT KÊ ĐƯỢC CÁC TÍNH NĂNG APP CẦN CUNG CẤP (sự liên kết các tính năng, vẽ cũng được, ko vẽ cũng ko sao)

START UML

Dấu hiệu KẾ THỪA (tam giác rỗng): **is a kind of**

---

2. ***ANALYSIS***
3. ***DOCUMENTATION/SPECIFICATION***
4. ***VADIDATION***

B. REQUIREMENTS MANAGEMENT

THUYẾT TRÌNH SRS

Tìm trên mạng 1 tài liệu SRS – Software Requirement Specification (~1/2 cuốn đồ án tốt nghiệp của sv SE trường mình)

- Search Google:
  "Software Requirement Specification for Online Shopping example"

  for Warehouse Management example"

  For Video Game example"

Thay example = pdf docx

- Ko dùng tài liệu tiếng Việt, ko dùng tài liệu của các sinh viên trường mình
- Chỉ dùng tài liệu download về = tiếng anh, của nước ngoài, các công ty nước ngoài, của các nhóm sinh viên nước ngoài.
- Xem qua

❖ **THUYẾT TRÌNH**

- Mỗi nhóm 15'
- Trình bày cấu trúc của cuốn SRS – mục lục
- Nói tóm tắt nội dung cuốn này: có mục gì, mỗi mục nói về điều gì, nhấn những thứ gì nhóm nghĩ là hấp dẫn/hay
- Có thể nói lướt qua về các diagram xuất hiện trong SRS: Use Case, ERD, UI, kiến trúc code
- Ko cần giải thích chi tiết, ko cần dịch tất cả các nội dung, chỉ cần tóm lược để người nghe biết được:
- SRS có cấu trúc, có mục gì
- App làm điều gì?? Functional & nonfunctional reqs, UC, ER
- Slide vẫn cần show bản gốc của document

ÔN BÀI NHANH VỀ UCD – USECASE DIAGRAM

- Là sơ đồ mô tả về functional reqs, [NOOOOO…. Non-functional reqs], user story, chức năng, tính năng, tên màn hình, xử lý của app để giúp cho user làm được việc gì đó
- Là sơ đồ liệt kê các tính năng của app và người dùng trải nghiệm tính năng đó
- Sơ đồ này có 4 thành phần tạo nên nó:
  1. **SYSTEM BOUNDARY** – HCN, CÓ TÊN APP TRÊN CÙNG
  2. **ACTOR** – NGƯỜI DÙNG, NOUN, ROLE
  3. **USE CASE** – ELIPSE, OVAL, FUNCTION REQS, USER STORY, CHỨC NĂNG, TÍNH NĂNG, TÊN MÀN HÌNH, VERB + OBJECT, ví dụ: Create an Order
  4. **LINK, KẾT NỐI, GIỮA**
     - ACTOR vs. ACTOR: generalization
     - ACTOR vs. USE CASE: association
     - USE CASE vs. USE CASE

- generalization (Cha) <--------- (Con)
- <<include>> (Base, nền) ------<<include>>------> (include, inclusion, thứ phụ thuộc)
- <<extend>> (Base, nền) <-------<<extend>>-------(extend, extension, plugin)

USE CASE DESCRIPTION – ĐẶC TẢ USECASE LÀ GÌ?

- Một UC được nhận diện, được hiểu là 1 tính năng, 1 chức năng, 1 màn hình, user reqs, functional req, user story và LÀM 1 ĐIỀU GÌ ĐÓ cho user tương ứng

VD:

: để ghi nhận việc mua hàng, không có tính năng này, thì ta vẫn bán hàng, bill giấy viết tay.

: để ghi nhận việc bạn là người quen của app, bạn được phân quyền, làm đúng tính năng

- NẾU CHỈ CÓ (VERB + OBJECT) THÌ 1 UC CÒN KHÁ CHUNG CHUNG, MẶC DÙ TA BIẾT ĐƯỢC MỤC ĐÍCH CỦA UC QUA TÊN GỌI. Nhưng nó vẫn còn chung chung khi trao đổi với nhóm dev, user
- UC (LOGIN) còn chung chung vì: Login có nhiều hình thức và nó có những ràng buộc đi kèm cần mô tả bằng nhiều chữ hơn!!!

  ví mô tả nhiều hơn về (Login) để nó bớt chung chung

  - Login = Gmail, FB, account riêng của app
  - Nếu quên pass (acc riêng) thì vẫn có thể reset
  - Nếu chưa có acc hay không chắc, thì cứ tạo mới
  - Thất bại login do sai pass 5 lần thì lock lại 30p

- **KHI THIẾT KẾ PHẦN MỀM, ĐẾN ĐOẠN VẼ UCD**, GIANG HỒ THƯỜNG LÀM THÊM 1 VIỆC SAU - ỨNG VỚI MỖI UC, HÌNH OVAL, TA **LUÔN ĐÍNH KÈM THEO 1 PHIẾU** (giấy, table trong Word) mô tả chi tiết hơn về UC đó. Phiếu này có thể viết tự do hoặc viết theo template có sẵn.
  - Thường giang hồ viết theo template có sẵn để thống nhất nội dung cho anh em trong team dễ đọc, và để giúp BA khi viết tránh sai sót
  - PHIẾU MÔ TẢ CHI TIẾT CHO TỪNG UC (giống như cuốn sổ tay User Manual hay đính kèm trong hộp điện thoại) ĐƯỢC GỌI LÀ USE CASE DESCRIPTION – ĐẶC TẢ USE CASE.

- **Một đặc tả UC sẽ gồm những nội dung chính như sau (~Hồ sơ cá nhân gồm những info cơ bản...)**
  1. Mã số, số thứ tự UC: thường đánh số UC-001, UC-002...
  2. Tên UC: Create an order, Reset password, Ban a user, Add new an item, Pay an order
  3. Ngày giờ tạo ra phiếu này: _____
  4. Ai tạo phiếu này: _____tên gã BA
  5. Ai xài UC này: _____
  6. Tóm tắt về UC này: _____
  7. Điều kiện tiên quyết để UC này chạy được (pre-conditions): _____
  8. Sau khi chạy xong UC này, tính năng done, thì app sẽ thế nào (post-condition): _____
  9. Trigger: khi nào UC này được gọi
  10. Main flow:
  11. Alternative flow:
  12. Exception flow:
  13. Business Rules

GHI CHÚ VÀI ĐIỀU VỀ UC DESCRIPTION

1. **TRIGGER:**
   - Q: Khi nào cái UC của mình đang mô tả nó được run, khi nào tính năng này được sử dụng?
   - A: Khi user xài nó cho mục đích họ muốn
     Viết câu tiếng Anh mô tả mong ước/mục đích của user khi họ xài tính năng!!

     The user/actor wants to...
     The user/actor indicates that he/she wants to...

❖ VD:

- (Create a prescription – tạo một đơn thuốc)
  - The doctor wants to store/create his prescription during the treatment process for a patient

- (CREATE AN ORDER) – THANH TOÁN TẠI QUẦY
    - TRIGGER:
    - The cashier indicates that he/she wants to store the buying transaction/shopping of a specification customer
    - DESCRIPTION OF (CREATE AN ORDER)
    - To store/record the buying transaction, the cashier must use the feature Create an Order first. In this feature, the cashier will input the buying items by using touch screen/or barcode reader.

2. **PRE-CONDITIONS** – để chạy được uc này, cần những thứ gì trước đó, điều kiện tiên quyết
    - Liệt kê data, thiết bị, thứ cần có trước khi run UC
    - Nếu xem UC là tên hàm, thì pre-condition là tham số đưa vào