Seminar Notes

*Dưới đây là những ghi chép bổ trợ thêm cho bài nói của 2 speaker trong Seminar "**SRS – Common Mistakes & Best Practices**" diễn ra vào tối 05/04/2022 vừa qua trên Google Meet.*

Đa tạ tất cả sinh viên thân yêu và các phòng ban liên quan đã nhiệt tình giúp đỡ, ủng hộ, hỗ trợ chúng tôi hoàn thành tốt đẹp buổi thuyết trình. <3

Mục I. Project Introduction của Capstone Project (Khoá luận tốt nghiệp)

- **Stakeholders:** Các bên liên quan trong quá trình làm dự án phần mềm, bao gồm khách hàng – doanh nghiệp đặt hàng phần mềm, các nhân viên phía khách hàng – người thụ hưởng – dùng phần mềm sau này, các phòng ban trong nội bộ công ty (nếu viết App cho nội bộ công ty sử dụng)..., và cả Dev team. Dự án phần mềm cần phải "deal", "cân đối" và làm hài lòng Requirements của các bên liên quan.
- **Scope & Limitations:** Scope – tầm vực – phạm vi của dự án, là những gì ta nhắm tới – dự định sẽ làm, những tính năng nào của phần mềm mà ta sẽ dự định cung cấp cho người dùng.
- Xác định được Scope là câu chuyện khó khăn. Bài toán quá lớn – hứa thật nhiều thất hứa thật nhiều – **Over-the-scope, Out-of-the-scope** hay **Scope-creep** (mất kiểm soát bài toán, lần mò/mò mẫm trong Scope) sẽ gây ra nhiều vấn đề, hệ luỵ cho các bên liên quan.
- Tên gọi của dự án tuy là cụm từ ngắn ngủi, nhưng phân tích chi tiết có khi là cả một Hệ thống bự "chà bá lửa". Ví dụ "Phần mềm Quản lí hoạt động doanh nghiệp" bản chất là dự án siêu to khổng lồ, nó có thể gồm vài, hay tất cả các module – bài toán – dự án con như liệt kê dưới đây, mà mỗi thứ trong số chúng đều xứng đáng là một project đủ hại não:
  - ✓ Quản lí nhân sự (& Quản lí tiền lương?)
  - ✓ Quản lí kế toán
  - ✓ Quản lí khách hàng
  - ✓ Quản lí bán hàng
  - ✓ Quản lí nhà cung cấp – chuỗi cung ứng
  - ✓ Quản lí kho
  - ✓ Quản lí tài sản cố định
  - ✓ Quản lí sản xuất
  - ✓ …

  Xét riêng "Quản lí nhân sự", lại bao gồm N việc khác bên trong như: Quản lí hồ sơ nhân sự (fulltime, partime, tập sự, thực tập, thuê người, cho thuê người,…); Quản lí tiền lương (chấm công, các thành phần của lương: lương căn bản, phụ cấp, hoa hồng, lương ăn theo sản phẩm,

ngạch lương, bậc lương, BHYT, BHXH, tạm ứng…); lương thì dính đến tiền, tiền thì dính đến Kế toán!!!

- Các thầy cô ở FU.HCM sẽ cùng giúp các bạn điều chỉnh Scope ở các vòng review, đừng lo.
- Ở ngoài đời sai lầm trong Scope sẽ trả giá bằng tiền túi của bạn; vì khách hàng không trả thêm cho phần bạn sai lầm hay phần bạn tự vẽ ra.
- **Question của sinh viên:** Đề tài em làm về XYZ, chủ yếu là quản lí data. Em có cảm giác nhỏ và đơn điệu, mong các thầy giải đáp.

  Answer của các thầy: Đề tài to hay nhỏ căn cứ theo thời gian làm dự án, tổng số tính năng App cung cấp cho mỗi loại user, số loại user và số lượng user được App hỗ trợ.

  Một thang đo được dùng để ước lượng độ lớn của dự án thông qua độ lớn của các Use Case được gọi là **Use Case Points** – **UCP**.

  URL tham khảo: https://www.tutorialspoint.com/estimation_techniques/estimation_techniques_use_case_points.htm

  Chúng ta sẽ không nói App có đơn điệu hay không mà hãy tự trả lời câu hỏi:

  App sinh ra để làm gì, có giải quyết đúng "chỗ ngứa, issues, problems, obstacles" của thị trường, của doanh nghiệp, của người dùng hay không? App có giải quyết được "Customers' Pain Points", nỗi đau của khách hàng hay không?

  Ta bán cái "giang hồ" cần. Ở cảnh giới cao hơn, Steve Jobs từng nói: "Sell Dreams, NOT Products", bán cái khách hàng đang mơ, bán cái biến giấc mơ của khách hàng thành hiện thực.

  Thách thức danh hài: làm cái người ta chưa từng nghĩ đến và bán được cái đó. Đẳng cấp bán sản phẩm: tạo ra nhu cầu!!!

- Cũng không nên "bi quan, yếm thế" khi so sánh App mình với App người ta và nói rằng App mình "kém" vì không có thuật toán cao cấp, phức tạp, đủ tầm.
- **Nên nhớ:** thuật toán chỉ là kĩ thuật giải quyết bài toán thôi mà, quan tâm chi chuyện to nhỏ cao sang. Quan trọng là bài toán có ích – đem lại ích lợi cho người sử dụng!!!

  Giải phương trình bậc 2 dùng Delta là một thuật toán; dùng công thức Viète tính nhanh nghiệm phương trình bậc 2 cũng là một thuật toán.

  Đừng nâng cao chữ thuật toán quá mức cần thiết!!! Điều quan trọng ở đây là người dùng có thích dùng App, sẵn lòng giới thiệu App cho người khác dùng? Đấy là App thành công!!!

Mục II. Project Management Plan của Capstone Project

- **Quản lí dự án:** Phương pháp luận phát triển phần mềm – Quy trình làm phần mềm – Software Development Methodology/Method/Model/Process, có các trường phái chính:
  - ✓ **Traditional**: Waterfall, V-Model, Sashimi…
  - ✓ **Agile**: Scrum, XP, Kanban, Lean, Lean Startup…
- **CI/CD/DevOps**: Liên quan đến quản lí các tài nguyên dự án; quản lí chất lượng code – chất lượng sản phẩm; và quản lí sản phẩm đầu cuối; quản lí mọi thứ từ lúc bắt đầu dự án đến lúc chuyển giao sản phẩm đến tay khách hàng hay sản phẩm go-live.

- Trong document của Capstone Project, khi viết về việc lựa chọn Quy trình phát triển phần mềm, cần có những câu lí luận, giải trình cụ thể để làm rõ tại sao bạn lại chọn phương pháp này mà không phải phương pháp kia, kiểu như:
  - ✓ Phải có câu chữ nói lên ý nghĩa đặc trưng của dự án đang làm.
  - ✓ Dự án đang làm có những đặc điểm gì để phù hợp với quy trình phát triển phần mềm được lựa chọn.
  - ✓ Waterfall: thường là Requirements đã ổn định, rõ ràng, khách hàng nói rõ được mong đợi của họ, định nghĩa được ngay những thứ họ cần.
  - ✓ Agile: Requirements còn mơ hồ, chưa rõ ràng, cần có prototype/pilot version để "dò" ý khách hàng.
  - ✓ Cách khách hàng muốn nhìn kết quả sản phẩm thế nào: xem sớm kết quả của sản phẩm một cách định kì, liên tục; muốn thấy tiến độ thực của dự án; hay xem kết quả ở 1 thời điểm đủ xa khi mọi thứ đã "hòm hòm".
  - ✓ Kĩ năng hiện có của các thành viên có phù hợp với đặc trưng của phương pháp lựa chọn?
  - ✓ Loại công nghệ nhóm bị áp đặt hay được chủ động lựa chọn; nhóm có quen công nghệ này hay không? Hay nhóm muốn thử nghiệm giải pháp, công nghệ trong quá trình phát triển sản phẩm?
- **Không copy** những câu vô cảm trên mạng, chung chung theo kiểu lí thuyết, mà phải phù hợp đặc trưng của dự án bạn đang làm!!! Thường anh em hay có 4 đến 5 cái gạch đầu dòng cho việc chọn lựa Scrum, nghe vô cảm, quen, nhàm kiểu clone từ document này sang document khác vậy!!!
- Ví dụ gợi ý về câu chữ cho phần lựa chọn Quy trình phát triển phần mềm:
  - ✓ App chúng tôi là App startup nên Requirements đang định hình; cần brain-storming, khảo sát thị trường/nhà đầu tư/user tiềm năng; cần lấy feedback liên tục sau mỗi chu trình thời gian để kịp điều chỉnh hướng nên phương pháp XYZ là phù hợp.
  - ✓ Team muốn cơ hội thử nghiệm về công nghệ mới, và phương pháp XYZ cho phép điều đó.
- Cần có bằng chứng chứng minh rằng bạn đã có áp dụng hay biến tấu 1 quy trình nào đó; và bạn cần hiểu – nắm được các thuật ngữ, khái niệm của công nghệ/kĩ thuật bạn đã áp dụng.
- **Đừng** lo lắng quá mức và "rào đón" cách các thầy hội đồng sẽ phản biện, hay tâm thế sợ sai, sợ bị soi từng ngóc ngách nhỏ. **Không cần vậy**!!! Chỉ cần bạn hiểu cách Quản trị dự án, bạn biết bạn đang làm gì là được rồi!
- Hội đồng chấm cần bằng chứng, hơn là soi mói, bắt lỗi tiểu tiết. **Nhớ đừng sai** kiến thức nền, kiến thức cơ sở hay lơ-tơ-mơ về nó, **khó cứu vớt**!!!
- Nếu chọn phương pháp Scrum, cần viết ra các bằng chứng liên quan đến đặc trưng Scrum trong tài liệu, ví dụ: Phân vai trò trong dự án – Roles, Product Backlog – một file Excel hay màn hình chụp các User Stories trên các tool hay công cụ nhóm đã sử dụng (Jira/Trello).
  - ✓ Đừng quên **User Story Template**:

    ```
    As a (who wants to accomplish something); I want to (what they want to accomplish); So that (why they want to accomplish that thing)
    ```

Mục III. Software Requirements Specification của Capstone Project

- Độc giả của **User Requirements**: Người dùng – khách hàng, là thế giới bên ngoài Dev team; và chính Dev team.
- Độc giả của **Use Case Diagram**: Chủ yếu là Dev team.
- **Use Case Diagram:** Một "khẩu quyết" khác tìm để Use Case (UC): **Tĩnh (Entity) có trước, động (Action – Use Case) có sau**.
  - ✓ Actor thực hiện các Action là các Hành động; và các Action này tác động trực tiếp lên các Thực thể – Entity thông qua nền tảng web hoặc mobile.
  - ✓ Entity chính là những thành phần tĩnh về mặt dữ liệu, là khởi nguồn để có ERD – Entity Relationship Diagram.
  - ✓ Sơ đồ Use Case chính là "phép nhân" của số Actor đấu chéo móc nối sang các Hành động trên các Thực thể.
  - ✓ Các Hành động – Nghiệp vụ của khách hàng – cách khách hàng giải quyết công việc hàng ngày của họ là vô chừng, nhưng các Entity dữ liệu "có vẻ" ổn định hơn về số lượng.

    ERD tuần sau thầy TàiNT và thầy PhươngLHK sẽ trình bày, đón xem

  - ✓ Ở mức khởi đầu, xác định cái cố định – tĩnh trong hệ thống sẽ dễ dàng hơn vì các thông tin cần lưu trữ đã có sẵn trên bàn làm việc, trong ngăn bàn của mỗi User/Actor sau này.
- **Business Rules (ràng buộc nghiệp vụ)**: Là những câu phát biểu của khách hàng/user đưa ra để gài/khống chế/ràng buộc cái ngữ cảnh hoạt động của một tác vụ/công việc/tính năng/chức năng mà người dùng vẫn làm và thao tác hàng ngày; còn gọi là những quy tắc xử lí thông tin.

  Ví dụ 1: Tính năng mượn sách, UC (Borrow books) sẽ có vài Business Rules được gài vào như liệt kê ở dưới đây, dùng để phản ánh cái nghiệp vụ hay những ràng buộc khi tiến hành công việc cho mượn sách trong môi trường đời thực mà cô thủ thư vẫn quen làm hàng ngày:
  - ✓ **BR1:** Sinh viên không được mượn sách của học kì mới nếu chưa trả sách của học kì cũ.
  - ✓ **BR2:** Sinh viên tại bất kì thời điểm nào chỉ được mượn tối đa 5 cuốn sách ngoài sách học của học kì. Nếu đang sẵn mượn 3 cuốn chưa trả, thì bữa nay hay những ngày sau chỉ được mượn thêm tối đa 2 cuốn!

  Ví dụ 2: Một Business Rule hay được gài cho UC (Đăng kí member): Password phải có ít nhất 12 kí tự và phải đồng thời chứa những kí tự sau: số, chữ cái hoa và thường, kí tự đặc biệt, không chứa 3 chữ cái liên tiếp trùng trong email đăng kí và fullname. Định kì 2 tháng user phải đổi password, và password mới không được trùng lại với tất cả các password cũ đã từng sử dụng.
- **Use Case Specification – Use Case Description – Bản đặc tả Use Case**:

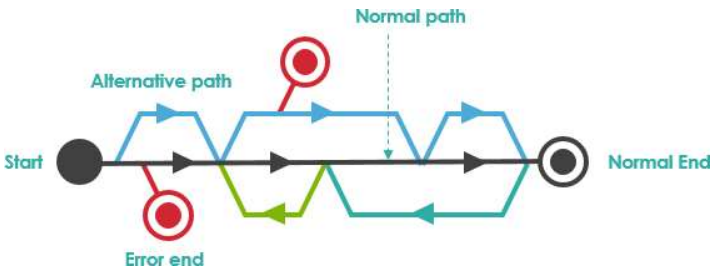
  Vì Use Case (UC) chỉ nói tên hành động, tên chức năng, tên gọi màn hình, what/target/goal đạt được của user, do đó nó quá kiệm lời/ngắn gọn để Dev team có thể hiện thực hoá hay cài đặt code cho nó.

  Cần phải có một bản ghi chép/mô tả chi tiết hơn về kịch bản hành xử, xử lí phía hậu trường của App khi Actor tương tác với UC, hay khi UC được thực thi, khi tính năng được sử dụng.

Cũng cần liệt kê các cài cắm ràng buộc dữ liệu/nghiệp vụ – business rule để Dev team hình dung ra cách cài đặt/viết code cho UC trong từng xử lí nhỏ; và cũng giúp cho Tester/QC dễ dàng thiết kế các Test Case sau này.

Bản mô tả chi tiết cho từng UC gọi là Use Case Specification/Use Case Description.

- Mỗi UC sẽ có một UC Description, được viết dưới dạng thuần text hay thuần text nhưng nằm trong một cái form – bảng biểu table cho dễ đọc. Dạng form – bảng biểu được **ưa thích hơn**.
- Nếu viết UC Description dưới dạng bảng biểu thì có 2 style trình bày, sự khác biệt giữa chúng nằm ở phân đoạn mô tả kịch bản tương tác giữa Actor và System.
  - ✓ Kịch bản tương tác Actor – System viết theo kiểu **tuyến tính liệt kê** các bước tương tác.
  - ✓ Kịch bản tương tác Actor – System viết theo kiểu **chia 2 cột** cho 2 bên **"bắn"** qua **"bắn"**lại.
- **Cách chia cột sẽ dễ viết, dễ theo dõi, dễ đọc hơn, "Mlem".**
- Hình bên mô tả kịch bản tương tác diễn ra giữa Actor và System; nó sẽ xuất hiện trong UC Description dù bạn viết theo style nào, mang ý nghĩa: UC bắt đầu thực thi, và UC sẽ kết thúc theo nhiều cách như dự kiến, và có thể bị chết giữa đường.



Source: https://www.visual–paradigm.com/guide/use–case/what–is–use–case–specification/

- **Ví dụ 1: Minh họa về UC Description viết theo style tuyến tính trên "giang hồ mạng":**

  URL tham khảo: https://www.visual–paradigm.com/guide/use–case/what–is–use–case–specification/

- **Ví dụ 2: Minh họa về UC Description viết theo style tuyến tính – theo template từ FU Hà Nội:**

Use Case Specification			
Use Case ID and Name:			
Created By:		**Date Created:**	
Primary Actor:		**Secondary Actors:**	
Trigger:			
Description:			
Preconditions:	**1.**		
Post–conditions:	**1.**		
Normal Flow:	**1.**		
Alternative Flows:			
Exceptions:			
Priority:	**High (Medium, Low)**		
Frequency of Use:			
Business Rules:			
Other Information:			
Assumptions:			

- **Ví dụ 3: Minh họa về UC Description viết theo style 2 cột bắn qua bắn lại – "Mlem":**

Use Case Specification

Use Case No.:	<Mã số UC>		
Use Case Name:	<Tên UC>		
Created By:	<Ai viết UC Spec này>		
Date:	<Ngày viết>	**Priority:**	<Mức độ ưu tiên cần hiện thực tính năng này, ví dụ: Must Have | High>
Actors:	<Những user nào sử dụng tính năng này>		
Summary:	<Mô tả ngắn gọn mục đích của UC này>		
Trigger:	<**Điều gì khiến** UC này được gọi, **mục đích** của việc sử dụng UC; *ví dụ: The admin indicates that he wants to udate a product*>		
Preconditions:	<Điều kiện tiên quyết cần có trước đó để UC này có thể chạy, *ví dụ: data/thiết bị sẵn dùng là…; user cần phải login trước khi sử dụng*>		
Post–conditions:	<Sau khi UC thực thi xong và thành công, hiện trạng hệ thống là gì, user đạt được điều gì; có thể liệt kê thêm kết quả của tình huống UC thực thi thất bại, *ví dụ: Đơn hàng được ghi nhận và lưu trữ*>		

Main Success Scenario/Main Flow/Normal Flow/Main Path:

Step.	Actor Action	System Response
1	<Luồng xử lí chính, trường hợp Happy Case, người dùng hay làm những điều này để đạt được mục đích UC như đã thiết kế> <Bước 1 người dùng làm gì/nhấn gì/nhập gì>	<Hệ thống xử lí/lưu trữ/hồi đáp lại người dùng cái gì>
2	<Bước 2 người dùng làm gì/nhấn gì/nhập gì tiếp> **[Alternative 1]** Có thể người dùng chọn rẽ nhánh khác để cùng đạt được mục đích UC. Các [Alternative X] được đánh số thứ tự tăng dần	<Hệ thống xử lí/lưu trữ/hồi đáp lại người dùng cái gì> **[Exception 1]** Có thể có tình huống ngoại lệ xảy ra ở bước này Các [Exception X] được đánh số thứ tự tăng dần
3	…	…

Alternative Flows:

No.	Actor Action	System Response
1	<Các tình huống rẽ nhánh, người dùng làm gì tiếp, hệ thống phản hồi gì để đạt cùng mục đích UC>	
2	…	…

Exceptions:

No.	Actor Action	System Response
1	<Các tình huống ngoại lệ, người dùng làm gì tiếp, hệ thống phản hồi theo kiểu để UC kết thúc không thành công, App hạ cánh an toàn>	
2	…	…

Business Rules:	<Những câu phát biểu của khách hàng/user đưa ra để gài/khống chế/ràng buộc cái ngữ cảnh hoạt động của một tác vụ/công việc/tính năng/chức năng mà người dùng vẫn làm và thao tác hàng ngày; còn gọi là những quy tắc xử lí thông tin, *ví dụ 1: Ngày giờ phải theo định dạng dd/mm/yyyy HH:MM, giờ đồng hồ theo thang đo 24 giờ; ví dụ 2: Đơn hàng từ 1.000.000đ sẽ giảm giá x% theo thông tin khuyến mãi tại thời điểm tạo đơn*>

- **Ví dụ 4: Xem minh họa về UC Description chuẩn của Microsoft về mặt nội dung, viết theo style tuyến tính – đề nghị anh em FU làm theo nhé, nếu cần thì độ phần chia cột!!!**

  (trích trong giáo trình Software Requirements, Karl Wiegers and Joy Beatty, 3rd, Microsoft Press, 2013, p.581)

  UC Description dưới đây mô tả cách người dùng sử dụng tính năng Đặt bữa ăn ở canteen công ty qua App tên là **Cafeteria Ordering System (COS)**

ID and Name:	**UC-1: Order a Meal**		
Created By:	Prithvi Raj	Date Created:	October 4, 2013
Primary Actor:	Patron	Secondary Actors:	Cafeteria Inventory System
Description:	A Patron accesses the Cafeteria Ordering System from either the corporate intranet or external Internet, views the menu for a specific date, selects food items, and places an order for a meal to be picked up in the cafeteria or delivered to a specified location within a specified 15-minute time window.		
Trigger:	A Patron indicates that he wants to order a meal.		
Preconditions:	PRE-1. Patron is logged into COS. PRE-2. Patron is registered for meal payments by payroll deduction.		
Postconditions:	POST-1. Meal order is stored in COS with a status of "Accepted." POST-2. Inventory of available food items is updated to reflect items in this order. POST-3. Remaining delivery capacity for the requested time window is updated.		
Normal Flow:	**1.0 Order a Single Meal** 1. Patron asks to view menu for a specific date. (see 1.0.E1, 1.0.E2) 2. COS displays menu of available food items and the daily special. 3. Patron selects one or more food items from menu. (see 1.1) 4. Patron indicates that meal order is complete. (see 1.2) 5. COS displays ordered menu items, individual prices, and total price, including taxes and delivery charge. 6. Patron either confirms meal order (continue normal flow) or requests to modify meal order (return to step 2). 7. COS displays available delivery times for the delivery date. 8. Patron selects a delivery time and specifies the delivery location. 9. Patron specifies payment method. 10. COS confirms acceptance of the order. 11. COS sends Patron an email message confirming order details, price, and delivery instructions. 12. COS stores order, sends food item information to Cafeteria Inventory System, and updates available delivery times.		
Alternative Flows:	**1.1 Order multiple identical meals** 1. Patron requests a specified number of identical meals. (see 1.1.E1) 2. Return to step 4 of normal flow. **1.2 Order multiple meals** 1. Patron asks to order another meal. 2. Return to step 1 of normal flow.		
Exceptions:	**1.0.E1 Requested date is today and current time is after today's order cutoff time** 1. COS informs Patron that it's too late to place an order for today. 2a. If Patron cancels the meal ordering process, then COS terminates use case. 2b. Else if Patron requests another date, then COS restarts use case. **1.0.E2 No delivery times left** 1. COS informs Patron that no delivery times are available for the meal date. 2a. If Patron cancels the meal ordering process, then COS terminates use case. 2b. Else if Patron requests to pick the order up at the cafeteria, then continue with normal flow, but skip steps 7 and 8. **1.1.E1 Insufficient inventory to fulfill multiple meal order** 1. COS informs Patron of the maximum number of identical meals he can order, based on current available inventory. 2a. If Patron modifies number of meals ordered, then return to step 4 of normal flow. 2b. Else if Patron cancels the meal ordering process, then COS terminates use case.		
Priority:	High		
Frequency of Use:	Approximately 300 users, average of one usage per day. Peak usage load for this use case is between 9:00 A.M. and 10:00 A.M. local time.		
Business Rules:	BR-1, BR-2, BR-3, BR-4, BR-11, BR-12, BR-33		
Other Information:	1. Patron shall be able to cancel the meal ordering process at any time prior to confirming it. 2. Patron shall be able to view all meals he ordered within the previous six months and repeat one of those meals as the new order, provided that all food items are available on the menu for the requested delivery date. (Priority = medium) *[Note: You could also show this as an alternative flow for the use case.]* 3. The default date is the current date if the Patron is using the system before today's order cutoff time. Otherwise, the default date is the next day that the cafeteria is open.		
Assumptions:	Assume that 15 percent of Patrons will order the daily special (Source: previous 6 months of cafeteria data).		

2 UC Description dưới đây mô tả cách người dùng sử dụng tính năng Đăng kí trả tiền ăn qua khấu trừ lương tháng; Chỉnh sửa thực đơn. Tác giả có giải thích họ viết vắn tắt hơn UC Description ở trên…

[Note: the following use case is written in less detail than UC-1, to illustrate that it isn't always necessary to fully specify every detail of the use case, provided developers have the necessary information available from some other source.]

ID and Name:	**UC-5 Register for Payroll Deduction**		
Created By:	Nancy Anderson	Date Created:	September 15, 2013
Primary Actor:	Patron	Secondary Actors:	Payroll System
Description:	Cafeteria patrons who use the COS and have meals delivered must be registered for payroll deduction. For noncash purchases made through the COS, the cafeteria will issue a payment request to the Payroll System, which will deduct the meal costs from the next scheduled employee payday direct deposit.		
Trigger:	Patron requests to register for payroll deduction, or Patron says yes when COS asks if he wants to register.		
Preconditions:	PRE-1. Patron is logged into COS.		
Postconditions:	POST-1. Patron is registered for payroll deduction.		
Normal Flow:	**5.0 Register for Payroll Deduction** 1. COS asks Payroll System if Patron is eligible to register for payroll deduction. 2. Payroll System confirms that Patron is eligible to register for payroll deduction. 3. COS asks Patron to confirm his desire to register for payroll deduction. 4. If so, COS asks Payroll System to establish payroll deduction for Patron. 5. Payroll System confirms that payroll deduction is established. 6. COS informs Patron that payroll deduction is established.		
Alternative Flows:	None		
Exceptions:	5.0.E1 Patron is not eligible for payroll deduction. 5.0.E2 Patron is already enrolled for payroll deduction.		
Priority:	High		
Business Rules:	BR-86 and BR-88 govern an employee's eligibility to enroll for payroll deduction.		
Other Information:	Expect high frequency of executing this use case within first 2 weeks after system is released.		

[Note: the following use case is written in a very brief form, to illustrate that it is not always necessary to fully complete the use case template, provided developers have the necessary information available from some other source. It's a good idea to plan out which use cases require detailing and which do not.]

ID and Name:	**UC-9 Modify a Menu**		
Created By:	Mark Hassall	Date Created:	October 7, 2013
Description:	The cafeteria Menu Manager may retrieve the menu for a specific date in the future, modify it to add new food items, remove or change food items, create or change a meal special, or change prices, and save the modified menu.		
Exceptions:	No menu exists for the specified date; show an error message and let the Menu Manager enter a new date.		
Priority:	High		
Business Rules:	BR-24		
Other Information:	Certain food items will not be deliverable, so the menu presented to the Patrons of the COS for delivery will not always exactly match the menu available for pickup in the cafeteria. The Menu Manager can set which items are not deliverable.		

HẸP GẶP CẢ NHÀ TRONG SEMINAR TIẾP THEO

TỔ CHỨC VÀO NGÀY 12/4/2022 20:00 TRÊN NỀN TẢNG GOOGLE MEET <3