* MEETING 01/12/2010
* REQUIREMENTS
  + Xem lại các vấn đề về requirements
    - Đã đầy đủ chưa: phần trả lời của anh hiếu (mail 30/11)
    - Có gì conflict không: tạm thời chưa
    - Có yêu cầu nào bị thừa không: chưa
    - Có yêu cầu nào không khả thi không: không
  + Tổng kết lại các requirements
    - Phân chia lại module lớn cho chuẩn
    - Đặt tên và cách thức traceability cũng như tracking thật tốt
      * Khi mà làm QA và design thì sẽ chỉnh sửa lại reqs.
      * Với sai sót req nào thì sẽ refer đến người làm req đó
      * phải có người review
      * có ảnh hưởng gì đến những phần đã có không?
* QUALITY ASSUARANCE
  + Giang và Thảo sẽ giới thiệu chung về công việc QA cho nhóm dự án
    - Các templates
      * Test case: ok
      * Risk management:
        + có template,
        + chưa define rõ rang nội dung.
        + Giang update.
        + Tìm hiểu thêm về ý nghĩa các cột mitigation và contigency
      * Bug management:
        + Chia các bug theo phase và module
        + Ai log lỗi sẽ có trách nhiệm inform người được assign và track xem lỗi đã được fix hay chưa.
      * Template log defect
        + Phân biệt defect, error, bug, issue v..v
        + Merge với bug management
    - Các định hướng trong việc develop TC và các công việc liên quan
      * IT design làm song song với DDD
      * IT và DDD hỗ trợ update SRS
    - Cách thức quản lý quy trình, log và xử lý lỗi
  + Các vấn đề nêu ra cho công việc QA
    - Integration test
      * Bám vào requirements có đầy đủ hết không
      * Bám vào design như thế nào
      * Cách thức chia function trong test case
      * Giang and Thao; start integration test
    - Unit test
      * Ai sẽ làm unit test: coder
      * Làm unit test design trước hay sau khi code: người làm detail design sẽ làm luôn unit test design. Người code sẽ bổ sung thêm nếu cần.
      * Cách thức làm thế nào
    - Cách thức mà nó liên quan đến requirements và design  
      Người làm design sẽ test và review, có lỗi sẽ chuyển đến cho người code phụ trách sửa.  
      Người code có vấn đề tì sẽ refer lại design, sau đó refer đến TC và SRS
* ARCHITECTURAL DESIGN
  + Long propose các mô hình có thể sử dụng, các architectural styles
    - Data flow:
      * Batch sequential
      * Pipe and filter
    - Implicit communication
      * Non-buffered event-based
      * Buffered message based
    - Hierarchy
      * Layer architecture design
      * Master-slave architecture design
      * Main-subroutine
    - Data center
      * Repository
      * Black-board
    - Interaction oriented
      * Presentation abstract controller
      * MVC
  + Cả nhóm sẽ thảo luận và lựa chọn các mô hình thích hợp
    - Đảm bảo quality attributes
    - Đảm bảo yêu cầu tính năng
  + Trong buổi họp phải thống nhất được mô hình, sau đó về nhà tiếp tục phát triển
    - Phần data center: A+B: Repository
    - Component-based: A+B (Dùng cho toàn bộ system)
    - Architecture Model dừng lại ở 2 tầng:
      * Architecture Model riêng cho module A và B
      * Architecture Model riêng cho các sub-module của A và B
    - Cân nhắc về MVC: Chính đưa ra 1 giải pháp demo để team xét tính khả thi
    - Module A: Component-based
      * System
      * Input
      * Quản lý tham số
      * Chấm điểm
      * Báo cáo
    - Module A
  + Các câu hỏi cho anh Hiếu
    - Người dùng hệ thống: số lượng nhiều hay ít?
    - Độ bảo mật
    - Độ tin cậy
    - Tốc độ xử lý
  + Phân ra mô hình, chia việc
  + Tìm hiểu sự liên quan đến Requirements và QA để tiện việc traceability và tracking
  + Quan tâm đặc biệt 2 issues lớn
    - Web interface?
    - Có nên làm Data warehouse?
* Trong buổi họp, phải nhấn mạnh vấn đề traceability, tức là sự liên quan và tracking giữa các giai đoạn để sau này điều chỉnh và update cho hợp lý
* Thiết kế Data base (ERD)