Revision

* Phan tich
  + Yeu cau- usecase- pre-condition, post-condition
  + Phan tich cau truc
    - Phân tích ra các đối tượng của hệ thống
    - Phân tích quan hệ giữa các đối tượng -> ERD
* Thiet ke(draw)
  + Thiết kế database
  + Thiết kế kiến trúc (MVC)
  + Thiết kế giao diện
  + Thiết kế kiểm thử

Backend

* Router
  + Định nghĩa các đường dẫn và phương thức gọi api
  + [địa chỉ ip]:[số cổng]/\*
  + Pre-condition – middleware express js – filter java, c#
  + Postcondition – middleware express js – filter java, c#
  + Đường dẫn nào gọi controller nào.
* Controller
  + Xử lí (body, params, query)
    - Body-parser – 3 loại body
      * App.use(BP.json()) – gửi dự liệu dạng json
      * BP.x-www-url-enconded – gửi dữ liệu dạng mã hóa
      * form-data – dùng để gửi dạng file – dùng thư viện multer
    - /:id => req.params.id
    - /order?page=1&size5 => req.query.page
    - Xử lý page, size => offset, size
* Gọi hàm lấy dữ liệu tương ứng
  + Đánh giá cái giữ liệu trả về và báo lỗi nếu có
* Service/model
  + Khởi tạo câu truy vấn SQL/mongodb/redis
  + Thực thi và nhận về dữ liệu
  + Trả về kết quả
    - Meta data
    - Data
* Middlewares
  + Auth
    - requireLogin – sai return 401
      * req.header.authorization

= `Bearer đaiaidạkdạkd`

* + - requireRole – sai return 403
      * res.status(403).send(“Vượt quyền”)
  + Pagination
    - Phân trang => xử lý cái page với size của query
    - Req.paging.offset = 4
    - Req.paging.size
* ErrorHandle
  + - Không bao h trả lỗi của sever cho thằng client. Try catch mọi thứ, nếu lỗi nằm ngoài 400-500 => trả về 500 (Interal Server Error)
    - tryCatch: bọc controller trong cái này, tryCatch phần thực thi của controller, sau đó nếu lỗi thì next(err)
    - errorHandle = (err, req,res, next) =>{} – phải viết đủ 4 giá trị thì nó mới là errHandle còn không thì nó là middleware hoặc contrlller
* Utils
  + Security
    - Mã hóa token
    - Giải mã token
    - Mã hóa mật khẩu
    - So khợp mật khẩu
  + Database connection
  + Log