NGÔN NGỮ JS

* Sức mạnh :
* JS: Là một ngôn ngữ lập trình khá mềm dẻo , kiểu trả về của nó luôn là kiểu Object .
* Nó có cách khai báo đơn giản , có thể sử dụng biến như là 1 hàm nên có thể truyền hàm vào 1 biến của hàm dễ dàng
* JS OOP : hàm có thể sử dụng như là 1 class , 1 đối tượng . Có thể thông qua Json Object để chỉ ra trực tiếp luôn 1 đối tượng .
* dễ dàng tương tác , điều khiển và tránh bớt việc xử lí từ phía server .
* Nhược điểm : bảo mật kém , không có khả năng giấu mã vì không biết kiểu dữ liệu truyền vào cụ thể là như thế nào ? mặc định là object hết .
* Validation : Kiểm tra tính hợp lệ của dữ liệu :
* Có 3 loại validation:
* phía client : verify data : Nên áp dụng khi ta gửi 1 request lên server vần phải kiểm tra xác nhận dữ liệu đã hợp lệ hay chưa ?Ví dụ : email đã đúng đinh dạng chưa ? Ngày tháng hợp lệ hay chưa? (khi mức độ cần bảo mật là kém nhất )
* phia server : validate business: ví dụ :khi muốn xác nhận tài khoản paypal ,....
* Phía Database : validate database (dữ liệu cần mức độ bảo mật cao nhất)
* Sử dụng Validation trong trường hợp nào thì ta tùy vào mức độ bảo mật của dữ liệu để sử dụng cho phù hợp với từng trường hợp cụ thể .
* Cấu trúc của validation: gồm 3 phần :
* ValidateObject:Từ định nghĩa của các đối tượng, xem thằng nào cần xử lí những nội dung gì ?Thông tin đầu vào : ID, ErrorID , val\_object , Value ;
* Validate method: Chứa các method validate của các đối tượng :
  + lấy Val\_Ọbject (lấy được từ thằng ValidateObject) .
  + Nó chứa các bộ tham số [.....]
  + valucheck= [...] : Vidu : kiểm tra có trong rank của nó hay k ? cái ranhk đó nó nằm trong mảng này .
* Validation :
* Mảng các đối tượng [....] : Nó sẽ lấy được đối tượng nào cần xử lí từ ValidateObject , các methode validate từ ValidateMethod
* RegisterValidate : cho phép ta đăng kí các hàm validate mà ta muốn sử dụng
* Run() ;Sau đó tiến hành chạy duyệt từng thằng trong danh sách các ValidateMethod . Muốn áp dụng cho đối tượng nào,method validate của nó là gì ?

///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

)Điểm mạnh nói chung của JS:  
- cho phép người dùng tương tác, điều khiển html ngay tại client mà không cần gửi request tới server.   
- chỉ có 1 kiểu dữ liệu duy nhất.   
Chính vì vậy:  
- JS OOP cực kì mềm dẻo. Người dùng có thể thao tác với đối tượng bất kì mà không cần quan tâm tới kiểu dữ liệu của nó. Việc tạo ra các phương thức, hàm không cần quan tâm tới kiểu trả về cũng như kiểu của tham số truyền vào. Có thể truyền vào đối tượng kiểu INT (theo ý hiểu là vậy vì trong js chỉ có 1 kiểu object duy nhất), hay kiểu string hoặc có thể là 1 hàm. Ngay cả 1 hàm cũng được coi là 1 đối tượng  
- trong việc thiết kế hướng đối tợng với js, ta có thể tạo các class gần tương tự như các ngôn ngữ hướng đối tuượng khác nhưng cú pháp thì hơi khác chút. chúng bao gồm thuộc tính , phương thức. Tuy nhiên với js, có thể khởi tạo trực tiếp đối tượng mà không cần thông qua phương thức khởi tạo.  
\*)Nhược điểm:  
- Do chỉ có 1 kiểu dữ liệu nên đây cũng chính là nhược điểm. Khó có thể kiểm soát được dữ liệu đưa vào. Dẫn đến vấn đề về bảo mật kém.  
- Do là ngôn ngữ client nên js có thể khiến web trở nên nặng hơn.  
2.  
\*)Validation là chuẩn hóa dữ liệu nhập vào (có thể hiểu là kiểm tra xem dữ liệu nhập vào đã đúng với quy chuẩn đã đưa ra hay chưa).  
\*)Các loại Validation:  
-Verify data: Kiểm tra dữ liệu tại client. Ngay khi nhập vào các input, các dữ liệu sẽ được kiểm tra trước khi gửi lên sever.  
-Validate business: kiểm tra đầu vào dữ liệu tại sever. Lúc này dữ liệu được gửi lên từ client sẽ tiếp tục đuược kiểm tra. đề phòng các dữ liệu không tốt pass qua validation ở client. Do mã js không đuược che giấu. Những ai có kiến thức khủng về js có thể dễ dàng debug thay đổi giá trị.  
-Validate database: kiểm tra dữ liệu tại tầng sâu nhất đó là database. ví dụ như nếu 1 chuỗi có chứa injection thì có khả vuượt qua validate bussiness. Lúc này validate Bussiness sẽ đảm nhận việc kiêm tra lần cuối trước khi thao tác cơ sở dữ liệu.  
Trên đây là 3 các Validate, thì cách thứ 3 là an toàn nhất, sau đó là cách thứ 2. Tuy nhiên, cách thứ 3 thì lại phải đánh đổi lại tốc độ truy xuất CSDL chậm nhất. Cho nên cần phải có sự cân bằng giữa 3 cách.  
3.  
Core Validation có kiến trúc gồm các phần:   
- validate Object: đây là lớp lấy các thông tin đầu vào của đối tượng cần xử lí. Ví dụ như: ID, Value Object  
-validate method: đây là lớp sẽ gom nhóm các method validate đối với đối tượng. lớp này chịu trách nhiệm gọi, thực thi các hàm validate, xử lý các kết quả.  
- Validation: Lớp này là lớp tương tác tực tiếp với ngời sử dụng. Cho phép người dùng đăng kí( chỉ định) ra các hàm validate muốn sử dụng