HƯỚNG DẪN PHÁT TRIỂN

PROJECT **MEDICAL INFORMATION SYSTEM** (MIS)

Version: 1.0 – 2014-07-18

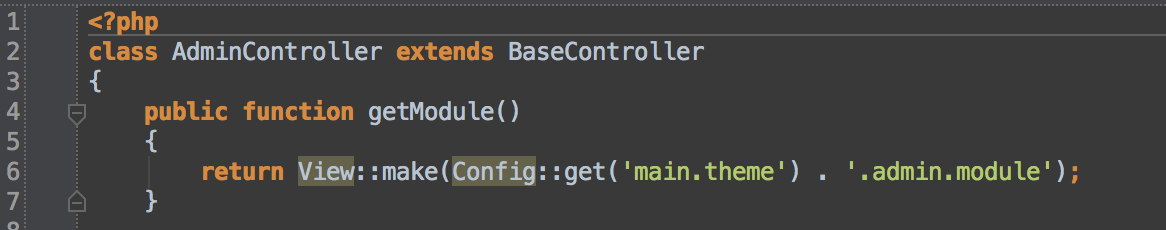
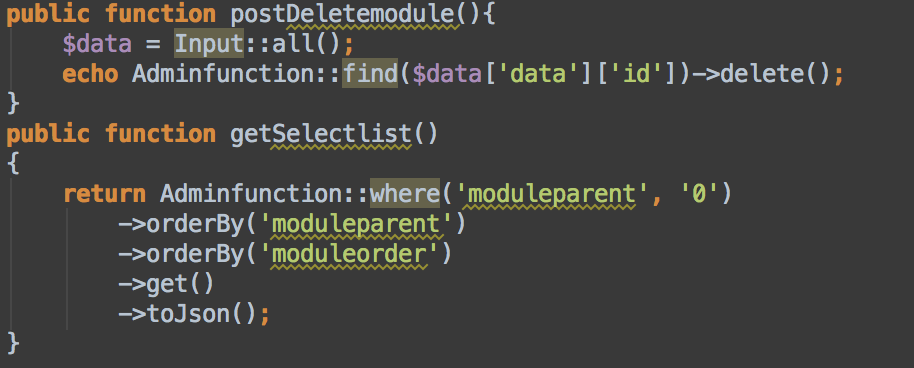
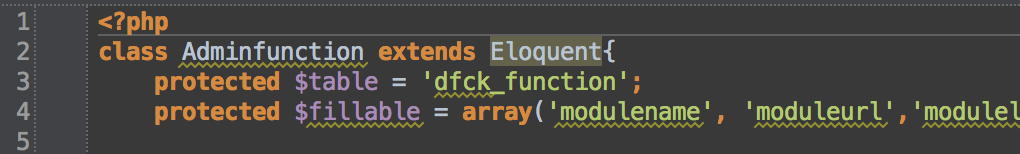
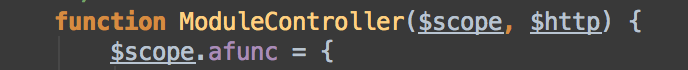
# Công cụ:

* XAMPP: <https://www.apachefriends.org/download.html>
* GIT:
  + Đăng ký nick, thông báo với PM
  + Download <https://windows.github.com>
  + Cài đặt, đăng nhập lần đầu.
  + Mở GIT SHELL, **cd** vào **xampp/htdocs** nhập: git clone <https://github.com/DFCK/his2.git>
* INTELLIJ:
  + <http://www.jetbrains.com/idea/download/> phiên bản Ultimate
  + Vào Files > Setting > Plugin > Browse Repositories > Search PHP > Download và restart.
  + File > New Project, chọn Webmodule và chọn đường dẫn đến thư mục htdocs/his2

# Tổ chức Source code trong folder his2

* **db**: chứa các scipt sql, tạo mới bảng, thêm trường mới hay bất cứ thao tác nào tới csdl đều xuất script và lưu vào đây.
* **www**: source web chính
  + **app**: folder chính để chứa các source code sẽ làm
    - **config**: các cấu hình
      * **app**.php: bật tắt debug, base url…
      * **database**.php: cấu hình mysql trong array mysql
      * tự tạo các file config riêng.
    - **Controllers**: các class controllers extend BaseController
    - **Model**: các class Model extend eloquent
    - **View**: các files template theo BLADE engine.
    - **Route**.php: file chứa các định nghĩa route
    - **Filters**.php: chứac các filter.
    - **Lang:**
      * **Vn:** chứa các file ngôn ngữ tiếng việt
    - **Libaries:** chứa các thư viện của người dùng
  + **src**: folder chứa các style, js, image
  + **uploads**: folder upload
  + Khác: thuộc laravel framework, KHÔNG CHỈNH SỬA
* **hisvhost.txt**: file cấu hình host ảo.
* các tài liệu liên quan

# Phát triển một module

* Tham khảo cách dùng Laravel FW tại <http://laravel.com/docs/>
* Nắm AngularJS tại <http://toddmotto.com/ultimate-guide-to-learning-angular-js-in-one-day/>
* Xem ví dụ module **adminfunction**
* Nếu his2 được clone theo chỉ dẫn sẽ có đường dẫn web là <http://localhost/his2/www/>
* **Giải thích về Mô hình MVC : Model View Controller**
  + Model: quản lý kết nối DB, thực hiện các câu truy vấn, đối tượng hóa các table, gửi dữ liệu cho controller dạng mạng/object hoặc
  + Controller: trung tâm xử lí, nhận yêu cầu gửi về từ front-end, gọi các truy vấn từ model cùng các thuật toán khác, cuối cùng đóng gói dữ liệu và gửi về front-end: dạng html, dạng text, dạng json.
  + View: Sử dụng Blade engine, cùng các dữ liệu từ controller để render các template. Nếu dữ liệu gửi về dạng JSON thì sử dụng angularJS để xử lí.
* **Tạo controller**:
  + Đặt tên theo cấu trúc [Tên Controller]Controller.php
  + Ví dụ **AdminController.php**
  + Tạo class
  + **Tên function**: *[phương thức][Tên function](các tham số)*
  + Ví dụ: **getModule**(): chỉ chấp nhận phương thức get, function tên Module, không có tham số nào.
  + Có thể dùng các phương thức: get, post, any (chấp nhận mọi phương thức)
  + Chú ý tên phải viết hoa chữ đầu, ví dụ ~~getmodult()~~ sẽ bị báo lỗi.
  + Lúc này đường dẫn tới function là **base\_url/admin/module**
  + Nếu đường dẫn chỉ có tên controller thì trong class cần có hàm **getIndex()** để làm trang mặc định
  + Để render template từ controller, sử dụng đoạn
    - **Return View::make(‘path-to-tpl-file’,$data-array)**
    - **$data-array[‘bien’]** thì ở tpl có thể dùng **$bien** để lấy dữ liệu.
    - Như trong hình ví dụ thì file tpl nằm trong ***views/default/admin/module.blade.php***
    - Dấu **“/”** trong đường dẫn có thể dùng dấu **“.”** để thay thế
    - Config::get(“main.theme”) là lấy config ***theme*** từ ***config/main.php/***
  + Để lấy dữ liệu từ model sử dụng Model::function(), xem thêm tại
  + 
  + Để lấy các dữ liệu được gửi về, dùng $data = Input::all();
* **Cấu hình routes.php** cho controller mới tạo
  + 
* **Tạo model** 
  + Đặt tên model gần giống tên table, mỗi model đại diện cho 1 table.
  + Ví dụ Adminfunction.php
  + Tạo class Adminfunction extends Eloquen
  + Khai báo tên bảng
  + Khai báo các cột có thể được thao tác
  + 
  + Để lấy tất cả dữ liệu sử dụng Adminfunction::all();
  + Xem thêm tại <http://laravel.com/docs/eloquent>
* **Views:**
  + Xem thêm về blade engine tại <http://laravel.com/docs/templates>
  + Tóm tắt:
    - Để render biến truyền từ controller, dùng **{{$biến}}**
    - Chạy vòng **foreach**: ***@foreach*** //action ***@endforeach***
    - **If else**: **@if** //action **@elseif** //action **@else** //action **@endif**
  + Chú ý là vì angularJS và balde dùng chung ký tự {{ }} nên với trường hợp nào để
    - render biến từ controller dùng **{{$bien}}**
    - render biến từ angularJS dùng **@{{$bien}}** , kí tự @ sẽ ngăn blade biên dịch.
  + Đặt file template vào folder default (tên theme đang dùng), theo cấu trúc
    - *Views/default/module/templatename.blade.php*
* **AngularJS**
  + Đọc từ link trên và file angularjs.pdf trong trello.
  + Tạo controller trong angularJS có thể tạo nhanh bằng 
  + $scope.biến dùng để truy vấn tất cả các biến trong phạm vi controller
  + $http được truyền vào khi muốn dùng factory $http để gửi ajax, tương tự nếu muốn dùng các factory khác.
  + Trong template muốn in **$scope.biena** khai báo trong controller thì dùng **@{{biena}}**
  + Vòng lặp thì dùng ng-repeat, If thì dùng ng-if v.v...
  + Xem thêm trong file views/default/admin/module.blade.php để hiểu cách dùng.
* **Bootstrap và SmartAdmin**
  + Xem demo tại <http://localhost/his2/www/smartadmin>
  + Có thể mở file smartadmin/views để xem cách sử dụng các chức năng.
* **Ngôn ngữ**:
  + Tạo file tương ứng với module (xem ví dụ), đặt các biến vào folder vn, en...
* **Hiển thị lên hệ thống**:
  + Từ menu chính: Quản lý hệ thống > Danh mục chức năng
  + Nhập thông tin tương ứng của module mình vừa tạo vào.
  + Quy ước: url: module/function, ngôn ngữ module.function, icon thì xem trong trang demo của icon bootstrap.
  + Kéo thả chỉnh thứ tự. Ở đây chỉ dựng cây thư mục 2 cấp cho gọn.
  + Note lại thong tin nhập liệu vào file changelog.txt
  + Xuất script các record mới tạo nếu có thể hoặc export nguyên table dfck\_function.