# Cơ sở lý thuyết

## Tutorials

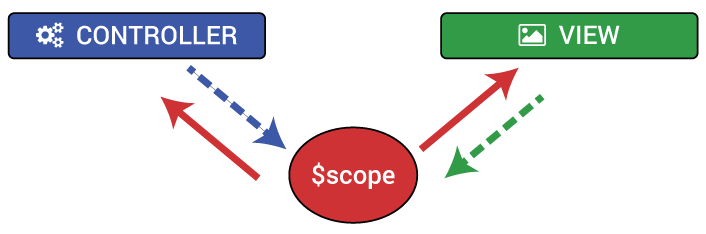
<https://www.w3schools.com/angular/>

## AngularJS

### Scope : là đối tượng có nhiệm vụ giao tiếp giữa controller và view của ứng dụng.

Dùng để binding 2 chiều biến hoặc hàm từ angular controller đến view

Vd:



$scope.jsonEntity = {}; //scope biến   
$scope.EditProduct = EditProduct; //scope hàm

function EditProduct() {  
 apiService.put('/eshopcore\_war/api/json', ***JSON***.stringify($scope.jsonEntity) , function (result) {  
 notificationService.displaySuccess("Sửa bản ghi thành công");  
 $state.go('product-list');  
 }, function () {  
 ***console***.log('Load product api failed.');  
 notificationService.displayError("Thêm mới bản ghi KHÔNG thành công");  
 });  
}

### Tổng quan AngularJS Events

**angularJS** là một thư viện **Javascript** vì vậy tất cả các cách xử lý sự kiện của **Javascript** đều có thể áp dụng trong **AngularJS**. Tuy nhiên **AngularJS** đề xuất cách tiếp cận mới, khi sự kiện xẩy ra một phương thức của **$scope** sẽ được gọi thay vì một hàm của **Javascript**. Để làm được việc này **AngularJS** đưa ra một danh sách các **Directive** đã được xây dựng sẵn:

**vd:**

**vd:** <button ng-click="search()" class="btn btn-primary">Search</button>

* **ng-blur**
* **ng-change**
* **ng-click ---**
* **ng-copy**
* **ng-cut**
* **ng-dblclick**
* **ng-focus**
* **ng-keydown**
* **ng-keypress**
* **ng-keyup**
* **ng-mousedown**
* **ng-mouseenter**
* **ng-mouseleave**
* **ng-mousemove**
* **ng-mouseover**
* **ng-mouseup**
* **ng-paste**

## Angular-ui-router

Dùng để chuyển hướng trang

Các từ khóa

* ui-sref=”tên state” : state// xem bên dưới

ui-sref="product-edit({id:item.id})" truyền param vào state cho trường hợp edit…

Lưu ý khi khai báo state có param

url: "/product-edit/:id", // truyền id vào trang

Chuyển hướng trong controller

$state.go('product-list');

# Cách tạo 1 component trong angularJS vd ở đây là product:

## Các component gồm 3 thành phần chính

Controller:

View: các view con dùng html và css

Module: // điều khiển việc chuyển hướng các trang

Lưu ý: Mỗi conponent có 1 file module duy nhất

Mỗi conponent có thể có nhiều controller phục vụ mục đích khác nhau như lấy listController, editController,createController…

Vd:product-module.js

## Các apiService cung cấp:

Get: lấy dữ liệu

Post: thêm

Put: sửa

Del: delete

Cách dùng: vd: Getlistproduct trong product-list-controller.js

var config = {  
 params: {  
 keyword: $scope.keyWord,  
 action: "getAll", //VỚI GET CONFIG LUÔN TRUYỀN ACTION VÀO ĐỂ HÀM DOGET XỬ LÝ TỪNG CASE  
 }  
}  
  
apiService.get('/eshopcore\_war/api/json', config, function (result) {  
 $scope.productList = result.data;  
 if (result.data.length == 0) {  
 notificationService.displayWarning("Không tìm thấy bản ghi nào");  
 } else {  
  
 notificationService.displaySuccess("Tìm thấy " + result.data.length + " bản ghi");  
 }  
  
  
}, function () {  
 ***console***.log('Load product api failed.');  
 notificationService.displayError("Không lấy được dữ liệu từ server");  
});

## Chi tiết các bước

### Tạo thư mục conponent mới trong thư mục app/components

Vd:product

### Tạo file module với định dạng (conponentname)-module.js, và khai báo các state hiện có trong conponent chưa có state thì chưa cần khai báo các state

Lưu ý: Tên các module phân biệt nhau cú pháp eshop-(tên conponent)

Tên các state phân biệt

vd:

(function () {

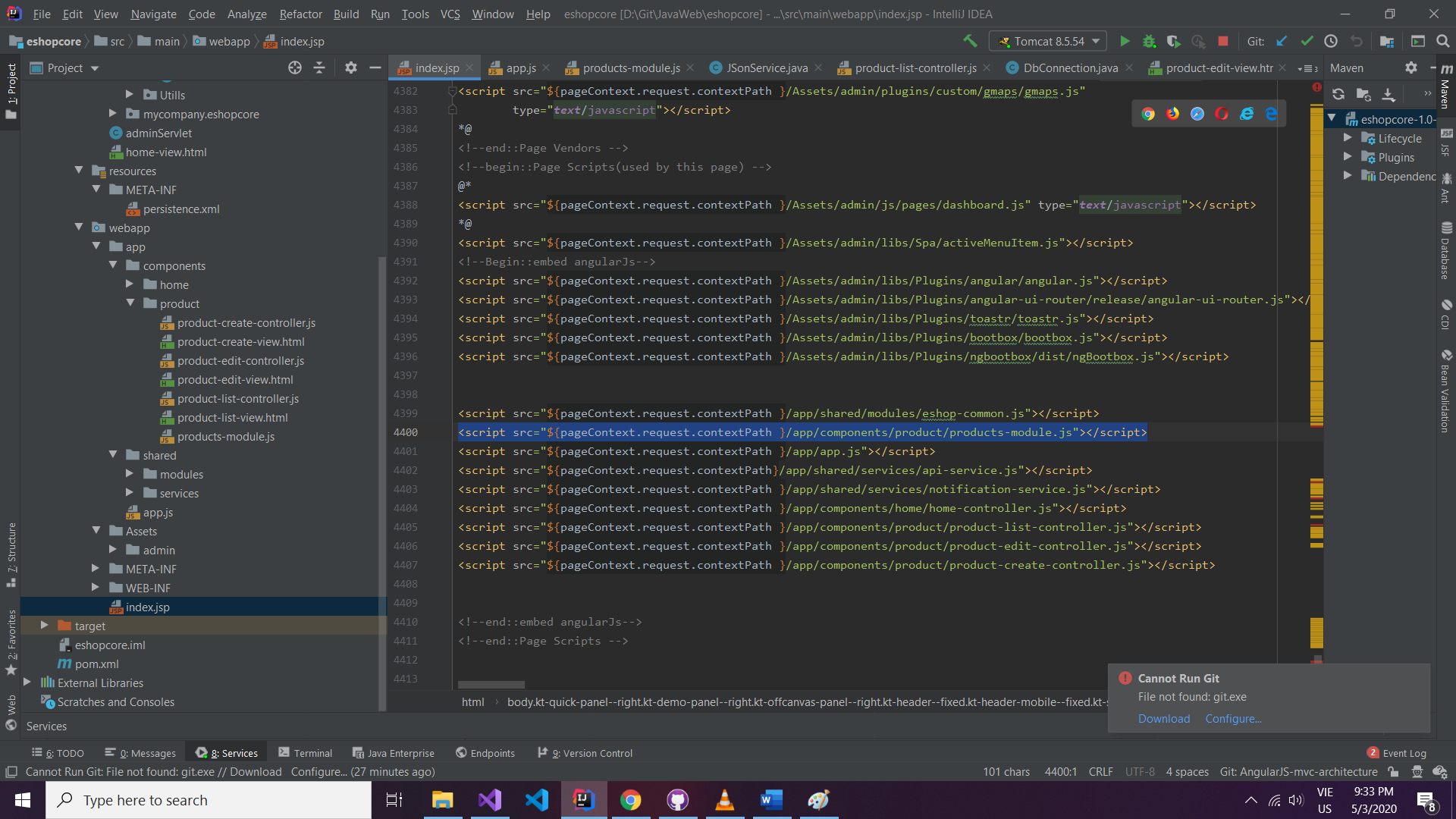
//'eshop-product': Tên module

//'eshop-common': Các module dùng chung  
 angular.*module*('eshop-product', ['eshop-common']).config(config);  
 config.$inject = ['$stateProvider', '$urlRouterProvider'];   
 function config($stateProvider, $urlRouterProvider) {  
 $stateProvider.state('product-edit', {  
 url: "/product-edit/:id", // truyền id vào trang  
 templateUrl: "./app/components/product/product-edit-view.html",  
 controller: "product-edit-controller"   
 }).state('product-list', { // các state có tên phân biệt nhau  
 url: "/product-list", //địa chỉ  
 templateUrl: "./app/components/product/product-list-view.html", //tên view  
 controller: "product-list-controller" ,// tên controller  
 }).state('product-create', {  
 url: "/product-create",  
 templateUrl: "./app/components/product/product-create-view.html",  
 controller: "product-create-controller"  
 });  
  
 }  
  
})();

### Mở file app/app.js khai báo tên module vừa tạo sau dấu [

1. (function () {  
    angular.*module*('eshop', ['eshop-product', 'eshop-common']).config(config);  
     
    config.$inject = ['$stateProvider', '$urlRouterProvider'];  
    function config($stateProvider, $urlRouterProvider) {  
    $stateProvider.state('home', {  
    url: "/home",  
    //missing a fucking dot  
    templateUrl: "./app/components/home/home-view.html",  
    controller: "home-controller"  
    });  
     
    //bắt lỗi 404 điều hướng tại đây  
    $urlRouterProvider.otherwise('/home');  
    }  
     
   })();

### Mờ file index.jsp nhúng đường dẫn module vừa tạo vào mục script sau eshop-common.js



### Trở lại thu mục component vừa tạo ở bước 1 , tạo các controller và view mới

Vd: product-create-controller.js

(function (app) {  
 app.controller('product-create-controller', productCreateController)

//Inject các service cần dùng  
 productCreateController.$inject = ['api-service','$scope', 'notification-service','$state'];  
 // Thứ tự các biến phải giống phần inject  
 function productCreateController(apiService,$scope,notificationService,$state) {  
 $scope.jsonEntity = {};  
 $scope.CreateProduct = CreateProduct;  
 function CreateProduct() {

//gọi api service thêm  
 apiService.post('/eshopcore\_war/api/json', ***JSON***.stringify($scope.jsonEntity) , function (result) {  
 notificationService.displaySuccess("Thêm mới bản ghi thành công");  
 $state.go('product-list'); // chuyển hướng trang về product-list  
 }, function () {  
 ***console***.log('Load product api failed.');  
 notificationService.displayError("Thêm mới bản ghi KHÔNG thành công");  
 });  
 }  
  
 }  
})(angular.*module*('eshop-product'));// khai báo controller thuộc module nào

Vd: product-create-view.html

//" ng-submit: form submit sẽ chạy vào hàm CreateProduct(); lưu ý nhớ gắn scope vào hàm createproduct trong controller để dùng

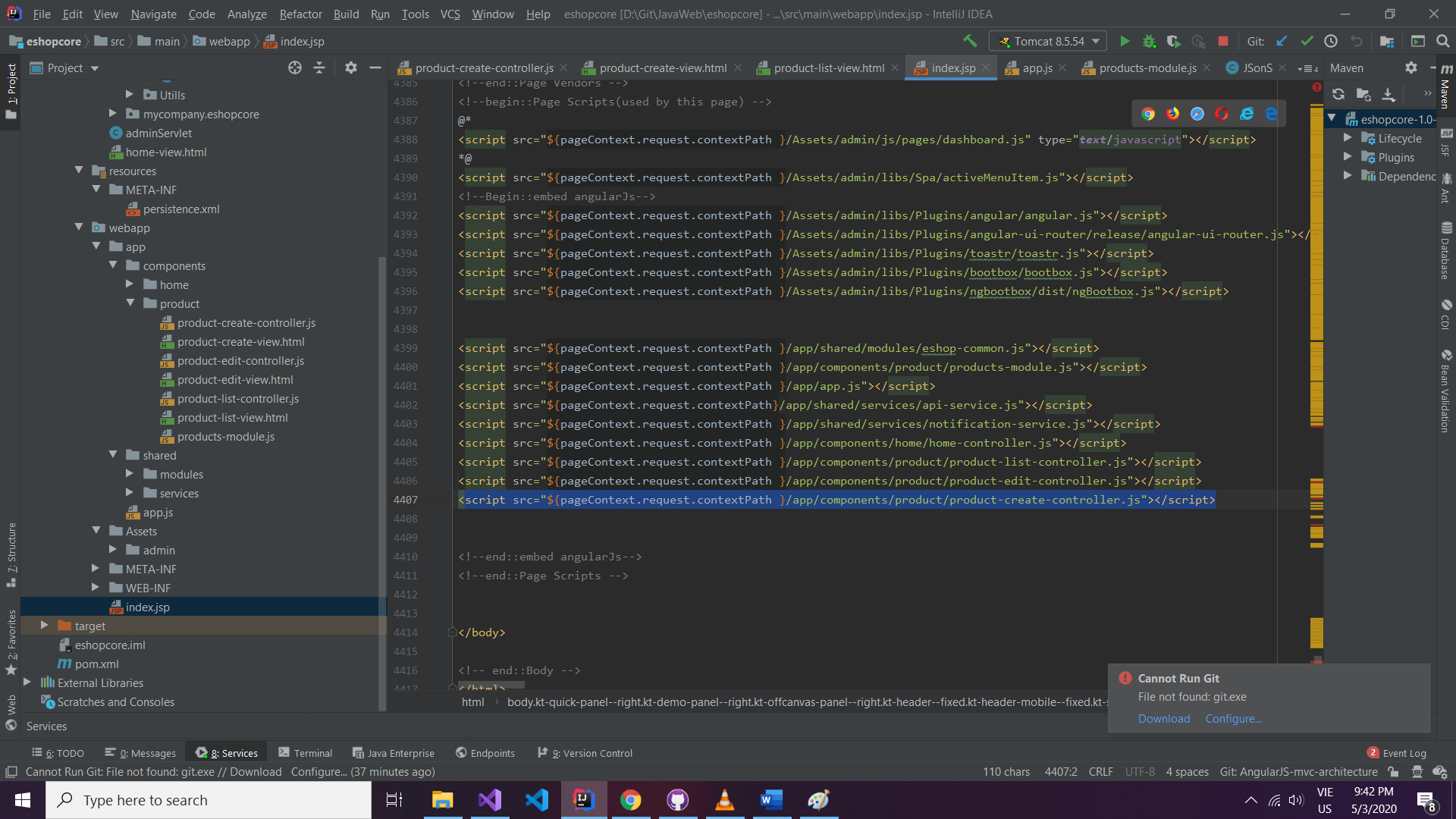
Vd $scope.createproduct= createproduct….. function createproduct(){…}

* Ng-model binding dữ liệu 2 chiều (nhớ gắn scope cho jsonEntity)

<form class="kt-form" name="frmCeateProduct" ng-submit="CreateProduct()" novalidate>  
 <div class="kt-portlet\_\_body">  
 <div class="form-group form-group-last">  
 <div class="alert alert-secondary" role="alert">  
 <div class="alert-icon"><i class="flaticon-warning kt-font-brand"></i></div>  
 <div class="alert-text">  
 The example form below demonstrates common HTML form elements that receive updated styles from Bootstrap with additional classes.  
 </div>  
 </div>  
 </div>  
 <div class="form-group">  
 <label>ID</label>  
 <input ng-model="jsonEntity.id" type="number" class="form-control" aria-describedby="emailHelp" placeholder="Enter email">  
 <span class="form-text text-muted">We'll never share your email with anyone else.</span>  
 </div>  
 <div class="form-group">  
 <label for="exampleInputPassword1">Name</label>  
 <input ng-model="jsonEntity.name" type="text" class="form-control" id="exampleInputPassword1" placeholder="Name">  
 </div>  
 </div>  
 <div class="kt-portlet\_\_foot">  
 <div class="form-group">  
 <button type="submit" class="btn btn-primary">Submit</button>  
 <button type="reset" class="btn btn-secondary">Cancel</button>  
 </div>  
 </div>  
</form>

### Trở lại bước 2 khai báo state mới cho module vừa tạo

### Nhúng file controller mới vào file index.jsp



### Mở file index.jsp tìm từ khóa ui-sref="home"

Copy hoặc sửa thẻ <li class="kt-menu\_\_item… tại đó thêm đường dẫn trùng với state đã tạo trong file module tại (sau này sẽ đọc từ csdl và tạo tự động)

class="kt-menu\_\_item kt-menu\_\_item--active" aria-haspopup="true"><a  
 ui-sref="Tên đường dẫn"….. sửa tên tại class="kt-menu\_\_link-text">TÊN MỚI</span> sau thẻ a.